Alibaba Cloud Object Storage Service

SDK Reference

Issue: 20200121

MORE THAN JUST CLOUD | Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. **You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.**
2. **No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.**
3. **The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.**
4. **This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequent**

ial, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Document conventions

Style	Description	Example
	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click Settings > Network > Set network type.
Bold	Bold formatting is used for buttons, menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands.	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *`Instance_ID`*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`

Style	Description	Example
{} or {a|b}	**This format is used for a required value, where only one item can be selected.**	`switch {active|stand}`

Contents

Legal disclaimer.....I
Document conventions.....I
1 Introduction..... 1
2 Java.....2
2.1 Preface..... 2
2.2 Installation.....3
2.3 Quick start..... 5
2.4 Initialization.....8
2.5 Buckets.....14
2.5.1 Manage a bucket.....14
2.5.2 Bucket tagging.....20
2.5.3 Manage lifecycle rules..... 22
2.5.4 Enable pay-by-requester mode.....25
2.5.5 Hotlink protection.....27
2.5.6 Set logging.....29
2.5.7 Static website hosting.....31
2.5.8 Cross-region replication..... 33
2.5.9 CORS..... 36
2.6 Upload objects.....38
2.6.1 Overview..... 39
2.6.2 Simple upload..... 39
2.6.3 Form upload.....42
2.6.4 Append upload..... 46
2.6.5 Resumable upload..... 47
2.6.6 Multipart upload..... 48
2.6.7 Upload progress bars.....59
2.6.8 Upload callback.....60
2.7 Download objects..... 62
2.7.1 Overview..... 62
2.7.2 Streaming download.....62
2.7.3 Download to your local file.....63
2.7.4 Range download..... 64
2.7.5 Resumable download.....65
2.7.6 Conditional download.....67
2.7.7 Download progress bars..... 69
2.8 Manage objects.....70
2.8.1 Overview..... 70
2.8.2 Determine whether a specified object exists.....71
2.8.3 Manage ACL for an object..... 71
2.8.4 Manage Object Meta..... 73

2.8.5 Convert the storage class of an object....77
2.8.6 List objects....79
2.8.7 Delete objects....89
2.8.8 Copy objects....93
2.8.9 Restore an archive object....98
2.8.10 Manage symbolic links....99
2.9 Versioning....101
2.10 Object tagging....101
2.10.1 Configure object tagging....101
2.10.2 Obtain the tags added to an object....110
2.10.3 Delete the tags added to an object....111
2.10.4 Object tagging and lifecycle rules....112
2.11 Single-connection bandwidth throttling....114
2.12 Server-side encryption....116
2.13 Authorized access....119
2.14 Data security....124
2.15 Image processing....127
2.16 Exception handling....134
2.17 Common errors....137
3 Python....152
3.1 Preface....152
3.2 Installation....153
3.3 Quick start....157
3.4 Initialization....159
3.5 Buckets....161
3.5.1 Bucket....161
3.5.2 Bucket tagging....165
3.5.3 Enable pay-by-requester mode....167
3.5.4 Manage lifecycle rules....169
3.5.5 Hotlink protection....172
3.5.6 Set logging....173
3.5.7 Static website hosting....175
3.5.8 CORS....176
3.6 Upload objects....178
3.6.1 Overview....178
3.6.2 Simple upload....178
3.6.3 Append upload....181
3.6.4 Resumable upload....182
3.6.5 Multipart upload....184
3.6.6 Upload progress bars....185
3.6.7 Upload callback....186
3.7 Download objects....187
3.7.1 Overview....187
3.7.2 Streaming download....187
3.7.3 Download to your local file....188

3.7.4 Range download..... 189
3.7.5 Resumable download.....189
3.7.6 Download progress bars..... 192
3.8 Manage objects..... 192
3.8.1 Overview..... 192
3.8.2 Determine whether a specified object exists.....193
3.8.3 Manage ACL for an object..... 193
3.8.4 Manage Object Meta.....195
3.8.5 Convert the storage class of an object.....197
3.8.6 List objects..... 199
3.8.7 Delete objects..... 200
3.8.8 Copy objects..... 201
3.8.9 Restore an archive object.....203
3.8.10 Manage a symbolic link.....203
3.9 Object tagging.....205
3.9.1 Configure object tagging..... 205
3.9.2 Obtain the tags added to an object..... 212
3.9.3 Delete the tags added to an object.....213
3.9.4 Object tagging and lifecycle rules.....214
3.10 Single-connection bandwidth throttling..... 216
3.11 Data encryption..... 218
3.11.1 Client-side encryption.....218
3.11.2 Server-side encryption.....221
3.12 Authorized access..... 224
3.13 Enable logging in Python SDK..... 226
3.14 Image processing..... 227
3.15 Chinese character and time.....235
3.16 Exception handling..... 237
4 Go..... 241
4.1 Preface..... 241
4.2 Installation..... 242
4.3 Quick start.....243
4.4 Initialization..... 247
4.5 Buckets.....250
4.5.1 Bucket..... 250
4.5.2 Bucket tagging.....257
4.5.3 Manage lifecycle rules.....260
4.5.4 Pay-by-requester mode..... 262
4.5.5 Hotlink protection..... 266
4.5.6 Set logging.....268
4.5.7 Static website hosting..... 270
4.5.8 CORS..... 272
4.6 Upload objects.....274
4.6.1 Overview..... 274
4.6.2 Simple upload.....274

4.6.3 Append upload..... 277
4.6.4 Resumable upload..... 278
4.6.5 Multipart upload..... 280
4.6.6 Progress bars..... 286
4.7 Download objects..... 287
4.7.1 Overview..... 288
4.7.2 Streaming download..... 288
4.7.3 Range download..... 290
4.7.4 Download to your local file..... 291
4.7.5 Resumable download..... 292
4.7.6 Conditional download..... 294
4.7.7 Download a compressed object..... 296
4.7.8 Progress bars..... 296
4.8 Manage objects..... 297
4.8.1 Overview..... 298
4.8.2 Determine whether a specified object exists..... 298
4.8.3 Manage ACL for an object..... 299
4.8.4 Manage Object Meta..... 300
4.8.5 Convert the storage class of an object..... 303
4.8.6 List objects..... 306
4.8.7 Delete objects..... 312
4.8.8 Copy objects..... 314
4.8.9 Restore an archive object..... 319
4.8.10 Manage a symbolic link..... 321
4.9 Object tagging..... 323
4.9.1 Configure object tagging..... 323
4.9.2 Obtain the tags added to an object..... 332
4.9.3 Delete the tags added to an object..... 332
4.9.4 Object tagging and lifecycle rules..... 333
4.10 Single-connection bandwidth throttling..... 335
4.11 Server-side encryption..... 339
4.12 Authorized access..... 342
4.13 Image processing..... 345
4.14 Error handling..... 354
5 C++..... 359
5.1 Overview..... 359
5.2 Installation..... 360
5.3 Quick start..... 363
5.4 Initialization..... 367
5.5 Buckets..... 371
5.5.1 Buckets..... 371
5.5.2 Bucket tagging..... 377
5.5.3 Manage lifecycle rules..... 381
5.5.4 Pay-by-requester mode..... 384
5.5.5 Hotlink protection..... 387

5.5.6 Access logging..... 390
5.5.7 Static website hosting..... 392
5.5.8 CORS..... 395
5.6 Upload objects..... 397
5.6.1 Overview..... 397
5.6.2 Simple upload..... 398
5.6.3 Append upload..... 400
5.6.4 Resumable upload..... 401
5.6.5 Multipart upload..... 404
5.6.6 Interrupt an upload task..... 412
5.6.7 Use a progress bar..... 413
5.6.8 Upload callback..... 414
5.7 Download objects..... 415
5.7.1 Overview..... 415
5.7.2 Download to a local file..... 415
5.7.3 Download to local memory..... 416
5.7.4 Range download..... 417
5.7.5 Resumable download..... 418
5.7.6 Use a progress bar..... 420
5.8 Manage objects..... 421
5.8.1 Overview..... 421
5.8.2 Determine whether the specified object exists..... 421
5.8.3 Manage the ACL of an object..... 422
5.8.4 Manage Object Meta..... 424
5.8.5 Convert the storage class of an object..... 427
5.8.6 List objects..... 429
5.8.7 Delete objects..... 443
5.8.8 Copy an object..... 444
5.8.9 Restore an archive object..... 448
5.8.10 Manage a symbolic link..... 449
5.9 Object tagging..... 451
5.9.1 Configure object tagging..... 451
5.9.2 Obtain the tags added to an object..... 461
5.9.3 Delete the tags added to an object..... 462
5.9.4 Object tagging and lifecycle rules..... 463
5.10 Single-connection bandwidth throttling..... 465
5.11 Server-side encryption..... 467
5.12 Authorized access..... 470
5.13 IMG..... 473
6 PHP..... 481
6.1 Preface..... 481
6.2 Installation..... 482
6.3 Quick start..... 484
6.4 Initialization..... 488
6.5 Buckets..... 491

6.5.1 Bucket..... 491
6.5.2 Manage lifecycle rules.....498
6.5.3 Hotlink protection..... 501
6.5.4 Set logging.....503
6.5.5 Static website hosting..... 506
6.5.6 CORS..... 508
6.5.7 Bind a custom domain name.....510
6.6 Upload objects..... 513
6.6.1 Overview..... 513
6.6.2 Simple upload.....513
6.6.3 Append upload..... 515
6.6.4 Multipart upload.....516
6.6.5 Upload callback.....524
6.7 Download objects.....527
6.7.1 Overview..... 527
6.7.2 Download to your local file..... 527
6.7.3 Download to your local memory..... 528
6.7.4 Range download..... 529
6.7.5 Conditional download.....530
6.8 Manage objects..... 531
6.8.1 Overview..... 531
6.8.2 Determine whether a specified object exists.....532
6.8.3 Manage ACL for an object..... 533
6.8.4 Manage Object Meta.....535
6.8.5 List objects..... 537
6.8.6 Delete objects..... 540
6.8.7 Copy objects..... 541
6.8.8 Restore an archive object.....543
6.8.9 Manage a symbolic link..... 544
6.8.10 Enable MD5 verification.....546
6.9 Authorized access..... 547
6.10 Image processing..... 550
6.11 Exception handling..... 557
7 C..... 560
7.1 Preface..... 560
7.2 Installation..... 561
7.3 Quick start.....565
7.4 Initialization.....572
7.5 Buckets.....576
7.5.1 Bucket..... 576
7.5.2 Manage lifecycle rules.....586
7.5.3 CORS..... 590
7.5.4 Set logging.....595
7.5.5 Hotlink protection..... 599
7.5.6 Static website hosting..... 603

7.6 Upload objects..... 606
7.6.1 Overview..... 606
7.6.2 Simple upload..... 607
7.6.3 Append upload..... 611
7.6.4 Resumable upload..... 615
7.6.5 Multipart upload..... 616
7.6.6 Upload progress bars..... 621
7.6.7 Upload callback..... 623
7.7 Download objects..... 624
7.7.1 Overview..... 624
7.7.2 Download an object to a local file..... 625
7.7.3 Download to your local memory..... 626
7.7.4 Range download..... 628
7.7.5 Resumable download..... 629
7.7.6 Download progress bars..... 631
7.8 Manage objects..... 633
7.8.1 Overview..... 633
7.8.2 Manage ACL for an object..... 633
7.8.3 Manage Object Meta..... 635
7.8.4 List objects..... 637
7.8.5 Delete objects..... 644
7.8.6 Copy objects..... 648
7.8.7 Restore an archive object..... 653
7.8.8 Manage a symbolic link..... 655
7.9 Authorized access..... 657
7.10 Image processing..... 662
7.11 Error handling..... 671
7.12 FAQ..... 674
8 . NET..... 675
8.1 Preface..... 675
8.2 Installation..... 677
8.3 Quick start..... 679
8.4 Initialization..... 682
8.5 Buckets..... 686
8.5.1 Manage a bucket..... 686
8.5.2 Manage lifecycle rules..... 689
8.5.3 Set logging..... 692
8.5.4 CORS..... 694
8.5.5 Static website hosting..... 696
8.5.6 Hotlink protection..... 698
8.6 Upload objects..... 700
8.6.1 Overview..... 700
8.6.2 Simple upload..... 701
8.6.3 Append upload..... 704
8.6.4 Resumable upload..... 705

8.6.5 Multipart upload....707
8.6.6 Upload progress bars....714
8.6.7 Upload callback....715
8.7 Download objects....717
8.7.1 Overview....717
8.7.2 Streaming download....717
8.7.3 Range download....718
8.7.4 Resumable download....719
8.7.5 Download progress bars....720
8.8 Manage objects....721
8.8.1 Overview....722
8.8.2 Determine whether a specified object exists....722
8.8.3 Manage ACL for an object....722
8.8.4 Manage Object Meta....724
8.8.5 List objects....725
8.8.6 Delete objects....731
8.8.7 Copy objects....732
8.8.8 Restore an archive object....735
8.8.9 Manage a symbolic link....737
8.9 Authorized access....738
8.10 Image processing....740
8.11 Exception handling....748
9 Android....751
9.1 Preface....751
9.2 Installation....751
9.3 Initialization....753
9.4 Quick start....758
9.5 Buckets....762
9.5.1 Manage buckets....762
9.5.2 Hotlink protection....764
9.6 Upload objects....766
9.6.1 Overview....766
9.6.2 Simple upload....766
9.6.3 Append upload....768
9.6.4 Multipart upload....770
9.6.5 Resumable upload....774
9.6.6 Progress bar....777
9.6.7 Upload callback....778
9.7 Download objects....779
9.7.1 Overview....779
9.7.2 Streaming download....779
9.7.3 Range download....781
9.7.4 Download an object to a local file....782
9.7.5 Conditional download....783
9.7.6 Resumable download....784

9.8 Manage objects..... 786
9.8.1 Overview..... 786
9.8.2 Determine whether a specified object exists..... 786
9.8.3 Copy an object..... 787
9.8.4 List objects..... 788
9.8.5 Delete an object..... 790
9.8.6 Set the Content-Type of an object..... 791
9.8.7 Obtain the metadata of an object..... 791
9.9 Data security..... 792
9.10 Sign a URL..... 794
9.11 Authorized access..... 794
9.12 Image processing..... 799
9.13 Exception handling..... 802
10 iOS..... 804
10.1 Preface..... 804
10.2 Installation..... 804
10.3 Quick start..... 806
10.4 Initialization..... 810
10.5 Manage a bucket..... 814
10.6 Upload objects..... 816
10.6.1 Overview..... 816
10.6.2 Simple upload..... 817
10.6.3 Multipart upload..... 819
10.6.4 Append upload..... 823
10.6.5 Resumable upload..... 824
10.6.6 Upload callback..... 826
10.7 Download objects..... 827
10.7.1 Overview..... 827
10.7.2 Simple download..... 827
10.7.3 Streaming download..... 828
10.7.4 Range download..... 829
10.7.5 Resumable download..... 829
10.8 Manage objects..... 838
10.8.1 Overview..... 838
10.8.2 Determine whether the specified object exists..... 838
10.8.3 Obtain object metadata..... 839
10.8.4 List objects..... 839
10.8.5 Copy an object..... 840
10.8.6 Delete an object..... 841
10.9 Data security..... 842
10.10 Signed URLs..... 844
10.11 Authorized access..... 844
10.12 IMG..... 849
10.13 Exception handling..... 851

11 Node. js..... 853
11.1 Installation..... 853
11.2 Options..... 855
11.3 Quick start..... 856
11.4 Initialization..... 858
11.5 Buckets..... 859
11.5.1 Manage a bucket..... 859
11.5.2 CORS..... 861
11.5.3 Set ACLs..... 863
11.5.4 Manage lifecycle rules..... 864
11.5.5 Set logging..... 867
11.5.6 Hotlink protection..... 868
11.5.7 Static website hosting..... 870
11.5.8 CNAME..... 871
11.6 Upload objects..... 872
11.6.1 Overview..... 872
11.6.2 Upload a local file..... 873
11.6.3 Upload from local memory..... 873
11.6.4 Streaming upload..... 874
11.6.5 Multipart upload..... 875
11.6.6 Append upload..... 877
11.6.7 Resumable upload..... 878
11.7 Download objects..... 879
11.7.1 Overview..... 879
11.7.2 Download to a local file..... 880
11.7.3 Download to local memory..... 880
11.7.4 Streaming download..... 880
11.7.5 HTTP-based download..... 881
11.8 Manage objects..... 882
11.8.1 Overview..... 882
11.8.2 Determine whether an object exists..... 882
11.8.3 Manage Object Meta..... 882
11.8.4 List all objects in a bucket..... 884
11.8.5 Copy an object..... 885
11.8.6 Delete objects..... 886
11.8.7 Restore an Archive object..... 887
11.9 Authorized access..... 888
11.10 Image processing..... 891
11.11 Exception handling..... 897
11.12 FAQ..... 898
12 Browser.js..... 900
12.1 Installation..... 900
12.2 Quick start..... 904
12.3 Options..... 910

12.4 Upload objects..... 911
12.5 Download objects.....917
12.6 Manage objects..... 917
12.7 CNAME.....921
12.8 Authorized access..... 922
12.9 Set ACLs.....925
12.10 Image processing..... 926
12.11 Browser.....932
12.12 Exception handling.....937
12.13 FAQ..... 937
13 Ruby..... 941
13.1 Installation..... 941
13.2 Quick start.....944
13.3 Buckets..... 947
13.3.1 Manage buckets..... 947
13.3.2 Set ACLs.....949
13.3.3 Manage lifecycle rules.....951
13.3.4 Hotlink protection..... 952
13.3.5 Set logging.....953
13.3.6 CORS.....955
13.3.7 Static website hosting..... 956
13.3.8 CNAME.....957
13.4 Upload objects..... 958
13.5 Download objects.....963
13.6 Manage objects..... 966
13.7 Use STS to access OSS.....970
13.8 Exception handling..... 971
13.9 Application of Rails.....972
14 Media-C.....980
14.1 Preface.....980
14.2 Installation..... 981
14.3 Initialization..... 984
14.4 Client operations.....985
14.5 Server operations.....992
14.6 HLS basic interfaces.....1002
14.7 HLS encapsulated interfaces.....1007
14.8 Scenarios.....1011

1 Introduction

Before using OSS SDK, you must:

- **Learn about and activate the** *Alibaba Cloud OSS service*.
- *Create an AccessKey*.

This topic lists the topics that describe how to install, develop, and troubleshoot OSS SDKs in the following mainstream languages and provide parameters and examples.

Language	Reference
Java	*Java SDK references*
Python	*Python SDK references*
Android	*Android SDK references*
iOS	*iOS SDK references*
.NET	*.NET SDK references*
Node.js	*Node.js SDK references*
Browser.js	*Browser.js SDK references*
PHP	*PHP SDK references*
C	*C SDK references*
Ruby	*Ruby SDK references*
Go	*Go SDK references*
Media-C	*Media-C SDK references*

2 Java

2.1 Preface

This topic is written for the OSS Java SDK V3.5.0.

Compatibility

- **For SDKs V3.×.×:**
  - **API: compatible**
  - **Namespace: compatible**
- **For SDKs V2.×.×:**
  - **API: compatible**
  - **Namespace: compatible**
- **For SDKs V1.0.×:**
  - **API: compatible**
  - **Namespace: incompatible. The Table Store code in version 1.0.x is removed from version 2.0.0. The package names `com.aliyun.openservices.*` and `com.aliyun.openservices.oss.*` are replaced with `com.aliyun.oss.*`.**

SDK source code and API documents

For the source code of the OSS Java SDK, visit *GitHub*. For more information, see *OSS Java SDK API documents*.

Sample code

The OSS Java SDK provides a variety of sample code for your reference or use. The following table describes the sample code provided.

Sample file	Content
GetStartedSample.java	*Quick start*
CreateFolderSample.java	**Create a folder for** *Simple upload*
AppendObjectSample.java	*Append upload*
UploadSample.java	*Resumable upload*
MultipartUploadSample.java	*Multipart upload*

Sample file	Content
GetProgressSample.java	*Upload progress bars*
CallbackSample.java	*Upload callback*
CRCSample.java	**CRC for object upload**
SimpleGetObjectSample.java	*Overview*
DownloadSample.java	*Resumable download*
ObjectMetaSample.java	*Manage Object Meta*
ListObjectsSample.java	*List objects*
UploadPartCopySample.java	*Copy objects*
DeleteObjectsSample.java	*Delete objects*
BucketOperationsSample.java	*Manage ACL for an object*, *Manage lifecycle rules*, **and configure Access logging,** *Hotlink protection*, **and** *CORS* **for a bucket**
ImageSample.java	*Image processing*
PostObjectSample.java	**PostObject in a specified bucket. The implementation is independent of Java SDK.**
ConcurrentGetObjectSample.java	**Download multiple objects at the same time. We recommend that you use** *Resumable download*.
TrafficLimitSample.java	**Configure** *Single-connection bandwidth throttling*
SetRequestPaymentSample.java	**Enable the** *Enable pay-by-requester mode*

2.2 Installation

This topic describes how to install OSS Java SDK.

Preparation

- **Environment requirements**

  You must use Java 1.8 or later versions.

- **Check the Java version**

  Run `java -version` to check Java version.

Download SDK

- *Direct download*
- *Download from GitHub*
- *Download previous versions*

Install SDK

- **Method One: Add dependencies to Maven (recommended)**

  To use OSS Java SDK in Maven, you only need to add the corresponding dependencies to the pom.xml file. This example uses OSS Java SDK 2.8.3. Add the following content to <dependencies>:

  ```
  <dependency>
      <groupId>com.aliyun.oss</groupId>
      <artifactId>aliyun-sdk-oss</artifactId>
      <version>2.8.3</version>
  </dependency>
  ```

- **Method Two: Import jar files into Eclipse**

  Take OSS Java SDK 2.8.3 as an example. To import jar files into Eclipse, follow these steps:

  1. **Download the *Java SDK*.**
  2. **Decompress the Java SDK.**
  3. **Copy the `aliyun-sdk-oss-2.8.3.jar` file and all files from the lib folder to your project.**
  4. **Right-click your project name in Eclipse. Select Properties > Java Build Path > Add JARs.**
  5. **Select all jar files you have copied in step 3 and import them to Libraries.**

- **Method Three: Import jar files into Intellij IDEA**

  Take OSS Java SDK 2.8.3 as an example. To import jar files into Intellij IDEA, follow these steps:

  1. **Download the *Java SDK*.**
  2. **Decompress the Java SDK.**
  3. **Copy the `aliyun-sdk-oss-2.8.3.jar` file and all jar files from the lib folder to your project.**
  4. **Right-click your project name in Intellij IDEA. Select File > Project Structure > Modules > Dependencies > + > JARs or directories.**
  5. **Select all jar files you have copied in step 3 and import them to External Libraries.**

2.3 Quick start

This topic describes how to use OSS Java SDK to perform routine operations such as bucket creation, object uploads, and object downloads.

Sample project

OSS Java SDK provides Maven-based and Ant-based sample projects. You can compile and run the sample projects on local devices or develop your own applications based on the sample projects. For the compilation and running method of the projects, see the README.md in the project directory.

- **Maven sample project:** *aliyun-oss-java-sdk-demo-mvn.zip*.
- **Ant sample project:** *aliyun-oss-java-sdk-demo-ant.zip*.

Create a bucket

A bucket is a global namespace in OSS. It is similar to a data container that stores files. Run the following code to create a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
```

```
// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Create a bucket.
ossClient.createBucket(bucketName);

// Close your OSSClient instance.
ossClient.shutdown();
```

For more information about bucket naming rules, see naming conventions in *Basic concepts*. For more information about how to create a bucket, see *Manage a bucket*.

For more information about endpoints, see *Regions and endpoints*.

Upload objects

Run the following code to upload a file to OSS:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Upload content to the specified bucket and save it as the specified
 object name.
String content = "Hello OSS";
ossClient.putObject(bucketName, objectName, new ByteArrayInputStream(
content.getBytes()));

// Close your OSSClient instance.
ossClient.shutdown();
```

For more information, see *Upload objects*.

Download objects

Run the following code to obtain object content:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
```

```
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Call ossClient.getObject to return an OSSObject instance. The
OSSObject instance contains the object content and Object Meta.
OSSObject ossObject = ossClient.getObject(bucketName, objectName);
// Call ossObject.getObjectContent to obtain object InputStream. Read
the InputStream to obtain object content.
InputStream content = ossObject.getObjectContent();
if (content ! = null) {
    BufferedReader reader = new BufferedReader(new InputStreamReader(
content));
    while (true) {
        String line = reader.readLine();
        if (line == null) break;
        System.out.println("\n" + line);
    }
    // If you do not close the reader after the data is read,
connection leaks may occur. Consequently, no available connections are
 left and an exception occurs.
    content.close();
}

// Close your OSSClient instance.
ossClient.shutdown();
```

For more information, see *Download objects*.

List objects

Use the following code to list objects in a bucket. You can list a maximum of 100 objects by default.

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Use ossClient.listObjects to return an ObjectListing instance. The
ObjectListing instance contains the response from listObject.
ObjectListing objectListing = ossClient.listObjects(bucketName);
// Use objectListing.getObjectSummaries to obtain the descriptions of
all objects.
```

```
for (OSSObjectSummary objectSummary : objectListing.getObjectSummaries
()) {
    System.out.println(" - " + objectSummary.getKey() + "  " +
            "(size = " + objectSummary.getSize() + ")");
}

// Close your OSSClient instance.
ossClient.shutdown();
```

For more information, see *List objects* **in** *Manage objects*.

Delete objects

For the complete code of deleting objects, see *GitHub*.

Use the following code to delete a specified object:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Delete objects.
ossClient.deleteObject(bucketName, objectName);

// Close your OSSClient instance.
ossClient.shutdown();
```

For more information, see *Delete objects* **in** *Manage objects*.

2.4 Initialization

OSSClient serves as the Java client to manage OSS resources such as buckets and objects. To use the Java SDK to initiate a request, you must initialize an OSSClient instance and modify the default configuration items of ClientConfiguration as needed.

Create an OSSClient instance

To create an OSSClient instance, you must specify an endpoint. For more information about endpoints, see *Regions and endpoints* **and** *Bind a custom domain name*.

- **Create an OSSClient instance based on an OSS endpoint**

  The following code provides an example on how to create an OSSClient instance based on an OSS endpoint:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

  You can use one or more OSSClient instances for your project. In other words, you can use multiple OSSClient instances simultaneously.

- **Create an OSSClient instance based on a custom domain name**

  The following code provides an example on how to create an OSSClient instance based on a custom domain name:

```
String endpoint = "<yourEndpoint>";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create a ClientConfiguration instance and modify the default
parameters based on your requirements.
ClientBuilderConfiguration conf = new ClientBuilderConfiguration();
// Enable CNAME. If CNAME is enabled, a custom domain name can be
bound to a bucket.
conf.setSupportCname(true);

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, conf);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

Note:

ossClient.listBuckets is not available when a custom domain name is used.

- **Create an OSSClient instance based on Apsara Stack or a private region**

  The following code provides an example on how to create an OSSClient instance based on Apsara Stack or a private region:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create a ClientConfiguration instance and modify the default
parameters based on your requirements.
ClientBuilderConfiguration conf = new ClientBuilderConfiguration();
// Disable CNAME.
conf.setSupportCname(false);

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, conf);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Create an OSSClient instance based on an IP address**

  The following code provides an example on how to create an OSSClient instance based on an IP address:

```
In some special cases (such as a private region), the IP address is
 used as the endpoint. In such cases, specify the actual IP address
based on your requirements.
String endpoint = "http://10.10.10.10";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create a ClientConfiguration instance.
ClientBuilderConfiguration conf = new ClientBuilderConfiguration();
// By default, access from a second-level domain is disabled. Enable
 OSS access from a second-level domain. You must configure this
```

```
 parameter for OSS Java SDK versions 2.1.2 or earlier, because
versions later than 2.1.2 automatically detect IP addresses.
conf.setSLDEnabled(true);

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, conf);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Create an OSSClient instance based on STS**

  The following code provides an example on how to create an OSSClient instance based on Security Token Service (STS):

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String securityToken = "<yourSecurityToken>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, securityToken);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

For more information, see *What are RAM and STS* **and** *Authorized access*.

Configure an OSSClient instance

ClientConfiguration is a configuration class of OSSClient. You can use ClientConfiguration to configure parameters such as host proxies, connection timeouts, and the maximum number of connections. The following table describes the parameters.

Parameter	Description	Configuration method
MaxConnections	Specifies the maximum number of HTTP connections that are allowed. Default value: 1024.	ClientConfiguration.setMaxConnections

Parameter	Description	Configuration method
SocketTimeout	Specifies the timeout period for data transmission at the socket layer. Unit: milliseconds. Default value: 50000.	ClientConfiguration.setSocketTimeout
ConnectionTimeout	Specifies the timeout period for establishing a connection. Unit: milliseconds. Default value: 50000.	ClientConfiguration.setConnectionTimeout
ConnectionRequestTimeout	Specifies the timeout period for obtaining a connection from the connection pool. Unit: milliseconds. By default, this parameter is not configured.	ClientConfiguration.setConnectionRequestTimeout
IdleConnectionTime	Specifies the timeout period for idle connections. The connection is closed when the timeout period expires. Unit: milliseconds. Default value: 60000.	ClientConfiguration.setIdleConnectionTime
MaxErrorRetry	Specifies the maximum number of allowed retry attempts in the case of a request error. Default value: 3.	ClientConfiguration.setMaxErrorRetry
SupportCname	Specifies whether CNAME can be used as an endpoint. By default, CNAME can be used as an endpoint.	ClientConfiguration.setSupportCname
SLDEnabled	Specifies whether access from second-level domains is enabled. By default, access from second-level domains is disabled.	ClientConfiguration.setSLDEnabled

Parameter	Description	Configuration method
Protocol	Specifies the protocol used to connect to OSS. Valid values: HTTP and HTTPS. Default value: HTTP.	ClientConfiguration.setProtocol
UserAgent	Specifies the user agent (the User-Agent field in the HTTP header). Default value: aliyun-sdk-java.	ClientConfiguration.setUserAgent
ProxyHost	Specifies the IP address to access the proxy host.	ClientConfiguration.setProxyHost
ProxyPort	Specifies the port for the proxy host.	ClientConfiguration.setProxyPort
ProxyUsername	Specifies the username verified by the proxy host.	ClientConfiguration.setProxyUsername
ProxyPassword	Specifies the password verified by the proxy host.	ClientConfiguration.setProxyPassword

The following code provides an example on how to configure parameters for an OSSClient instance with ClientConfiguration:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create a ClientConfiguration instance. ClientConfiguration is a
configuration class of OSSClient. You can use ClientConfiguration to
 configure parameters such as host proxies, connection timeouts, and
the maximum number of connections.
ClientBuilderConfiguration conf = new ClientBuilderConfiguration();

// Configure the maximum number of HTTP connections allowed by an
OSSClient instance. Default value: 1024.
conf.setMaxConnections(200);
// Configure the timeout period for data transmission at the socket
layer. Unit: milliseconds. Default value: 50000.
conf.setSocketTimeout(10000);
// Configure the timeout period for establishing a connection. Unit:
milliseconds. Default value: 50000.
conf.setConnectionTimeout(10000);
```

```
// Configure the timeout period for obtaining a connection from the
connections pool. Unit: milliseconds. By default, this parameter is
not configured.
conf.setConnectionRequestTimeout(1000);
// Configure the timeout period for idle connections. The connection
is closed when the timeout period expires. Default value: 60000.
conf.setIdleConnectionTime(10000);
// Configure the maximum number of allowed retry attempts in the case
of a request error. Default value: 3.
conf.setMaxErrorRetry(5);
// Configure whether CNAME can be used as an endpoint. By default,
CNAME can be used as an endpoint.
conf.setSupportCname(true);
// Configure whether access from second-level domains is enabled. By
default, access from second-level domains is disabled.
conf.setSLDEnabled(true);
// Configure the protocol used to connect to OSS. Valid values: HTTP
and HTTPS. Default value: HTTP.
conf.setProtocol(Protocol.HTTP);
// Configure the user agent (the User-Agent field in the HTTP header
). Default value: aliyun-sdk-java.
conf.setUserAgent("aliyun-sdk-java");
// Configure the port for the proxy host.
conf.setProxyHost("<yourProxyHost>");
// Configure the username verified by the proxy host.
conf.setProxyUsername("<yourProxyUserName>");
// Configure the password verified by the proxy host.
conf.setProxyPassword("<yourProxyPassword>");

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, conf);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

2.5 Buckets

2.5.1 Manage a bucket

A bucket serves as a container that stores objects. Objects belong to a bucket.

Create a bucket

Run the following code to create a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
```

```
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Create a bucket.
String bucketName = "<yourBucketName>";
// The default storage class and ACL of the created bucket are set to
standard and private read respectively.
ossClient.createBucket(bucketName);

// Close your OSSClient.
ossClient.shutdown();
```

For more information about bucket naming rules, see *naming conventions* **in** *Basic concepts*.

You can specify the ACL and the storage class when you create a bucket, as shown in *Bucket-level permissions* **and** *Storage class*. **To create Infrequent Access or archive buckets, use Java SDK versions 2.6.0 or later. The sample code is as follows:**

```
CreateBucketRequest createBucketRequest = new CreateBucketRequest(
bucketName);
// Set the ACL to public read for a bucket. The default value is
Private.
createBucketRequest.setCannedACL(CannedAccessControlList.PublicRead);
// Set the storage class to Infrequent Access for a bucket. The
default value is Standard.
createBucketRequest.setStorageClass(StorageClass.IA);
ossClient.createBucket(createBucketRequest);
```

List buckets

Buckets are listed alphabetically. You can list all buckets or buckets that meet the specified conditions. To list Infrequent Access or archive buckets, use Java SDK versions 2.6.0 or later.

- **List all buckets**

  Run the following code to list all buckets:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId,
accessKeySecret);

// List buckets.
List<Bucket> buckets = ossClient.listBuckets();
```

```
for (Bucket bucket : buckets) {
    System.out.println(" - " + bucket.getName());
}

// Close your OSSClient.
ossClient.shutdown();
```

- **List buckets with a specified prefix**

  Run the following code to list buckets with a specified prefix:

```
ListBucketsRequest listBucketsRequest = new ListBucketsRequest();
// List buckets by a specified prefix. If no prefix is specified,
list all buckets.
listBucketsRequest.setPrefix("<yourBucketPrefix>");
BucketList bucketList = ossClient.listBuckets(listBucketsRequest);
for (Bucket bucket : bucketList.getBucketList()) {
    System.out.println(" - " + bucket.getName());
}
```

- **List buckets by marker**

  The value of the marker parameter indicates the name of a bucket. Run the following code to list buckets after the specified bucket (the value of marker):

```
ListBucketsRequest listBucketsRequest = new ListBucketsRequest();
// List buckets after the specified marker. If the marker is not
specified, list all buckets.
listBucketsRequest.setMarker("<yourBucketMarker>");
BucketList bucketList = ossClient.listBuckets(listBucketsRequest);
for (Bucket bucket : bucketList.getBucketList()) {
    System.out.println(" - " + bucket.getName());
}
```

- **List a specified number of buckets**

  Run the following code to list a specified number of buckets:

```
ListBucketsRequest listBucketsRequest = new ListBucketsRequest();
// Set the number of buckets that can be listed to 500. The default
value is 100. The maximum value is 1,000.
listBucketsRequest.setMaxKeys(500);
BucketList bucketList = ossClient.listBuckets(listBucketsRequest);
for (Bucket bucket : bucketList.getBucketList()) {
    System.out.println(" - " + bucket.getName());
}
```

Determine whether a bucket exists

Run the following code to determine whether a specified bucket exists:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
```

```
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

boolean exists = ossClient.doesBucketExist("<yourBucketName>");

// Close your OSSClient.
ossClient.shutdown();
```

Configure an ACL for a bucket

The ACL of a bucket includes the following permissions.

Permission	Description	Value in Java SDK
Private	The bucket owner and the authorized users can read and write objects in the bucket. Other users cannot perform any operation on the objects.	CannedAccessControlList.Private
Public read	The bucket owner and the authorized users can read and write objects in the bucket. Other users can only read the objects in the bucket. Authorize this permission with caution.	CannedAccessControlList.PublicRead
Public read-write	All users can read and write objects in the bucket. Authorize this permission with caution.	CannedAccessControlList.PublicReadWrite

Run the following code to configure the ACL for a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);
```

```
// Set the ACL for a bucket to private read and write.
ossClient.setBucketAcl("<yourBucketName>", CannedAccessControlList.
Private);

// Close your OSSClient.
ossClient.shutdown();
```

Obtain the ACL for a bucket

Run the following code to obtain the ACL for a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);
// Obtain the access ACL for the bucket
AccessControlList acl = ossClient.getBucketAcl("<yourBucketName>");
System.out.println(acl.toString());

// Close your OSSClient.
ossClient.shutdown();
```

Obtain the region of a bucket

Run the following code to obtain the region for a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

String location = ossClient.getBucketLocation("<yourBucketName>");
System.out.println(location);

// Close your OSSClient.
ossClient.shutdown();
```

For more information about regions, see *Region* in *Basic concepts*.

Obtain bucket information

Run the following code to obtain bucket information:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We strongly recommend that you create a RAM account and use it for
API access and daily O&M. Log on to https://ram.console.aliyun.com to
create a RAM account.
String accessKeyId = "<accessKeyId>";
String accessKeySecret = "<accessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

Bucket information includes regions (region or location), creation
dates (CreationDate), owners (Owner), and permissions (Grants).
BucketInfo info = ossClient.getBucketInfo("<yourBucketName>");
// Obtain region information.
info.getBucket().getLocation();
// Obtain the creation date.
info.getBucket().getCreationDate();
// Obtain owner information.
info.getBucket().getOwner();
// Obtain permission information.
info.getGrants();

// Close your OSSClient.
ossClient.shutdown();
```

Delete a bucket

Before a bucket is deleted, ensure that all objects in the bucket and fragments that are generated from multipart upload are deleted.

Note:

To delete the fragments that are generated from multipart upload, use *Bucket.ListMultipartUploads* to list all fragments, and then use *Bucket.AbortMultipartUpload* to delete the fragments.

Run the following code to delete a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
```

```
// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

/// Delete the bucket.
ossClient.deleteBucket("<yourBucketName>");

// Close your OSSClient.
ossClient.shutdown();
```

2.5.2 Bucket tagging

By using the bucket tagging function, you can classify your buckets to manage them. For example, you can specify that only buckets with specified tags are returned in ListBucket operations.

- **Only the bucket owner and authorized RAM users can add tags to a bucket. Otherwise, the 403 Forbidden error is returned with the error code: AccessDeni ed.**
- **You can add a maximum of 20 tags (key-value pairs) to a bucket.**
- **The maximum size of a key is 64 bytes. The key of a tag cannot be null and cannot be prefixed with `http://`, `https://`, or `Aliyun`.**
- **The maximum size of a tag value is 128 bytes. The value of a tag can be null.**
- **The key and value of a tag must be UTF-8 encoded.**
- **If you use PutBucketTagging to add tags to a bucket, the original tags added to the bucket are completely overwritten.**

Add tags to a bucket

You can run the following code to add tags to a bucket:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Adds tags to the bucket.
SetBucketTaggingRequest request = new SetBucketTaggingRequest("<
yourBucketName>");
request.setTag("<yourTagkey1>", "<yourTagValue1>");
request.setTag("<yourTagkey2>", "<yourTagValue2>");
```

```
ossClient.setBucketTagging(request);

// Closes the OSSClient.
ossClient.shutdown();
```

Obtain the tags added to a bucket

You can run the following code to obtain the tags to a bucket:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtains the tags added to the bucket.
TagSet tagSet = ossClient.getBucketTagging(new GenericRequest("<
yourBucketName>"));
Map<String, String> tags = tagSet.getAllTags();

// Closes the OSSClient instance.
ossClient.shutdown();
```

List buckets with a specified tag

You can run the following code to list buckets with a specified tag:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Lists buckets with the specified tag.
ListBucketsRequest listBucketsRequest = new ListBucketsRequest();
listBucketsRequest.setTag("<yourTagKey>", "<yourTagValue>");
BucketList bucketList = ossClient.listBuckets(listBucketsRequest);
for (Bucket o : bucketList.getBucketList()) {
  System.out.println("list result bucket: " + o.getName());
}

// Closes the OSSClient instance.
ossClient.shutdown();
```

Delete the tags added to a bucket

You can run the following code to delete the tags added to a bucket:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Deletes the tags added to the bucket.
ossClient.deleteBucketTagging(new GenericRequest("<yourBucketName>"));

// Closes the OSSClient instance.
ossClient.shutdown();
```

2.5.3 Manage lifecycle rules

This topic describes how to manage lifecycle rules.

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**
- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**

- **Status: This lifecyle rule takes effect or not.**

The parts uploaded by uploadPart method also support lifecycle rules. The last modification time of an object is the time the multipart upload event is initiated.

For more information about lifecycle management, see *Manage object lifecycle*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

SetBucketLifecycleRequest request = new SetBucketLifecycleRequest(
bucketName);

// Configure the rule ID and object name prefix.
String ruleId0 = "rule0";
String matchPrefix0 = "A0/";
String ruleId1 = "rule1";
String matchPrefix1 = "A1/";
String ruleId2 = "rule2";
String matchPrefix2 = "A2/";
String ruleId3 = "rule3";
String matchPrefix3 = "A3/";

// Set the expiration duration to 3 days after last modified date.
request.AddLifecycleRule(new LifecycleRule(ruleId0, matchPrefix0,
RuleStatus.Enabled, 3));

// Configure expiration date to delete objects that are created before
 the date.
LifecycleRule rule = new LifecycleRule(ruleId1, matchPrefix1,
RuleStatus.Enabled);
rule.setCreatedBeforeDate(DateUtil.parseISO8601Date("2022-10-12T00:00:
00.000Z"));
request.AddLifecycleRule(rule);

// Set the expiration duration to 3 days for parts in an object.
rule = new LifecycleRule(ruleId2, matchPrefix2, RuleStatus.Enabled);
LifecycleRule.AbortMultipartUpload abortMultipartUpload = new
LifecycleRule.AbortMultipartUpload();
abortMultipartUpload.setExpirationDays(3);
rule.setAbortMultipartUpload(abortMultipartUpload);
request.AddLifecycleRule(rule);
```

```
// Configure expiration date to delete parts that are created before
the date.
rule = new LifecycleRule(ruleId3, matchPrefix3, RuleStatus.Enabled);
abortMultipartUpload = new LifecycleRule.AbortMultipartUpload();
abortMultipartUpload.setCreatedBeforeDate(DateUtil.parseISO8601Date("
2022-10-12T00:00:00.000Z"));
rule.setAbortMultipartUpload(abortMultipartUpload);
request.AddLifecycleRule(rule);

ossClient.setBucketLifecycle(request);

// Close your OSSClient.
ossClient.shutdown();
```

View lifecycle rules

Run the following code to view lifecycle rules:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

List<LifecycleRule> rules = ossClient.getBucketLifecycle(bucketName);
for (LifecycleRule rule : rules) {
    System.out.println(rule.getId());
    System.out.println(rule.getPrefix());
    System.out.println(rule.getExpirationDays());
    System.out.println(rule.getCreatedBeforeDate());
    if(rule.hasAbortMultipartUpload()) {
        System.out.println(rule.getAbortMultipartUpload().getExpirat
ionDays());
        System.out.println(rule.getAbortMultipartUpload().getCreated
BeforeDate());
    }
}

// Close your OSSClient.
ossClient.shutdown();
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

ossClient.deleteBucketLifecycle(bucketName);

// Close your OSSClient.
ossClient.shutdown();
```

2.5.4 Enable pay-by-requester mode

After pay-by-requester mode is enabled for a bucket in Alibaba Cloud OSS, the requester instead of the bucket owner pays the cost of requests and traffic. The bucket owner always pays the cost of storing data. You can enable this feature to share data without having to pay for all requests and traffic by yourself.

Request methods

- **Access from anonymous users is not allowed**

  If you enable pay-by-requester mode for a bucket, anonymous users are not allowed to access the bucket. Requesters must provide authentication informatio n so that OSS can identify and charge them, but not the bucket owner, for requests and traffic.

  If requesters assume the RAM role of an Alibaba Cloud account to request data, OSS charges this Alibaba Cloud account for such requests and traffic.
- **Requesters must include the x-oss-request-payer request header**

  If you enable pay-by-requester mode for a bucket, requesters must include the x-oss-request-payer:requester request header in a PUT, POST, GET, or HEAD request. This field indicates that requesters understand that their requests and data downloads incur fees. Otherwise, requests cannot pass authentication.

  Bucket owners do not need to include the x-oss-request-payer request header when accessing their own buckets. In this case, bucket owners are requesters who pay for such requests and traffic.

For more information about pay-by-requester mode, see *#unique_48* in the OSS Developer Guide.

Enable pay-by-requester mode

The following code provides an example of how to enable pay-by-requester mode:

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Enable requester payment mode.
Payer payer = Payer.Requester;
ossClient.setBucketRequestPayment(bucketName, payer);

// Close your OSSClient.
ossClient.shutdown();
```

Obtain pay-by-requester configurations

The following code provides an example of how to obtain pay-by-requester configurations:

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtain pay-by-requester configurations.
GetBucketRequestPaymentResult result = ossClient.getBucketRequestPaym
ent(bucketName);
System.out.println("Payer:" + result.getPayer());

// Close your OSSClient.
```

```
ossClient.shutdown();
```

Specify a third party to pay for access to objects

The following code provides an example of how to specify a third party to pay for access to objects when you use PutObject, GetObject, and DeleteObject:

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
Payer payer = Payer.Requester;

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Specify the payer when you call the PutObject operation.
String content = "hello";
PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName,
objectName, new ByteArrayInputStream(content.getBytes()));
putObjectRequest.setRequestPayer(payer);
ossClient.putObject(putObjectRequest);

// Specify the payer when you call the GetObject operation.
GetObjectRequest getObjectRequest = new GetObjectRequest(bucketName,
objectName);
getObjectRequest.setRequestPayer(payer);
OSSObject ossObject = ossClient.getObject(getObjectRequest);
ossObject.close();

// Specify the payer when you call the DeleteObject operation.
GenericRequest genericRequest = new GenericRequest(bucketName,
objectName);
genericRequest.setRequestPayer(payer);
ossClient.deleteObject(genericRequest);

// Close your OSSClient.
ossClient.shutdown();
```

2.5.5 Hotlink protection

This topic describes how to use hotlink protection.

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**

- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49*.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

List<String> refererList = new ArrayList<String>();
// Add the referer field. The referer field allows question marks (?)
and asterisks (*) for wildcard use.
refererList.add("http://www.aliyun.com");
refererList.add("http://www. *.com");
refererList.add("http://www.?.aliyuncs.com");
// Configure the referer list for a bucket. Set the parameter to true
, which allows the referer field to be empty.
BucketReferer br = new BucketReferer(true, refererList);
ossClient.setBucketReferer(bucketName, br);

// Close your OSSClient.
ossClient.shutdown();
```

Obtain a referer whiltelist

Run the following code to obtain a referer whiltelist:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
```

```
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtain the referer list for a bucket.
BucketReferer br = ossClient.getBucketReferer(bucketName);
List<String> refererList = br.getRefererList();
for (String referer : refererList) {
    System.out.println(referer);
}

// Close your OSSClient.
ossClient.shutdown();
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// You cannot clear a referer whitelist directly. To clear a referer
 whitelist, you need to create the rule that allows an empty referer
field and replace the original rule with the new rule.
BucketReferer br = new BucketReferer();
ossClient.setBucketReferer(bucketName, br);

// Close your OSSClient.
ossClient.shutdown();
```

2.5.6 Set logging

You can enable access logs to record bucket access to log files, which are stored in a specified bucket.

The log file format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

For more information about access log files, see *Set access logging*.

Enable access logging

Run the following code to enable bucket access logging:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

SetBucketLoggingRequest request = new SetBucketLoggingRequest("<
yourSourceBucketName>");
// Configure the bucket that stores log files.
request.setTargetBucket("<yourTargetBucketName>");
// Configure the log file storage directory.
request.setTargetPrefix("<yourTargetPrefix>");
ossClient.setBucketLogging(request);

// Close your OSSClient.
ossClient.shutdown();
```

View access logging configurations

Run the following code to view the access logging configurations for a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

BucketLoggingResult result = ossClient.getBucketLogging("<yourSource
BucketName>");
System.out.println(result.getTargetBucket());
System.out.println(result.getTargetPrefix());

// Close your OSSClient.
ossClient.shutdown();
```

Disable access logging

Run the following code to disable access logging for a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

SetBucketLoggingRequest request = new SetBucketLoggingRequest("<
yourSourceBucketName>");
request.setTargetBucket(null);
request.setTargetPrefix(null);
ossClient.setBucketLogging(request);

// Close your OSSClient.
ossClient.shutdown();
```

2.5.7 Static website hosting

You can set your bucket configuration to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

For more information about static website hosting, see *Static website hosting*.

Configure static website hosting

Run the following code to configure static website hosting:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
```

```
SetBucketWebsiteRequest request = new SetBucketWebsiteRequest("<
yourBucketName>");
request.setIndexDocument("index.html");
request.setErrorDocument("error.html");
ossClient.setBucketWebsite(request);

// Close your OSSClient.
ossClient.shutdown();
```

View static website hosting configurations

Run the following code to view static website hosting configurations:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

BucketWebsiteResult result = ossClient.getBucketWebsite("<yourBucket
Name>");
System.out.println(result.getIndexDocument());
System.out.println(result.getErrorDocument());

// Close your OSSClient.
ossClient.shutdown();
```

Delete static website hosting configurations

Run the following code to delete static website hosting configurations:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

ossClient.deleteBucketWebsite("<yourBucketName>");

// Close your OSSClient.
ossClient.shutdown();
```

2.5.8 Cross-region replication

Cross-region replication is used to automatically and asynchronously copy objects across buckets in different regions. Any changes (creation, replacement, and deletion) to objects in the source bucket will be synchronized to the target bucket. This function is used to meet the requirements of cross-region disaster recovery and data replication.

For more information about cross-region replication, see *Manage cross-region replication*.

Enable cross-region replication

Run the following code to enable cross-region replication:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

AddBucketReplicationRequest request = new AddBucketReplicationRequest(
bucketName);
request.setReplicationRuleID("<yourRuleId>");
request.setTargetBucketName("<yourTargetBucketName>");
// This example uses the target endpoint China (Beijing).
request.setTargetBucketLocation("oss-cn-beijing");
// Disable historical data replication. Historical data replication is
 enabled by default.
request.setEnableHistoricalObjectReplication(false);
ossClient.addBucketReplication(request);

// Close your OSSClient.
ossClient.shutdown();
```

View cross-region replication configurations

Use the following code to view the cross-region replication configurations for a bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

List<ReplicationRule> rules = ossClient.getBucketReplication(
bucketName);
for (ReplicationRule rule : rules) {
    System.out.println(rule.getReplicationRuleID());
    System.out.println(rule.getTargetBucketLocation());
    System.out.println(rule.getTargetBucketName());
}

// Close your OSSClient.
ossClient.shutdown();
```

View cross-region replication progress

Cross-region replication progress includes the progress of historical and real-time data synchronization.

- **Percent signs (%) indicate the progress of historical data synchronization. This function only works for buckets for which historical data synchronization has been enabled.**
- **Time points (the point of latest time the data is written) indicate the progress of real-time data synchronization. If a time point is displayed, the data created before this time point has been synchronized.**

Run the following code to view the cross-region replication progress:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

BucketReplicationProgress process = ossClient.getBucketReplication
Progress(bucketName, "<yourRuleId>");
```

```
System.out.println(process.getReplicationRuleID());
// Check whether historical data synchronization is enabled.
System.out.println(process.isEnableHistoricalObjectReplication());
// Print historical data synchronization progress.
System.out.println(process.getHistoricalObjectProgress());
// Print real-time data synchronization progress.
System.out.println(process.getNewObjectProgress());

// Close your OSSClient.
ossClient.shutdown();
```

View the target regions for synchronization

Run the following code to obtain the target regions to which buckets can be synchronized:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

List<String> locations = ossClient.getBucketReplicationLocation(
bucketName);
for (String loc : locations) {
    System.out.println(loc);
}

// Close your OSSClient.
ossClient.shutdown();
```

Disable cross-region replication

Use the following code to disable cross-region replication:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
```

```
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Disable cross-region replication. After this function is disabled
, objects in the target bucket exist and all changes in the source
objects cannot be synchronized.
ossClient.deleteBucketReplication(bucketName, "<yourRuleId>");

// Close your OSSClient.
ossClient.shutdown();
```

2.5.9 CORS

This topic describes how to perform CORS.

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* and *PutBucketcors* in OSS Developer Guide.

Configure CORS rules

Run the following code to configure CORS rules for the specified bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

SetBucketCORSRequest request = new SetBucketCORSRequest(bucketName);

// Create a CORS rule. A maximum of 10 rules can be configured for
each bucket.
ArrayList<CORSRule> putCorsRules = new ArrayList<CORSRule>();

CORSRule corRule = new CORSRule();

ArrayList<String> allowedOrigin = new ArrayList<String>();
// Specify the source of the cross-origin access request.
allowedOrigin.add( "http://www.b.com");

ArrayList<String> allowedMethod = new ArrayList<String>();
// Specify the cross-region request methods (GET, PUT, DELETE, POST,
and HEAD) that are allowed.
```

```
allowedMethod.add("GET");

ArrayList<String> allowedHeader = new ArrayList<String>();
// Specify whether the header specified in Access-Control-Request-
Headers of pre-flight request (OPTIONS) is allowed.
allowedHeader.add("x-oss-test");

ArrayList<String> exposedHeader = new ArrayList<String>();
// Specify the response header that allows user access from applicatio
ns.
exposedHeader.add("x-oss-test1");
// AllowedOrigins and AllowedMethods allow only one wildcard asterisk
 (*). Wildcard asterisks (*) indicate that all sources of the cross-
origin requests and operations are allowed.
corRule.setAllowedMethods(allowedMethod);
corRule.setAllowedOrigins(allowedOrigin);
// AllowedHeaders and ExposeHeaders do not allow wildcard asterisks
 (*).
corRule.setAllowedHeaders(allowedHeader);
corRule.setExposeHeaders(exposedHeader);
// Specify the cache time (seconds) for the response of browser pre-
flight (OPTIONS) requests to a specific resource.
corRule.setMaxAgeSeconds(10);

// A maximum of 10 rules is allowed.
putCorsRules.add(corRule);
// The existing rules will be replaced.
request.setCorsRules(putCorsRules);
ossClient.setBucketCORS(request);

// Close your OSSClient.
ossClient.shutdown();
```

Obtain CORS rules

Run the following code to obtain CORS rules:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

ArrayList<CORSRule> corsRules;
// Obtain the CORS rule list.
corsRules =  (ArrayList<CORSRule>) ossClient.getBucketCORSRules(
bucketName);
for (CORSRule rule : corsRules) {
    for (String allowedOrigin1 : rule.getAllowedOrigins()) {
        // Obtain allowed sources of cross-origin requests.
        System.out.println(allowedOrigin1);
```

```
    }

    for (String allowedMethod1 : rule.getAllowedMethods()) {
        // Obtain the allowed cross-origin request method.
        System.out.println(allowedMethod1);
    }

    if (rule.getAllowedHeaders().size() > 0){
        for (String allowedHeader1 : rule.getAllowedHeaders()) {
            // Obtain the header list for allowed cross-origin
requests.
            System.out.println(allowedHeader1);
        }
    }

    if (rule.getExposeHeaders().size() > 0) {
        for (String exposeHeader : rule.getExposeHeaders()) {
            // Obtain the header for an allowed cross-origin request.
            System.out.println(exposeHeader);
        }
    }

    if ( null ! = rule.getMaxAgeSeconds()) {
        System.out.println(rule.getMaxAgeSeconds());
    }
}

// Close your OSSClient.
ossClient.shutdown();
```

Delete CORS rules

Run the following code to delete all CORS rules for the specified bucket:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Delete all CORS rules for a specified bucket.
ossClient.deleteBucketCORSRules(bucketName);

// Close your OSSClient.
ossClient.shutdown();
```

2.6 Upload objects

2.6.1 Overview

This topic is the overview of object upload in OSS Java SDK.

In OSS, an object is the basic unit for data operations. OSS Java SDK provides the following file upload methods:

- *Simple upload*: **supports the upload of a file up to 5 GB in size. It includes streaming upload and file upload.**
- *Form upload*: **supports the upload of a file up to 5 GB in size.**
- *Append upload*: **supports the upload of a file up to 5 GB in size.**
- *Resumable upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files. It supports concurrent upload and you can define the size of each part.**
- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files.**

Note:

For more information about the usage scenarios of each upload method, see "Upload files" in OSS Developer Guide.

During the file upload, you can set *Object Meta*, **and view the upload progress in the** *upload progress bar*. **After you complete the file upload, you can perform** *upload callback*.

2.6.2 Simple upload

Simple upload allows you to use the PutObject operation to upload one object to OSS. This topic describes how to perform simple upload by using the Java SDK.

Simple upload includes stream upload and file upload. Stream upload uses InputStream as the source object. File upload uses a local file as the source object.

Note:

For the complete code used to perform simple upload, visit *GitHub*.

Stream upload

Use ossClient.putObject to upload a data stream to OSS.

- **Upload a string**

  The following code provides an example on how to upload a string:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create a PutObjectRequest object.
String content = "Hello OSS";
// <yourObjectName> indicates the complete path of the object you
want to upload to OSS, and must include the file extension of the
object. For example, set <yourObjectName> to abc/efg/123.jpg.
PutObjectRequest putObjectRequest = new PutObjectRequest("<
yourBucketName>", "<yourObjectName>", new ByteArrayInputStream(
content.getBytes()));

// Optional. Specify the storage class and ACL.
// ObjectMetadata metadata = new ObjectMetadata();
// metadata.setHeader(OSSHeaders.OSS_STORAGE_CLASS, StorageClass.
Standard.toString());
// metadata.setObjectAcl(CannedAccessControlList.Private);
// putObjectRequest.setMetadata(metadata);

// Upload a string.
ossClient.putObject(putObjectRequest);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Upload a byte array**

  The following code provides an example on how to upload a byte array:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
```

```
// Upload a byte array.
byte[] content = "Hello OSS".getBytes();
ossClient.putObject("<yourBucketName>", "<yourObjectName>", new
ByteArrayInputStream(content));

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Upload a network stream**

  The following code provides an example on how to upload a network stream:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Upload a network stream.
InputStream inputStream = new URL("https://www.aliyun.com/").
openStream();
ossClient.putObject("<yourBucketName>", "<yourObjectName>",
inputStream);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Upload a file stream**

  The following code provides an example on how to upload a file stream:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
 Alibaba Cloud account to log on to OSS, because the account has
 permissions on all API operations. To ensure cloud security,
we recommend that you follow best practices of Resource Access
Management and use a RAM user account to call API operations or
perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Upload a file stream.
InputStream inputStream = new FileInputStream("<yourlocalFile>");
ossClient.putObject("<yourBucketName>", "<yourObjectName>",
inputStream);
```

```
// Shut down the OSSClient instance.
ossClient.shutdown();
```

File upload

The following code provides an example on how to upload a local file to OSS:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create a PutObjectRequest object.
PutObjectRequest putObjectRequest = new PutObjectRequest("<yourBucket
Name>", "<yourObjectName>", new File("<yourLocalFile>"));

// Optional. Specify the storage class and ACL.
// ObjectMetadata metadata = new ObjectMetadata();
// metadata.setHeader(OSSHeaders.OSS_STORAGE_CLASS, StorageClass.
Standard.toString());
// metadata.setObjectAcl(CannedAccessControlList.Private);
// putObjectRequest.setMetadata(metadata);

// Upload a local file.
ossClient.putObject(putObjectRequest);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

2.6.3 Form upload

Form upload uses an HTML form to upload a file to a specified bucket. The maximum size of files is 5 GB for each form upload.

For more information about the application scenarios of form upload, see *Form upload* in OSS Developer Guide.

Run the following code for form upload:

```
public class PostObjectSample {
    // Upload a file.
    private String localFilePath = "<yourLocalFile>";
    // This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
    private String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
    // It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
    private String accessKeyId = "<yourAccessKeyId>";
    private String accessKeySecret = "<yourAccessKeySecret>";
    // Specify the bucket name.
    private String bucketName = "<yourBucketName>";
    // Specify the file name.
    private String objectName = "<yourObjectName>";
    /**
     * Start form upload.
     * @throws Exception
     */
    private void PostObject() throws Exception {
        // Add the bucket name to the URL. The URL format is http://
yourBucketName.oss-cn-hangzhou.aliyuncs.com.
        String urlStr = endpoint.replace("http://", "http://" +
bucketName+ ".") ;
        // Create Map for the form.
        Map<String, String> formFields = new LinkedHashMap<String,
String>();
        // Specify the file name.
        formFields.put("key", this.objectName);
        // Configure Content-Disposition.
        formFields.put("Content-Disposition", "attachment;filename="
                + localFilePath);
        // Configure the callback parameter.
        Callback callback = new Callback();
        // Configure the IP address of the server you want to send the
 callback request to, for example, http://oss-demo.aliyuncs.com:23450
or http://127.0.0.1:9090.
        callback.setCallbackUrl("<yourCallbackServerUrl>");
        // Configure the host field value in the callback request
header, such as oss-cn-hangzhou.aliyuncs.com.
        callback.setCallbackHost("<yourCallbackServerHost>");
        // Configure the body field value for the callback request.
        callback.setCallbackBody("{\\\"mimeType\\\":${mimeType},\\\"
size\\\":${size}}");
        // Configure Content-Type for the callback request.
        callback.setCalbackBodyType(CalbackBodyType.JSON);
        // Configure the custom parameters used for sending the
callback request. Each custom parameter consists of a key and a value
. The key must be in lowercase and start with x:.
        callback.addCallbackVar("x:var1", "value1");
        callback.addCallbackVar("x:var2", "value2");
        // Configure the callback parameter for Map in the form.
        setCallBack(formFields, callback);
        // Configure OSSAccessKeyId.
        formFields.put("OSSAccessKeyId", accessKeyId);
        String policy = "{\"expiration\": \"2120-01-01T12:00:00.000Z
\",\"conditions\": [[\"content-length-range\", 0, 104857600]]}";
        String encodePolicy = new String(Base64.encodeBase64(policy.
getBytes()));
        // Configure the policy.
        formFields.put("policy", encodePolicy);
        // Generate the URL for signature.
        String signaturecom = com.aliyun.oss.common.auth.ServiceSig
nature.create().computeSignature(accessKeySecret, encodePolicy);
        // Configure the signature.
        formFields.put("Signature", signaturecom);
        String ret = formUpload(urlStr, formFields, localFilePath);
        System.out.println("Post Object [" + this.objectName + "] to
bucket [" + bucketName + "]");
        System.out.println("post reponse:" + ret);
```

```
    }
    private static String formUpload(String urlStr, Map<String, String
> formFields, String localFile)
            throws Exception {
        String res = "";
        HttpURLConnection conn = null;
        String boundary = "9431149156168";
        try {
            URL url = new URL(urlStr);
            conn = (HttpURLConnection) url.openConnection();
            conn.setConnectTimeout(5000);
            conn.setReadTimeout(30000);
            conn.setDoOutput(true);
            conn.setDoInput(true);
            conn.setRequestMethod("POST");
            conn.setRequestProperty("User-Agent",
                    "Mozilla/5.0 (Windows; U; Windows NT 6.1; zh-CN;
rv:1.9.2.6)");
            // Configure the MD5 value. The MD5 value is calculated
based on the entire body.
            conn.setRequestProperty("Content-MD5", "<yourContentMD5
>");
            conn.setRequestProperty("Content-Type",
                    "multipart/form-data; boundary=" + boundary);
            OutputStream out = new DataOutputStream(conn.getOutputS
tream());
            // Recursively read all Map data in the form and write the
 data to the OutputStream.
            if (formFields ! = null) {
                StringBuffer strBuf = new StringBuffer();
                Iterator<Entry<String, String>> iter = formFields.
entrySet().iterator();
                int i = 0;
                while (iter.hasNext()) {
                    Entry<String, String> entry = iter.next();
                    String inputName = entry.getKey();
                    String inputValue = entry.getValue();
                    if (inputValue == null) {
                        continue;
                    }
                    if (i == 0) {
                        strBuf.append("--").append(boundary).append("\
r\n");
                        strBuf.append("Content-Disposition: form-data
; name=\""
                                + inputName + "\"\r\n\r\n");
                        strBuf.append(inputValue);
                    } else {
                        strBuf.append("\r\n").append("--").append(
boundary).append("\r\n");
                        strBuf.append("Content-Disposition: form-data
; name=\""
                                + inputName + "\"\r\n\r\n");
                        strBuf.append(inputValue);
                    }
                    i++;
                }
                out.write(strBuf.toString().getBytes());
            }
            // Read the file and write it to the OutputStream.
            File file = new File(localFile);
            String filename = file.getName();
            String contentType = new MimetypesFileTypeMap().getContent
Type(file);
```

```
            if (contentType == null || contentType.equals("")) {
                contentType = "application/octet-stream";
            }
            StringBuffer strBuf = new StringBuffer();
            strBuf.append("\r\n").append("--").append(boundary)
                    .append("\r\n");
            strBuf.append("Content-Disposition: form-data; name=\"file
\"; "
                    + "filename=\"" + filename + "\"\r\n");
            strBuf.append("Content-Type: " + contentType + "\r\n\r\n
");
            out.write(strBuf.toString().getBytes());
            DataInputStream in = new DataInputStream(new FileInputS
tream(file));
            int bytes = 0;
            byte[] bufferOut = new byte[1024];
            while ((bytes = in.read(bufferOut)) ! = -1) {
                out.write(bufferOut, 0, bytes);
            }
            in.close();
            byte[] endData = ("\r\n--" + boundary + "--\r\n").getBytes
();
            out.write(endData);
            out.flush();
            out.close();
            // Read the returned data.
            strBuf = new StringBuffer();
            BufferedReader reader = new BufferedReader(new InputStrea
mReader(conn.getInputStream()));
            String line = null;
            while ((line = reader.readLine()) ! = null) {
                strBuf.append(line).append("\n");
            }
            res = strBuf.toString();
            reader.close();
            reader = null;
        } catch (Exception e) {
            System.err.println("Send post request exception: " + e);
            throw e;
        } finally {
            if (conn ! = null) {
                conn.disconnect();
                conn = null;
            }
        }
        return res;
    }
     private static void setCallBack(Map<String, String> formFields,
Callback callback) {
        if (callback ! = null) {
            String jsonCb = OSSUtils.jsonizeCallback(callback);
            String base64Cb = BinaryUtil.toBase64String(jsonCb.
getBytes());
            formFields.put("callback", base64Cb);
            if (callback.hasCallbackVar()) {
                Map<String, String> varMap = callback.getCallbackVar
();
                for (Entry<String, String> entry : varMap.entrySet())
 {
                    formFields.put(entry.getKey(), entry.getValue());
                }
            }
        }
    }
```

```
    public static void main(String[] args) throws Exception {
        PostObjectSample ossPostObject = new PostObjectSample();
        ossPostObject.PostObject();
    }
}
```

Note:

For more information about callback, see *Callback*.

2.6.4 Append upload

This topic describes how to use append upload.

You cannot perform copyObject for appended objects.

Run the following code for append upload:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String content1 = "Hello OSS A \n";
String content2 = "Hello OSS B \n";
String content3 = "Hello OSS C \n";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

ObjectMetadata meta = new ObjectMetadata();
// Specify the content type of the object you want to upload.
meta.setContentType("text/plain");

// Configure parameters with AppendObjectRequest.
AppendObjectRequest appendObjectRequest = new AppendObjectRequest
("<yourBucketName>", "<yourObjectName>", new ByteArrayInputStream(
content1.getBytes()),meta);

// Configure a single parameter with AppendObjectRequest.
// Specify the bucket name.
//appendObjectRequest.setBucketName("<yourBucketName>");
// Specify the object name.
//appendObjectRequest.setKey("<yourObjectName>");
// Configure the type of content you want to append. Two types of
content are available: InputStream and File. Select InputStream.
//appendObjectRequest.setInputStream(new ByteArrayInputStream(content1
.getBytes()));
// Configure the type of content you want to append. Two types of
content are available: InputStream and File. Select File.
//appendObjectRequest.setFile(new File("<yourLocalFile>"));
// Specify object meta, because the specified object meta takes effect
 from the first append.
```

```
//appendObjectRequest.setMetadata(meta);

// Start the first append.
// Configure the position where the object is appended based on the
previous object length.
appendObjectRequest.setPosition(0L);
AppendObjectResult appendObjectResult = ossClient.appendObject(
appendObjectRequest);
// Calculate the 64-bit CRC value. The value is calculated based on
 the ECMA-182 standard. This value is calculated based on ECMA-182
criteria.
System.out.println(appendObjectResult.getObjectCRC());

// Start the second append.
// nextPosition indicates the position provided for the next request,
namely, the length of the current object.
appendObjectRequest.setPosition(appendObjectResult.getNextPosition());
appendObjectRequest.setInputStream(new ByteArrayInputStream(content2.
getBytes()));
appendObjectResult = ossClient.appendObject(appendObjectRequest);

// Start the third append.
appendObjectRequest.setPosition(appendObjectResult.getNextPosition());
appendObjectRequest.setInputStream(new ByteArrayInputStream(content3.
getBytes()));
appendObjectResult = ossClient.appendObject(appendObjectRequest);

// Close your OSSClient.
ossClient.shutdown();
```

For more information about append upload, see *Append object* in OSS Developer Guide. For the complete code of append upload, see *GitHub*.

2.6.5 Resumable upload

You can use resumable upload to split a file you want to upload into several parts and upload them simultaneously. After you have uploaded all parts, you can combine these parts into a complete object. Thus, you can complete uploading the entire object.

For more information about resumable upload, see *Resumable upload* in the OSS Developer Guide. For the complete code of resumable upload, see *GitHub*.

Run the following code for resumable upload:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
```

```
// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

ObjectMetadata meta = new ObjectMetadata();
// Specify the content type of the object you want to upload.
meta.setContentType("text/plain");

// Configure parameters with UploadFileRequest.
UploadFileRequest uploadFileRequest = new UploadFileRequest("<
yourBucketName>","<yourObjectName>");

// Configure parameters with UploadFileRequest.
// Specify the bucket name.
//uploadFileRequest.setBucketName("<yourBucketName>");
// Specify the object name.
//uploadFileRequest.setKey("<yourObjectName>");
// Specify the local file you want to upload.
uploadFileRequest.setUploadFile("<yourLocalFile>");
// Specify the number of threads for simultaneous upload. The default
value is 1.
uploadFileRequest.setTaskNum(5);
// Specify the size of a part you want to upload. The value can be
from 100 KB to 5 GB. The default size of a part is calculated as: file
 size/10,000.
uploadFileRequest.setPartSize(1 * 1024 * 1024);
// Start resumable upload (that is disabled by default.).
uploadFileRequest.setEnableCheckpoint(true);
// Configure the file used to record the result of local part uploads
. You need to configure this parameter when you enable resumable
upload. This file stores progress information. If a part upload fails
, it can be continued based on the progress information recorded in
this file. After you have uploaded the file, the file is deleted. By
default, this file shares the same directory (uploadFile.ucp) as the
file you want to upload.
uploadFileRequest.setCheckpointFile("<yourCheckpointFile>");
// Configure object meta.
uploadFileRequest.setObjectMetadata(meta);
// Configure callback. Set the yourCallbackEvent parameter to Callback
.
uploadFileRequest.setCallback("<yourCallbackEvent>");

// Start resumable upload.
ossClient.uploadFile(uploadFileRequest);

// Close your OSSClient.
ossClient.shutdown();
```

2.6.6 Multipart upload

This topic describes how to use multipart upload.

For the complete code used to perform multipart upload, visit *GitHub*.

To perform multipart upload, follow these steps:

1. **Initiate a multipart upload task.**

   Use the ossClient.initiateMultipartUpload method to obtain an upload ID that is globally unique in OSS.
2. **Start multipart upload.**

   Use the ossClient.uploadPart method to upload the parts.

   Note:

   - **For parts that share an upload ID, partNumber identifies the relative position of a part in the entire object. If you have uploaded a part and used its partNumber again to upload another part, the latter part overwrites the former part.**
   - **OSS places the MD5 value of part data in the ETag header and returns the MD5 value to the user.**
   - **OSS calculates the MD5 value of uploaded data and compares it with the MD5 value calculated by the SDK. If the two values are different, the InvalidDigest error code is returned.**
3. **Complete multipart upload.**

   After all of the parts have been uploaded, use the ossClient.completeMultipartUpload method to combine these parts into a complete object.

The following code provides an example on how to use multipart upload:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
// <yourObjectName> indicates the complete path of the object you want
 to upload to OSS, and must include the file extension of the object.
For example, set <yourObjectName> to abc/efg/123.jpg.
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create an InitiateMultipartUploadRequest object.
InitiateMultipartUploadRequest request = new InitiateMultipartUpl
oadRequest("<yourBucketName>", "<yourObjectName>");
```

```
// Optional. Specify the storage class.
// ObjectMetadata metadata = new ObjectMetadata();
// metadata.setHeader(OSSHeaders.OSS_STORAGE_CLASS, StorageClass.
Standard.toString());
// request.setObjectMetadata(metadata);

// Initiate a multipart upload task.
InitiateMultipartUploadResult upresult = ossClient.initiateMu
ltipartUpload(request);
// Obtain an upload ID. The upload ID is the unique identification of
 a multipart upload task. You can initiate related operations such as
canceling multipart upload and querying multipart upload based on the
upload ID.
String uploadId = upresult.getUploadId();

// partETags is a set of ETags of parts. A PartETag consists of an
ETag and a part number.
List<PartETag> partETags =  new ArrayList<PartETag>();
// Calculate the total number of upload parts.
final long partSize = 1 * 1024 * 1024L;   // 1MB
final File sampleFile = new File("<localFile>");
long fileLength = sampleFile.length();
int partCount = (int) (fileLength / partSize);
if (fileLength % partSize ! = 0) {
    partCount++;
 }
// Upload each part until all parts are uploaded.
for (int i = 0; i < partCount; i++) {
    long startPos = i * partSize;
    long curPartSize = (i + 1 == partCount) ? (fileLength - startPos
) : partSize;
    InputStream instream = new FileInputStream(sampleFile);
    // Skip the parts that have been uploaded.
    instream.skip(startPos);
    UploadPartRequest uploadPartRequest = new UploadPartRequest();
    uploadPartRequest.setBucketName(bucketName);
    uploadPartRequest.setKey(objectName);
    uploadPartRequest.setUploadId(uploadId);
    uploadPartRequest.setInputStream(instream);
    // Configure the size of each part. All parts are at least 100 KB
except for the last part. This limit does not apply to the last part.
    uploadPartRequest.setPartSize(curPartSize);
    // Configure part numbers. Each part is configured with a part
number. The value can be from 1 to 10000. If you configure a number
beyond this range, OSS returns an InvalidArgument error code.
    uploadPartRequest.setPartNumber( i + 1);
    // Parts are not necessarily uploaded in order. They can be
uploaded from different OSS clients. OSS sorts the parts based on
their part numbers and combines them to obtain a complete object.
    UploadPartResult uploadPartResult = ossClient.uploadPart(
uploadPartRequest);
    // When you have uploaded a part, OSS returns a result that
contains a PartETag. The PartETag is stored in partETags.
    partETags.add(uploadPartResult.getPartETag());
}


// Create a CompleteMultipartUploadRequest object.
// When you have completed multipart upload, you must provide all
 valid partETags. After receiving the partETags, OSS verifies the
validity of all parts one by one. If part verification is successful,
OSS combines these parts into a complete object.
CompleteMultipartUploadRequest completeMultipartUploadRequest =
```

```
        new CompleteMultipartUploadRequest(bucketName, objectName,
uploadId, partETags);

// Optional. Set the ACL.
// completeMultipartUploadRequest.setObjectACL(CannedAccessControlList
.PublicRead);

// Complete multipart upload.
CompleteMultipartUploadResult completeMultipartUploadResult =
ossClient.completeMultipartUpload(completeMultipartUploadRequest);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

Cancel a multipart upload task

You can use the ossClient.abortMultipartUpload method to cancel a multipart upload task. If a multipart upload task is canceled, the upload ID can no longer be used to perform further operations. The parts uploaded with that upload ID are also deleted.

The following code provides an example on how to cancel a multipart upload task:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Cancel the multipart upload task. The upload ID is returned by
InitiateMultipartUpload.
AbortMultipartUploadRequest abortMultipartUploadRequest =
        new AbortMultipartUploadRequest("<yourBucketName>", "<
yourObjectName>", "<uploadId>");
ossClient.abortMultipartUpload(abortMultipartUploadRequest);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

List uploaded parts

You can use the ossClient.listParts method to list all parts that are uploaded with the specified upload ID.

- **Simple list**

  The following code provides an example for simple list:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List uploaded parts. The upload ID is returned by InitiateMu
ltipartUpload.
ListPartsRequest listPartsRequest = new ListPartsRequest("<
yourBucketName>", "<yourObjectName>", "<uploadId>");
 // Configure an upload ID.
 //listPartsRequest.setUploadId(uploadId);
 // Set the maximum number of parts to list on each page to 100. By
default, 1,000 tasks are listed.
 listPartsRequest.setMaxParts(100);
 // Specify the start point of the list. Only the parts with part
numbers greater than the value of this parameter are listed.
 listPartsRequest.setPartNumberMarker(2);
PartListing partListing = ossClient.listParts(listPartsRequest);

for (PartSummary part : partListing.getParts()) {
    // Obtain part numbers.
    part.getPartNumber();
    // Obtain the part size.
    part.getSize();
    // Obtain the ETag.
    part.getETag();
    // Obtain the last modification time.
    part.getLastModified();
}

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **List all uploaded parts**

  By default, listParts can list up to 1,000 parts at a time. The following code provides an example on how to list more than 1,000 parts:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
```

```
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List all uploaded parts.
PartListing partListing;
ListPartsRequest listPartsRequest = new ListPartsRequest("<
yourBucketName>", "<yourObjectName>", "<uploadId>");

do {
    partListing = ossClient.listParts(listPartsRequest);

    for (PartSummary part : partListing.getParts()) {
        // Obtain part numbers.
        part.getPartNumber();
        // Obtain the part size.
        part.getSize();
        // Obtain the ETag.
        part.getETag();
        // Obtain the last modification time.
        part.getLastModified();
    }
// Specify the start point of the list. Only the parts with part
numbers greater than the value of this parameter are listed.
listPartsRequest.setPartNumberMarker(partListing.getNextPar
tNumberMarker());
} while (partListing.isTruncated());

// Shut down the OSSClient instance.
ossClient.shutdown()
```

- **List all uploaded parts by page**

  The following code provides an example on how to specify the maximum number of parts to list on each page and list all uploaded parts by page:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List all uploaded parts by page.
PartListing partListing;
ListPartsRequest listPartsRequest = new ListPartsRequest("<
yourBucketName>", "<yourObjectName>", "<uploadId>");
```

```
// Set the maximum number of parts to list on each page to 100.
listPartsRequest.setMaxParts(100);

do {
    partListing = ossClient.listParts(listPartsRequest);

    for (PartSummary part : partListing.getParts()) {
        // Obtain part numbers.
        part.getPartNumber();
        // Obtain the part size.
        part.getSize();
        // Obtain the ETag.
        part.getETag();
        // Obtain the last modification time.
        part.getLastModified();
    }

    listPartsRequest.setPartNumberMarker(partListing.getNextPar
tNumberMarker());

} while (partListing.isTruncated());

// Shut down the OSSClient instance.
ossClient.shutdown();
```

List multipart upload tasks

You can use the ossClient.listMultipartUploads method to list all ongoing multipart upload tasks (tasks that have been initiated but not completed or tasks that have been canceled). The following table describes the parameters.

Parameter	Description	Configuration method
prefix	**Specifies the prefix that returned object names must contain. Note that if you use a prefix for a query, the returned object name contains the prefix.**	**ListMultipartUploads Request.setPrefix(String prefix)**
delimiter	**Groups objects by name . The substring between the specified prefix and the occurrence of the first delimiter is commonPref ixes. A substring of object names starts with the prefix and ends with the next occurrence of the specified delimiter.**	**ListMultipartUploads Request.setDelimiter(String delimiter)**

Parameter	Description	Configuration method
maxUploads	Specifies the maximum number of multipart upload tasks returned at a time. The default value is 1000. The maximum value is 1000.	ListMultipartUploadsRequest.setMaxUploads(Integer maxUploads)
keyMarker	Specifies the name of the object after which the listing of multipart upload tasks begins. All multipart upload tasks with objects whose names start with a letter that comes after the keyMarker parameter value in alphabetical order are listed. You can use this parameter with the uploadIdMarker parameter to specify the start point to list the returned results.	ListMultipartUploadsRequest.setKeyMarker(String keyMarker)

Parameter	Description	Configuration method
uploadIdMarker	UploadIdMarker is used with the keyMarker parameter to specify the start point to list the returned results. If the keyMarker parameter is not configured, the uploadIdMarker parameter is invalid. If the keyMarker parameter is configured, the query result includes: · All objects whose names start with a letter that comes after the keyMarker parameter value in alphabetical order. · All objects whose names start with a letter that is the same as the keyMarker parameter value in alphabetical order and of which the uploadId parameter value is greater than the uploadIdMarker parameter value.	ListMultipartUploadsRequest.setUploadIdMarker(String uploadIdMarker)

· **Simple list**

The following code provides an example for simple list:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
```

```
// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List multipart upload tasks. By default, 1,000 tasks are listed.
ListMultipartUploadsRequest listMultipartUploadsRequest = new
ListMultipartUploadsRequest(bucketName);
MultipartUploadListing multipartUploadListing = ossClient.listMultip
artUploads(listMultipartUploadsRequest);

for (MultipartUpload multipartUpload : multipartUploadListing.
getMultipartUploads()) {
    // Obtain upload IDs.
    multipartUpload.getUploadId();
    // Obtain object names.
    multipartUpload.getKey();
    // Obtain the time the multipart upload tasks are initiated.
    multipartUpload.getInitiated();
}

// Shut down the OSSClient instance.
ossClient.shutdown();
```

If the value of the isTruncated field in the returned result is false, values of nextKeyMarker and nextUploadIdMarker are returned and used as the start point for the next object reading. If all multipart upload tasks cannot be returned at a time, list them by page.

- **List all multipart upload tasks**

  By default, listMultipartUploads can list up to 1,000 multipart upload tasks at a time. The following code provides an example on how to list more than 1,000 multipart upload tasks:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List multipart upload tasks.
MultipartUploadListing multipartUploadListing;
ListMultipartUploadsRequest listMultipartUploadsRequest = new
ListMultipartUploadsRequest(bucketName);

do {
```

```
    multipartUploadListing = ossClient.listMultipartUploads(
listMultipartUploadsRequest);

    for (MultipartUpload multipartUpload : multipartUploadListing.
getMultipartUploads()) {
        // Obtain upload IDs.
        multipartUpload.getUploadId();
        // Obtain object names.
        multipartUpload.getKey();
        // Obtain the time the multipart upload tasks are initiated.
        multipartUpload.getInitiated();
    }

    listMultipartUploadsRequest.setKeyMarker(multipartUploadListing.
getNextKeyMarker());

    listMultipartUploadsRequest.setUploadIdMarker(multipartU
ploadListing.getNextUploadIdMarker());
} while (multipartUploadListing.isTruncated());

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **List all multipart upload tasks by page**

  The following code provides an example on how to list all multipart upload tasks by page:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List multipart upload tasks.
MultipartUploadListing multipartUploadListing;
ListMultipartUploadsRequest listMultipartUploadsRequest = new
ListMultipartUploadsRequest(bucketName);
// Specify the number of multipart upload tasks to list on each page
.
listMultipartUploadsRequest.setMaxUploads(50);

do {
    multipartUploadListing = ossClient.listMultipartUploads(
listMultipartUploadsRequest);

    for (MultipartUpload multipartUpload : multipartUploadListing.
getMultipartUploads()) {
        // Obtain upload IDs.
        multipartUpload.getUploadId();
```

```
            // Obtain object names.
            multipartUpload.getKey();
            // Obtain the time the multipart upload tasks are initiated.
            multipartUpload.getInitiated();
        }

        listMultipartUploadsRequest.setKeyMarker(multipartUploadListing.
    getNextKeyMarker());
        listMultipartUploadsRequest.setUploadIdMarker(multipartU
    ploadListing.getNextUploadIdMarker());

    } while (multipartUploadListing.isTruncated());

    // Shut down the OSSClient instance.
    ossClient.shutdown();
```

2.6.7 Upload progress bars

You can use progress bars to indicate the upload or download progress.

The following code is used as an example to describe how to view progress information with ossClient.putObject:

```
public class PutObjectProgressListener implements ProgressListener {
    private long bytesWritten = 0;
    private long totalBytes = -1;
    private boolean succeed = false;

    @Override
    public void progressChanged(ProgressEvent progressEvent) {
        long bytes = progressEvent.getBytes();
        ProgressEventType eventType = progressEvent.getEventType();
        switch (eventType) {
        case TRANSFER_STARTED_EVENT:
            System.out.println("Start to upload......");
            break;
        case REQUEST_CONTENT_LENGTH_EVENT:
            this.totalBytes = bytes;
            System.out.println(this.totalBytes + " bytes in total will
 be uploaded to OSS");
            break;
        case REQUEST_BYTE_TRANSFER_EVENT:
            this.bytesWritten += bytes;
            if (this.totalBytes ! = -1) {
                int percent = (int)(this.bytesWritten * 100.0 / this.
totalBytes);
                System.out.println(bytes + " bytes have been written
at this time, upload progress: " + percent + "%(" + this.bytesWritten
 + "/" + this.totalBytes + ")");
            } else {
                System.out.println(bytes + " bytes have been written
 at this time, upload ratio: unknown" + "(" + this.bytesWritten +
 "/...)");
            }
            break;
        case TRANSFER_COMPLETED_EVENT:
            this.succeed = true;
            System.out.println("Succeed to upload, " + this.bytesWritt
en + " bytes have been transferred in total");
            break;
```

```
            case TRANSFER_FAILED_EVENT:
                System.out.println("Failed to upload, " + this.bytesWritt
en + " bytes have been transferred");
                break;
            default:
                break;
            }
        }

        public boolean isSucceed() {
            return succeed;
        }

        public static void main(String[] args) {
            // This example uses endpoint China (Hangzhou). Specify the
actual endpoint based on your requirements.
            String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
            // It is highly risky to log on with AccessKey of an Alibaba
 Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs or
perform routine operations and maintenance. To create a RAM account,
log on to https://ram.console.aliyun.com.
            String accessKeyId = "<yourAccessKeyId>";
            String accessKeySecret = "<yourAccessKeySecret>";
            String bucketName = "<yourBucketName>";
            String objectName = "<yourObjectName>";

            // Create an OSSClient instance.
            OSS ossClient = new OSSClientBuilder().build(endpoint,
accessKeyId, accessKeySecret);

            try {
                // Upload with a progress bar displayed.
                ossClient.putObject(new PutObjectRequest(bucketName,
objectName, new File("<yourLocalFile>")).
                        <PutObjectRequest>withProgressListener(new
PutObjectProgressListener()));

            } catch (Exception e) {
                e.printStackTrace();
            }
            // Close your OSSClient instance.
            ossClient.shutdown();
        }
}
```

You can view progress information with ossClient.putObject, ossClient.getObject , or ossClient.uploadPart. However, you cannot view progress information with ossClient.uploadFile or ossClient.downloadFile.

For the complete code of upload progress bar, see *GitHub*.

2.6.8 Upload callback

This topic describes how to use

For the complete code of upload callback, see *GitHub*.

Run the following code for upload callback:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";
// Specify the IP address of the server you want to send the callback
request to, for example, http://oss-demo.aliyuncs.com:23450 or http://
127.0.0.1:9090.
String callbackUrl = "<yourCallbackServerUrl>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

String content = "Hello OSS";
PutObjectRequest putObjectRequest = new PutObjectRequest(bucketName,
objectName,new ByteArrayInputStream(content.getBytes()));

// Configure upload callback parameters.
Callback callback = new Callback();
callback.setCallbackUrl(callbackUrl);
// Configure the value of the host field carried in the callback
request header, such as oss-cn-hangzhou.aliyuncs.com.
callback.setCallbackHost("oss-cn-hangzhou.aliyuncs.com");
// Configure the value of the body field carried in the callback
request.
callback.setCallbackBody("{\\\"mimeType\\\":${mimeType},\\\"size\\\":
${size}}");
// Configure Content-Type for the callback request.
callback.setCalbackBodyType(CallbackBodyType.JSON);
// Configure the custom parameters used to initiate a callback request
. Each custom parameter consists of a key and a value. The key must
start with x:.
callback.addCallbackVar("x:var1", "value1");
callback.addCallbackVar("x:var2", "value2");
putObjectRequest.setCallback(callback);

PutObjectResult putObjectResult = ossClient.putObject(putObjectRequest
);

// Read the message returned from upload callback.
byte[] buffer = new byte[1024];
putObjectResult.getCallbackResponseBody().read(buffer);
// If you do not close the reader after the data is read, connection
leaks may occur. Consequently, no available connections are left and
an exception occurs.
putObjectResult.getCallbackResponseBody().close();

// Close your OSSClient instance.
ossClient.shutdown();
```

For more information, see *Upload callback* in OSS Developer Guide.

2.7 Download objects

2.7.1 Overview

This topic is the overview of object download in OSS Java SDK.

OSS Java SDK provides the following download methods:

- *Streaming download*
- *Download to your local file*
- *Range download*
- *Resumable download*
- *Conditional download*

You can view the download progress shown in the *download progress bar*.

2.7.2 Streaming download

If you have a large object to download or it is time-consuming to download the entire object at a time, you can use streaming download. Streaming download enables you to download part of the object each time until you have downloaded the entire object.

If you do not close OssObject after it is used, connection leaks may occur. Consequently, no connection is available and the program cannot run properly. Run the following code to close OssObject:

```
OSSObject ossObject = ossClient.getObject(bucketName, objectName);
ossObject.close();
```

The following code is used for streaming downloads:

```
// This example uses the endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
```

```
// ossObject includes the bucket name, key (object name), object meta
 (meta information), and InputStream.
OSSObject ossObject = ossClient.getObject(bucketName, objectName);

// Read the object content.
System.out.println("Object content:");
BufferedReader reader = new BufferedReader(new InputStreamReader(
ossObject.getObjectContent()));
while (true) {
    String line = reader.readLine();
    if (line == null) break;

    System.out.println("\n" + line);
}
// If you do not close the reader after the data is read, connection
 leaks may occur. Consequently, no connection is available and the
program cannot run properly.
reader.close();

//Close your OSSClient.
ossClient.shutdown();
```

For the complete code of streaming download, see *GitHub*.

2.7.3 Download to your local file

This topic describes how to download an object from OSS to a local file.

You can run the following code to download a specified object from OSSClient to your local file:

```
//This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Download an object to your local file. If the name of the object
is the same as that of your local file, the object will replace the
local file. If the name of the object is different from that of your
local file, the object will be downloaded and a new file will be added
 to your local device.
ossClient.getObject(new GetObjectRequest(bucketName, objectName), new
File("<yourLocalFile>"));

//Close your OSSClient.
```

```
ossClient.shutdown();
```

2.7.4 Range download

If you only need part of the data in an object, you can use range downloads to download specified content.

Run the following code for range download:

```
//This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

GetObjectRequest getObjectRequest = new GetObjectRequest(bucketName,
objectName);
// Obtain data between the starting byte value of 0 and the maximum
byte value of 1,000 inclusive (that is, a total of 1,001 bytes). If
the specified value range is invalid, for example, the specified range
 includes a negative number, or the specified value is larger than the
 object size, the entire object is downloaded.
getObjectRequest.setRange(0, 1000);

// Start range download.
OSSObject ossObject = ossClient.getObject(getObjectRequest);

// Read data.
byte[] buf = new byte[1024];
InputStream in = ossObject.getObjectContent();
for (int n = 0; n ! = -1; ) {
    n = in.read(buf, 0, buf.length);
}

// If you do not close the reader after the data is read, connection
 leaks may occur. Consequently, no connections are available and an
exception occurs.
in.close();

//Close your OSSClient.
ossClient.shutdown();
```

All data cannot be read at one time when it is streamed. To stream a 64 KB block of data, use the following code to read the data multiple times until the entire object is downloaded. For more information, see *InputStream.read*.

```
byte[] buf = new byte[1024];
```

```
InputStream in = ossObject.getObjectContent();
for (int n = 0; n ! = -1; ) {
    n = in.read(buf, 0, buf.length);
}
in.close();
```

2.7.5 Resumable download

Large-sized objects may fail to be downloaded because the network is unstable or the Java program exits abnormally and the entire object needs to be downloaded again. However, the object may still fail to be downloaded after multiple attempts. Therefore, OSS provides resumable download.

To enable resumable download, you need to split the object you want to download into multiple parts and download them separately. After you have downloaded all parts, these parts will be combined into a complete object.

You can use ossClient.downloadFile to enable resumable download. DownloadFileRequest contains the following parameters:

Parameter	Description	Required or optional	Default value	Configuration methods
bucket	Specifies the name of a bucket.	Required	None	Constructor
key	Specifies the name of an object.	Required	None	Constructor
downloadFile	Specifies a local file. You can download an object to the local file.	Optional	Name of the object	Constructor or setDownloadFile
partSize	Specifies the size of a part. The value can be between 1 byte to 5 GB.	Optional	Size of an object/10000	setPartSize

Parameter	Description	Required or optional	Default value	Configuration methods
taskNum	Specifies the number of parts you need to download simultaneously.	Optional	1	setTaskNum
enableCheckpoint	Specifies whether resumable download is enabled.	Optional	Disable	setEnableCheckpoint
checkpointFile	Specifies the file to record the results of partial downloads. You need to configure this parameter to enable resumable download. This file stores information about the download progress. If you fail to download a part, the download will continue based on the recorded progress. After the object is downloaded to your local device, this file will be deleted.	Optional	downloadFile.ucp (share the same directory with DownloadFile)	setCheckpointFile

Run the following code for resumable download:

```
//This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
//It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Send a request to download 10 tasks simultaneously and start
resumable download.
DownloadFileRequest downloadFileRequest = new DownloadFileRequest(
bucketName, objectName);
downloadFileRequest.setDownloadFile("<yourDownloadFile>");
downloadFileRequest.setPartSize(1 * 1024 * 1024);
downloadFileRequest.setTaskNum(10);
downloadFileRequest.setEnableCheckpoint(true);
downloadFileRequest.setCheckpointFile("<yourCheckpointFile>");

// Download the object to your local file.
DownloadFileResult downloadRes = ossClient.downloadFile(downloadFi
leRequest);
// After the download succeeds, object meta is returned.
downloadRes.getObjectMetadata();

//Close your OSSClient.
ossClient.shutdown();
```

2.7.6 Conditional download

You can configure one or more conditions for downloads. If the specified conditions are met, the object can be downloaded. If the specified conditions are not met, an error code is returned and the object cannot be downloaded. You can configure the following conditions:

Parameter	Description
If-Modified-Since	If the specified time is earlier than the time the object is modified, the object can be downloaded. Otherwise, an error (304 Not modified) is returned.

Parameter	Description
If-Unmodified-Since	If the specified time is later than or equal to the time the object is modified , the object can be downloaded. Otherwise, an error (412 Precondition failed) is returned.
If-Match	If the specified ETag matches that of an object, the object can be downloaded. Otherwise, an error (412 Precondition failed) is returned.
If-None-Match	If the specified ETag does not match that of an object, the object can be downloaded. Otherwise, an error (304 Not modified) is returned.

Both If-Modified-Since and If-Unmodified-Since can exist at the same time as object download conditions. Both If-Match and If-None-Match can exist at the same time as object download conditions.

ossClient.getObjectMeta can be used to obtain the ETag.

Run the following code for conditional download:

```
//This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

GetObjectRequest request = new GetObjectRequest(bucketName, objectName
);
// Configure conditions.
request.setModifiedSinceConstraint(new Date());

// Download an object to a local file.
ossClient.getObject(request, new File("<yourLocalFile>"));

//Close your OSSClient.
ossClient.shutdown();
```

Conditional download supports ossClient.getObject and ossClient.downloadFile.

2.7.7 Download progress bars

You can use progress bars to indicate the upload or download progress.

The following code is used as an example to describe how to view progress information with ossClient.getObject:

```
static class GetObjectProgressListener implements ProgressListener {
        private long bytesRead = 0;
        private long totalBytes = -1;
        private boolean succeed = false;
        @Override
        public void progressChanged(ProgressEvent progressEvent) {
            long bytes = progressEvent.getBytes();
            ProgressEventType eventType = progressEvent.getEventType
();
            switch (eventType) {
            case TRANSFER_STARTED_EVENT:
                System.out.println("Start to download......");
                break;
            case RESPONSE_CONTENT_LENGTH_EVENT:
                this.totalBytes = bytes;
                System.out.println(this.totalBytes + " bytes in total
will be downloaded to a local file");
                break;
            case RESPONSE_BYTE_TRANSFER_EVENT:
                this.bytesRead += bytes;
                if (this.totalBytes ! = -1) {
                    int percent = (int)(this.bytesRead * 100.0 / this.
totalBytes);
                    System.out.println(bytes + " bytes have been read
at this time, download progress: " +
                            percent + "%(" + this.bytesRead + "/" +
this.totalBytes + ")");
                } else {
                    System.out.println(bytes + " bytes have been read
at this time, download ratio: unknown" +
                            "(" + this.bytesRead + "/...)");
                }
                break;
            case TRANSFER_COMPLETED_EVENT:
                this.succeed = true;
                System.out.println("Succeed to download, " + this.
bytesRead + " bytes have been transferred in total");
                break;
            case TRANSFER_FAILED_EVENT:
                System.out.println("Failed to download, " + this.
bytesRead + " bytes have been transferred");
                break;
            default:
                break;
            }
        }
        public boolean isSucceed() {
            return succeed;
        }
    }
```

```
public static void main(String[] args) {
    // This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
    String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
    // It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
    String accessKeyId = "<yourAccessKeyId>";
    String accessKeySecret = "<yourAccessKeySecret>";
    String bucketName = "<yourBucketName>";
    String objectName = "<yourObjectName>";
    OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);
    try {
            // Download with a progress bar displayed.
            ossClient.getObject(new GetObjectRequest(bucketName,
objectName).
                    <GetObjectRequest>withProgressListener(new
GetObjectProgressListener()),
                    new File("<yourLocalFile>"));
    } catch (Exception e) {
        e.printStackTrace();
    }
    // Close your OSSClient instance.
    ossClient.shutdown();
}
```

You can view progress information with ossClient.putObject, ossClient.getObject , or ossClient.uploadPart. However, you cannot view progress information with ossClient.uploadFile or ossClient.downloadFile.

For the complete code of download progress bar, see *GitHub*.

2.8 Manage objects

2.8.1 Overview

This topic is the overview of object management in OSS Java SDK.

You can perform the following operations on objects in a bucket with APIs:

- *Determine whether an object exists*
- *Manage ACL for an object*
- *Manage Object Meta*
- *Convert the storage class of an object*
- *List objects*
- *Delete an object*
- *Copy an object*

- *Restore an archive object*
- *Manage a symbolic link*

2.8.2 Determine whether a specified object exists

This topic describes how to determine whether a specified object exists

Run the following code to determine whether a specified object exists:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Determine whether the specified object exists. The doesObjectExist
 includes a parameter: isOnlyInOSS. If the value of this parameter
 is true, 302 Redirect or OSS mirroring is ignored. Otherwise, 302
Redirect or OSS mirroring is considered.

boolean found = ossClient.doesObjectExist("<yourBucketName>", "<
yourObjectName>");
System.out.println(found);

// Close your OSSClient.
ossClient.shutdown();
```

2.8.3 Manage ACL for an object

This topic describes how to manage ACL for an object.

The following table describes the permissions included in the Access Control List (ACL) for an object.

Permission	Description	Value
default	The ACL of an object is the same with that of its bucket.	CannedAccessControlList. Default
Private	Only the object owner and authorized users can read and write the object.	CannedAccessControlList .Private

Permission	Description	Value
Public read	Only the object owner and authorized users can read and write the object. Other users can only read the object. Authorize this permission with caution.	CannedAccessControlList.PublicRead
Public read-write	All users can read and write the object. Authorize this permission with caution.	CannedAccessControlList.PublicReadWrite

The ACL of objects take precedence over that of buckets. For example, if the ACL of a bucket is private, while the object ACL is public read-write, all users can read and write the object. If an object is not configured with an ACL, its ACL is the same as that of its bucket by default.

Configure an ACL for an object

You can run the following code to configure an ACL for an object:

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Configure the ACL for the object to public read.
ossClient.setObjectAcl("<yourBucketName>", "<yourObjectName>",
CannedAccessControlList.PublicRead);

// Close your OSSClient.
ossClient.shutdown();
```

Obtain the ACL for an object

You can run the following code to obtain the ACL for an object:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtain the ACL for an object.
ObjectAcl objectAcl = ossClient.getObjectAcl("<yourBucketName>", "<
yourObjectName>");
System.out.println(objectAcl.getPermission().toString());

// Close your OSSClient.
ossClient.shutdown();
```

2.8.4 Manage Object Meta

Object Meta includes HTTP headers and user-defined Object Meta.

Note:

For more information, see *Object Meta* in OSS Developer Guide.

Configure HTTP headers

- **Configure HTTP headers**

  Run the following code to configure HTTP headers:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String content = "Hello OSS";

// Create Object Meta. You can configure HTTP headers with Object
Meta.
ObjectMetadata meta = new ObjectMetadata();

String md5 = BinaryUtil.toBase64String(BinaryUtil.calculateMd5(
content.getBytes()));
// Start object content verification with MD5 values. After the
verification is started, the MD5 value calculated by OSS is compared
 with that of an object. If the two values are different, an error
occurs.
meta.setContentMD5(md5);
```

```
// Specify the type of content you want to upload. The content type
determines the form and format in which your browser reads an object
. The object extension is used as the content type if the content
type is not specified. If no object extension exists, the default
value application/octet-stream is used.
meta.setContentType("text/plain");
// Specify the name of the content when you download an object.
meta.setContentDisposition("attachment; filename=\"DownloadFilename
\"");
// Configure the length for an object. If the actual object length
 is greater than the configured length, only the configured length
 will be uploaded. If the configured length is greater than the
actual object length, the entire object will be uploaded.
meta.setContentLength(content.length());
// Configure the cache action of a webpage when an object is
downloaded.
meta.setCacheControl("Download Action");
// Configure the expiration time (GMT is used).
meta.setExpirationTime(DateUtil.parseIso8601Date("2022-10-12T00:00:
00.000Z"));
// Configure the encoding format when an object is downloaded.
meta.setContentEncoding("utf-8");
// Configure the HTTP header.
meta.setHeader("<yourHeader>", "<yourHeaderValue>");
// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId,
accessKeySecret);

// Upload the object.
ossClient.putObject("<yourBucketName>", "<yourObjectName>", new
ByteArrayInputStream(content.getBytes()), meta);

// Close your OSSClient.
ossClient.shutdown();
```

For more information about HTTP headers, see *RFC2616*.

- **Configure user-defined Object Meta**

  You can define user-defined Object Meta for object descriptions.

  Run the following code to configure user-defined Object Meta:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String content = "Hello OSS";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create Object Meta.
```

```
ObjectMetadata meta = new ObjectMetadata();

// Set property to property-value for user-defined Object Meta. We
recommend that you use Base64 encoding method.
meta.addUserMetadata("property", "property-value");

// Upload the object.
ossClient.putObject("<yourBucketName>", "<yourObjectName>", new
ByteArrayInputStream(content.getBytes()), meta);

// Obtain Object Meta.
ossClient.getObjectMetadata("<yourBucketName>", "<yourObjectName>");

// Close your OSSClient.
ossClient.shutdown();
```

When an object is downloaded, its Object Meta is also downloaded. An object can contain multiple pieces of Object Meta, with the total maximum size smaller than 8 KB.

Modify Object Meta

Run the following code to modify Object Meta:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String sourceBucketName = "<yourSourceBucketName>";
String sourceObjectName = "<yourSourceObjectName>";
String destinationBucketName = "<yourDestinationBucketName>";
String destinationObjectName = "<yourDestinationObjectName>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId,accessKeyS
ecret);

// Replace the source object with the target object. Call ossClient.
copyObject to modify Object Meta.
CopyObjectRequest request = new CopyObjectRequest(sourceBucketName,
sourceObjectName, destinationBucketName, destinationObjectName);

ObjectMetadata meta = new ObjectMetadata();
// Specify the type of content you want to upload. The content type
 determines the form and format that your browser uses to read an
 object. The object extension is used for the content type if the
content type is not specified. If no object extension exists, the
default value application/octet-stream is used.
meta.setContentType("text/plain");
// Specify the name of the content when you download an object.
meta.setContentDisposition("Download File Name");
// Configure the cache action of a webpage when an object is
downloaded.
```

```
meta.setCacheControl("Download Action");
// Configure the expiration time (GMT is used).
meta.setExpirationTime(DateUtil.parseIso8601Date("2022-10-12T00:00:00.
000Z"));
// Configure the encoding type when the object is downloaded.
meta.setContentEncoding("utf-8");
// Configure the HTTP header.
meta.setHeader("<yourHeader>", "<yourHeaderValue>");
// Set property to property-value for user-defined Object Meta.
meta.addUserMetadata("property", "property-value");
request.setNewObjectMetadata(meta);

//Modify Object Meta.
ossClient.copyObject(request);

// Close your OSSClient.
ossClient.shutdown();
```

Obtain Object Meta

The following table lists the methods you can use to obtain Object Meta.

Method	Description	Advantage
ossClient.getSimplifiedObjectMeta	**Obtains ETag, Size, and LastModified (the latest time an object is modified.) for the object.**	**More lightweight and faster**
ossClient.getObjectMetadata	**Obtains all metadata.**	**None**

Run the following code to obtain Object Meta:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Obtain partial Object Meta.
SimplifiedObjectMeta objectMeta = ossClient.getSimplifiedObjectMeta("<
yourBucketName>", "<yourObjectName>");
System.out.println(objectMeta.getSize());
System.out.println(objectMeta.getETag());
System.out.println(objectMeta.getLastModified());

// Obtain all Object Meta.
```

```
ObjectMetadata metadata = ossClient.getObjectMetadata("<yourBucketName
>", "<yourObjectName>");
System.out.println(metadata.getContentType());
System.out.println(metadata.getLastModified());
System.out.println(metadata.getExpirationTime());

// Close your OSSClient.
ossClient.shutdown();
```

2.8.5 Convert the storage class of an object

OSS provides the following storage classes to cover various data storage scenarios from hot data to cold data: Standard, Infrequent Access (IA), and Archive. This topic describes how to convert the storage class of an object.

For more information about these storage classes, see *#unique_43* and *#unique_77* in the OSS Developer Guide.

The following code provides an example of how to convert the storage class of an object.

- **The following code provides an example of how to convert the storage class of an object from Standard or IA to Archive:**

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram
.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// This example requires a bucket containing an object of the
Standard or IA storage class.
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create CopyObjectRequest.
CopyObjectRequest request = new CopyObjectRequest(bucketName,
objectName, bucketName, objectName) ;

// Create ObjectMetadata.
ObjectMetadata objectMetadata = new ObjectMetadata();

// Encapsulate the header. Set the storage class to Archive.
objectMetadata.setHeader("x-oss-storage-class", StorageClass.Archive
);
```

```
request.setNewObjectMetadata(objectMetadata);

// Modify the storage class of the object.
CopyObjectResult result = ossClient.copyObject(request);

// Close your OSSClient.
ossClient.shutdown();
```

- **The following code provides an example of how to convert the storage class of an object from Archive to IA:**

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram
.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// This example requires a bucket containing an object of the
Archive storage class.
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtain the object metadata.
ObjectMetadata objectMetadata = ossClient.getObjectMetadata(
bucketName, objectName);

// Check whether the storage class of the source object is Archive
. If the storage class of the source object is Archive, you must
restore the object before you can modify the storage class.
StorageClass storageClass = objectMetadata.getObjectStorageClass();
System.out.println("storage type:" + storageClass);
if (storageClass == StorageClass.Archive) {
    // Restore the object.
    ossClient.restoreObject(bucketName, objectName);

    // Wait until the object is restored.
    do {
        Thread.sleep(1000);
        objectMetadata = ossClient.getObjectMetadata(bucketName,
objectName);
    } while (! objectMetadata.isRestoreCompleted());
}

 // Create CopyObjectRequest.
CopyObjectRequest request = new CopyObjectRequest(bucketName,
objectName, bucketName, objectName) ;

// Create ObjectMetadata.
objectMetadata = new ObjectMetadata();

// Encapsulate the header. Set the storage class to IA.
```

```
objectMetadata.setHeader("x-oss-storage-class", StorageClass.IA);
request.setNewObjectMetadata(objectMetadata);

// Modify the storage class of the object.
CopyObjectResult result = ossClient.copyObject(request);

// Close your OSSClient.
ossClient.shutdown();
```

For more information about the storage classes and storage fees, see the *#unique_78/unique_78_Connect_42_section_jmi_2z4_hka* section in Billing items.

2.8.6 List objects

This topic describes how to list objects.

Objects are listed alphabetically. You can use ossClient.listObjects to list objects in a bucket. The following parameters can be used for ListObjects:

- **ObjectListing listObjects(String bucketName): lists objects in a bucket. A maximum of 100 objects can be listed.**
- **ObjectListing listObjects(String bucketName, String prefix): lists objects with the specified prefix in a bucket. A maximum of 100 objects can be listed.**
- **ObjectListing listObjects(ListObjectsRequest listObjectsRequest): provides multiple filtering functions to flexibly query objects.**

The following table describes parameters for ObjectListing.

Parameter	Description	Configuration method
objectSummaries	Specifies the returned Object Meta.	List<OSSObjectSummary> getObjectSummaries()
prefix	Specifies the prefix you configure to list required objects.	String getPrefix()
delimiter	Specifies a delimiter used to group objects.	String getDelimiter()
marker	Specifies the initial object in the list.	String getMarker()
maxKeys	Specifies the maximum number of objects that can be listed.	int getMaxKeys()
nextMarker	Specifies the initial position for the next object.	String getNextMarker()

Parameter	Description	Configuration method
isTruncated	Specifies whether the listed objects are truncated. · If the objects are all listed without truncation, a false value is returned. · If the listed objects are truncated, a true value is returned.	boolean isTruncated()
commonPrefixes	Specifies a set of objects whose names share a prefix and end with a forward slash (/) delimiter.	List<String> getCommonPrefixes()
encodingType	Specifies the encoding type used in the response.	String getEncodingType()

Simple list

Run the following code to list objects in a specified bucket. A maximum of 100 objects can be listed by default.

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String KeyPrefix = "<yourKeyPrefix>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List objects. If you do not configure KeyPrifex, all objects in
 the bucket are listed. If you configure KeyPrifex, objects with a
specified prefix in the bucket are listed.
ObjectListing objectListing = ossClient.listObjects(bucketName,
KeyPrefix);
List<OSSObjectSummary> sums = objectListing.getObjectSummaries();
for (OSSObjectSummary s : sums) {
    System.out.println("\t" + s.getKey());
}
```

```
// Close your OSSClient.
ossClient.shutdown();
```

List objects with ListObjectsRequest

You can configure parameters for ListObjectsRequest to flexibly query objects. The following table describes parameters for ListObjectsRequest.

Parameter	Description	Configuration method
prefix	Specifies the prefix that must be included in the returned objects.	setPrefix(String prefix)
delimiter	Specifies a delimiter of a forward slash (/) used to group objects based on their names. The substring between the specified prefix and the first occurrence of the delimiter of a forward slash (/) is commonPrefixes.	setDelimiter(String delimiter)
marker	Lists the objects after the value of marker.	setMarker(String marker)
maxKeys	Specifies the maximum number of objects that can be listed. The default value is 100. The maximum value is 1,000.	setMaxKeys(Integer maxKeys)
encodingType	Specifies the encoding type of an object name in the response body. Only the URL encoding type is supported.	setEncodingType(String encodingType)

- **List a specified number of objects**

  Run the following code to list a specified number of objects:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Configure the maximum number.
final int maxKeys = 200;
// List objects.
ObjectListing objectListing = ossClient.listObjects(new ListObject
sRequest(bucketName).withMaxKeys(maxKeys));
List<OSSObjectSummary> sums = objectListing.getObjectSummaries();
for (OSSObjectSummary s : sums) {
    System.out.println("\t" + s.getKey());
}

// Close your OSSClient.
ossClient.shutdown();
```

- **List objects by a specified prefix**

  Run the following code to list objects with a specified prefix:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Specify a prefix.
final String keyPrefix = "<yourkeyPrefix>";

// List objects with a specified prefix. By default, a maximum of
100 objects can be listed.
ObjectListing objectListing = ossClient.listObjects(new ListObject
sRequest(bucketName).withPrefix(keyPrefix));
List<OSSObjectSummary> sums = objectListing.getObjectSummaries();
for (OSSObjectSummary s : sums) {
    System.out.println("\t" + s.getKey());
}

// Close your OSSClient.
ossClient.shutdown();
```

```
```

- **List objects by marker**

  The marker parameter indicates the name of an object from which the listing begins. Run the following code to specify an object (specified with marker) after which the listing begins:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
final String marker = "<yourMarker>";

// List objects placed after the object (specified with marker). By
default, a maximum of 100 objects can be listed.
ObjectListing objectListing = ossClient.listObjects(new ListObject
sRequest(bucketName).withMarker(marker));
List<OSSObjectSummary> sums = objectListing.getObjectSummaries();
for (OSSObjectSummary s : sums) {
    System.out.println("\t" + s.getKey());
}

// Close your OSSClient.
ossClient.shutdown();
```

- **List all objects on one or more pages**

  Use the following code to list all objects on one or more pages. The maximum number of objects listed on each page is specified with maxKeys.

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

final int maxKeys = 200;
```

```
String nextMarker = null;
ObjectListing objectListing;

do {
    objectListing = ossClient.listObjects(new ListObjectsRequest(
bucketName).withMarker(nextMarker).withMaxKeys(maxKeys));

    List<OSSObjectSummary> sums = objectListing.getObjectSummaries
();
    for (OSSObjectSummary s : sums) {
        System.out.println("\t" + s.getKey());
    }

    nextMarker = objectListing.getNextMarker();

} while (objectListing.isTruncated());

// Close your OSSClient.
ossClient.shutdown();
```

- **List objects by a specified prefix on one or more pages**

  Run the following code to list objects by a specified prefix on one or more pages:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// A maximum of 200 objects can be listed on each page.
final int maxKeys = 200;
final String keyPrefix = "<yourkeyPrefix>";
String nextMarker = "<yourNextMarker>";
ObjectListing objectListing;

do {
    objectListing = ossClient.listObjects(new ListObjectsRequest(
bucketName).
            withPrefix(keyPrefix).withMarker(nextMarker).withMaxKeys
(maxKeys));

    List<OSSObjectSummary> sums = objectListing.getObjectSummaries
();
    for (OSSObjectSummary s : sums) {
        System.out.println("\t" + s.getKey());
    }

    nextMarker = objectListing.getNextMarker();

} while (objectListing.isTruncated());
```

```
// Close your OSSClient.
ossClient.shutdown();
```

- **Specify the name of an object for encoding.**

  You must encode the name of an object if the name contains the following special characters. OSS supports encoding with URL.

  - **Single quotation marks (‘)**
  - **Double quotations marks (“)**
  - **ampersands (&)**
  - **angle brackets** (< >)
  - **Pause marker (、)**
  - **Chinese characters**

  Run the following code to encode the name of a specified object:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

final int maxKeys = 200;
final String keyPrefix = "<yourkeyPrefix>";
String nextMarker = "<yourNextMarker>";
ObjectListing objectListing;

do {
    ListObjectsRequest listObjectsRequest = new ListObjectsRequest(
bucketName);
    listObjectsRequest.setPrefix(keyPrefix);
    listObjectsRequest.setMaxKeys(maxKeys);
    listObjectsRequest.setMarker(nextMarker);

    // Specify the name of an object for encoding.
    listObjectsRequest.setEncodingType("url");

    objectListing = ossClient.listObjects(listObjectsRequest);

    // Decode the object.
    for (OSSObjectSummary objectSummary: objectListing.getObjectS
ummaries()) {
        System.out.println("Key:" + URLDecoder.decode(objectSummary.
getKey(), "UTF-8"));
    }
```

```
    // Decode the commonPrefixes parameter.
    for (String commonPrefixes: objectListing.getCommonPrefixes()) {
        System.out.println("CommonPrefixes:" + URLDecoder.decode(
commonPrefixes, "UTF-8"));
    }

    // Decode the nextMarker parameter.
    if (objectListing.getNextMarker() ! = null) {
        nextMarker = URLDecoder.decode(objectListing.getNextMarker
(), "UTF-8");
    }
} while (objectListing.isTruncated());

// Close your OSSClient.
ossClient.shutdown();
```

Folder simulation

OSS does not use folders. All elements are stored as objects. To simulate a folder on the OSS console, you actually create a 0 MB object whose name ends with a forward slash (/). This object can be uploaded and downloaded. By default, the OSS console displays an object whose name ends with a forward slash (/) as a folder.

The delimiter and prefix parameters can be used to simulate folder functions.

- **Prefix: specifies the prefix as the name of a folder. The folder is used to list all files (in the folder) and subfolders (directories in the folder) that start with this prefix. These files and subfolders are included in the Objects list.**
- **Delimiter: If the delimiter is specified as a forward slash (/), only the files and subfolders (directories) are displayed. Subfolders (directories) are included in the CommonPrefixes list, while the files and folders in subfolders are not displayed.**

For more information about folders, see *Folder simulation*. For the complete code of creating an object, see *GitHub*.

Assume that the following objects are stored in a bucket: oss.jpg, fun/test.jpg, fun /movie/001.avi, and fun/movie/007.avi. Forward slashes (/) are used as delimiters for folders. The subsequent examples show how to simulate folder functions.

- **List all objects in a bucket**

  Run the following code to list all objects in a bucket:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create ListObjectsRequest.
ListObjectsRequest listObjectsRequest = new ListObjectsRequest(
bucketName);

// List objects.
ObjectListing listing = ossClient.listObjects(listObjectsRequest);

// Print all objects.
System.out.println("Objects:");
for (OSSObjectSummary objectSummary : listing.getObjectSummaries())
 {
    System.out.println(objectSummary.getKey());
}

// Print all commonPrefixes.
System.out.println("CommonPrefixes:");
for (String commonPrefix : listing.getCommonPrefixes()) {
    System.out.println(commonPrefix);
}

// Disable the OSSClient.
// Close your OSSClient.
```

The returned results are as follows:

```
Objects:
fun/movie/001.avi
fun/movie/007.avi
fun/test.jpg
oss.jpg

CommonPrefixes:
```

- **List all objects in a specified folder**

  Run the following code to list all objects in a specified folder:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
```

```
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create ListObjectsRequest.
ListObjectsRequest listObjectsRequest = new ListObjectsRequest(
bucketName);
// Configure the Prefix parameter to obtain all objects in the fun
folder.
listObjectsRequest.setPrefix("fun/");

// Recursively list all objects in the fun folder.
ObjectListing listing = ossClient.listObjects(listObjectsRequest);

// Print all objects.
System.out.println("Objects:");
for (OSSObjectSummary objectSummary : listing.getObjectSummaries())
 {
    System.out.println(objectSummary.getKey());
}

// Print all commonPrefixes.
System.out.println("\nCommonPrefixes:");
for (String commonPrefix : listing.getCommonPrefixes()) {
    System.out.println(commonPrefix);
}

// Close your OSSClient.
ossClient.shutdown();
```

The returned results are as follows:

```
Objects:
fun/movie/001.avi
fun/movie/007.avi
fun/test.jpg

CommonPrefixes:
```

- **List objects and subfolders in a folder**

  Use the following code to list objects and subfolders in a folder:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
```

```
// Create ListObjectsRequest.
ListObjectsRequest listObjectsRequest = new ListObjectsRequest(
bucketName);

// Set forward slashes (/) as the delimiter for folders.
listObjectsRequest.setDelimiter("/");

// List all objects and folders in the fun folder.
listObjectsRequest.setPrefix("fun/");

ObjectListing listing = ossClient.listObjects(listObjectsRequest);

// Print all objects.
System.out.println("Objects:");
// List objects in the fun folder in a objectSummaries list.
for (OSSObjectSummary objectSummary : listing.getObjectSummaries())
 {
    System.out.println(objectSummary.getKey());
}

// Print all commonPrefixes.
System.out.println("\nCommonPrefixes:");
// commonPrefixs lists all subfolders in the fun folder. The fun/
movie/001.avi and fun/movie/007.avi objects are not listed, because
 they are objects in the movie folder (a subfolder of the fun folder
).
for (String commonPrefix : listing.getCommonPrefixes()) {
    System.out.println(commonPrefix);
}

// Close your OSSClient.
ossClient.shutdown();
```

The returned results are as follows:

```
Objects:
fun/test.jpg

CommonPrefixes:
fun/movie/
```

2.8.7 Delete objects

This topic describes how to delete objects.

Warning:

You cannot recover deleted objects. Exercise caution when you delete objects.

Delete one object

The following code provides an example on how to delete an object:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Delete the object. To delete a folder, set ObjectName to the folder
 name. If the folder is not empty, you must delete all objects in the
folder before you can delete the folder.
ossClient.deleteObject(bucketName, objectName);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

Delete multiple objects

A maximum of 1,000 objects can be deleted each time. Objects can be deleted in the following two modes:

- **verbose: returns a list of deleted objects. The default mode is verbose.**
- **quiet: returns a list of objects you failed to delete.**

The following table describes the parameters for DeleteObjectsRequest.

Parameter	Description	Configuration method
Keys	**Specifies the objects you want to delete.**	**setKeys(List<String>)**
quiet	**Specifies the return mode . True indicates quiet. False indicates verbose. Default value: false.**	**setQuiet(boolean)**
encodingType	**Specifies the mode for encoding returned object names. Only URL encoding is supported.**	**setEncodingType(String)**

The following table describes the parameters for DeleteObjectsResult.

Parameter	Description	Configuration method
deletedObjects	The results of the operation. The list of deleted objects is returned when the verbose mode is used. The list of objects that fail to be deleted is returned when the quiet mode is used.	List<String> getDeletedObjects()
encodingType	The method for encoding the objects names listed in the deletedObjects field. When the parameter is empty, the object names are not encoded.	getEncodingType()

The following code provides an example on how to delete multiple objects at a time:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Delete the objects. The key is the ObjectName, which indicates the
 complete path of the object you want to delete from OSS, and must
include the file extension of the object. For example, set ObjectName
to abc/efg/123.jpg.
List<String> keys = new ArrayList<String>();
keys.add("key0");
keys.add("key1");
keys.add("key2");

DeleteObjectsResult deleteObjectsResult = ossClient.deleteObjects(new
DeleteObjectsRequest(bucketName).withKeys(keys));
List<String> deletedObjects = deleteObjectsResult.getDeletedObjects();

// Shut down the OSSClient instance.
ossClient.shutdown();
```

For the complete code used to delete multiple objects, visit *GitHub*.

Delete objects whose names contain a specified prefix

The following code provides an example on how to delete objects whose names contain a specified prefix:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Specify a prefix.
final String prefix = "<yourkeyPrefix>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// List all objects whose names contain the specified prefix and
delete them.
String nextMarker = null;
ObjectListing objectListing = null;
do {
    ListObjectsRequest listObjectsRequest = new ListObjectsRequest(
bucketName)
            .withPrefix(prefix)
            .withMarker(nextMarker);

    objectListing = ossClient.listObjects(listObjectsRequest);
    if (objectListing.getObjectSummaries().size() > 0) {
        List<String> keys = new ArrayList<String>();
        for (OSSObjectSummary s : objectListing.getObjectSummaries())
 {
            System.out.println("key name: " + s.getKey());
            keys.add(s.getKey());
        }
        DeleteObjectsRequest deleteObjectsRequest = new DeleteObje
ctsRequest(bucketName).withKeys(keys);
        ossClient.deleteObjects(deleteObjectsRequest);
    }

    nextMarker = objectListing.getNextMarker();
} while (objectListing.isTruncated());

// Shut down the OSSClient instance.
```

```
ossClient.shutdown();
```

2.8.8 Copy objects

OSS allows you to copy objects of any sizes.

Copy a small-sized object

You can use ossClient.copyObject to copy an object smaller than 1 GB from a bucket (source bucket) to another bucket (target bucket) in the same region. ossClient.copyObject supports the following configuration methods.

Configuration method	Description
CopyObjectResult copyObject(String sourceBucketName, String sourceKey, String destinationBucketName, String destinationKey)	Allows you to specify a source bucket, source object, target bucket, and target object. After the copy operation is completed, the target object content and its Object Meta are the same as those of the source object. This is called simple copy.
CopyObjectResult copyObject(CopyObjectRequest copyObjectRequest)	Allows you to specify target Object Meta and copy conditions. If the URL of the source bucket is the same as that of the target object, the Object Meta of the source object is replaced with that of the target object.

The following table describes the parameters you can configure for CopyObject Request.

Parameter	Description	Configuration method
sourceBucketName	Specifies the name of the source bucket.	setSourceBucketName(String sourceBucketName)
sourceKey	Specifies the name of a source object.	setSourceKey(String sourceKey)
destinationBucketName	Specifies the name of a target bucket.	setDestinationBucket Name(String destinatio nBucketName)
destinationKey	Specifies the name of a target object.	setDestinationKey(String destinationKey)

Parameter	Description	Configuration method
newObjectMetadata	Specifies the target Object Meta.	setNewObjectMetadata (ObjectMetadata newObjectMetadata)
matchingETagConstraints	Restrictions on copies. Specifies the conditions for object copy. If ETag of a source object is the same as what OSS calculates, the source object is copied. Otherwise, an error is returned.	setMatchingETagConstraints(List<String> matchingETagConstraints)
nonmatchingEtagConstraints	Specifies the conditions for object copy. If ETag of a source object is different from the ETag the user provides, the source object is copied. Otherwise, an error is returned.	setNonmatchingETagConstraints(List<String> nonmatchingEtagConstraints)
unmodifiedSinceConstraint	Specifies the conditions for object copy. If the specified time is later than or equal to the time an object is modified, the source object is copied. Otherwise, an error is returned.	setUnmodifiedSinceConstraint(Date unmodifiedSinceConstraint)
modifiedSinceConstraint	Specifies the conditions for object copy. If a source object is modified after the specified time, the source object is copied. Otherwise, an error is returned.	setModifiedSinceConstraint(Date modifiedSinceConstraint)

The following table describes the parameters you can configure for CopyObject Request.

Parameter	Description	Configuration method
ETag	Specifies the unique identifier of an object.	String getETag()
Lastmodified	Specifies the latest time an object is modified.	Date getLastModified()

Note:

Object copy rules are as follows:

- **Users have permissions to read and write the source object.**
- **Data cannot be copied across regions. For example, you cannot copy an object in a bucket from China East1 (Hangzhou) to China North1 (Qingdao).**
- **The object size is limited to a maximum of 1 GB.**

- **Simple copy**

  Run the following code for simple copy:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String sourceBucketName = "<yourSourceBucketName>";
String sourceObjectName = "<yourSourceObjectName>";
String destinationBucketName = "<yourDestinationBucketName>";
String destinationObjectName = "<yourDestinationObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Copy an object.
CopyObjectResult result = ossClient.copyObject(sourceBucketName,
sourceObjectName, destinationBucketName, destinationObjectName);
System.out.println("ETag: " + result.getETag() + " LastModified: "
 + result.getLastModified());

// Close your OSSClient.
ossClient.shutdown();
```

- **CopyObjectRequest-based copy**

  Run the following code for CopyObjectRequest-based copy:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String sourceBucketName = "<yourSourceBucketName>";
String sourceObjectName = "<yourSourceObjectName>";
String destinationBucketName = "<yourDestinationBucketName>";
String destinationObjectName = "<yourDestinationObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create a CopyObjectRequest object.
CopyObjectRequest copyObjectRequest = new CopyObjectRequest
(sourceBucketName, sourceObjectName, destinationBucketName,
destinationObjectName);

// Configure Object Meta for a new object.
ObjectMetadata meta = new ObjectMetadata();
meta.setContentType("text/html");
copyObjectRequest.setNewObjectMetadata(meta);

// Copy the object.
CopyObjectResult result = ossClient.copyObject(copyObjectRequest);
System.out.println("ETag: " + result.getETag() + " LastModified: "
 + result.getLastModified());

// Close your OSSClient.
ossClient.shutdown();
```

Copy a large-sized object

To copy an object larger than 1 GB, you need to use UploadPartCopy. In other words , you need to split the object into multiple parts and copy each of them simultaneo usly. To enable UploadPartCopy, perform the following steps:

1. **Use ossClient.initiateMultipartUpload to initiate an UploadPartCopy task.**
2. **Start UploadPartCopy with ossClient.uploadPartCopy. Aside from the last part, all parts must be larger than 100 KB.**

3. **Submit the UploadPartCopy task with ossClient.completeMultipartUpload.**

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String sourceBucketName = "<yourSourceBucketName>";
String sourceObjectName = "<yourSourceObjectName>";
String destinationBucketName = "<yourDestinationBucketName>";
String destinationObjectName = "<yourDestinationObjectName>";

ObjectMetadata objectMetadata = ossClient.getObjectMetadata(sourceBuck
etName, sourceObjectName);
// Obtain the size of the object you want to copy.
long contentLength = objectMetadata.getContentLength();

// Set the minimum size to 10 MB for each part.
long partSize = 1024 * 1024 * 10;

// Calculate the total number of parts.
int partCount = (int) (contentLength / partSize);
if (contentLength % partSize ! = 0) {
    partCount++;
}
System.out.println("total part count:" + partCount);

// Initiate an UploadPartCopy task. You can specify Object Meta for
the target object with InitiateMultipartUploadRequest.
InitiateMultipartUploadRequest initiateMultipartUploadRequest = new
 InitiateMultipartUploadRequest(destinationBucketName, destinatio
nObjectName);
InitiateMultipartUploadResult initiateMultipartUploadResult =
ossClient.initiateMultipartUpload(initiateMultipartUploadRequest);
String uploadId = initiateMultipartUploadResult.getUploadId();

// Start the UploadPartCopy task.
List<PartETag> partETags = new ArrayList<PartETag>();
for (int i = 0; i < partCount; i++) {
     // Calculate the size of each part.
    long skipBytes = partSize * i;
    long size = partSize < contentLength - skipBytes ? partSize :
contentLength - skipBytes;

    // Create UploadPartCopyRequest. You can specify conditions with
UploadPartCopyRequest.
    UploadPartCopyRequest uploadPartCopyRequest =
        new UploadPartCopyRequest(sourceBucketName, sourceObjectName,
destinationBucketName, destinationObjectName);
    uploadPartCopyRequest.setUploadId(uploadId);
    uploadPartCopyRequest.setPartSize(size);
    uploadPartCopyRequest.setBeginIndex(skipBytes);
    uploadPartCopyRequest.setPartNumber(i + 1);
    UploadPartCopyResult uploadPartCopyResult = ossClient.uploadPart
Copy(uploadPartCopyRequest);

    // Save the returned ETag of parts to partETags.
```

```
    partETags.add(uploadPartCopyResult.getPartETag());
}

// Submit the UploadPartCopy task.
CompleteMultipartUploadRequest completeMultipartUploadRequest = new
CompleteMultipartUploadRequest(
                    destinationBucketName, destinationObjectName,
uploadId, partETags);
ossClient.completeMultipartUpload(completeMultipartUploadRequest);

// Close your OSSClient.
ossClient.shutdown();
```

For more information about UploadPartCopy, see *UploadPartCopy*. For the complete code of UploadPartCopy, see *GitHub*.

2.8.9 Restore an archive object

This topic describes how to restore an archive object.

For the complete code of restoring an archive object, see *GitHub*.

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

1. **An archive object is in the frozen state.**
2. **After you submit it for restoration, the server restores the object. The object is in the restoring state.**
3. **You can read the object after it is restored. The restored state of the object lasts one day by default. You can prolong this period to a maximum of seven days. Once this period expires, the object returns to the frozen state.**

Run the following code to restore an object:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
```

```
ObjectMetadata objectMetadata = ossClient.getObjectMetadata(bucketName
, key);

// Verify whether the object is an archive object.// Verify that the
file is an archive file.
StorageClass storageClass = objectMetadata.getObjectStorageClass();
if (storageClass == StorageClass.Archive) {
    // Restore the object.
    ossClient.restoreObject(bucketName, objectName);

    // Wait until the object is restored.
    do {
        Thread.sleep(1000);
        objectMetadata = ossClient.getObjectMetadata(bucketName,
objectName);
    } while (! objectMetadata.isRestoreCompleted());
}

// Obtain the restored object.
OSSObject ossObject = ossClient.getObject(bucketName, objectName);
ossObject.getObjectContent().close();

// Close your OSSClient.
ossClient.shutdown();
```

For more information about the archive storage classes, see *Introduction to storage classes*.

2.8.10 Manage symbolic links

This topic describes how to create a symbolic link and obtain the name of the object to which the symbolic link points.

Create a symbolic link

A symbolic link is a special object that points to an object. It is similar to a shortcut used in Windows. You can use a symbolic link to customize object metadata.

The following code provides an example on how to create a symbolic link:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String symLink = "<yourSymLink>";
String destinationObjectName = "<yourDestinationObjectName>";

// Create an OSSClient instance.
```

```
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create metadata for the object to upload.
ObjectMetadata metadata = new ObjectMetadata();
metadata.setContentType("text/plain");
// Set the value of the property parameter for the user metadata to
property-value.
metadata.addUserMetadata("property", "property-value");

// Create a request to create the symbolic link.
CreateSymlinkRequest createSymlinkRequest = new CreateSymlinkRequest(
bucketName, symLink, destinationObjectName);

// Set the object metadata.
createSymlinkRequest.setMetadata(metadata);

// Create the symbolic link.
ossClient.createSymlink(createSymlinkRequest);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

For more information about how to create a symbolic link, see *PutSymlink*.

Obtain the name of the object to which the symbolic link points

To obtain a symbolic link, you must have the read permissions on it. The following code provides an example on how to obtain the name of the object to which the symbolic link points:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String symLink = "<yourSymLink>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtain the symbolic link.
OSSSymlink symbolicLink = ossClient.getSymlink(bucketName, symLink);
// Obtain the content of the object to which the symbolic link points.
System.out.println(symbolicLink.getSymlink());
System.out.println(symbolicLink.getTarget());
System.out.println(symbolicLink.getRequestId());

// Shut down the OSSClient instance.
ossClient.shutdown();
```

For more information about how to obtain a symbolic link, see *GetSymlink*.

2.9 Versioning

2.10 Object tagging

2.10.1 Configure object tagging

This topic describes how to configure object tagging.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Note:

- You can add tags to an object when uploading it or add tags for an uploaded object. If add tags to an object that already has tags, the original tags are overwritten. For more information about adding tags, see *#unique_87*.
- To add tags to an object, you must have the permission to call PutObjectTagging.
- The Last-Modified value of an object is not updated when its tags are changed.
- A tag can contain letters, numbers, spaces, and the following symbols: + - = . _ : /

Add tags to an object when uploading it

- Add tags to an object when uploading it by calling putObject.

  Run the following code to add tags to an object when uploading it by calling putObject:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify
an actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

Map<String, String> tags = new HashMap<String, String>();
```

```
tags.put("key1", "value1");
tags.put("key2", "value2");

// Configures the tags in the HTTP headers.
ObjectMetadata metadata = new ObjectMetadata();
metadata.setObjectTagging(tags);

// Uploads the object and add tags to it.
String content = "<yourtContent>";
ossClient.putObject(bucketName, objectName, new ByteArrayInputStream
(content.getBytes()), metadata);

// Closes the OSSClient.
ossClient.shutdown();
```

- **Add tags to an object when uploading it in the multipart upload method.**

  Run the following code to add tags to an object when uploading it in the multipart upload method.

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify
an actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";
String localFile = "<yourLocalFile>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

/* Step 1: Initializes a multipart upload task.
*/
// Configures the tags in the HTTP headers.
Map<String, String> tags = new HashMap<String, String>();
tags.put("key1", "value1");
tags.put("key2", "value2");

ObjectMetadata metadata = new ObjectMetadata();
metadata.setObjectTagging(tags);

// Sends an InitiateMultipartUploadRequest request and adds tags to
the object.
InitiateMultipartUploadRequest request = new InitiateMultipartUpl
oadRequest(bucketName, objectName);
InitiateMultipartUploadResult result = ossClient.initiateMu
ltipartUpload(request);
// Obtains the uploadId, which uniquely identifies the multipart
upload task. You can use it to cancel or query the multipart upload
task.
String uploadId = result.getUploadId();

/* Step 2: Uploads the parts.
*/
```

```
// The partEtags parameter is a set of PartTags. A PartETag is
composed of the ETag and number of the part.
List<PartETag> partETags =  new ArrayList<PartETag>();
// Calculates the number of parts that the object can be divided
into.
final long partSize = 1 * 1024 * 1024L;   // 1MB
final File sampleFile = new File("<localFile>");
long fileLength = sampleFile.length();
int partCount = (int) (fileLength / partSize);
if (fileLength % partSize != 0) {
    partCount++;
 }
// Uploads the parts one by one.
for (int i = 0; i < partCount; i++) {
    long startPos = i * partSize;
    long curPartSize = (i + 1 == partCount) ? (fileLength - startPos
) : partSize;
    InputStream instream = new FileInputStream(sampleFile);
    // Skips the uploaded parts.
    instream.skip(startPos);
    UploadPartRequest uploadPartRequest = new UploadPartRequest();
    uploadPartRequest.setBucketName(bucketName);
    uploadPartRequest.setKey(objectName);
    uploadPartRequest.setUploadId(uploadId);
    uploadPartRequest.setInputStream(instream);
    // Sets the part size. The size of a part must be equal to
or larger than 100 KB. However, the size of the last part is not
limited.
    uploadPartRequest.setPartSize(curPartSize);
    // Sets the serial number of each part. Each part has a serial
 number ranging from 1 to 10,000. If the serial number of a part
exceeds the range, the InvalidArgument error code is returned.
    uploadPartRequest.setPartNumber( i + 1);
  // The parts can be uploaded in different orders and even from
 different clients. OSS combines the parts following the serial
numbers into the complete object.
    UploadPartResult uploadPartResult = ossClient.uploadPart(
uploadPartRequest);
  // A PartTag is returned each time when a part is uploaded and is
stored in the partETags parameter.
    partETags.add(uploadPartResult.getPartETag());
}

/* Step 3: Completes the multipart upload task.
*/
// Sorts the partETags in an ascending order.
Collections.sort(partETags, new Comparator<PartETag>() {
    public int compare(PartETag p1, PartETag p2) {
        return p1.getPartNumber() - p2.getPartNumber();
    }
});
// All valid partETags must be provided to perform this step. OSS
verifies the availability of each part according to the partETags
and combines the parts into the complete object.
CompleteMultipartUploadRequest completeMultipartUploadRequest =
        new CompleteMultipartUploadRequest(bucketName, objectName,
uploadId, partETags);
ossClient.completeMultipartUpload(completeMultipartUploadRequest);

// Views the tags added to the object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, objectName);
Map<String, String> getTags = tagSet.getAllTags();
System.out.println("object tagging: "+ getTags.toString());
```

```
// Closes the OSSClient.
ossClient.shutdown();
```

- **Add tags to an object when uploading it in the append upload method.**

  Run the following code to add tags to an object when uploading it in the append upload method:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify
an actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

String content1 = "Hello OSS A \n";
String content2 = "Hello OSS B \n";
String content3 = "Hello OSS C \n";

Map<String, String> tags = new HashMap<String, String>();
tags.put("key1", "value1");
tags.put("key2", "value2");

ObjectMetadata meta = new ObjectMetadata();
// Configures the tags to be added to the object that needs to be
uploaded.
 meta.setObjectTagging(tags);
// Specifies the content type of the object to be uploaded.
meta.setContentType("text/plain");

// Sets parameters by calling AppendObjectRequest.
AppendObjectRequest appendObjectRequest = new AppendObjectRequest(
bucketName, objectName, new ByteArrayInputStream(content1.getBytes
()), meta);

// Sets each parameter by calling AppendObjectRequest.
// Sets the name of the bucket that the object is uploaded to.
//appendObjectRequest.setBucketName("<yourBucketName>");
// Sets the object name.
//appendObjectRequest.setKey("<yourObjectName>");
// Sets the content to be appended to the object, which has two
types: InputStream and File. This line sets the type to InputStream.
//appendObjectRequest.setInputStream(new ByteArrayInputStream(
content1.getBytes()));
// Sets the content to be appended to the object, which has two
types: InputStream and File. This line sets the type to File.
//appendObjectRequest.setFile(new File("<yourLocalFile>"));
// Sets the metadata of the object. The metadata settings take
effect only when the object is appended for the first time.
//appendObjectRequest.setMetadata(meta);
```

```
// Appends the object for the first time. Only the tags set when the
 object is appended for the first time are added to the object.
// Sets the position from where the object is appended.
appendObjectRequest.setPosition(0L);
AppendObjectResult appendObjectResult = ossClient.appendObject(
appendObjectRequest);
// Outputs the CRC64 value of the object, which is calculated
according to the ECMA-182 standard.
System.out.println(appendObjectResult.getObjectCRC());

// Appends the object for the second time.
// The nextPosition parameter indicates the position (object size)
required for the next upload request.nextPosition.
appendObjectRequest.setPosition(appendObjectResult.getNextPosition
());
appendObjectRequest.setInputStream(new ByteArrayInputStream(content2
.getBytes()));
appendObjectResult = ossClient.appendObject(appendObjectRequest);

// Appends the object for the third time.
appendObjectRequest.setPosition(appendObjectResult.getNextPosition
());
appendObjectRequest.setInputStream(new ByteArrayInputStream(content3
.getBytes()));
appendObjectResult = ossClient.appendObject(appendObjectRequest);

// Views the tags added to the uploaded object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, objectName);
Map<String, String> getTags = tagSet.getAllTags();
System.out.println("object tagging: "+ getTags.toString());

// Closes the OSSClient.
ossClient.shutdown();
```

- **Add tags to an object when uploading it in the resumable upload method.**

  Run the following code to add tags to an object when uploading it in the resumable upload method:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify
an actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Configures the tags to be added to the object.
Map<String, String> tags = new HashMap<String, String>();
tags.put("key1", "value1");
tags.put("key2", "value2");

ObjectMetadata meta = new ObjectMetadata();
// Specifies the content type of the object to be uploaded.
```

```
meta.setContentType("text/plain");
// Sets the tags to be added to the object.
meta.setObjectTagging(tags);

// Sets parameters by calling UploadFileRequest.
UploadFileRequest uploadFileRequest = new UploadFileRequest("<
yourBucketName>","<yourObjectName>");

// Sets each parameter by calling UploadFileRequest.
// Sets the name of the bucket that the object is uploaded to.
//uploadFileRequest.setBucketName("<yourBucketName>");
// Sets the object name.
//uploadFileRequest.setKey("<yourObjectName>");
// Specifies the local file to be uploaded.
uploadFileRequest.setUploadFile("<yourLocalFile>");
// Sets the number of upload tasks that can be simultaneously
performed. The default value is 1.
uploadFileRequest.setTaskNum(5);
// Specifies the size of the parts to be uploaded. The part size
ranges from 100 KB to 5 GB. The default value is: object size/10000.
uploadFileRequest.setPartSize(1 * 1024 * 1024);
// Enables the resumable upload, which is disabled by default.
uploadFileRequest.setEnableCheckpoint(true);
// Sets the local file used to store the results of resumable upload
 tasks. This parameter must be set if the resumable upload function
 is enabled. The upload progress is stored in the file. If a part
 fails to be uploaded, the next upload task continues from the
position recorded in the file. This file is deleted after the object
 is uploaded and is stored in the same directory as the local file
that needs to be uploaded, that is, uploadFile.ucp.
uploadFileRequest.setCheckpointFile("<yourCheckpointFile>");
// Sets the metadata of the object.
uploadFileRequest.setObjectMetadata(meta);
// Sets the upload callback function. The parameter type is Callback
.
uploadFileRequest.setCallback("<yourCallbackEvent>");

// Starts the resumable upload task and adds the tags to the object.
ossClient.uploadFile(uploadFileRequest);

// Views the tags added to the uploaded object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, objectName);
Map<String, String> getTags = tagSet.getAllTags();
System.out.println("object tagging: "+ getTags.toString());

// Closes the OSSClient.
ossClient.shutdown();
```

Add tags to an uploaded object or modify the tags added to an object

Run the following code to add tags to an uploaded object or modify the tags added to an object:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
```

```
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

Map<String, String> tags = new HashMap<String, String>();
tags.put("key1", "value1");
tags.put("key2", "value2");

// Adds tags to the object.
ossClient.setObjectTagging(bucketName, objectName, tags);

// Closes the OSSClient.
ossClient.shutdown();
```

Add tags to an object when copying it

You can specify one of the following tagging rules when copying an object:

- **Copy (default): Copy the tags of the source object to the destination object.**
- **Replace: Add the tags specified in the request to the destination object.**

Examples of adding tags to objects of different sizes when copying them are described as follows:

- **Run the following code to add tags to an object smaller than 1 GB when directly copying it:**

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify
an actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String sourceBucketName = "<yourSourceBucketName>";
String sourceObjectName = "<yourSourceObjectName>";
String destinationBucketName = "<yourDestinationBucketName>";
String destinationObjectName = "<yourDestinationObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Creates a CopyObjectRequest object.
CopyObjectRequest copyObjectRequest = new CopyObjectRequest
(sourceBucketName, sourceObjectName, destinationBucketName,
destinationObjectName);

// Configures the tags to be added to the object. If the headers
are not configured, the tags of the source object are added to the
destination object by default.
```

```
Map<String, String> headers = new HashMap<String, String>();
headers.put(OSSHeaders.COPY_OBJECT_TAGGING_DIRECTIVE, "REPLACE");
headers.put(OSSHeaders.OSS_TAGGING, "key1=value1&key2=value2");
copyObjectRequest.setHeaders(headers);

// Copies the source object to the destination object.
CopyObjectResult result = ossClient.copyObject(copyObjectRequest);
System.out.println("ETag: " + result.getETag() + " LastModified: "
 + result.getLastModified());

// Views the tags added to the destination object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, destinatio
nObjectName);
Map<String, String> getTags = tagSet.getAllTags();
System.out.println("dest object tagging: "+ getTags.toString());

// Closes the OSSClient.
ossClient.shutdown();
```

- **Run the following code to add tags to an object larger than 1 GB when copying it in the multipart copy method:**

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify
an actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String sourceBucketName = "<yourSourceBucketName>";
String sourceObjectName = "<yourSourceObjectName>";
String destinationBucketName = "<yourDestinationBucketName>";
String destinationObjectName = "<yourDestinationObjectName>";

ObjectMetadata objectMetadata = ossClient.getObjectMetadata(
sourceBucketName, sourceObjectName);
// Obtains the size of the source object.
long contentLength = objectMetadata.getContentLength();
// Sets the part size to 10 MB.
long partSize = 1024 * 1024 * 10;
// Calculates the number of parts.
int partCount = (int) (contentLength / partSize);
if (contentLength % partSize != 0) {
    partCount++;
}

System.out.println("total part count:" + partCount);

// Configures the tags in the HTTP headers.
Map<String, String> tags2 = new HashMap<String, String>();
tags2.put("key3", "value3");
tags2.put("key4", "value4");

ObjectMetadata metadata = new ObjectMetadata();
metadata.setObjectTagging(tags2);

// Initializes a copy task and sets the tags to be added to the
destination object.
```

```
InitiateMultipartUploadRequest initiateMultipartUploadRequest = new
 InitiateMultipartUploadRequest(destinationBucketName, destinatio
nObjectName, metadata);
InitiateMultipartUploadResult initiateMultipartUploadResult =
ossClient.initiateMultipartUpload(initiateMultipartUploadRequest);
String uploadId = initiateMultipartUploadResult.getUploadId();

// Copy the object part by part.
List<PartETag> partETags = new ArrayList<PartETag>();
for (int i = 0; i < partCount; i++) {
     // Calculates the part size.
    long skipBytes = partSize * i;
    long size = partSize < contentLength - skipBytes ? partSize :
contentLength - skipBytes;

    // Creates an UploadPartCopyReuqest. You can specify conditions
by setting UploadPartCopyReuqest.
    UploadPartCopyRequest uploadPartCopyRequest =
        new UploadPartCopyRequest(sourceBucketName, sourceObjectName
, destinationBucketName, destinationObjectName);
    uploadPartCopyRequest.setUploadId(uploadId);
    uploadPartCopyRequest.setPartSize(size);
    uploadPartCopyRequest.setBeginIndex(skipBytes);
    uploadPartCopyRequest.setPartNumber(i + 1);

    UploadPartCopyResult uploadPartCopyResult = ossClient.uploadPart
Copy(uploadPartCopyRequest);
    // Stores the returned ETags of parts in partETags.
    partETags.add(uploadPartCopyResult.getPartETag());
}

// Completes the multipart copy task.
CompleteMultipartUploadRequest completeMultipartUploadRequest = new
CompleteMultipartUploadRequest(
                    destinationBucketName, destinationObjectName,
uploadId, partETags);
CompleteMultipartUploadResult completeMultipartUploadResult =
ossClient.completeMultipartUpload(completeMultipartUploadRequest);

// Views the tags added to the source object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, sourceObje
ctName);
Map<String, String> getTags = tagSet.getAllTags();
System.out.println("src object tagging: "+ getTags.toString());

// Views the tags added to the destination object.
tagSet = ossClient.getObjectTagging(bucketName, destinationObjectNam
e);
getTags = tagSet.getAllTags();
System.out.println("dest object tagging: "+ getTags.toString());

// Closes the OSSClient.
ossClient.shutdown();
```

Add tags to a symbolic link object

Run the following code to add tags to a symbolic link object:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String symLink = "<yourSymLink>";
String destinationObjectName = "<yourDestinationObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Configures the tags to be added to the symbolic link object.
Map<String, String> tags = new HashMap<String, String>();
tags.put("key1", "value1");
tags.put("key2", "value2");

// Generates the metadata of the object.
ObjectMetadata metadata = new ObjectMetadata();
metadata.setObjectTagging(tags);

// Creates a CreateSymlinkRequest.
CreateSymlinkRequest createSymlinkRequest = new CreateSymlinkRequest(
bucketName, symLink, destinationObjectName);

// Sets the metadata of the symbolic link object.
createSymlinkRequest.setMetadata(metadata);

// Creates a symbolic link object.
ossClient.createSymlink(createSymlinkRequest);

// Views the tags added to the symbolic link object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, symLink);
Map<String, String> getTags = tagSet.getAllTags();
System.out.println("symLink tagging: "+ getTags.toString());

// Closes the OSSClient.
ossClient.shutdown();
```

2.10.2 Obtain the tags added to an object

This topic describes how to obtain the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to obtain the tags added to an object:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
```

```
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtains the tags added to the object.
TagSet tagSet = ossClient.getObjectTagging(bucketName, objectName);
Map<String, String> tags = tagSet.getAllTags();

// Views the tags added to the object.
System.out.println("object tagging: "+ tags.toString());

// Closes the OSSClient.
ossClient.shutdown();
```

For more information about how to obtain the tags added to an object, see *#unique_89*.

2.10.3 Delete the tags added to an object

This topic describes how to delete the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to delete the tags added to an object:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Deletes the tags added to the object.
ossClient.deleteObjectTagging(bucketName, objectName);
```

For more information about how to delete the tags added to an object, see *#unique_91*.

2.10.4 Object tagging and lifecycle rules

Lifecycle rules can take effect on objects with specified prefixes or tags. You can also specify prefixes and tags as the condition of a lifecycle rule at the same time.

Note:

If you set a specified tag as the condition of a lifecycle rule, the rule applies to an object only when the key and value in the tag of the object both match the specified tag. If you specify a prefix and multiple tags as the condition of a lifecycle rule, the rule applies to an object only when the prefix and tags of the object match the prefix and all tags specified in the rule.

Set tags as the matching condition of a lifecycle rule

Run the following code to set tags as the matching condition of a lifecycle rule:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Creates a SetBucketLifecycleRequest.
SetBucketLifecycleRequest request = new SetBucketLifecycleRequest(
bucketName);

// Sets the rule ID and the prefixes and tags used as the matching
condition.
String ruleId0 = "rule0";
String matchPrefix0 = "A0/";
Map<String, String> matchTags0 = new HashMap<String, String>();
matchTags0.put("key0", "value0");


String ruleId1 = "rule1";
String matchPrefix1 = "A1/";
Map<String, String> matchTags1 = new HashMap<String, String>();
matchTags1.put("key1", "value1");


String ruleId2 = "rule2";
String matchPrefix2 = "A2/";

String ruleId3 = "rule3";
String matchPrefix3 = "A3/";
```

```
// Sets an expiration rule for objects so that objects are expired
three days after they are modified for the last time.
LifecycleRule rule = new LifecycleRule(ruleId0, matchPrefix0,
RuleStatus.Enabled, 3);
rule.setTags(matchTags0)
request.AddLifecycleRule(rule);

// Sets an expiration rule for objects so that objects created before
the specified date are expired.
rule = new LifecycleRule(ruleId1, matchPrefix1, RuleStatus.Enabled);
rule.setCreatedBeforeDate(DateUtil.parseIso8601Date("2022-10-12T00:00:
00.000Z"));
rule.setTags(matchTags1)
request.AddLifecycleRule(rule);

// Sets an expiration rules for parts so that parts are expired 3 days
 after they are generated.
rule = new LifecycleRule(ruleId2, matchPrefix2, RuleStatus.Enabled);
LifecycleRule.AbortMultipartUpload abortMultipartUpload = new
LifecycleRule.AbortMultipartUpload();
abortMultipartUpload.setExpirationDays(3);
rule.setAbortMultipartUpload(abortMultipartUpload);
request.AddLifecycleRule(rule);

// Sets an expiration rules for parts so that parts generated before
the specified date are expired.
rule = new LifecycleRule(ruleId3, matchPrefix3, RuleStatus.Enabled);
abortMultipartUpload = new LifecycleRule.AbortMultipartUpload();
abortMultipartUpload.setCreatedBeforeDate(DateUtil.parseIso8601Date("
2022-10-12T00:00:00.000Z"));
rule.setAbortMultipartUpload(abortMultipartUpload);
request.AddLifecycleRule(rule);

ossClient.setBucketLifecycle(request);

// Closes the OSSClient.
ossClient.shutdown();
```

View the tags set as the matching condition of a lifecycle rule

Run the following code to view the tags set as the matching condition of a lifecycle rule:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";

// Creates an OSSClient.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtains the lifecycle rule set for the object.
List<LifecycleRule> rules = ossClient.getBucketLifecycle(bucketName);
```

```
// Views the lifecycle rule set for the object.
for (LifecycleRule rule1 : rules) {
    // Views the rule ID.
    System.out.println("rule id: " + rule1.getId());

    // Views the prefixes set as the matching condition.
    System.out.println("rule prefix: " + rule1.getPrefix());

    // Views the tags set as the matching condition.
    if (rule1.hasTags()) {
     System.out.println("rule tagging: "+ rule1.getTags().toString());
    }

    // Views the expiration rules in which objects are expired some
days after they are modified for the last time.
    if (rule1.hasExpirationDays()) {
     System.out.println("rule expiration days: " + rule1.getExpirat
ionDays());
    }

    // Views the expiration rules in which objects created before a
specified date are expired.
    if (rule1.hasCreatedBeforeDate()) {
      System.out.println("rule expiration create before days: " +
rule1.getCreatedBeforeDate());
    }

    // Views the expiration rule for parts.
    if(rule1.hasAbortMultipartUpload()) {
        if(rule1.getAbortMultipartUpload().hasExpirationDays()) {
          System.out.println("rule abort uppart days: " + rule1.
getAbortMultipartUpload().getExpirationDays());
        }

        if (rule1.getAbortMultipartUpload().hasCreatedBeforeDate()) {
          System.out.println("rule abort uppart create before date: "
 + rule1.getAbortMultipartUpload().getCreatedBeforeDate());
        }
    }
}

// Closes the OSSClient.
ossClient.shutdown();
```

2.11 Single-connection bandwidth throttling

This topic describes how to ensure other applications have sufficient bandwidth when you perform upload and download operations on OSS. You can include the x-oss-traffic-limit parameter in your requests and set bandwidth limits.

Note:

For more information about the scenarios and precautions of the single-connection bandwidth throttling feature, see *#unique_93* in the OSS Developer Guide.

Configure bandwidth throttling for common uploads and downloads

Use the following code to configure bandwidth throttling when you upload and download objects to and from OSS:

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

String bucketName = "<yourBucketName>"
String objectName = "<yourObjectName>"
String localFileName = "<yourLocalFileName>"
String downLoadFileName = "<yourDownLoadFileName>"

// Set the bandwidth limit to 100 KB/s (819200 bit/s).
int limitSpeed = 100 * 1024 * 8;

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Configure bandwidth throttling for the object to be uploaded.
InputStream inputStream = new FileInputStream(localFileName);
PutObjectRequest PutObjectRequest = new PutObjectRequest(bucketName,
key, inputStream);
PutObjectRequest.setTrafficLimit(limitSpeed);
ossClient.putObject(PutObjectRequest);

// Configure bandwidth throttling for the object to be downloaded.
GetObjectRequest getObjectRequest = new GetObjectRequest(bucketName,
key);
getObjectRequest.setTrafficLimit(limitSpeed);
File localFile = new File(downLoadFileName);
ossClient.getObject(getObjectRequest, localFile);

 ossClient.shutdown();
```

Configure bandwidth throttling for uploads and downloads that use signed URLs

Use the following code to configure bandwidth throttling when you upload and download objects from and to OSS by using signed URLs.

```
// This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you log on with the AccessKey pair
of an Alibaba Cloud account because the account has permissions on
 all API operations in OSS. We recommend that you log on with a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
```

```
String accessKeySecret = "<yourAccessKeySecret>";

String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";
String localFileName = "<yourLocalFileName>";
String downLoadFileName = "<yourDownLoadFileName>";

// Set the bandwidth limit to 100 KB/s (819200 bit/s).
int limitSpeed = 100 * 1024 * 8;

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create a signed URL for uploading an object and set the valid
period of the URL to 60s.
Date date = new Date();
date.setTime(date.getTime() + 60 * 1000);
GeneratePresignedUrlRequest request = new GeneratePresignedUrlRequest(
bucketName, objectName, HttpMethod.PUT);
request.setExpiration(date);
request.setTrafficLimit(limitSpeed);
URL signedUrl = ossClient.generatePresignedUrl(request);
System.out.println("put object url" = signedUrl);

// Configure bandwidth throttling for the object to be uploaded.
InputStream inputStream = new FileInputStream(localFileName);
ossClient.putObject(signedUrl, instream, -1, null, true);

// Create a signed URL for downloading an object and set the valid
period of the URL to 60s.
date = new Date();
date.setTime(date.getTime() + 60 * 1000);
request = new GeneratePresignedUrlRequest(bucketName, key, HttpMethod.
GET);
request.setExpiration(date);
request.setTrafficLimit(limitSpeed);
signedUrl = ossClient.generatePresignedUrl(request);
System.out.println("get object url" = signedUrl);

// Configure bandwidth throttling for the object to be downloaded.
GetObjectRequest getObjectRequest =  new GetObjectRequest(signedUrl,
null);
ossClient.getObject(getObjectRequest, downLoadFileName);

ossClient.shutdown();
```

2.12 Server-side encryption

OSS supports server-side encryption for uploaded data. This means that when user data is uploaded, OSS encrypts the data and permanently stores the data. Then, when the data is downloaded by a user, OSS automatically decrypts the data, returns the original data to the user, and declares in the header of the response that the data has been encrypted on the server.

The following server-side encryption methods are available for different application scenarios:

- **Server-side encryption that uses CMKs managed by KMS for encryption and decryption (SSE-KMS)**

  When uploading an object, you can use a specified CMK ID or the default CMK managed by KMS to encrypt and decrypt a large amount of data. This method is cost-effective because you do not need to send user data to the KMS server through networks for encryption and decryption.

  Notice:

  API calling fees are incurred if you use CMKs to encrypt or decrypt data.

- **Server-side encryption fully managed by OSS (SSE-OSS)**

  When you upload an object, OSS encrypts the object on the server side by using AES256 fully managed by OSS. In this method, OSS uses AES256 to encrypt each object with an individual key. Furthermore, the individual keys are encrypted by a CMK that is updated periodically for higher security. This method applies to encrypt or decrypt bulk data.

Notice:

- **Only one server-side encryption method can be used for an object at one time.**
- **If you configure server-side encryption for a bucket, you can still configure the encryption method for a single object when uploading or copying it. In this case, the encryption method configured for the object takes precedence. For more information, see *PutObject*.**
- **For more information about server-side encryption, see *Server-side encryption*.**

Configure server-side encryption for a bucket

You can run the following code to configure the default encryption method for a bucket. After the method is configured, all objects that are uploaded to the bucket without their encryption methods configured are encrypted in the configured default encryption method.

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
```

```
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Configures server-side encryption for the bucket.
ServerSideEncryptionByDefault applyServerSideEncryptionByDefault = new
 ServerSideEncryptionByDefault(SSEAlgorithm.KMS);
applyServerSideEncryptionByDefault.setKMSMasterKeyID("<yourTestKmsId
>");
ServerSideEncryptionConfiguration sseConfig = new ServerSideEncryption
Configuration();
sseConfig.setApplyServerSideEncryptionByDefault(applyServerSideEncry
ptionByDefault);
SetBucketEncryptionRequest request = new SetBucketEncryptionRequest("<
yourBucketName>", sseConfig);
ossClient.setBucketEncryption(request);

// Closes the OSSClient instance.
ossClient.shutdown();
```

Obtain the server-side encryption settings of a bucket

You can run the following code to obtain the server-side encryption settings of a bucket:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Obtains the server-side encryption settings of the bucket.
ServerSideEncryptionConfiguration sseConfig = ossClient.getBucketE
ncryption("<yourBucketName>");
System.out.println("get Algorithm: " + sseConfig.getApplyServerSideEn
cryptionByDefault().getSSEAlgorithm());
System.out.println("get kmsid: " + sseConfig.getApplyServerSideEn
cryptionByDefault().getKMSMasterKeyID());

// Closes the OSSClient instance.
ossClient.shutdown();
```

Delete the server-side encryption settings of a bucket

You can run the following code to delete the server-side encryption settings of a bucket:

```
// This example uses the China East 1 (Hangzhou) endpoint. Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Creates an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Deletes the server-side encryption settings of the bucket.
ossClient.deleteBucketEncryption("<yourBucketName>");

// Closes the OSSClient instance.
ossClient.shutdown();
```

2.13 Authorized access

This topic describes how to authorize access to OSS.

Use STS to authorize temporary access

You can use Security Token Service (STS) to authorize temporary access to OSS. STS is a web service that provides temporary access tokens for cloud computing users. You can use STS to grant a third-party application or a RAM user (whose user ID is managed by you) an access credential with a customized validity period and permissions. For more information about STS, see *What is STS?*

STS has the following benefits:

- **You only need to generate an access token and send the access token to a third-party application, rather than exposing your long-term key (AccessKey) to the third-party application. You can customize the access permissions and validity period of this token.**
- **The access token automatically expires after the validity period.**

For more information about the process of accessing OSS with STS, see *#unique_97* **in OSS Developer Guide.**

The following code provides an example on how to create a signature request with STS:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String securityToken = "<yourSecurityToken>";

// After you obtain a temporary STS credential for your OSSClient,
the OSSClient is generated based on the security token and temporary
AccessKey pair (AccessKey ID and AccessKey secret).
// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, securityToken);

// Perform an operation in OSS. For example, you can upload or
download an object.
// Upload an object to OSS.
ossClient.putObject(putObjectRequest);

// Download an object to a local file. If the name of the object is
the same as that of the local file, the object will replace the local
 file. If the name of the object is different from that of the local
file, the object will be downloaded.
// ossClient.getObject(new GetObjectRequest(bucketName, objectName),
new File("<yourLocalFile>"));

// Shut down the OSSClient instance.
ossClient.shutdown();
```

Use a signed URL to authorize temporary access

- **Generate a signed URL**

  You can provide the signed URL that is generated to a visitor to grant the visitor temporary access. When generating a signed URL, you can specify the validity period of the URL to restrict the period of access from visitors.

- **Generate a signed URL that allows HTTP GET requests**

  The following code provides an example on how to generate a signed URL that allows HTTP GET requests:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
```

```
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Set the validity period of the URL to one hour.
Date expiration = new Date(new Date().getTime() + 3600 * 1000);
// Generate the signed URL that allows HTTP GET requests. Visitors
can use a browser to access relevant content.
URL url = ossClient.generatePresignedUrl(bucketName, objectName,
expiration);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Generate a signed URL that allows other HTTP methods**

  A signed URL must be generated for temporary access from a visitor to perform operations such as uploading or deleting objects. For example, the visitor can upload an object to OSS by using a signed URL that is generated to allow HTTP PUT requests.

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String securityToken = "<yourSecurityToken>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// After you obtain a temporary STS credential for your OSSClient,
the OSSClient is generated based on the security token and temporary
 AccessKey pair (AccessKey ID and AccessKey secret).
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret, securityToken);

GeneratePresignedUrlRequest request = new GeneratePresignedUrl
Request(bucketName, objectName, HttpMethod.PUT);
// Set the validity period of the URL to one hour.
Date expiration = new Date(new Date().getTime() + 3600 * 1000);
request.setExpiration(expiration);
// Set ContentType.
request.setContentType(DEFAULT_OBJECT_CONTENT_TYPE);
// Set user metadata.
request.addUserMetadata("author", "aliy");
```

```
// Generate a signed URL that allows HTTP PUT requests.
URL signedUrl = ossClient.generatePresignedUrl(request);

Map<String, String> requestHeaders = new HashMap<String, String>();
requestHeaders.put(HttpHeaders.CONTENT_TYPE, DEFAULT_OBJECT_CONTE
NT_TYPE);
requestHeaders.put(OSS_USER_METADATA_PREFIX + "author", "aliy");

// Use the signed URL to upload an object.
ossClient.putObject(signedUrl, new ByteArrayInputStream("Hello OSS".
getBytes()), -1, requestHeaders, true);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

Visitors can specify the HttpMethod.PUT parameter and use the signed URL to upload objects.

- **Add specified parameters to a URL**

  The following code provides an example on how to generate a signed URL with specified parameters:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
operations and maintenance. To create your RAM user, log on to https
://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);

// Create a request.
GeneratePresignedUrlRequest generatePresignedUrlRequest = new
GeneratePresignedUrlRequest(bucketName, objectName);
// Set HttpMethod to PUT.
generatePresignedUrlRequest.setMethod(HttpMethod.PUT);
// Add user metadata.
generatePresignedUrlRequest.addUserMetadata("author", "baymax");
// Set ContentType.
generatePresignedUrlRequest.setContentType("application/octet-stream
");
// Set the validity period of the URL to one hour.
Date expiration = new Date(new Date().getTime() + 3600 * 1000);
generatePresignedUrlRequest.setExpiration(expiration);
// Generate the signed URL.
URL url = ossClient.generatePresignedUrl(generatePresignedUrlRequest
);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

- **Use a signed URL to obtain or upload an object**
  - **Use a signed URL to obtain an object**

    The following code provides an example on how to obtain a specified object through a signed URL:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
 operations and maintenance. To create your RAM user, log on to
https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);

Date expiration = DateUtil.parseRfc822Date("Wed, 18 Mar 2022 14:20
:00 GMT");
GeneratePresignedUrlRequest request = new GeneratePresignedUrl
Request(bucketName, objectName, HttpMethod.GET);
// Set the validity period.
request.setExpiration(expiration);
// Generate a signed URL that allows HTTP GET requests.
URL signedUrl = ossClient .generatePresignedUrl(request);
System.out.println("signed url for getObject: " + signedUrl);

// Use the signed URL to send a request.
Map<String, String> customHeaders = new HashMap<String, String>();
// Add the request headers to GetObject.
customHeaders.put("Range", "bytes=100-1000");
OSSObject object = ossClient.getObject(signedUrl,customHeaders);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

  - **Use a signed URL to upload an object**

    The following code provides an example on how to upload an object through a signed URL:

```
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use your
 RAM user's credentials to call API operations or perform routine
```

```
 operations and maintenance. To create your RAM user, log on to
https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);

// Generate a signed URL.
Date expiration = DateUtil.parseRfc822Date("Thu, 19 Mar 2019 18:00
:00 GMT");
GeneratePresignedUrlRequest request = new GeneratePresignedUrl
Request(bucketName, objectName, HttpMethod.PUT);
// Set the validity period.
request.setExpiration(expiration);
// Set ContentType.
request.setContentType("application/octet-stream");
// Add user metadata.
request.addUserMetadata("author", "aliy");
// Generate a signed URL that allows HTTP PUT requests.
URL signedUrl = ossClient.generatePresignedUrl(request);
System.out.println("signed url for putObject: " + signedUrl);

// Use the signed URL to send a request.
File f = new File("<yourLocalFile>");
FileInputStream fin = new FileInputStream(f);
// Add the request headers to PutObject.
Map<String, String> customHeaders = new HashMap<String, String>();
customHeaders.put("Content-Type", "application/octet-stream");
customHeaders.put("x-oss-meta-author", "aliy");

PutObjectResult result = ossClient.putObject(signedUrl, fin, f.
length(), customHeaders);

// Shut down the OSSClient instance.
ossClient.shutdown();
```

2.14 Data security

OSS Java SDK provides data integrity checks and file encryption on the server and client to guarantee data security during file (object) uploads, downloads, and replication.

Data integrity checks

OSS Java SDK provides MD5 authentication and cyclic redundancy checks (CRC) to guarantee data integrity for object (file) uploads, downloads, and replication.

- **MD5 authentication**

  If you configure Content-MD5 for file uploads, OSS calculates the MD5 value based on the received content. If the MD5 value is inconsistent with that of the uploaded file, InvalidDigest is returned. Thus, data integrity is guaranteed.

  Run the following code for MD5 authentication during the file upload:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId,
accessKeySecret);

// Upload a string.
String content = "Hello OSS";

ObjectMetadata meta = new ObjectMetadata();
// Configure MD5 authentication.
String md5 = BinaryUtil.toBase64String(BinaryUtil.calculateMd5(
content.getBytes()));
meta.setContentMD5(md5);

ossClient.putObject("<yourBucketName>", "<yourObjectName>", new
ByteArrayInputStream(content.getBytes()), meta);

// Close your OSSClient instance.
ossClient.shutdown();
```

  Note:
  MD5 authentication can be used for putObject, getObject, appendObject, postObject, and uploadPart.

- **CRC64**

  CRC is enabled by default during object (file) uploads, downloads, and replicatio n to guarantee data integrity.

  Run the following code to check whether CRC takes effect during the file upload:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId,
accessKeySecret);

// Upload a string.
String content = "Hello OSS";

PutObjectResult result = ossClient.putObject("<yourBucketName>", "<
yourObjectName>", new ByteArrayInputStream(content.getBytes()));

// Check whether CRC is enabled. True indicates it is enabled while
false indicates it is disabled.
if (ossClient.getClientConfiguration().isCrcCheckEnabled()) {
 // Check whether CRC takes effect based on the response.
    OSSUtils.checkChecksum(result.getClientCRC(), result.getServerC
RC(), result.getRequestId());
 }
// Close your OSSClient instance.
ossClient.shutdown();
```

Note:

- **You can use CRC64 authentication for putObject, getObject, appendObject, and uploadPart.**
- **CRC64 authentication affects the upload and download progress because CRC64 uses CPU space.**

File encryption on the server

You can set the serverSideEncryption parameter in Object Meta to AES256 or KMS for file uploads to encrypt the file and save it on the server. For more information, see *Server-side encryption* in the OSS Developer Guide.

Run the following code for file encryption on the server during file uploads:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";

// Create an OSSClient instance.
OSSClient ossClient = new OSSClient(endpoint, accessKeyId, accessKeyS
ecret);

// Upload a string.
```

```
String content = "Hello OSS";

ObjectMetadata meta = new ObjectMetadata();
// Set the server-side encryption code to AES256 for the file.
meta.setServerSideEncryption(ObjectMetadata.AES_256_SERVER_SIDE_
ENCRYPTION);

ossClient.putObject("<yourBucketName>", "<yourObjectName>", new
ByteArrayInputStream(content.getBytes()), meta);

// Close your OSSClient instance.
ossClient.shutdown();
```

Client-side encryption

To encrypt a file on the client, you can encrypt the data on your local device, upload it to OSS, download it to the local device, and decrypt it. For more information, see *Introduction to client-side encryption SDK* **in OSS Developer Guide.**

2.15 Image processing

Image Processing (IMG) is a massive, secure, cost-effective and highly reliable image processing service. After source images are uploaded to OSS, you can process images on any Internet device at any anytime, from anywhere through simple RESTful APIs.

For more information about IMG, see *Image Processing***.**

For the complete code of IMG, see *GitHub***.**

Basic features

OSS offers the following IMG features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**
- *Image processing*
- *Image processing access rules*: **calls multiple IMG functions.**

Usage

IMG uses standard HTTP GET. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is private read and write, only authorized users are allowed for access.

- **Anonymous access**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Specifies the name of a bucket.
endpoint	Specifies endpoint used to access a region.
object	Specifies the name of an image object.
image	Specifies the reserved identifier for IMG.
action	Specifies the operations on an image, such as scaling, cropping, and rotating .

Parameter	Description
param	Specifies the parameter that indicates the operation on an image.

- **Basic operations**

  For example, scale an image to a width of 100 px. Adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Customized image styles**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=style/<yourStyleName>
```

  - **style: specifies a reserved identifier of a customized image style.**
  - **yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.**

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=style/oss-pic-style-w-100
```

- **Cascade operations**

  Cascade operations allow you to perform multiple operations on an image sequentially.

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss
-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<
yourParamValue2>/...
```

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100/rotate,90
```

- **HTTPS access**

IMG supports access through HTTPS. Example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  Authorized access allows you to customize an image style, HTTPS access, and cascade operations.

  Run the following code to generate a signed URL for IMG:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";
// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId,
accessKeySecret);
// Customize the image style.
String style = "image/resize,m_fixed,w_100,h_100/rotate,90";
// Set the expiration time to 10 minutes.
Date expiration = new Date(new Date().getTime() + 1000 * 60 * 10 );
GeneratePresignedUrlRequest req = new GeneratePresignedUrlRequest(
bucketName, objectName, HttpMethod.GET);
req.setExpiration(expiration);
req.setProcess(style);
URL signedUrl = ossClient.generatePresignedUrl(req);
System.out.println(signedUrl);
// Close your OSSClient.
ossClient.shutdown();sClient.shutdown();
```

- **Access with SDK**

  You can use SDK to access and process an image object with any ACLs.

  SDK allows you to customize image styles, HTTPS access, and cascade operations.

  - **Basic operations**

    Run the following code to perform basic operations on an image:

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in
 OSS. We recommend that you log on as a RAM user to access APIs
```

```
 or perform routine operations and maintenance. To create a RAM
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";
// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);
// Scale the image.
String style = "image/resize,m_fixed,w_100,h_100";
GetObjectRequest request = new GetObjectRequest(bucketName,
objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-resize.jpg"));
// Crop the image.
style = "image/crop,w_100,h_100,x_100,y_100,r_1";
request = new GetObjectRequest(bucketName, objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-crop.jpg"));
// Rotate the image.
style = "image/rotate,90";
request = new GetObjectRequest(bucketName, objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-rotate.jpg"));
// Sharpen the image.
style = "image/sharpen,100";
request = new GetObjectRequest(bucketName, objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-sharpen.jpg"));
// Add watermarks.
style = "image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ";
request = new GetObjectRequest(bucketName, objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-watermark.jpg"));
// Convert the image format.
style = "image/format,png";
request = new GetObjectRequest(bucketName, objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-format.png"));
// Obtain image information.
style = "image/info";
request = new GetObjectRequest(bucketName, objectName);
request.setProcess(style);
ossClient.getObject(request, new File("example-info.txt"));
// Close your OSSClient.
ossClient.shutdown();
```

- **Customize an image style**

  Run the following code to customize an image style:

```
// This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba
Cloud account because the account has permissions on all APIs in
 OSS. We recommend that you log on as a RAM user to access APIs
 or perform routine operations and maintenance. To create a RAM
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
```

```
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);

// Customize the image style.
String style = "style/<yourCustomStyleName>";
GetObjectRequest request = new GetObjectRequest(bucketName,
objectName);
request.setProcess(style);

ossClient.getObject(request, new File("example-new.jpg"));

// Close your OSSClient.
ossClient.shutdown();
```

- **Cascade operations**

  Run the following code to perform cascade operations on an image:

```
// This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba
Cloud account because the account has permissions on all APIs in
 OSS. We recommend that you log on as a RAM user to access APIs
 or perform routine operations and maintenance. To create a RAM
account, log on to https://ram.console.aliyun.com.
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String objectName = "<yourObjectName>";

// Create an OSSClient instance.
OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);

// Perform cascade operations.
String style = "image/resize,m_fixed,w_100,h_100/rotate,90";
GetObjectRequest request = new GetObjectRequest(bucketName,
objectName);
request.setProcess(style);

ossClient.getObject(request, new File("example-new.jpg"));

// Disable the OSSClient.
ossClient.shutdown();
```

Image processing persistence

Run the following code for image processing persistence:

```
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance.
To create a RAM account, log on to https://ram.console.aliyun.com.
```

```
String accessKeyId = "<yourAccessKeyId>";
String accessKeySecret = "<yourAccessKeySecret>";
String bucketName = "<yourBucketName>";
String sourceImage = "<yourSourceImageName>";

try {
    // Create an OSSClient instance.
    OSS ossClient = new OSSClientBuilder().build(endpoint, accessKeyId
, accessKeySecret);

    // Image processing persistence: Scale the image.
    StringBuilder sbStyle = new StringBuilder();
    Formatter styleFormatter = new Formatter(sbStyle);
    String styleType = "image/resize,m_fixed,w_100,h_100";
    String targetImage = "example-resize.png";
    styleFormatter.format("%s|sys/saveas,o_%s,b_%s", styleType,
            BinaryUtil.toBase64String(targetImage.getBytes()),
            BinaryUtil.toBase64String(bucketName.getBytes()));
    System.out.println(sbStyle.toString());
    ProcessObjectRequest request = new ProcessObjectRequest(bucketName
, sourceImage, sbStyle.toString());
    GenericResult processResult = ossClient.processObject(request);
    String json = IOUtils.readStreamAsString(processResult.getResponse
().getContent(), "UTF-8");
    processResult.getResponse().getContent().close();
    System.out.println(json);

    // Image processing persistence: Clip the image.
    sbStyle.delete(0, sbStyle.length());
    styleType = "image/crop,w_100,h_100,x_100,y_100,r_1";
    targetImage = "example-crop.png";
    styleFormatter.format("%s|sys/saveas,o_%s,b_%s", styleType,
            BinaryUtil.toBase64String(targetImage.getBytes()),
            BinaryUtil.toBase64String(bucketName.getBytes()));
    System.out.println(sbStyle.toString());
    request = new ProcessObjectRequest(bucketName, sourceImage,
sbStyle.toString());
    processResult = ossClient.processObject(request);
    json = IOUtils.readStreamAsString(processResult.getResponse().
getContent(), "UTF-8");
    processResult.getResponse().getContent().close();
    System.out.println(json);

    // Image processing persistence: Rotate the object.
    sbStyle.delete(0, sbStyle.length());
    styleType = "image/rotate,90";
    targetImage = "example-rotate.png";
    styleFormatter.format("%s|sys/saveas,o_%s,b_%s", styleType,
            BinaryUtil.toBase64String(targetImage.getBytes()),
            BinaryUtil.toBase64String(bucketName.getBytes()));
    request = new ProcessObjectRequest(bucketName, sourceImage,
sbStyle.toString());
    processResult = ossClient.processObject(request);
    json = IOUtils.readStreamAsString(processResult.getResponse().
getContent(), "UTF-8");
    processResult.getResponse().getContent().close();
    System.out.println(json);

    // Image processing persistence: Sharpen the image.
    sbStyle.delete(0, sbStyle.length());
    styleType = "image/sharpen,100";
    targetImage = "example-sharpen.png";
    styleFormatter.format("%s|sys/saveas,o_%s,b_%s", styleType,
            BinaryUtil.toBase64String(targetImage.getBytes()),
```

```
            BinaryUtil.toBase64String(bucketName.getBytes()));
    request = new ProcessObjectRequest(bucketName, sourceImage,
sbStyle.toString());
    processResult = ossClient.processObject(request);
    json = IOUtils.readStreamAsString(processResult.getResponse().
getContent(), "UTF-8");
    processResult.getResponse().getContent().close();
    System.out.println(json);

    // Image processing persistence: Add watermarks.
    sbStyle.delete(0, sbStyle.length());
    styleType = "image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ";
    targetImage = "example-watermark.png";
    styleFormatter.format("%s|sys/saveas,o_%s,b_%s", styleType,
            BinaryUtil.toBase64String(targetImage.getBytes()),
            BinaryUtil.toBase64String(bucketName.getBytes()));
    request = new ProcessObjectRequest(bucketName, sourceImage,
sbStyle.toString());
    processResult = ossClient.processObject(request);
    json = IOUtils.readStreamAsString(processResult.getResponse().
getContent(), "UTF-8");
    processResult.getResponse().getContent().close();
    System.out.println(json);

    // Image processing persistence: Convert the image format.
    sbStyle.delete(0, sbStyle.length());
    styleType = "image/format,jpg";
    targetImage = "example-formatconvert.jpg";
    styleFormatter.format("%s|sys/saveas,o_%s,b_%s", styleType,
            BinaryUtil.toBase64String(targetImage.getBytes()),
            BinaryUtil.toBase64String(bucketName.getBytes()));
    request = new ProcessObjectRequest(bucketName, sourceImage,
sbStyle.toString());
    processResult = ossClient.processObject(request);
    json = IOUtils.readStreamAsString(processResult.getResponse().
getContent(), "UTF-8");
    processResult.getResponse().getContent().close();
    System.out.println(json);

} catch (Exception e) {
    e.printStackTrace();
}
```

IMG tools

You can use the IMG viewer *ImageStyleViever* to directly view the IMG result.

2.16 Exception handling

OSS Java SDK has two types of exceptions: ClientException and OSSException. Both share the properties of RuntimeException.

Example for handling exceptions

Run the following code to handle exceptions:

```
try {
    // Perform operations on OSS, for example, object uploads.
    ossClient.putObject(...);
```

```
} catch (OSSException oe) {
    System.out.println("Caught an OSSException, which means your
request made it to OSS, "
            + "but was rejected with an error response for some reason
.") ;
    System.out.println("Error Message: " + oe.getErrorMessage());
    System.out.println("Error Code:       " + oe.getErrorCode());
    System.out.println("Request ID:      " + oe.getRequestId());
    System.out.println("Host ID:           " + oe.getHostId());
} catch (ClientException ce) {
    System.out.println("Caught an ClientException, which means the
client encountered "
            + "a serious internal problem while trying to communicate
with OSS, "
            + "such as not being able to access the network.") ;
    System.out.println("Error Message: " + ce.getMessage());
} finally {
    if (ossClient ! = null) {
        ossClient.shutdown();
    }
}
```

ClientException

The ClientException exception occurs when the client attempts to send requests and when the data is transmitted to OSS. For example, when a request is sent, ClientException may occur due to unavailable network connections. ClientException may occur during file uploads due to IO exceptions.

OSSException

ServiceException indicates a server exception. The exception occurs because a server error message is parsed. OSSException includes the error code and information returned from OSS. It is used to locate and handle problems.

The following table describes common OSSException information.

Parameter	Description
Code	Specifies the error code returned by OSS.
Message	Specifies the detailed error information returned by OSS.
RequestId	Specifies the UUID. It is unique and used to identify a request. If a problem persists, send the RequestId to OSS developers for help.

Parameter	Description
HostId	Identifies an OSS cluster. Ensure that the value of HostId is consistent with the Host (endpoint to access a bucket) used in the request.

Common OSS error codes

Error Code	Description	HTTP status code
AccessDenied	Access denied	403
BucketAlreadyExists	The bucket already exists.	409
BucketNotEmpty	The bucket is not empty.	409
EntityTooLarge	The entity size exceeds the maximum limit.	400
EntityTooSmall	The entity size is below the minimum limit.	400
FileGroupTooLarge	The total file group size exceeds the maximum limit.	400
FilePartNotExist	No such part exists.	400
FilePartStale	The part has expired.	400
InvalidArgument	The parameter format is invalid.	400
InvalidAccessKeyId	No such AccessKeyId exists.	403
InvalidBucketName	The bucket name is invalid.	400
InvalidDigest	The digest is invalid.	400
InvalidObjectName	The object name is invalid .	400
InvalidPart	The part is invalid.	400
InvalidPartOrder	The part order is invalid.	400
InvalidTargetBucketForLogging	The buckets that store log files are invalid.	400
InternalError	Internal OSS error	500
MalformedXML	The XML format is invalid.	400

Error Code	Description	HTTP status code
MethodNotAllowed	The method is not allowed.	405
MissingArgument	Parameters are not configured.	411
MissingContentLength	The content length is not configured.	411
NoSuchBucket	No such bucket exists.	404
NoSuchKey	No such object exists.	404
NoSuchUpload	No such uploadId exists.	404
NotImplemented	The methods are not implemented.	501
PreconditionFailed	Precondition error	412
RequestTimeTooSkewed	The local time set for OSSClient is deviated from the time set for the OSS server by over 15 minutes.	403
RequestTimeout	Request timeout	400
SignatureDoesNotMatch	Signature mismatch	403
InvalidEncryptionAlgorithmError	The specified encryption algorithm (the entropy encoding type) is invalid.	400

2.17 Common errors

This topic lists the causes and solutions for some common errors you may experience when you use the OSS Java SDK.

JAR package conflicts

- **Cause analysis**

  If a similar error is displayed when you use the OSS Java SDK, there is a JAR package conflict in your project:

```
Exception in thread "main" java.lang.NoClassDefFoundError: org/
apache/http/ssl/TrustStrategy
    at com.aliyun.oss.OSSClient.<init>(OSSClient.java:268)
    at com.aliyun.oss.OSSClient.<init>(OSSClient.java:193)
    at com.aliyun.oss.demo.HelloOSS.main(HelloOSS.java:77)
```

```
Caused by: java.lang.ClassNotFoundException: org.apache.http.ssl.
TrustStrategy
    at java.net.URLClassLoader$1.run(URLClassLoader.java:366)
    at java.net.URLClassLoader$1.run(URLClassLoader.java:355)
    at java.security.AccessController.doPrivileged(Native Method)
    at java.net.URLClassLoader.findClass(URLClassLoader.java:354)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:425)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:308)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:358)
    ... 3 more
```

Or

```
Exception in thread "main" java.lang.NoSuchFieldError: INSTANCE
 at org.apache.http.impl.io.DefaultHttpRequestWriterFactory.<init>(
DefaultHttpRequestWriterFactory.java:52)
 at org.apache.http.impl.io.DefaultHttpRequestWriterFactory.<init>(
DefaultHttpRequestWriterFactory.java:56)
 at org.apache.http.impl.io.DefaultHttpRequestWriterFactory.<clinit
>(DefaultHttpRequestWriterFactory.java:46)
 at org.apache.http.impl.conn.ManagedHttpClientConnectionFactory.<
init>(ManagedHttpClientConnectionFactory.java:82)
 at org.apache.http.impl.conn.ManagedHttpClientConnectionFactory.<
init>(ManagedHttpClientConnectionFactory.java:95)
 at org.apache.http.impl.conn.ManagedHttpClientConnectionFactory.<
init>(ManagedHttpClientConnectionFactory.java:104)
 at org.apache.http.impl.conn.ManagedHttpClientConnectionFactory.<
clinit>(ManagedHttpClientConnectionFactory.java:62)
 at org.apache.http.impl.conn.PoolingHttpClientConnectionManager$
InternalConnectionFactory.<init>(PoolingHttpClientConnectionManager.
java:572)
 at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.<
init>(PoolingHttpClientConnectionManager.java:174)
 at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.<
init>(PoolingHttpClientConnectionManager.java:158)
 at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.<
init>(PoolingHttpClientConnectionManager.java:149)
 at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.<
init>(PoolingHttpClientConnectionManager.java:125)
 at com.aliyun.oss.common.comm.DefaultServiceClient.createHttp
ClientConnectionManager(DefaultServiceClient.java:237)
 at com.aliyun.oss.common.comm.DefaultServiceClient.<init>(
DefaultServiceClient.java:78)
 at com.aliyun.oss.OSSClient.<init>(OSSClient.java:268)
 at com.aliyun.oss.OSSClient.<init>(OSSClient.java:193)
 at OSSManagerImpl.upload(OSSManagerImpl.java:42)
 at OSSManagerImpl.main(OSSManagerImpl.java:63)
```

A possible cause is that the OSS Java SDK uses Apache httpclient 4.4.1, whereas your project uses a version of Apache httpclient or commons-httpclient that conflicts with Apache httpclient 4.4.1. To view the JAR package and its version used in your project, run the `mvn dependency:tree` command in the directory of

the project. The following figure shows that your project uses Apache httpclient 4.3.

```
[INFO] --- maven-dependency-plugin:2.2:tree (default-cli) @ maven-demo ---
[INFO] com.aliyun.oss:maven-demo:jar:0.1.1-SNAPSHOT
[INFO] +- junit:junit:jar:4.10:test
[INFO] |  \- org.hamcrest:hamcrest-core:jar:1.1:test
[INFO] +- org.apache.httpcomponents:httpclient:jar:4.3:compile
[INFO] |  +- org.apache.httpcomponents:httpcore:jar:4.3:compile
[INFO] |  +- commons-logging:commons-logging:jar:1.1.3:compile
[INFO] |  \- commons-codec:commons-codec:jar:1.6:compile
[INFO] \- com.aliyun.oss:aliyun-sdk-oss:jar:2.2.1:compile
[INFO]    +- org.jdom:jdom:jar:1.1:compile
[INFO]    \- net.sf.json-lib:json-lib:jar:jdk15:2.4:compile
[INFO]       +- commons-beanutils:commons-beanutils:jar:1.8.0:compile
[INFO]       +- commons-collections:commons-collections:jar:3.2.1:compile
[INFO]       +- commons-lang:commons-lang:jar:2.5:compile
[INFO]       \- net.sf.ezmorph:ezmorph:jar:1.0.6:compile
```

- You can use either of the following methods to resolve the JAR package conflict:
  - Use consistent versions of Apache httpclient. If your project uses a version that conflicts with Apache httpclient 4.4.1, use Apache httpclient 4.4.1 and delete Apache httpclient dependencies of other versions in the pom.xml file. If your project uses commons-httpclient, conflicts may also occur. To resolve these conflicts, delete commons-httpclient.
  - Resolve dependency conflicts. If your project is dependent on multiple third-party packages and the third-party packages are dependent on different versions of Apache httpclient, your project may contain dependency conflicts. Use exclusion to resolve these conflicts. For more information, see *Maven Guides*.

The OSS Java SDK is dependent on the following versions of packages. The solution to conflicts is similar to that for httpclient.

```
[INFO] com.aliyun.oss:maven-demo:jar:0.1.1-SNAPSHOT
[INFO] +- junit:junit:jar:4.10:test
[INFO] |  \- org.hamcrest:hamcrest-core:jar:1.1:test
[INFO] \- com.aliyun.oss:aliyun-sdk-oss:jar:2.2.1:compile
[INFO]    +- org.apache.httpcomponents:httpclient:jar:4.4.1:compile
[INFO]    |  +- org.apache.httpcomponents:httpcore:jar:4.4.1:compile
[INFO]    |  +- commons-logging:commons-logging:jar:1.2:compile
[INFO]    |  \- commons-codec:commons-codec:jar:1.9:compile
[INFO]    +- org.jdom:jdom:jar:1.1:compile
[INFO]    \- net.sf.json-lib:json-lib:jar:jdk15:2.4:compile
[INFO]       +- commons-beanutils:commons-beanutils:jar:1.8.0:compile
[INFO]       +- commons-collections:commons-collections:jar:3.2.1:compile
[INFO]       +- commons-lang:commons-lang:jar:2.5:compile
[INFO]       \- net.sf.ezmorph:ezmorph:jar:1.0.6:compile
```

Missing packages

- **Cause analysis**

  If a similar error is displayed when you use the OSS Java SDK, your project may lack the necessary files to compile or run the OSS Java SDK:

```
Exception in thread "main" java.lang.NoClassDefFoundError: org/
apache/http/auth/Credentials
        at com.aliyun.oss.OSSClient.<init>(OSSClient.java:268)
        at com.aliyun.oss.OSSClient.<init>(OSSClient.java:193)
        at com.aliyun.oss.demo.HelloOSS.main(HelloOSS.java:76)
Caused by: java.lang.ClassNotFoundException: org.apache.http.auth.
Credentials
        at java.net.URLClassLoader$1.run(URLClassLoader.java:366)
        at java.net.URLClassLoader$1.run(URLClassLoader.java:355)
        at java.security.AccessController.doPrivileged(Native Method
)
        at java.net.URLClassLoader.findClass(URLClassLoader.java:354
)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:425)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:
308)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:358)
        ... 3 more
```

  Or

```
Exception in thread "main" java.lang.NoClassDefFoundError: org/
apache/http/protocol/HttpContext
        at com.aliyun.oss.OSSClient.<init>(OSSClient.java:268)
        at com.aliyun.oss.OSSClient.<init>(OSSClient.java:193)
        at com.aliyun.oss.demo.HelloOSS.main(HelloOSS.java:76)
Caused by: java.lang.ClassNotFoundException: org.apache.http.
protocol.HttpContext
        at java.net.URLClassLoader$1.run(URLClassLoader.java:366)
        at java.net.URLClassLoader$1.run(URLClassLoader.java:355)
        at java.security.AccessController.doPrivileged(Native Method
)
        at java.net.URLClassLoader.findClass(URLClassLoader.java:354
)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:425)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:
308)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:358)
        ... 3 more
```

  Or

```
Exception in thread "main" java.lang.NoClassDefFoundError: org/jdom/
input/SAXBuilder
        at com.aliyun.oss.internal.ResponseParsers.getXmlRootElement
(ResponseParsers.java:645)
        at … …
        at com.aliyun.oss.OSSClient.doesBucketExist(OSSClient.java:
471)
```

```
        at com.aliyun.oss.OSSClient.doesBucketExist(OSSClient.java:
465)
        at com.aliyun.oss.demo.HelloOSS.main(HelloOSS.java:82)
Caused by: java.lang.ClassNotFoundException: org.jdom.input.
SAXBuilder
        at java.net.URLClassLoader$1.run(URLClassLoader.java:366)
        at java.net.URLClassLoader$1.run(URLClassLoader.java:355)
        at java.security.AccessController.doPrivileged(Native Method
)
        at java.net.URLClassLoader.findClass(URLClassLoader.java:354
)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:425)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:
308)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:358)
        ... 11 more
```

Dependencies of the OSS Java SDK are as follows:

- **aliyun-sdk-oss-2.2.1.jar**
- **hamcrest-core-1.1.jar**
- **jdom-1.1.jar**
- **commons-codec-1.9.jar**
- **httpclient-4.4.1.jar**
- **commons-logging-1.2.jar**
- **httpcore-4.4.1.jar**
- **log4j-1.2.15.jar**

In the preceding dependencies, all packages except the log4j-1.2.15.jar package are required. You must add this package to enable logging.

- **Solution**

  Use one of the following methods to add OSS Java SDK dependencies to your project:

  - **If your project is in Eclipse, import dependencies to Eclipse, as described in the "Method 2: Import a JAR package" section of *Installation*.**
  - **If your project is in Ant, add OSS Java SDK dependencies to the lib directory.**
  - **If you directly use the .javac or .java file, run the `-classpath` or `-cp` command to specify the path where OSS Java SDK dependencies are stored, or save the Java SDK dependencies to classpath.**

Connection timeout

- **Cause analysis**

  If a similar error is displayed when the OSS Java SDK program is run, possible causes are endpoint errors or unavailable networks:

```
com.aliyun.oss.ClientException: SocketException
    at com.aliyun.oss.common.utils.ExceptionFactory.createNetw
orkException(ExceptionFactory.java:71)
    at com.aliyun.oss.common.comm.DefaultServiceClient.sendReques
tCore(DefaultServiceClient.java:116)
    at com.aliyun.oss.common.comm.ServiceClient.sendRequestImpl(
ServiceClient.java:121)
    at com.aliyun.oss.common.comm.ServiceClient.sendRequest(
ServiceClient.java:67)
    at com.aliyun.oss.internal.OSSOperation.send(OSSOperation.java:
92)
    at com.aliyun.oss.internal.OSSOperation.doOperation(OSSOperation
.java:140)
    at com.aliyun.oss.internal.OSSOperation.doOperation(OSSOperation
.java:111)
    at com.aliyun.oss.internal.OSSBucketOperation.getBucketInfo(
OSSBucketOperation.java:1152)
    at com.aliyun.oss.OSSClient.getBucketInfo(OSSClient.java:1220)
    at com.aliyun.oss.OSSClient.getBucketInfo(OSSClient.java:1214)
    at com.aliyun.oss.demo.HelloOSS.main(HelloOSS.java:94)
Caused by: org.apache.http.conn.HttpHostConnectException: Connect
to oss-test.oss-cn-hangzhou-internal.aliyuncs.com:80 [oss-test.oss-
cn-hangzhou-internal.aliyuncs.com/10.84.135.99] failed: Connection
timed out: connect
    at org.apache.http.impl.conn.DefaultHttpClientConnectionOperator
.connect(DefaultHttpClientConnectionOperator.java:151)
    at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.
connect(PoolingHttpClientConnectionManager.java:353)
    at org.apache.http.impl.execchain.MainClientExec.establishRoute(
MainClientExec.java:380)
    at org.apache.http.impl.execchain.MainClientExec.execute(
MainClientExec.java:236)
    at org.apache.http.impl.execchain.ProtocolExec.execute(
ProtocolExec.java:184)
    at org.apache.http.impl.execchain.RedirectExec.execute(
RedirectExec.java:110)
    at org.apache.http.impl.client.InternalHttpClient.doExecute(
InternalHttpClient.java:184)
    at org.apache.http.impl.client.CloseableHttpClient.execute(
CloseableHttpClient.java:82)
    at com.aliyun.oss.common.comm.DefaultServiceClient.sendReques
tCore(DefaultServiceClient.java:113)
    ... 9 more
```

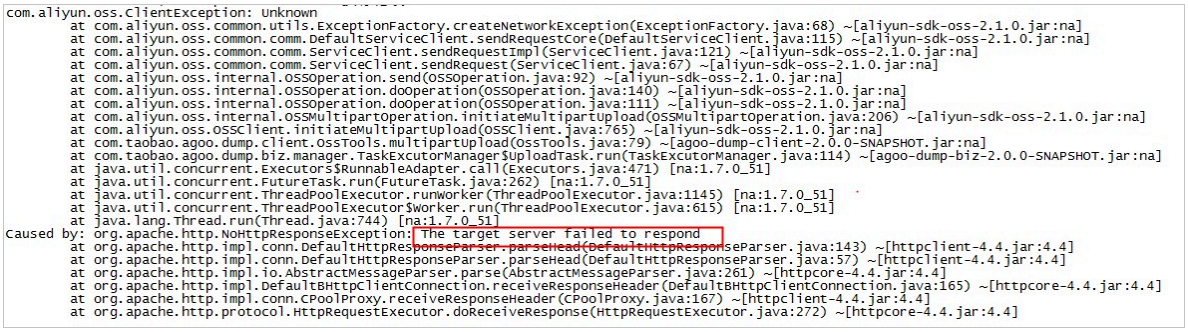
- **Solution**

  You can use the *ossutil* tool to troubleshoot problems.

org.apache.http.NoHttpResponseException: The target server failed to respond

- **Cause analysis**

  A similar error is displayed when the OSS Java SDK program is run:



  The preceding error may occur if expired connections are used. This error occurs only when a Java SDK version earlier than 2.1.2 is used.

- **Solution**

  Upgrade the OSS Java SDK to V2.1.2 or later.

An error occurs when the OSS Java SDK is called

- **Cause analysis**

  The OSS Java SDK stops responding when the Java program is called. Run the `jstack -l pid` command to view the stack. A similar error is displayed:

```
"main" prio=6 tid=0x000000000291e000 nid=0xc40 waiting on condition
 [0x0000000002dae000]
java.lang.Thread.State: WAITING (parking)
    at sun.misc.Unsafe.park(Native Method)
    - parking to wait for  <0x00000007d85697f8> (a java.util.
concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
    at java.util.concurrent.locks.LockSupport.park(LockSupport.java:
186)
    at java.util.concurrent.locks.AbstractQueuedSynchronizer$
ConditionObject.await(AbstractQueuedSynchronizer.java:2043)
    at org.apache.http.pool.PoolEntryFuture.await(PoolEntryFuture.
java:138)
    at org.apache.http.pool.AbstractConnPool.getPoolEntryBlocking(
AbstractConnPool.java:306)
    at org.apache.http.pool.AbstractConnPool.access$000(AbstractCo
nnPool.java:64)
    at org.apache.http.pool.AbstractConnPool$2.getPoolEntry(
AbstractConnPool.java:192)
    at org.apache.http.pool.AbstractConnPool$2.getPoolEntry(
AbstractConnPool.java:185)
    at org.apache.http.pool.PoolEntryFuture.get(PoolEntryFuture.java
:107)
    at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.
leaseConnection(PoolingHttpClientConnectionManager.java:276)
    at org.apache.http.impl.conn.PoolingHttpClientConnectionManager$
1.get(PoolingHttpClientConnectionManager.java:263)
```

```
    at org.apache.http.impl.execchain.MainClientExec.execute(
MainClientExec.java:190)
    at org.apache.http.impl.execchain.ProtocolExec.execute(
ProtocolExec.java:184)
    at org.apache.http.impl.execchain.RedirectExec.execute(
RedirectExec.java:110)
    at org.apache.http.impl.client.InternalHttpClient.doExecute(
InternalHttpClient.java:184)
    at org.apache.http.impl.client.CloseableHttpClient.execute(
CloseableHttpClient.java:82)
    at com.aliyun.oss.common.comm.DefaultServiceClient.sendReques
tCore(DefaultServiceClient.java:113)
    at com.aliyun.oss.common.comm.ServiceClient.sendRequestImpl(
ServiceClient.java:123)
    at com.aliyun.oss.common.comm.ServiceClient.sendRequest(
ServiceClient.java:68)
    at com.aliyun.oss.internal.OSSOperation.send(OSSOperation.java:
94)
    at com.aliyun.oss.internal.OSSOperation.doOperation(OSSOperation
.java:146)
    at com.aliyun.oss.internal.OSSOperation.doOperation(OSSOperation
.java:113)
    at com.aliyun.oss.internal.OSSObjectOperation.getObject(
OSSObjectOperation.java:229)
    at com.aliyun.oss.OSSClient.getObject(OSSClient.java:629)
    at com.aliyun.oss.OSSClient.getObject(OSSClient.java:617)
    at samples.HelloOSS.main(HelloOSS.java:49)
```

A possible cause is the connection leaks in the connection pool. It may be because ossObject is not closed after use.

- **Solution**

  Check your program and ensure that no connection leaks occur. The following code provides an example of how to close ossObject:

```
// Read an object.
OSSObject ossObject = ossClient.getObject(bucketName, objectName);
// Perform operations on OSS.
// Close your ossObject.
ossObject.close();
```

Connection closure

- **Cause analysis**

  A similar error is displayed when you use ossClient.getObject:

```
Exception in thread "main" org.apache.http.ConnectionClosedException
: Premature end of Content-Length delimited message body (expected:
11990526; received: 202880)
    at org.apache.http.impl.io.ContentLengthInputStream.read(
ContentLengthInputStream.java:180)
    at org.apache.http.impl.io.ContentLengthInputStream.read(
ContentLengthInputStream.java:200)
    at org.apache.http.impl.io.ContentLengthInputStream.close(
ContentLengthInputStream.java:103)
```

```
    at org.apache.http.impl.execchain.ResponseEntityProxy.streamClos
ed(ResponseEntityProxy.java:128)
    at org.apache.http.conn.EofSensorInputStream.checkClose(
EofSensorInputStream.java:228)
    at org.apache.http.conn.EofSensorInputStream.close(EofSensorI
nputStream.java:174)
    at java.io.FilterInputStream.close(FilterInputStream.java:181)
    at java.io.FilterInputStream.close(FilterInputStream.java:181)
    at com.aliyun.oss.event.ProgressInputStream.close(ProgressIn
putStream.java:147)
    at java.io.FilterInputStream.close(FilterInputStream.java:181)
    at samples.HelloOSS.main(HelloOSS.java:39)
```

The possible cause is that the interval between the two data reading attempts exceeds one minute. OSS closes the connection if no data is sent or received for more than one minute.

- **Solution**

  If you need only to read part of data during unspecified periods of time, use *Range download* to avoid connection closure during data reading.

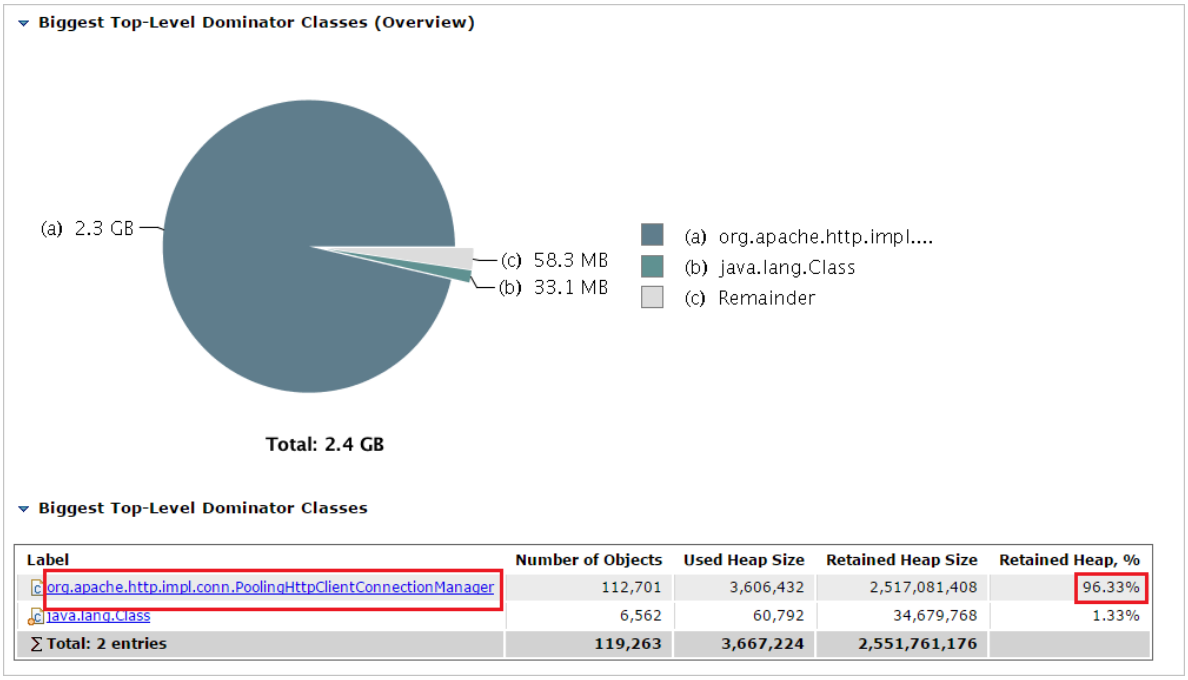
Memory leaks

- **Cause analysis**

  Memory leaks occur when the OSS Java SDK program is run for a period of time (from several hours to several days based on the amount of access traffic). We recommend that you use *Eclipse Memory Analyzer (MAT)* to analyze memory usage. For more information about use methods, see *Use MAT for Heap Dump File Analysis*.

  As shown in the following figure, PoolingHttpClientConnectionManager occupies 96 percent of memory. If the analysis result is similar to the following figure, a

possible cause is that the new OSSClient command is run multiple times while ossClient.shutdown is not called. Consequently, memory leaks occur.



- Solution

  Ensure that ossClient.shutdown is called after the new OSSClient command is run.

InterruptedException occurs when ossClient.shutdown is called

- Cause analysis

  A similar error is displayed when ossClient.shutdown is called:

```
java.lang.InterruptedException: sleep interrupted
        at java.lang.Thread.sleep(Native Method)
        at com.aliyun.oss.common.comm.IdleConnectionReaper.run(
IdleConnectionReaper:76)
```

  A possible cause is that the back-end thread (IdleConnectionReaper) of ossClient closes connections periodically. If ossClient.shutdown is called when IdleConnectionReaper is in the Sleep mode, the preceding exception occurs.

- Solution

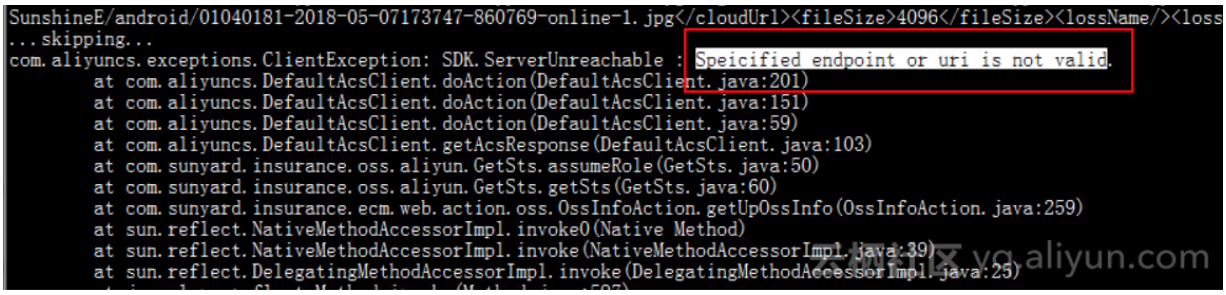
  This exception has been resolved in OSS Java SDK V2.3.0.

  For OSS Java SDK V2.3.0 and earlier, use the following code to ignore the exception:

```
try {
```

```
    ossClient.shutdown();
} catch(Exception e) {
}
```

"SDK. ServerUnreachable: Specified endpoint or uri is not valid"



- **Cause analysis**

  Possible causes are as follows: highly concurrent requests sent from the client to STS, timeout of connections to the server, and invalid STS SDK or SDK core version. Consequently, the user fails to connect to the Alibaba Cloud endpoint.

- **Solution**

  - **Reduce requests sent to STS if the requests are highly concurrent while ECS on the client or local PC is not able to support such bursts of requests.**
  - **If the connection to the server times out, check and analyze captured packets.**
  - **We recommend that you upgrade the STS SDK or SDK core to the latest version .**

NoSuchKey

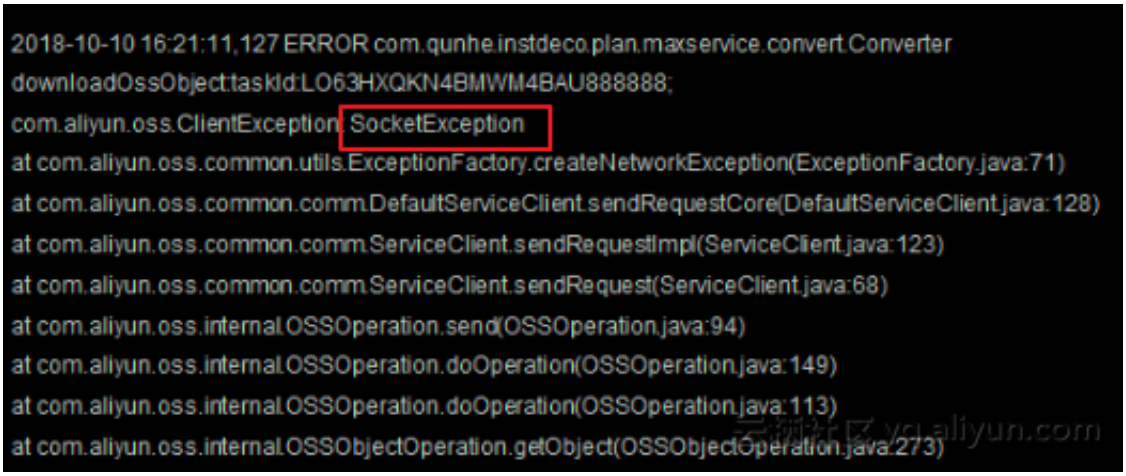
- **Cause analysis**

  The possible cause is the original file does not exist.

- **Solution**

  For more information, see *Troubleshoot 404 errors*.

socketException



- **Cause analysis**

  The possible cause is that socket fails to be specified during initialization. Consequently, the request is not received by OSS.

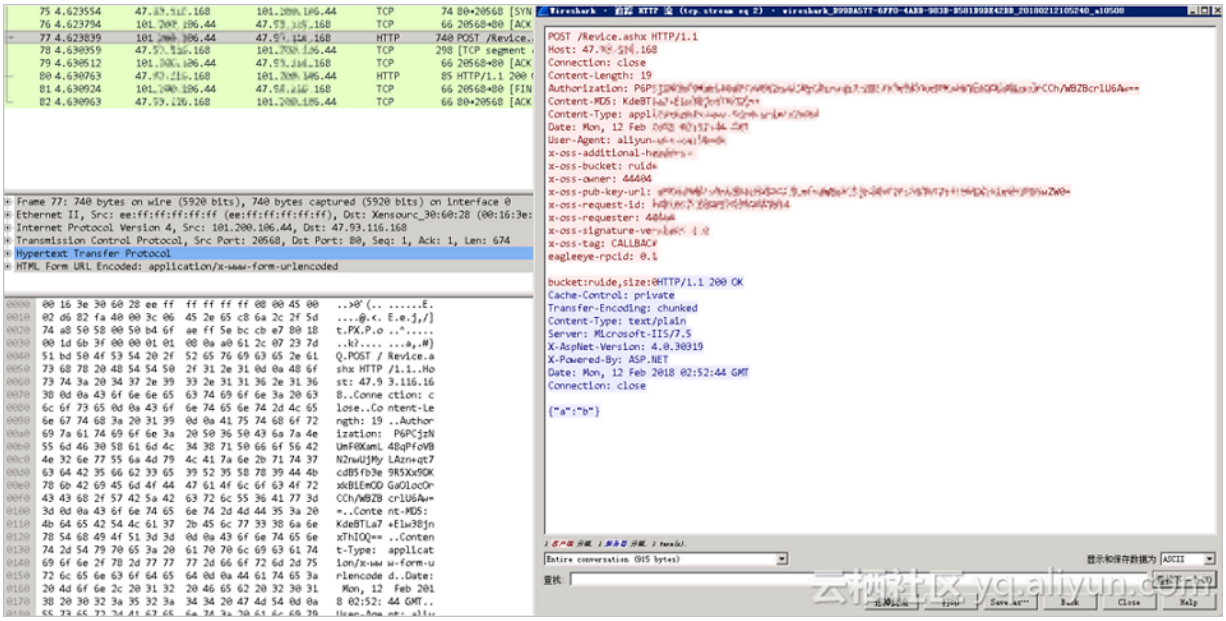
- **Solution**

  We recommend that you troubleshoot the exception as follows:

  - **Check whether there are jitters.**
  - **Check whether socket resources have been used by other processes.**
  - **Check the maximum number of connections configured in the SDK. If the actual number of connections exceeds the maxconnection setting, a socket exception occurs.**
  - **If the problem persists, we recommend that you deploy tcpdump or use wireshark to capture packets, reproduce the exception, and analyze the packets.**

What do I do if the callback operation of OSS PostObject fails whereas the callback operation of PutObject succeeds?

In most cases, if the JSON format is incorrect or a callback fails, an error message will be returned. In this case, you must test whether the callback operations of Put and Post are triggered.



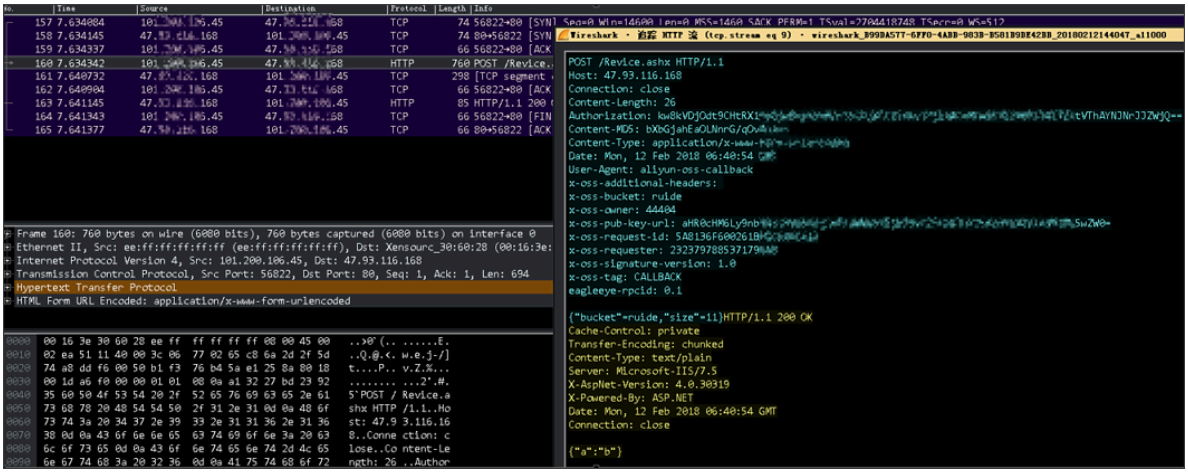
- **Cause analysis**

  The callback parameter follows the file parameter when you send the callback request.

- **Solution**

  Adjust the position of the callback parameter.

  The test result shows that the server captures the request.



Connection pool shut down

```
Caused by: java.lang.IllegalStateException: Connection pool shut down
  at org.apache.http.util.Asserts.check(Asserts.java:34)
  at org.apache.http.pool.AbstractConnPool.lease(AbstractConnPool.java
:184)
  at org.apache.http.impl.conn.PoolingHttpClientConnectionManager.
requestConnection(PoolingHttpClientConnectionManager.java:251)
  at org.apache.http.impl.execchain.MainClientExec.execute(MainClient
Exec.java:175)
  at org.apache.http.impl.execchain.ProtocolExec.execute(ProtocolExec.
java:184)
  at org.apache.http.impl.execchain.RedirectExec.execute(RedirectExec.
java:110)
  at org.apache.http.impl.client.InternalHttpClient.doExecute(
InternalHttpClient.java:184)
  at org.apache.http.impl.client.CloseableHttpClient.execute(
CloseableHttpClient.java:82)
  at com.aliyun.oss.common.comm.DefaultServiceClient.sendRequestCore(
DefaultServiceClient.java:124)
  at com.aliyun.oss.common.comm.ServiceClient.sendRequestImpl(
ServiceClient.java:133)
```

```
... 8 more
```

- **Cause analysis**

  After ossClient.shutdown() is called, requests are sent through ossClient.

- **Solution**

  Check the call logic. Ensure that no requests are sent through ossClient after ossClient.shutdown() is called.

Other errors

For more information about how to rectify other errors returned by OSS, see *OSS 403*.

3 Python

3.1 Preface

This topic describes the sample code of the OSS Python SDK in various use cases.

Source code

For the source code of the OSS Python SDK, visit *GitHub*.

Sample code

The OSS Python SDK provides a variety of sample code for your reference or use. The following table describes the sample code provided.

Sample file	Content
object_basic.py	*Quick start*, including creating a bucket and uploading, downloading, listing, and deleting objects
object_extra.py	Upload and manage objects, including *Manage Object Meta*, *Copy objects*, and *Append upload*
upload.py	Upload objects, including *Resumable upload* and *Multipart upload*
download.py	Download objects, including *Streaming download*, *Range download*, and *Resumable download*
object_check.py	Data verification methods during uploads and downloads, including MD5 and *CRC*
object_progress.py	*Upload progress bars* and *Download progress bars*
object_callback.py	*Upload callback* after an object is uploaded
object_post.py	Operations related to *#unique_127*
sts.py	How to use STS, including obtaining a temporary user key by assuming RAM roles and using the obtained key to access OSS

Sample file	Content
live_channel.py	Operations related to *LiveChannel*
image.py	Operations related to *Image processing*
bucket.py	*Bucket*, including creating, deleting, and listing buckets, *Static website hosting*, and *Manage lifecycle rules*
object_request_payment.py	Enable the *Enable pay-by-requester mode*
traffic_limit.py	Configure *Single-connection bandwidth throttling*

3.2 Installation

Environment requirements

This version of Python SDK is applicable to Python 2.6, 2.7, 3.3, 3.4 and 3.5.

- **Installation environment**

  Follow the instructions on *python official website* to install an appropriate version of Python.

- **Check Python version**

  Run `python -V` to check the Python version.

  In Windows, if a "Not an internal or external command" message prompts, edit the `Path` in environment variables to add the installation path of Python and pip. In general, pip is installed in the Script directory located in the installation path of Python.

  > **Note:**
  > **You may need to restart your computer make sure the environment variable takes effect.**

Download SDK

- *Download from GitHub*
- *Download previous versions*

For more information, see *OSS API documents*.

Install Python-devel

Python SDK uses the crcmod library which depends on the Python.h to caculate CRC. Therefore, if the Python.h file is missing, Python SDK can be successfully installed while the installation of crcmod C extension mode fails, which significantly reduces the efficiency of operations, such as upload and download. Therefore, you must install the python-devel package if it is missing.

- **You do not need to install the python-devel package in Windows and Mac OS X because header files that Python depends on are installed together with Python.**
- **For CentOS, RHEL, and Fedora, run the following command to install python-devel:**

```
yum install python-devel
```

- **For Debian and Ubuntu, run the following command to install python-devel:**

```
apt-get install python-dev
```

Install SDK

- **Install SDK by Pip**

  Run the following command:

```
pip install oss2
```

Note:
If Pip is not installed in your environment, see *Pip official website* to install Pip.

- **Install SDK by source code**

  Download the SDK package of an appropriate version from *GitHub* and decompress the package.Make sure that the setup.py file exists in the directory and run the following command:

  ```
  python setup.py install
  ```

Verification

- **Verify the SDK version**

  1. **Input `python` in the command line and press Enter to enter the Python environment.**
  2. **Run the following command to verify the SDK version:**

     ```
     >>> import oss2
     >>> oss2.__version__
     '2.0.0'
     ```

     If the preceding information is displayed, it means that you have successfully installed the Python SDK 2.0.0.

- **Verify the crcmod**

  Two methods are available for crcmod to calculate CRC: the C extension mode (in which crcmod calls the extension library _crcfunext.so to calculate CRC) and the Python-only mode. The performance of C extension mode is far better than the Python-only mode. The crcmod library is automatically installed when oss2 is installed. For more information about crcmod, see *crcmod introduction*.

  If you find that the speed of upload and download interfaces is significantly slow compared with other tools (such as *ossutil*) or other SDKs after the Python SDK is installed, it is probably because that the C extension mode of crcmod

library is not installed successfully, resulting that the Python-only mode is used to calculate CRC in upload and download.

You can verify whether the C extension mode of crcmode is successfully installed as follows:

1. **Input `python` in the command line and press Enter to enter the Python environment.**
2. **Input `import crcmod._crcfunext` and press Enter.**
   - **If no exception is reported, it indicates that the C extension mode has been successfully installed.**
   - **If the following message is displayed, it indicates that the C extension mode is not successfully installed.**

     ```
     >>> import crcmod._crcfunext
     Traceback (most recent call last):
     File "<stdin>", line 1, in <module>
     ImportError: No module named _crcfunext
     ```

     The error occurs because the Python.h file that _crcfunext.so depends on is missing when the crcmod is compiled, resulting in that the _crcfunext.so fails to be generated. In this case, CRC is calculated in the Python-only

method. The SDK is installed successfully. However, the efficiency of operations such as upload and download is low.

Perform the following operations if the problem occurs.

a. Run the following command to uninstall crcmod:

```
pip uninstall crcmod
```

b. *Install python-devel.*

c. Run the following command to install crcmod again:

```
pip install crcmod
```

If the installation still fails, uninstall crcmod and run the following command to check detailed information:

```
pip install crcmod -v
```

Uninstall SDK

If the SDK fails to be installed, we recommend you to uninstall the SDK using pip and install it again. You can run the following command to uninstall the SDK.

```
pip uninstall oss2
```

3.3 Quick start

This topic describes how to use OSS Python SDK to perform routine operations such as bucket creation, object uploads, and object downloads.

Create a bucket

Run the following code to create a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Set the ACL of the bucket to private.
```

```
bucket.create_bucket(oss2.models.BUCKET_ACL_PRIVATE)
```

For more information about endpoints, see *Regions and endpoints*.

Upload objects

Run the following code to upload a file to OSS:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# <yourLocalFile> consists of a local file path and a file name with
extension, for example: /users/local/myfile.txt
bucket.put_object_from_file('<yourObjectName>', '<yourLocalFile>')
```

Download objects

Run the following code to download a specified object to a local file:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# <yourLocalFile> consists of a local file path and a file name with
extension, for example: /users/local/myfile.txt
bucket.get_object_to_file('<yourObjectName>', '<yourLocalFile>')
```

List objects

Run the following code to list 10 objects in a specified bucket:

```
# -*- coding: utf-8 -*-
import oss2
from itertools import islice

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We strongly recommend that you create a RAM account and use it for
```

```
API access and daily O&M. Log on to https://ram.console.aliyun.com to
create a RAM account.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# oss2.ObjectIteratorr is used to traverse objects.
for b in islice(oss2. ObjectIterator(bucket), 10):
    print(b.key)
```

Delete objects

Run the following code to delete a specified object:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.delete_object('<yourObjectName>')
```

3.4 Initialization

This topic describes how to initialize OSS Python SDK.

Most operations in OSS Python SDK are performed through oss2.Service and oss2.Bucket.

- **The oss2.Service class is used to list buckets.**
- **The oss2.Bucket class is used to upload, download, and delete objects, and configure buckets.**

To initialize the two classes, you must specify an endpoint. The oss2.Service class does not support access with the custom domain (CNAME). For more information about endpoints, see *Regions and endpoints* **and** *Bind a custom domain name***.**

Initialize the oss2.Service class

For more information, see the bucket listing part in *Manage buckets***.**

Initialize the oss2.Bucket class.

- **Use a OSS domain name to initialize the class**

  Run the following code to initialize the oss2.Bucket class with a OSS domain name.

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This case uses the Hangzhou endpoint as an example. Fill in the
region endpoint name according to the actual circumstances.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'

bucket = oss2.Bucket(auth, endpoint, '<yourBucketName>')
```

- **Use a custom domain name to initialize the class**

  Run the following code to initialize the oss2.Bucket class with a custom domain name

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')

# Use a custom domain name: my-domain.com as an example. is_cname
=True indicates that the CNAME is enabled. Cname means binding a
custom domain name to the storage space. CNAME indicates a custom
domain bound to a bucket.
cname = 'http://my-domain.com'
bucket = oss2.Bucket(auth, cname, '<yourBucketName>', is_cname=True)
```

- **Set connection timeout**

  Run the following code to set connection timeout:

```
# -*- coding: utf-8 -*-
import oss2

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
```

```
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'

# Set the connection timeout to 30 seconds.
bucket = oss2.Bucket(auth, endpoint, '<yourBucketName>', connect_ti
meout=30)
```

- **Disable CRC verification**

  CRC verification is enabled by default during uploads and downloads to ensure data integrity during uploads and downloads. Run the following code to disable CRC verification:

  Warning:

  We recommended that you do not disable CRC verification. If you disable CRC verification, the data integrity during uploads and downloads is not guaranteed.

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'

bucket = oss2.Bucket(auth, endpoint, '<yourBucketName>', enable_crc=
False)
```

3.5 Buckets

3.5.1 Bucket

A bucket serves as a container that stores objects. Objects belong to a bucket.

Create a bucket

Run the following code to create a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# You can create a new bucket in a specified region by specifying the
 endpoint and bucket name. This example uses endpoint China (Hangzhou
). Specify the actual endpoint based on your requirements.
```

```
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# The default storage class and ACL of the created bucket are set to
standard and private read respectively.
bucket.create_bucket()
```

For more information about bucket naming rules, see *naming conventions* **in** *Basic concepts*.

You can specify the ACL and the storage class when you create a bucket, as shown in *Bucket-level permissions* **and** *Storage class*. **The sample code is as follows:**

```
# Set the bucket ACL to public read and set the storage class to
Infrequent Access.
bucket.create_bucket(oss2. BUCKET_ACL_PUBLIC_READ, oss2.models.
BucketCreateConfig(oss2. BUCKET_STORAGE_CLASS_IA))
```

List buckets

Run the following code to list buckets:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
service = oss2. Service(auth, 'http://oss-cn-hangzhou.aliyuncs.com')

print([b.name for b in oss2. BucketIterator(service)])
```

Configure an ACL for a bucket

The ACL of a bucket includes the following permissions.

Permission	Description	Value
Private	The bucket owner and the authorized users can read and write objects in the bucket. Other users cannot perform any operation on the objects.	oss2. BUCKET_ACL _PRIVATE

Permission	Description	Value
Public read	The bucket owner and the authorized users can read and write objects in the bucket. Other users can only read the objects in the bucket. Authorize this permission with caution.	oss2. BUCKET_ACL _PUBLIC_READ
Public read-write	All users can read and write objects in the bucket. Authorize this permission with caution.	oss2. BUCKET_ACL _PUBLIC_READ_WRITE

Run the following code to configure an ACL for a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Set the ACL for the bucket to private.
bucket.put_bucket_acl(oss2. BUCKET_ACL_PRIVATE)
```

Obtain the ACL for a bucket

Run the following code to obtain the ACL for a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
```

```
print(bucket.get_bucket_acl().acl)
```

Obtain bucket information

Run the following code to obtain bucket information:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtain bucket information
bucket_info = bucket.get_bucket_info()
print('name: ' + bucket_info.name)
print('storage class: ' + bucket_info.storage_class)
print('creation date: ' + bucket_info.creation_date)
print('intranet_endpoint: ' + bucket_info.intranet_endpoint)
print('extranet_endpoint ' + bucket_info.extranet_endpoint)
print('owner: ' + bucket_info.owner.id)
print('grant: ' + bucket_info.acl.grant)
```

Delete a bucket

Before a bucket is deleted, ensure that all objects in the bucket and fragments that are generated from multipart upload are deleted.

Note:

To delete the fragments that are generated from multipart upload, use *Bucket.ListMultipartUploads* to list all fragments, and then use *Bucket.AbortMultipartUpload* to delete the fragments.

Run the following code to delete a bucket:

```
# -*- coding: utf-8 -*-
import oss2
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
try:
    # Delete the bucket.
```

```
    bucket.delete_bucket()
except oss2.exceptions.BucketNotEmpty:
    print('bucket is not empty.')
except oss2.exceptions.NoSuchBucket:
    print('bucket does not exist')
```

For buckets which are not empty, you can list and delete objects stored in these buckets first (stop uploading for objects uploaded by multipart upload), and then delete the buckets.

3.5.2 Bucket tagging

By using the bucket tagging function, you can classify your buckets to manage them. For example, you can specify that only buckets with specified tags are returned in ListBucket operations.

- **Only the bucket owner and authorized RAM users can add tags to a bucket. Otherwise, the 403 Forbidden error is returned with the error code: AccessDeni ed.**
- **You can add a maxito a bucketmum of 20 tags (key-value pairs) to a bucket.**
- **The maximum size of a key is 64 bytes. The key of a tag cannot be null and cannot be prefixed with `http://`, `https://`, or `Aliyun`.**
- **The maximum size of a tag value is 128 bytes. The value of a tag can be null.**
- **The key and value of a tag must be UTF-8 encoded.**
- **If you use PutBucketTagging to add tags to a bucket, the original tags added to the bucket are completely overwritten.**

Add a tag to a bucket

You can run the following code to add a tag to a bucket:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import Tagging, TaggingRule

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Creates a tagging rule.
rule = TaggingRule()
rule.add('key1', 'value1')
rule.add('key2', 'value2')
```

```
# Creates a tag.
tagging = Tagging(rule)
# Adds the tag to the bucket.
result = bucket.put_bucket_tagging(tagging)
# Checks the returned HTTP status code.
print('http status:', result.status)
```

Obtain the tags added to a bucket

You can run the following code to obtain the tags to a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.

auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtains the tagging information about to a bucketsult = bucket.
get_bucket_tagging()
# Views the obtained tagging rule.
tag_rule = result.tag_set.tagging_rule
print('tag rule:', tag_rule)
```

List buckets with a specified tag

Run the following code to list buckets with a specified tag:

```
# -*- coding: utf-8 -*-
import oss2

# Creates a server object.
# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
service = oss2.Service(auth,  'http://oss-cn-hangzhou.aliyuncs.com')

# Inputs the tag-key and tag-value fields to the params parameter of
the list_buckets interface.
params = {}
params['tag-key'] = '<yourTagging_key>'
params['tag-value'] = '<yourTagging_value>'

# Lists buckets with the specified tag.
result = service.list_buckets(params=params)
# Views the list results.
for bucket in result.buckets:
```

```
    print('result bucket_name:', bucket.name)
```

Delete the tags added to a bucket

You can run the following code to delete the tags added to a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Deletes the tags added to the bucket.
result = bucket.delete_bucket_tagging()
# Checks the returned HTTP status code.
print('http status:', result.status)
```

3.5.3 Enable pay-by-requester mode

After pay-by-requester mode is enabled for a bucket in Alibaba Cloud OSS, the requester instead of the bucket owner pays the cost of requests and traffic. The bucket owner always pays the cost of storing data. You can enable this feature to share data without paying for all requests and traffic by yourself.

Request methods

- **Access from anonymous users is not allowed**

  If you enable pay-by-requester mode for a bucket, anonymous users are not allowed to access the bucket. Requesters must provide authentication informatio n so that OSS can identify and charge them, but not the bucket owner, for requests and traffic.

  If requesters assume the RAM role of an Alibaba Cloud account to request data, OSS charges this Alibaba Cloud account for such requests and traffic.
- **Requesters must include the x-oss-request-payer request header**

  If you enable pay-by-requester mode for a bucket, requesters must include the x-oss-request-payer:requester request header in a PUT, POST, GET, or HEAD

request. This field indicates that requesters understand that their requests and data downloads incur fees. Otherwise, requests cannot pass authentication.

Bucket owners do not need to include the x-oss-request-payer request header when accessing their own buckets. In this case, bucket owners are requesters who pay for such requests and traffic.

For more information about pay-by-requester mode, see *#unique_48* in the OSS Developer Guide.

Enable pay-by-requester mode

The following code provides an example of how to enable pay-by-requester mode:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.models import PAYER_BUCKETOWNER, PAYER_REQUESTER

# Security risks may arise if you log on with the AccessKey pair of
 an Alibaba Cloud account because the account has permissions on all
 API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Enable the pay-by-requester mode. The default request payment mode
is PAYER_BUCKETOWNER.
result = bucket.put_bucket_request_payment(PAYER_REQUESTER)

print("http respon status: ", result.status)
```

Obtain pay-by-requester configurations

The following code provides an example of how to obtain pay-by-requester configurations:

```
# -*- coding: utf-8 -*-

import oss2

# Security risks may arise if you log on with the AccessKey pair of
 an Alibaba Cloud account because the account has permissions on all
 API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
```

```
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtain pay-by-requester configurations.
result = bucket.get_bucket_request_payment()
print('payer:', result.payer)
```

Specify a third party to pay for access to objects

The following code provides an example of how to specify a third party to pay for access to objects when you use PutObject and DeleteObject:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.headers import OSS_REQUEST_PAYER

# Security risks may arise if you log on with the AccessKey pair of
 an Alibaba Cloud account because the account has permissions on all
 API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourRequestBucketName>')

object_name = '<yourRequestobjectName>'
headers = dict()
headers[OSS_REQUEST_PAYER] = "requester"

# Specify headers when you upload an object.
result = bucket.put_object(object_name, 'test-content', headers=
headers)

# Specify headers when you delete an object.
result = bucket.delete_object(object_name, headers=headers);
```

3.5.4 Manage lifecycle rules

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**

- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**
- **Status: This lifecyle rule takes effect or not.**

The parts uploaded by upload_part method also support lifecycle rules. The last modification time of an object is the time the multipart upload event is initiated.

For more information about lifecycle management, see *Manage object lifecycle*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import LifecycleExpiration, LifecycleRule, BucketLife
cycle,AbortMultipartUpload
import datetime

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This case uses the Hangzhou endpoint as an example. Fill in the
region endpoint name according to the actual circumstances.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Set the expiration duration to 3 days after last modified date.
rule1 = LifecycleRule('rule1', 'tests/',
                      status=LifecycleRule.ENABLED,
                      expiration=LifecycleExpiration(days=3))

# Configure expiration date to delete objects that are created before
the date.
rule2 = LifecycleRule('rule2', 'logging-',
                      status=LifecycleRule.DISABLED,
expiration=LifecycleExpiration(created_before_date=datetime.date(2018
, 12, 12)))

# Set the expiration duration to 3 days for parts in an object.
```

```
rule3 = LifecycleRule('rule3', 'tests1/',
                      status=LifecycleRule.ENABLED,
             abort_multipart_upload=AbortMultipartUpload(days=3))

# Configure expiration date to delete parts that are created before
the date.
rule4 = LifecycleRule('rule4', 'logging1-',
                      status=LifecycleRule.DISABLED,
                      abort_multipart_upload = AbortMultipartUpload(
created_before_date=datetime.date(2018, 12, 12)))

lifecycle = BucketLifecycle([rule1, rule2,rule3,rule4])

bucket.put_bucket_lifecycle(lifecycle)
```

View lifecycle rules

Run the following code to view lifecycle rules:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

lifecycle = bucket.get_bucket_lifecycle()

for rule in lifecycle.rules:
    if rule.abort_multipart_upload is None:
        print('id={0}, prefix={1}, status={2}, days={3}, created_be
fore_date={4}'
              .format(rule.id, rule.prefix, rule.status,
                      rule.expiration.days,
                      rule.expiration.created_before_date))
    else:
        print('id={0}, prefix={1}, status={2}, days={3}, created_be
fore_date={4}'
              .format(rule.id, rule.prefix, rule.status,
                      rule.abort_multipart_upload.days,
                      rule.abort_multipart_upload.created_before_date
))
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.delete_bucket_lifecycle()

# An exception occurs when you view lifecycle rules again.
try:
    lifecycle = bucket.get_bucket_lifecycle()
except oss2.exceptions.NoSuchLifecycle:
    print('lifecycle is not configured')
```

3.5.5 Hotlink protection

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49* in the OSS Developer Guide.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import BucketReferer

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Configure to allow empty Referers (True indicates that an empty
Referer is allowed, and False indicates that an empty Referer is not
allowed), and configure the referer whitelist.
bucket.put_bucket_referer(BucketReferer(True, ['http://aliyun.com', '
http://*.aliyuncs.com']))
```

Obtain a referer whiltelist

Run the following code to obtain a referer whiltelist:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

config = bucket.get_bucket_referer()
print('allow empty referer={0}, referers={1}'.format(config.allow_empt
y_referer, config.referers))
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# You cannot clear a referer whitelist directly. To clear a referer
whitelist, you need to create the rule that allows an empty referer
field and replace the original rule with the new rule.
bucket.put_bucket_referer(BucketReferer(True, []))
```

3.5.6 Set logging

You can enable access logs to record bucket access to log files, which are stored in a specified bucket.

The log file format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

For more information about access log files, see *Set access logging*.

Enable access logging

Run the following code to enable bucket access logging:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import BucketLogging

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Enable access logging. Store log files in the current bucket and set
 the log file storage directory to 'logging/'.
logging = bucket.put_bucket_logging(BucketLogging(bucket.bucket_name,
 'logging/'))
if logging.status == 200:
    print("Enable access logging")
else:
    print("request_id : ", logging.request_id)
    print("resp : ", logging.resp.response)
```

View access logging configurations

Run the following code to view the access logging configurations for a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

logging = bucket.get_bucket_logging()
print('TargetBucket={0}, TargetPrefix={1}'.format(logging.target_buc
ket, logging.target_prefix))
```

Disable access logging

Run the following code to disable access logging for a bucket:

```
# -*- coding: utf-8 -*-
import oss2
```

```
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

logging = bucket.delete_bucket_logging()
if logging.status == 204:
    print("Disable access logging")
else:
    print("request_id : ", logging.request_id)
    print("resp : ", logging.resp.response)
```

3.5.7 Static website hosting

You can set your bucket configuration to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

For more information about static website hosting, see *Static website hosting*.

Configure static website hosting

Run the following code to configure static website hosting:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import BucketWebsite

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Enable the static website hosting mode. Set the index page to index.
html and the error page to error.html.
bucket.put_bucket_website(BucketWebsite('index.html', 'error.html'))
```

View static website hosting configurations

Run the following code to view static website hosting configurations:

```
# -*- coding: utf-8 -*-
import oss2
```

```
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

try:
 # If the static website hosting mode is not enabled, a NoSuchWebsite
exception is thrown by get_bucket_website.
    website = bucket.get_bucket_website()
    print('Index file is {0}, error file is {1}'.format(website.
index_file, website.error_file))
except oss2.exceptions.NoSuchWebsite as e:
    print('Website is not configured, request_id={0}'.format(e.
request_id))
```

Delete static website hosting configurations

Run the following code to delete static website hosting configurations:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.delete_bucket_website()
```

3.5.8 CORS

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* and *PutBucketcors* in the OSS Developer Guide.

Configure CORS rules

Run the following code to configure CORS rules for the specified bucket:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import BucketCors, CorsRule
```

```
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

rule = CorsRule(allowed_origins=['*'],
                allowed_methods=['GET', 'HEAD'],
                allowed_headers=['*'],
                max_age_seconds=1000)

#  The existing rules will be replaced.
bucket.put_bucket_cors(BucketCors([rule]))
```

Obtain CORS rules

Run the following code to obtain CORS rules:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

try:
    cors = bucket.get_bucket_cors()
except oss2.exceptions.NoSuchCors:
    print('cors is not set')
else:
    for rule in cors.rules:
        print('AllowedOrigins={0}'.format(rule.allowed_origins))
        print('AllowedMethods={0}'.format(rule.allowed_methods))
        print('AllowedHeaders={0}'.format(rule.allowed_headers))
        print('ExposeHeaders={0}'.format(rule.expose_headers))
        print('MaxAgeSeconds={0}'.format(rule.max_age_seconds))
```

Delete CORS rules

Run the following code to delete all CORS rules for the specified bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.delete_bucket_cors()
```

3.6 Upload objects

3.6.1 Overview

This topic is the overview of object upload in OSS Python SDK.

In OSS, an object is the basic unit for data operations. OSS Python SDK provides the following file upload methods:

- *Simple upload*: **supports the upload of a file up to 5 GB in size.**
- *Append upload*: **supports the upload of a file up to 5 GB in size.**
- *Resumable upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files. It supports concurrent upload and you can define the size of each part.**
- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files.**

Note:

For more information about the usage scenarios of each upload method, see “Upload files” in OSS Developer Guide.

During the file upload, you can set *Object Meta*, **and view the upload progress in the** *upload progress bar*. **After you complete the file upload, you can perform** *upload callback*.

3.6.2 Simple upload

In simple uploads, you can upload files of the following five types: strings, bytes, Unicode, local files, and file streams.

You can use the bucket.put_object method to upload a file. This method supports multiple types of inputs, which are listed in the following table.

Type	Upload method
String	Direct uploaded
Bytes	Direct uploaded
Unicode	Uploaded after being converted to bytes encoded in UTF-8.
Local files	Uploaded as File objects, which must be opened in the binary mode.
File stream	Uploaded as Iterable objects in Chunked Encoding method.

Note:

If you upload an object whose name is the same as that of an existing object that you can access, the existing object is overwritten by the uploaded object, and a 200 OK message is returned.

For more information about PutObject, see *Detail analysis*.

Upload a string

You can run the following code to upload a string:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Return value
result = bucket.put_object('<yourObjectName>', 'content of object')
# HTTP return code
print('http status: {0}'.format(result.status))
# Request ID The request ID is the unique identification of a request
. We strongly recommend you to add this parameter in the program log.
print('request_id: {0}'.format(result.request_id))
# Etag is a unique attribute of the value returned by put_object.
print('ETag: {0}'.format(result.etag))
# HTTP response header
```

```
print('date: {0}'.format(result.headers['date']))
```

Upload bytes

You can run the following code to upload bytes:

```
# -*- coding: utf-8 -*-

import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.put_object('<yourObjectName>', b'content of object')
```

Upload Unicode

You can run the following code to upload Unicode:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.put_object('<yourObjectName>', u'content of object')
```

Upload local files

You can run the following code to upload a local file:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
```

```
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Files must be opened in binary mode because the number of bytes
included in the file is required for upload.
with open('<yourLocalFile>', 'rb') as fileobj:
    # The seek method specifies that the read or write operations
start from the 1000th byte. The upload starts from the 1000th byte
until the end of the file.
    fileobj.seek(1000, os.SEEK_SET)
    # The tell method is used to return to the current location.
    current = fileobj.tell()
    bucket.put_object('<yourObjectName>', fileobj)
```

OSS Python SDK provides a more convenient method to upload local files:

```
bucket.put_object_from_file('<yourObjectName>', '<yourLocalFile>')
```

Upload a file stream

You can run the following code to upload a file stream:

```
# -*- coding: utf-8 -*-
import oss2
import requests

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# The requests.get method returns an Iterable object, and the Python
SDK uploads the file stream in Chunked Encoding method.
input = requests.get('http://www.aliyun.com')
bucket.put_object('<yourObjectName>', input)
```

3.6.3 Append upload

This topic describes how to use append upload.

Run the following code for append upload:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
```

```
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Set the position (the Position parameter) where the object is
appended to 0.
result = bucket.append_object('<yourObjectName>', 0, 'content of first
 append')
# If the object is not uploaded for the first time, you can use the
bucket.head_object method or the next_position attribute returned in
the last append upload to get the append position.
bucket.append_object('<yourObjectName>', result.next_position, '
content of second append')
```

If the object already exists, exceptions occur in the following two scenarios:

- **The file cannot be uploaded with append upload. In this case, an ObjectNotA ppendable exception occurs.**
- **The file can be uploaded with append upload, but the append position does not match the current file length. In this case, a PositionNotEqualToLength exception occurs.**

3.6.4 Resumable upload

You can use resumable upload to split a file you want to upload into several parts and upload them simultaneously. After you have uploaded all parts, you can combine these parts into a complete object. Thus, you can complete uploading the entire object.

You can use the oss2.resumable_upload method to upload a specified file with resumable upload, You can configure the following parameters.

Parameter	Description	Required or not	Default value
bucket	Specifies the bucket name.	Required	None
key	Specifies the object name.	Required	None
filename	Specifies the name of the local file to upload.	Required	None
store	Specifies the directory where the progress infomration is stored.	Optional	Directory '.py-oss-upload' under the HOME directory

Parameter	Description	Required or not	Default value
headers	Specifies the HTTP header.	Optional	None
multipart_threshold	Specifies a object length threshold. If the object length is larger than the threshold, the resumable upload is used.	Optional	10 MB
part_size	Specifies the part size.	Optional	Auto-calculated
progress_callback	Specifies the upload progress callback function.	Optional	None
num_threads	Specifies the number of concurrent upload threads.	Optional	1

Run the following code for resumable upload:

```
# -*- coding: utf-8 -*-
import oss2
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
# When the file length is larger than or be equal to the value of
the optional parameter multipart_threshold (10 MB by default), the
resumable upload is used. If you do not use the store parameter to
specify the directory, a py-oss-upload directory is created under the
HOME directory to save upload progress information.
oss2.resumable_upload(bucket, '<yourObjectName>', '<yourLocalFile>')
```

Python SDK 2.1.0 and later support resumable upload with optional parameters, as shown in the following code:

```
# If you use the store parameter to specify the directory, the
progress information is stored in the specified directory. If you use
the num_threads to set the number of concurrent upload threads, ensure
 that the value of oss2.defaults.connection_pool_size is no less than
the thread number.. The number of concurrent threads is 1 by default.
```

```
oss2.resumable_upload(bucket, '<yourObjectName>', '<yourLocalFile>',
    store=oss2. ResumableStore(root='/tmp'),
    multipart_threshold=100*1024,
    part_size=100*1024,
    num_threads=4)
```

For more information about resumable upload, see *Resumable upload* in the OSS Developer Guide.

Note:

- **Multiple threads are launched in resumable uploads or downloads. Therefore, you do not need to encapsulate multiple upload or download threads when calling the method. Otherwise, data may be transmitted repeatedly.**
- **We recommend you set a larger part size when the network connection is stable. Otherwise, set a smaller part size.**

3.6.5 Multipart upload

This topic describes how to use multipart upload.

To enable multipart upload, perform the following steps:

1. **Initialization (using bucket.init_multipart_upload): Obtain the Upload ID.**
2. **Upload parts (using bucket.upload_part): Upload data parts. You can upload multiple parts concurrently.**
3. **Completing multipart upload (using bucket.complete_multipart_upload): After you have uploaded all parts, combine these parts into a complete object.**

Run the following code for multipart upload:

```
# -*- coding: utf-8 -*-
import os
from oss2 import SizedFileAdapter, determine_part_size
from oss2.models import PartInfo
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

key = '<yourObjectName>'
filename = '<yourLocalFile>'
```

```
total_size = os.path.getsize(filename)
# Use determine_part_size to determine the part size.
part_size = determine_part_size(total_size, preferred_size=100 * 1024)

# Initialize a multipart upload event.
upload_id = bucket.init_multipart_upload(key).upload_id
parts = []

# Upload parts one by one.
with open(filename, 'rb') as fileobj:
    part_number = 1
    offset = 0
    while offset < total_size:
        num_to_upload = min(part_size, total_size - offset)
  # The SizedFileAdapter(fileobj, size) method generates a new object
, and re-calculates the initial append location.
        result = bucket.upload_part(key, upload_id, part_number,
                                    SizedFileAdapter(fileobj,
num_to_upload))
        parts.append(PartInfo(part_number, result.etag))

        offset += num_to_upload
        part_number += 1

# Complete multipart upload.
bucket.complete_multipart_upload(key, upload_id, parts)

# Verify the multipart upload.
with open(filename, 'rb') as fileobj:
    assert bucket.get_object(key).read() == fileobj.read()
```

Note:

We recommend you set a larger part size when the network connection is stable. Otherwise, set a smaller part size.

3.6.6 Upload progress bars

You can use progress bars to indicate the upload or download progress.

For the complete code of progress bars, see *GitHub*.

The following code is used as an example to describe how to view progress information with bucket.put_object:

```
# -*- coding: utf-8 -*-
from __future__ import print_function
import os, sys
import oss2
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
```

```
# If the length of the data to be uploaded cannot be determined, the
value of total_bytes is None.
def percentage(consumed_bytes, total_bytes):
    if total_bytes:
        rate = int(100 * (float(consumed_bytes) / float(total_bytes)))
        print('\r{0}% '.format(rate), end='')
        sys.stdout.flush()
# progress_callback is an optional parameter used to implement
progress bars.
bucket.put_object('<yourObjectName>', 'a'*1024*1024, progress_callback
=percentage)
```

3.6.7 Upload callback

This topic describes how to use upload callback.

For the complete code of upload callback, see *GitHub*.

Run the following code for upload callback:

```
# -*- coding: utf-8 -*-
import json
import base64
import os
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Prepare callback parameters.
callback_dict = {}
# Configure the IP address of the server you want to send the callback
 request to, for example, http://oss-demo.aliyuncs.com:23450 or http
://127.0.0.1:9090.
callback_dict['callbackUrl'] = 'http://oss-demo.aliyuncs.com:23450'
# Configure the value of the Host field carried in the callback
request header, such as oss-cn-hangzhou.aliyuncs.com.
callback_dict['callbackHost'] = 'oss-cn-hangzhou.aliyuncs.com'
# Configure the value of the body field carried in the callback
request.
callback_dict['callbackBody'] = 'filename=${object}&size=${size}&
mimeType=${mimeType}'
# Configure Content-Type for the callback request.
callback_dict['callbackBodyType'] = 'application/x-www-form-urlencoded
'
# The callback parameter is in the JSON format and Base64-encoded.
callback_param = json.dumps(callback_dict).strip()
base64_callback_body = base64.b64encode(callback_param)
# Encoded callback parameters are carried in the request header and
are sent to OSS.
headers = {'x-oss-callback': base64_callback_body}

# Upload and callback.
result = bucket.put_object('<yourObjectName>', 'a'*1024*1024, headers)
```

```
```

The put_object, put_object_from_file, and complete_multipart_upload methods provide upload callback features. For more information about upload callback, see *Upload callback* in the Development Guide.

3.7 Download objects

3.7.1 Overview

This topic is the overview of object download in OSS Python SDK.

OSS Python SDK provides the following download methods:

- *Streaming download*
- *Download to your local file*
- *Range download*
- *Resumable download*

You can view the download progress shown in the *download progress bar*.

3.7.2 Streaming download

If you have a large object to download or it is time-consuming to download the entire object at a time, you can use streaming download. Streaming download enables you to download part of the object each time until you have downloaded the entire object.

Run the following code for streaming download:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
// This example uses the endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# The return value of bucket.get_object is a file-like object, which
is also an iterable object.
object_stream = bucket.get_object('<yourObjectName>')
print(object_stream.read())
```

```
# The get_object interface returns a stream, which must be read by
read() before being used to calculate the CRC checksum of the returned
 object data. Therefore, you must perform CRC verification after
calling the interface.
if object_stream.client_crc ! = object_stream.server_crc:
    print "The CRC checksum between client and server is inconsistent
!"
```

Run the following code to download data to a local file:

```
import shutil

# object_stream is a file-like object. Therefore, you can use the
shutil.copyfileobj method to download data to a local file.
object_stream = bucket.get_object('<yourObjectName>')
with open('<yourLocalFile>', 'wb') as local_fileobj:
    shutil.copyfileobj(object_stream, local_fileobj)
```

Run the following code to copy an object to another object using streaming copy.

```

# object_stream is an iterable object, which can be copied to another
object using streaming copy.
object_stream = bucket.get_object('<yourObjectName>')
bucket.put_object('<yourBackupObjectName>', object_stream)
```

3.7.3 Download to your local file

This topic describes how to download an object from OSS to a local file.

You can run the following code to download the specified OSS object to a local file:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
#  This example uses the endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Download an object to your local file. If the name of the object is
the same as that of your local file, the object will replace the local
 file. If the name of the object is different from that of your local
 file, the object will be downloaded and a new file will be added to
your local device.
```

```
bucket.get_object_to_file('<yourObjectName>', '<yourLocalFile>')
```

3.7.4 Range download

If you only need part of the data in an object, you can use range downloads to download specified content.

Run the following code for range download:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtain the data ranging from byte 0 to the 99th byte of the object
 (including byte 0 and the 99th byte , a total of 100 bytes). If the
 specified value range is invalid, for example, the specified range
includes a negative number, or the specified value is larger than the
object size, the entire object is downloaded.
object_stream = bucket.get_object('<yourObjectName>', byte_range=(0,
99))
```

3.7.5 Resumable download

Large-sized objects may fail to be downloaded because the network is unstable or the program exits abnormally and the entire object needs to be downloaded again. However, the object may still fail to be downloaded after multiple attempts. Therefore, OSS provides resumable download.

The process of resumable download is as follows:

1. **A temporary local file is created. The file name consists of the original object name and a random suffix.**
2. **Read the OSS object based on the range you specified in the Range header in the HTTP request and write the read content to the corresponding position of the temporary file.**
3. **After the download is complete, rename the temporary file as the target file. If the target file already exists, it is overwritten. If the target file does not exist, a new file is created.**

You can use oss2.resumable_download to enable resumable download. It contains the following parameters:

Parameter	Description	Required or optional	Default value
bucket	Specifies the name of a bucket.	Required	None
key	Specifies the name of an object.	Required	None
filename	Specifies the name of a local file. The OSS object is download to the local file.	Required	None
store	Specifies the local file that records the download progress information. Download progress information is stored in the file. If the download fails at a point, the next download starts from the point based on the recorded progress information.	Optional	The .py-OSS-download directory created under the HOME directory
multipart_threshold	Specifies a length threshold for the object size. If the object size exceeds the threshold, resumable download is enabled.	Optional	10 MB
part_size	Specifies the size of a part.	Optional	Auto-calculated
progress_callback	Specifies the download progress callback function	Optional	None

Parameter	Description	Required or optional	Default value
num_threads	Specifies the number of parts you need to download simultaneously.	Optional	1

Run the following code for resumable download:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

oss2.resumable_download(bucket, '<yourObjectName>', '<yourLocalFile>')
```

Python SDK 2.1.0 and later support optional parameters for resumbale download, as shown in the following code:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Make sure that the value of oss2.defaults.connection_pool_size
is no less than the number of the download threads, and the value
of part_size is no less than the value of oss2.defaults.multiget_p
art_size.
oss2.resumable_download(bucket, '<yourObjectName>', '<yourLocalFile>',
  store=oss2.ResumableDownloadStore(root='/tmp'),
  multiget_threshold=20*1024*1024,
  part_size=10*1024*1024,
  num_threads=3)
```

Note:

Do not allow multiple programs (threads) to download the same object to the same target file simultaneously by calling the method. Because the progress information of the threads overwrites each other, and the temporary file names specified by the threads may conflict.

3.7.6 Download progress bars

You can use progress bars to indicate the upload or download progress.

For the complete code of progress bars, see *GitHub*.

The following code is used as an example to describe how to view progress information with bucket.get_object_to_file:

```
# -*- coding: utf-8 -*-
from __future__ import print_function
import os, sys
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
# If the HTTP response header does not carry Content-Length, the value
 of total_bytes is None.
def percentage(consumed_bytes, total_bytes):
    if total_bytes:
        rate = int(100 * (float(consumed_bytes) / float(total_bytes)))
        print('\r{0}% '.format(rate), end='')

        sys.stdout.flush()

# progress_callback is an optional parameter used to implement
progress bar.
bucket.get_object_to_file('<yourObjectName>', '<yourLocalFile>',
progress_callback=percentage)
```

3.8 Manage objects

3.8.1 Overview

This topic is the overview of object management in OSS Python SDK.

You can perform the following operations on objects in a bucket with APIs:

- *Determine whether an object exists*

- *Manage ACL for an object*
- *Manage Object Meta*
- *List objects*
- *Delete an object*
- *Copy an object*
- *Restore an archive object*
- *Manage a symbolic link*

3.8.2 Determine whether a specified object exists

This topic describes how to determine whether a specified object exists.

Run the following code to determine whether a specified object exists:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

exist = bucket.object_exists('<yourObjectName>')
# If the returned value is true, it indicates that the specified
object exists. If the returned value is false, it indicates that the
specified object does not exist.
if exist:
    print('object exist')
else:
    print('object not exist')
```

3.8.3 Manage ACL for an object

The following table describes the permissions included in the Access Control List (ACL) for an object.

Permission	Description	Value
default	The ACL of an object is the same with that of its bucket.	oss2. OBJECT_ACL _DEFAULT

Permission	Description	Value
Private	Only the object owner and authorized users can read and write the object.	oss2. OBJECT_ACL _PRIVATE
Public read	Only the object owner and authorized users can read and write the object. Other users can only read the object. Authorize this permission with caution.	oss2. OBJECT_ACL _PUBLIC_READ
Public read-write	All users can read and write the object. Authorize this permission with caution.	oss2. OBJECT_ACL _PUBLIC_READ_WRITE

The ACL of objects take precedence over that of buckets. For example, if the ACL of a bucket is private, while the object ACL is public read-write, all users can read and write the object. If an object is not configured with an ACL, its ACL is the same as that of its bucket by default.

Configure an ACL for an object

Run the following code to configure an ACL for an object:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Configure the ACL for the object.
bucket.put_object_acl('<yourObjectName>', oss2. OBJECT_ACL_PUBLIC_READ
)
```

Obtain the ACL for an object

You can run the following code to obtain the ACL for an object:

```
# -*- coding: utf-8 -*-
import oss2
```

```
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

print(bucket.get_object_acl('<yourObjectName>').acl)
```

3.8.4 Manage Object Meta

This topic details how to manage Object Meta. For more information, see *Object Meta* in the OSS Developer Guide.

Configure HTTP headers

OSS supports user-defined HTTP headers. Run the following code to configure HTTP headers:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.put_object('<yourObjectName>', '{"age": 1}', headers={'Content-
Type': 'application/json; charset=utf-8'})
```

For more information about HTTP headers, see *RFC2616*.

Configure user-defined Object Meta

You can define user-defined Object Meta for object descriptions. Run the following code to configure user-defined Object Meta:

```
# -*- coding: utf-8 -*-

import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
```

```
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.put_object('', 'a novel', headers={'x-oss-meta-author': 'O.
Henry', 'Content-Type': 'application/json; charset=utf-8'})
```

Note:

Parameters with the x-oss-meta- prefix are used as the headers of user-defined Object Meta. For more information, see *PutObject*.

Modify Object Meta

Run the following code to modify Object Meta:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operation and maintenance. To create a RAM account, log on to
https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.update_object_meta('<yourObjectName>', {'x-oss-meta-author': 'O
. Henry'})
# User-defined Object Meta is cleared each time the bucket.update_obj
ect_meta is called and must be configured again.
bucket.update_object_meta('<yourObjectName>', {'x-oss-meta-price': '
100 dollar'})
```

Run the following code to change Object Meta, such as Content-Type:

```
bucket.update_object_meta('<yourObjectName>', {'x-oss-meta-author': 'O
. Henry'})
# User-defined Object Meta is cleared each time the bucket.update_obj
ect_meta is called and must be configured again.
bucket.update_object_meta('<yourObjectName>', {'Content-Type': 'text/
plain'})
```

Obtain Object Meta

You can use the APIs provided by OSS Python SDK to obtain Object Meta.

Method	Description	Advantage
get_object_meta	Obtains the ETag, Size (object size), and LastModified (the time when the object is last modified) of an object.	Simpler and faster
head_object	Obtains all Object Meta.	None

Run the following code to obtain Object Meta:

```
# -*- coding: utf-8 -*-

import oss2

# We recommend that you do not log on with the AccessKey of an Alibaba
 Cloud account because the Alibaba Cloud account has permissions
 on all APIs in OSS. Instead, we recommend that you log on as an
authorized RAM user to access APIs or perform routine operation and
maintenance. To create a RAM account, log on to https://ram.console.
aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the endpoint China East 1 (Hangzhou). In actual
scenarios, specify your required endpoint.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtain part of Object Meta.
simplifiedmeta = bucket.get_object_meta("<yourObjectName>")
print(simplifiedmeta.headers['Last-Modified'])
print(simplifiedmeta.headers['Content-Length'])
print(simplifiedmeta.headers['ETag'])

# Obtain all Object Meta.
objectmeta = bucket.head_object("<yourObjectName>")
print(objectmeta.headers['Content-Type'])
print(objectmeta.headers['Last-Modified'])
print(objectmeta.headers['x-oss-object-type'])
```

3.8.5 Convert the storage class of an object

OSS provides the following storage classes to cover various data storage scenarios from hot data to cold data: Standard, Infrequent Access (IA), and Archive. This topic describes how to convert the storage class of an object.

For more information about these storage classes, see *#unique_43* and *#unique_77* in the OSS Developer Guide.

The following code provides an example of how to convert the storage class of an object.

- **The following code provides an example of how to convert the storage class of an object from Standard or IA to Archive:**

```
# -*- coding: utf-8 -*-
import oss2
import os

# Security risks may arise if you log on with the AccessKey pair of
an Alibaba Cloud account because the account has permissions on all
API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')

# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements. This example requires a
bucket containing an object of the Standard or IA storage class.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
object_name = '<yourObjectName>'

# Add a header that specifies the storage class. Set the storage
class to Archive.
headers = {'x-oss-storage-class': oss2.BUCKET_STORAGE_CLASS_ARCHIVE}

# Modify the storage class of the object.
bucket.copy_object(bucket.bucket_name, object_name, object_name,
headers)
```

- **The following code provides an example of how to convert the storage class of an object from Archive to IA:**

```
# -*- coding: utf-8 -*-
import oss2
import os
import time

# Security risks may arise if you log on with the AccessKey pair of
an Alibaba Cloud account because the account has permissions on all
API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')

# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements. This example requires a
bucket containing an object of the Archive storage class.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
object_name = '<yourObjectName>'

# Obtain the object metadata.
meta = bucket.head_object(object_name)

# Check whether the storage class of the object is Archive. If the
storage class of the source object is Archive, you must restore the
 object before you can modify the storage class. Wait for about one
minute until the object is restored.
```

```
if meta.resp.headers['x-oss-storage-class'] == oss2.BUCKET_STO
RAGE_CLASS_ARCHIVE:
    bucket.restore_object(object_name)
    while True:
        meta = bucket.head_object(object_name)
        if meta.resp.headers['x-oss-restore'] == 'ongoing-request="
true"':
            time.sleep(5)
        else:
            break

# Add a header that specifies the storage class. Set the storage
class to IA.
headers = {'x-oss-storage-class': oss2.BUCKET_STORAGE_CLASS_IA}

# Modify the storage class of the object.
bucket.copy_object(bucket.bucket_name, object_name, object_name,
headers)
```

For more information about the storage classes and storage fees, see the *#unique_78/unique_78_Connect_42_section_jmi_2z4_hka* section in Billing items.

3.8.6 List objects

Objects are listed alphabetically. You can use OSS Python SDK to list objects in different methods.

Simple list

Run the following code to list 10 objects in a specified bucket.

```
# -*- coding: utf-8 -*-
import oss2
from itertools import islice

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

for b in islice(oss2. ObjectIterator(bucket), 10):
    print(b.key)
```

List objects by a specified prefix

Run the following code to list objects with a specified prefix:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# List objects by a specified prefix 100 objects are listed by default
.
for obj in oss2. ObjectIterator(bucket, prefix = 'img-'):
    print(obj.key)
```

List all objects in a bucket

Run the following code to list all objects in a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Set the delimiter parameter to a slash (/).
for obj in oss2. ObjectIterator(bucket, delimiter = '/'):
 # Determine whether the obj is a directory using the is_prefix method
.
    if obj.is_prefix():  # directory
        print('directory: ' + obj.key)
    else:                # object
        Print ('file: '+ obj. Key)
```

3.8.7 Delete objects

This topic describes how to delete objects.

Warning:

Delete objects with caution because deleted objects cannot be recovered.

Delete an object

Run the following code to delete a single object:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2. Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2. Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

bucket.delete_object('<yourObjectName>')
```

Delete multiple objects

Run the following code to delete multiple objects simultaneously:

```
# Delete 3 objects. You can delete a maximum of 1,000 objects
simultaneously.
result = bucket.batch_delete_objects(['<yourObjectName-a>', '<
yourObjectName-b>', '<yourObjectName-c>'])
# Print the names of deleted objects.
print('\n'.join(result.deleted_keys))
```

3.8.8 Copy objects

You can copy an object from a bucket (source bucket) to another bucket (target bucket) in the same region.

You must note the following limits before copy an object:

- **You must have the read and write permission on the source object.**
- **You cannot copy an object from a region to another region. For example, you cannot copy an object from a bucket in Hangzhou region to a bucket in Qingdao region.**

Simple copy

You can use simple copy for objects smaller than 1 GB. Run the following code for simple copy:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourDestinationBucketName>')

bucket.copy_object('<yourSourceBucketName>', '<yourSourceObjectName>',
 '<yourDestinationObjectName>')
```

Copy a large-sized object

To copy an object larger than 1 GB, you need to use UploadPartCopy. To enable UploadPartCopy, perform the following steps:

1. **Use bucket.init_multipart_upload to initiate an UploadPartCopy task.**
2. **Start UploadPartCopy with bucket.upload_part_copy. Aside from the last part, all parts must be larger than 100 KB.**
3. **Submit the UploadPartCopy task with bucket.complete_multipart_upload.**

Run the following code for UploadPartCopy task

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import PartInfo
from oss2 import determine_part_size

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

src_object = '<yourSourceObjectName>'
dst_object = '<yourDestinationObjectName>'

total_size = bucket.head_object(src_object).content_length
part_size = determine_part_size(total_size, preferred_size=100 * 1024)

# Initiate an UploadPartCopy task
upload_id = bucket.init_multipart_upload(dst_object).upload_id
parts = []

# Copy parts one by one.
part_number = 1
offset = 0
while offset < total_size:
    num_to_upload = min(part_size, total_size - offset)
    byte_range = (offset, offset + num_to_upload - 1)

    result = bucket.upload_part_copy(bucket.bucket_name, src_object,
byte_range,dst_object, upload_id, part_number)
    parts.append(PartInfo(part_number, result.etag))

    offset += num_to_upload
    part_number += 1

# Complete the UploadPartCopy task.
bucket.complete_multipart_upload(dst_object, upload_id, parts)
```

```
```

For more information about UploadPartCopy, see *UploadPartCopy*

3.8.9 Restore an archive object

This topic describes how to restore an archive object.

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

1. An archive object is in the frozen state.
2. After you submit it for restoration, the server restores the object. The object is in the restoring state.
3. You can read the object after it is restored. The restored state of the object lasts one day by default. You can prolong this period to a maximum of seven days. Once this period expires, the object returns to the frozen state.

Run the following code to restore an object:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')
objectName = '<yourObjectName>'

bucket.restore_object(objectName)
```

For more information about the archive storage classes, see *Introduction to storage classes*.

3.8.10 Manage a symbolic link

This topic describes how to manage a symbolic link.

Create a symbolic link

A symbolic link is a special object that maps to an object similar to a shortcut used in Windows.

Run the following code to create a symbolic link:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

objectName = '<yourObjectName>'
symlink = "<yourSymlink>";

bucket.put_symlink(objectName, symlink)
```

For more information about symbolic links, see *PutSymlink*.

Obtain object content for a symbolic link

To obtain a symbolic link, you must have the read permission on it. Run the following code to obtain the object content for a symbolic link:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

symlink = "<yourSymlink>";

bucket.get_symlink(symlink)
```

For more information about symbolic links, see *GetSymlink*.

3.9 Object tagging

3.9.1 Configure object tagging

This topic describes how to configure object tagging.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Note:

- **You can add tags to an object when uploading it or add tags for an uploaded object. If add tags to an object that already has tags, the original tags are overwritten. For more information about adding tags, see** *#unique_87*.
- **To add tags to an object, you must have the permission to call PutObjectTagging.**
- **The Last-Modified value of an object is not updated when its tags are changed.**
- **A tag can contain letters, numbers, spaces, and the following symbols:** + - = . _ : /

Add tags to an object when uploading it

- **Add tags to an object when uploading it by calling put_object.**

  Run the following code to add tags to an object when uploading it by calling put_object:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.headers import OSS_OBJECT_TAGGING

# It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

object_name = '<yourObjectName>'

# Configures the tags in the HTTP headers.
headers = dict()
headers[OSS_OBJECT_TAGGING] = "k1=v1&k2=v2&k3=v3"

# Set the headers of put_object so that the tags are added to the
object to be uploaded.
result = bucket.put_object(object_name, 'content', headers=headers)
print('http response status: ', result.status)
```

```
# View the tags added to the uploaded object.
result = bucket.get_object_tagging('yourObjectName')
for key in result.tag_set.tagging_rule:
    print('tagging key: {}, value: {}'.format(key, result.tag_set.
tagging_rule[key]))
```

- **Add tags to an object when uploading it by calling multipart_upload.**

  Run the following code to add tags to an object when uploading it by calling multipart_upload:

```
# -*- coding: utf-8 -*-
import os
import oss2
from oss2 import SizedFileAdapter, determine_part_size
from oss2.models import PartInfo
from oss2.headers import OSS_OBJECT_TAGGING

# It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

object_name = '<yourObjectName>'
filename = '<yourLocalFile>'

total_size = os.path.getsize(filename)
# Uses the determine_part_size method to determine the part size.
part_size = determine_part_size(total_size, preferred_size=100 *
1024)

# Configures the tags in the HTTP headers.
headers = dict()
headers[OSS_OBJECT_TAGGING] = 'k1=v1&k2=v2'

# Initializes the multipart upload task.
# Set the headers of init_multipart_upload so that the tags are
added to the object to be uploaded.
upload_id = bucket.init_multipart_upload(object_name, headers=
headers).upload_id
parts = []

# Uploads the parts one by one.
with open(filename, 'rb') as fileobj:
    part_number = 1
    offset = 0
    while offset < total_size:
        num_to_upload = min(part_size, total_size - offset)
    # SizedFileAdapter(fileobj, size) generates a new file object
and recalculates the position where the next upload task starts.
        result = bucket.upload_part(object_name, upload_id,
part_number,
                                    SizedFileAdapter(fileobj,
num_to_upload))
        parts.append(PartInfo(part_number, result.etag))
```

```
        offset += num_to_upload
        part_number += 1

# Completes the multipart upload task.
result = bucket.complete_multipart_upload(object_name, upload_id,
parts)
print('http response status: ', result.status)

# Views the tags added to the object.
result = bucket.get_object_tagging(object_name)
for key in result.tag_set.tagging_rule:
    print('tagging key: {}, value: {}'.format(key, result.tag_set.
tagging_rule[key]))

# Verifies the result of the multipart upload task.
with open(filename, 'rb') as fileobj:
    assert bucket.get_object(object_name).read() == fileobj.read()
```

- **Add tags to an object when uploading it by calling append_object.**

  Run the following code to add tags to an object when uploading it by calling append_object:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.headers import OSS_OBJECT_TAGGING

# It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

object_name = '<yourObjectName>'

# Configures the tags in the HTTP headers.
headers = dict()
headers[OSS_OBJECT_TAGGING] = "k1=v1&k2=v2&k3=v3"

# Uploads the object by calling append_object. Set the headers
of append_object so that the tags are added to the object to be
uploaded.
# Only the tags added when the object is uploaded by calling
append_object for the first time take effect.
result = bucket.append_object(object_name, 0, '<yourContent>',
headers=headers)

# Views the tags added to the object.
result = bucket.get_object_tagging('yourObjectName')
for key in result.tag_set.tagging_rule:
```

```
    print('tagging key: {}, value: {}'.format(key, result.tag_set.
tagging_rule[key]))
```

- **Add tags to an object when uploading it by calling resumable_upload.**

  Run the following code to add tags to an object when uploading it by calling resumable_upload:

```
# -*- coding: utf-8 -*-
import oss2
from oss2.headers import OSS_OBJECT_TAGGING

# It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

object_name = '<yourObjectName>'
local_file = '<yourLocalFileName>'

# Configures the tags in the HTTP headers.
headers = dict()
headers[OSS_OBJECT_TAGGING] = "k1=v1&k2=v2&k3=v3"

# An object is uploaded by calling resumable_upload when the object
size is equal to or larger than the value of the optional parameter
 multiple_threshold (10 MB by default). If the store parameter is
not specified, a .py-oss-upload directory is created under HOME to
store breakpoint information.
# Set the headers of resumable_upload so that the tags are added to
the object to be uploaded.


oss2.resumable_upload(bucket, object_name, local_file, headers=
headers)

result = bucket.get_object_tagging(object_name)
for key in result.tag_set.tagging_rule:
    print('object tagging key: {}, value: {}'.format(key, result.
tag_set.tagging_rule[key]))
```

Add tags to an uploaded object or modify the tags added to an object

Run the following code to add tags to an uploaded object or modify the tags added to an object:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.models import Tagging, TaggingRule

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Creates a tagging rule.
rule = TaggingRule()
rule.add('key1', 'value1')
rule.add('key2', 'value2')

# Creates a tag.
tagging = Tagging(rule)

# Adds the tag to the object.
result = bucket.put_object_tagging('<yourObjectName>', tagging)
# Views the returned HTTP status code.
print('http response status:', result.status)
```

Add tags to an object when copying it

You can specify one of the following tagging rules when copying an object:

- **Copy (default): Copy the tags of the source object to the destination object.**
- **Replace: Add the tags specified in the request to the destination object.**

Examples of adding tags to objects of different sizes when copying them are described as follows:

- **Run the following code to add tags to an object smaller than 1 GB when directly copying it:**

```
# -*- coding: utf-8 -*-

import oss2
from oss2.headers import OSS_OBJECT_TAGGING, OSS_OBJECT_TAGGING_C
OPY_DIRECTIVE

# It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Specifies the source object.
src_object_name = '<yourtSrcObjectName>'
# Specifies the destination object 1.
dest_object_name1 = '<yourtDestObjectName1>'
# Specifies the destination object 2.
dest_object_name2 = '<yourtDestObjectName2>'
```

```
# Specifies OSS_OBJECT_TAGGING_COPY_DIRECTIVE to COPY or keeps the
 default value in the HTTP headers, so that the tags added to the
source object is also added to the destination object 1.
headers=dict()
headers[OSS_OBJECT_TAGGING_COPY_DIRECTIVE] = 'COPY'
bucket.copy_object(bucket.bucket_name, src_object_name, dest_objec
t_name1, headers=headers)

# Specifies OSS_OBJECT_TAGGING_COPY_DIRECTIVE to REPLACE in the HTTP
 headers, so that the tags specified in OSS_OBJECT_TAGGING are added
 to the destination object 2.
headers[OSS_OBJECT_TAGGING_COPY_DIRECTIVE] = 'REPLACE'
headers[OSS_OBJECT_TAGGING] = "key4=value4&key5=value5"
bucket.copy_object(bucket.bucket_name, src_object_name, dest_objec
t_name2, headers=headers)

# Views the tags added to the objet src_object_name.
result = bucket.get_object_tagging(src_object_name)
for key in result.tag_set.tagging_rule:
    print('src tagging key: {}, value: {}'.format(key, result.
tag_set.tagging_rule[key]))

# Views the tags added to the destination object 1. The tags are the
 same as thos of the source object.
result = bucket.get_object_tagging(dest_object_name1)
for key in result.tag_set.tagging_rule:
    print('dest1 object tagging key: {}, value: {}'.format(key,
result.tag_set.tagging_rule[key]))

# Views the tags added to the destination object 2. The added tags
are specified in OSS_OBJECT_TAGGING.
result = bucket.get_object_tagging(dest_object_name2)
for key in result.tag_set.tagging_rule:
    print('dest2 object tagging key: {}, value: {}'.format(key,
result.tag_set.tagging_rule[key]))
```

- **Run the following code to add tags to an object larger than 1 GB when copying it by calling multipart_upload:**

```
# -*- coding: utf-8 -*-
import os
import oss2
from oss2 import determine_part_size
from oss2.models import PartInfo
from oss2.headers import OSS_OBJECT_TAGGING

# It is highly risky to log on with the AccessKey of an Alibaba
Cloud account because the account has permissions on all the APIs
in OSS. We recommend that you log on as a RAM user to access APIs
or perform routine operations and maintenance. To create a RAM user
account, log on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
 actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Specifies the source object.
src_object_name = '<yourSrcObject>'
# Specifies the destination object.
dest_object_name = '<yourDestObject>'
```

```
# Obtains the size of the source object.
head_info = bucket.head_object(src_object_name)
total_size = head_info.content_length
print('src object size:', total_size)

# Calls determin_part_size to determine the part size.
part_size = determine_part_size(total_size, preferred_size=100 *
1024)
print('part_size:', part_size)

# Configures the tags in the HTTP headers.
headers = dict()
headers[OSS_OBJECT_TAGGING] = 'k3=v3'

# Initializes a multipart upload task.
# Set the headers of multipart_upload so that the tags are added to
the destination object.
upload_id = bucket.init_multipart_upload(dest_object_name, headers=
headers).upload_id
parts = []

# Uploads the parts one by one.
part_number = 1
offset = 0
while offset < total_size:
    num_to_upload = min(part_size, total_size - offset)
    end = offset + num_to_upload - 1;
    result = bucket.upload_part_copy(bucket.bucket_name, src_object
_name, (offset, end), dest_object_name, upload_id, part_number)
    # Saves thee part information.
    parts.append(PartInfo(part_number, result.etag))

    offset += num_to_upload
    part_number += 1

# Completes the multipart upload task.
result = bucket.complete_multipart_upload(dest_object_name,
upload_id, parts)

# Obtains the metadata of the destination object.
head_info = bucket.head_object(dest_object_name)

# Views the size of the destination object.
dest_object_size = head_info.content_length
print('dest object size:', dest_object_size)

# Compares the size of the destination object with that of the
source object.
assert dest_object_size == total_size

# Views the tags added to the source object.
result = bucket.get_object_tagging(src_object_name)
for key in result.tag_set.tagging_rule:
    print('src tagging key: {}, value: {}'.format(key, result.
tag_set.tagging_rule[key]))

# Views the tags added to the destination object.
result = bucket.get_object_tagging(dest_object_name)
for key in result.tag_set.tagging_rule:
```

```
    print('dest tagging key: {}, value: {}'.format(key, result.
tag_set.tagging_rule[key]))
```

Add tags to a symbolic link object

Run the following code to add tags to a symbolic link object:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.headers import OSS_OBJECT_TAGGING

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

object_name = '<yourObjectName>'
symlink_name = '<yourSymlinkName>'

# Configures the tags in the HTTP headers.
headers = dict()
headers[OSS_OBJECT_TAGGING] = "k1=v1&k2=v2&k3=v3"

# Adds a symbolic link to the object.
# Set the headers of put_symbolic so that the tags are added to the
symbolic link object.
result = bucket.put_symlink(object_name, symlink_name, headers=headers
)
print('http response status: ', result.status)

# Views the tags added to the symbolic link object.
result = bucket.get_object_tagging(symlink_name)
for key in result.tag_set.tagging_rule:
    print('tagging key: {}, value: {}'.format(key, result.tag_set.
tagging_rule[key]))
```

3.9.2 Obtain the tags added to an object

This topic describes how to obtain the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to obtain the tags added to an object:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.models import Tagging, TaggingRule

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtains the tags added to the object.
result = bucket.get_object_tagging('<yourObjectName>')

# Views the tags added to the object.
for key in result.tag_set.tagging_rule:
    print('tagging key: {}, value: {}'.format(key, result.tag_set.
tagging_rule[key]))
```

For more information about how to obtain the tags added to an object, see *#unique_89*.

3.9.3 Delete the tags added to an object

This topic describes how to delete the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to delete the tags added to an object:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.models import Tagging, TaggingRule

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Deletes the tags added to the object.
result = bucket.delete_object_tagging('<yourObjectName>')
print('http response status: ', result.status)
```

For more information about how to delete the tags added to an object, see *#unique_91*.

3.9.4 Object tagging and lifecycle rules

Lifecycle rules can take effect on objects with specified prefixes or tags. You can also specify prefixes and tags as the condition of a lifecycle rule at the same time.

Note:

If you set a specified tag as the condition of a lifecycle rule, the rule applies to an object only when the key and value in the tag of the object both match the specified tag. If you specify a prefix and multiple tags as the condition of a lifecycle rule, the rule applies to an object only when the prefix and tags of the object match the prefix and all tags specified in the rule.

Set tags as the matching condition of a lifecycle rule

Run the following code to set tags as the matching condition of a lifecycle rule:

```
# -*- coding: utf-8 -*-
import oss2
import datetime
from oss2.models import (LifecycleExpiration, LifecycleRule,
                         BucketLifecycle,AbortMultipartUpload,
                         TaggingRule, Tagging, StorageTransition)

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Sets an expiration rule for objects so that objects are expired
three days after they are modified for the last time.
rule1 = LifecycleRule('rule1', 'tests/',
                      status=LifecycleRule.ENABLED,
                      expiration=LifecycleExpiration(days=3))

# Sets an expiration rule for objects so that objects created before
the specified date are expired.
rule2 = LifecycleRule('rule2', 'logging-',
                      status=LifecycleRule.DISABLED,
expiration=LifecycleExpiration(created_before_date=datetime.date(2018
, 12, 12)))

# Sets an expiration rules for parts so that parts are expired 3 days
after they are generated.
rule3 = LifecycleRule('rule3', 'tests1/',
                      status=LifecycleRule.ENABLED,
            abort_multipart_upload=AbortMultipartUpload(days=3))

# Sets an expiration rules for parts so that parts generated before
the specified date are expired.
rule4 = LifecycleRule('rule4', 'logging1-',
```

```
                      status=LifecycleRule.DISABLED,
                      abort_multipart_upload = AbortMultipartUpload(
created_before_date=datetime.date(2018, 12, 12)))

# Sets tags as the matching condition.
tagging_rule = TaggingRule()
tagging_rule.add('key1', 'value1')
tagging_rule.add('key2', 'value2')
tagging = Tagging(tagging_rule)

# Sets a storage class conversion rule for objects so that the storage
 class of an object is converted to Archive 365 days after it is
modified for the last time.
# Different from the preceding rules, rule5 includes tags that are
 specified as the matching condition, so that it only applies to
objects with the key1=value1 and key2=value2 tags at the same time.
rule5 = LifecycleRule('rule5', 'logging2-',
                      status=LifecycleRule.ENABLED,
                      storage_transitions=[StorageTransition(days=356
, storage_class=oss2.BUCKET_STORAGE_CLASS_ARCHIVE)],
                      tagging = tagging)

lifecycle = BucketLifecycle([rule1, rule2, rule3, rule4, rule5])

bucket.put_bucket_lifecycle(lifecycle)
```

View the tags set as the matching condition of a lifecycle rule

Run the following code to view the tags set as the matching condition of a lifecycle rule:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify an
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Views the lifecycle rules set for a bucket.
lifecycle = bucket.get_bucket_lifecycle()

for rule in lifecycle.rules:
    # Views the expiration rules for parts.
    if rule.abort_multipart_upload is not None:
        print('id={0}, prefix={1}, tagging={2}, status={3}, days={4},
created_before_date={5}'
              .format(rule.id, rule.prefix, rule.tagging, rule.
status,
                      rule.abort_multipart_upload.days,
                      rule.abort_multipart_upload.created_before_date))

    # Views the expiration rules for objects
    if rule.expiration is not None:
        print('id={0}, prefix={1}, tagging={2}, status={3}, days={4},
created_before_date={5}'
```

```
                .format(rule.id, rule.prefix, rule.tagging, rule.
status,
                    rule.expiration.days,
                    rule.expiration.created_before_date))
  # Views the storage class conversion rule.
    if len(rule.storage_transitions) > 0:
        storage_trans_info = ''
        for storage_rule in rule.storage_transitions:
            storage_trans_info += 'days={0}, created_before_date={1},
storage_class={2} **** '.format(
                storage_rule.days, storage_rule.created_before_date,
storage_rule.storage_class)

        print('id={0}, prefix={1}, tagging={2}, status={3},,
StorageTransition={4}'
                .format(rule.id, rule.prefix, rule.tagging, rule.
status, storage_trans_info))
```

3.10 Single-connection bandwidth throttling

This topic describes how to ensure other applications have sufficient bandwidth when you perform upload and download operations on OSS. You can include the x-oss-traffic-limit parameter in your requests and set bandwidth limits.

Note:
For more information about the scenarios and precautions of the single-connection bandwidth throttling feature, see *#unique_93* in the OSS Developer Guide.

Configure bandwidth throttling for common uploads and downloads

Use the following code to configure bandwidth throttling when you upload and download objects to and from OSS:

```
# -*- coding: utf-8 -*-

import oss2
from oss2.models import OSS_TRAFFIC_LIMIT

# Security risks may arise if you log on with the AccessKey pair of
 an Alibaba Cloud account because the account has permissions on all
 API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourRequestBucketName>')

object_name = '<yourobjectName>'
local_file_name = '<yourLocalFileName>'
down_file_name = '<yourDownloadFileName>'

# Set the bandwidth limit to 100 KB/s (819200 bit/s) in headers.
```

```
limit_speed = (100 * 1024 * 8)
headers = dict()
headers[OSS_TRAFFIC_LIMIT] = str(limit_speed);

# Configure bandwidth throttling for the object to be uploaded.
result = bucket.put_object_from_file(object_name, local_file_name,
headers=headers)
print('http response status:', result.status)

# Configure bandwidth throttling for the object to be downloaded.
result = bucket.get_object_to_file(object_name, down_file_name,
headers=headers)
print('http response status:', result.status)
```

Configure bandwidth throttling for uploads and downloads that use signed URLs

Use the following code to configure bandwidth throttling when you upload and download objects from and to OSS by using signed URLs.

```
# -*- coding: utf-8 -*-

import oss2
from oss2.models import OSS_TRAFFIC_LIMIT

# Security risks may arise if you log on with the AccessKey pair of
 an Alibaba Cloud account because the account has permissions on all
 API operations in OSS. We recommend that you log on with a RAM user
 account to call API operations or perform routine operations and
 maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses China (Hangzhou) as the endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourRequestBucketName>')

object_name = '<yourobjectName>'
local_file_name = '<yourLocalFileName>'
down_file_name = '<yourDownloadFileName>'

# Set the bandwidth limit to 100 KB/s (819200 bit/s) in request
parameters.
limit_speed = (100 * 1024 * 8)
params = dict()
params[OSS_TRAFFIC_LIMIT] = str(limit_speed);

# Create a signed URL for uploading an object and set the valid period
 of the URL to 60s.
url = bucket.sign_url('PUT', object_name, 60, params=params)
print('put object url:', url)

# Configure bandwidth throttling for the object to be uploaded.
result = bucket.put_object_with_url_from_file(url, local_file_name)
print('http response status:', result.status)

# Create a signed URL for downloading an object and set the valid
period of the URL to 60s.
url = bucket.sign_url('GET', object_name, 60, params=params)
print('get object url:', url)

# Configure bandwidth throttling for the object to be downloaded.
result = bucket.get_object_with_url_to_file(url, down_file_name)
```

```
print('http response status:', result.status)
```

3.11 Data encryption

3.11.1 Client-side encryption

Client-side encryption is performed locally on data before the data is uploaded to OSS.

Notice:

- **Only objects smaller than 5 GB can be encrypted at the client side.**
- **After uploading an object encrypted at the client side, you are not allowed to call the update_objet_meta or *#unique_165* interface to modify the metadata of the object.**

For the complete code of client-side encryption, see *GitHub*.

You can use the following two methods to manage keys:

- **Manually manage keys (RSA).**
- **Use Key Management Service (KMS) to manage keys.**

Using the two methods, you can prevent the keys from being leaked and therefore ensure the security of data on the client. Additionally, others cannot decrypt the data to obtain the original data even the data is leaked.

For detailed information about client-side encryption, see *Introduction to client-side encryption SDK* in the OSS Developer Guide.

Encryption meta information

Parameter	Description	Required or not
x-oss-meta-oss-crypto-key	Specifies the encrypted key, which is a string encoded with Base64 after being encrypted by the RSA algorithm.	Required

Parameter	Description	Required or not
x-oss-meta-oss-crypto-start	Specifies the initial value of the encrypted data, which is generated randomly. The initial value is a string encoded with Base64 after being encrypted by the RSA algorithm.	Required
x-oss-meta-oss-cek-alg	Specifies the encryption algorithm with the following available values: AES, GCM, and NoPadding.	Required
x-oss-meta-oss-wrap-alg	Specifies the encryption algorithm with the following available values: rsa and kms.	Required
x-oss-meta-oss-matdesc	Specifies the description of the Content Encryption Key (CEK) in the JSON format. This parameter does not take effect currently.	Optional
x-oss-meta-unencrypted-content-length	Specifies the length of data before encryption. This parameter is not generated if the content-length is not specified.	Optional
x-oss-meta-unencrypted-content-md5	Specifies the MD5 of the data before encryption. If the MD5 is not specified, this parameter is not generated.	Optional

Upload and download objects with the keys manually managed

You can upload and download objects with the keys manually managed. The code is as follows:

```
# -*- coding: utf-8 -*-
import os
import oss2
```

```
from  oss2.crypto import LocalRsaProvider
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# Create a bucket and use the RSA method to encrypted the data. This
method
bucket = oss2.CryptoBucket(auth,'<yourEndpoint>', '<yourBucketName>',
crypto_provider=LocalRsaProvider())
key = 'motto.txt'
content = b'a' * 1024 * 1024
filename = 'download.txt'
# Upload an object.
bucket.put_object(key, content, headers={'content-length': str(1024 *
1024)})
# Download an object.
result = bucket.get_object(key)
# Perform verification.
content_got = b''
for chunk in result:
    content_got += chunk
assert content_got == content
# Download an object to a local file.
result = bucket.get_object_to_file(key, filename)
# Perform verification.
with open(filename, 'rb') as fileobj:
    assert fileobj.read() == content
os.remove(filename)
```

Upload and download objects with the keys managed by KMS

- **Prerequisites**

  1. **You have activated the Alibaba Cloud Key Management Service.**
  2. **The key ID for your region has been generated.**
  3. **You have created the custom authorization policies as follows: Log on to the RAM console, select > RAM > Policies > Custom Policy > Create Authorization Policy, and associate corresponding users with the policy.**

  Refer to the following format to create a custom policy:

```
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateKey",
        "kms:GenerateDataKey",
        "kms:ListKeys",
        "kms:Encrypt",
        "kms:Decrypt"
      ],
      "Resource": [
        "acs:kms:<yourRegion>:<yourloginUserId>:key/*"
```

```
      ]
    }
  ]
}
```

- **Sample code**

```
# -*- coding: utf-8 -*-
import os
import oss2
from  oss2.crypto import AliKMSProvider
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# Create a bucket and use the KMS method to encrypt the data. This
 method only applies to scenarios where objects are uploaded or
downloaded entirely.
bucket = oss2.CryptoBucket(auth,'<yourEndpoint>', 'liusiman123456
',crypto_provider=AliKMSProvider('<yourAccessKeyId>', '<yourAccess
KeySecret>', '<yourRegion>', '<yourCMK>', '1234'))
key = 'motto.txt'
content = b'a' * 1024 * 1024
filename = 'download.txt'
# Upload an object.
bucket.put_object(key, content, headers={'content-length': str(1024
 * 1024)})
# Download an object.
result = bucket.get_object(key)
# Perform verification.
content_got = b''
for chunk in result:
    content_got += chunk
assert content_got == content
# Download an object to a local file.
result = bucket.get_object_to_file(key, filename)
# Perform verification.
with open(filename, 'rb') as fileobj:
    assert fileobj.read() == content
os.remove(filename)
```

3.11.2 Server-side encryption

OSS supports server-side encryption for uploaded data. This means that when user data is uploaded, OSS encrypts the data and permanently stores the data. Then, when the data is downloaded by a user, OSS automatically decrypts the data, returns the original data to the user, and declares in the header of the response that the data has been encrypted on the server.

The following server-side encryption methods are available for different application scenarios:

- **Server-side encryption that uses CMKs managed by KMS for encryption and decryption (SSE-KMS)**

  When uploading an object, you can use a specified CMK ID or the default CMK managed by KMS to encrypt and decrypt a large amount of data. This method is cost-effective because you do not need to send user data to the KMS server through networks for encryption and decryption.

  Notice:

  API calling fees are incurred if you use CMKs to encrypt or decrypt data.

- **Server-side encryption fully managed by OSS (SSE-OSS)**

  When you upload an object, OSS encrypts the object on the server side by using AES256 fully managed by OSS. In this method, OSS uses AES256 to encrypt each object with an individual key. Furthermore, the individual keys are encrypted by a CMK that is updated periodically for higher security. This method applies to encrypt or decrypt bulk data.

Notice:

- **Only one server-side encryption method can be used for an object at one time.**
- **If you configure server-side encryption for a bucket, you can still configure the encryption method for a single object when uploading or copying it. In this case, the encryption method configured for the object takes precedence. For more information, see *PutObject*.**
- **For more information about server-side encryption, see *Server-side encryption*.**

Configure server-side encryption for a bucket

You can run the following code to configure the default encryption method for a bucket. After the method is configured, all objects that are uploaded to the bucket without their encryption methods configured are encrypted in the configured default encryption method.

```
# -*- coding: utf-8 -*-
import oss2
from oss2.models import ServerSideEncryptionRule

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
```

```
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Configures the encryption method for the bucket. In this example,
AE256 is specified as an example.
rule = ServerSideEncryptionRule()
rule.sse_algorithm = oss2.SERVER_SIDE_ENCRYPTION_AES256
# Configures the CMK ID when the encryption method is set to KMS. To
encrypt data by using a specified CMK, input the specified CMK ID. If
 you use the CMK managed by OSS to encrypt data, set this parameter
to null. If you use the AES256 method to encrypt data, this parameter
must be set to null.
rule.kms_master_keyid = ""

# Configures server-side encryption for the bucket.
result = bucket.put_bucket_encryption(rule)
# Checks the returned HTTP status code.
print('http response code:', result.status)
```

Obtain the server-side encryption settings of a bucket

You can run the following code to obtain the server-side encryption settings of a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Obtains the server-side encryption settings of the bucket.
result = bucket.get_bucket_encryption()
# Prints the obtained server-side encryption settings.
print('sse_algorithm:', result.sse_algorithm)
print('kms_master_keyid:', result.kms_master_keyid) # If the bucket
encryption method is set to AES256, the value of kms_master_keyid is
None.
```

Delete the server-side encryption settings of a bucket

You can run the following code to delete the server-side encryption settings of a bucket:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with the AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM user account, log
on to https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses the China East 1 (Hangzhou) endpoint. Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Deletes the server-side encryption settings of the bucket.
result = bucket.delete_bucket_encryption()
# Checks the returned HTTP status code.
print('http status:', result.status)
```

3.12 Authorized access

This topic describes how to authorize access to users.

Use STS for temporary access authorization

For the complete code of using STS, see *GitHub*.

You can use Security Token Service (STS) for temporary access authorization. For more information about STS, see *Introduction* in the access control API (STS) reference. For more information about accounts and authorization, see in best practices.

You must install the official Python STS client first:

```
pip install aliyun-python-sdk-sts
```

Please use version 2.0.6 and later. Run the following code to upload files with STS temporary authorization:

```
# -*- coding: utf-8 -*-

from aliyunsdkcore import client
from aliyunsdksts.request.v20150401 import AssumeRoleRequest
import json
import oss2

# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
endpoint = 'oss-cn-hangzhou.aliyuncs.com'
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
access_key_id = '<yourAccessKeyId>'
access_key_secret = '<yourAccessKeySecret>'
bucket_name = '<yourBucketName>'
# role_arn is the resource name of the role.
```

```
role_arn = '<yourRoleArn>'

clt = client.AcsClient(access_key_id, access_key_secret, 'cn-hangzhou
')
req = AssumeRoleRequest.AssumeRoleRequest()

# Set the format of returned values to JSON.
req.set_accept_format('json')
req.set_RoleArn(role_arn)
req.set_RoleSessionName('session-name')
body = clt.do_action(req)

# Use the AccessKeyId and AccessKeySecret to apply for a temporary
token from STS.
token = json.loads(body)

# Use the authentication information in the temporary token to
initialize the StsAuth instance.
auth = oss2.StsAuth(token['Credentials']['AccessKeyId'],
                    token['Credentials']['AccessKeySecret'],
                    token['Credentials']['SecurityToken'])

# Use the StsAuth instance to initialize the bucket.
bucket = oss2.Bucket(auth, endpoint, bucket_name)

# Upload a string.
bucket.put_object('object-name.txt', b'hello world')
```

Sign a URL to authorize temporary access

You can provide a signed URL to a visitor for temporary access. When you sign a URL, you can specify the expiration time for a URL to restrict the period of access from visitors.

Run the following code to download objects with a signed URL:

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

# Set the expiration time of the  URL to one hour.
print(bucket.sign_url('GET', '<yourObjectName>', 60))
```

3.13 Enable logging in Python SDK

OSS Python SDK provides a logging function to easily track problems. This function is disabled by default.

With this function, you can locate and collect log information about OSS operations and save the information as log files in local disks.

Note:
The logging function is only available in OSS Python SDK 2.6.x and later.

The format of a log file is as follows:

```
<time><name><level><threadId><message>
```

Log levels can be arranged in the following order of severity: CRITICAL > ERROR > WARNING > INFO > DEBUG > NOTSET.

Note:
After you specify a log level, only log information of levels that are equal to or higher than the specified level is recorded in local log files. For example, if you specify the log level as INFO, log information of levels CRITICAL, ERROR, WARNING, and INFO, is recorded in log files.

Enable logging in Python SDK

To enable the logging function in the OSS Python SDK, run the following code:

```
# -*- coding: utf-8 -*-

import os
import logging
import oss2
from itertools import islice

# Initialize information such as AccessKeyId, AccessKeySecret, and
Endpoint.
# Replace <AccessKeyId>, <AccessKeySecret>, <Bucket>, and <Endpoint>
with actual values.
access_key_id = '<AccessKeyId>'
access_key_secret = '<AccessKeySecret>'
bucket_name = '<Bucket>'
endpoint = '<Endpoint>'

log_file_path = "log.log"
# Enable the logging function.
oss2.set_file_logger(log_file_path, 'oss2', logging.INFO)
```

```
# Create a bucket.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret),
endpoint, bucket_name)

# Traverse all objects in the bucket.
for b in islice(oss2. ObjectIterator(bucket), 10):
    print(b.key)
# Obtain the object meta.
object_meta = bucket.get_object_meta('object')
```

The recorded log information is as follows:

```
2018-11-20 16:37:46,325 oss2.api [INFO] 26716 : Init oss bucket,
endpoint: oss-cn-shenzhen.aliyuncs.com, isCname: False, connect_ti
meout: None, app_name: , enabled_crc: True
2018-11-20 16:37:46,326 oss2.api [INFO] 26716 : Start to List objects
, bucket: funrily-hello, prefix: , delimiter: , marker: , max-keys:
100
2018-11-20 16:37:46,430 oss2.api [INFO] 26716 : List objects done,
req_id: 5BF3C7DA236B3A201CE645F7, status_code: 200
2018-11-20 16:37:46,430 oss2.api [INFO] 26716 : Start to get object
metadata, bucket: bucket-hello, key: object
2018-11-20 16:37:46,437 oss2.api [ERROR] 26716 : Exception: {
    'status': 404,
    'x-oss-request-id': '5BF3C7DA236B3A201CE64679',
    'details': {
     'HostId': 'bucket-hello.oss-cn-shenzhen.aliyuncs.com',
        'Message': 'The specified key does not exist.',
        'Code': 'NoSuchKey',
        'RequestId': '5BF3C7DA236B3A201CE64679',
        'Key': 'object'
    }
}
```

According to the preceding log information, all objects in a bucket are traversed when the bucket is initialized. If a requested object does not exist, a `NoSuchKey` error occurs.

3.14 Image processing

Image Processing (IMG) is a massive, secure, cost-effective, and highly reliable image processing service. After source images are uploaded to OSS, you can process images on any Internet device at any anytime and from anywhere through simple RESTful APIs.

For more information about IMG, see *Image Processing*.

For the complete code of IMG, see *GitHub*.

Basic features

OSS provides the following IMG features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**
- *Image processing*
- *Image processing access rules*: **calls multiple IMG functions.**

Usage

IMG uses standard HTTP GET requests. You can configure IMG parameters in the QueryString included in the URL.of an image.

If the ACL for an image is private read and write, only authorized users are allowed can access the image.

- **Anonymous access**

  You can use a level-3 domain name in the following format to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Specifies the name of a bucket.
endpoint	Specifies the endpoint used to access the region that the bucket is created.
object	Specifies the name of an image object.
image	Specifies the reserved identifier for IMG.
action	Specifies the operation performed on an image, such as scaling, cropping, and rotating.

Parameter	Description
param	Specifies the parameter that indicates the operation performed on an image.

- **Basic operations**

  For example, scale an image to a width of 100 px and adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Customize image styles**

  You can use a level-3 domain name in the following format to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=style/<yourStyleName>
```

  - **style: specifies the reserved identifier of a customized image style.**
  - **yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.**

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=style/oss-pic-style-w-100
```

- **Cascade operations**

  Cascade operations allow you to perform multiple operations on an image in sequence.

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss
-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<
yourParamValue2>/...
```

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100/rotate,90
```

- **Access through HTTPS**

IMG allows you to access images through the HTTPS protocol. For example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- Authorized access

  Authorized access allows you to customize an image style, access images through the HTTPS protocol, and perform cascade operations.

  Run the following code to generate a signed URL for IMG:

```
# -*- coding: utf-8 -*-
import oss2

# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 strongly recommend that you create a RAM user and use it for API
access and daily operations and maintenance. Log on to https://ram.
console.aliyun.com to create a RAM user.
access_key_id = '<yourAccessKeyId>'
access_key_secret = '<yourAccessKeySecret>'
bucket_name = '<yourBucketName>'
key = 'example.jpg'

# Creates a bucket instance. All object-related methods must be
called by the bucket instance.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret),
endpoint, bucket_name)

# Uploads a sample image.
bucket.put_object_from_file(key, 'example.jpg')

# Generates a signed URL and sets the expiration time to 10 minutes
. The unit of expiration time is second.
style = 'image/resize,m_fixed,w_100,h_100/rotate,90'
url = bucket.sign_url('GET', key, 10 * 60, params={'x-oss-process':
style})
print(url)
```

- Access image objects with SDK

  You can use OSS SDK to access and process an image object.

  SDK allows you to customize image styles, access images through the HTTPS protocol, and perform cascade operations.

  - Basic operations

    The get_object and get_object_to_file methods support image processing.

    Use the following code to perform basic operations on an image:

```
# -*- coding: utf-8 -*-
```

```
import os
import oss2

# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 strongly recommend that you create a RAM user and use it for API
access and daily operations and maintenance. Log on to https://ram
.console.aliyun.com to create a RAM user.
access_key_id = '<yourAccessKeyId>'
access_key_secret = '<yourAccessKeySecret>'
bucket_name = '<yourBucketName>'
key = 'example.jpg'
new_pic = 'example-new.jpg'

# Creates a bucket instance. All object-related methods must be
called by the bucket instance.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret
), endpoint, bucket_name)

# Uploads a sample image.
bucket.put_object_from_file(key, 'example.jpg')

# Scales the image.
style = 'image/resize,m_fixed,w_100,h_100'
bucket.get_object_to_file(key, new_pic, process=style)

# Crops the image.
style = 'image/crop,w_100,h_100,x_100,y_100,r_1'
bucket.get_object_to_file(key, new_pic, process=style)

# Rotates the image.
style = 'image/rotate,90'
bucket.get_object_to_file(key, new_pic, process=style)

# Sharpens the image.
style = 'image/sharpen,100'
bucket.get_object_to_file(key, new_pic, process=style)

# Adds watermarks.
style = 'image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ'
bucket.get_object_to_file(key, new_pic, process=style)

# Converts the image format.
style = 'image/format,png'
bucket.get_object_to_file(key, new_pic, process=style)

# Deletes the sample image.
bucket.delete_object(key)
# Clears local files.
os.remove(new_pic)
```

- **Customize an image style**

  You can run the following code to customize an image style:

```
# -*- coding: utf-8 -*-
import os
import oss2
```

```
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 strongly recommend that you create a RAM user and use it for API
access and daily operations and maintenance. Log on to https://ram
.console.aliyun.com to create a RAM account.
access_key_id = '<yourAccessKeyId>'
access_key_secret = '<yourAccessKeySecret>'
bucket_name = '<yourBucketName>'
key = 'example.jpg'
new_pic = 'example-new.jpg'

# Creates a bucket instance. All object-related methods must be
called by the bucket instance.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret
), endpoint, bucket_name)

# Uploads a sample image.
bucket.put_object_from_file(key, 'example.jpg')

# Customizes the image style.
style = 'style/oss-pic-style-w-100'

# Processes the image.
bucket.get_object_to_file(key, new_pic, process=style)

# Deletes the sample image.
bucket.delete_object(key)
# Clears local files.
os.remove(new_pic)
```

- **Cascade operations**

  Run the following code to perform cascade operations on an image:

```
# -*- coding: utf-8 -*-
import os
import oss2

# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'
# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all APIs in OSS. We
 strongly recommend that you create a RAM user and use it for API
access and daily operations and maintenance. Log on to https://ram
.console.aliyun.com to create a RAM user.
access_key_id = '<yourAccessKeyId>'
access_key_secret = '<yourAccessKeySecret>'
bucket_name = '<yourBucketName>'
key = 'example.jpg'
new_pic = 'example-new.jpg'

# Creates a bucket instance. All object-related methods must be
called by the bucket instance.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret
), endpoint, bucket_name)

# Uploads a sample image.
bucket.put_object_from_file(key, 'example.jpg')
```

```
# Performs cascade operations.
style = 'image/resize,m_fixed,w_100,h_100/rotate,90'

# Processes the image.
bucket.get_object_to_file(key, new_pic, process=style)

# Deletes the sample image.
bucket.delete_object(key)
# Clears local files.
os.remove(new_pic)
```

Image processing persistence

OSS provides the “saveas” operation for data processing. With this feature, you can save the processing result to a specified bucket as resources and assign the result with a specified key. After the resources are saved, you can access the resources directly by specifying the bucket to speed up resource download. This feature applies to ultra-large image cropping or other high-latency operations.

Run the following code for image processing persistence:

```
# -*- coding: utf-8 -*-
import os
import base64
import oss2

# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
endpoint = 'http://oss-cn-hangzhou.aliyuncs.com'
# It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS.
Instead, we strongly recommend that you create a RAM user and use it
 for API access and daily operations and maintenance. Log on to https
://ram.console.aliyun.com to create a RAM account.
# source_bucket_name and taget_bucket_name are the same bucket.
access_key_id = '<yourAccessKeyId>'
access_key_secret = '<yourAccessKeySecret>'
source_bucket_name = '<yourBucketName>'
taget_bucket_name = '<yourTargetBucketName>'
source_image_name = 'example.jpg'

# Creates a bucket instance. All object-related methods must be called
 by the bucket instance.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret),
endpoint, source_bucket_name)

# Scales the image.
style = 'image/resize,m_fixed,w_100,h_100'
Target_image_name = 'example-resize.jpg'
process = "{0}|sys/saveas,o_{1},b_{2}".format(style,
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(target_image_name))),
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(taget_bucket_name))))
result = bucket.process_object(source_image_name, process)
print(result)

# Crops the image.
style = 'image/crop,w_100,h_100,x_100,y_100,r_1'
```

```
target_image_name = 'example-crop.jpg'
process = "{0}|sys/saveas,o_{1},b_{2}".format(style,
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(target_image_name))),
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(taget_bucket_name))))
result = bucket.process_object(source_image_name, process)
print(result)

# Rotates the image.
style = 'image/rotate,90'
target_image_name = 'example-rotate.jpg'
process = "{0}|sys/saveas,o_{1},b_{2}".format(style,
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(target_image_name))),
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(taget_bucket_name))))
result = bucket.process_object(source_image_name, process)
print(result)

# Sharpens the image.
style = 'image/sharpen,100'
target_image_name = 'example-sharpen.jpg'
process = "{0}|sys/saveas,o_{1},b_{2}".format(style,
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(target_image_name))),
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(taget_bucket_name))))
result = bucket.process_object(source_image_name, process)
print(result)

# Adds watermarks.
style = 'image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ'
target_image_name = 'example-watermark.jpg'
process = "{0}|sys/saveas,o_{1},b_{2}".format(style,
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(target_image_name))),
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(taget_bucket_name))))
result = bucket.process_object(source_image_name, process)
print(result)

# Converts the image format.
style = 'image/format,png'
target_image_name = 'example-formatconvert.png'
process = "{0}|sys/saveas,o_{1},b_{2}".format(style,
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(target_image_name))),
    oss2.compat.to_string(base64.urlsafe_b64encode(oss2.compat.
to_bytes(taget_bucket_name))))
result = bucket.process_object(source_image_name, process)
```

```
print(result)
```

3.15 Chinese character and time

This topic describes the Chinese characters and time used by OSS Python SDK.

Chinese character

Errors may occur if Chinese characters are used in the Python code. Therefore, you must declare the Chinese character encoding at the beginning of the code. For example:

```
# -*- coding: utf-8 -*-
```

- **Data type**

  Python 2.x supports the following two data types:

Data type	Description
str	**Character strings, corresponding to the bytes type in Python 3.x**
unicode	**Unicode streams, of which the length is measured in characters. For example, the length of `u'中文'` is 2.**

  Python 3.x supports the following two data types:

Data type	Description
str	**Character strings, corresponding to the unicode type in Python 2.x**
bytes	**Byte streams, of which the length is measured in bytes. For example, the length of `b'中文'` depends on the encoding, which is 6 if it is encoded with UTF-8.**

- **Input and output conventions**

  The following table describes the conventions of input parameter types for OSS Python SDK.

Input	Data type	Remarks
OSS object name	str	If the input type is bytes, it must be encoded with UTF-8.
Local file name	str, unicode	If the input type is bytes , it must be encoded with UTF-8, such as the yourLocalFile parameter in bucket.get_object _to_file.
Input data stream	bytes	For example, the data parameter in bucket. put_object.

The following table describes the conventions of output parameter types for OSS Python SDK.

Output	Data type	Remarks
Results parsed from the XML	str	For example, a string in the result obtained by bucket.list_object.
Downloaded content	bytes	Python SDK regards that the bytes type is encoded with UTF-8 by default . Therefore, make sure that the Python source file is also encoded with UTF-8.

- **Type conversion functions**

  OSS Python SDK provides three functions for type conversion.

Function	Description
to_bytes	- Converts the unicode type to the str type in Python 2.x. Other types are returned without changes. - Converts the str type to the bytes type in Python 3.x. Other types are returned without changes.
to_unicode	- Converts the str type to the unicode type in Python 2.x. Other types are returned without changes. - Converts the bytes type to the str type in Python 3.x. Other types are returned without changes.
to_string	Acts as to_bytes in Python 2.x. Acts as to_unicode in Python 3.x.

Time

OSS Python SDK converts all timestamp strings retrieved from the server to *Unix Time* **, that is, the number of seconds since UTC 00:00 on January 1, 1970. For example, the last_modified time in the results returned by bucket.get_object is the Unix Time of the int type.**

You can use the datetime.datetime.fromtimestamp() method to convert the time, obtaining the timestamp string.

3.16 Exception handling

OSS Python SDK has three types of exceptions: ClientError, RequestError, and ServerError, which are defined in the oss2.exceptions sub-module.

The following table lists the member variables of exceptions.

Variable	Type	Description
status	int	· ServerError: HTTP status code · ClientError and RequestError: fixed values
request_id	str	· ServerError: request ID returned by the OSS server · ClientError and RequestError: empty string
Code and message	str	Correspond to text text in the Code and Message XML Tags in *OSS error response*.

Example for handling exceptions

The following code demonstrates how to handle exceptions when downloading an object that does not exist, and how to print the HTTP status code and request ID of the error messages.

```
# -*- coding: utf-8 -*-
import oss2

# It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
auth = oss2.Auth('<yourAccessKeyId>', '<yourAccessKeySecret>')
# This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
bucket = oss2.Bucket(auth, 'http://oss-cn-hangzhou.aliyuncs.com', '<
yourBucketName>')

try:
    stream = bucket.get_object('random-key.txt')
except oss2.exceptions.NoSuchKey as e:
    print('status={0}, request_id={1}'.format(e.status, e.request_id))
```

ClientError

ClientError is caused by input error on the client side. For example, a ClientError is thrown when an empty object name list is received when the bucket.batch_dele

te_objects method is used. The status value of ClientError objects is oss2.exceptions .OSS_CLIENT_ERROR_STATUS.

RequestError

When an exception is thrown by the underlying HTTP library, the Python SDK will convert the exception to RequestError. The status value of ClientError objects is fixed to oss2.exceptions.OSS_CLIENT_ERROR_STATUS.

ServerError

When an HTTP error code is returned by the OSS server, Python SDK converts it into a ServerError. ServerError is divided into multiple subclasses based on the HTTP status code and OSS error code. The NotFound subclass corrresponds to all 404 exceptions, and the Conflict subclass corresponds to all 409 exceptions.

The following table lists common error codes.

Exception class	HTTP status code	OSS error code	Description
NotModified	304	Null	It is not modified.
AccessDenied	403	AccessDenied	The access is denied.
NoSuchBucket	404	NoSuchBucket	The bucket does not exist.
NoSuchKey	404	NoSuchKey	The file does not exist.
NoSuchUpload	404	NoSuchUpload	The multipart upload does not exist.
NoSuchWebsite	404	NoSuchWebsiteConfiguration	The static website hosting is not configured.
NoSuchLifecycle	404	NoSuchLifecycle	The lifecycle management is not configured.
NoSuchCors	404	NoSuchCORSConfiguration	CORS is not configured.
BucketNotEmpty	409	BucketNotEmpty	The bucket is not empty.

Exception class	HTTP status code	OSS error code	Description
PositionNotEqualToLength	409	PositionNotEqualToLength	The Append position is not equal to the object length.
ObjectNotAppendable	409	ObjectNotAppendable	The object is not appendable.

4 Go

4.1 Preface

This topic describes the sample code of the OSS Go SDK in various use cases.

SDK source code and API documents

For the source code of the OSS Go SDK, visit *GitHub*. For more information, see *OSS Go SDK API documents*.

Sample programs

The OSS Go SDK provides a variety of sample programs for your reference or use. The following table describes the sample programs provided.

Sample file	Content
new_bucket.go	*Initialization*
put_object.go	**Upload objects, including** *Simple upload* **and** *Resumable upload*
append_object.go	*Append upload*
get_object.go	**Download objects, including** *Streaming download*, *Conditional download*, **and** *Download a compressed object*
delete_object.go	*Delete objects*
copy_object.go	*Copy objects within the same bucket*, *Copy objects*, **and** *Copy objects*
list_objects.go	*List objects*, **including listing objects with a specified prefix and listing a specified number of objects**
object_meta.go	*Manage Object Meta*
object_acl.go	*Manage ACL for an object*
sign_url.go	*Authorized access*
cname_sample.go	*Initialization*
create_bucket.go	*Create a bucket*

Sample file	Content
list_buckets.go	*Bucket* based on default or specified parameters
bucket_acl.go	*Bucket*
bucket_referer.go	Add, query, and delete the *Hotlink protection* configurations for a bucket
bucket_logging.go	Add, query, and delete the *Set logging* configurations for a bucket
bucket_lifecycle.go	Add, query, and delete the *Manage lifecycle rules* for a bucket
bucket_cors.go	Add, query, and delete the *CORS* configurations for a bucket
object_tagging.go	*Configure object tagging*, *Obtain the tags added to an object*, *Delete the tags added to an object*, and *Object tagging and lifecycle rules*.
bucket_requestpayment.go	Enable the *Pay-by-requester mode*
bucket_encryption.go	Add, query, and delete the *Server-side encryption* configurations for a bucket

4.2 Installation

Preparation

- **Environment requirements**

  You must use Golang 1.6 and later versions.

  Refer to *Installing Go from source* to download and install the compiling and running environment of Go. After Go is installed, set the GOPATH to the path of your code. For more information about GOPATH, run `go help gopath`.

- **Check the version of Go**

  Run `go version` to check the version of Go.

Download SDK

- *Download from GitHub*
- *Download previous versions*

Install SDK

Run the following command to install OSS Go SDK:

```
go get github.com/aliyun/aliyun-oss-go-sdk/oss
```

Note:

No messages are printed to the interface during installation. Please wait. In case of installation time-out, run the command again.

Check SDK version

Run the following code to check the version of OSS Go SDK:

```
package main

import (
  "fmt"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  fmt.Println("OSS Go SDK Version: ", oss.Version)
}
```

4.3 Quick start

This topic describes how to use OSS Go SDK to perform routine operations such as bucket creation, object uploads, and object downloads.

Create a bucket

Run the following code to create a bucket:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func handleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}
func main() {
    // This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
    endpoint := "http://oss-cn-hangzhou.aliyuncs.com"
    // It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
    accessKeyId := "<yourAccessKeyId>"
```

```
    accessKeySecret := "<yourAccessKeySecret>"
    bucketName := "<yourBucketName>"
    // Create an OSSClient instance.
    client, err := oss.New(endpoint, accessKeyId, accessKeySecret)
    if err ! = nil {
        handleError(err)
    }
    // Create a bucket.
    err = client.CreateBucket(bucketName)
    if err ! = nil {
        handleError(err)
    }
}
```

For more information about endpoints, see *Regions and endpoints*.

Upload objects

Run the following code to upload a file to OSS:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func handleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}
func main() {
    // This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
    endpoint := "http://oss-cn-hangzhou.aliyuncs.com"
    // It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
    accessKeyId := "<yourAccessKeyId>"
    accessKeySecret := "<yourAccessKeySecret>"
    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
    localFileName := "<yourLocalFileName>"
    // Create an OSSClient instance.
    client, err := oss.New(endpoint, accessKeyId, accessKeySecret)
    if err ! = nil {
        handleError(err)
    }
    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        handleError(err)
    }
    // Upload a file.
    err = bucket.PutObjectFromFile(objectName, localFileName)
    if err ! = nil {
        handleError(err)
    }
```

```
}
```

Download objects

Run the following code to download an object to local:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func handleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}
func main() {
    // This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
    endpoint := "http://oss-cn-hangzhou.aliyuncs.com"
    // It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
    accessKeyId := "<yourAccessKeyId>"
    accessKeySecret := "<yourAccessKeySecret>"
    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
    downloadedFileName := "<yourDownloadedFileName>"
    // Create an OSSClient instance.
    client, err := oss.New(endpoint, accessKeyId, accessKeySecret)
    if err ! = nil {
        handleError(err)
    }
    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        handleError(err)
    }
    // Download the object.
    err = bucket.GetObjectToFile(objectName, downloadedFileName)
    if err ! = nil {
        handleError(err)
    }
}
```

List objects

Run the following code to list objects in a specified bucket. You can list a maximum of 100 objects by default.

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func HandleError(err error) {
    fmt.Println("Error:", err)
```

```
    os.Exit(-1)
}
func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        HandleError(err)
    }
    // Obtain the bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        HandleError(err)
    }
    // List objects.
    marker := ""
    for {
        lsRes, err := bucket.ListObjects(oss.Marker(marker))
        if err ! = nil {
            HandleError(err)
        }
        // Print the listed objects. A maximum of 100 objects are
listed by default.
        for _, object := range lsRes.Objects {
            fmt.Println("Bucket: ", object.Key)
        }
        if lsRes.IsTruncated {
            marker = lsRes.NextMarker
        } else {
            break
        }
    }
}
```

Delete objects

Run the following code to delete a specified object:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func handleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}
func main() {
    // This example uses endpoint China (Hangzhou). Specify the actual
 endpoint based on your requirements.
    endpoint := "http://oss-cn-hangzhou.aliyuncs.com"
    // It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
    accessKeyId := "<yourAccessKeyId>"
    accessKeySecret := "<yourAccessKeySecret>"
    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
```

```
    // Create an OSSClient instance.
    client, err := oss.New(endpoint, accessKeyId, accessKeySecret)
    if err ! = nil {
        handleError(err)
    }
    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        handleError(err)
    }
    // Delete the object.
    err = bucket.DeleteObject(objectName)
    if err ! = nil {
        handleError(err)
    }
}
```

4.4 Initialization

OSSClient serves as the OSS Java client to manage OSS resources such as buckets and objects.

To create an OSSClient instance, you need to specify an endpoint. For more information about endpoints, see *Regions and endpoints* **and** *Bind a custom domain name*.

The following table describes the parameters that you can configure:

Parameter	Description	Default value
UseCname	Specifies whether the CNAME is used.	No
Timeout	Specifies the request timeout time in seconds, including the connection timeout and socket read-write timeout.	30 seconds for the connection timeout and 60 seconds for the read-write timeout
SecurityToken	Specifies the securityto ken for temporary users	Null
EnableMD5	Specifies whether the MD5 verification is enabled . CRC verification is recommended because it is more efficient than MD5 verification.	No
EnableCRC	Specifies whether CRC verification is enabled.	Yes

Parameter	Description	Default value
Proxy	Specifies the proxy server, such as `http: // 8.8.8.8 : 3128`	Null
AuthProxy	Specifies a proxy server that requires an account and a password to login.	Null

Use an OSS domain to create an OSSClient instance

Run the following code to create an OSSClient instance with a domain assigned by OSS:

```
import "github.com/aliyun/aliyun-oss-go-sdk/oss"

client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
}
```

Use a custom domain (CNAME) to create an OSSClient instance

For the complete code of custome domain, see *GitHub*.

Run the following code to create an OSSClient instance with CNAME:

```
import "github.com/aliyun/aliyun-oss-go-sdk/oss"

// oss.UseCname (true) indicates thath the CNAME is enabled. CNAME
indicates a custom domain bound to a bucket.
client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>", oss.UseCname(true))
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
}
```

Notice:

If the CNAME is enabled, you cannot list buckets because the CNAME is bound to a certain bucket.

Use STS to create an OSSClient instance

Run the following code to create an OSSClient instance with Security Token Service (STS):

```
import "github.com/aliyun/aliyun-oss-go-sdk/oss"
```

```
client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>", oss.SecurityToken("<yourSecurityToken>"))
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
}
```

For more information, see *What is RAM and STS* **and** *Authorized access*.

Set timeout

Run the following code to set connection timeout:

```
import "github.com/aliyun/aliyun-oss-go-sdk/oss"

// oss.Timeout(10, 120) indicates that the HTTP connection timeout
is set to 10 seconds (the default value is 30 seconds) and the HTTP
 read-write timeout is set to 120 seconds (the default value is 60
seconds). The value 0 indicates that timeout never occurs, which is
not recommend.
client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>", oss.Timeout(10, 120))
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
}
```

CRC verification

CRC verification is enabled by default during uploads and downloads to ensure data integrity. Run the following code to disable CRC verification:

Warning:

We recommended that you do not disable CRC verification. If you disable CRC verification, the data integrity during uploads and downloads is not guaranteed.

```
import "github.com/aliyun/aliyun-oss-go-sdk/oss"

client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>", oss.EnableCRC(false))
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
```

```
}
```

4.5 Buckets

4.5.1 Bucket

A bucket serves as a container that stores objects. Objects belong to a bucket.

Create a bucket

For the complete code of creating a bucket, see *GitHub*.

Run the following code to create a bucket:

```
package main

 import (
  "fmt"
  "os"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
 )

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Create a bucket. The default storage class and ACL of the created
bucket are set to standard and private read respectively.
 err = client.CreateBucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

For more information about bucket naming rules, see naming conventions in *Basic concepts*. You can specify the ACL and the storage class when you create a bucket, as shown in *Bucket-level permissions* and *Storage class*.

You can run the following code to create a bucket with specified permission:

```
 // Create a bucket and set the bucket ACL to public read (which is
private by default).
 err = client.CreateBucket("<yourBucketName>", oss.ACL(oss.ACLPublicR
ead))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
```

```
 }
```

You can run the following code to creates a bucket of the archive class:

```
 err = client.CreateBucket("<yourBucketName>", oss.StorageClass(oss.
StorageArchive))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

You can run the followign code to create a bucket of the Infrequent Access class:

```
 err = client.CreateBucket("<yourBucketName>", oss.StorageClass(oss.
StorageIA))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

List buckets

For the complete code of listing buckets, see *GitHub*.

- **List all buckets**

  Buckets are displayed in alphabetical order. You can list all buckets or specify a bucket that meets certain conditions.

  Run the following code to list all buckets:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // List all buckets.
 marker := ""
 for {
  lsRes, err := client.ListBuckets(oss.Marker(marker))
  if err ! = nil {
   fmt.Println("Error:", err)
   os.Exit(-1)
  }

  // 100 buckets are listed by default.
```

```
  for _, bucket := range lsRes.Buckets {
   fmt.Println("Bucket: ", bucket.Name)
  }

  if lsRes.IsTruncated {
   marker = lsRes.NextMarker
  } else {
   break
  }
 }
}
```

- **List buckets with a specified prefix**

  Run the following code to list buckets with a specified prefix:

```
// List buckets by a specified prefix.
    lsRes, err = client.ListBuckets(oss.Prefix("<yourBucketPrefix
>"))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Print the bucket list.
    fmt.Println("Buckets with prefix: ", lsRes.Buckets)
    for _, bucket := range lsRes.Buckets {
        fmt.Println("Bucket with prefix: ", bucket.Name)
    }
```

- **List buckets by marker**

  The value of the marker parameter indicates the name of a bucket. Run the following code to list buckets after the specified bucket (the value of marker):

```
// List buckets after the specified marker.
lsRes, err = client.ListBuckets(oss.Marker("<yourBucketMarker>"))
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
}

// Print the bucket list.
fmt.Println("My buckets with marker :", lsRes.Buckets)
for _, bucket := range lsRes.Buckets {
 fmt.Println("Bucket with marker: ", bucket.Name)
}
```

- **List a specified number of buckets**

  Run the following code to list a specified number (maxKeys) of buckets:

```
// Set the number of buckets that can be listed to 500. The default
value is 100. The maximum value is 1,000.
lsRes, err = client.ListBuckets(oss.MaxKeys(500))
if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
}

// Print the bucket list.
```

```
 fmt.Println("My buckets max num:", lsRes.Buckets)
 for _, bucket := range lsRes.Buckets {
  fmt.Println("Bucket with maxKeys: ", bucket.Name)
 }
```

Determine whether a bucket exists

Run the following code to determine whether a specified bucket exists:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Determine whether a bucket exists.
 isExist, err := client.IsBucketExist("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("IsBucketExist result : ", isExist)
}
```

Configure an ACL for a bucket

The ACL of a bucket includes the following permissions.

Permission	Description	Value
Private	The bucket owner and the authorized users can read and write objects in the bucket. Other users cannot perform any operation on the objects.	oss.ACLPrivate
Public read	The bucket owner and the authorized users can read and write objects in the bucket. Other users can only read the objects in the bucket. Authorize this permission with caution.	oss.ACLPublicRead

Permission	Description	Value
Public read-write	All users can read and write objects in the bucket. Authorize this permission with caution.	oss.ACLPublicReadWrite

For more information about ACL, see *access control* in the developer guide. For the complete code of configure ACL for a bucket, see *GitHub*.

Use the following code to configure the ACL for a bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Set the ACL for the bucket to public read.
 err = client.SetBucketACL("<yourBucketName>", oss.ACLPublicRead)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

Obtain the ACL for a bucket

Use the following code to obtain the ACL for a bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
 // Obtain the ACL for the bucket
 aclRes, err := client.GetBucketACL("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("Bucket ACL:", aclRes.ACL)
}
```

Obtain the region of a bucket

Run the following code to obtain the region (or location) of a bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the region of the bucket:
 loc, err := client.GetBucketLocation("<yourBucketName>")
    if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("Bucket Location:", loc)
}
```

Obtain bucket information

Use the following code to obtain bucket information:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
 // Bucket information includes regions (region or location), creation
 dates (CreationDate), permissions (ACL), owners (Owner), and storage
class (StorageClass).
 res, err := client.GetBucketInfo("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("BucketInfo.Location: ", res.BucketInfo.Location)
 fmt.Println("BucketInfo.CreationDate: ", res.BucketInfo.CreationDate)
 fmt.Println("BucketInfo.ACL: ", res.BucketInfo.ACL)
 fmt.Println("BucketInfo.Owner: ", res.BucketInfo.Owner)
 fmt.Println("BucketInfo.StorageClass: ", res.BucketInfo.StorageClass)
 fmt.Println("BucketInfo.ExtranetEndpoint: ", res.BucketInfo.
ExtranetEndpoint)
 fmt.Println("BucketInfo.IntranetEndpoint: ", res.BucketInfo.
IntranetEndpoint)
}
```

Delete a bucket

Before a bucket is deleted, ensure that all objects in the bucket, and fragments that are generated from multipart upload are deleted.

Note:

To delete the fragments that are generated from multipart upload, use Bucket.ListMultipartUploads to list all fragments, and then use Bucket.AbortMultipartUpload to delete the fragments. For more information, see *Multipart upload* **.**

Run the following code to delete a bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Delete the bucket.
 err = client.DeleteBucket("<yourBucketName>")
    if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
    }
```

```
}
```

4.5.2 Bucket tagging

By using the bucket tagging function, you can classify your buckets to manage them. For example, you can specify that only buckets with specified tags are returned in ListBucket operations.

- **Only the bucket owner and authorized RAM users can add tags to a bucket. Otherwise, the 403 Forbidden error is returned with the error code: AccessDeni ed.**
- **You can add a maximum of 20 tags (key-value pairs) to a bucket.**
- **The maximum size of a key is 64 bytes. The key of a tag cannot be null and cannot be prefixed with `http://`, `https://`, or `Aliyun`.**
- **The maximum size of a tag value is 128 bytes. The value of a tag can be null.**
- **The key and value of a tag must be UTF-8 encoded.**
- **If you use PutBucketTagging to add tags to a bucket, the original tags added to the bucket are completely overwritten.**

Add tags to a bucket

You can run the following code to add tags to a bucket:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Initializes the tags.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
```

```
  }
  // Add the tags to the bucket.
  err = client.SetBucketTagging("yourBucketName", tagging)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

Obtain the tags added to a bucket

You can run the following code to obtain the tags added to a bucket:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the tags added to the bucket.
  ret, err := client.GetBucketTagging("yourBucketName")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Prints the number of tags added to the bucket.
  fmt.Println("Tag length: ", len(ret.Tags))
}
```

List buckets with a specified tag

Run the following code to list buckets with a specified tag:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
```

```
  }

  // Searches for the bucket based on TagKey.
  ret, err := client.ListBuckets(oss.TagKey("yourTaggingKey"))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Prints the bucket information.
  for _, bucket := range ret.Buckets{
    fmt.Println("bucket:", bucket)
  }
  // Searches for the bucket based on TagKey and TagValue.
  // TagValue must be used with TagKey. You can leave the value of
TagValue as null. In this case, TagValue is not used as a filter to
search for the bucket.
  res, err := client.ListBuckets(oss.TagKey("yourTaggingKey"), oss.
TagValue("yourTaggingValue"))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Prints the bucket information.
  for _, b := range res.Buckets{
    fmt.Println("bucket:", b)
  }
}
```

Delete the tags added to a bucket

You can run the following code to delete the tags added to a bucket:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Deletes the tags added to the bucket.
  err = client.DeleteBucketTagging("yourBucketName")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
}
```

4.5.3 Manage lifecycle rules

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**
- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**
- **Status: This lifecyle rule takes effect or not.**

The parts uploaded by uploadPart method also support lifecycle rules. The last modification time of an object is the time the multipart upload event is initiated.

For more information about lifecycle management, see *Manage object lifecycle*. For the complete code of lifecycle management, see *GitHub*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
package main

import (
    "fmt"
     "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
```

```
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Lifecycle rule 1 (id: "rule1", enable:true, prefix:"foo/",
expiry:Days 3), indicates that an object with the prefix foo expires
three days after the last modified date.
    rule1 := oss.BuildLifecycleRuleByDays("rule1", "foo/", true, 3)
    rules := []oss.LifecycleRule{rule1}

    // Configure lifecycle rules.
    bucketName := "<yourBucketName>"
    err = client.SetBucketLifecycle(bucketName, rules)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

View lifecycle rules

Run the following code to view lifecycle rules:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // View lifecycle rules.
    lcRes, err := client.GetBucketLifecycle(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Lifecycle Rules:", lcRes.Rules)
}
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
package main

import (
    "fmt"
```

```
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Clear lifecycle rules.
    err = client.DeleteBucketLifecycle(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
```

4.5.4 Pay-by-requester mode

After pay-by-requester mode is enabled for a bucket in Alibaba Cloud OSS, the requester instead of the bucket owner pays the cost of requests and traffic. The bucket owner always pays the cost of storing data. You can enable this feature to share data without paying for all requests and traffic by yourself.

Request methods

- **Access from anonymous users is not allowed**

  If you enable pay-by-requester mode for a bucket, anonymous users are not allowed to access the bucket. Requesters must provide authentication informatio n so that OSS can identify and charge them, but not the bucket owner, for requests and traffic.

  If requesters assume the RAM role of an Alibaba Cloud account to request data, OSS charges this Alibaba Cloud account for such requests and traffic.

- **Requesters must include the x-oss-request-payer request header**

  If you enable pay-by-requester mode for a bucket, requesters must include the x-oss-request-payer:requester request header in a PUT, POST, GET, or HEAD

request. This field indicates that requesters understand that their requests and data downloads incur fees. Otherwise, requests cannot pass authentication.

Bucket owners do not need to include the x-oss-request-payer request header when accessing their own buckets. In this case, bucket owners are requesters who pay for such requests and traffic.

For more information about pay-by-requester mode, see *#unique_48* in the OSS Developer Guide.

Enable pay-by-requester mode

The following code provides an example of how to enable pay-by-requester mode:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Create an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Initialize pay-by-requester mode.
  reqPayConf := oss.RequestPaymentConfiguration{
    Payer: "Requester",
  }

  // Enable pay-by-requester mode for the bucket.
  err = client.SetBucketRequestPayment("<yourBucketName>", reqPayConf)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

Obtain pay-by-requester configurations

The following code provides an example of how to obtain pay-by-requester configurations:

```
package main

import (
  "fmt"
  "os"
```

```
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Create an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
    }

  // Obtain pay-by-requester configurations for the bucket.
  ret, err := client.GetBucketRequestPayment("<yourBucketName>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Display pay-by-requester configurations for the bucket.
  fmt.Println("Bucket request payer:", ret.Payer)
}
```

Specify a third party to pay for access to objects

The following code provides an example of how to specify a third party to pay for access to objects when you use PutObject, ListObjects, GetObject, and DeleteObject:

```
package main

import (
  "fmt"
  "os"
  "io/ioutil"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
  "strings"
)

func main() {
  // Create an OSSClient instance.
  // <Endpoint>: specifies the domain name that is used to access the
bucket.
  // <yourAccessKeyId>  <yourAccessKeySecret>: specify the AccessKey
ID and AccessKey secret that are assigned to third parties to access
resources in OSS.
  payerClient, err := oss.New("<Endpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err ! = nil {
    fmt.Println("New Error:", err)
    os.Exit(-1)
  }

  // Obtain pay-by-requester configurations for the bucket.
    //The bucket has enabled pay-by-requester mode and permissions
have been assigned to a third party.
  payerBucket, err := payerClient.Bucket("<bucketName>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // If the bucket owner enables pay-by-requester mode, requesters
must include the oss.RequestPayer(oss.Requester) field in the request
```

```
. This field indicates that requesters understand that their requests
and data downloads incur fees.
    // If the bucket owner does not enable pay-by-requester mode,
requesters do not need to include the oss.RequestPayer(oss.Requester)
field in the request.
    // For more information about request payment modes of buckets,
see SetBucketRequestPayment.

  // Upload an object. The third party has been authorized to perform
corresponding operations on the bucket.
  key := "youObjectName"
  err = payerBucket.PutObject(key, strings.NewReader("<objectValue
>"), oss.RequestPayer("requester"))
  if err ! = nil {
    fmt.Println("put Error:", err)
    os.Exit(-1)
  }

  // List all objects in the bucket. The third party has been
authorized to perform corresponding operations on the bucket.
  // The third party must include the oss.RequestPayer(oss.Requester)
field in the request header.
  lor, err := payerBucket.ListObjects(oss.RequestPayer(oss.Requester))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Display object names.
  for _, l := range lor.Objects {
    fmt.Println("the Key name is :", l.Key)
  }

  // Download an object. The third party has been authorized to
perform corresponding operations on the bucket.
  body, err := payerBucket.GetObject(key, oss.RequestPayer(oss.
Requester))
  if err ! = nil {
    fmt.Println("Get Error:", err)
    os.Exit(-1)
  }
  // If you do not close the connection after it is used, connection
 leaks may occur. Consequently, no connections are available and the
program cannot run properly.
  defer body.Close()

  // Read and display the obtained content.
  data, err := ioutil.ReadAll(body)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  fmt.Println("data:", string(data))

  // Delete an object. The third party has been authorized to perform
corresponding operations on the bucket.
  err = payerBucket.DeleteObject(key, oss.RequestPayer(oss.Requester))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
}
```

4.5.5 Hotlink protection

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49*. For the complete code of configuring hotlink protection, see *GitHub*.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Add the referer field. Empty referer is not allowed. The
referer field allows question marks (?) and asterisks (*) for wildcard
 use.
    referers := []string{"http://www.aliyun.com",
                         "http://www.???.aliyuncs.com",
                         "http://www. *.com"}
    err = client.SetBucketReferer(bucketName, referers, false)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

Obtain a referer whiltelist

Run the following code to obtain a referer whiltelist:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the referer whitelist.
    refRes, err := client.GetBucketReferer(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Referers: ", refRes.RefererList)
    fmt.Println("AllowEmptyReferer: ", refRes.AllowEmptyReferer)
}
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Clear the referer whitelist. // You cannot clear a referer
whitelist directly. To clear a referer whitelist, you need to create
```

```
the rule that allows an empty referer field and replace the original
rule with the new rule.
    err = client.SetBucketReferer(bucketName, []string{}, true)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

4.5.6 Set logging

You can enable access logs to record bucket access to log files, which are stored in a specified bucket.

The log file format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

For more information about access log files, see *Set access logging*. For the complete code of set logging, see *GitHub*.

Enable access logging

Run the following code to enable bucket access logging:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    targetBucketName := "<yourTargetBucketName>"
    targetPrefix := "<yourTargetPrefix>"

    // Enable access logging for a bucket. targetBucketName is the
bucket that logs are stored, targetPrefix is the prefix of the stored
log files, that is, the path of the log files.
    err = client.SetBucketLogging(bucketName, targetBucketName,
targetPrefix, true)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

View access logging configurations

Run the following code to view the access logging configurations for a bucket:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // View the access logging configurations for a bucket.
    logRes, err := client.GetBucketLogging(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Target Bucket: ", logRes.LoggingEnabled.TargetBucket)
    fmt.Println("Target Prefix: ", logRes.LoggingEnabled.TargetPrefix)
}
```

Disable access logging

Run the following code to disable access logging for a bucket:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    // client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>",
 "<yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Disable access logging.
    err = client.DeleteBucketLogging(bucketName)
```

```
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

4.5.7 Static website hosting

You can set your bucket configuration to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

For more information about static website hosting, see *Static website hosting*.

Configure static website hosting

Run the following code to configure static website hosting:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"

 // Configure static web hosting: The index page is index.html and the
 error page is error.html.
 err = client.SetBucketWebsite(bucketName, "index.html", "error.html")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

View static website hosting configurations

Run the following code to view static website hosting configurations:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
```

```
func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"

 // View static web hosting configurations.
 wsRes, err := client.GetBucketWebsite(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("indexWebsite: ", wsRes.IndexDocument.Suffix)
 fmt.Println("errorWebsite: ", wsRes.ErrorDocument.Key)
}
```

Delete static website hosting configurations

Run the following code to delete static website hosting configurations:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"

 // Delete static website hosting configurations.
 err = client.DeleteBucketWebsite(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
}
```

4.5.8 CORS

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* and *PutBucketcors* in OSS Developer Guide.

For the complete code of CORS, see *GitHub*.

Configure CORS rules

Run the following code to configure CORS rules for the specified bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"

 rule1 := oss.CORSRule{
  AllowedOrigin: []string{"*"},
  AllowedMethod: []string{"PUT", "GET"},
  AllowedHeader: []string{},
  ExposeHeader:  []string{},
  MaxAgeSeconds: 200,
 }

 rule2 := oss.CORSRule{
  AllowedOrigin: []string{"http://www.a.com", "http://www.b.com"},
  AllowedMethod: []string{"POST"},
  AllowedHeader: []string{"Authorization"},
  ExposeHeader:  []string{"x-oss-test", "x-oss-test1"},
  MaxAgeSeconds: 100,
 }

 // Configure CORS rules.
 err = client.SetBucketCORS(bucketName, []oss.CORSRule{rule1, rule2})
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
}
```

Obtain CORS rules

Run the following code to obtain CORS rules:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"

 // Obtain the CORS ruless.
 corsRes, err := client.GetBucketCORS(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("Bucket CORS:", corsRes.CORSRules)
}
```

Delete CORS rules

Run the following code to delete all CORS rules for the specified bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"

 // Delete CORS rules.
 err = client.DeleteBucketCORS(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
```

```
  os.Exit(-1)
 }
}
```

4.6 Upload objects

4.6.1 Overview

In OSS, an object is the basic unit for data operations. OSS Go SDK provides the following file upload methods:

- *Simple upload*: **supports the upload of a file up to 5 GB in size.**
- *Append upload*: **supports the upload of a file up to 5 GB in size.**
- *Resumable upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files. It supports concurrent upload and you can define the size of each part.**
- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files.**

Note:
For more information about the usage scenarios of each upload method, see “Upload files” in OSS Developer Guide.

During the file upload, you can set *Object Meta*, **and view the upload progress in the** *upload progress bar*.

4.6.2 Simple upload

For the complete code of simple upload, see *GitHub*.

Upload a string

You can run the following code to upload a string:

```
package main

import (
 "fmt"
 "os"
 "strings"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
```

```
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Upload a string.
 err = bucket.PutObject("<yourObjectName>", strings.NewReader("
yourObjectValue"))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

Upload a byte array

You can run the following code to upload a byte array:

```
package main

import (
 "fmt"
 "os"
 "bytes"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Upload a byte array.
 err = bucket.PutObject("<yourObjectName>", bytes.NewReader([]byte("
yourObjectValueByteArrary")))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
    }
```

```
}
```

Upload a file stream

You can run the following code to upload a file stream:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Read the local file.
 fd, err := os.Open("<yourLocalFile>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 defer fd.Close()

 // Upload a file stream.
 err = bucket.PutObject("<yourObjectName>", fd)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

Upload a local file

You can run the following code to upload a local file:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
```

```
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Upload a local file.
 err = bucket.PutObjectFromFile("<yourObjectName>", "<yourLocalFile>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

4.6.3 Append upload

You cannot perform copyObject for appended objects. When you call the `AppendObject`, if the target object does not exist, a new appendable object is created. If the target object exists, the uploaded content is appended to the end of the object.

Run the following code to upload a file with append upload:

```
package main

import (
 "fmt"
 "os"
 "strings"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 var nextPos int64 = 0
```

```
 // If the object is appended for the first time, the append position
is 0, and the returned value is the position for the next append. The
position for the next append is the current length of the object.
 nextPos, err = bucket.AppendObject("<yourObjectName>", strings.
NewReader("YourObjectAppendValue1"), nextPos)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Start the second append.
 nextPos, err = bucket.AppendObject("<yourObjectName>", strings.
NewReader("YourObjectAppendValue2"), nextPos)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // You can append content to an object for multiple times.
}
```

You can specify the Object Meta only when you append an object for the first time (that is, the append postion is 0).

```
 // Specify the Object Meta in the first append upload.
 nextPos, err = bucket.AppendObject("<yourObjectName>", strings.
NewReader("YourObjectValue"), 0, oss.Meta("MyProp", "MyPropVal"))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

4.6.4 Resumable upload

You can use resumable upload to split a file you want to upload into several parts and upload them simultaneously. After you have uploaded all parts, you can combine these parts into a complete object. Thus, you can complete uploading the entire object.

Current upload progress is recorded into a checkpoint file during the upload. If a part fails to be uploaded during the process, the object is uploaded from the part recorded in the checkpoint file when the upload restarts, that is, the resumable upload. After the upload is complete, the checkpoint is deleted.

Note:

- **The SDK records the upload progress information in the checkpoint file. Therefore, you must ensure that the program has the write permission to the checkpoint file. Do not modify the checkpoint file because it includes verificati on information. If the checkpoint file is corrupted, all parts will be upload again .**

- If the local file is changed during file upload, all the parts will be uploaded again .

You can use `Bucket.UploadFile` to achieve resumable upload. You can configure the following parameters.

Parameter	Description
objectKey	Specifies the name of the object uploaded to OSS.
filePath	Specifies the path of the local file.
partSize	Specifies the upload part size, ranging from 100 KB to 5 GB.
options	Specifies the options, including: - Routines: Specifies the number of parts that are uploaded at the same time. The default value is 1, which means that parts are not uploaded concurrently. - Checkpoint: Specifies whether the resumable upload is enabled and configures the checkpoint file. Resumable upload is disabled by default. For example, `oss.Checkpoint(true, "")` indicates that the resumable upload is enabled, and the checkpoint file is the `file.cp` stored in the directory that the local file needs to be uploaded is stored. The `file` is the name of the local file. You can also use `oss.Checkpoint(true, "your-cp-file.cp")` to specify the checkpoint file. - Other metadata information: See *Manage Object Meta*.

Run the following code for resumable upload:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
```

```
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // The part size is 100 KB. Parts are uploaded through three
coroutines concurrently. The resumable upload is enabled.
 // <yourObjectName>" is the objectKey, "LocalFile" is the filePath,
and 100*1024 is the partSize.
 err = bucket.UploadFile("<yourObjectName>", "LocalFile", 100*1024,
oss.Routines(3), oss.Checkpoint(true, ""))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

4.6.5 Multipart upload

This topic describes how to upload an object in multiple parts

To enable multipart upload, perform the following steps:

1. **Initiate a multipart upload event.**

   You can call Bucket.InitiateMultipartUpload to return the globally unique uploadId created in OSS.

2. **Start the multipart upload task.**

   You can call Bucket.UploadPart to upload data in multiple parts.

> **Notice:**
>
> - **Parts with a same uploadId are sequenced by their part numbers. If you have uploaded a part and use the same part number to upload another part, the later part will replace the former part.**
> - **The MD5 values of a data part is included in the ETag header and is returned to the user.**

- **OSS SDK automatically configures Content-MD5. OSS calculates the MD5 value of the uploaded data and compares it with the MD5 value calculated by SDK. If the two values are different, the InvalidDigest error code is returned.**

3. **Complete the multipart upload task.**

   After you upload all parts, call Bucket.CompleteMultipartUpload to combine these parts into a complete object.

The following code is used as a complete example that describes the process of multipart upload:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func main() {
    // Creates an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
    locaFilename := "<yourLocalFilename>"
    // Obtains a bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    chunks, err := oss.SplitFileByPartNum(locaFilename, 3)
    fd, err := os.Open(locaFilename)
    defer fd.Close()
    // Step 1: Initiates a multipart upload event.
    imur, err := bucket.InitiateMultipartUpload(objectName)
    // Step 2: Uploads parts.
    var parts []oss.UploadPart
    for _, chunk := range chunks {
        fd.Seek(chunk.Offset, os.SEEK_SET)
        // Calls UploadPart to upload each part.
        part, err := bucket.UploadPart(imur, fd, chunk.Size, chunk.
Number)
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }
        parts = append(parts, part)
    }
    // Step 3: Completes the multipart upload task.
    cmur, err := bucket.CompleteMultipartUpload(imur, parts)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
```

```
    fmt.Println("cmur:", cmur)
}
```

Cancel a multipart upload event

Run the following code to cancel a multipart upload event:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func main() {
    // Creates an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Obtains a bucket.
    bucket, err := client.Bucket("<yourBucketName>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Initiates a multipart upload event.
    imur, err := bucket.InitiateMultipartUpload("<yourObjectName>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Cancels the multipart upload event.
    err = bucket.AbortMultipartUpload(imur)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

List uploaded parts

Run the following code to list parts uploaded in a multipart upload event:

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func main() {
    // Creates an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
```

```
    locaFilename := "<yourLocalFilename>"
    uploadID := "<yourUploadID>"
    // Obtains a bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Splits the local file into multiple parts (3 parts in this
example).
    chunks, err := oss.SplitFileByPartNum(locaFilename, 3)
    fd, err := os.Open(locaFilename)
    defer fd.Close()
    // Initiates a multipart upload event.
    imur, err := bucket.InitiateMultipartUpload(objectName)
    uploadID = imur.UploadID
    fmt.Println("InitiateMultipartUpload UploadID: ", uploadID)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Uploads parts.
    var parts []oss.UploadPart
    for _, chunk := range chunks {
        fd.Seek(chunk.Offset, os.SEEK_SET)
        // Calls UploadPart to upload each part.
        part, err := bucket.UploadPart(imur, fd, chunk.Size, chunk.
Number)
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }
        fmt.Println("UploadPartNumber: ", part.PartNumber, ", ETag:
 ", part.ETag)
        parts = append(parts, part)
    }
    // Lists uploaded parts based on InitiateMultipartUploadResult
    lsRes, err := bucket.ListUploadedParts(imur)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Prints the uploaded parts.
    fmt.Println("\nParts:", lsRes.UploadedParts)
    for _, upload := range lsRes.UploadedParts {
        fmt.Println("List PartNumber:  ", upload.PartNumber, ", ETag:
 " ,upload.ETag, ", LastModified: ", upload.LastModified)
    }
    // Generates InitiateMultipartUploadResult based on objectName
 and UploadID, and then list all uploaded parts, which applies to
scenarios where you know the objectName and UploadID.
    var imur_with_uploadid oss.InitiateMultipartUploadResult
    imur_with_uploadid.Key = objectName
    imur_with_uploadid.UploadID = uploadID
    // Lists uploaded parts.
    lsRes, err = bucket.ListUploadedParts(imur_with_uploadid)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Prints the uploaded parts.
    fmt.Println("\nListUploadedParts by UploadID: ", uploadID)
    for _, upload := range lsRes.UploadedParts {
```

```
        fmt.Println("List PartNumber:  ", upload.PartNumber, ", ETag:
 " ,upload.ETag, ", LastModified: ", upload.LastModified)
    }
    // Completes the multipart upload task.
    cmur, err := bucket.CompleteMultipartUpload(imur, parts)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("cmur:", cmur)
}
```

List all part upload events for a bucket

Call `Bucket.ListMultipartUploads` to list all ongoing multipart upload events (events that are initiated but not completed or events that are canceled). You can configure the following parameters:

Parameter	Description
Delimiter	**Specifies a delimiter of a forward slash (/) used to group object names. All objects whose names contain the specified prefix and between which the delimiter occurs for the first time form a group of elements.**
MaxUploads	**Specifies the maximum number of part upload events. The maximum value (also the default value) you can set is 1, 000.**
KeyMarker	**Lists all multipart upload events in which object names start with the letter that follows the KeyMarker value in alphabetical order. This parameter can be used together with UploadIDMarker to specify the start point of the returned result.**
Prefix	**Specifies the prefix that must be included in the returned object name. If you use a prefix for query, the returned object name will contain the prefix.**

Parameter	Description
UploadIDMarker	UploadIDMarker is used together with the KeyMarker parameter to specify the start point of the returned result. · This parameter is ignored if KeyMarker is not configured. · If KeyMarker is configured, the query result contains: - Multipart upload events in which all object names start with the letter that follows the KeyMarker value in alphabetical order. - Multipart upload events in which the object names equal to the value of KeyMarker but the upload IDs of the objects are larger than the value of UploadIDMarker.

· **Use default parameters**

```
package main
import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
func main() {
    // Creates an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Obtains a bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    // Lists all multipart upload events.
    keyMarker := ""
    uploadIdMarker := ""
    for {
        // 1000 records are listed at a time by default.
        lsRes, err := bucket.ListMultipartUploads(oss.KeyMarker(
keyMarker), oss.UploadIDMarker(uploadIdMarker))
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }
        // Prints multipart upload events.
```

```
        for _, upload := range lsRes.Uploads {
            fmt.Println("Upload: ", upload.Key, ", UploadID: ",
upload.UploadID)
        }
        if lsRes.IsTruncated {
            keyMarker = lsRes.NextKeyMarker
            uploadIdMarker = lsRes.NextUploadIDMarker
        } else {
            break
        }
    }
}
```

- **Run the following code to list multipart upload events with a specified prefix:**

```
    lsRes, err := bucket.ListMultipartUploads(oss.Prefix("<
yourObjectNamePrefix>"))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Uploads:", lsRes.Uploads)
```

- **Run the following code to list 100 multipart upload events in maximum at a time:**

```
    lsRes, err := bucket.ListMultipartUploads(oss.MaxUploads(100))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Uploads:", lsRes.Uploads)
```

- **Run the following code to list multipart upload events with a specified prefix and a maximum number:**

```
    lsRes, err := bucket.ListMultipartUploads(oss.Prefix("<
yourObjectNamePrefix>"), oss.MaxUploads(100))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Uploads:", lsRes.Uploads)
```

4.6.6 Progress bars

You can use progress bars to indicate the upload or download progress.

The following code is used as an example to describe how to view progress information with Bucket.PutObjectFromFile:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

// Define a progress bar listener.
```

```
type OssProgressListener struct {
}

// Define the progress event handler.
func (listener *OssProgressListener) ProgressChanged(event *oss.
ProgressEvent) {
 switch event.EventType {
 case oss.TransferStartedEvent:
  fmt.Printf("Transfer Started, ConsumedBytes: %d, TotalBytes %d.\n",
   event.ConsumedBytes, event.TotalBytes)
 case oss.TransferDataEvent:
  fmt.Printf("\rTransfer Data, ConsumedBytes: %d, TotalBytes %d, %d%
%.",
   event.ConsumedBytes, event.TotalBytes, event.ConsumedBytes*100/
event.TotalBytes)
 case oss.TransferCompletedEvent:
  fmt.Printf("\nTransfer Completed, ConsumedBytes: %d, TotalBytes %d.\
n",
   event.ConsumedBytes, event.TotalBytes)
 case oss.TransferFailedEvent:
  fmt.Printf("\nTransfer Failed, ConsumedBytes: %d, TotalBytes %d.\n",
   event.ConsumedBytes, event.TotalBytes)
 default:
 }
}

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"
 objectName := "<yourObjectName>"
 localFile := "<yourLocalFile>"

 // Obtain a bucket.
 bucket, err := client.Bucket(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Upload with a progress bar displayed.
 err = bucket.PutObjectFromFile(objectName, localFile, oss.Progress(&
OssProgressListener{}))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

4.7 Download objects

4.7.1 Overview

For the complete code of object download, see *GitHub*.

OSS Go SDK provides the following download methods:

- *Streaming download*
- *Download to your local file*
- *Resumable download*
- *Conditional download*
- *Download a compressed object*

You can view the download progress shown in the *download progress bar*.

4.7.2 Streaming download

Download an object to a stream.

Run the following code to download a specified object to a stream:

```
package main

import (
 "fmt"
 "os"
 "io/ioutil"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Download an object to a stream.
 body, err := bucket.GetObject("<yourObjectName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 // If you do not close the reader after the data is read, connection
 leaks may occur. Consequently, no connection is available and the
program cannot run properly.
 defer body.Close()
```

```
 data, err := ioutil.ReadAll(body)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("data:", string(data))
}
```

Download an object to the local cache

Run the following code to download a specified object to the local cache:

```
package main

import (
 "fmt"
 "os"
 "io"
 "bytes"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Download the object to the local cache.
 body, err := bucket.GetObject("<yourObjectName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 defer body.Close()

 buf := new(bytes.Buffer)
 io.Copy(buf, body)
 fmt.Println("buf:", buf)
}
```

Download an object to a local file stream

Run the following code to download a specified object to a local file stream:

```
package main

import (
 "fmt"
 "os"
 "io"
```

```
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Download an object to a local file stream.
 body, err := bucket.GetObject("<yourObjectName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 defer body.Close()

 fd, err := os.OpenFile("LocalFile", os.O_WRONLY|os.O_CREATE, 0660)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 defer fd.Close()

 io.Copy(fd, body)
}
```

4.7.3 Range download

If you want to download a part of data in an object, you can specify a download range to download only the data that you want.

Run the following code to download data within a specified range:

```
package main

import (
  "fmt"
  "os"
  "io/ioutil"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
  // Obtains the bucket.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains data between the starting byte value of 15 and the
maximum byte value of 35 inclusive (that is, a total of 21 bytes).
  // If the specified value range is invalid, for example, the
specified range includes a negative number, or the specified value is
larger than the object size, the entire object is downloaded.
  body, err := bucket.GetObject("<yourObjectName>", oss.Range(15, 35))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // If you do not close the reader after the data is read, connection
 leaks may occur. Consequently, no connections are available and an
exception occurs.
  defer body.Close()

  data, err := ioutil.ReadAll(body)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  fmt.Println("data:", string(data))
}
```

4.7.4 Download to your local file

This topic describes how to download an object from OSS to a local file.

Run the following code to download a specified object from OSSClient to your local file:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
 // Download an object to your local file.
 err = bucket.GetObjectToFile("<yourObjectName>", "LocalFile")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

4.7.5 Resumable download

Large-sized objects may fail to be downloaded because the network is unstable or the program exits abnormally and the entire object needs to be downloaded again. However, the object may still fail to be downloaded after multiple attempts. Therefore, OSS provides resumable download.

To enable resumable download, you need to split the object you want to download into multiple parts and download them separately. After you have downloaded all parts, these parts will be combined into a complete object.

The progress information of the current download is recorded in a checkpoint file . If a part fails to be download, the next download starts from the part based on the progress information recorded in the checkpoint file, achieving the resumable download. After the object is downloaded, the checkpoint file is deleted.

Note:

- **OSS SDK records download progress information in a checkpoint file. Therefore , make sure that the program has the write permission to the checkpoint file. Do not modify the checkpoint file because it includes verification information. If the checkpoint file for an object is corrupted, the entire object is downloaded again.**
- **If the object being downloaded is modified (the etag of the object changes) during the download, or part files are missing or modified, the entire object is download again.**

You can use `Bucket.DownloadFile` to enable resumable download.

DownloadFileRequest contains the following parameters:

Parameter	Description
objectKey	Specifies the name of the object to be downloaded.

Parameter	Description
filePath	Specifies the path of the local file that you download the object to.
partSize	Specifies the size of a part. The value can be between 1 byte to 5 GB.
options	Specifies download options, including: · Routines: Specifies the number of parts you need to download simultaneously. The default value is 1. · Checkpoint: Specifies whether the resumable upload is enabled and configures the checkpoint file. The resumable upload is disabled by default. For example, `oss.Checkpoint(true, "")` indicates that the resumable download is enabled, and the checkpoint file is `file.cp`, in which `file` is the name of the local file. The checkpoint file is located in the directory where the local file is located. You can use `oss.Checkpoint(true, "your-cp-file.cp")` to specify the checkpoint file. · If you want to set conditions for the download, see *Conditional download*.

Run the following code for resumable download:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
```

```
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Start the resumable download. Three parts are downloaded
simultaneously.
 // <yourObjectName>" is the objectKey, "LocalFile" is the filePath,
and 100*1024 is the partSize.
 err = bucket.DownloadFile("<yourObjectName>", "LocalFile", 100*1024,
oss.Routines(3), oss.Checkpoint(true, ""))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

4.7.6 Conditional download

You can configure one or more conditions for downloads. If the specified conditions are met, the object can be downloaded. If the specified conditions are not met, an error code is returned and the object cannot be downloaded. You can configure the following conditions: You can configure the following conditions:

Parameter	Description	Configuration method
IfModifiedSince	If the specified time is earlier than the time the object is modified, the object can be downloaded. Otherwise, an error (304 Not modified) is returned.	oss.IfModifiedSince
IfUnmodifiedSince	If the specified time is later than or equal to the time the object is modified, the object can be downloaded. Otherwise, an error (412 Precondition failed) is returned.	oss.IfUnmodifiedSince
IfMatch	If the specified ETag matches that of an object, the object can be downloaded. Otherwise, an error (412 Precondition failed) is returned.	oss.IfMatch

Parameter	Description	Configuration method
IfNoneMatch	If the specified ETag does not match that of an object, the object can be downloaded. Otherwise, an error (304 Not modified) is returned.	oss.IfNoneMatch

Both If-Modified-Since and If-Unmodified-Since can exist at the same time as object download conditions. Both If-Match and If-None-Match can exist at the same time as object download conditions.

You can use Bucket.GetObjectDetailedMeta to Obtain the ETag.

```
package main

import (
 "fmt"
 "os"
 "time"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Configure time conditions.
 date := time.Date(2015, time.November, 10, 23, 0, 0, 0, time.UTC)

 // The condition is not met, and the object will not be downloaded.
 err = bucket.GetObjectToFile("<yourObjectName>", "LocalFile", oss.
IfModifiedSince(date))
 if err == nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // The conditions are met, and the object will be downloaded.
 err = bucket.GetObjectToFile("<yourObjectName>", "LocalFile", oss.
IfUnmodifiedSince(date))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
}
```

4.7.7 Download a compressed object

This topic describes how to download a compressed object.

You can download a compressed object. Currently, GZIP is supported for object compression. Bucket.GetObject and Bucket.GetObjectToFile support compression in downloads.

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Download an compressed object.
 err = bucket.GetObjectToFile("<yourObjectName>", "LocalFile.gzip",
oss.AcceptEncoding("gzip"))
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

4.7.8 Progress bars

You can use progress bars to indicate the upload or download progress.

The following code is used as an example to describe how to view progress information with Bucket.GetObjectToFile:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
```

```
// Define a progress bar listener.
type OssProgressListener struct {
}

// Define the progress event handler.
func (listener *OssProgressListener) ProgressChanged(event *oss.
ProgressEvent) {
 switch event.EventType {
 case oss.TransferStartedEvent:
  fmt.Printf("Transfer Started, ConsumedBytes: %d, TotalBytes %d.\n",
   event.ConsumedBytes, event.TotalBytes)
 case oss.TransferDataEvent:
  fmt.Printf("\rTransfer Data, ConsumedBytes: %d, TotalBytes %d, %d%
%.",
   event.ConsumedBytes, event.TotalBytes, event.ConsumedBytes*100/
event.TotalBytes)
 case oss.TransferCompletedEvent:
  fmt.Printf("\nTransfer Completed, ConsumedBytes: %d, TotalBytes %d.\
n",
   event.ConsumedBytes, event.TotalBytes)
 case oss.TransferFailedEvent:
  fmt.Printf("\nTransfer Failed, ConsumedBytes: %d, TotalBytes %d.\n",
   event.ConsumedBytes, event.TotalBytes)
 default:
 }
}

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"
 objectName := "<yourObjectName>"
 localFile := "<yourLocalFile>"

 // Obtain the bucket.
 bucket, err := client.Bucket(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Download with a progress bar displayed.
 err = bucket.GetObjectToFile(objectName, localFile, oss.Progress(&
OssProgressListener{}))
 if err ! = nil {
 fmt.Println("Error:", err)
 os.Exit(-1)
 }
 fmt.Println("Transfer Completed.")
}
```

4.8 Manage objects

4.8.1 Overview

You can manage objects stored in buckets with APIs, including the following operations:

- *Determine whether an object exists*
- *Manage ACL for an object*
- *Manage Object Meta*
- *List objects*
- *Delete objects*
- *Copy objects*
- *Restore an archive object*
- *Manage a symbolic link*

4.8.2 Determine whether a specified object exists

This topic describes how to determine whether a specified object exists

Run the following code to determine whether a specified object exists:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Determine whether the specified object exists.
 isExist, err := bucket.IsObjectExist("<yourObjectName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("Exist:", isExist)
```

```
}
```

4.8.3 Manage ACL for an object

For the complete code demo of ACL, see *GitHub*.

The following table describes the permissions included in the Access Control List (ACL) for an object.

Permission	Description	Value
default	The ACL of an object is the same with that of its bucket.	oss.ACLDefault
Private	Only the object owner and authorized users can read and write the object.	oss.ACLPrivate
Public read	Only the object owner and authorized users can read and write the object. Other users can only read the object. Authorize this permission with caution.	oss.ACLPublicRead
Public read-write	All users can read and write the object. Authorize this permission with caution.	oss.PublicReadWrite

The ACL privileges of objects take precedence over that of buckets. For example, if the ACL of a bucket is private, while the object ACL is public read-write, all users can read and write the object. If an object is not configured with an ACL, its ACL is the same as that of its bucket by default.

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
 // Obtain the bucket.
 bucket, err := client.Bucket("<yourBucketName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Configure an ACL for an object.
 err = bucket.SetObjectACL("<yourObjectName>", oss.ACLPublicReadWrite)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Obtain the ALC for an object.
 aclRes, err := bucket.GetObjectACL("<yourObjectName>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
 fmt.Println("Object ACL:", aclRes.ACL)
}
```

4.8.4 Manage Object Meta

For more information about Object Meta, see *Object Meta* in OSS Developer Guide. For the complete code of Object Meta, see *GitHub*.

Configure Object Meta

You can configure the Object Meta when uploading an objct. The following table lists the Object Meta you can configure:

Parameter	Description
CacheControl	Speicifies the cache action of a webpage when an object is downloaded.
ContentDisposition	// Specify the name of the object when it is downloaded.
ContentEncoding	Specifies the encoding format when an object is downloaded.
Expires	Specifies the cache expiration time (GMT is used).
ServerSideEncryption	Specifies the server-side encryption algorithm when the OSS creates an object. Valid value: AES256.
ObjectACL	Specifies the ACL for an object created by OSS.

Parameter	Description
Meta	Speicifies the custom Object Meta, with `X-Oss-Meta-` as the prefix.

Run the following code to configure Object Meta:

```
package main

import (
    "fmt"
    "os"
    "time"
    "strings"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Obtain the bucket.
    bucket, err := client.Bucket("<yourBucketName>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Configure Object Meta: The expiration time is 2049-01-10, 23:00
:00 in the GMT format. The ACL is public read. The custom Object Meta
is MyProp (the value is MyPropVal).
    expires := time.Date(2049, time.January, 10, 23, 0, 0, time.UTC
)
    options := []oss.Option{
        oss.Expires(expires),
        oss.ObjectACL(oss.ACLPublicRead),
        oss.Meta("MyProp", "MyPropVal"),
    }

    // Use the data stream to upload.
    err = bucket.PutObject("<yourObjectName>", strings.NewReader("
MyObjectValue"), options...)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Obtain Object Meta.
    props, err := bucket.GetObjectDetailedMeta("<yourObjectName>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Object Meta:", props)
}
```

Modify Object Meta

Run the following code to modify one or more Object Meta at a time:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Modify one Object Meta at a time.
    err = bucket.SetObjectMeta(objectName, oss.Meta("MyMeta", "
MyMetaValue1"))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Modify multiple Object Metas at one time.
    options := []oss.Option{
        oss.Meta("MyMeta", "MyMetaValue2"),
        oss.Meta("MyObjectLocation", "HangZhou"),
    }
    err = bucket.SetObjectMeta(objectName, options...)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Obtain Object Meta.
    props, err := bucket.GetObjectDetailedMeta(objectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Object Meta:", props)
}
```

Obtain Object Meta

Run the following code to obtain Object Meta:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Obtain Object Meta.
    props, err := bucket.GetObjectDetailedMeta(objectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Object Meta:", props)
}
```

4.8.5 Convert the storage class of an object

OSS provides the following storage classes to cover various data storage scenarios from hot data to cold data: Standard, Infrequent Access (IA), and Archive. This topic describes how to convert the storage class of an object.

For more information about these storage classes, see *#unique_43* **and** *#unique_77* **in the OSS Developer Guide.**

The following code provides an example of how to convert the storage class of an object.

- **The following code provides an example of how to convert the storage class of an object from Standard or IA to Archive:**

```
package main

import (
    "fmt"
    "os"

    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"

    // Obtain information about the bucket based on the bucket name.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Modify the storage class of the object. Set the storage class
 to Archive.
    _, err = bucket.CopyObject(objectName, objectName, oss.
ObjectStorageClass(oss.StorageArchive))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

- **The following code provides an example of how to convert the storage class from Archive to Standard:**

```
package main

import (
    "fmt"
    "os"
    "time"

    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
```

```
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"

    // Obtain information about the bucket based on the bucket name.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Check whether the storage class of the source object is
Archive. If the storage class of the source object is Archive, you
must restore the object before you can modify the storage class.
    meta, err := bucket.GetObjectDetailedMeta(objectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    fmt.Println("X-Oss-Storage-Class : ", meta.Get(oss.HTTPHeader
OssStorageClass))
    if meta.Get(oss.HTTPHeaderOssStorageClass) == string(oss.
StorageArchive) {
        // Restore the object.
        err = bucket.RestoreObject(objectName)
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }

        // Wait for about one minute until the object is restored.
        meta, err = bucket.GetObjectDetailedMeta(objectName)
        for meta.Get("X-Oss-Restore") == "ongoing-request=\"true
\"" {
            fmt.Println("X-Oss-Restore:" + meta.Get("X-Oss-Restore
"))
            time.Sleep(1 * time.Second)
            meta, err = bucket.GetObjectDetailedMeta(objectName)
        }
    }

    //Modify the storage class of the restored object. Set the
storage class to Standard.
    _, err = bucket.CopyObject(objectName, objectName, oss.
ObjectStorageClass(oss.StorageStandard))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

For more information about the storage classes and storage fees, see the *#unique_78/unique_78_Connect_42_section_jmi_2z4_hka* **section in Billing items.**

4.8.6 List objects

Objects are listed alphabetically. You can use `Bucket.ListObjects` to list objects in a bucket. The following parameters can be used:

Parameter	Description
delimiter	**Specifies a delimiter used to group objects. The substring between the specified prefix and the first occurrence of the delimiter of a forward slash (/) is commonPrefixes.**
prefix	**Specifies the prefix you configure to list required objects.**
maxKeys	**Specifies the maximum number of objects that can be listed. The default value is 100. The maximum value is 1, 000.**
marker	**Specifies the initial object in the list.**

For the complete code of ListObjects, see *GitHub*.

Simple list

You can use the default parameters to obtain the object list of a bucket. 100 objects can be listed by default. Run the following code to list objects in a bucket:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        HandleError(err)
    }

    // Obtain the bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
```

```
        HandleError(err)
    }

    // List all objects.
    marker := ""
    for {
        lsRes, err := bucket.ListObjects(oss.Marker(marker))
        if err ! = nil {
            HandleError(err)
        }

        // Prints the object list. 100 records are returned by default
.
        for _, object := range lsRes.Objects {
            fmt.Println("Bucket: ", object.Key)
        }

        if lsRes.IsTruncated {
            marker = lsRes.NextMarker
        } else {
            break
        }
    }
}
```

List a specified number of objects

Run the following code to list a specified number of objects:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Configure the maximum number of lthe listed object and list
objects.
    lsRes, err := bucket.ListObjects(oss.MaxKeys(200))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
```

```
    }

    // Print the results.
    fmt.Println("Objects:", lsRes.Objects)
    for _, object := range lsRes.Objects {
        fmt.Println("Object:", object.Key)
    }
}
```

List objects by a specified prefix

Run the following code to list objects with a specified prefix:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // List objects with a specified prefix. 100 objects are listed by
 default.
    lsRes, err := bucket.ListObjects(oss.Prefix("my-object-"))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Print the results.
    fmt.Println("Objects:", lsRes.Objects)
    for _, object := range lsRes.Objects {
        fmt.Println("Object:", object.Key)
    }
}
```

List objects by marker

The marker parameter indicates the name of an object from which the listing begins. Run the following code to specify an object (specified with marker) after which the listing begins:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // List objects placed after the object (specified with marker).
100 objects are listed by default.
    lsRes, err := bucket.ListObjects(oss.Marker("my-object-xx"))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Print the results.
    fmt.Println("Objects:", lsRes.Objects)
    for _, object := range lsRes.Objects {
        fmt.Println("Object:", object.Key)
    }
}
```

List all objects on one or more pages

Run the following code to list all objects on one or more pages. The maximum number of objects listed on each page is specified with maxKeys.

```
package main

import (
    "fmt"
```

```
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // List all objects on multiple pages. 200 objects are listed in
one page.
    marker := oss.Marker("")
    for {
        lsRes, err := bucket.ListObjects(oss.MaxKeys(200), marker)
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }
        marker = oss.Marker(lsRes.NextMarker)

        fmt.Println("Objects:", lsRes.Objects)

        if ! lsRes.IsTruncated {
            break
        }
    }
}
```

List objects by marker on one or more pages

Run the following code to specify an object (specified with marker) after which the listing begins on one or more pages: The maximum number of objects listed on each page is specified with maxKeys.

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
```

```
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // List objects placed after the object (specified with marker) on
 multiple pages. 50 objects are listed in one page.
    marker = oss.Marker("my-object-xx")
    for {
        lsRes, err := bucket.ListObjects(oss.MaxKeys(50), marker)
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }

        marker = oss.Marker(lsRes.NextMarker)

        fmt.Println("Objects:", lsRes.Objects)

        if ! lsRes.IsTruncated {
            break
        }
    }
}
```

List objects by a specified prefix on one or more pages

Run the following code to list objects by a specified prefix on one or more pages:

The maximum number of objects listed on each page is specified with maxKeys.

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
```

```
        os.Exit(-1)
    }

    // List objects with a specified prefix on multiple pages. 80
objects are listed in one page.
    prefix := oss.Prefix("my-object-")
    marker := oss.Marker("")
    for {
        lsRes, err := bucket.ListObjects(oss.MaxKeys(80), marker,
prefix)
        if err ! = nil {
            fmt.Println("Error:", err)
            os.Exit(-1)
        }

        prefix = oss.Prefix(lsRes.Prefix)
        marker = oss.Marker(lsRes.NextMarker)

        fmt.Println("Objects:", lsRes.Objects)

        if ! lsRes.IsTruncated {
            break
        }
    }
}
```

4.8.7 Delete objects

Warning:

Delete objects with caution because deleted objects cannot be recovered.

For the complete code of deleting objects, see *GitHub*.

Delete an object

Run the following code to delete a single object:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"

    // Obtain the bucket.
```

```
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Delete an object.
    err = bucket.DeleteObject(objectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

Delete multiple objects

You can use `Bucket.DeleteObjects` to delete multiple objects and specify the `DeleteObjectsQuiet` parameter to determine whether the result of the delete operation is returned. The results of the deleted objects are returned by default.

Run the following code to delete multiple objects:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // The results of deleted objects are returned.
    delRes, err := bucket.DeleteObjects([]string{"my-object-1", "my-
object-2"})
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println("Deleted Objects:", delRes.DeletedObjects)

    // The deletion results are not returned.
```

```
    _, err = bucket.DeleteObjects([]string{"my-object-3", "my-object-4
"},
    oss.DeleteObjectsQuiet(true))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
}
```

4.8.8 Copy objects

For the complete code of copying objects, see *GitHub*.

Copy objects within the same bucket

Run the following code to copy objects within the same bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"
 objectName := "<yourObjectName>"
 destObjectName := "<yourDestObjectName>"

 // Obtain the bucket.
 bucket, err := client.Bucket(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Copy an object to another object in the same bucket.
 _, err = bucket.CopyObject(objectName, destObjectName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
```

```
}
```

Copy objects to another bucket

You can copy an object from the current bucket to another bucket or from other buckets to the current bucket. Run the following code to copy objects to another bucket:

```
package main

import (
 "fmt"
 "os"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"
 srcBucketName := "<yourSrcBucketName>"
 dstBucketName := "<yourDstBucketName>"
 srcObjectName := "<yourSrcObjectName>"
 dstObjectName := "<yourDstObjectName>"

 // Obtain the bucket.
 bucket, err := client.Bucket(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Copy the source object (srcObjectName) from other buckets (
srcBucketName) to the current bucket.
 bucket.CopyObjectFrom(srcBucketName, srcObjectName, dstObjectName)
 if err ! = nil {
  fmt.Println("CopyObjectFrom Error:", err)
  os.Exit(-1)
 }

 // Copy the source object (srcObjectName) from the current bucket to
another bucket (dstBucketName).
 bucket.CopyObjectTo(dstBucketName, dstObjectName, srcObjectName)
 if err ! = nil {
  fmt.Println("CopyObjectTo Error:", err)
  os.Exit(-1)
 }
```

```
}
```

- **Process object metadata during copy**

  You can process the object metadata when copying the object by specifying the `MetadataDirective` parameter:

  - **oss.MetaCopy: The default value, indicating that the metadata of the target object is the same with that of the source object, that is, copying the source object metadata.**
  - **oss.MetaReplace: Replace the source object metadata with the new metadata.**

  Notice:
  If you specify the MetadataDirective parameter to oss.MetaReplace, the metadata of the source object is overwritten and cannot be recovered. Therefore, perform this operation with caution.

  The following metadata can be specified if the value of `MetadataDirective` is oss.MetaReplace:

Metadata	Description
CacheControl	Specifies the cache action of the web page when the new object is downloaded.
ContentDisposition	Specifies the name of the new object when it is downloaded.
ContentEncoding	Specifies the content encoding format when a new object is downloaded.
Expires	Specify the expiration time (in GMT) of the new object.
ServerSideEncryption	Specifies the server-side encryption algorithm used when the OSS creates an object. Valid value: AES256
ObjectACL	Specifies ACL of an OSS object when it is created.
Meta	Custom parameter prefixed with `X-Oss-Meta-`.

  Run the following code to process object metadata during copy:

```
package main
```

```
import (
 "fmt"
 "os"
 "time"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 bucketName := "<yourBucketName>"
 objectName := "<yourObjectName>"
 destObjectName := "<yourDestObjectName>"

 // Obtain the bucket.
 bucket, err := client.Bucket(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // Specify the metadata of the target object.
 expires := time.Date(2049, time.January, 10, 23, 0, 0, 0, time.UTC)
 options := []oss.Option{
  oss.MetadataDirective(oss.MetaReplace),
  oss.Expires(expires),
  oss.ObjectACL(oss.ACLPublicRead),
  oss.Meta("MyMeta", "MyMetaValue")}

 // Overwrites the metadata of the source object with the specified
metadata.
 _, err = bucket.CopyObject(objectName, destObjectName, options...)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }
}
```

Conditional copy

You can set conditions for object copy. The object is copied only when the conditions are satisfied. Otherwise, an error is reported and the object does not get copied. Available conditions include:

Parameter	Description
CopySourceIfMatch	If the source object’s ETag value is the same as the ETag provided by the user, the copy operation is executed. Otherwise, an error is returned.

Parameter	Description
CopySourceIfNoneMatch	If the source object’s ETag value is not the same as the ETag provided by the user, the copy operation is actioned. Otherwise, an error is returned.
CopySourceIfModifiedSince	If the specified time in the input parameter is the same as or later than the actual modification time of the source object, the object is copied normally. Otherwise, an error is returned.
CopySourceIfUnmodifiedSince	If the source object has been modified after the time specified by the user, the object is copied. Otherwise, an error is returned.

CopySourceIfMatch and CopySourceIfNoneMatch can be used as conditions together. CopySourceIfModifiedSince and CopySourceIfUnmodifiedSince can be used as conditions together.

Run the following code for conditional copy:

```
package main

import (
 "fmt"
 "os"
 "time"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
 }

 // bucketName := "<yourBucketName>"
 // objectName := "<yourObjectName>"
 bucketName := "<yourBucketName>"
 objectName := "<yourObjectName>"
 destMatchObjectName := "<yourMatchDestObjectName>"
 destUnMatchObjectName := "<yourUnmatchDestObjectName>"

 // Obtain the bucket.
 bucket, err := client.Bucket(bucketName)
 if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
```

```
 }

 date := time.Date(2011, time.November, 10, 23, 0, 0, 0, time.UTC)

 // The conditions are satisfied, copy the object.
 _, err = bucket.CopyObject(objectName, destMatchObjectName, oss.
CopySourceIfModifiedSince(date))
 if err ! = nil {
  fmt.Println("CopyObject CopySourceIfModifiedSince Error:", err)
 }

 // The conditions are not met, the object is not copied.
 _, err = bucket.CopyObject(objectName, destUnMatchObjectName, oss.
CopySourceIfUnmodifiedSince(date))
 if err ! = nil {
  fmt.Println("CopyObject CopySourceIfUnmodifiedSince Error:", err)
 }
}
```

4.8.9 Restore an archive object

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

- **An archive object is in the frozen state.**
- **After you submit it for restoration, the server restores the object. The object is in the restoring state.**
- **One minute later, the object enters the restored state and can be read. The restored state lasts for one day by default, and can be prolonged to a maximum of seven days. Once this period ends, the object returns to the frozen state.**

For detailed information about archive storage class, see *Introduction to storage classes* **For detailed information about related status code, see** *RestoreObject*. **For the complete code of restoring an archive object, see** *GitHub*.

Run the following code to restore an object:

```
package main

import (
 "fmt"
 "os"
 "time"
 "strings"
 "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
 fmt.Println("Error:", err)
 os.Exit(-1)
}
```

```
func main() {
 // Create an OSSClient instance.
 client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
 if err ! = nil {
  HandleError(err)
 }

 bucketName := "<yourBucketName>"
 objectName := "<yourObjectName>"
 localFilename := "<yourLocalFilename>"

 // Create a bucket.
 err = client.CreateBucket(bucketName, oss.StorageClass(oss.StorageArc
hive))
 if err ! = nil {
  HandleError(err)
 }

 // Obtain the bucket.
 archiveBucket, err := client.Bucket(bucketName)
 if err ! = nil {
  HandleError(err)
 }

 // Upload the archived object.
 var val = "More than just cloud."
 err = archiveBucket.PutObject(objectName, strings.NewReader(val))
 if err ! = nil {
  HandleError(err)
 }

 // Check whether the object is an archive object.
 meta, err := archiveBucket.GetObjectDetailedMeta(objectName)
 if err ! = nil {
  HandleError(err)
 }

 fmt.Println("X-Oss-Storage-Class : ", meta.Get("X-Oss-Storage-Class
"))
 if meta.Get("X-Oss-Storage-Class") == string(oss.StorageArchive) {
  // Restore the archive object for read.
  err = archiveBucket.RestoreObject(objectName)
  if err ! = nil {
   HandleError(err)
  }

  // Wait until the object is restored.
  meta, err = archiveBucket.GetObjectDetailedMeta(objectName)
  for meta.Get("X-Oss-Restore") == "ongoing-request=\"true\"" {
   fmt.Println("x-oss-restore:" + meta.Get("X-Oss-Restore"))
   time.Sleep(1000 * time.Second)
   meta, err = archiveBucket.GetObjectDetailedMeta(objectName)
  }
 }

 // Download the restored object.
 err = archiveBucket.GetObjectToFile(objectName, localFilename)
 if err ! = nil {
  HandleError(err)
 }

 // Restore the object again.
 err = archiveBucket.RestoreObject(objectName)
```

```
 fmt.Println("ArchiveSample completed")
}
```

4.8.10 Manage a symbolic link

A symbolic link is a special object that maps to an object similar to a shortcut used in Windows. You can configure user-defined Object Meta for symbolic links.

Run the following code to create a symbolic link:

```
package main

import (
    "fmt"
    "os"
    "strings"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    // The symbolic link.
    objectName := "<yourSymlinkName>"
    // The target object that the symbolic link directs to
    targetObjectName := "<yourSymlinkTargetName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Create a symbolic link.
    err = bucket.PutSymlink(objectName, targetObjectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Upload a file to the object that the symbolic link directs to.
    err = bucket.PutObject(targetObjectName, strings.NewReader("target
"))
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    err = bucket.PutSymlink(objectName, targetObjectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
```

```
    }

    // Obtain the content of the object that the symbolic link directs
 to.
    meta, err := bucket.GetSymlink(objectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println(meta.Get(oss.HTTPHeaderOssSymlinkTarget))
}
```

For more information about creating symbolic links, see *PutSymlink*.

Obtain object content for a symbolic link

To obtain a symbolic link, you must have the read permission on it. Run the following code to obtain the object content for a symbolic link:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    bucketName := "<yourBucketName>"
    // Symbolic Link name.
    objectName := "<yourSymlinkObjectName>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }

    // Obtain the content of the object that the symbolic link directs
 to.
    meta, err := bucket.GetSymlink(objectName)
    if err ! = nil {
        fmt.Println("Error:", err)
        os.Exit(-1)
    }
    fmt.Println(meta.Get(oss.HTTPHeaderOssSymlinkTarget))
}
```

For more information about obtaining symbolic links, see *GetSymlink*.

4.9 Object tagging

4.9.1 Configure object tagging

This topic describes how to configure object tagging.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Note:

- **You can add tags to an object when uploading it or add tags for an uploaded object. If add tags to an object that already has tags, the original tags are overwritten. For more information about adding tags, see** *#unique_87*.
- **To add tags to an object, you must have the permission to call PutObjectTagging.**
- **The Last-Modified value of an object is not updated when its tags are changed.**
- **A tag can contain letters, numbers, spaces, and the following symbols:** + - = . _ : /

Add tags to an object when uploading it

- **Add tags to an object when uploading it by calling PutObject.**

  Run the following code to add tags to an object when uploading it by calling PutObject:

```
package main

import (
  "fmt"
  "os"
  "strings"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
  // Configures the tags to be added to the object and the tagging
rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  // Adds tags to the object when uploading it.
  err = bucket.PutObject("<yourObjectKey>", strings.NewReader("<
yourObjectValue>"), oss.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

- **Add tags to an object when uploading it in the multipart upload method.**

  Run the following code to add tags to an object when uploading it in the multipart upload method:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // Configures the tags to be added to the object and the tagging
 rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
```

```
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  // Obtains the name and path of the local file to be uploaded.
  fileName := "<youFileNameAndPath>"

  // Divides the life into 3 parts and uploads the parts. The number
 of parts are calculated based on the file size.
  chunks, err := oss.SplitFileByPartNum(fileName, 3)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Opens the file.
  fd, err := os.Open(fileName)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  defer fd.Close()

  // Initializes the multipart upload task and sets the tags to be
added.
  imur, err := bucket.InitiateMultipartUpload("<yourObjectName>",
oss.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Uploads the object part by part.
  var parts []oss.UploadPart
  for _, chunk := range chunks {
    fd.Seek(chunk.Offset, os.SEEK_SET)
    part, err := bucket.UploadPart(imur, fd, chunk.Size, chunk.
Number)
    if err != nil {
      fmt.Println("Error:", err)
      os.Exit(-1)
    }
    parts = append(parts, part)
  }
  _, err = bucket.CompleteMultipartUpload(imur, parts)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

- **Add tags to an object when uploading it in the append upload method.**

  Run the following code to add tags to an object when uploading it in the append upload method:

```
package main

import (
  "fmt"
  "os"
```

```
  "strings"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // Configures the tags to be added to the object and the tagging
 rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  var nextPos int64
  // Uploads the file to the appendable object for the first time.
Only the tags set when AppendObject is called for the first time.
  nextPos, err = bucket.AppendObject("<yourObjectName>", strings.
NewReader("<yourObjectValue>"), nextPos, oss.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Uploads the file for the second time.
  nextPos, err = bucket.AppendObject("<yourObjectName>", strings.
NewReader("<yourObjectAppandValue>"), nextPos)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

- **Add tags to an object when uploading it in the resumable upload method.**

  Run the following code to add tags to an object when uploading it in the resumable upload method:

```
package main

import (
  "fmt"
  "os"
```

```
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // Configures the tags to be added to the object and the tagging
 rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  // Divides the object into multiple parts of 100 KB and uses three
 threads to upload the parts simultaneously. Tags are added to the
object when it is being uploaded.
  err = bucket.UploadFile("<yourObjectName>", "<yourNeedPuttingFile
>", 100*1024, oss.Routines(3), oss.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

Add tags to an uploaded object or modify the tags added to an object

Run the following code to add tags to an uploaded object or modify the tags added to an object:

```
package main

import (
  "fmt"
  "os"
  "strings"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
```

```
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Configures the tags to be added to the object and the tagging
rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }
  // Adds the tags to the object.
  err = bucket.PutObjectTagging("<yourObjectName>", tagging)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

Add tags to an object when copying it

You can specify one of the following tagging rules when copying an object:

- **Copy (default): Copy the tags of the source object to the destination object.**
- **Replace: Add the tags specified in the request to the destination object.**

Examples of adding tags to objects of different sizes when copying them are described as follows:

- **Run the following code to add tags to an object smaller than 1 GB when directly copying it:**

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
```

```
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // Configures the tags to be added to the object and the tagging
 rule.

  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  // If you only set the tagging parameter when copying the object,
the tags are not added.
  _, err = bucket.CopyObject("<yourSourceObjectName>", "<yourDirect
ObjectName>", oss.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // The tags are added only when the TaggingReplace and tagging
parameters are both set.
  _, err = bucket.CopyObject("<yourSourceObjectName>", "<yourDirect
ObjectName>", oss.SetTagging(tagging), oss.TaggingDirective(oss.
TaggingReplace))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

- **Run the following code to add tags to an object larger than 1 GB when copying it in the multipart copy method:**

```
package main

import (
  "fmt"
  "os"
  "strings"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
```

```
func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // Configures the tags to be added to the object and the tagging
 rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  // Uploads an object for copying.
  content := "this your object value"
  err = bucket.PutObject("<yourObjectName>", strings.NewReader(
content))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Initializes an object named <desObjectName> as the destination
object and sets the tags to be added to the object.
  imur, err := bucket.InitiateMultipartUpload("<desObjectName>", oss
.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // The yourSourObject indicates the source object. The source
object is uploaded as one part.
  part, err := bucket.UploadPartCopy(imur, "<yourBucketName>", "<
yourSourObject>", 0, int64(len(content)), 1)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  parts := []oss.UploadPart{part}
  _, err = bucket.CompleteMultipartUpload(imur, parts)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
}
```

Add tags to a symbolic link object

Run the following code to add tags to a symbolic link object:

```
package main

import (
  "fmt"
  "os"
  "strings"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

    // Configures the tags to be added to the object and the tagging
rule.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }
  tagging := oss.Tagging{
    Tags: []oss.Tag{tag1, tag2},
  }

  err = bucket.PutObject("<yourObjectName>", strings.NewReader("<
yourObjectValue>"))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  err = bucket.PutSymlink("<yourObjectName>", "<yourObjectSymLinkName
>", oss.SetTagging(tagging))
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
}
```

4.9.2 Obtain the tags added to an object

This topic describes how to obtain the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to obtain the tags added to an object:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the tags added to the object.
  taggingResult, err := bucket.GetObjectTagging("<yourObjectName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  fmt.Printf("Object Tagging: %v\n", taggingResult)
}
```

For more information about how to obtain the tags added to an object, see *#unique_89*.

4.9.3 Delete the tags added to an object

This topic describes how to delete the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to delete the tags added to an object:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the bucket that stores the object.
  bucket, err := client.Bucket("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Deletes the tags added to the object.
  err = bucket.DeleteObjectTagging("<yourObjectName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

For more information about how to delete the tags added to an object, see *#unique_91*.

4.9.4 Object tagging and lifecycle rules

Lifecycle rules can take effect on objects with specified prefixes or tags. You can also specify prefixes and tags as the condition of a lifecycle rule at the same time.

Note:

If you set a specified tag as the condition of a lifecycle rule, the rule applies to an object only when the key and value in the tag of the object both match the specified tag. If you specify a prefix and multiple tags as the condition of a lifecycle rule, the rule applies to an object only when the prefix and tags of the object match the prefix and all tags specified in the rule.

Set tags as the matching condition of a lifecycle rule

Run the following code to set tags as the matching condition of a lifecycle rule:

```
package main
```

```
import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Sets the lifecycle rule ID to "rule2" and applies it to objects
prefixed with "one".
  // The storage class of Objects prefixed with "one" is converted to
IA after 3 days and is converted to Archive after 30 days.
    Days: 3,
    StorageClass: oss.StorageIA,
  }
  transitionArch := oss.LifecycleTransition{
    Days: 30,
    StorageClass: oss.StorageArchive,
  }

  // Configures the tags used as the matching condition.
  tag1 := oss.Tag{
    Key:   "key1",
    Value: "value1",
  }
  tag2 := oss.Tag{
    Key:   "key2",
    Value: "value2",
  }

  // Sets the bucket lifecycle rule, which includes the tags set as
the matching condition.
  rule1 := oss.LifecycleRule{
    ID: "rule1",
    Prefix: "one",
    Status: "Enabled",
    Transitions: []oss.LifecycleTransition{transitionIA, transition
Arch},
    Tags:[]oss.Tag{tag1,tag2},
  }
  rules := []oss.LifecycleRule{rule}
  err = client.SetBucketLifecycle("<yourBucketName>", rules)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
}
```

View the tags set as the matching condition of a lifecycle rule

Run the following code to view the tags set as the matching condition of a lifecycle rule:

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains lifecycle rule for the bucket that stores the object.
  lc, err := client.GetBucketLifecycle("<yourBucketName>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Prints the lifecycle rule that includes the tags set as the
matching condition.
  for i, v := range lc.Rules {
    fmt.Printf("Bucket %d Lifecycle: %v", i, v)
    if  v.Expiration != nil {
      fmt.Printf(", Expiration:%v", *v.Expiration)
    }
    fmt.Printf("\n")
  }
}
```

4.10 Single-connection bandwidth throttling

This topic describes how to ensure other applications have sufficient bandwidth when you perform upload and download operations on OSS. You can include the x-oss-traffic-limit parameter in your requests and set bandwidth limits.

Note:

For more information about the scenarios and precautions of the single-connection bandwidth throttling feature, see *#unique_93* in the OSS Developer Guide.

Configure bandwidth throttling for common uploads and downloads

Use the following code to configure bandwidth throttling when you upload and download objects to and from OSS:

```
package main

import (
  "fmt"
  "os"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  //
  // You can use the single-connection bandwidth throttling feature
when you call the following operations:
  // PutObject/AppendObject/PostObject/CopyObject/UploadPart/
UploadPartCopy
  // The following code provides examples of how to configure
bandwidth throttling when you call these operations.
  //

  // Create an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtain the bucket.
  bucket, err := client.Bucket("<yourBucketName>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Specify the object to be uploaded.
  fd, err := os.Open("<yourLocalFile>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  defer fd.Close()

  // Set the upload bandwidth limit. The value of the parameter must
be a number. The value must be expressed in bit/s.
  // 41943040 indicates that the bandwidth limit for the request is 40
 Mbit/s, or 5 MB/s.
  var traffic int64 = 41943040
  // Example 1: Configure bandwidth throttling for the object to be
uploaded.
  err = bucket.PutObject("<youObjectName>", fd, oss.TrafficLimitHeader
(traffic))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Example 2: Configure bandwidth throttling for the object to be
downloaded. newFile indicates the name of the downloaded object.
```

```
  err = bucket.GetObjectToFile("<youObjectName>", "newFile", oss.
TrafficLimitHeader(traffic))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Example 3: Configure bandwidth throttling for append upload. In
this case, after localFileOne is uploaded, you can append localFileTwo
 .
  localFileOne := "<localFileOne>"
  localFileTwo := "<localFileTwo>"
  fd1, err := os.Open(localFileOne)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  defer fd.Close()
  fd2, err := os.Open(localFileTwo)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  defer fd.Close()

  var nextPos int64
  nextPos, err = bucket.AppendObject("<yourObjectName>", fd1, nextPos
, oss.TrafficLimitHeader(traffic))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  nextPos, err = bucket.AppendObject("<yourObjectName>", fd2, nextPos
, oss.TrafficLimitHeader(traffic))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Example 4: Configure bandwidth throttling for multipart upload of
 a large object.
  // You can split an object into multiple parts based on the object
size.
  chunks, err := oss.SplitFileByPartNum("localFile", 3)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Open the object.
  fd, err = os.Open("fileName")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  defer fd.Close()
  // Initialize the multipart upload and configure bandwidth
throttling.
  imur, err := bucket.InitiateMultipartUpload("<yourObjectName>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Upload parts and set the bandwidth limit.
  var parts []oss.UploadPart
```

```
  for _, chunk := range chunks {
    fd.Seek(chunk.Offset, os.SEEK_SET)
    part, err := bucket.UploadPart(imur, fd, chunk.Size, chunk.Number
, oss.TrafficLimitHeader(traffic))
  if err ! = nil {
  fmt.Println("Error:", err)
  os.Exit(-1)
  }
    parts = append(parts, part)
  }
  // Complete the multipart upload.
  _, err = bucket.CompleteMultipartUpload(imur, parts)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

Configure bandwidth throttling for uploads and downloads that use signed URLs

Use the following code to configure bandwidth throttling when you upload and download objects from and to OSS by using signed URLs.

```
package main

import (
  "fmt"
  "os"
  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Create an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtain the bucket.
  bucket, err := client.Bucket("<yourBucketName>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Example 1: Use the signed URL to upload an object.
  //
  // Open the object.
  fd, err := os.Open("<localfile>")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  defer fd.Close()

  // Set the upload bandwidth limit. The value of the parameter must
be a number. The value must be expressed in bit/s.
  // 41943040 indicates that the bandwidth limit for the request is 40
 Mbit/s, or 5 MB/s.
  var traffic int64 = 41943040
```

```
  // Obtain the signed URL for uploading an object.
  strURL, err := bucket.SignURL("<yourObjectName>", oss.HTTPPut, 60,
oss.TrafficLimitParam(traffic))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Upload the object.
  err = bucket.PutObjectWithURL(strURL, fd)
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Example 2: Use the signed URL to download an object.
  // Obtain the signed URL for downloading an object.
  strURL, err = bucket.SignURL("<yourObjectName>", oss.HTTPGet, 60,
oss.TrafficLimitParam(traffic))
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Download the object. newFile indicates the name of the downloaded
 object.
  err = bucket.GetObjectToFileWithURL(strURL, "newFile")
  if err ! = nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

4.11 Server-side encryption

OSS supports server-side encryption for uploaded data. This means that when user data is uploaded, OSS encrypts the data and permanently stores the data. Then, when the data is downloaded by a user, OSS automatically decrypts the data, returns the original data to the user, and declares in the header of the response that the data has been encrypted on the server.

The following server-side encryption methods are available for different application scenarios:

- **Server-side encryption that uses CMKs managed by KMS for encryption and decryption (SSE-KMS)**

  When uploading an object, you can use a specified CMK ID or the default CMK managed by KMS to encrypt and decrypt a large amount of data. This method

is cost-effective because you do not need to send user data to the KMS server through networks for encryption and decryption.

> **Notice:**
> API calling fees are incurred if you use CMKs to encrypt or decrypt data.

- Server-side encryption fully managed by OSS (SSE-OSS)

  When you upload an object, OSS encrypts the object on the server side by using AES256 fully managed by OSS. In this method, OSS uses AES256 to encrypt each object with an individual key. Furthermore, the individual keys are encrypted by a CMK that is updated periodically for higher security. This method applies to encrypt or decrypt bulk data.

> **Notice:**
> - Only one server-side encryption method can be used for an object at one time.
> - If you configure server-side encryption for a bucket, you can still configure the encryption method for a single object when uploading or copying it. In this case, the encryption method configured for the object takes precedence. For more information, see *PutObject*.
> - For more information about server-side encryption, see *Server-side encryption*.

Configure server-side encryption for a bucket

You can run the following code to configure the default encryption method for a bucket. After the method is configured, all objects that are uploaded to the bucket without their encryption methods configured are encrypted in the configured default encryption method.

```
package main

import (
  "fmt"
  "os"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
```

```
  // Initializes an encryption rule. In this example, the encryption
method is set to AES256.
  config := oss.ServerEncryptionRule{SSEDefault: oss.SSEDefaultRule{
SSEAlgorithm: "AES256"}}
  err = client.SetBucketEncryption("yourBucketName", config)
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

Obtain the server-side encryption settings of a bucket

You can run the following code to obtain the server-side encryption settings of a bucket:

```
package main

import (
  "fmt"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Obtains the server-side encryption rule for the bucket.
  ret, err := client.GetBucketEncryption("yourBucketName")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
  // Prints the obtained encryption rule.
  fmt.Println("Encrypt rule", ret.SSEDefault.SSEAlgorithm)
}
```

Delete the server-side encryption settings of a bucket

You can run the following code to delete the server-side encryption settings of a bucket:

```
package main

import (
  "fmt"

  "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func main() {
  // Creates an OSSClient instance.
```

```
  client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }

  // Deletes the encryption rule.
  err = client.DeleteBucketEncryption("yourBucketName")
  if err != nil {
    fmt.Println("Error:", err)
    os.Exit(-1)
  }
}
```

4.12 Authorized access

This topic describes how to authorize access to OSS.

Use STS for temporary access authorization

OSS supports Alibaba Cloud Security Token Service (STS) for temporary access authorization. STS is a web service that provides a temporary access token to a cloud computing user. Through the STS, you can assign a third-party application or a RAM user (you can manage the user ID) an access credential with a custom validity period and permissions. For more information about STS, see *STS introduction*.

STS advantages:

- **Your long-term key (AccessKey) is not exposed to a third-party application. You only need to generate an access token and send the access token to the third-party application. You can customize access permissions and the validity of this token.**
- **You do not need to keep track of permission revocation issues. The access token automatically becomes invalid when it expires.**

For more information about the process of access to OSS with STS, see *RAM and STS scenario practices* in OSS Developer Guide.

Run the following code to create a signature request with STS:

```
import "github.com/aliyun/aliyun-oss-go-sdk/oss"

// After a user obtains a temporary STS credential, the OSSClient is
generated with the security token and temporary access key (AccessKeyI
d and AccessKeySecre).
// Create an OSSClient instance.
client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>", oss.SecurityToken("<yourSecurityToken>"))
if err ! = nil {
    fmt.Println("Error:", err)
```

```
    os.Exit(-1)

// Perform operations on OSS.
}
```

Sign a URL to authorize temporary access

You can provide a signed URL to a visitor for temporary access. When you sign a URL, you can specify the expiration time for a URL to restrict the period of access from visitors.

For the complete code of signing a URL to authorize temporary access, see *GitHub*.

- **Use a signed URL to upload a file**

  Run the following code to upload a file with a signed URL:

```
package main

import (
    "fmt"
    "os"
    "strings"

    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        HandleError(err)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
    localFilename := "<yourLocalFilename>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        HandleError(err)
    }

    // Directly upload the file after the the URL is signed.
    signedURL, err := bucket.SignURL(objectName, oss.HTTPPut, 60)
    if err ! = nil {
        HandleError(err)
    }

    var val = "Choose Alibaba Cloud"
    err = bucket.PutObjectWithURL(signedURL, strings.NewReader(val))
    if err ! = nil {
        HandleError(err)
```

```
    }

    // Directly upload the file after the the URL is signed, in
which you can configure optional parameters.
    options := []oss.Option{
        oss.Meta("myprop", "mypropval"),
        oss.ContentType("image/tiff"),
    }

    signedURL, err = bucket.SignURL(objectName, oss.HTTPPut, 60,
options...)
    if err ! = nil {
        HandleError(err)
    }

    err = bucket.PutObjectFromFileWithURL(signedURL, localFilename,
options...)
    if err ! = nil {
        HandleError(err)
    }
}
```

Note:

For more information about the optional parameters, see *Manage Object Meta* **in** *Manage objects*.

- **Use a signed URL to download an object**

  Run the following code to use a signed URL to download an object:

```
package main

import (
    "fmt"
    "os"
    "io/ioutil"

    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        HandleError(err)
    }

    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
    localDownloadedFilename := "<yourDownloadedFilename>"

    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
```

```
        HandleError(err)
    }

    // Directly download the object to the stream after the URL is
signed.
    signedURL, err := bucket.SignURL(objectName, oss.HTTPGet, 60)
    if err ! = nil {
        HandleError(err)
    }

    body, err := bucket.GetObjectWithURL(signedURL)
    if err ! = nil {
        HandleError(err)
    }
    // Read the object content.
    data, err := ioutil.ReadAll(body)
    body.Close()
    data = data // use data

    // Directly download the object to a local file after the URL is
 signed.
    err = bucket.GetObjectToFileWithURL(signedURL, localDownl
oadedFilename)
    if err ! = nil {
        HandleError(err)
    }
}
```

4.13 Image processing

Image Processing (IMG) is a massive, secure, cost-effective and highly reliable image processing service. After source images are uploaded to OSS, you can process images on any Internet device at any anytime, from anywhere through simple RESTful APIs.

For more information about IMG, see *Image Processing*.

Basic features

OSS offers the following IMG features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**

- *Image processing*
- *Image processing access rules*: **calls multiple IMG functions.**

Usage

IMG uses standard HTTP GET. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is private read and write, only authorized users are allowed for access.

- **Anonymous access**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Specifies the name of a bucket.
endpoint	Specifies endpoint used to access a region.
object	Specifies the name of an image object.
image	Specifies the reserved identifier for IMG.
action	Specifies the operations on an image, such as scaling, cropping, and rotating .

Parameter	Description
param	Specifies the parameter that indicates the operation on an image.

- **Basic operations**

  For example, scale an image to a width of 100 px. Adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Customized image styles**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=style/<yourStyleName>
```

  - **style: specifies a reserved identifier of a customized image style.**
  - **yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.**

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=style/oss-pic-style-w-100
```

- **Cascade operations**

  Cascade operations allow you to perform multiple operations on an image sequentially.

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss
-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<
yourParamValue2>/...
```

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100/rotate,90
```

- **HTTPS access**

IMG supports access through HTTPS. Example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  Authorized access allows you to customize an image style, HTTPS access, and cascade operations.

  Use the following code to generate a signed URL for IMG:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
 HandleError(err)
    }

    // Obtain the bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
 HandleError(err)
    }

    ossImageName := "<yourImageName>"
    signedURL, err := bucket.SignURL(ossImageName, oss.HTTPGet, 600
, oss.Process("image/format,png"))
    if err ! = nil {
 HandleError(err)
    } else {
 fmt.Println(signedURL)
    }
```

```
}
```

- **Access with SDK**

  You can use SDK to access and process an image object.

  SDK allows you to customize image styles, HTTPS access, and cascade operations .

  - **Basic operations**

    Run the following code to perform basic operations on an image:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>",
 "<yourAccessKeySecret>")
    if err ! = nil {
 HandleError(err)
    }

    // Obtain the bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
 HandleError(err)
    }

    // Scale the image.
    sourceImageName := "example.png"
    targetImageName := "example-resize.png"
    style := "image/resize,m_fixed,w_100,h_100"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }

    // Crop the image.
    targetImageName = "example-crop.png"
    style = "image/crop,w_100,h_100,x_100,y_100,r_1"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }

    // Rotate the image.
```

```
    targetImageName = "example-rotate.png"
    style = "image/rotate,90"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }

    // Sharpen the image.
    targetImageName = "example-sharpen.png"
    style = "image/sharpen,100"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }

    // Add watermarks.
    targetImageName = "example-watermark.png"
    style = "image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }

    // Convert the image format.
    targetImageName = "example-formatconvert.jpg"
    style = "image/format,jpg"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }
}
```

- **Customize an image style**

  Run the following code to customize an image style:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>",
 "<yourAccessKeySecret>")
    if err ! = nil {
 HandleError(err)
    }

    // Obtain the bucket.
    bucketName := "<yourBucketName>"
```

```
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
  HandleError(err)
    }

    // Customize the image style.
    sourceImageName := "example.png"
    targetImageName := "example-custom.png"
    style := "style/<yourCustomStyleName>"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
 HandleError(err)
    }
}
```

- **Cascade operations**

  Run the following code to perform cascade operations on an image:

```
package main

import (
    "fmt"
    "os"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>",
 "<yourAccessKeySecret>")
    if err ! = nil {
 HandleError(err)
    }

    // Obtain the bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
 HandleError(err)
     }

    // Customize an image style.
    sourceImageName := "example.png"
    targetImageName := "example-resize.png"
    style := "image/resize,m_fixed,w_100,h_100/rotate,90"
    err = bucket.GetObjectToFile(sourceImageName, targetImageName
, oss.Process(style))
    if err ! = nil {
  HandleError(err)
    }
```

```
}
```

Image processing persistence

Run the following code for image processing persistence:

```
package main

import (
    "fmt"
    "os"
    "encoding/base64"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)

func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}

func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
 HandleError(err)
    }

    // Obtain the bucket.
    bucketName := "<yourBucketName>"
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
  HandleError(err)
    }


    // Image processing persistence: Scale the image. The target image
 is stored in the same bucket by default.
    sourceImageName := "example.png"
    targetImageName := "example-resize.png"
    style := "image/resize,m_fixed,w_100,h_100"
    process := fmt.Sprintf("%s|sys/saveas,o_%v", style, base64.
URLEncoding.EncodeToString([]byte(targetImageName)))
    result, err := bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
        fmt.Println(result)
    }

    // Image processing persistence: Scale the image. The target image
 is stored in the different bucket.
    targetBucketName := "<yourTargetBucketName>"
    targetImageName = "example-crop-different-bucket.png"
    style = "image/resize,m_fixed,w_100,h_100"
    process = fmt.Sprintf("%s|sys/saveas,o_%v,b_%v", style, base64
. URLEncoding.EncodeToString([]byte(targetImageName)), base64.
URLEncoding.EncodeToString([]byte(targetBucketName)))
    result, err = bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
```

```
        fmt.Println(result)
    }

    // Image processing persistence: Crop the image.
    targetImageName = "example-crop.png"
    style = "image/crop,w_100,h_100,x_100,y_100,r_1"
    process = fmt.Sprintf("%s|sys/saveas,o_%v", style, base64.
URLEncoding.EncodeToString([]byte(targetImageName)))
    result, err = bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
        fmt.Println(result)
    }

    // Image processing persistence: Rotate the image.
    targetImageName = "example-rotate.png"
    style = "image/rotate,90"
    process = fmt.Sprintf("%s|sys/saveas,o_%v", style, base64.
URLEncoding.EncodeToString([]byte(targetImageName)))
    result, err = bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
        fmt.Println(result)
    }

    // Image processing persistence: Sharpen the image.
    targetImageName = "example-sharpen.png"
    style = "image/sharpen,100"
    process = fmt.Sprintf("%s|sys/saveas,o_%v", style, base64.
URLEncoding.EncodeToString([]byte(targetImageName)))
    result, err = bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
       fmt.Println(result)
    }

    // Image processing persistence: Add watermarks.
    targetImageName = "example-watermark.png"
    style = "image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ"
    process = fmt.Sprintf("%s|sys/saveas,o_%v", style, base64.
URLEncoding.EncodeToString([]byte(targetImageName)))
    result, err = bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
        fmt.Println(result)
    }

    // Image processing persistence: Covert the image format.
    targetImageName = "example-formatconvert.jpg"
    style = "image/format,jpg"
    process = fmt.Sprintf("%s|sys/saveas,o_%v", style, base64.
URLEncoding.EncodeToString([]byte(targetImageName)))
    result, err = bucket.ProcessObject(sourceImageName, process)
    if err ! = nil {
 HandleError(err)
    } else {
        fmt.Println(result)
    }
```

```
}
```

4.14 Error handling

This topic lists the causes and solutions for some common errors you may experience when you use the OSS Go SDK.

Client errors

Call errors in Go SDK are returned through a unified error interface. The error interface is defined as follows:

```
type error interface {
    Error() string
}
```

Other errors are also returned through this error interface. For example, HTTP request errors are returned as follows:

```
//net.Error
type Error interface {
    error
    Timeout() bool   // Is the error a timeout
    Temporary() bool // Is the error temporary
}
```

Server errors

If the request errors occur when OSS Go SDK is used, corresponding errors are returned. In case of HTTP request errors and I/O errors, errors preset in Go are returned. In case of request processing errors of the OSS Server, the following error is returned through the error interface:

```
type ServiceError struct {
    Code       string    // Error code returned by OSS to a user
    Message    string    // Error details provided by OSS
    RequestId  string    // UUID that uniquely identifies the request
    HostId     string    // OSS cluster that identifies the access
    StatusCode int       // HTTP status code
}
```

If the HTTP status code returned by OSS is not as expected, the following error is returned, also through the error interface:

```
type UnexpectedStatusCodeError struct {
    allowed []int    // Expected HTTP status code returned by OSS
    got     int      // Actual HTTP status code returned by OSS
```

```
}
```

Example of error handling

The following code is used to demonstrate error handling:

```
package main
import (
    "fmt"
    "os"
    "strings"
    "github.com/aliyun/aliyun-oss-go-sdk/oss"
)
// Error handling function
func HandleError(err error) {
    fmt.Println("Error:", err)
    os.Exit(-1)
}
func main() {
    // Create an OSSClient instance.
    client, err := oss.New("<yourEndpoint>", "<yourAccessKeyId>", "<
yourAccessKeySecret>")
    if err ! = nil {
        HandleError(err)
    }
    bucketName := "<yourBucketName>"
    objectName := "<yourObjectName>"
    // Obtain the bucket.
    bucket, err := client.Bucket(bucketName)
    if err ! = nil {
        HandleError(err)
    }
    // Simple upload: string upload
    err = bucket.PutObject(objectName, strings.NewReader("yourObject
ValueContentString"))
    if err ! = nil {
        HandleError(err)
}
```

OSS error codes

The following table lists the OSS error codes.

Error code	Description	HTTP status code
AccessDenied	The access is denied.	403
BucketAlreadyExists	The bucket already exists.	409
BucketNotEmpty	The bucket is not empty.	409
EntityTooLarge	The entity is too large.	400
EntityTooSmall	The entity is too small.	400
FileGroupTooLarge	The object group is too large.	400

Error code	Description	HTTP status code
InvalidLinkName	The name of the object link is the same as the linked object.	400
LinkPartNotExist	The object to which the object link is linked does not exist.	400
ObjectLinkTooLarge	The object link has too many objects.	400
FieldItemTooLong	The table field in the post request is too large.	400
FilePartInterity	The object part has been changed.	400
FilePartNotExist	The object part does not exist.	400
FilePartStale	The object part has expired.	400
IncorrectNumberOfFilesInPOSTRequest	The number of objects in the post request is incorrect.	400
InvalidArgument	The parameter format is invalid.	400
InvalidAccessKeyId	The AccessKeyId does not exist.	403
InvalidBucketName	The bucket name is invalid.	400
InvalidDigest	The digest is invalid.	400
InvalidEncryptionAlgorithmError	The specified entropy encryption algorithm is invalid.	400
InvalidObjectName	The object name is invalid.	400
InvalidPart	The part is invalid.	400
InvalidPartOrder	The part order is invalid.	400
InvalidPolicyDocument	The policy document is invalid.	400

Error code	Description	HTTP status code
InvalidTargetBucketForLogging	A target bucket for logging is invalid.	400
InternalError	An error occurs in OSS.	500
MalformedXML	The XML format is invalid.	400
MalformedPOSTRequest	The format of the body in the post request is invalid.	400
MaxPOSTPreDataLengthExceededError	The body, except the object content, uploaded in the post request is too large.	400
MethodNotAllowed	The method is not supported.	405
MissingArgument	A parameter is missing.	411
MissingContentLength	The content length is missing.	411
NoSuchBucket	The bucket does not exist.	404
NoSuchKey	The object does not exist.	404
NoSuchUpload	The multipart upload ID does not exist.	404
NotImplemented	The method cannot be implemented.	501
PreconditionFailed	The preconditioning fails.	412
RequestTimeTooSkewed	Client local time and OSS server time differ by more than 15 minutes.	403
RequestTimeout	The request times out.	400
RequestIsNotMultiPartContent	The content type in the post request is invalid.	400
SignatureDoesNotMatch	The signature is incorrect.	403
TooManyBuckets	The number of the user's buckets exceeds the limit.	400
InvalidEncryptionAlgorithmError	The specified entropy encryption algorithm is invalid.	400

Note:

In the preceding table, the `Error code` is OssServiceError.Code, and the `HTTP status code` is OssServiceError.StatusCode.

5 C++

5.1 Overview

This document is written based on C++ SDK V1.2.0.

SDK source code

For the SDK source code, see *GitHub*.

Sample code

The OSS C++ SDK provides a variety of sample code for your reference or use. The following table describes the content in the sample code.

Sample file	Content
PutObjectFromFile	*Upload objects*
GetObjectToFile	*Download objects*
AppendObject	*Append upload*
MultipartUploadObject	*Multipart upload*
HeadObject	*Manage Object Meta*
ListObjects	*List objects*
CopyObject	*Copy an object*
DeleteObject	*Delete an object*
PutObjectProgress	*View upload progress*
DownloadObjectProcess	*View download progress*
RestoreArchiveObject	*Restore an archive object*
BucketSample.cc	**Set static website hosting, CORS, lifecycle, access logging, and authorized access for a bucket, as shown in *Static website hosting*, *CORS*, *Lifecycle*, *Access logging*, and *Authorized access*.**

5.2 Installation

This topic describes how to install the OSS C++ SDK.

Prerequisites

Install a compiler that supports C++ V11 and later.

- **Visual Studio V2013 and later**
- **GCC V4.8 and later**
- **Clang V3.3 and later**

Download the SDK

Download the SDK directly

Download from GitHub

Install the SDK

You can install the SDK in Linux, Windows, Android, and Mac.

- **Linux:**

  1. **Install CMake.**

     Install CMake V3.1 and later. Open the SDK installation package and use CMake to compile and generate the required files.

     The compilation commands are as follows:

```
cd <path/to/aliyun-oss-cpp-sdk>
mkdir build
cd build
```

```
cmake ..
```

2. **Install third-party libraries libcurl and openssl.**

   RedHat/Centos:

   ```
   yum -y install libcurl-devel openssl-devel
   ```

   Fedora:

   ```
   sudo dnf install libcurl-devel openssl-devel
   ```

3. **Install the SDK.**

   ```
   make && make install
   ```

Note:
RTTI is disabled in the C++ SDK by default. Therefore, add `-std=c++11`, `-fno-rtti`, and `-lalibabacloud-oss-cpp-sdk` when you use g++ to compile and run the SDK. Example:

```
g++ test.cpp -std=c++11 -fno-rtti -lalibabacloud-oss-cpp-sdk -o
test.bin
```

- **Windows:**

Note:
The *`alibabacloud-oss-cpp-sdk.sln`* project file is not included in the downloaded SDK package. You must run the `cmake` command to generate the required project file.

1. **Install CMake. Open `cmd` to go to the directory that stores the SDK files, create a folder named build, and run `cmake` to generate the required files, as shown in the following figure.**

```
D:\Item\C++\aliyun-oss-cpp-sdk\build>cmake ..
-- Selecting Windows SDK version 10.0.17134.0 to target Windows 6.1.7601.
-- Project version: 1.1.0
-- TARGET_OS: WINDOWS
-- Configuring done
-- Generating done
-- Build files have been written to: D:/Item/C++/aliyun-oss-cpp-sdk/build
```

Note:

To construct project files in the x64 architecture, run the `cmake -A x64 ..` command.

2. **Run `VS developer command prompt` as an administrator. Go to the build folder and run the following command to compile and install the SDK:**

```
msbuild ALL_BUILD.vcxproj
msbuild INSTALL.vcxproj
```

You can also use Visual Studio to open alibabacloud-oss-cpp-sdk.sln to generate a solution.

- **Android:**

Build a project based on the android-ndk-r16 tool chain in Linux. Install third-party libraries libcurl and libopenssl in *`$ANDROID_NDK/sysroot`* and run the following command to compile and install the SDK:

```
cmake -DCMAKE_TOOLCHAIN_FILE=$ANDROID_NDK/build/cmake/android.
toolchain.cmake  \
      -DANDROID_NDK=$ANDROID_NDK    \
      -DANDROID_ABI=armeabi-v7a     \
      -DANDROID_TOOLCHAIN=clang     \
      -DANDROID_PLATFORM=android-21 \
      -DANDROID_STL=c++_shared ..
make
```

- **Mac:**

In Mac, you must specify the path where OpenSSL is installed.

For example, if OpenSSL is installed in *`/usr/local/Cellar/openssl/1.0.2p`*, run the following command to compile and install the SDK:

```
cmake -DOPENSSL_ROOT_DIR=/usr/local/Cellar/openssl/1.0.2p  \
      -DOPENSSL_LIBRARIES=/usr/local/Cellar/openssl/1.0.2p/lib  \
      -DOPENSSL_INCLUDE_DIRS=/usr/local/Cellar/openssl/1.0.2p/
include/ ..
```

```
make
```

5.3 Quick start

This topic describes how to use the OSS C++ SDK to complete routine operations such as bucket creation, file uploads, and object downloads.

Create a bucket

A bucket is a global namespace in OSS. It is similar to a data container that stores files. Use the following code to create a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Specify the name, storage class, and ACL of the bucket. */
    CreateBucketRequest request(BucketName, StorageClass::IA,
CannedAccessControlList::PublicReadWrite);

    /* Create the bucket. */
    auto outcome = client.CreateBucket(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CreateBucket fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

For more information about the bucket naming rules, see the "Naming conventions" section in *Basic concepts*. For more information about how to create a bucket, see *Buckets*.

For more information about how to obtain endpoint information, see *Regions and endpoints*.

Upload a file

Use the following code to upload a file to OSS:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Upload the file. */
    auto outcome = client.PutObject(BucketName, ObjectName,"yourLocalF
ilename");

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "PutObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Download an object

Use the following code to download an object to a local file:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    /* Set the name of the object you want to download. */
```

```
    std::string FileNametoSave = "yourFileName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Download the object to the local file. */
    GetObjectRequest request(BucketName, ObjectName);
    request.setResponseStreamFactory([=]() {return std::make_shared<
std::fstream>(FileNametoSave, std::ios_base::out | std::ios_base::in
 | std::ios_base::trunc| std::ios_base::binary); });

    auto outcome = client.GetObject(request);

    if (outcome.isSuccess()) {
        std::cout << "GetObjectToFile success, ContentLength is" <<
outcome.result(). Metadata(). ContentLength() << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetObjectToFile fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

List objects

Use the following code to list objects in the specified bucket: You can list a maximum of 100 objects by default.

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List the objects in the specified bucket. */
    ListObjectsRequest request(BucketName);
    auto outcome = client.ListObjects(request);

    if (! outcome.isSuccess()) {
```

```
        /* Handle exceptions. */
        std::cout << "ListObjects fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        for (const auto& object : outcome.result(). ObjectSummarys())
 {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",last modified time:" << object.LastModified() << std::
endl;
        }
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Delete an object

Use the following code to delete the specified object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    DeleteObjectRequest request(BucketName, ObjectName);

    /* Delete the object. */
    auto outcome = client.DeleteObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
```

```
    ShutdownSdk();
    return 0;
}
```

For more information, see the "*Delete an object*" section in Manage files.

5.4 Initialization

OSSClient is used to manage OSS resources such as buckets and objects. To use the C++ SDK to initiate a request, you must initialize an OSSClient instance and modify the default configuration items of ClientConfiguration as needed.

Configure the OSSClient instance

ClientConfiguration is a configuration class of OSSClient. You can use ClientConfiguration to configure parameters such as user agents, host proxies, connection timeout, and the maximum number of connections. You can configure the following parameters.

Name	Description
userAgent	The user agent (the User-Agent field in the HTTP header). The default value is aliyun-sdk-cpp/1. X.X
maxConnections	The maximum number of connection pools. The default value is 16.
requestTimeoutMs	The request timeout period. The connection is closed if no data is received during the timeout period. The default value is 10,000 ms.
connectTimeoutMs	The timeout period to establish a connection. The default value is 5,000 ms.
retryStrategy	The maximum number of retry attempts in the case of a request error.
proxyScheme	The proxy protocol. The default value is HTTP.
proxyPort	The port for the proxy host.
proxyPassword	The password for proxy host verification .
proxyUserName	The username for proxy host verificati on.

Name	Description
verifySSL	Indicates whether to enable SSL verification. SSL verification is disabled by default.
enableCrc64	Indicates whether to enable the CRC64 check. The CRC64 check is enabled by default.
enableDateSkewAdjustment	Indicates whether to enable automatic correction of HTTP request time. This function is enabled by default.
sendRateLimiter	The limit to the upload speed, in Kbit/s.
recvRateLimiter	The limit to the download speed, in Kbit /s.

Configure a timeout period

Use the following code to configure a timeout period:

```
#include <alibabacloud/oss/OssClient.h>

using namespace AlibabaCloud::OSS;

int main(void)
{

    /* Initialize network resources. */
    InitializeSdk();
    ClientConfiguration conf;

    /* Set the maximum number of connection pools. The default value
is 16. */
    conf.maxConnections = 20;

    /* Configure the request timeout period. If no data is received
during the period, the connection is closed. The default value is 10,
000 ms. */
    conf.requestTimeoutMs = 8000;

    /* Configure the timeout period in milliseconds when connections
are established. The default value is 5,000 ms. */
    conf.connectTimeoutMs = 8000;

    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Configure speed limits

Use the following code to configure speed limits for uploads and downloads:

```
#include <alibabacloud/oss/OssClient.h>
#include <alibabacloud/oss/client/RateLimiter.h>

using namespace AlibabaCloud::OSS;

class  UserRateLimiter : public RateLimiter
{
public:
    DefaultRateLimiter() :rate_(0) {};
    ~DefaultRateLimiter() {};
    virtual void setRate(int rate) { rate_ = rate; };
    virtual int Rate() const { return rate_; };
private:
    int rate_;
};

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;

    auto sendrateLimiter = std::make_shared<UserRateLimiter>();
    auto recvrateLimiter = std::make_shared<UserRateLimiter>();
    conf.sendRateLimiter = sendrateLimiter;
    conf.recvRateLimiter = recvrateLimiter;

    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Set the speed limit for the download , in Kbit/s. */
    recvrateLimiter->setRate(256);

    /* Set the speed limit for the upload, in Kbit/s. */
    sendrateLimiter->setRate(256);

    /* Upload the file. */
    auto outcome = client.PutObject(BucketName, ObjectName,"yourLocalF
ilename");

    /* Update the speed limit during the upload, in Kbit/s. */
    sendrateLimiter->setRate(300);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Configure a retry policy

Use the following code to configure a retry policy:

```
#include <alibabacloud/oss/OssClient.h>
#include <alibabacloud/oss/client/RetryStrategy.h>

using namespace AlibabaCloud::OSS;

class UserRetryStrategy : public RetryStrategy
{
public:

    /* maxRetries indicates the maximum number of retry attempts.
scaleFactor indicates the factor that is used to calculate the waiting
 time of a retry attempt. */
    DefaultRetryStrategy(long maxRetries = 3, long scaleFactor = 300
) :
        m_scaleFactor(scaleFactor), m_maxRetries(maxRetries)
    {}

    /* You can define the shouldRetry function to determine whether to
 retry a request.*/
    bool shouldRetry(const Error & error, long attemptedRetries) const
;

    /* You can define the calcDelayTimeMs function to calculate the
waiting time (delay) of a retry attempt. */
    long calcDelayTimeMs(const Error & error, long attemptedRetries)
const;

private:
    long m_scaleFactor;
    long m_maxRetries;
};

bool DefaultRetryStrategy::shouldRetry(const Error & error, long
attemptedRetries) const
{
    if (attemptedRetries >= m_maxRetries)
        return false;

    long responseCode = error.Status();

    //http code
    if ((responseCode == 403 && error.Message().find("RequestTim
eTooSkewed")) ||
        (responseCode > 499 && responseCode < 599)) {
        return true;
    }
    else {
        switch (responseCode)
        {
        //curl error code
        case (ERROR_CURL_BASE + 7):  //CURLE_COULDNT_CONNECT
        case (ERROR_CURL_BASE + 18): //CURLE_PARTIAL_FILE
        case (ERROR_CURL_BASE + 23): //CURLE_WRITE_ERROR
        case (ERROR_CURL_BASE + 28): //CURLE_OPERATION_TIMEDOUT
        case (ERROR_CURL_BASE + 52): //CURLE_GOT_NOTHING
        case (ERROR_CURL_BASE + 55): //CURLE_SEND_ERROR
```

```
        case (ERROR_CURL_BASE + 56): //CURLE_RECV_ERROR
            return true;
        default:
            break;
        };
    }

    return false;
}

long DefaultRetryStrategy::calcDelayTimeMs(const Error & error, long
attemptedRetries) const
{
    UNUSED_PARAM(error);
    return (1 << attemptedRetries) * m_scaleFactor;
}

int main(void)
{
    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;

    /* The maximum number of retry attempts in the case of a request
error. The default value is 3. */
    auto defaultRetryStrategy = std::make_shared<UserRetryStrategy>(5
);
    conf.retryStrategy = defaultRetryStrategy;

    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5 Buckets

5.5.1 Buckets

A bucket serves as a container that stores objects. Objects belong to the bucket they are created in or uploaded to.

Create a bucket

Use the following code to create a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
```

```
    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Specify the name, storage class, and ACL of the bucket. */
    CreateBucketRequest request(BucketName, StorageClass::IA,
CannedAccessControlList::PublicReadWrite);

    /* Create the bucket. */
    auto outcome = client.CreateBucket(request);

    if (! outcome.isucces()) {
        /* Handle exceptions. */
        std::cout << "CreateBucket fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

For more information about the bucket naming rules, see the "Naming conventions" section in *Basic concepts*. You can specify the ACL and the storage class when you create a bucket, as shown in *Bucket ACLs* and *Storage classes*.

List buckets

Use the following code to list all buckets:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List the buckets. */
    ListBucketsRequest request;
    auto outcome = client.ListBuckets(request);

    if (outcome.isSuccess()) {
        /* Display the bucket information. */
        std::cout <<" success, and bucket count is" << outcome.result
(). Buckets().size() << std::endl;
```

```
        std::cout << "Bucket name is" << std::endl;
        for (auto result : outcome.result(). Buckets())
        {
            std::cout << result.Name() << std::endl;
        }
    }
    else {
        /* Handle exceptions. */
        std::cout << "ListBuckets fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Configure the ACL of a bucket

The following table describes the permissions included in the ACL of a bucket.

Permission	Description	Value
Private	**Only the owner or authorized users of a bucket have read and write permissions on objects in the bucket.**	**OSS_ACL_PRIVATE**
Public Read	**Only the owner or authorized users of a bucket have read and write permissions on objects in the bucket. Other users (including anonymous users) have only read permissions on objects in the bucket. Use caution when using this permission.**	**OSS_ACL_PUBLIC_READ**
Public	**All users of a bucket have read and write permissions on all objects in the bucket. Use caution when using this permission.**	**OSS_ACL_PUBLIC_READ_WRITE**

For more information about the ACL, see the "*Access control*" section in OSS Developer Guide.

Use the following code to configure the ACL of a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Configure the ACL of the bucket. */
    SetBucketAclRequest request(BucketName, CannedAccessControlList::
Private);
    auto outcome = client.SetBucketAcl(request);

    if (outcome.isSuccess()) {
        std::cout << " setBucketAcl to private success " << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "SetBucketAcl fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Obtain the ACL of a bucket

Use the following code to obtain the ACL of a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
```

```
    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain the ACL of the bucket. */
    GetBucketAclRequest request(BucketName);
    auto outcome = client.GetBucketAcl(request);

    if (outcome.isSuccess()) {
        std::cout << "getBucketAcl success, acl: " << outcome.result
(). Acl() << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "getBucketAcl fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Obtain the region of a bucket

Use the following code to obtain the region where a bucket resides:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain the region where the bucket resides. */
    GetBucketLocationRequest request(BucketName);
    auto outcome = client.GetBucketLocation(request);

    if (outcome.isSuccess()) {
        std::cout << "getBucketLocation success, location: " <<
outcome.result(). Location() << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "getBucketLocation fail" <<
        ",code:" << outcome.error(). Code() <<
```

```
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Determine whether a bucket exists

Use the following code to determine whether the specified bucket exists:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Determine whether the bucket exists. */
    auto outcome = client.DoesBucketExist(BucketName);

    if (outcome) {
        std::cout << " The Bucket exists" << std::endl;
    }
    else {
        std::cout << "The Bucket does not exist" << std::endl;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Delete a bucket

Before deleting a bucket, you must delete all objects in the bucket and fragments generated by multipart upload.

Note:

To delete the fragments that are generated from multipart upload, use *ListMultipartUploads* to list all fragments, and then use *AbortMultipartUpload* to delete the fragments.

Use the following code to delete a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Specify the name of the bucket you want to delete. */
    DeleteBucketRequest request(BucketName);

    /* Delete the bucket. */
    auto outcome = client.DeleteBucket(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteBucket fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.2 Bucket tagging

By using the bucket tagging function, you can classify your buckets to manage them. For example, you can specify that only buckets with specified tags are returned in ListBucket operations.

Note:

- **Only the bucket owner and authorized RAM users can add tags to a bucket. Otherwise, the 403 Forbidden error is returned with the error code: AccessDenied.**
- **You can add a maximum of 20 tags (key-value pairs) to a bucket.**
- **The maximum size of a key is 64 bytes. The key of a tag cannot be null and cannot be prefixed with `http://`, `https://`, or `Aliyun`.**

- **The maximum size of a tag value is 128 bytes. The value of a tag can be null.**
- **The key and value of a tag must be UTF-8 encoded.**
- **If you use PutBucketTagging to add tags to a bucket, the original tags added to the bucket are completely overwritten.**

Configure bucket tagging

You can run the following code to configure tags to a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Configure bucket tagging. */
    SetBucketTaggingRequest request(BucketName);
    Tag tag1("yourTagkey1","yourTagValue1");
    Tag tag2("yourTagkey2", "yourTagValue2");
    TagSet tagset;
    tagset.push_back(tag1);
    tagset.push_back(tag2);
    Tagging taging;
    taging.setTags(tagset);
    request.setTagging(taging);
    auto outcome = client.SetBucketTagging(request);

    if (outcome.isSuccess()) {
            std::cout << " SetBucketTagging success " << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "SetBucketTagging fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Obtain bucket tags

You can run the following code to obtain the tags to a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain bucket tags. */
    GetBucketTaggingRequest request(BucketName);
    auto outcome = client.GetBucketTagging(request);

    if (outcome.isSuccess()) {
        std::cout << " GetBucketTagging success " << std::endl;
        for (const auto& tag : outcome.result().Tagging().Tags()) {
            std::cout << "tag key:" << tag.Key() <<
            "tag value:" << tag.Value() << std::endl;
        }
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketTagging fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Lists buckets that have specific tags

You can run the following code to list buckets with a specified tag:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
```

```
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List buckets that have specific tags. */
    ListBucketsRequest request(BucketName);
    Tag tag1("yourTagkey1","yourTagValue1");

    request.setTag(tag1);
    auto outcome = client.ListBuckets(request);

    if (outcome.isSuccess()) {
        std::cout << "ListBuckets success" << std::endl;
        for (const auto& bucket : outcome.result().Buckets()) {
            std::cout << "bucket name:" << bucket.Name() << std::endl;
        }
    }
    else {
        /* Handle exceptions. */
        std::cout << "ListBuckets fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Delete bucket tagging configurations

You can run the following code to delete the tags added to a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Delete bucket tagging configurations. */
    DeleteBucketTaggingRequest request(BucketName);
    auto outcome = client.DeleteBucketTagging(request);
```

```
    if (outcome.isSuccess()) {
            std::cout << " DeleteBucketTagging success " << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "DeleteBucketTagging fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.3 Manage lifecycle rules

This topic describes how to manage lifecycle rules.

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**
- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**
- **Status: This lifecyle rule takes effect or not.**

Lifecycle rules apply to resources uploaded by using the oss_upload_file method . The last modification time of an object will be changed to the time that the multipart upload event is initiated.

For more information about lifecycle management, see *Manage object lifecycles*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    SetBucketLifecycleRequest request(BucketName);
    std::string date("2022-10-12T00:00:00.000Z");

    /* Specify the period for which to retain objects after their most
 recent modification time. When the period expires, the objects are
deleted. */
    auto rule1 = LifecycleRule();
    rule1.setID("rule1");
    rule1.setPrefix("test1/");
    rule1.setStatus(RuleStatus::Enabled);
    rule1.setExpiration(3);

    /* Set the period for which to convert the storage class. When the
 period expires, the storage class is converted. */
    auto transition = LifeCycleTransition();
    transition.Expiration().setDays(180);
    transition.setStorageClass(StorageClass::IA);
    rule1.addTransition(transition);

    transition.Expiration().setDays(365);
    transition.setStorageClass(StorageClass::Archive);
    rule1.addTransition(transition);

    /* Configure lifecycle rules. */
    LifecycleRuleList list{rule1};
    request.setLifecycleRules(list);
    auto outcome = client.SetBucketLifecycle(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketLifecycle fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

View lifecycle rules

Run the following code to view lifecycle rules:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, c onf);

    /* View lifecycle rules. */
    auto outcome = client.GetBucketLifecycle(BucketName);

    if (outcome.isSuccess()) {
        std::cout << "GetBucketLifecycle success," << std::endl;
        for (auto const rule : outcome.result().LifecycleRules()) {
            std::cout << "rule:" << rule.ID() << "," << rule.Prefix()
 << "," << rule.Status() << ","
            "hasExpiration:" << rule.hasExpiration() << "," <<
            "hasTransitionList:" << rule.hasTransitionList() << ","
 << std::endl;
        }
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketLifecycle fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
```

```
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Clear lifecycle rules. */
    DeleteBucketLifecycleRequest request(BucketName);
    auto outcome = client.DeleteBucketLifecycle(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteBucketLifecycle fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.4 Pay-by-requester mode

After pay-by-requester mode is enabled for a bucket in Alibaba Cloud OSS, the requester instead of the bucket owner pays the cost of requests and traffic. The bucket owner always pays the cost of storing data. You can enable this feature to share data without having to pay for all requests and traffic by yourself.

Request methods

- **Access from anonymous users is not allowed**

  If you enable pay-by-requester mode for a bucket, anonymous users are not allowed to access the bucket. Requesters must provide authentication informatio n so that OSS can identify and charge them, but not the bucket owner, for requests and traffic.

  If requesters assume the RAM role of an Alibaba Cloud account to request data, OSS charges this Alibaba Cloud account for such requests and traffic.

- **Requesters must include the x-oss-request-payer request header**

  If you enable pay-by-requester mode for a bucket, requesters must include the x-oss-request-payer:requester request header in a PUT, POST, GET, or HEAD

request. This field indicates that requesters understand that their requests and data downloads incur fees. Otherwise, requests cannot pass authentication.

Bucket owners do not need to include the x-oss-request-payer request header when accessing their own buckets. In this case, bucket owners are requesters who pay for such requests and traffic.

For more information about pay-by-requester mode, see *#unique_48* in the OSS Developer Guide.

Enable pay-by-requester mode

The following code provides an example of how to enable pay-by-requester mode:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Enable pay-by-requester mode. */
    SetBucketRequestPaymentRequest request(BucketName);
    request.setRequestPayer(RequestPayer::Requester);

    auto outcome = client.SetBucketRequestPayment(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketRequestPayment fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Obtain pay-by-requester configurations

The following code provides an example of how to obtain pay-by-requester configurations:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain pay-by-requester configurations. */
    GetBucketRequestPaymentRequest request(BucketName)
    auto outcome = client.GetBucketRequestPayment(request);

    if (outcome.isSuccess())
    {
        std::cout << "GetBucketRequestPayment success Payer:" <<
outcome.result().Payer() << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketPayment fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Specify a third party to pay for access to objects

The following code provides an example of how to specify a third party to pay for access to objects when you use PutObject, GetObject, and DeleteObject:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
```

```
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string PayerAccessKeyId = "yourPayerAccessKeyId";
    std::string PayerAccessKeySecret = "yourPayerAccessKeySecret";
    std::string PayerEndpoint = "yourPayerEndpoint";
    std::string BucketName = "yourBucketName";
    std::string PayerUID = "PayerUID";
    std::string ObjectName = "yourObjectName";


    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    OssClient payerClient(PayerEndpoint, PayerAccessKeyId, PayerAcces
sKeySecret, conf);

    /* Create the bucket. */
    CreateBucketOutcome outCome = client.CreateBucket(CreateBuck
etRequest(BucketName));

    /* Enable pay-by-requester mode. */
    SetBucketRequestPaymentRequest request(BucketName);
    request.setRequestPayer(RequestPayer::Requester);
    auto payoutcome = client.SetBucketRequestPayment(request);

    /* Enable pay-by-requester mode when you upload the object. */
    *content << "test cpp sdk";
    PutObjectRequest putrequest(BucketName, ObjectName, content);
    putrequest.setRequestPayer(RequestPayer::Requester);
    auto putoutcome = payerClient.PutObject(putrequest);

    /* Enable pay-by-requester mode when you download the object to
the local memory. */
    GetObjectRequest getrequest(BucketName, ObjectName);
    getrequest.setRequestPayer(RequestPayer::Requester);
    auto getoutcome = payerClient.GetObject(getrequest);

    /* Enable pay-by-requester mode when you delete the object. */
    DeleteObjectRequest delrequest(BucketName, ObjectName);
    delrequest.setRequestPayer(RequestPayer::Requester);
    auto deloutcome = payerClient.DeleteObject(delrequest);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.5 Hotlink protection

This topic describes how to use hotlink protection.

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**

- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see the hotlink protection section in the OSS Developer Guide.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf) ;

    /* Configure the referer whitelist. */
    SetBucketRefererRequest request(BucketName);
    request.addReferer("http://www.referersample.com");
    request.addReferer("https://www.referersample.com");
    request.addReferer("https://www.?.referersample.com");
    request.addReferer("https://www. *.cn");
    request.setAllowEmptyReferer(false);

    auto outcome = client.SetBucketReferer(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketReferer fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Obtain a referer whitelist

Run the following code to obtain a referer whitelist:

```
#include <alibabacloud/oss/OssClient.h>
```

```
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);


    /* Obtain the referer list for a bucket. */
    GetBucketRefererRequest request(BucketName);
    auto outcome = client.GetBucketReferer(request);

    if (outcome.isSuccess()) {
        std::cout << " GetBucketReferer success, AllowEmptyReferer: "
 << outcome.result(). AllowEmptyReferer() <<
        " ,Referer size: " << outcome.result(). RefererList().size()
 << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketReferer fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf);
```

```
    /* You cannot clear a referer whitelist directly. To clear a
referer whitelist, you must create a rule that allows an empty referer
 field and replace the original rule with the new rule. */
    SetBucketRefererRequest request(BucketName);
    request.setAllowEmptyReferer(true);

    auto outcome = client.SetBucketReferer(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CleanBucketReferer fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.6 Access logging

You can enable access logging to record access to a bucket in a log. The log is stored in a specified bucket.

The log format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

For more information about access logging, see the "*Configure access logging*" section in OSS Developer Guide.

Enable access logging

Use the following code to enable access logging for a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
      /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string TargetBucketName = "yourTargetBucketName";
    std::string TargetPrefix  ="yourTargetPrefix";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret ,  conf);
```

```
    /* Enable access logging. */
    SetBucketLoggingRequest request(BucketName, TargetBucketName,
TargetPrefix);
    auto outcome = client.SetBucketLogging(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketLogging fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

View access logging configurations

Use the following code to view the access logging configurations of a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
      /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* View the access logging configurations. */
    GetBucketLoggingRequest request(BucketName);
    auto outcome = client.GetBucketLogging(request);

    if (outcome.isSuccess()) {
        std::cout <<" GetBucketLogging success, TargetBucket: " <<
outcome.result(). TargetBucket() <<
        ",TargetPrefix: " << outcome.result(). TargetPrefix() << std::
endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketLogging fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
```

```
    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Disable access logging

Use the following code to disable access logging for a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Disable access logging. */
    DeleteBucketLoggingRequest request(BucketName);
    auto outcome = client.DeleteBucketLogging(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteBucketLogging fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.7 Static website hosting

You can set the static website hosting mode for your bucket. After the configuration takes effect, you can use the bucket domain to access this static website and be redirected to a specified index page or error page.

For more information about static website hosting, see the "*Configure static website hosting*" section in OSS Developer Guide.

Configure static website hosting

Use the following code to configure static website hosting:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
      /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf);

    /* Configure static website hosting. */
    SetBucketWebsiteRequest request(BucketName);
    request.setIndexDocument("index.html");
    request.setErrorDocument("error.html");

    auto outcome = client.SetBucketWebsite(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketWebsite fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

View static website hosting configurations

Use the following code to view static website hosting configurations:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
      /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();
```

```
    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* View static website hosting configurations. */
    GetBucketWebsiteRequest request(BucketName);
    auto outcome = client.GetBucketWebsite(request);

    if (outcome.isSuccess()) {
        std::cout << "GetBucketWebsite success,IndexDocument: " <<
outcome.result(). IndexDocument() <<
        " ,ErrorDocument: " << outcome.result(). ErrorDocument() <<
std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketWebsite fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Delete static website hosting configurations

Use the following code to delete static website hosting configurations:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf);


    /* Delete static website hosting configurations. */
    DeleteBucketWebsiteRequest request(BucketName);

    auto outcome = client.DeleteBucketWebsite(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteBucketWebsite fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
```

```
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.5.8 CORS

This topic describes how to perform cross-origin resource sharing (CORS).

CORS allows Web applications to access resources that belong to another region. OSS provides CORS APIs to facilitate cross-origin access control.

For more information, see the "*CORS*" and "*PutBucketcors*" sections in OSS Developer Guide.

Configure CORS rules

Use the following code to configure CORS rules for a specified bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
      /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    SetBucketCorsRequest request(BucketName);

    /* Configure CORS rules. */
    auto rule1 = CORSRule();
    /* Specify the cross-origin source to share resources.*/
    rule1.addAllowedOrigin("http://www.a.com");
    /* Specify the allowed CORS request method. (You can select GET,
PUT, POST, DELETE or HEAD.)*/
    rule1.addAllowedMethod("POST");
    /* Determine whether to allow the header specified in Access-
Control-Request-Headers of the pre-flight (OPTIONS) request. */
    rule1.addAllowedHeader("*");
    /* Specify the response header that allows user access from
applications. */
    rule1.addExposeHeader("x-oss-test");
    /* You can set a maximum of 10 rules. */
    request.addCORSRule(rule1);

    auto rule2 = CORSRule();
    rule2.addAllowedOrigin("http://www.b.com");
```

```
    rule2.addAllowedMethod("GET");
    rule2.addExposeHeader("x-oss-test2");
    rule2.setMaxAgeSeconds(100);
    request.addCORSRule(rule2);

    auto outcome = client.SetBucketCors(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketCors fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

View CORS rules

Use the following code to view CORS rules set for a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, c onf);


    /* View CORS rules. */
    auto outcome = client.GetBucketCors(BucketName);

    if (outcome.isSuccess()) {
        std::cout << "GetBucketCors success" << std::endl;
        for (auto const rule : outcome.result(). CORSRules()) {
            std::cout << "Get Bucket Cors List:" << std::endl;
            for (auto const origin : rule.AllowedOrigins()) {
                std::cout << "Allowed origin:" << origin << std::endl;
            }
        }
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetBucketCors fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
```

```
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Delete CORS rules

Use the following code to delete the CORS rules set for a specified bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf);


    /* Delete the CORS rules. */
    auto outcome = client.DeleteBucketCors(BucketName);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteBucketCors fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.6 Upload objects

5.6.1 Overview

Objects are the basic unit for data operations in OSS. The OSS C++ SDK provides the following object upload methods:

- *Simple upload*: supports the upload of a file up to 5 GB in size. It includes upload from memory and local file upload.
- *Append upload*: supports the upload of a file up to 5 GB in size.
- *Resumable upload*: supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files. It supports concurrent upload and you can define the size of each part.
- *Multipart upload*: supports the upload of files up to 48.8 TB in size. It applies to the upload of large files.

Note:
For more information about the scenarios of each upload method, see the "Upload files" section in OSS Developer Guide.

During the file upload, you can set Object Meta and view upload progress, as shown in *Configure Object Meta* and *Use progress bars*. After you upload the file, you can perform upload callback, as shown in *Upload callback*.

5.6.2 Simple upload

This topic describes how to upload a file from memory or a local disk to OSS.

Upload from memory

Use the following code to upload a file from memory to OSS:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "Thank you for using Aliyun Object Storage Service!" ;
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Upload the file. */
    auto outcome = client.PutObject(request);
```

```
    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "PutObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Local file upload

Use the following code to upload a local file to OSS:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Upload the file. */
    auto outcome = client.PutObject(BucketName, ObjectName,"yourLocalF
ilename");

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "PutObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

5.6.3 Append upload

This topic describes how to append data from a local file.

Use the following code to append data from a local file:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    auto meta = ObjectMetaData();
    meta.setContentType("text/plain");

    // If the object is appended for the first time, the append
position is 0, and the returned value is the position for the next
append. The position for the next append is the length of the current
object. */
    std::shared_ptr<std::iostream> content1 = std::make_shared<std::
stringstream>();
    *content1 <<"Thank you for using Aliyun Object Storage Service!" ;
    AppendObjectRequest request(BucketName, ObjectName, content1, meta
);
    request.setPosition(0L);

    /* Start the first append. */
    auto result = client.AppendObject(request);

    if (! result.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "AppendObject fail" <<
        ",code:" << result.error(). Code() <<
        ",message:" << result.error(). Message() <<
        ",requestId:" << result.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    std::shared_ptr<std::iostream> content2 = std::make_shared<std::
stringstream>();
    *content2 <<"Thank you for using Aliyun Object Storage Service!" ;
    auto position = result.result(). Length();
    AppendObjectRequest appendObjectRequest(BucketName, ObjectName,
content2);
    appendObjectRequest.setPosition(position);

    /* Start the second append. */
```

```
    auto outcome = client.AppendObject(appendObjectRequest);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "AppendObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

If the object already exists, one of the following situations occurs:

- **If the object is not appendable, ObjectNotAppendable is thrown.**
- **If the object is appendable, PutObject is used to upload the object, the object overwrites the existing object, and the object type becomes Normal Object. After the HeadObject operation is performed, OSS returns x-oss-object-type.**

5.6.4 Resumable upload

You can use resumable upload to split a file into several parts and upload them separately. After you have uploaded all parts, you can combine these parts into a complete object. In this way, you can complete uploading the entire object.

For more information about resumable upload, see the "*Resumable upload*" section in OSS Developer Guide. For the complete code, see *GitHub*.

You can use OSSClient. ResumableUploadObject to implement resumable upload. The following table describes the parameters contained in UploadObjectRequest.

Name	Description	Required	Default value	Configuration method
bucket	The name of the bucket.	Yes	None	Constructor
key	The name of the object.	Yes	None	Constructor

Name	Description	Required	Default value	Configuration method
filePath	The path to the local file. The file located in this path is uploaded to OSS.	No	The path to the local file	Constructor
partSize	The size of each part. Valid values: [100 KB, 5 GB].	No	8 MB	setPartSize
threadNum	The number of parts you need to upload simultaneously .	No	3	Constructor or setThreadNum

Name	Description	Required	Default value	Configuration method
checkpointDir	The directory that records the upload results of each local part. This parameter must be configured to enable resumable upload. This file stores information about upload progress. If you fail to upload a part, the part upload continues based on the recorded progress. After the entire local file is uploaded, this file is deleted.	No	The same directory as that of DownloadFile	Constructor or setCheckpointDir

Use the following code for resumable upload:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    std::string UploadFilePath = "yourUploadfilePath";
    std::string CheckpointFilePath = "yourCheckpointFilepath"

    /* Initialize network resources. */
    InitializeSdk();
```

```
    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Start resumable upload. */
    UploadObjectRequest request(BucketName, ObjectName, UploadFilePath
, CheckpointFilePath);
    auto outcome = client.ResumableUploadObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "ResumableUploadObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.6.5 Multipart upload

This topic describes how to use multipart upload.

To perform multipart upload, follow these steps:

1. **Initiate a multipart upload task.**

   Use the InitiateMultipartUpload method to obtain an upload ID that is globally unique in OSS.

2. **Start multipart upload.**

   Use the UploadPart method to upload the parts.

Note:

- **For parts with a same uploadId, parts are sequenced by their part numbers. If you have uploaded a part and use the same part number to upload another part, the later part will replace the former part.**
- **OSS places the MD5 value of part data in ETag and returns the MD5 value to the user.**
- **SDK automatically configures Content-MD5. OSS calculates the MD5 value of uploaded data and compares it with the MD5 value calculated by SDK. If the two values vary, the error code of InvalidDigest is returned.**

3. **Complete multipart upload.**

   After all of the parts have been uploaded, use the CompleteMultipartUpload method to combine these parts into a complete object.

The following code provides an example on how to use multipart upload:

```
#include <alibabacloud/oss/OssClient.h>

int64_t getFileSize(const std::string& file)
{
    std::fstream f(file, std::ios::in | std::ios::binary);
    f.seekg(0, f.end);
    int64_t size = f.tellg();
    f.close();
    return size;
}

using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    InitiateMultipartUploadRequest initUploadRequest(BucketName,
ObjectName);
    /* Optional. Specify the storage class. */
    //initUploadRequest.MetaData().addHeader("x-oss-storage-class", "
Standard");

    /* Initiate the multipart upload task. */
    auto uploadIdResult = client.InitiateMultipartUpload(initUpload
Request);
    auto uploadId = uploadIdResult.result().UploadId();
    std::string fileToUpload = "yourLocalFilename";
    int64_t partSize = 100 * 1024;
    PartList partETagList;
    auto fileSize = getFileSize(fileToUpload);
    int partCount = static_cast<int>(fileSize / partSize);
    /* Calculate the number of upload parts. */
    if (fileSize % partSize ! = 0) {
        partCount++;
    }

    /* Upload each part sequentially. */
    for (int i = 1; i <= partCount; i++) {
        auto skipBytes = partSize * (i - 1);
        auto size = (partSize < fileSize - skipBytes) ? partSize : (
fileSize - skipBytes);
        std::shared_ptr<std::iostream> content = std::make_shared<std
::fstream>(fileToUpload, std::ios::in|std::ios::binary);
```

```
        content->seekg(skipBytes, std::ios::beg);

        UploadPartRequest uploadPartRequest(BucketName, ObjectName,
content);
        uploadPartRequest.setContentLength(size);
        uploadPartRequest.setUploadId(uploadId);
        uploadPartRequest.setPartNumber(i);
        auto uploadPartOutcome = client.UploadPart(uploadPartRequest);
        if (uploadPartOutcome.isSuccess()) {
            Part part(i, uploadPartOutcome.result().ETag());
            partETagList.push_back(part);
        }
        else {
            std::cout << "uploadPart fail" <<
            ",code:" << uploadPartOutcome.error().Code() <<
            ",message:" << uploadPartOutcome.error().Message() <<
            ",requestId:" << uploadPartOutcome.error().RequestId() <<
std::endl;
        }

    }

    /* Complete multipart upload. */
    CompleteMultipartUploadRequest request(BucketName, ObjectName);
    request.setUploadId(uploadId);
    request.setPartList(partETagList);
    /* Optional. Set the ACL. */
    //request.setAcl(CannedAccessControlList::Private);

    auto outcome = client.CompleteMultipartUpload(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CompleteMultipartUpload fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Note:

Obtain the uploaded parts. You must provide the ETag value of each part to call `CompleteMultipartUpload`**.**

You can use either of the following methods to obtain the ETag value:

- **Method 1: When a part is uploaded, save and use the ETag value of the part that is included in the response.**
- **Method 2: Call the ListParts operation to obtain the ETag values of the uploaded parts. In this example, the first method is used.**

List uploaded parts

You can use the ListParts method to list all parts that are uploaded with the specified upload ID.

- **List all uploaded parts**

  Note:
  By default, the ListParts method can list up to 1,000 parts at a time. To list more than 1,000 parts, the results must be listed by page.

  The following code provides an example on how to list all uploaded parts:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List the uploaded parts. The default number of parts returned
 is 1,000. */
    ListPartsRequest listuploadrequest(BucketName, ObjectName);
    listuploadrequest.setUploadId(uploadId);
    do {
        auto listuploadresult = client.ListParts(listuploadrequest);
        if (! listUploadResult.isSuccess()) {
            /* Handle exceptions. */
            std::cout << "ListParts fail" <<
            ",code:" << listuploadresult.error().Code() <<
            ",message:" << listuploadresult.error().Message() <<
            ",requestId:" << listuploadresult.error().RequestId()
 << std::endl;
            break;
        }
        else {
            for (const auto& part : listuploadresult.result().
PartList()) {
                std::cout << "part"<<
                ",name:" << part.PartNumber() <<
                ",size:" << part.Size() <<
                ",etag:" << part.ETag() <<
                ",lastmodify time:" << part.LastModified() << std::
endl;
            }
        }
```

```
        listuploadrequest.setPartNumberMarker(listuploadresult.
result().NextPartNumberMarker());
    } while (listuploadresult.result().IsTruncated());

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **List all uploaded parts by page**

  The following code provides an example on how to specify the maximum number of parts to list on each page and list all parts by page:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List all uploaded parts by page. */
    /* Set the maximum number of parts to list on each page. */
    ListPartsRequest listuploadrequest(BucketName, ObjectName);
    listuploadrequest.setMaxParts(50);
    listuploadrequest.setUploadId(uploadId);
    do {
        listuploadresult = client.ListParts(listuploadrequest);
        if (! listUploadResult.isSuccess()) {
            /* Handle exceptions. */
            std::cout << "ListParts fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << listuploadresult.error().Message() <<
            ",requestId:" << listuploadresult.error().RequestId()
 << std::endl;
            break;
        }
        else {
            for (const auto& part : listuploadresult.result().
PartList()) {
                std::cout << "part"<<
                ",name:" << part.PartNumber() <<
                ",size:" << part.Size() <<
                ",etag:" << part.ETag() <<
                ",lastmodify time:" << part.LastModified() << std::
endl;
            }
        }
        listuploadrequest.setPartNumberMarker(listuploadresult.
result().NextPartNumberMarker());
    } while (listuploadresult.result().IsTruncated());
```

```
    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

List multipart upload tasks

You can use the ListMultipartUploads method to list all ongoing multipart upload tasks (tasks that have been initiated but not completed or tasks that have been canceled).

- **List all multipart upload tasks**

  Note:
  By default, the ListMultipartUploads method can list up to 1,000 multipart upload tasks at a time. To list more than 1,000 multipart upload tasks, the results must be listed by page.

  The following code provides an example on how to list all multipart upload tasks :

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List the multipart upload tasks. The default number of parts
returned is 1,000. */
    ListMultipartUploadsRequest listmultiuploadrequest(BucketName);
    do {
        auto listresult = client.ListMultipartUploads(listmultiu
ploadrequest);
        if (! listresult.isSuccess()) {
            /* Handle exceptions. */
            std::cout << "ListMultipartUploads fail" <<
            ",code:" << listresult.error().Code() <<
            ",message:" << listresult.error().Message() <<
            ",requestId:" << listresult.error().RequestId() << std::
endl;
            break;
        }
        else {
```

```
            for (const auto& part : listresult.result().MultipartU
ploadList()) {
                std::cout << "part"<<
                ",name:" << part.Key <<
                ",uploadid:" << part.UploadId <<
                ",initiated time:" << part.Initiated << std::endl;
            }
        }
        listmultiuploadrequest.setKeyMarker(listresult.result().
NextKeyMarker());
        listmultiuploadrequest.setUploadIdMarker(listresult.result
().NextUploadIdMarker());
    } while (listresult.result().IsTruncated());

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **List all multipart upload tasks by page**

  The following code provides an example on how to list all multipart upload tasks by page:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* List all multipart upload tasks by page. */
    /* Set the maximum number of multipart upload tasks to list on
each page. */
    ListMultipartUploadsRequest  listmultiuploadrequest(BucketName);
    listmultiuploadrequest.setMaxUploads(50);
    do {
        listresult = client.ListMultipartUploads(listmultiu
ploadrequest);
        if (! listresult.isSuccess()) {
            /* Handle exceptions. */
            std::cout << "ListMultipartUploads fail" <<
            ",code:" << listresult.error().Code() <<
            ",message:" << listresult.error().Message() <<
            ",requestId:" << listresult.error().RequestId() << std::
endl;
            break;
        }
        else {
            for (const auto& part : listresult.result().MultipartU
ploadList()) {
                std::cout << "part"<<
```

```
                ",name:" << part.Key <<
                ",uploadid:" << part.UploadId <<
                ",initiated time:" << part.Initiated << std::endl;
            }
        }
        listmultiuploadrequest.setKeyMarker(listresult.result().
NextKeyMarker());
        listmultiuploadrequest.setUploadIdMarker(listresult.result
().NextUploadIdMarker());
    } while (listresult.result().IsTruncated());

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Cancel a multipart upload task

The following code provides an example on how to cancel a multipart upload task:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    InitiateMultipartUploadRequest initUploadRequest(BucketName,
ObjectName);

    /* Initiate the multipart upload task. */
    auto uploadIdResult = client.InitiateMultipartUpload(initUpload
Request);
    auto uploadId = uploadIdResult.result().UploadId();

    /* Cancel the multipart upload task. */
    AbortMultipartUploadRequest  abortUploadRequest(BucketName, key,
uploadId);
    auto abortUploadIdResult = client.AbortMultipartUpload(abortUploa
dRequest);

    if (! abortUploadIdResult.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "AbortMultipartUpload fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
```

```
    ShutdownSdk();
    return 0;
}
```

Note:

If a multipart upload task is canceled, the upload ID can no longer be used to perform further operation. The parts uploaded with that upload ID are also deleted.

5.6.6 Interrupt an upload task

You can interrupt an upload task at any time when it is being uploaded.

Use the following code to interrupt an upload task:

```
#include <alibabacloud/oss/OssClient.h>
#include <fstream>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName ";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::vector<PutObjectOutcomeCallable> Callables;
    for (int i=0; i < 4; i++)  {
        std::shared_ptr<std::iostream> content = std::make_shared<std
::fstream>("yourLocalFilename", std::ios::in|std::ios::binary);
        PutObjectRequest request(BucketName, ObjectName, content);
        auto outcomeCallable = client.PutObjectCallable(request);
        Callables.emplace_back(std::move(outcomeCallable));
    }
    std::this_thread::sleep_for(std::chrono::milliseconds(1000));

    /* Interrupt the upload task. */
    client.DisableRequest();

    for (size_t i = 0; i < Callables.size(); i++) {
        auto outcome = Callables[i].get();
        if (outcome.error(). Code() == "ClientError:100002" ||
        outcome.error(). Code() == "ClientError:200042") {
            std::cout << "disable putobject success" << std::endl;
        }
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Note:

An interrupted upload and a cancelled multipart upload have the following differences:

- **When you call the upload API to remotely upload an object to OSS, you can interrupt the upload task. In other words, you stopped the upload task. After you cancel the upload task, OSS no longer processes other multipart upload tasks that use the same upload IP as that of the cancelled task.**
- **If you interrupt the upload task, you can still call the upload ID to continue uploading the content although the current upload task is interrupted. If you cancel a multipart upload task, the upload ID is unavailable for future uploads.**
- **Interrupted upload applies to all APIs. However, cancelled multipart upload applies only to multipart upload.**

5.6.7 Use a progress bar

Progress bars are used to indicate the progress of an upload or download.

The following code uses PutObject as an example to show you how to view progress information:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

void ProgressCallback(size_t increment, int64_t transfered, int64_t
total, void* userData)
{
    std::cout << "ProgressCallback[" << userData << "] => " <<
    increment <<" ," << transfered << "," << total << std::endl;
}


int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    std::shared_ptr<std::iostream> content = std::make_shared<std::
fstream>("yourLocalFilename", std::ios::in|std::ios::binary);
```

```
    PutObjectRequest request(BucketName, ObjectName, content);

    TransferProgress progressCallback = { ProgressCallback , nullptr
 };
    request.setTransferProgress(progressCallback);

    /* Upload the object. */
    auto outcome = client.PutObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "PutObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.6.8 Upload callback

This topic describes how to use upload callback.

Use the following code for upload callback:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    /* Specify the IP address of the server you want to send the
callback request to, such as http://oss-demo.aliyunc s.com:23450. */
    std::string ServerName = "yourServerName";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "Thank you for using Aliyun Object Storage Service!" ;

    /* Set the upload callback parameters. */
    std::string callbackBody = "bucket=${bucket}&object=${object}&etag
=${etag}&size=${size}&mimeType=${mimeType}&my_var1=${x:var1}";
    ObjectCallbackBuilder builder(ServerName, callbackBody, "",
ObjectCallbackBuilder::Type::URL);
    std::string value = builder.build();
    ObjectCallbackVariableBuilder varBuilder;
    varBuilder.addCallbackVariable("x:var1", "value1");
```

```
    std::string varValue = varBuilder.build();
    PutObjectRequest request(BucketName, ObjectName, content);
    request.MetaData().addHeader("x-oss-callback", value);
    request.MetaData().addHeader("x-oss-callback-var", varValue);
    auto outcome = client.PutObject(request);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

For more information about upload callback, see the "*Upload callback*" section in OSS Developer Guide. For more information about debugging methods and troubleshooting, see *Upload callback errors and troubleshooting*. For more information about the implementation of upload callback, see the "*Callback*" section in API reference.

5.7 Download objects

5.7.1 Overview

The OSS C++ SDK provides the following object download methods:

- *Download to a local file*
- *Download to local memory*
- *Range download*
- *Resumable download*

You can view the download progress, as show in *Use a progress bar*.

5.7.2 Download to a local file

This topic describes how to download an object from OSS to a local file.

Use the following code to download an object to a local file:

```
#include <alibabacloud/oss/OssClient.h>
#include <memory>
#include <fstream>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    /* Set the name of the local file you want to download to. */
```

```
    std::string FileNametoSave = "yourFileName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Download the object to the local file. */
    GetObjectRequest request(BucketName, ObjectName);
    request.setResponseStreamFactory([=]() {return std::make_shared<
std::fstream>(FileNametoSave, std::ios_base::out | std::ios_base::in
 | std::ios_base::trunc| std::ios_base::binary); });

    auto outcome = client.GetObject(request);

    if (outcome.isSuccess()) {
        std::cout << "GetObjectToFile success" << outcome.result().
Metadata(). ContentLength() << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GetObjectToFile fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.7.3 Download to local memory

This topic describes how to download an object to local memory.

Run the following code to download an object to local memory:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf);

    /*Download the object to local memory.*/
    GetObjectRequest request(BucketName, ObjectName);
    auto outcome = client.GetObject(request);
```

```
    if (outcome.isSuccess()) {
        std::cout << "getObjectToBuffer" << " success, Content-Length
:" << outcome.result().Metadata().ContentLength() << std::endl;
    }
    else {
        /*Handle exceptions.*/
        std::cout << "getObjectToBuffer fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.7.4 Range download

If you only need part of the data in an object, you can use range download to download the specified content.

Use the following code for range download:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObectName ";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,  conf );

    /* Obtain the object. */
    GetObjectRequest request(BucketName,  ObjectName);
    /* Set the download range. */
    request.setRange(0, 1);
    auto outcome = client.GetObject(request);

    if (! outcome.isSuccess ()) {
        /* Handle exceptions. */
        std::cout << "getObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
         return -1;
    }

    /* Release network resources. */
```

```
    ShutdownSdk();
    return 0;
}
```

5.7.5 Resumable download

Large-sized objects may fail to be downloaded because the network is unstable or the program exits abnormally. Objects may still fail to be downloaded after multiple attempts. To address this problem, OSS provides resumable download.

To enable resumable download, you must split the object you want to download into multiple parts and download them separately. After you download all parts, these parts are combined into a complete object.

You can use OSSClient. ResumableDownloadObject to achieve resumable download. The following table describes the parameters contained in DownloadObjectRequest.

Name	Description	Required	Default value	Configuration method
bucket	The name of the bucket.	Yes	None	Constructor
key	The name of the object.	Yes	None	Constructor
filePath	The path to the local file. The file located in this path is downloaded to OSS.	No	Name of the object	Constructor
partSize	The size of each part. Valid values: [100 KB, 5 GB].	No	8 MB	setPartSize
threadNum	The number of parts you need to download simultaneously .	No	3	Constructor or setThreadNum

Name	Description	Required	Default value	Configuration method
checkpointDir	The file to record the results of multipart download. This parameter must be configured to enable resumable upload. This file stores information about download progress. If you fail to download a part, the part download continues based on the recorded progress. After the object is downloaded to your local device, the checkpoint file is deleted.	No	The same directory as that of DownloadFile	Constructor or setCheckpointDir

Use the following code for resumable download:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    std::string DownloadFilePath = "yourDownloadFilePath";
    std::string CheckpointFilePath = "yourCheckpointFilepath"
```

```
    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Start resumable download. */
    DownloadObjectRequest request(BucketName, ObjectName, DownloadFi
lePath, CheckpointFilePath);
    auto outcome = client.ResumableDownloadObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "ResumableDownloadObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.7.6 Use a progress bar

Progress bars are used to indicate the progress of an upload or download.

For the complete code of download progress bars, see *GitHub*.

Use the following code to view the progress information:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

void ProgressCallback(size_t increment, int64_t transfered, int64_t
total, void* userData)
{
    std::cout << "ProgressCallback[" << userData << "] => " <<
                 increment <<" ," << transfered << "," << total << std
::endl;
}

int main(void )
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName" ;

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeyS ecre t, conf);

    /* Obtain the object and its download progress information. */
```

```
    GetObjectRequest request(BucketName, ObjectName);
    TransferProgress progressCallback = { ProgressCallback , nullptr
 };
    request.setTransferProgress(progressCallback);

    auto outcome = client.GetObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "getObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk ();
         return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.8 Manage objects

5.8.1 Overview

You can use APIs to perform the following operations on objects in a bucket:

- *Manage the ACL of an object*
- *Manage Object Meta*
- *List objects*
- *Copy an object*
- *Delete objects*
- *Determine whether the specified object exists*
- *Restore an archive object*
- *Manage a symbolic link*

5.8.2 Determine whether the specified object exists

This topic describes how to determine whether an object exists.

Use the following code to determine whether an object exists:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
```

```
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Determine whether the object exists. */
    auto outcome = client.DoesObjectExist(BucketName, ObjectName);

    if (outcome) {
      std::cout << "The Object exists!" << std::endl;

    }
    else {
      std::cout << "The Object does not exist!" << std::endl;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.8.3 Manage the ACL of an object

This topic describes how to manage the access control list (ACL) of an object.

The following table describes the permissions included in the ACL of an object.

Permission	Description	Value
Inherit Bucket	**The ACL of an object is the same as that of its bucket.**	**CannedAccessControlList::Default**
Private	**Only the owner or authorized users have read and write permissions on the object.**	**CannedAccessControlList::Private**
Public Read	**Only the owner or authorized users of an object have read and write permissions on the object. Other users (including anonymous users) have only read permissions on the object. Use caution when using this operation.**	**CannedAccessControlList::PublicRead**

Permission	Description	Value
Public	All users of an object have read and write permissions on the object. Use caution when using this operation.	CannedAccessControlList::PublicReadWrite

The ACL privileges of objects take precedence over those of buckets. For example, the ACL of a bucket is Private, while the object ACL is Public. (All users of an object have read and write permissions on the object.) If an object does not have an ACL, the ACL of its bucket is used by default.

Configure an ACL of an object

Use the following code to configure an ACL for a specified object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";


    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Set the ACL of the object. */
    SetObjectAclRequest request(BucketName, ObjectName);
    request.setAcl(CannedAccessControlList::Private);
    auto outcome = client.SetObjectAcl(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetObjectAcl fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Obtain the ACL of an object

Use the following code to obtain an ACL to access a specified object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain the ACL of the object. */
    GetObjectAclRequest request(BucketName, ObjectName);
    auto outcome = client.GetObjectAcl(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "GetObjectAcl fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        std::cout << " GetObjectAcl success, Acl:" << outcome.result
(). Acl() << std::endl;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.8.4 Manage Object Meta

Object Meta contains HTTP headers and User Meta. For more information, see *Object Meta* in OSS Developer Guide.

Configure Object Meta

Use the following code to configure Object Meta:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;
```

```
int main(void)
{

    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret , conf);


    /* Set the HTTP header. */
    auto meta = ObjectMetaData();
    meta.setContentType("text/plain");
    meta.setCacheControl("max-ag e=3");
    /* Set User Meta. */
    meta.UserMetaData()["meta"] = "meta-value";

    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "Thank you for using Aliyun Object Storage Service!" ;
    client.PutObject(BucketName, ObjectName, content, meta);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "PutObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Obtain Object Meta

The following table lists the methods you can use to obtain Object Meta.

Method	Description	Benefit
GetObjectMeta	**Obtains some Object Meta such as the ETag, Size, and LastModified (the latest time an object was modified) values of the object.**	**More lightweight and faster**
HeadObject	**Obtains all Object Meta.**	**None**

Use the following code to obtain Object Meta:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain some Object Meta. */
    auto outcome = client.GetObjectMeta(BucketName, ObjectName);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "GetObjectMeta fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        auto metadata = outcome.result();
        std::cout << " get metadata success, ETag:" << metadata.ETag()
 << "; LastModified:"
        << metadata.LastModified() << "; Size:" << metadata.ContentLen
gth() << std::endl;
    }

    /* Obtain all Object Meta. */
    outcome = client.HeadObject(BucketName, ObjectName);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "HeadObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        auto headMeta = outcome.result();
        std::cout <<"headMeta success, ContentType:"
        << headMeta.ContentType() << "; ContentLength:" << headMeta.
ContentLength()
        << "; CacheControl:" << headMeta.CacheControl() << std::endl;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Note:

For more information about HTTP headers, see *RFC2616*.

5.8.5 Convert the storage class of an object

OSS provides the following storage classes to cover various data storage scenarios from hot data to cold data: Standard, Infrequent Access (IA), and Archive. This topic describes how to convert the storage class of an object.

For more information about these storage classes, see *#unique_43* and *#unique_77* in the OSS Developer Guide.

The following code provides an example of how to convert the storage class of an object.

- **The following code provides an example of how to convert the storage class of an object from Standard or IA to Archive:**

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string SourceBucketName = "yourSourceBucketName";
    std::string SourceObjectName = "yourSourceObjectName";


    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Set the storage class to Archive. */
    ObjectMetaData objectMeta;
    objectMeta.addHeader("x-oss-storage-class", "Archive");
    CopyObjectRequest request(SourceBucketName, SourceBucketName,
objectMeta);
    request.setCopySource(SourceBucketName, SourceObjectName);

    /* Modify the storage class. */
    auto outcome = client.CopyObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CopyObject fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
```

```
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **The following code provides an example of how to convert the storage class of an object from Archive to IA:**

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string SourceBucketName = "yourSourceBucketName";
    std::string SourceObjectName = "yourSourceObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Check whether the storage class of the source object is
Archive. If the storage class of the source object is Archive, you
must restore the object before you can modify the storage class. */
    auto restoreOutcome = client.RestoreObject(SourceBucketName,
SourceObjectName);
    if (! restoreOutcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "RestoreObject fail" <<
        ",code:" << restoreOutcome.error().Code() <<
        ",message:" << restoreOutcome.error().Message() <<
        ",requestId:" << restoreOutcome.error().RequestId() << std::
endl;
        ShutdownSdk();
        return -1;
    }

    std::string onGoingRestore("ongoing-request=\"false\"");

    int maxWaitTimeInSeconds = 600;
    while (maxWaitTimeInSeconds > 0)
    {
        auto meta = client.HeadObject(SourceBucketName, SourceObje
ctName);

        std::string restoreStatus = meta.result().HttpMetaData()["x-
oss-restore"];
        std::transform(restoreStatus.begin(), restoreStatus.end(),
restoreStatus.begin(), ::tolower);
        if (! restoreStatus.empty() &&
        restoreStatus.compare(0, onGoingRestore.size(), onGoingRes
tore)==0) {
```

```
            std::cout << " success, restore status:" << restoreSta
tus << std::endl;
            /* The Archive object is restored.*/
            break;
        }

        std::cout << " info, WaitTime:" << maxWaitTimeInSeconds
        << "; restore status:" << restoreStatus << std::endl;

        std::this_thread::sleep_for(std::chrono::seconds(10));
        maxWaitTimeInSeconds--;
    }

    /* Set the storage class to IA. */
    ObjectMetaData objectMeta;
    objectMeta.addHeader("x-oss-storage-class", "IA");
    CopyObjectRequest request(SourceBucketName, SourceBucketName,
objectMeta);
    request.setCopySource(SourceBucketName, SourceObjectName);

    /* Modify the storage class. */
    auto outcome = client.CopyObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CopyObject fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

For more information about the storage classes and storage fees, see the *#unique_78/unique_78_Connect_42_section_jmi_2z4_hka* section in Billing items.

5.8.6 List objects

This topic describes how to list the objects stored in a bucket.

Objects are listed alphabetically. You can use OssClient.ListObjects to list the objects in a bucket. ListObjects supports three types of parameter formats:

- **ListObjectOutcome ListObjects(const std::string& bucket) const: lists objects in a bucket. A maximum of 1,000 objects can be listed.**
- **ListObjectOutcome ListObjects(const std::string& bucket, const std::string& prefix) const: lists objects with the specified prefix in a bucket. A maximum of 1,000 objects can be listed.**
- **ListObjectOutcome ListObjects(const ListObjectsRequest& request) const: provides multiple filtering conditions to flexibly query objects.**

The following table describes the main parameters for object listing.

Parameter	Description
delimiter	**Groups objects by name. CommonPrefixes specifies a set of substrings of object names. The substrings start with the prefix and end with the next occurrence of the specified delimiter.**
prefix	**Specifies the prefix that returned object names must contain.**
maxKeys	**Specifies the maximum number of objects that can be listed each time. The default value is 100. The maximum value is 1000.**
marker	**Specifies the name of the object after which the list begins.**

Simple list

Run the following code to list objects in a specified bucket. A maximum of 100 objects can be listed by default.

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

     /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /*List the objects.*/
    ListObjectsRequest request(BucketName);
    auto outcome = client.ListObjects(request);

    if (! outcome.isSucces s())  {
        /*Handle exceptions.*/
        std::cout << "ListObjects fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
```

```
    else {
        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",lastmodify time:" << object.LastModified() << std:: endl
;
         }
    }

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

List a specified number of objects

Run the following code to list a specified number of objects:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

     /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf) ;

    /*List the objects.*/
    ListObjectsRequest request(BucketName);
    /*Set the maximum number of objects that can be listed.*/
    request.setMaxKeys(200);
    auto outcome = client.ListObjects(request);

    if (! outcome.isSuccess( )) {
        /*Handle exceptions.*/
        std::cout << "ListObjects fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",lastmodify time:" << object.LastModified() << std ::endl
;
        }
    }

    /*Release network resources.*/
```

```
    ShutdownSdk();
    return 0;
}
```

List objects and subdirectories in a specified directory

The following code provides an example on how to list the objects and subdirectories in a specified directory:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string keyPrefix = "yourkeyPrefix ";

     /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    std::string nextMarker = "";
    do {
        /*List the objects.*/
        ListObjectsRequest request(BucketName);
         /* Set the delimiter to a forward slash (/).*/
        request.setDelimiter("/");
        request.setPrefix(keyPrefix);
        request.setMarker(nextMarker);
        auto outcome = client.ListObjects(request);

        if (! outcome.isSuccess ()) {
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::endl;
            break;
        }
        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",lastmodify time:" << object.LastModified() << std::endl;
        }
        for (const auto& commonPrefix : outcome.result().CommonPref
ixes()) {
            std::cout << "commonPrefix" << ",name:" << commonPrefix
 << std::endl;
        }
        nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
```

```
}
```

List sizes of objects in a specified directory

The following code provides an example on how to query sizes of objects in a specified directory:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    static int64_t calculateFolderLength (const OssClient &client,
const std::string &bucketName, const std::string &folder)
    {
        std::string nextMarker = "";
        int64_t size = 0;
        do {
            /*List the objects.*/
            ListObjectsRequest request(BucketName);
            request.setPrefix(folder);
            request.setMarker(nextMarker);
            auto outcome = client.ListObjects(request);

            if (! outcome.isSuccess()) {
                /* Handle exceptions. */
                std::cout << "ListObjects fail" <<
                ",code:" << outcome.error().Code() <<
                ",message:" << outcome.error().Message() <<
                ",requestId:" << outcome.error().RequestId() << std::
endl;
                break;
            }

            for (const auto& object : outcome.result().ObjectSummarys
()) {
                size += object.Key()
            }

            nextMarker = outcome.result().NextMarker();
      } while (outcome.result().IsTruncated());

      return size;
    }

    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string keyPrefix = "yourkeyPrefix ";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);


    std::string nextMarker = "";
    do {
        /*List the objects that need to be encoded.*/
```

```
        ListObjectsRequest request(BucketName);
        /*Set the delimiter to a forward slash (/).*/
        request.setDelimiter("/");
        request.setPrefix(keyPrefix);
        request.setMarker(nextMarker);
        auto outcome = client.ListObjects(request);

        if (! outcome.isSuccess()) {
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::endl;
            break;
        }

        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object" <<
            ",name:" << object.Key() <<
            ",size:" << object.Size() << std::endl;
        }

        for (const auto& commonPrefix : outcome.result().CommonPref
ixes()) {
            int64_t foldersize = calculateFolderLength(client,
BucketName, commonPrefix);
            std::cout << "folder" <<
            ",name:" << commonPrefix <<
            ",size:" << foldersize << std::endl;
        }
        nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

List objects with a specified prefix

Run the following code to list objects with a specified prefix:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string keyPrefix = "yourkeyPrefix ";

     /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    std::string nextMarker = "";
    do {
```

```
        /*List the objects.*/
        ListObjectsRequest request(BucketName);
        /*Specify the prefix.*/
        request.setPrefix(keyPrefix);
        request.setMarker(nextMarker);
        auto outcome = client.ListObjects(request);

        if (! outcome.isSuccess ()) {
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::endl;
            break;
        }
        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",lastmodify time:" << object.LastModified() << std::endl;
        }
        nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

List the objects that follow a specified marker

The marker parameter indicates the name of an object from which the listing begins. Run the following code to specify an object (specified with marker) after which the listing begins:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string YourMarker = "yourMarker ";

     /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    do {
        /*List the objects.*/
        ListObjectsRequest request(BucketName);
         /*Specify the name of the object after which the list begins
.*/
        request.setMarker(YourMarker);
        auto outcome = client.ListObjects(request);
```

```
        if (! outcome.isSuccess())  {
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::endl;
            break;
        }
        YourMarker = outcome.result().NextMarker();
        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",lastmodify time:" << object.LastModified() << std::endl;
        }
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

Encode a specified object name

An object name that contains any of the following special characters needs to be encoded before it is transmitted. Only URL encoding is supported in OSS.

- **Single quotation marks (')**
- **Double quotations marks (")**
- **Ampersands (&)**
- **Angle brackets (<>)**
- **Chinese characters**

The following code provides an example on how to encode a specified object name:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

     /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    std::string nextMarker = "";
    do {
        /*List the objects.*/
        ListObjectsRequest request(BucketName) ;
        /*Encode the specified object name.*/
```

```
        request.setEncodingType("url");
        request.setMarker(nextMarker);
        auto outcome = client.ListObjects(request);

        if (! outcome.isSuccess())
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::endl;
            break;
        }
        for (const auto& object : outcome.result().ObjectSummarys()) {
            std::cout << "object"<<
            ",name:" << object.Key() <<
            ",size:" << object.Size() <<
            ",lastmodify time:" << object.LastModified() << std::endl;
        }
        nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

Folder simulation

OSS does not use folders. All elements are stored as objects. To simulate a folder on the OSS console, you actually create a 0 MB object whose name ends with a forward slash (/). This object can be uploaded and downloaded. By default, the OSS console displays an object whose name ends with a forward slash (/) as a folder.

The delimiter and prefix parameters can be used to simulate folder functions.

- **Prefix: specifies the prefix as the name of a folder. The folder is used to list all files (in the folder) and subfolders (directories in the folder) that start with this prefix. These files and subfolders are included in the Objects list.**
- **Delimiter: If the delimiter is specified as a forward slash (/), only the files and subfolders (directories) are displayed. Subfolders (directories) are included in the CommonPrefixes list, while the files and folders in subfolders are not displayed.**

For more information about folders, see *#unique_79/unique_79_Connect_42_section_bby_5yy_5db*. **For the complete code used to create a folder, visit** *GitHub*.

Assume that the following objects are stored in a bucket: oss.jpg, fun/test.jpg, fun/movie/001.avi, and fun/movie/007.avi. Forward slashes (/) are used as delimiters for folders. The subsequent examples show how to simulate folder functions.

- **List all objects in a bucket**

  Run the following code to list all objects in a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);


    ListObjectOutcome outcome;

    std::string nextMarker = "";

    do {
        /*List the objects.*/
        ListObjectsRequest request(BucketName);
        request.setMarker(nextMarker);
        outcome = client.ListObjects(request);

        if (! outcome.isSuccess()) {
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::
endl;
            ShutdownSdk();
            return -1;
        }
        else {
            for (const auto& object : outcome.result().ObjectSumm
arys()) {
                std::cout << "object"<<
                ",name:" << object.Key() <<
                ",size:" << object.Size() <<
                ",lastmodify time:" << object.LastModified() << std
::endl;
            }
        }
        nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
```

```
}
```

Example output:

```
Objects:
fun/movie/001.avi
fun/movie/007.avi
fun/test.jpg
oss.jpg

CommonPrefixes:
```

- **List all objects in a specified directory**

  Run the following code to list all objects in a specified folder:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    ListObjectOutcome outcome;
    std::string nextMarker = "";
    do {
            /*List the objects.*/
            ListObjectsRequest request(BucketName);
            request.setPrefix("fun/");
            request.setMarker(nextMarker);
            outcome = client.ListObjects(request);

            if (! outcome.isSuccess()) {
                /*Handle exceptions.*/
                std::cout << "ListObjects fail" <<
                ",code:" << outcome.error().Code() <<
                ",message:" << outcome.error().Message() <<
                ",requestId:" << outcome.error().RequestId() << std
::endl;
                break;
            }
            for (const auto& object : outcome.result().ObjectSumm
arys()) {
                std::cout << "object"<<
                ",name:" << object.Key() <<
                ",size:" << object.Size() <<
                ",lastmodify time:" << object.LastModified() << std
::endl;
            }
            for (const auto& commonPrefix : outcome.result().
CommonPrefixes()) {
```

```
                std::cout << "commonPrefix" << ",name:" <<
commonPrefix << std::endl;
            }
            nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

Example output:

```
Objects:
fun/movie/001.avi
fun/movie/007.avi
fun/test.jpg

CommonPrefixes:
```

- **List objects and subdirectories in a specified directory**

  Use the following code to list objects and subfolders in a folder:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    ListObjectOutcome outcome;
    std::string nextMarker = "";
    do {
            /*List the objects.*/
            ListObjectsRequest request(BucketName);
            /*Set the delimiter to a forward slash (/).*/
            request.setDelimiter("/");
            request.setPrefix("fun/");
            request.setMarker(nextMarker);
            outcome = client.GetObject(request);

            if (! outcome.isSuccess()) {
                /*Handle exceptions.*/
                std::cout << "ListObjects fail" <<
                ",code:" << outcome.error().Code() <<
                ",message:" << outcome.error().Message() <<
                ",requestId:" << outcome.error().RequestId() << std
::endl;
                break;
            }
```

```
            for (const auto& object : outcome.result().ObjectSumm
arys()) {
                std::cout << "object"<<
                ",name:" << object.Key() <<
                ",size:" << object.Size() <<
                ",lastmodify time:" << object.LastModified() << std
::endl;
            }
            for (const auto& commonPrefix : outcome.result().
CommonPrefixes()) {
                std::cout << "commonPrefix" << ",name:" <<
commonPrefix << std::endl;
            }
            nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

Example output:

```
Objects:
fun/test.jpg

CommonPrefixes:
fun/movie/
```

- **List the sizes of objects in a specified directory**

  The following code provides an example on how to query the sizes of objects in a specified directory:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

static int64_t calculateFolderLength (const OssClient &client, const
 std::string &bucketName, const std::string &folder)
{
    std::string nextMarker = "";
    int64_t size = 0;
    ListObjectOutcome outcome;
    do {
            /*List the objects.*/
            ListObjectsRequest request(bucketName);
            request.setPrefix(folder);
            request.setMarker(nextMarker);
            outcome = client.GetObject(request);

            if (! outcome.isSuccess()) {
                /*Handle exceptions.*/
                std::cout << "ListObjects fail" <<
                ",code:" << outcome.error().Code() <<
                ",message:" << outcome.error().Message() <<
                ",requestId:" << outcome.error().RequestId() << std
::endl;
                break;
            }
```

```
            for (const auto& object : outcome.result().ObjectSumm
arys()) {
                size += object.Size();
            }

            nextMarker = outcome.result().NextMarker();
        } while (outcome.result().IsTruncated());

      return size;
}

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    ListObjectOutcome outcome;
    std::string nextMarker = "";
    do {
        /*List the objects.*/
        ListObjectsRequest request(BucketName);
        /*Set the delimiter to a forward slash (/).*/
        request.setDelimiter("/");
        request.setPrefix("fun/");
        request.setMarker(nextMarker);
        outcome = client.ListObjects(request);

        if (! outcome.isSuccess()) {
            /*Handle exceptions.*/
            std::cout << "ListObjects fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::
endl;
            break;
        }

        for (const auto& object : outcome.result().ObjectSummarys())
 {
            std::cout << "object" <<
            ",name:" << object.Key() <<
            ",size:" << object.Size() << std::endl;
        }

        for (const auto& commonPrefix : outcome.result().CommonPref
ixes()) {
            int64_t foldersize = calculateFolderLength(client,
BucketName, commonPrefix);
            std::cout << "folder" <<
            ",name:" << commonPrefix <<
            ",size:" << foldersize << std::endl;
        }
        nextMarker = outcome.result().NextMarker();
    } while (outcome.result().IsTruncated());
```

```
    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

5.8.7 Delete objects

This topic describes how to delete objects.

Warning:

Deleted objects cannot be recovered. Exercise caution when you delete objects.

Delete an object

The following code provides an example on how to delete an object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* ObjectName indicates the complete path of the object you want
to delete from OSS, and must include the file extension of the object
. For example, set ObjectName to abc/efg/123.jpg. */
    /* To delete a folder, set ObjectName to the folder name. If the
 folder is not empty, you need to delete all objects in the folder
before you can delete the folder. */
    DeleteObjectRequest request(BucketName, ObjectName);

    /* Delete the object. */
    auto outcome = client.DeleteObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteObject fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Delete multiple objects

The following code provides an example on how to delete multiple objects at a time:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    DeleteObjectsRequest request(BucketName);
    /* Add the names of the objects you want to delete. */
    request.addKey(ObjectName);

    /* Delete the objects. */
    auto outcome = client.DeleteObjects(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteObjects fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.8.8 Copy an object

OSS allows you to copy objects of any sizes.

Copy a small object

You can use OssClient::CopyObject to copy an object smaller than 1 GB from a bucket (source bucket) to another bucket (destination bucket) within the same region. The configuration method is as follows:

Configuration method	Description
CopyObjectOutcome OssClient:: CopyObject(const CopyObjectRequest & request)	Allows you to specify destination Object Meta and copy conditions. If the URLs of the source object and destination object for the COPY operation are the same, the source Object Meta is directly replaced.

Note:

Object copy rules are as follows:

- **You must have read and write permissions on the source object.**
- **You cannot copy the object from a region to another region. For example, you cannot copy an object in a bucket from China (Hangzhou) to China (Qingdao).**
- **The object size is limited to a maximum of 1 GB.**

Use the following code to copy a small object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string SourceBucketName = "yourSourceBucketName";
    std::string CopyBucketName = "yourCopyBucketName";
    std::string SourceObjectName = "yourSourceObjectName";
    std::string CopyObjectName = "yourCopyObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    CopyObjectRequest request(CopyBucketName, CopyObjectName);
    request.setCopySource(SourceBucketName, SourceObjectName);

    /* Copy the object. */
    auto outcome = client.CopyObject(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CopyObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
```

```
    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Copy a large object

To copy an object larger than 1 GB, you must use UploadPartCopy. In other words, you must split the object into multiple parts and copy each of them sequentially. To enable UploadPartCopy, perform the following steps: Use ossClient.initiateMultipartUpload to initiate an UploadPartCopy task. Use ossClient.uploadPartCopy to perform the multipart copy task. Aside from the last part, all parts must be larger than 100 KB. Use $ossClient->completeMultipartUpload to submit the multipart copy task.

Note:

Large objects cannot be copied from one region to another region. For example, you cannot copy an object in a bucket from China (Hangzhou) to China (Qingdao).

Use the following code to copy a large object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string SourceBucketName = "yourSourceBucketName";
    std::string CopyBucketName = "yourCopyBucketName";
    std::string SourceObjectName = "yourSourceObjectName";
    std::string CopyObjectName = "yourCopyObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    auto getObjectMetaReq = GetObjectMetaRequest(SourceBucketName,
SourceObjectName);
    auto getObjectMetaResult = GetObjectMeta(getObjectMetaReq);
    if (! getObjectMetaResult.isSuccess()) {
        std::cout << "GetObjectMeta fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    /* Obtain the size of the object you want to copy. */
```

```
    auto objectSize = getObjectMetaResult.result(). ContentLength();

    /* Copy the large object. */
    InitiateMultipartUploadRequest initUploadRequest(CopyBucketName,
CopyObjectName);

    /* Initiate an UploadPartCopy task. */
    auto uploadIdResult = client.InitiateMultipartUpload(initUpload
Request);
    auto uploadId = uploadIdResult.result(). UploadId();
    int64_t partSize = 100 * 1024;
    PartList partETagList;
    int partCount = static_cast<int>(objectSize / partSize);
    /* Calculate the number of parts.*/
    if (objectSize % part Size ! = 0) {
        partCount++;
    }

    /* Copy each part sequentially. */
    for (int i = 1; i <= partCount; i++) {
        auto skipBytes = partSize * (i - 1);
        auto size = (partSize < objectSize - skipBytes) ? partSize : (
objectSize - skipBytes);
        auto uploadPartCopyReq = UploadPartCopyRequest(CopyBucketName
, CopyObjectName, SourceBucketName, SourceObjectName,uploadId, i + 1);
        uploadPartCopyReq.setCopySourceRange(skipBytes, skipBytes +
size -1);
        auto uploadPartOutcome = client.UploadPartCopy(uploadPart
CopyReq);
        if (uploadPartOutcome.isSuccess()) {
            Part part(i, uploadPartOutcome.result(). ETag());
            partETagList.push_back(part);
        }
        else {
            std::cout << "UploadPartCopy fail" <<
            ",code:" << outcome.error(). Code() <<
            ",message:" << outcome.error(). Message() <<
            ",requestId:" << outcome.error(). RequestId() << std::endl
;
        }

    }

    /* Complete the multipart copy task. */
    CompleteMultipartUploadRequest request(CopyBucketName, CopyObject
Name, partETagList, uploadId);
    auto outcome = client.CompleteMultipartUpload(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CompleteMultipartUpload fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

5.8.9 Restore an archive object

This topic describes how to restore an archive object.

You must restore an archive object before you can read it. Do not call RestoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

- **The archive object is initially in the frozen state.**
- **After you submit it for restoration, the server restores the object. The object is in the restoring state.**

  Note:
  Typically, it takes about one minute to restore an archive object.
- **You can read the object after it is restored. The restored state of the object lasts one day by default. You can prolong this period to a maximum of seven days. After this period expires, the object returns to the frozen state.**

Use the following code to restore an archive object:

```
#include <alibabacloud/oss/OssClient.h>
#include <thread>
#include <chrono>
using namespace AlibabaCloud::OSS;

int main(void)
{

    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Restore the archive object. */
    auto outcome = client.RestoreObject(BucketName, ObjectName);

    /* Non-archive objects cannot be restored. */
    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "RestoreObject fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
```

```
        ShutdownSdk();
        return -1;
    }

    std::string onGoingRestore("ongoing-request=\"false\"");

    int maxWaitTimeInSeconds = 600;
    while (maxWaitTimeInSeconds > 0)
    {
        auto meta = client.HeadObject(BucketName, ObjectName);

        std::string restoreStatus = meta.result(). HttpMetaData()["x-
oss-restore"];
        std::transform(restoreStatus.begin(), restoreStatus.end(),
restoreStatus.begin(), ::tolower);
        if (! restoreStatus.empty() &&
        restoreStatus.compare(0, onGoingRestore.size(), onGoingRestore
)==0) {
            std::cout << " success, restore status:" << restoreStatus
 << std::endl;
            /* The archive object is restored.*/
            break;
        }

        std::cout << " info, WaitTime:" << maxWaitTimeInSeconds
        << "; restore status:" << restoreStatus << std::endl;

        std::this_thread::sleep_for(std::chrono::seconds(10));
        maxWaitTimeInSeconds--;
    }

    if (maxWaitTimeInSeconds == 0)
    {
        std::cout << "RestoreObject fail, TimeoutException" << std::
endl;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

For more information about archive storage classes, see *Introduction to storage classes*.

For more information about related status codes, see *RestoreObject* **in API Reference.**

5.8.10 Manage a symbolic link

This document describes how to create a symbolic link and obtain the object content to which the symbolic link maps.

Create a symbolic link

A symbolic link is a special object that maps to an object. It is similar to a shortcut used in Windows. You can use a symbolic link to customize Object Meta.

Use the following code to create a symbolic link:

```
#include <alibabacloud/oss/OssClient.h>
```

```
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    std::string LinkName = "yourLinkName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);



    /* Set the HTTP header. */
    auto meta = ObjectMetaData();
    meta.setContentType("text/plain");

    /* Set User Meta. */
    meta.UserMetaData()["meta"] = "meta-value";

    /* Create the symbolic link. */
    CreateSymlinkRequest request(BucketName, ObjectName, meta);
    request.SetSymlinkTarget(LinkObjectName);
    auto outcome = client.CreateSymlink(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "CreateSymlink fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Obtain object content of a symbolic link

Use the following code to obtain the object content of a symbolic link:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string LinkName = "yourLinkName";
```

```
    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Obtain the object content of a symbolic link. */
    GetSymlinkRequest request(BucketName, LinkName);
    auto outcome = client.GetSymlink(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "GetSymlink fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        std::cout << " GetSymlink success Symlink name:" << outcome.
result(). SymlinkTarget() << std::endl;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Note:

To obtain a symbolic link, you must have the read permission on it. For more information about symbolic links, see *GetSymlink*.

5.9 Object tagging

5.9.1 Configure object tagging

This topic describes how to configure object tagging.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Note:

- **You can add tags to an object when uploading it or add tags for an uploaded object. If add tags to an object that already has tags, the original tags are overwritten. For more information about adding tags, see *#unique_87*.**
- **To add tags to an object, you must have the permission to call PutObjectTagging.**
- **The Last-Modified value of an object is not updated when its tags are changed.**

- **A tag can contain letters, numbers, spaces, and the following symbols:** + - = . _ : /

Add tags to an object when uploading it

- **Add tags to an object when uploading it directly.**

  Run the following code to add tags to an object when uploading it directly:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Sets x-oss-tagging to configure the tags to be added.*/
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    request.setTagging(tagging.toQueryParameters());

    /* Uploads the object. */
    auto outcome = client.PutObject(request);

    if (!outcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "PutObject fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

- **Add tags to an object when uploading it in the multipart upload method.**

  Run the following code to add tags to an object when uploading it in the multipart upload method:

```
#include <alibabacloud/oss/OssClient.h>
```

```
int64_t getFileSize(const std::string& file)
{
    std::fstream f(file, std::ios::in | std::ios::binary);
    f.seekg(0, f.end);
    int64_t size = f.tellg();
    f.close();
    return size;
}

using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    InitiateMultipartUploadRequest initUploadRequest(BucketName,
ObjectName);

    /* Sets x-oss-tagging to configure the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    initUploadRequest.setTagging(tagging.toQueryParameters());

    /* Initializes a multipart upload task. */
    auto uploadIdResult = client.InitiateMultipartUpload(initUpload
Request);
    auto uploadId = uploadIdResult.result().UploadId();
    std::string fileToUpload = "yourLocalFilename";
    int64_t partSize = 100 * 1024;
    PartList partETagList;
    auto fileSize = getFileSize(fileToUpload);
    int partCount = static_cast<int>(fileSize / partSize);
    /* Calculates the number of parts. */
    if (fileSize % partSize != 0) {
        partCount++;
    }

    /* Uploads each part. */
    for (int i = 1; i <= partCount; i++) {
        auto skipBytes = partSize * (i - 1);
        auto size = (partSize < fileSize - skipBytes) ? partSize : (
fileSize - skipBytes);
        std::shared_ptr<std::iostream> content = std::make_shared<
std::fstream>(fileToUpload, std::ios::in|std::ios::binary);
        content->seekg(skipBytes, std::ios::beg);

        UploadPartRequest uploadPartRequest(BucketName, ObjectName,
content);
        uploadPartRequest.setContentLength(size);
        uploadPartRequest.setUploadId(uploadId);
        uploadPartRequest.setPartNumber(i);
```

```
        auto uploadPartOutcome = client.UploadPart(uploadPartRequest
);
        if (uploadPartOutcome.isSuccess()) {
            Part part(i, uploadPartOutcome.result().ETag());
            partETagList.push_back(part);
        }
        else {
            std::cout << "uploadPart fail" <<
            ",code:" << uploadPartOutcome.error().Code() <<
            ",message:" << uploadPartOutcome.error().Message() <<
            ",requestId:" << uploadPartOutcome.error().RequestId()
 << std::endl;
        }

    }

    /* Completes the multipart upload task. */
    CompleteMultipartUploadRequest request(BucketName, ObjectName);
    request.setUploadId(uploadId);
    request.setPartList(partETagList);
    auto outcome = client.CompleteMultipartUpload(request);

    if (!outcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "CompleteMultipartUpload fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

- **Add tags to an object when uploading it by calling append_object.**

  Run the following code to add tags to an object when uploading it in the append object method:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    auto meta = ObjectMetaData();
    meta.setContentType("text/plain");
```

```
    /* Appends the file to the object. The default position from
where the file is appended is 0. */
    std::shared_ptr<std::iostream> content1 = std::make_shared<std::
stringstream>();
    *content1 <<"Thank you for using Aliyun Object Storage Service
!";
    AppendObjectRequest request(BucketName, ObjectName, content1,
meta);

    /* Configures the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    request.setTagging(tagging.toQueryParameters());

    /* Appends the file to the object. */
    auto result = client.AppendObject(request);

    if (!result.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "AppendObject fail" <<
        ",code:" << result.error().Code() <<
        ",message:" << result.error().Message() <<
        ",requestId:" << result.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

- **Add tags to an object when uploading it in the resumable upload method.**

  Run the following code to add tags to an object when uploading it in the resumable upload method:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    std::string UploadFilePath = "yourUploadfilePath";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Uploads the object in the resumable upload method. */
    UploadObjectRequest request(BucketName, ObjectName, YourUpload
FileName);
```

```
    /* Configures the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    request.setTagging(tagging.toQueryParameters());

    auto outcome = client.ResumableUploadObject(request);

    if (!outcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "ResumableUploadObject fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

Add tags to an uploaded object or modify the tags added to an object

Run the following code to add tags to an uploaded object or modify the tags added to an object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Uploads the object. */
    auto outcome = client.PutObject(request);

    /* Configures the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    auto putTaggingOutcome = client.SetObjectTagging(SetObjectT
aggingRequest(BucketName, ObjectName, tagging));

    if (!putTaggingOutcome.isSuccess()) {
        /* Handles exceptions. */
```

```
        std::cout << "SetObjectTagging fail" <<
        ",code:" << putTaggingOutcome.error().Code() <<
        ",message:" << putTaggingOutcome.error().Message() <<
        ",requestId:" << putTaggingOutcome.error().RequestId() << std
::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

Add tags to an object when copying it

You can specify one of the following tagging rules when copying an object:

- **Copy (default): Copy the tags of the source object to the destination object.**
- **Replace: Add the tags specified in the request to the destination object.**

Examples of adding tags to objects of different sizes when copying them are described as follows:

- **Run the following code to add tags to an object smaller than 1 GB when directly copying it:**

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    std::string CopyObjectName = "yourCopyObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Sets x-oss-tagging to configure the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    request.setTagging(tagging.toQueryParameters());

    /* Uploads the object. */
    auto outcome = client.PutObject(request);
```

```
    /* Copies the object and sets the tags to be added. */
    CopyObjectRequest cpRequest(yourBucketName, CopyObjectName);
    cpRequest.setCopySource(yourBucketName, yourObjectName);
    Tagging tagging2;
    tagging2.addTag(Tag("key1-2", "value1-2"));
    tagging2.addTag(Tag("key2-2", "value2-2"));
    tagging2.addTag(Tag("key3-2", "value3-2"));
    cpRequest.setTagging(tagging2.toQueryParameters());

    /* Ignores the tags of the source object. Adds the tags
specified in the request to the object. */
    cpRequest.setTaggingDirective(CopyActionList::Replace);

    /* Copies the object. */
    auto copyoutcome = client.CopyObject(cpRequest);
    if (!copyoutcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "CopyObject fail" <<
        ",code:" << copyoutcome.error().Code() <<
        ",message:" << copyoutcome.error().Message() <<
        ",requestId:" << copyoutcome.error().RequestId() << std::
endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

- **Run the following code to add tags to an object larger than 1 GB when copying it the multipart upload method:**

```
#include <alibabacloud/oss/OssClient.h>
#include <sstream>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string SourceBucketName = "yourSourceBucketName";
    std::string CopyBucketName = "yourCopyBucketName";
    std::string SourceObjectName = "yourSourceObjectName";
    std::string CopyObjectName = "yourCopyObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    auto getObjectMetaReq = GetObjectMetaRequest(SourceBucketName,
SourceObjectName);
    auto getObjectMetaResult = GetObjectMeta(getObjectMetaReq);
    if (!getObjectMetaResult.isSuccess()) {
        std::cout << "GetObjectMeta fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
```

```
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }
    /* Obtains the size of the source object. */
    auto objectSize = getObjectMetaResult.result().ContentLength();

    /* Copies the large object. */
    ObjectMetaData metaData;
    std::stringstream ssDesc;
    ssDesc << "/" << SourceBucketName << "/" << SourceObjectName;
    std::string value = ssDesc.str();
    metaData.addHeader("x-oss-copy-source", value);
    InitiateMultipartUploadRequest initUploadRequest(BucketName,
ObjectName, metaData);

    /* Sets x-oss-tagging to configure the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    initUploadRequest.setTagging(tagging.toQueryParameters());

    /* Initializes a multipart copy task. */
    auto uploadIdResult = client.InitiateMultipartUpload(initUpload
Request);
    auto uploadId = uploadIdResult.result().UploadId();
    int64_t partSize = 100 * 1024;
    PartList partETagList;
    int partCount = static_cast<int>(objectSize / partSize);
    /* Calculates the number of parts.*/
    if (objectSize % partSize != 0) {
        partCount++;
    }

    /* Copies each part. */
    for (int i = 1; i <= partCount; i++) {
        auto skipBytes = partSize * (i - 1);
        auto size = (partSize < objectSize - skipBytes) ? partSize :
 (objectSize - skipBytes);
        auto uploadPartCopyReq = UploadPartCopyRequest(CopyBucket
Name, CopyObjectName, SourceBucketName, SourceObjectName,uploadId, i
 + 1);
        uploadPartCopyReq.setCopySourceRange(skipBytes, skipBytes +
size -1);
        auto uploadPartOutcome = client.UploadPartCopy(uploadPart
CopyReq);
        if (uploadPartOutcome.isSuccess()) {
            Part part(i, uploadPartOutcome.result().ETag());
            partETagList.push_back(part);
        }
        else {
            std::cout << "UploadPartCopy fail" <<
            ",code:" << outcome.error().Code() <<
            ",message:" << outcome.error().Message() <<
            ",requestId:" << outcome.error().RequestId() << std::
endl;
        }

    }

    /* Completes the multipart copy task. */
    CompleteMultipartUploadRequest request(CopyBucketName,
CopyObjectName, partETagList, uploadId);
    auto outcome = client.CompleteMultipartUpload(request);
```

```
    if (!outcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "CompleteMultipartUpload fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

Add tags to a symbolic link object

Run the following code to add tags to a symbolic link object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";
    std::string LinkName = "yourLinkName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Uploads the object. */
    auto outcome = client.PutObject(request);

    /* Creates a symbolic link. */
    CreateSymlinkRequest csRequest(BucketName, ObjectName);
    csRequest.SetSymlinkTarget(LinkName);

    /* Sets x-oss-tagging to configure the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    csRequest.setTagging(tagging.toQueryParameters());

    auto csoutcome = client.CreateSymlink(csRequest);

    if (!csoutcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "CreateSymlink fail" <<
        ",code:" << csoutcome.error().Code() <<
```

```
        ",message:" << csoutcome.error().Message() <<
        ",requestId:" << csoutcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

5.9.2 Obtain the tags added to an object

This topic describes how to obtain the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to obtain the tags added to an object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Uploads the object. */
    auto outcome = client.PutObject(request);

    /* Configures the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
    auto putTaggingOutcome = client.SetObjectTagging(SetObjectT
aggingRequest(BucketName, ObjectName, tagging));

    /* Obtains the tags added to the object. */
    getTaggingOutcome = client.GetObjectTagging(GetObjectTaggingRequ
est(BucketName, key));

    if (!getTaggingOutcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "getTaggingOutcome fail" <<
        ",code:" << getTaggingOutcome.error().Code() <<
        ",message:" << getTaggingOutcome.error().Message() <<
```

```
        ",requestId:" << getTaggingOutcome.error().RequestId() << std
::endl;
        ShutdownSdk();
        return -1;
    }
    else {
        auto taglist = getTaggingOutcome.result().Tagging();
        for (const auto& tag : taglist)
        {
          std::cout <<"GetObjectTagging success, Key:"
          << tag.Key() << "; Value:" << tag.Value() << std::endl;
        }
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

For more information about how to obtain the tags added to an object, see *#unique_89*.

5.9.3 Delete the tags added to an object

This topic describes how to delete the tags added to an object.

The object tagging function uses a key-value pair to tag an object. For more information about the object tagging function, see *#unique_86*.

Run the following code to delete the tags added to an object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";
    PutObjectRequest request(BucketName, ObjectName, content);

    /* Uploads the obje */
    auto outcome = client.PutObject(request);

    /* Configures the tags to be added. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));
```

```
    auto putTaggingOutcome = client.SetObjectTagging(SetObjectT
aggingRequest(BucketName, ObjectName, tagging));

    /* Deletes the tags added to the object. */
    auto delTaggingOutcome = client.DeleteObjectTagging(DeleteObje
ctTaggingRequest(BucketName, ObjectName));

    if (!delTaggingOutcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "delTaggingOutcome fail" <<
        ",code:" << delTaggingOutcome.error().Code() <<
        ",message:" << delTaggingOutcome.error().Message() <<
        ",requestId:" << delTaggingOutcome.error().RequestId() << std
::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

For more information about how to delete the tags added to an object, see *#unique_91*.

5.9.4 Object tagging and lifecycle rules

Lifecycle rules can take effect on objects with specified prefixes or tags. You can also specify prefixes and tags as the condition of a lifecycle rule at the same time.

Note:

If you set a specified tag as the condition of a lifecycle rule, the rule applies to an object only when the key and value in the tag of the object both match the specified tag. If you specify a prefix and multiple tags as the condition of a lifecycle rule, the rule applies to an object only when the prefix and tags of the object match the prefix and all tags specified in the rule.

Set tags as the matching condition of a lifecycle rule

Run the following code to set tags as the matching condition of a lifecycle rule:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initializes resources such as networks. */
    InitializeSdk();
```

```
    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    SetBucketLifecycleRequest request(BucketName);

    /* Configures the tags to be set as the matching condition. */
    Tagging tagging;
    tagging.addTag(Tag("key1", "value1"));
    tagging.addTag(Tag("key2", "value2"));

    /* Specifies the lifecycle rule for the object. */
    auto rule1 = LifecycleRule();
    rule1.setID("rule1");
    rule1.setPrefix("test1/");
    rule1.setStatus(RuleStatus::Enabled);
    rule1.setExpiration(3);
    rule1.setTags(tagging.Tags());

    /* Sets the tags as the matching condition of the lifecycle rule.
 */
    LifecycleRuleList list{rule1};
    request.setLifecycleRules(list);
    auto outcome = client.SetBucketLifecycle(request);

    if (!outcome.isSuccess()) {
        /* Handles exceptions. */
        std::cout << "SetBucketLifecycle fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

View the tags set as the matching condition of a lifecycle rule

Run the following code to view the tags set as the matching condition of a lifecycle rule:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initializes the Alibaba Cloud OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /* Initializes resources such as networks. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);
```

```
    /* Views the lifecycle rule that includes the tags set as the
matching condition. */
    auto outcome = client.GetBucketLifecycle(BucketName);

    if (outcome.isSuccess()) {
        std::cout << "GetBucketLifecycle success," << std::endl;
        for (auto const rule : outcome.result().LifecycleRules()) {
            std::cout << "rule:" << rule.ID() << "," << rule.Prefix()
 << "," << rule.Status() << ","
            "hasExpiration:" << rule.hasExpiration() << "," <<
            "hasTransitionList:" << rule.hasTransitionList() << ","
 << std::endl;

          auto taglist = rule.Tags();
          for (const auto& tag : taglist)
          {
              std::cout <<"GetBucketLifecycle tag success, Key:"
              << tag.Key() << "; Value:" << tag.Value() << std::endl;
          }
        }
    }
    else {
        /* Handles exceptions. */
        std::cout << "GetBucketLifecycle fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Releases resources such as networks. */
    ShutdownSdk();
    return 0;
}
```

5.10 Single-connection bandwidth throttling

This topic describes how to ensure other applications have sufficient bandwidth when you perform upload and download operations on OSS. You can include the x-oss-traffic-limit parameter in your requests and set bandwidth limits.

Note:

For more information about the scenarios and precautions of the single-connection bandwidth throttling feature, see *#unique_93* in the OSS Developer Guide.

Configure bandwidth throttling for common uploads and downloads

Use the following code to configure bandwidth throttling when you upload and download objects to and from OSS:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
```

```
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Upload the object. */
    std::shared_ptr<std::iostream> content = std::make_shared<std::
fstream>("yourLocalFilename", std::ios::in|std::ios::binary);
    PutObjectRequest putrequest(BucketName, ObjectName,content);
    /* Set the upload bandwidth limit to 100 KB/s. */
    getrequest.setTrafficLimit(819200);
    auto putoutcome = client.PutObject(putrequest);

    /* Download the object to local memory. */
    GetObjectRequest getrequest(BucketName, ObjectName);
    /* Set the download bandwidth limit to 100 KB/s. */
    getrequest.setTrafficLimit(819200);
    auto getoutcome = client.GetObject(getrequest);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Configure bandwidth throttling for uploads and downloads that use signed URLs

Use the following code to configure bandwidth throttling when you upload and download objects from and to OSS by using signed URLs.

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string PutobjectUrlName = "yourPutobjectUrlName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Specify the valid period for the URL.*/
    std::time_t t = std::time(nullptr) + 1200;
    /* Generate the signed URL for uploading an object. */
    GeneratePresignedUrlRequest putrequest(BucketName, PutobjectU
rlName, Http::Put);
    putrequest.setExpires(expires);
```

```
    /* Set the upload bandwidth limit to 100 KB/s. */
    putrequest.setTrafficLimit(819200);
    auto genOutcome = client.GeneratePresignedUrl(putrequest);

    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";

    /* Use the signed URL to upload the object. */
    auto outcome = client.PutObjectByUrl(genOutcome.result(), content
);

    /* Generate the signed URL for downloading an object. */
    GeneratePresignedUrlRequest getrequest(BucketName, PutobjectU
rlName, Http::Get);
    getrequest.setExpires(expires);
    /* Set the download bandwidth limit to 100 KB/s. */
    getrequest.setTrafficLimit(819200);
    genOutcome = client.GeneratePresignedUrl(getrequest);

    /* Use the signed URL to download the object. */
    outcome = client.GetObjectByUrl(genOutcome.result());

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.11 Server-side encryption

OSS supports server-side encryption for uploaded data. When you upload data, OSS encrypts the data and stores the encrypted data. When you download data, OSS decrypts the data and returns the original data. The returned HTTP request header declares that the data has been encrypted on the server.

OSS provides two methods of server-side encryption:

- **Server-side encryption with CMKs stored in KMS (SSE-KMS)**

  When you upload objects to OSS, you can use a specified CMK ID or a CMK stored in KMS to encrypt data. This method is cost-effective because you do not need to send user data to the KMS server over networks for encryption and decryption.

  Notice:

  You are charged for making API calls when you use CMKs to encrypt or decrypt data.

- **Server-side encryption with OSS-managed keys (SSE-OSS)**

  When you upload objects to OSS, the OSS server uses the fully managed 256-bit AES keys to encrypt data. Each object is encrypted with a unique data key. The

data key is further encrypted using a master key that is rotated on a regular basis to ensure the security of the data.

Notice:

- **Only one server-side encryption method can be used for an object at a time.**
- **If you configure server-side encryption for a bucket, you can still configure an encryption method for individual objects in the bucket when uploading or copying objects. In this case, the encryption method configured for the object takes precedence. For more information, see *PutObject*.**
- **For more information about server-side encryption, see *#unique_99*.**

Configure server-side encryption for a bucket

You can use the following code to configure the default encryption method for a bucket. The encryption method of the bucket applies to all uploaded objects that do not have the encryption method configured.

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    SetBucketEncryptionRequest setrequest(BucketName);
    setrequest.setSSEAlgorithm(SSEAlgorithm::KMS);
undefined   /*Configure KMS-based server-side encryption for a bucket
.*/
    auto outcome = client.SetBucketEncryption(setrequest);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "SetBucketEncryption fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
```

```
}
```

Obtain the encryption configurations of a bucket

Use the following code to obtain the encryption configurations of a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /*Obtain the server-side encryption configurations of a bucket.*/

    GetBucketEncryptionRequest request(BucketName);
    auto outcome = client.GetBucketEncryption(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "GetBucketEncryption fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

Delete the encryption configurations for a bucket

Use the following code to delete the encryption configurations for a bucket:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /*Initialize the OSS account information.*/
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";

    /*Initialize network resources.*/
    InitializeSdk();
```

```
    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /*Delete the server-side encryption configurations for a bucket.*/

    DeleteBucketEncryptionRequest request(BucketName);
    auto outcome = client.DeleteBucketEncryption(request);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "DeleteBucketEncryption fail" <<
        ",code:" << outcome.error().Code() <<
        ",message:" << outcome.error().Message() <<
        ",requestId:" << outcome.error().RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /*Release network resources.*/
    ShutdownSdk();
    return 0;
}
```

5.12 Authorized access

This topic describes how to authorize access to OSS.

Use STS for temporary access authorization

You can use Alibaba Cloud STS to authorize temporary access to OSS. STS is a Web service that provides a temporary access token to a cloud computing user. You can use STS to grant a third-party application or a RAM user (whose user ID is managed by yourself) an access credential with a customized validity period and permissions. For more information about STS, see *STS Introduction*.

STS benefits:

- **You do not need to expose your long-term key (AccessKey) to a third-party application. You need only to generate an access token and send the access token to the third-party application. You can customize the access permissions and validity period of this token.**
- **The access token automatically becomes invalid after expiration.**

For more information about the process of access to OSS with STS, see the "*Identity authentication*" section in OSS Developer Guide.

Use the following code to create a signature request with STS:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;
```

```
int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string SecurityToken  = "securityToken";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret,
SecurityToken, conf)

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

Use a signed URL to authorize temporary access

You can provide a signed URL to a visitor for temporary access. When you generate a signature for a URL, you can specify the valid period for the URL. After the valid period expires, access from visitors is restricted.

- **Use a signed URL to upload an object**

  Use the following code to generate a signed URL to upload an object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string PutobjectUrlName = "yourPutobjectUrlName" ;

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Specify the valid period for the URL.*/
    std::time_t t = std::time(nullptr) + 1200;
    /* Generate the signed URL for uploading an object. */
    auto genOutcome = client.GeneratePresignedUrl(BucketName,
PutobjectUrlName, t, Http::Put);
    if (genOutcome.isSuccess()) {
        std::cout << "GeneratePresignedUrl success, Gen url:" <<
genOutcome.result().c_str() << std::endl;
    }
    else {
        /* Handle exceptions. */
        std::cout << "GeneratePresignedUrl fail" <<
```

```
        ",code:" << genOutcome.error(). Code() <<
        ",message:" << genOutcome.error(). Message() <<
        ",requestId:" << genOutcome.error(). RequestId() << std::
endl;
        ShutdownSdk();
        return -1;
    }

    std::shared_ptr<std::iostream> content = std::make_shared<std::
stringstream>();
    *content << "test cpp sdk";

    /* Use the URL to upload the object. */
    auto outcome = client.PutObjectByUrl(genOutcome.result(),
content);

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "PutObjectByUrl fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **Use a signed URL to download an object**

  Use the following code to generate a signed URL to download an object:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
    /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string GetobjectUrlName = "yourGetobjectUrlName";

    /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Specify the valid period for the URL.*/
    std::time_t t = std::time(nullptr) + 1200;
    /* Generate the URL to download the object. */
    auto genOutcome = client.GeneratePresignedUrl(BucketName,
GetobjectUrlName, t, Http::Get);
    if (genOutcome.isSuccess()) {
        std::cout << "GeneratePresignedUrl success, Gen url:" <<
genOutcome.result().c_str() << std::endl;
    }
    else {
```

```
        /* Handle exceptions. */
        std::cout << "GeneratePresignedUrl fail" <<
        ",code:" << genOutcome.error(). Code() <<
        ",message:" << genOutcome.error(). Message() <<
        ",requestId:" << genOutcome.error(). RequestId() << std::
endl;
        ShutdownSdk();
        return -1;
    }

    /* Use the URL to download the object. */
    auto outcome = client.GetObjectByUrl(genOutcome.result());

    if (! outcome.isSuccess()) {
        /* Handle exceptions. */
        std::cout << "GetObjectByUrl fail" <<
        ",code:" << outcome.error(). Code() <<
        ",message:" << outcome.error(). Message() <<
        ",requestId:" << outcome.error(). RequestId() << std::endl;
        ShutdownSdk();
        return -1;
    }

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

5.13 IMG

Image Processing (IMG) is a secure, cost-effective, and highly reliable image processing service provided by OSS. IMG is capable of processing large amounts of images. After source images are uploaded to OSS, you can use Representational State Transfer (RESTful) APIs to process images anytime, anywhere, and on any Internet device.

For more information about IMG, see *OSS Image Processing Guide*.

Basic features

IMG has the following features:

- *Obtain image information*
- *Convert formats*
- *Resize images*
- *Crop images*
- *Rotate images*
- *Apply effects*

- *Add watermarks*: allows you to add image, text, or image-text watermarks to another image.
- *Customize image styles*
- *Cascade operations*: calls multiple IMG operations.

How to use IMG

IMG uses the standard HTTP GET request. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is Private, only authorized users are allowed access.

- Anonymous access

  You can use level-3 domains to access processed images. The access format is as follows:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Name	Description
bucket	The name of the bucket.
endpoint	The endpoint used to access the region to which the bucket belongs.
object	The name of the image object.
image	The reserved identifier for IMG.
action	Operations on the image, such as resizing, cropping, and rotating.

Name	Description
param	The parameter corresponding to the operations performed on the image.

- Basic operations

  For example, scale an image to a width of 100 px. Adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- Customize an image style

  You can use level-3 domains to access processed images. The access format is as follows:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=style/<yourStyleName>
```

  - style: specifies the reserved identifier of a custom image style.
  - yourStyleName: specifies the name of a custom image style. It is the name of the rule that is specified in the OSS console.

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=style/oss-pic-style-w-100
```

- Cascade operations

  Cascade operations allow you to perform multiple operations on an image in sequence. The format is as follows:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss
-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<
yourParamValue2>/...
```

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100/rotate,90
```

- Access through HTTPS

IMG supports access through HTTPS. Example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  Authorized access allows you to access an image through HTTPS, customize the image style, and perform cascade operations.

  Use the following code to generate a signed URL for IMG:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
      /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

      /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);


    /* Generate the signed URL. */
    std::string Process = "image/resize,m_fixed,w_100,h_100";
    GeneratePresignedUrlRequest request(BucketName, ObjectName, Http
::Get);
    request.setProcess(Process);
    auto outcome = client.GeneratePresignedUrl(request);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **Access with an SDK**

  You can use an SDK to access and process an image object.

  The SDK allows you to access an image through HTTPS, customize the image style, and perform cascade operations.

  - **Basic operations**

    Use the following code to perform basic operations on an image:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;
```

```
int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf
);

    /* Scale the image. */
    std::string Process = "image/resize,m_fixed,w_100,h_100";
    GetObjectRequest request(BucketName, ObjectName);
    request.setProcess(Process);
    auto outcome = client.GetObject(request);

    /* Crop the image. */
    Process = "image/crop,w_100,h_100,x_100,y_100,r_1";
    request.setProcess(Process);
    outcome = client.GetObject(request);

    /* Rotate the image. */
    Process = "image/rotate,90";
    request.setProcess(Process);
    outcome = client.GetObject(request);

    /* Sharpen the image. */
    Process = "image/sharpen,100";
    request.setProcess(Process);
    outcome = client.GetObject(request);

    /* Add the watermark. */
    Process = "image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ";
    request.setProcess(Process);
    outcome = client.GetObject(request);

    /* Convert the image format. */
    Process = "image/format,png";
    request.setProcess(Process);
    outcome = client.GetObject(request);

    /* Obtain image information. */
    request = request(BucketName, ObjectName, "image/info") ;
    outcome = client.GetObject(request);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **Customize an image style**

  Use the following code to customize an image style:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;
```

```
int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf
) ;

    /* Customize the image style. */
    std::string Process = "style/<yourCustomStyleName";
    GetObjectRequest request(BucketName, ObjectName);
    request.setProcess(Process);
    auto outcome = client.GetObject(request);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

- **Cascade operations**

  Use the following code to perform cascade operations on an image:

```
#include <alibabacloud/oss/OssClient.h>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string ObjectName = "yourObjectName";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf
);

    /* Perform cascade operations. */
    std::string Process = "image/resize,m_fixed,w_100,h_100/rotate
,90";
    GetObjectRequest request(BucketName, ObjectName);
    request.setProcess(Process);
    auto outcome = client.GetObject(request);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
```

```
}
```

Implement permanent storage for a processed image

Use the following code to store a processed image in a bucket for permanent storage:

```
#include <alibabacloud/oss/OssClient.h>
#include <sstream>
using namespace AlibabaCloud::OSS;

int main(void)
{
     /* Initialize the OSS account information. */
    std::string AccessKeyId = "yourAccessKeyId";
    std::string AccessKeySecret = "yourAccessKeySecret";
    std::string Endpoint = "yourEndpoint";
    std::string BucketName = "yourBucketName";
    std::string SourceObjectName = "yourSourceObjectName";
    std::string TargetObjectName = "yourTargetObjectName";

     /* Initialize network resources. */
    InitializeSdk();

    ClientConfiguration conf;
    OssClient client(Endpoint, AccessKeyId, AccessKeySecret, conf);

    /* Scale the image. */
    std::string Process = "image/resize,m_fixed,w_100,h_100";
    std::stringstream ss;
    ss  << Process
    <<"|sys/saveas"
    << ",o_" << Base64EncodeUrlSafe(TargetObjectName)
    << ",b_" << Base64EncodeUrlSafe(BucketName);
    ProcessObjectRequest request(BucketName, SourceObjectName, ss.str
());
    auto outcome = client.ProcessObject(request);

    /* Crop the image. */
    Process = "image/crop,w_100,h_100,x_100,y_100,r_1";
    ss.str("");
    ss  << Process
    <<"|sys/saveas"
    << ",o_" << Base64EncodeUrlSafe(TargetObjectName)
    << ",b_" << Base64EncodeUrlSafe(BucketName);
    request = request(BucketName, SourceObjectName, ss.str());
    outcome = client.ProcessObject(request);

    /* Rotate the image. */
    Process = "image/rotate,90";
    ss.str("");
    ss  << Process
    <<"|sys/saveas"
    << ",o_" << Base64EncodeUrlSafe(TargetObjectName)
    << ",b_" << Base64EncodeUrlSafe(BucketName);
    request = request(BucketName, SourceObjectName, ss.str());
    outcome = client.ProcessObject(request);

    /* Sharpen the image. */
    Process = "image/sharpen,100";
    ss.str("");
    ss  << Process
```

```
    <<"|sys/saveas"
    << ",o_" << Base64EncodeUrlSafe(TargetObjectName)
    << ",b_" << Base64EncodeUrlSafe(BucketName);
    request = request(BucketName, SourceObjectName, ss.str());
    outcome = client.ProcessObject(request);

    /* Add the watermark. */
    Process = "image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ";
    ss.str("");
    ss  << Process
    <<"|sys/saveas"
    << ",o_" << Base64EncodeUrlSafe(TargetObjectName)
    << ",b_" << Base64EncodeUrlSafe(BucketName);
    request = request(BucketName, SourceObjectName, ss.str());
    outcome = client.ProcessObject(request);

    /* Convert the image format. */
    Process = "image/format,png";
    ss.str("");
    ss  << Process
    <<"|sys/saveas"
    << ",o_" << Base64EncodeUrlSafe(TargetObjectName)
    << ",b_" << Base64EncodeUrlSafe(BucketName);
    request = request(BucketName, SourceObjectName, ss.str());
    outcome = client.ProcessObject(request);

    /* Release network resources. */
    ShutdownSdk();
    return 0;
}
```

6 PHP

6.1 Preface

Source code

For the source code, see *GitHub*.

Code samples

OSS SDK for PHP provides a variety of code samples for your reference or use. The following table describes the content of code samples:

Sample file	Content
Object.php	**Object operations, including *upload objects*, *download objects*, and *manage objects***
MultipartUpload.php	*Multipart upload*
Signature.php	*Authorized access with the signed URL*
Callback.php	*Upload callback*
Image.php	*Image processing*
Bucket.php	**Bucket operations, including *create a bucket*, *delete a bucket*, *list buckets*, and *ACL*.**
BucketLifecycle.php	**Configure, view, and delete the *lifecycle rules* of a bucket**
BucketLogging.php	**Configure, view, and delete the *access log files* of a bucket**
BucketReferer.php	**Configure, view, and delete the *Hotlink protection* settings of a bucket**
BucketWebsite.php	**Configure, view, and delete the *static website hosting* settings of a bucket**
BucketCors.php	**Configure, view, and delete the *CORS* settings of a bucket**

6.2 Installation

Preparation

OSS PHP SDK is applicable to PHP 5.3 and later versions. PHP 5.6.22 is used in this topic as an example.

- **Installation environment**

  You must install PHP cURL extensions:

  - **In Windows, see** *Compile and use Alibaba Cloud OSS PHP SDK in Windows* **to install PHP cURL extension. In Windows, if you cannot find the required module, specify the extension_dir to** `C:/Windows/System32/` **in php.ini.**
  - **In Ubuntu, you can use the apt-get package manager to install the PHP cURL extension by running** `sudo apt-get install php-curl`**.**
  - **In CentOS, you can use the yum package manager to install the PHP cURL extension by running** `sudo yum install php-curl`**.**

- **Check PHP version**

  - **Run** `php -v` **to check the current PHP version.**
  - **Run** `php -m` **to check whether cURL extension is installed.**

Download SDK

- *Download from GitHub*
- *Download previous versions*

For more information, see *OSS API documents*.

> **Note:**
> **We recommend that you use the latest version of the SDK. You can download documents of OSS PHP SDK 2.0.0 and earlier versions from** *here*.

Install SDK

You can install the SDK in three methods:

- **Composer**

  You can install the SDK through composer as follows:

  1. **Run `composer require aliyuncs/oss-sdk-php` in the root directory of your project, or declare the dependency on Alibaba Cloud OSS SDK for PHP in `composer.json` as follows:**

     ```
     "require": {
         "aliyuncs/oss-sdk-php": "~2.x.x"
     }
     ```

  2. **Run `composer install` to install the dependency. After the dependency is installed, check whether your directory structure complies with the following structure:**

     ```
     .
     ├── app.php
     ├── composer.json
     ├── composer.lock
     └── vendor
     ```

     The `app.php` is a user application. The `vendor/` directory contains the dependent library. You must introduce the dependency in `app.php`:

     ```
     require_once __DIR__ . '/vendor/autoload.php';
     ```

  Note:

  - **If your project references `autoload.php`, you do not need to introduce the file again after adding the SDK dependency.**
  - **If a network error occurs when you use the Composer Dependency Manager, you can use the *Image Source* in the Composer China region by running the following command: `composer config -g repositories.packagist composer http://packagist.phpcomposer.com`.**

- **Phar**

  You can install the SDK through phar as follows:

  1. **Select the required version and download the Phar package on *GitHub*.**
  2. **Introduce the file in your code:**

```
require_once '/path/to/oss-sdk-php.phar';
```

Source code

You can install the SDK through source code as follows:

1. **Select the required version and download the zip package on *GitHub*.**
2. **Find the `autoload.php` file in the root directory of the unzipped package, and introduce the file in your code:**

```
require_once '/path/to/oss-sdk/autoload.php';
```

6.3 Quick start

This topic describes how to use OSS PHP SDK to perform routine operations such as bucket creation, object uploads, and object downloads.

Create a bucket

Run the following code to create a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket = "<yourBucketName>";

try {
```

```
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint
);
 $ossClient->createBucket($bucket);
} catch (OssException $e) {
 print $e->getMessage();
}
```

For more information about bucket naming rules, see naming conventions in *Basic concepts*. For more information about how to create a bucket, see *Manage a bucket*.

For more information about endpoints, see *Regions and endpoints*.

Upload objects

Run the following code to upload a file to OSS:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= " <yourBucketName>";
// Object name
$object = " <yourObjectName>";
$content = "Hi, OSS.";

try {
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint
);
 $ossClient->putObject($bucket, $object, $content);
} catch (OssException $e) {
 print $e->getMessage();
}
```

For more information, see *Upload objects*.

Download objects

Run the following code to download an object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
```

```
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
// Object name
$object = "<yourObjectName>";

try {
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint
);
 $content = $ossClient->getObject($bucket, $object);
    print("object content: " . $content);
} catch (OssException $e) {
 print $e->getMessage();
}
```

For more information, see *Download objects*.

List objects

Run the following code to list objects in a specified bucket. You can list a maximum of 100 objects by default.

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
```

```
// Bucket name
$bucket= "<yourBucketName>";

try {
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint
);

 $listObjectInfo = $ossClient->listObjects($bucket);
 $objectList = $listObjectInfo->getObjectList();
 if (! empty($objectList)) {
  foreach ($objectList as $objectInfo) {
  print($objectInfo->getKey() . "\t" . $objectInfo->getSize() . "\t" .
 $objectInfo->getLastModified() . "\n");
  }
 }
} catch (OssException $e) {
 print $e->getMessage();
}
```

For more information, see *List objects* **in** *Manage objects*.

Delete objects

Run the following code to delete a specified object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
// Object name
$object = "<yourObjectName>";

try {
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint
);
 $ossClient->deleteObject($bucket, $object);
} catch (OssException $e) {
 print $e->getMessage();
}
```

For more information, see *Delete objects* **in** *Manage objects*.

6.4 Initialization

OSSClient serves as the OSS Java client to manage OSS resources such as buckets and objects.

Create an OSSClient instance

To create an OSSClient instance, you need to specify an endpoint. For more information about endpoints, see *Regions and endpoints* and *Bind a custom domain name*.

- **Use an OSS domain to create an OSSClient instance**

  Run the following code to create an OSSClient instance with a domain assigned by OSS:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

try {
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);
} catch (OssException $e) {
 print $e->getMessage();
}
```

- **Use a custom domain (CNAME) to create an OSSClient instance**

  Run the following code to create an OSSClient instance with CNAME:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
```

```
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
$endpoint = "<yourEndpoint>";

try {
 // The true parameter indicates that the CNAME is enabled. CNAME
indicates a custom domain bound to a bucket.
 $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint, true);
} catch (OssException $e) {
 print $e->getMessage();
}
```

Note:

The ossClient.listBuckets method is not available when CNAME is used.

- **Use STS to create an OSSClient instance**

  Run the following code to create an OSSClient instance with Security Token Service (STS):

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$securityToken = "<yourSecurityToken>";

try {
 $ossClient = new OssClient(
        $accessKeyId, $accessKeySecret, $endpoint, false, $
securityToken);
} catch (OssException $e) {
 print $e->getMessage();
```

```
}
```

For more information, see *What is RAM and STS* **and** *Authorized access*.

- **Use a proxy server to create an OSSClient instance**

  Run the following code to create an OSSClient instance with a proxy server, which is supported by PHP 5.3 and later.

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// The address of the proxy server, for example, http://<username>:<
password>@<proxy ip>:<proxy port>
$requestProxy = "<yourRequestProxy>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint, false, $requestProxy);
} catch (OssException $e) {
    print $e->getMessage();
}
```

Configure network parameters

Run the following code to configure the network parameters of an OSSClient instance:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

try {
    $ossClient = new OssClient(
        $accessKeyId, $accessKeySecret, $endpoint);
 // Set the timeout time in seconds for data transmission at the
sockets layer. The default value is 5184000.
 $ossClient->setTimeout(3600);
 // Set the timeout time in seconds when connections are established.
The default value is 10.
 $ossClient->setConnectTimeout(10);
} catch (OssException $e) {
    print $e->getMessage();
}
```

6.5 Buckets

6.5.1 Bucket

A bucket serves as a container that stores objects. Objects belong to a bucket.

For the complete code of the following scenarios, see *GitHub*.

Create a bucket

Run the following code to create a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
try {
```

```
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);
    // Set the storage class to Infrequent Access for a bucket. The
default value is Standard.
    $options = array(
        OssClient::OSS_STORAGE => OssClient::OSS_STORAGE_IA
    );
    // Set the ACL to public read for a bucket. The default value is
Private.
    $ossClient->createBucket($bucket, OssClient::OSS_ACL_TYPE_PUBLIC_
READ, $options);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

For more information about bucket naming rules, see naming conventions in *Basic concepts*.

Determine whether a bucket exists

Run the following code to determine whether a specified bucket exists:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $res = $ossClient->doesBucketExist($bucket);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
if ($res === true) {
    print(__FUNCTION__ . ": OK" . "\n");
} else {
```

```
    print(__FUNCTION__ . ": FAILED" . "\n");
}
```

Lisk buckets

Run the following code to list buckets:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $bucketListInfo = $ossClient->listBuckets();
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
$bucketList = $bucketListInfo->getBucketList();
foreach($bucketList as $bucket) {
    print($bucket->getLocation() . "\t" . $bucket->getName() . "\t" .
 $bucket->getCreatedate() . "\n");
}
```

Configure an ACL for a bucket

The ACL of a bucket includes the following permissions.

Permission	Description	Value
Private	The bucket owner and the authorized users can read and write objects in the bucket. Other users cannot perform any operation on the objects.	OssClient::OSS_ACL_TYPE_PRIVATE

Permission	Description	Value
Public read	The bucket owner and the authorized users can read and write objects in the bucket. Other users can only read the objects in the bucket. Authorize this permission with caution.	OssClient::OSS_ACL_TYPE_PUBLIC_READ
Public read-write	All users can read and write objects in the bucket. Authorize this permission with caution.	OssClient::OSS_ACL_TYPE_PUBLIC_READ_WRITE

You can run the following code to configure an ACL for a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
// Set the ACL of the bucket to private.
$acl = OssClient::OSS_ACL_TYPE_PRIVATE;
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putBucketAcl($bucket, $acl);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

Obtain the ACL for a bucket

Run the following code to obtain the ACL for a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $res = $ossClient->getBucketAcl($bucket);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
print('acl: ' . $res);
```

Obtain the region of a bucket

Run the following code to obtain the region (or location) for a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $Regions = $ossClient->getBucketLocation($bucket);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
var_dump($Regions);
```

For more information about region, see region in the *Basic concepts*.

Obtain bucket information

Run the following code to obtain bucket information:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $Metas = $ossClient->getBucketMeta($bucket);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
```

```
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
var_dump($Metas);
```

Delete a bucket

Before a bucket is deleted, ensure that all objects in the bucket and fragments that are generated from multipart upload are deleted.

Note:

To delete the fragments that are generated from multipart upload, use *Bucket.ListMultipartUploads* to list all fragments, and then use *Bucket.AbortMultipartUpload* to delete the fragments.

Run the following code to delete a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->deleteBucket($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

6.5.2 Manage lifecycle rules

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**
- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**
- **Status: This lifecyle rule takes effect or not.**

For more information about lifecycle management, see *Manage object lifecycle*. For the complete code of lifecycle management, see *GitHub*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Model\LifecycleConfig;
use OSS\Model\LifecycleRule;
use OSS\Model\LifecycleAction;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

// Configure the rule ID and object name prefix.
$ruleId0 = "rule0";
$matchPrefix0 = "A0/";
$ruleId1 = "rule1";
$matchPrefix1 = "A1/";

$lifecycleConfig = new LifecycleConfig();
$actions = array();
// Set the expiration duration to 3 days after last modified date.
$actions[] = new LifecycleAction(OssClient::OSS_LIFECYCLE_EXPIRATION,
OssClient::OSS_LIFECYCLE_TIMING_DAYS, 3);
$lifecycleRule = new LifecycleRule($ruleId0, $matchPrefix0, "Enabled",
 $actions);
$lifecycleConfig->addRule($lifecycleRule);
$actions = array();
// Configure expiration date to delete objects that are created before
 the date.
$actions[] = new LifecycleAction(OssClient::OSS_LIFECYCLE_EXPIRATION,
OssClient::OSS_LIFECYCLE_TIMING_DATE, '2022-10-12T00:00:00.000Z');
$lifecycleRule = new LifecycleRule($ruleId1, $matchPrefix1, "Enabled",
 $actions);
$lifecycleConfig->addRule($lifecycleRule);
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putBucketLifecycle($bucket, $lifecycleConfig);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

View lifecycle rules

Run the following code to view lifecycle rules:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$lifecycleConfig = null;
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $lifecycleConfig = $ossClient->getBucketLifecycle($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
print($lifecycleConfig->serializeToXml() . "\n");
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->deleteBucketLifecycle($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

6.5.3 Hotlink protection

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49*. For the complete code of hotlink protection, see *GitHub*.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Model\RefererConfig;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$refererConfig = new RefererConfig();
// Allow empty referers.
$refererConfig->setAllowEmptyReferer(true);
// Add the referer field. The referer field allows question marks (?)
and asterisks (*) for wildcard use.
$refererConfig->addReferer("www.aliiyun.com");
$refererConfig->addReferer("www.aliiyuncs.com");
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);
```

```
    $ossClient->putBucketReferer($bucket, $refererConfig);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Obtain a referer whiltelist

Run the following code to obtain a referer whiltelist:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Model\RefererConfig;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$refererConfig = null;
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $refererConfig = $ossClient->getBucketReferer($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
print($refererConfig->serializeToXml() . "\n");
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
```

```
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Model\RefererConfig;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$refererConfig = new RefererConfig();
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    // You cannot clear a referer whitelist directly. To clear a
referer whitelist, you need to create the rule that allows an empty
referer field and replace the original rule with the new rule.
    $ossClient->putBucketReferer($bucket, $refererConfig);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.5.4 Set logging

You can enable access logs to record bucket access to log files, which are stored in a specified bucket.

The log file format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

For more information about access log files, see *Set access logging*. For the complete code of access logging, see *GitHub*.

Enable access logging

Run the following code to enable bucket access logging:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}
```

```
use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

$option = array();
// Configure the bucket that stores log files.
$targetBucket = "<yourBucketName>";
$targetPrefix = "access.log";
$bucket= "<yourBucketName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    // Enable access logging.
    $ossClient->putBucketLogging($bucket, $targetBucket, $targetPrefix
, $option);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

View access logging configurations

Run the following code to view the access logging configurations for a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
```

```
$loggingConfig = null;
$options = array();
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    // View access log configurations.
    $loggingConfig = $ossClient->getBucketLogging($bucket, $options);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
print($loggingConfig->serializeToXml() . "\n");
```

Disable access logging

Run the following code to disable access logging for a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    // Disable access logging.
    $ossClient->deleteBucketLogging($bucket);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.5.5 Static website hosting

You can set your bucket configuration to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

For more information about static website hosting, see *Static website hosting*.

For the complete code of the following scenarios, see *GitHub*.

Configure static website hosting

Run the following code to configure static website hosting:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Model\WebsiteConfig;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

// Configure static website hosting: The index paga is index.html and
the error page is error.html.
$websiteConfig = new WebsiteConfig("index.html", "error.html");
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putBucketWebsite($bucket, $websiteConfig);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

View static website hosting configurations

Run the following code to view static website hosting configurations:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$websiteConfig = null;
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $websiteConfig = $ossClient->getBucketWebsite($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
print($websiteConfig->serializeToXml() . "\n");
```

Delete static website hosting configurations

Run the following code to delete static website hosting configurations:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->deleteBucketWebsite($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.5.6 CORS

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* in OSS Developer Guide. For the complete code of CORS, see *GitHub*.

Configure CORS rules

Run the following code to configure CORS rules for the specified bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Model\CorsConfig;
use OSS\Model\CorsRule;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
```

```
$corsConfig = new CorsConfig();
$rule = new CorsRule();
// AllowedHeaders and ExposeHeaders do not allow wildcards.
$rule->addAllowedHeader("x-oss-header");
// AllowedOlowedMethods allows only one wildcard asterisk (*).
Wildcard asterisks (*) indicate that all sources of the cross-origin
requests and operations are allowed.
$rule->addAllowedOrigin("http://www.b.com");
$rule->addAllowedMethod("POST");
$rule->setMaxAgeSeconds(10);
// A maximum of 10 rules are allowed for each bucket.
$corsConfig->addRule($rule);

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    // The existing rules will be replaced.
    $ossClient->putBucketCors($bucket, $corsConfig);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Obtain CORS rules

Run the following code to obtain CORS rules:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$corsConfig = null;
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $corsConfig = $ossClient->getBucketCors($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
```

```
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
print($corsConfig->serializeToXml() . "\n");
```

Delete CORS rules

Run the following code to delete all CORS rules for a specified bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->deleteBucketCors($bucket);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.5.7 Bind a custom domain name

You can bind a custom domain name to your OSS bucket and by adding a CNAME record in the Domain Name System (DNS). After you bind a custom domain name to a bucket, you can access the bucket with your own domain name. For example, if your domain name is "my-domain.com" and all your images were previously located at `http://img.my-domain.com/x.jpg`, after you migrate the image data to OSS, you can still access the images with the original IP address by binding a custom domain name to the bucket.

For more information, see *Bind a custom domain name* in the OSS Developer Guide.

Add a CNAME record

Run the following code to add a CNAME record for a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    Quire_once _ DIR __. '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

// Configure the custom domain name.
$myDomain = '<yourDomainName>';
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

 // Add a CNAME record.
    $ossClient->addBucketCname($bucket, $myDomain);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

View CNAME records

Run the following code to obtain CNAME records added to the bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
```

```
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

 // View CNAME records.
    $cnameConfig = $ossClient->getBucketCname($bucket);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
var_dump($cnameConfig);
print(__FUNCTION__ . ": OK" . "\n");
```

Delete a CNAME record

Run the following code to delete a CNAME record added to a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

// Configure the custom domain name.
$myDomain = '<yourDomainName>';
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

 // Delete a CNAME record.
    $ossClient->deleteBucketCname($bucket, $myDomain);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
```

```
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.6 Upload objects

6.6.1 Overview

In OSS, an object is the basic unit for data operations. OSS PHP SDK provides the following file upload methods:

- *Simple upload*: **supports the upload of a file up to 5 GB in size.**
- *Append upload*: **supports the upload of a file up to 5 GB in size.**
- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files**

Note:

For more information about the usage scenarios of each upload method, see "Upload files" in OSS Developer Guide. After you complete the file upload, you can perform *upload callback*.

6.6.2 Simple upload

Simple upload applies to two scenarios: upload a string and upload a local file.

Upload a string

You can run the following code to upload a string to OSS:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
```

```
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
// Object name
$object = "<yourObjectName>";
// Object content
$content = "Hello OSS";
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putObject($bucket, $object, $content);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Upload a local file

You can run the following code to upload a local file to OSS:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Bucket name
$bucket= "<yourBucketName>";
// Object name
$object = "<yourObjectName>";
// <yourLocalFile> consists of the local file path and the file name (
including the extension), such as /users/local/myfile.txt
$filePath = "<yourLocalFile>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->uploadFile($bucket, $object, $filePath);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

6.6.3 Append upload

You can append a string or a file to an OSS object.

You cannot perform copyObject for appended objects.

Append a string

Run the following code to append a string to an OSS object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Specify the bucket name.
$bucket= "<yourBucketName>";
// Specify the object name.
$object = "<yourObjectName>";
// Specify the content of the string.
$content_array = array('Hello OSS', 'Hi OSS', 'OSS OK');
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $position = $ossClient->appendObject($bucket, $object, $content_ar
ray[0], 0);
    $position = $ossClient->appendObject($bucket, $object, $content_ar
ray[1], $position);
    $position = $ossClient->appendObject($bucket, $object, $content_ar
ray[2], $position);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

Append a file

Run the following code to append a local file to an OSS object:

```
 <? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
// Specify the bucket name.
$bucket= "<yourBucketName>";
// Specify the object name.
String objectName = "<yourObjectName>";
// Specify the first local file.
$filePath = "<yourLocalFile1>";
// Specify the second local file
$filePath1 = "<yourLocalFile2>";
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $position = $ossClient->appendFile($bucket, $object, $filePath, 0
);
    $position = $ossClient->appendFile($bucket, $object, $filePath1, $
position);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.6.4 Multipart upload

To enable multipart upload, perform the following steps:

1. **Initiate a multipart upload event.**

   You can call $ossClient->initiateMultipartUpload to return the globally unique uploadId created in OSS.

2. **Upload parts.**

   You can call $ossClient->uploadPart to upload part data.

   > **Notice:**
   >
   > - **For parts with a same uploadId, parts are sequenced by their part numbers. If you have uploaded a part and use the same part number to upload another part, the later part will replace the former part.**
   > - **Except for the last part, the minimum size of other size is 100 KB. The size of the last part is not limited.**

3. **Complete multipart upload.**

   Call $ossClient->completeMultipartUpload to combine these parts into a complete object.

For the complete code of multipart upload, see *GitHub*.

The following code is used as a complete example that describes the process of multipart upload:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Core\OssUtil;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$uploadFile = "<yourLocalFile>";

/**
 *  Step 1: Initiate a multipart upload event and obtain the uploadId.
 */
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);
```

```
    // An uploadId is returned. It is the unique identifier for a
part upload event. You can initiate related operations (such as part
upload cancelation and query) based on the uploadId.
    $uploadId = $ossClient->initiateMultipartUpload($bucket, $object);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": initiateMultipartUpload FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": initiateMultipartUpload OK" . "\n");
/*
 * Step 2: Upload parts.
 */
$partSize = 10 * 1024 * 1024;
$uploadFileSize = filesize($uploadFile);
$pieces = $ossClient->generateMultiuploadParts($uploadFileSize, $
partSize);
$responseUploadPart = array();
$uploadPosition = 0;
$isCheckMd5 = true;
foreach ($pieces as $i => $piece) {
    $fromPos = $uploadPosition + (integer)$piece[$ossClient::
OSS_SEEK_TO];
    $toPos = (integer)$piece[$ossClient::OSS_LENGTH] + $fromPos - 1;
    $upOptions = array(
        $ossClient::OSS_FILE_UPLOAD => $uploadFile,
        $ossClient::OSS_PART_NUM => ($i + 1),
        $ossClient::OSS_SEEK_TO => $fromPos,
        $ossClient::OSS_LENGTH => $toPos - $fromPos + 1,
        $ossClient::OSS_CHECK_MD5 => $isCheckMd5,
    );
    // MD5 check.
    if ($isCheckMd5) {
        $contentMd5 = OssUtil::getMd5SumForFile($uploadFile, $fromPos,
 $toPos);
        $upOptions[$ossClient::OSS_CONTENT_MD5] = $contentMd5;
    }
    try {
        // Upload parts.
        $responseUploadPart[] = $ossClient->uploadPart($bucket, $
object, $uploadId, $upOptions);
    } catch(OssException $e) {
        printf(__FUNCTION__ . ": initiateMultipartUpload, uploadPart
 - part#{$i} FAILED\n");
        printf($e->getMessage() . "\n");
        return;
    }
    printf(__FUNCTION__ . ": initiateMultipartUpload, uploadPart -
part#{$i} OK\n");
}
// $uploadParts is an array composed by the ETags and part number (
PartNumber) of each part.
$uploadParts = array();
foreach ($responseUploadPart as $i => $eTag) {
    $uploadParts[] = array(
        'PartNumber' => ($i + 1),
        'ETag' => $eTag,
    );
}
/**
 * Step 3: Complete multipart upload.
 */
try {
```

```
    // You must provide all valid $uploadParts when you perform this
 operation. OSS verifies the validity of all parts one by one after
it receives $uploadParts. After part verification is successful, OSS
combines these parts into a complete object.
    $ossClient->completeMultipartUpload($bucket, $object, $uploadId, $
uploadParts);
}  catch(OssException $e) {
    printf(__FUNCTION__ . ": completeMultipartUpload FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
printf(__FUNCTION__ . ": completeMultipartUpload OK\n");
```

In the example above, $options includes the following parameters:

Parameter	Description
$ossClient::OSS_FILE_UPLOAD	**File upload**
OssClient::OSS_PART_NUM	**Part number**
OssClient::OSS_SEEK_TO	**Specifies a start location.**
OssClient::OSS_LENGTH	**Object length**
OssClient::OSS_PART_SIZE	**Part size**
OssClient::OSS_CHECK_MD5	**Determines whether MD5 check is enabled. The MD5 check is enabled if the value is true.**

Cancel a multipart upload event

You can call $ossClient->abortMultipartUpload to cancel a multipart upload event . If you cancel a multipart upload event, you are not allowed to perform any other operations with this uploadId anymore. The uploaded parts will be deleted.

Run the following code to cancel a multipart upload event:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
```

```
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$upload_id = "<yourUploadId>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->abortMultipartUpload($bucket, $object, $upload_id);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Upload of a local file using multipart upload

Run the following code to upload a local file using multipart upload:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$file = "<yourLocalFile>";

$options = array(
    OssClient::OSS_CHECK_MD5 => true,
    OssClient::OSS_PART_SIZE => 1,
);
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->multiuploadFile($bucket, $object, $file, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
```

```
}
print(__FUNCTION__ . ":  OK" . "\n");
```

Upload a directory using multipart upload

Run the following code to upload a local directory (including all files in it) using multipart upload:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$localDirectory = ".";
$prefix = "samples/codes";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->uploadDir($bucket, $prefix, $localDirectory);
}  catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ":  OK" . "\n");
```

List uploaded parts

Run the following code to list uploaded parts:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
```

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$uploadId = "<yourUploadId>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $listPartsInfo = $ossClient->listParts($bucket, $object, $uploadId
);
    foreach ($listPartsInfo->getListPart() as $partInfo) {
        print($partInfo->getPartNumber() . "\t" . $partInfo->getSize
() . "\t" . $partInfo->getETag() . "\t" . $partInfo->getLastModified
() . "\n");
    }
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

List all multipart upload events

Run the following code to list all multipart upload events:

```
<? php
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
```

```
$options = array(
    'delimiter' => '/',
    'max-uploads' => 100,
    'key-marker' => '',
    'prefix' => '',
    'upload-id-marker' => ''
);
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $listMultipartUploadInfo = $ossClient->listMultipartUploads($
bucket, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": listMultipartUploads FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
printf(__FUNCTION__ . ": listMultipartUploads OK\n");
$listUploadInfo = $listMultipartUploadInfo->getUploads();
var_dump($listUploadInfo);
```

Parameters of $options are described as follows:

Parameter	Description
delimiter	**Specifies a delimiter of a forward slash (/) used to group object names. The object between the specified prefix and the first occurrence of a delimiter of a forward slash (/) is commonPrefixes.**
key-marker	**Lists all part upload events with the object whose names start with a letter that comes after the key-marker value in the alphabetical order. You can use this parameter with the upload-id-marker parameter to specify the initial position for the specified returned result.**
max-uploads	**Specifies the maximum number of part upload events. The maximum value (also default value) you can set is 1,000.**
prefix	**Specifies the prefix that must be included in the returned object name. Note that if you use a prefix for query, the returned object name will contain the prefix.**

Parameter	Description
upload-id-marker	You can use this parameter with the key-marker parameter to specify the initial position for the specified returned result. If you do not configure keyMarker, the uploadIdMarker parameter is invalid. If you configure key-marker, the query result contains: · All objects whose names start with a letter that comes after the keyMarker value. · All objects whose names start with a letter that is the same as the key-marker value in the alphabetical order and the value of uploadId greater than that of upload-id-marker.

6.6.5 Upload callback

Configure callback for simple upload

Run the following code to configure callback for simple upload.

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint);

// Configure callback for uploads.
// callbackUrl is the IP address of the callback server, such as http
://oss-demo.aliyuncs.com:23450 or http://127.0.0.1:9090.
```

```
// callbackHost is the value of the Host field carried in the callback
 request, such as oss-cn-hangzhou.aliyuncs.com.
$url =
    '{
        "callbackUrl":"<yourCallbackServerUrl>",
        "callbackHost":"<yourCallbackServerHost>",
        "callbackBody":"bucket=${bucket}&object=${object}&etag=${
etag}&size=${size}&mimeType=${mimeType}&imageInfo.height=${imageInfo
.height}&imageInfo.width=${imageInfo.width}&imageInfo.format=${
imageInfo.format}&my_var1=${x:var1}&my_var2=${x:var2}",
        "callbackBodyType":"application/x-www-form-urlencoded"
    }';

// Configure the custom parameters used to initiate a callback request
. Each parameter consists of a key and a value. The key must start
with x:.
$var =
    '{
        "x:var1":"value1",
        "x:var2":"value2"
    }';
$options = array(OssClient::OSS_CALLBACK => $url,
    OssClient::OSS_CALLBACK_VAR => $var
);
$result = $ossClient->putObject($bucket, $object, file_get_contents(
__FILE__), $options);
print_r($result['body']);
print_r($result['info']['http_code']);
```

Configure callback for multipart upload

Run the following code to configure callback for multipart upload:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}
use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Core\OssUtil;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China (Hangzhou). Specify the actual
endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$uploadFile = "<yourLocalFile>";

/**
 * Step 1: Initiate a multipart upload event and get the uploadId.
 */
```

```
$ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint);
// An uploadId is returned. It is the unique identifier for a
multipart upload event. You can initiate related operations (such as
cancel or query a multipart upload event) based on the uploadId.
$uploadId = $ossClient->initiateMultipartUpload($bucket, $object);

print(__FUNCTION__ . ": initiateMultipartUpload OK" . "\n");
/*
 /* Step 2: Upload parts.
 */
$partSize = 10 * 1024 * 1024;
$uploadFileSize = filesize($uploadFile);
$pieces = $ossClient->generateMultiuploadParts($uploadFileSize, $
partSize);
$responseUploadPart = array();
$uploadPosition = 0;
$isCheckMd5 = true;
foreach ($pieces as $i => $piece) {
    $fromPos = $uploadPosition + (integer)$piece[$ossClient::
OSS_SEEK_TO];
    $toPos = (integer)$piece[$ossClient::OSS_LENGTH] + $fromPos - 1;
    $upOptions = array(
        $ossClient::OSS_FILE_UPLOAD => $uploadFile,
        $ossClient::OSS_PART_NUM => ($i + 1),
        $ossClient::OSS_SEEK_TO => $fromPos,
        $ossClient::OSS_LENGTH => $toPos - $fromPos + 1,
        $ossClient::OSS_CHECK_MD5 => $isCheckMd5,
    );
    // Perform the MD5 verification.
    if ($isCheckMd5) {
        $contentMd5 = OssUtil::getMd5SumForFile($uploadFile, $fromPos,
 $toPos);
        $upOptions[$ossClient::OSS_CONTENT_MD5] = $contentMd5;
    }
    try {
        // Upload the parts.
        $responseUploadPart[] = $ossClient->uploadPart($bucket, $
object, $uploadId, $upOptions);
    } catch(OssException $e) {
        printf(__FUNCTION__ . ": initiateMultipartUpload, uploadPart
 - part#{$i} FAILED\n");
        printf($e->getMessage() . "\n");
        return;
    }
    printf(__FUNCTION__ . ": initiateMultipartUpload, uploadPart -
part#{$i} OK\n");
}
// $uploadParts is an array that consists of the ETags and the
PartNumbers of each part.
$uploadParts = array();
foreach ($responseUploadPart as $i => $eTag) {
    $uploadParts[] = array(
        'PartNumber' => ($i + 1),
        'ETag' => $eTag,
    );
}

/**
 * Step 3: Complete the upload.
 */
// The callbackUrl is the IP address of the callback server, such as
http://oss-demo.aliyuncs.com:23450 or http://127.0.0.1:9090.
// callbackHost is the value of the Host field carried in the callback
 request, such as oss-cn-hangzhou.aliyuncs.com.
```

```
$json =
    '{
        "callbackUrl":"<yourCallbackServerUrl>",
        "callbackHost":"<yourCallbackServerHost>",
        "callbackBody":"{\"mimeType\":${mimeType},\"size\":${size},\"x
:var1\":${x:var1},\"x:var2\":${x:var2}}",
        "callbackBodyType":"application/json"
    }';

// Configure the custom parameters used to initiate a callback request
. Each parameter consists of a key and a value. The key must start
with x:.
$var =
    '{
        "x:var1":"value1",
        "x:var2":"value2"
    }';
$options = array(OssClient::OSS_CALLBACK => $json,
    OssClient::OSS_CALLBACK_VAR => $var);
// To perform the operation, all valid $uploadParts must be provided
. OSS verifies the validity of all parts one by one after it receives
 the $uploadParts. After all the data parts are verified, the OSS
combines these parts into a complete object.
$ossClient->completeMultipartUpload($bucket, $object, $uploadId, $
uploadParts, $options);

printf(__FUNCTION__ . ": completeMultipartUpload OK\n");
```

6.7 Download objects

6.7.1 Overview

This topic is the overview of object download in OSS PHP SDK.

For the complete code of object download, see *GitHub*.

OSS PHP SDK provides the following download methods:

- *Download to your local file*
- *Download to local memory*
- *Range download*
- *Conditional download*

6.7.2 Download to your local file

You can run the following code to download the specified OSS object to a local file:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}
```

```
use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
//This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$localfile = "<yourLocalFile>";
$options = array(
        OssClient::OSS_FILE_DOWNLOAD => $localfile
    );

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->getObject($bucket, $object, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK, please check localfile: 'upload-test-
object-name.txt'" . "\n");
```

6.7.3 Download to your local memory

Run the following code to download a specified object to local memory and print the content of the object.

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
```

```
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $content = $ossClient->getObject($bucket, $object);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print($content);
print(__FUNCTION__ . ": OK" . "\n");
```

6.7.4 Range download

If you only need part of the data in an object, you can use range downloads to download specified content.

Run the following code to download the content of an object in a range of [0, 4] to local memories:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

//It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
//This example uses the endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
// Obtain data between the starting byte value of 0 and the maximum
 byte value of 4 inclusive (that is, a total of 5 bytes). If the
specified value range is invalid, for example, the specified range
includes a negative number, or the specified value is larger than the
object size, the entire object is downloaded.
$options = array(OssClient::OSS_RANGE => '0-4');
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $content = $ossClient->getObject($bucket, $object, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print ($content);
print(__FUNCTION__ . ": OK" . "\n");
```

6.7.5 Conditional download

You can configure one or more conditions for downloads. If the specified conditions are met, the object can be downloaded. If the specified conditions are not met, an error code is returned and the object cannot not downloaded. You can configure the following conditions:

Parameter	Description	Configuration method
If-Modified-Since	If the specified time is earlier than the time the object is modified, the object can be downloaded. Otherwise, an error (304 Not modified) is returned.	OssClient::OSS_IF_MODIFIED_SINCE
If-Unmodified-Since	If the specified time is later than or equal to the time the object is modified, the object can be downloaded. Otherwise, an error (412 Precondition failed) is returned.	OssClient::OSS_IF_UNMODIFIED_SINCE
If-Match	If the specified ETag matches that of an object, the object can be downloaded. Otherwise, an error (412 Precondition failed) is returned.	OssClient::OSS_IF_MATCH
If-None-Match	If the specified ETag does not match that of an object, the object can be downloaded. Otherwise, an error (304 Not modified) is returned.	OssClient::OSS_IF_NONE_MATCH

Note:

- **Both If-Modified-Since and If-Unmodified-Since can exist at the same time as object download conditions. Both If-Match and If-None-Match can exist at the same time as object download conditions.**
- **$ossClient->getObjectMeta can be used to obtain the ETag.**

You can download an OSS object to local memories or a local file in conditional downloads. Run the following code for conditional download:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

try{
    $options = array(
        OssClient::OSS_HEADERS => array(
            OssClient::OSS_IF_MODIFIED_SINCE => "Fri, 13 Nov 2015 14:
47:53 GMT"),
    );

    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $content = $ossClient->getObject($bucket, $object, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print ($content);
print(__FUNCTION__ . ": OK" . "\n");
```

6.8 Manage objects

6.8.1 Overview

You can perform the following operations on objects in a bucket with APIs:

- *Determine whether an object exists*
- *Manage ACL for an object*
- *Manage Object Meta*
- *List objects*
- *Delete an object*
- *Copy an object*
- *Restore an archive object*
- *Manage a symbolic link*
- *Enable MD5 verification*

6.8.2 Determine whether a specified object exists

Use the following code to determine whether a specified object exists:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $exist = $ossClient->doesObjectExist($bucket, $object);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

```
var_dump($exist);
```

6.8.3 Manage ACL for an object

The following table describes the permissions included in the Access Control List (ACL) for an object.

Permission	Description	Value
default	The ACL of an object is the same with that of its bucket.	default
Private	Only the object owner and authorized users can read and write the object.	private
Public read	Only the object owner and authorized users can read and write the object. Other users can only read the object. Authorize this permission with caution.	public-read
Public read-write	All users can read and write the object. Authorize this permission with caution.	public-read-write

The ACL of objects take precedence over that of buckets. For example, if the ACL of a bucket is private, while the object ACL is public read-write, all users can read and write the object. If an object is not configured with an ACL, its ACL is the same as that of its bucket by default.

Configure an ACL for an object

You can run the following code to configure an ACL for a specified object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
```

```
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
// Set the ACL for the object to public read, which is inherited from
the bucket by default.
$acl = "public-read";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putObjectAcl($bucket, $object, $acl);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Obtain the ACL for an object

You can run the following code to obtain the ACL for a specified object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $objectAcl = $ossClient->getObjectAcl($bucket, $object);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
var_dump($objectAcl);
```

6.8.4 Manage Object Meta

For more information about Object Meta, see *Object Meta* in OSS Developer Guide.

Configure Object Meta

Run the following code to configure Object Meta:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$content = file_get_contents(__FILE__);
$options = array(
    OssClient::OSS_HEADERS => array(
        'Expires' => '2012-10-01 08:00:00',
        'Content-Disposition' => 'attachment; filename="xxxxxx"',
        'x-oss-meta-self-define-title' => 'user define meta info',
    ));
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putObject($bucket, $object, $content, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Modify Object Meta

Run the following code to modify Object Meta:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
```

```
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$fromBucket = "<yourFromBucketName>";
$fromObject = "<yourFromObjectName>";
$toBucket = "<yourToBucketName>";
$toObject = "<yourToObjectName>";
$copyOptions = array(
    OssClient::OSS_HEADERS => array(
        'Expires' => '2018-10-01 08:00:00',
        'Content-Disposition' => 'attachment; filename="xxxxxx"',
        'x-oss-meta-location' => 'location',
    ),
);
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->copyObject($fromBucket, $fromObject, $toBucket, $
toObject, $copyOptions);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Obtain Object Meta

Run the following code to obtain Object Meta:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
```

```
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ Objectmeta = $ ossclient-> getobjectmeta ($ bucket, $ object );
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
if (isset($objectMeta[strtolower('Content-Disposition')]) &&
    'attachment; filename="xxxxxx"' === $objectMeta[strtolower('
Content-Disposition')]
) {
    print(__FUNCTION__ . ": ObjectMeta checked OK" . "\n");
} else {
    print(__FUNCTION__ . ": ObjectMeta checked FAILED" . "\n");
}
```

6.8.5 List objects

This topic describes how to list the objects stored in a bucket.

List all objects

The following code provides an example on how to list all objects in a bucket:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you log on to the
 OSS console as a RAM user to call API operations or perform routine
 operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint);

$nextMarker = '';
```

```
while (true) {
    try {
        $options = array(
            'delimiter' => '',
            'marker' => $nextMarker,
        );
        $listObjectInfo = $ossClient->listObjects($bucket, $options);
    } catch (OssException $e) {
        printf(__FUNCTION__ . ": FAILED\n");
        printf($e->getMessage() . "\n");
        return;
    }
    // Obtain nextMarker to list the remaining objects, starting with
 the object next to the last object that is previously obtained by
calling the ListObjects operation.
    $nextMarker = $listObjectInfo->getNextMarker();
    $listObject = $listObjectInfo->getObjectList();
    $listPrefix = $listObjectInfo->getPrefixList();

    if (! empty($listObject)) {
        print("objectList:\n");
        foreach ($listObject as $objectInfo) {
            print($objectInfo->getKey() . "\n");
        }
    }
    if (! empty($listPrefix)) {
        print("prefixList: \n");
        foreach ($listPrefix as $prefixInfo) {
            print($prefixInfo->getPrefix() . "\n");
        }
    }
    if ($listObjectInfo->getIsTruncated() ! == "true") {
       break;
    }
}
```

Note:

In the preceding sample code, $options is optional and can be used to list all objects in the entire bucket or in a directory.

List desired objects

The following code provides an example on how to list desired objects:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you log on to the
```

```
 OSS console as a RAM user to call API operations or perform routine
 operations and maintenance. To create a RAM user, log on to https://
ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint);

$prefix = 'dir/';
$delimiter = '/';
$nextMarker = '';
$maxkeys = 10;
$options = array(
    'delimiter' => $delimiter,
    'prefix' => $prefix,
    'max-keys' => $maxkeys,
    'marker' => $nextMarker,
);
try {
    $listObjectInfo = $ossClient->listObjects($bucket, $options);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
$objectList = $listObjectInfo->getObjectList(); // object list
$prefixList = $listObjectInfo->getPrefixList(); // directory list
if (! empty($objectList)) {
    print("objectList:\n");
    foreach ($objectList as $objectInfo) {
        print($objectInfo->getKey() . "\n");
    }
}
if (! empty($prefixList)) {
    print("prefixList: \n");
    foreach ($prefixList as $prefixInfo) {
        print($prefixInfo->getPrefix() . "\n");
    }
}
```

The following table describes the parameters contained in $options involved in the preceding example.

Parameter	Description	Required
delimiter	Specifies a character used to group object names. CommonPrefixes is a set of objects whose names share a prefix and end with a forward slash (/) delimiter.	No

Parameter	Description	Required
prefix	Specifies the prefix that must be included in the names of the returned objects.	No
max-keys	Specifies the maximum number of objects that can be listed. Default value: 100. Maximum value: 1000.	No
marker	Specifies the name of the object after which the list begins.	No

6.8.6 Delete objects

Delete an object

Run the following code to delete a single object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->deleteObject($bucket, $object);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
```

```
print(__FUNCTION__ . ": OK" . "\n");
```

Delete multiple objects

Run the following code to delete multiple objects simultaneously:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";

$objects = array();
$objects[] = "<yourObjectName1>";
$objects[] = "<yourObjectName2>";
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->deleteObjects($bucket, $objects);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.8.7 Copy objects

Simple copy

Run the following code to copy objects smaller than 1 GB:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
```

```
// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$from_bucket = "<yourFromBucketName>";
$from_object = "<yourFromObjectName>";
$to_bucket = $bucket;
$to_object = $from_object . '.copy';

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->copyObject($from_bucket, $from_object, $to_bucket, $
to_object);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Copy a large-sized object

To copy an object larger than 1 GB, you need to use multipart copy. To use multipart copy, perform the following steps:

1. **Use $ossClient->initiateMultipartUpload to initiate an UploadPartCopy task.**
2. **Use $ossClient->uploadPartCopy to start a multipart copy. Aside from the last part, all parts must be larger than 100 KB.**
3. **Submit the multipart copy task with $ossClient->completeMultipartUpload.**

Run the following code for multipart copy:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
```

```
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

$src_bucket = "<yourSourceBucketName>";
$src_object = "<yourSourceObjectName>";
$dst_bucket = "<yourDestinationBucketName>";
$dst_object = "<yourDestinationObjectName>";

try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    // Initialize a multipart copy.
    $upload_id = $ossClient->initiateMultipartUpload($dst_bucket, $
dst_object);

    $copyId = 1;
    // Copy the parts one-by-one.
    $eTag = $ossClient->uploadPartCopy( $src_bucket, $src_object, $
dst_bucket, $dst_object,$copyId, $upload_id);
    $upload_parts[] = array(
        'PartNumber' => $copyId,
        'ETag' => $eTag,
    );

    // Complete the multipart copy.
    $result = $ossClient->completeMultipartUpload($dst_bucket, $
dst_object, $upload_id, $upload_parts);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

6.8.8 Restore an archive object

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

1. **An archive object is in the frozen state.**
2. **After you submit it for restoration, the server restores the object. The object is in the restoring state.**
3. **You can read the object after it is restored. The restored state of the object lasts one day by default. You can prolong this period to a maximum of seven days. Once this period expires, the object returns to the frozen state.**

Run the following code to restore an object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
```

```
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->restoreObject($bucket, $object);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

For more information about the archive storage classes, see *Introduction to storage classes*.

6.8.9 Manage a symbolic link

Create a symbolic link

A symbolic link is a special object that maps to an object similar to a shortcut used in Windows. You can configure user-defined Object Meta for symbolic links.

Run the following code to create a symbolic link:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
```

```
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$symlink = "<yourSymlink>";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->putSymlink($bucket, $symlink, $object);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

Obtain object content for a symbolic link

To obtain a symbolic link, you must have the read permission on it. Run the following code to obtain the object content for a symbolic link:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
 account because the account has permissions on all APIs in OSS. We
 recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$symlink = "<yourSymlink>";
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $Symlinks = $ossClient->getSymlink($bucket, $symlink);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

```
var_dump($Symlinks);
```

6.8.10 Enable MD5 verification

MD5 verification is performed to ensure data integrity during transmission. MD5 verification brings performance loss. MD5 verification is disabled by default during file uploads.

Run the following code to enable MD5 verification during file uploads:

```
<? php
if (is_file(__DIR__ . '/../Autopoload. php ')){
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
routine operations and maintenance. To create a RAM account, log on to
 https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

$options = array(OssClient::OSS_CHECK_MD5 => true);
try{
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

    $ossClient->uploadFile($bucket, $object, __FILE__, $options);
} catch(OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": OK" . "\n");
```

The putObject, uploadFile, appendObject, appendFile and multiuploadFile methods support MD5 verification.

6.9 Authorized access

This topic describes how to authorize access to OSS.

Use STS for temporary access authorization

You can use Security Token Service (STS) to authorize temporary access to OSS. STS is a web service that provides temporary access tokens for cloud computing users. You can use STS to grant a third-party application or a RAM user (whose user ID is managed by you) an access credential with a customized validity period and permissions. For more information about STS, see *What is STS?*

STS has the following benefits:

- **You only need to generate an access token and send the access token to a third-party application, rather than exposing your long-term key (AccessKey) to the third-party application. You can customize the access permissions and validity period of this token.**
- **The access token automatically expires after the validity period.**

For more information about the process of accessing OSS with STS, see *#unique_97* **in OSS Developer Guide.**

Use the following code to upload an object by using STS for temporary access authorization:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;

// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram.
console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$securityToken = "<yourSecurityToken>";
```

```
$content = "Hi, OSS.";

try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint, false, $securityToken);

    $ossClient->putObject($bucket, $object, $content);
} catch (OssException $e) {
    print $e->getMessage();
}
```

Use a signed URL for temporary access authorization

You can provide the signed URL that is generated to a visitor to grant the visitor temporary access. When generating a signed URL, you can specify the validity period of the URL to restrict the period of access from visitors.

For the complete code of authorized access, see *GitHub*.

- **Generate a signed URL to download an object**

  Use the following code to generate a signed URL to download an object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Http\RequestCore;
use OSS\Http\ResponseCore;

// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram
.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$securityToken = "<yourSecurityToken>";

// Set the expiration time of a URL to 3600 seconds.
$timeout = 3600;
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint, false, $securityToken);

    // Generate the signed URL for GetObject.
```

```
    $signedUrl = $ossClient->signUrl($bucket, $object, $timeout);
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": signedUrl: " . $signedUrl . "\n");

// You can run code to access the signed URL, or access the signed
URL in a browser.
$request = new RequestCore($signedUrl);
// Set GET as the default method to access the singed URL.
$request->set_method('GET');
$request->add_header('Content-Type', '');
$request->send_request();
$res = new ResponseCore($request->get_response_header(), $request->
get_response_body(), $request->get_response_code());
if ($res->isOK()) {
    print(__FUNCTION__ . ": OK" . "\n");
} else {
    print(__FUNCTION__ . ": FAILED" . "\n");
};
```

- **Generate a signed URL to upload an object**

  Use the following code to generate a signed URL to upload an object:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;
use OSS\Core\OssException;
use OSS\Http\RequestCore;
use OSS\Http\ResponseCore;

// Security risks may arise if you use the AccessKey pair of an
Alibaba Cloud account to log on to OSS, because the account has
permissions on all API operations. We recommend that you use a RAM
 user account to call API operations or perform routine operations
and maintenance. To create a RAM user account, log on to https://ram
.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses the endpoint of the China (Hangzhou) region.
Specify the actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";
$securityToken = "<yourSecurityToken>";

$timeout = 3600;
try {
    $ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint, false, $securityToken);

    // Generate the signed URL for PutObject.
```

```
    $signedUrl = $ossClient->signUrl($bucket, $object, $timeout, "
PUT");
} catch (OssException $e) {
    printf(__FUNCTION__ . ": FAILED\n");
    printf($e->getMessage() . "\n");
    return;
}
print(__FUNCTION__ . ": signedUrl: " . $signedUrl . "\n");

$content = "Hello OSS.";
$request = new RequestCore($signedUrl);
// Set PUT as the default method to access the signed URL.
$request->set_method('PUT');
$request->add_header('Content-Type', '');
$request->add_header('Content-Length', strlen($content));
$request->set_body($content);
$request->send_request();
$res = new ResponseCore($request->get_response_header(),
    $request->get_response_body(), $request->get_response_code());
if ($res->isOK()) {
    print(__FUNCTION__ . ": OK" . "\n");
} else {
    print(__FUNCTION__ . ": FAILED" . "\n");
};
```

6.10 Image processing

Image Processing (IMG) is a massive, secure, cost-effective and highly reliable image processing service. After source images are uploaded to OSS, you can process images on any Internet device at any anytime, from anywhere through simple RESTful APIs.

For more information about IMG, see *Image Processing*. For the complete code of image processing, see *GitHub*.

Basic features

OSS offers the following IMG features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**

- *Image processing*
- *Image processing access rules*: **calls multiple IMG functions.**

Usage

IMG uses standard HTTP GET. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is private read and write, only authorized users are allowed for access.

- **Anonymous access**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Specifies the name of a bucket.
endpoint	Specifies endpoint used to access a region.
object	Specifies the name of an image object.
image	Specifies the reserved identifier for IMG.
action	Specifies the operations on an image, such as scaling, cropping, and rotating .

Parameter	Description
param	Specifies the parameter that indicates the operation on an image.

- **Basic operations**

  For example, scale an image to a width of 100 PX. Adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=image/resize,w_100
```

- **Customized image styles**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-process=style/<yourStyleName>
```

  - **style: specifies a reserved identifier of a customized image style.**
  - **yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.**

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=style/oss-pic-style-w-100
```

- **Cascade operations**

  Cascade operations allow you to perform multiple operations on an image sequentially.

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<yourParamValue2>/...
```

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=image/resize,w_100/rotate,90
```

- **HTTPS access**

IMG supports access through HTTPS. Example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  Authorized access allows you to customize an image style, HTTPS access, and cascade operations.

  Run the following code to generate a signed URL for IMG:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba Cloud
account because the account has permissions on all the APIs in OSS.
We recommend that you log on as a RAM user to access APIs or perform
 routine operations and maintenance. To create a RAM account, log on
 to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket= "<yourBucketName>";
$object = "<yourObjectName>";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $endpoint
);

// Generate a signed URL that is valid for 3600 seconds and can be
directly accessed by browsers.
$timeout = 3600;
$options = array(
    OssClient::OSS_PROCESS => "image/resize,m_lfit,h_100,w_100" );
$signedUrl = $ossClient->signUrl($bucket, $object, $timeout, "GET",
 $options);
```

```
print("rtmp url: \n" . $signedUrl);
```

- **Access with SDK**

  You can use SDK to access and process an image object.

  SDK allows you to customize image styles, HTTPS access, and cascade operations

  .

  - **Basic operations**

    Run the following code to perform basic operations on an image:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba
Cloud account because the account has permissions on all APIs in
 OSS. We recommend that you log on as a RAM user to access APIs
 or perform routine operations and maintenance. To create a RAM
account, log on to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket = "<yourBucketName>";
$object = "<yourObjectName>";
$download_file = "download.jpg";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

// Upload a sample image.
$ossClient->uploadFile($bucket, $object, "example.jpg");

// Scale the image.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/resize,m_fixed,h_100,w_100
" );
$ossClient->getObject($bucket, $object, $options);

// Crop the image.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/crop,w_100,h_100,x_100,y_100,
r_1" );
$ossClient->getObject($bucket, $object, $options);

// Rotate the image.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/rotate,90" );
```

```
$ossClient->getObject($bucket, $object, $options);

// Sharpen the image.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/sharpen,100" );
$ossClient->getObject($bucket, $object, $options);

// Add watermarks.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/watermark,text_SGVsbG8g5Zu-
54mH5pyN5YqhIQ" );
$ossClient->getObject($bucket, $object, $options);

// Convert the image format.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/format,png" );
$ossClient->getObject($bucket, $object, $options);

// Obtain image information.
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => "image/info" );
$ossClient->getObject($bucket, $object, $options);

// Delete the sample image.
$ossClient->deleteObject($bucket, $object);
```

- **Customize an image style**

  Run the following code to customize an image style:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba
Cloud account because the account has permissions on all APIs in
 OSS. We recommend that you log on as a RAM user to access APIs
 or perform routine operations and maintenance. To create a RAM
account, log on to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket = "<yourBucketName>";
$object = "<yourObjectName>";
$download_file = "download.jpg";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint);

// Upload a sample image.
$ossClient->uploadFile($bucket, $object, "example.jpg");
```

```
// Customize an image style
$style = "style/oss-pic-style-w-300";
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => $style);
$ossClient->getObject($bucket, $object, $options);

// Delete the sample image.
$ossClient->deleteObject($bucket, $object);
```

- **Cascade operations**

  Use the following code to perform cascade operations on an image:

```
<? php
if (is_file(__DIR__ . '/../autoload.php')) {
    require_once __DIR__ . '/../autoload.php';
}
if (is_file(__DIR__ . '/../vendor/autoload.php')) {
    require_once __DIR__ . '/../vendor/autoload.php';
}

use OSS\OssClient;

// It is highly risky to log on with AccessKey of an Alibaba
Cloud account because the account has permissions on all APIs in
 OSS. We recommend that you log on as a RAM user to access APIs
 or perform routine operations and maintenance. To create a RAM
account, log on to https://ram.console.aliyun.com.
$accessKeyId = "<yourAccessKeyId>";
$accessKeySecret = "<yourAccessKeySecret>";
// This example uses endpoint China East 1 (Hangzhou). Specify the
 actual endpoint based on your requirements.
$endpoint = "http://oss-cn-hangzhou.aliyuncs.com";
$bucket = "<yourBucketName>";
$object = "<yourObjectName>";
$download_file = "download.jpg";

$ossClient = new OssClient($accessKeyId, $accessKeySecret, $
endpoint, false);

// Upload a sample image.
$ossClient->uploadFile($bucket, $object, "example.jpg");

// Perform cascade operations.
$style = "image/resize,m_fixed,w_100,h_100/rotate,90";
$options = array(
    OssClient::OSS_FILE_DOWNLOAD => $download_file,
    OssClient::OSS_PROCESS => $style);
$ossClient->getObject($bucket, $object, $options);

// Delete the sample image.
$ossClient->deleteObject($bucket, $object);
```

IMG tools

You can use the IMG viewer *ImageStyleViever* to directly view the IMG result.

6.11 Exception handling

OSS PHP SDK exceptions (OssException) include various errors, such as invalid parameters and file missing. You can use the getMessage method to obtain error messages.

For more information about OssException, see *GitHub*.

Example for handling exceptions

The following code shows how to handle exceptions when creating a existing bucket and prints error messages.

```
    try {
        $ossClient->createBucket($bucket);
    } catch (OssException $e) {
print("Exception:" . $e->getMessage() . "\n");
    }
```

You can also obtain the following information:

Parameter	Description
HTTPStatus	HTTP status code. You can get the status code using the getHTTPStatus method.
ErrorCode	Specifies the error code returned by OSS. You can get the error code using the getErrorCode method.
ErrorMessage	Specifies the error messages returned by OSS. You can get the error messages using the getErrorMessage method.
RequestId	Specifies the UUID. It is unique and used to identify a request. If a problem persists, send the RequestId to OSS developers for help. You can get the RequestId using the getRequestId method.
Details	Specifies detailed error messages returned by OSS. You can get the detailed error messages using the getDetails method.

Common OSS error codes

Error code	Description	HTTP status code
AccessDenied	Access denied	403
BucketAlreadyExists	The bucket already exists.	409
BucketNotEmpty	The bucket is not empty.	409
EntityTooLarge	The entity size exceeds the maximum limit.	400
EntityTooSmall	The entity size is below the minimum limit.	400
FileGroupTooLarge	The total file group size exceeds the maximum limit.	400
FilePartNotExist	No such part exists.	400
FilePartStale	The part has expired.	400
InvalidArgument	The parameter format is invalid.	400
InvalidAccessKeyId	No such AccessKeyId exists.	403
InvalidBucketName	The bucket name is invalid.	400
InvalidDigest	The digest is invalid.	400
InvalidObjectName	The object name is invalid.	400
InvalidPart	The part is invalid.	400
InvalidPartOrder	The part order is invalid.	400
InvalidTargetBucketForLogging	The buckets that store log files are invalid.	400
InternalError	Internal OSS error	500
MalformedXML	The XML format is invalid.	400
MethodNotAllowed	The method is not allowed.	405
MissingArgument	Parameters are not configured.	411

Error code	Description	HTTP status code
MissingContentLength	The content length is not configured.	411
NoSuchBucket	No such bucket exists.	404
NoSuchKey	No such object exists.	404
NoSuchUpload	No such uploadId exists.	404
NotImplemented	The methods are not implemented.	501
PreconditionFailed	Precondition error	412
RequestTimeTooSkewed	The local time set for OSSClient is deviated from the time set for the OSS server by over 15 minutes.	403
RequestTimeout	Request timeout	400
SignatureDoesNotMatch	Signature mismatch	403
InvalidEncryptionAlgorithmError	The specified encryption algorithm (the entropy encoding type) is invalid.	400

7 C

7.1 Preface

This document is written based on OSS C SDK 3.5.0.

Compatibility

- **For SDK series of 3.*.*: compatible**
- **For SDK series of 2.*.*:**
  - **Windows: compatible**
  - **Linux API: compatible except for the linked list (aos_list_t) traversal interface.**
    - **os_list_for_each_entry**
    - **aos_list_for_each_entry_reverse**
    - **aos_list_for_each_entry_safe**
    - **aos_list_for_each_entry_safe_reverse**
- **For SDK series of 1.0.0: incompatible for only the following APIs**
  - **oss_request_options_t**
  - **oss_get_object_to_buffer**
  - **oss_get_object_to_file**
  - **oss_get_object_to_buffer_by_url**
  - **oss_get_object_to_file_by_url**
  - **oss_init_multipart_upload**
  - **oss_complete_multipart_upload**
- **For SDK series of 0.0.*: incompatible**

SDK source code

For the Java SDK source code, see *GitHub*.

Code samples

OSS SDK for C provides a variety of code samples for your reference or use. The following table describes the content of code samples:

Sample file	Content
oss_put_object_sample	*Upload objects*
oss_get_object_sample.c	*Download objects*
oss_append_object_sample.c	*Append upload*
oss_multipart_upload_sample.c	*Multipart upload*
oss_resumable_sample.c	*Resumable upload*, *Resumable download*
oss_head_object_sample.c	*Manage the metadata of an object*
oss_list_object_sample.c	*List objects*
oss_delete_object_sample.c	*Delete an object*
oss_callback_sample.c	*Upload callback*
oss_progress_sample.c	*Progress bar*, *Progress bar*
oss_crc_sample.c	**CRC check during upload or download**
oss_image_sample.c	*Image processing*

7.2 Installation

This topic describes how to install OSS C SDK.

Install OSS C SDK in Linux

- **Install CMake and third-party libraries**

  When installing OSS C SDK in Linux, you must install CMake and the following third-party libraries: curl, apr, apr-util and minixml.

  The following parameters are required to install the environment:

Parameter	Description	Version requirement
CMake	**Build and installation tool**	**Version 2.6.0 and later**
curl	**Focuses on network problems.**	**Version 7.32.0 and later**
apr-util	**Focuses on memory management and cross-platform operations.**	**Version 1.5.2 and later**

Parameter	Description	Version requirement
minixml	Parses XML returned by a request.	We recommend you use version 2.9.

Select the operation system that you need to install CMake and third-party libraries.

- Ubuntu/Debian
  - Install CMake

```
sudo apt-get install cmake
```

  - Install third-party libraries

```
sudo apt-get install libcurl4-openssl-dev libapr1-dev
libaprutil1-dev libmxml-dev
```

    If the mxml installation package is not included in the yum source, you can install the libraries using RPM. Download the rpm installation package based on your operation system:

    - 64-bit: *mxml-2.9-1.x86_64.rpm*
    - 32-bit: *mxml-2.9-1.i386.rpm*

    Run the following code to install mxml with the rpm installation package:

```
rpm -ivh mxml-2.9-1.x86_64.rpm
```

- RedHat/Aliyun/CentOS
  - Install CMake

```
sudo yum install cmake
```

  - Install third-party libraries

```
sudo yum install curl-devel apr-devel apr-util-devel  mxml mxml
-devel
```

- SuSE
  - Install CMake

```
zypper install cmake
```

```
```

■ **Install third-party libraries**

```
zypper install libcurl-devel libapr1-devel libapr-util1-devel
mxml-devel
```

- **Other Linux**

■ **Install CMake:** *Download link*

The common installation method is as follows:

```
./configure
make
make install
```

Note:

When running ./configure, the default installation path is /usr/local/. If you want to specify another path, use the ./configure --prefix option.

■ **Install libcurl:** *Download link*

The common installation method is as follows:

```
./configure
make
make install
```

■ **Install apr:** *Download link*

The common installation method is as follows:

```
./configure
make
make install
```

■ **Install apr-util:** *Download link*

The common installation method is as follows:

```
// You must specify the --with-apr option in the installation.
./configure --with-apr=/your/apr/install/path
make
make install
```

■ **Install minixml:** *Download link*

Note:

We recommend you use minixml version 2.xx.

The common installation method is as follows:

```
./configure
make
sudo make install
```

- **Download SDK**
  - *Download SDK directly*
  - *Download SDK from GitHub*
  - *Download historical SDK*
- **Install SDK**

  Use `cmake`. If third-party libraries, such as url, apr, apr-util and mxml are not installed in the default method during the compiling process, you must specify the path of their header files and library files.

  Install the SDK as follows:

```
cmake .
make
make install
```

  An installation example in CentOS is as follows:

```
cmake -f CMakeLists.txt
// The compilation type is Release. Common compilation types include
: Debug, Release, RelWithDebInfo, and MinSizeRel, in which Debug is
the default type.
-DCMAKE_BUILD_TYPE=Release
// Customize the installation directory.
-DCMAKE_INSTALL_PREFIX=/usr/local/
// Specify the path where the header files and library files of
third party libaries, such as curl, apr, apr-util, and xml, are
located,
-DCURL_INCLUDE_DIR=/usr/include/curl
-DCURL_LIBRARY=/usr/lib64/libcurl.so
-DAPR_INCLUDE_DIR=/usr/include/apr-1
-DAPR_LIBRARY=/usr/lib64/libapr-1.so
-DAPR_UTIL_INCLUDE_DIR=/usr/include/apr-1
-DAPR_UTIL_LIBRARY=/usr/lib64/libaprutil-1.so
-DMINIXML_INCLUDE_DIR=/usr/include
-DMINIXML_LIBRARY=/usr/lib64/libmxml.so
// If the "Could not find apr-config/apr-1-config" error occurs
during the compiling process, it indicates that the apr-1-config
file cannot be found in the default path. Add this option.
-DAPR_CONFIG_BIN=/path/to/bin/apr-1-config
// If the "Could not find apu-config/apu-1-config" error occurs, add
 this option.
-DAPU_CONFIG_BIN=/path/to/bin/apu-1-config
```

Install OSS C SDK in Windows

- **Download SDK**
  - *Download SDK directly*
  - *Download SDK from GitHub*
  - *Download historical SDK*
- **Install SDK**

  For detailed steps and common problems when using Visual Studio to compile OSS C SDK, see *Compile and Use Alibaba Cloud OSS C SDK in Windows*.

  Note:
  If you use Visual Studio 2012 and later versions, it prompts you to upgrade the project to the latest compiler and libraries. We recommended you to determine whether to upgrade based on your project. You can upgrade if your project uses the latest compiler and libraries.

7.3 Quick start

This topic describes how to use OSS C SDK to complete routine operations such as bucket creation, objects uploads, and object downloads.

Demo

- **Linux demo**

  Linux-based demo: *aliyun-oss-c-sdk-demo.tar.gz*.

  1. **Download and extract the demo project.**
  2. **You must specify the installation directory for oss-c-sdk-demo-specified-installation:** `/home/your/oss/csdk`**. You do not need to specify installation directories for other demo projects because they are automatically installed based on the OSS C SDK and other dependent third-party libraries.**
  3. **Replace the OSS_ENDPOINT, ACCESS_KEY_ID, ACCESS_KEY_SECRET, and BUCKET_NAME in the demo with valid values. Use** `LD_LIBRARY_PATH` **to specify**

the directory of the OSS C SDK and the dynamic libraries of the dependent libraries if they are not in the system directory.

4. Enter the directory of the project (oss-c-sdk-demo-xxx), run the `make` command to compile the demo project. Then, run the `./main` command to run the executable program. If the project needs to be re-compiled, run the `make clean` command first before compiling.

- Windows demo

  Windows-based demo: *aliyun-oss-c-sdk-sample.zip*. You can download more demo projects from *GitHub*.

  1. Download and extract the demo project.
  2. Use Visual Studio to open the demo project, and then replace the following parameters in the *oss_config.c* with valid values: OSS_ENDPOINT, ACCESS_KEY_ID, ACCESS_KEY_SECRET, BUCKET_NAME, OBJECT_NAME, MULTIPART_UPLOAD_FILE_PATH, and DIR_NAME. Compile and run the project. You can develop your own project based on the demo project.

Create a bucket

A bucket is a global namespace on OSS used to store objects. It is equivalent to a data container. You can run the following code to create a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates the CNAME is
not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as the network and memory. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
```

```
    }
    /* Initialize a memory pool used to manage memories, which is
equivalent to apr_pool_t. The implementation code is included in the
apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Create the bucket. */
    aos_status_t *resp_status;
    aos_table_t *resp_headers;
    oss_acl_e oss_acl = OSS_ACL_PRIVATE;
    aos_string_t bucket;
    /* Assign a char* to the bucket. */
    aos_str_set(&bucket, bucket_name);
    resp_status = oss_create_bucket(oss_client_options, &bucket,
oss_acl, &resp_headers);
    /* Identify whether the request succeeds */
    if (aos_status_is_ok(resp_status)) {
        printf("create bucket succeeded\n");
    } else {
        printf("create bucket failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

For more information about endpoints, see *Regions and endpoints*.

Upload an object

You can run the following code to upload an object to OSS:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates the CNAME is
not used. */
```

```
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) != AOSE_OK) { = AOSE_OK) {
        exit(1);
    }
    /* Initialize a memory pool used to manage memories, which is
equivalent to apr_pool_t. The implementation code is included in the
apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pool. */ */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memory in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_string_t object;
    aos_status_t *resp_status;
    aos_table_t *headers;
    aos_table_t *resp_headers;
    aos_list_t buffer;
    aos_buf_t *content = NULL;
    const char *data = "More than just cloud.";
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    headers = aos_table_make(pool, 0);
    /* Convert the char* type data to the aos_list_t type data. */
    aos_list_init(&buffer);
    content = aos_buf_pack(oss_client_options->pool, data, strlen(data
));
    aos_list_add_tail(&content->node, &buffer);
    /* Upload an object. */
    resp_status = oss_put_object_from_buffer(oss_client_options, &
bucket, &object, &buffer, headers, &resp_headers);
    /* Determine whether the upload request is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("put file succeeded\n");
    } else {
        printf("put file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

Download an object

You can run the following code to download a specified object:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *download_filename = "<yourDownloadedFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates the CNAME is
not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Initialize a memory pool used to manage memories, which is
equivalent to apr_pool_t. The implementation code is included in the
apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_status_t *resp_status;
    aos_table_t *headers = NULL;
    aos_table_t *params = NULL;
    aos_table_t *resp_headers = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, download_filename);
    headers = aos_table_make(pool, 0);
```

```
    /* Download an object. */
    resp_status = oss_get_object_to_file(oss_client_options, &bucket,
 &object, headers, params, &file, &resp_headers);
    /* Determine whether the download request is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("get object to local file succeeded\n");
    } else {
        printf("get object to local file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

List objects in a bucket

You can run the following code to list objects stored in a specified bucket.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates the CNAME is
not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Initialize a memory pool used to manage memories, which is
equivalent to apr_pool_t. The implementation code is included in the
apr library. */
    aos_pool_t *pool = NULL;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pool. */ */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options = NULL;
    /* Allocate memory in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
```

```
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_status_t *resp_status = NULL;
    oss_list_object_params_t *params = NULL;
    oss_list_object_content_t *content = NULL;
    int size = 0;
    char *line = NULL;
    aos_str_set(&bucket, bucket_name);
    params = oss_create_list_object_params(pool);
    /* List objects */
    resp_status = oss_list_object(oss_client_options, &bucket, params
, NULL);
    /* Determine whether the list request is successful. */
    if (! aos_status_is_ok(resp_status))
    {
        printf("list object failed\n");
    }
    else{
        /* Print objects. */
        printf("Object\tSize\tLastModified\n");
        aos_list_for_each_entry(oss_list_object_content_t, content, &
params->object_list, node) {
            ++size;
            line = apr_psprintf(pool, "%.*s\t%.*s\t%.*s\n", content->
key.len, content->key.data,
                content->size.len, content->size.data,
                content->last_modified.len, content->last_modified.
data);
            printf("%s", line);
        }
        printf("Total %d\n", size);
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Delete an object.

You can run the following code to delete an object:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates the CNAME is
not used. */
```

```
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Initialize a memory pool used to manage memories, which is
equivalent to apr_pool_t. The implementation code is included in the
apr library. */
    aos_pool_t *pool = NULL;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pool. */ */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options = NULL;
    /* Allocate memory in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_string_t object;
    aos_status_t *resp_status = NULL;
    aos_table_t *resp_headers = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    /* Delete an object. */
    resp_status = oss_delete_object(oss_client_options, &bucket, &
object, &resp_headers);
    /* Determine whether the delete request is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("delete object succeeded\n");
    } else {
        printf("delete object failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.4 Initialization

To use OSS C SDK, you must initialize request options (oss_request_options_t) and specify an endpoint. For more information about endpoint, see *Regions and endpoints* **and** *Bind a custom domain name*.

Use the OSS domain name to initialize request options.

You can run the following code to initialize request options with the OSS domain name:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main() {
    aos_pool_t *p;
    oss_request_options_t *options;
    /* Initialize global variables, which are run first before the
program starts. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        return -1;
    }
    /* Initialize the memory pool and options. */
    aos_pool_create(&p, NULL);
    options = oss_request_options_create(p);
    init_options(options);
    /* Logical code, which is not listed here. */
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(p);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Note:

For the configuration of request options, see related documents. *http://bbs.aliyun.com/read/261421.html?spm=5176.bbsr248504.0.0.NoFhRX*

Use a custom domain name to initialize request options.

You can run the following code to initialize request options with a domain name:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourCustomEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
```

```
const char *access_key_secret = "<yourAccessKeySecret>";
void init_options(oss_request_options_t *options) {
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Enable the CNAME and bind the custom domain name to the bucket.
 */
    options->config->is_cname = 1;
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main() {
    aos_pool_t *p;
    oss_request_options_t *options;
    /* Initialize global variables, which are run before the program
starts. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        return -1;
    }
    /* Initialize the memory pool and options. */
    aos_pool_create(&p, NULL);
    options = oss_request_options_create(p);
    init_options(options);
    /* Logical code, which is not listed here. */
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(p);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Use STS to initialize request options

You can run the following code to initialize request options with STS:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *sts_token = "<yourSecurityToken>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Configure STS */
    aos_str_set(&options->config->sts_token, sts_token);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main() {
    aos_pool_t *p;
    oss_request_options_t *options;
```

```
    /* Initialize global variables, which are run first before the
program starts. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        return -1;
    }
    /* Initialize the memory pool and options. */
    aos_pool_create(&p, NULL);
    options = oss_request_options_create(p);
    init_options(options);
    /* Logical code, which is not listed here. */
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(p);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Set the timeout value.

You can run the following code to set the timeout value:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
    /* Set the connection timeout value, which is 10 seconds by
default. */
    options->ctl->options->connect_timeout = 10;
    /* Set the DNS timeout value, which is 60 seconds by default. */
    options->ctl->options->dns_cache_timeout = 60;
    /* Set the request timeout value:
    Set the value of speed_limit to specify the minimum speed that can
 be accepted, which is 1,024 Bytes/s by default.
    Set the value of speed_time to specify the maximum time period
that can be accepted, which is 15 seconds by default.
    The following code indicates that if the trasmission speed is
continuously lower than 1,024 Bytes/s for 15 seconds, the request is
time-out. */
    options->ctl->options->speed_limit = 1024;
    options->ctl->options->speed_time = 15;
```

```
}
```

7.5 Buckets

7.5.1 Bucket

A bucket serves as a container that stores objects. Objects belong to a bucket.

Create a bucket

Run the following code to create a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    oss_acl_e oss_acl = OSS_ACL_PRIVATE;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign a char* to the aos_string_t bucket. */
    aos_str_set(&bucket, bucket_name);
```

```
    /* Create a bucket. */
    resp_status = oss_create_bucket(oss_client_options, &bucket,
oss_acl, &resp_headers);
    /* Determine whether the request is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("create bucket succeeded\n");
    } else {
        printf("create bucket failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

For more information about bucket naming rules, see naming conventions in *Basic concepts*. You can specify the ACL and the storage class when you create a bucket, as shown in *Bucket-level permissions* and *Storage class*.

You can run the following code to create a bucket with a specified ACL:

```
    /* Create a bucket and set the bucket ACL to public read (which is
 private by default). */
    resp_status = oss_create_bucket(oss_client_options, &bucket,
OSS_ACL_PUBLIC_READ, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("create bucket succeeded\n");
    } else {
        Printf ("Create Bucket failed \ n ");
    }
```

You can run the following code to creates a bucket of the archive class:

```
    resp_status = oss_create_bucket_with_storage_class(oss_client
_options, &bucket, OSS_ACL_PRIVATE, OSS_STORAGE_CLASS_ARCHIVE);
    /* Determine whether the request is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("create bucket succeeded\n");
    } else {
        printf("create bucket failed\n");
    }
```

You can run the followign code to create a bucket of the Infrequent Access class:

```
    resp_status = oss_create_bucket_with_storage_class(oss_client
_options, &bucket, OSS_ACL_PRIVATE, OSS_STORAGE_CLASS_IA);
    /* Determine whether the request is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("create bucket succeeded\n");
    } else {
        printf("create bucket failed\n");
```

```
    }
```

List buckets

Run the following code to list all buckets:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout.
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as   networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_list_buckets_params_t *params = NULL;
    oss_list_bucket_content_t *content = NULL;
    int size = 0;
    params = oss_create_list_buckets_params(pool);
    /* List all buckets. */
    resp_status = oss_list_bucket(oss_client_options, params, &
resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("list buckets succeeded\n");
```

```
    } else {
        printf("list buckets failed\n");
    }
    /* Print bucket information. */
    aos_list_for_each_entry(oss_list_bucket_content_t, content, &
params->bucket_list, node) {
        printf("BucketName: %s\n", content->name.data);
        ++size;
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Configure an ACL for a bucket

The ACL of a bucket includes the following permissions.

Permission	Description	Value
Private	The bucket owner and the authorized users can read and write objects in the bucket. Other users cannot perform any operation on the objects.	OSS_ACL_PRIVATE
Public read	The bucket owner and the authorized users can read and write objects in the bucket. Other users can only read the objects in the bucket. Authorize this permission with caution.	OSS_ACL_PUBLIC_READ
Public read-write	All users can read and write objects in the bucket. Authorize this permission with caution.	OSS_ACL_PUBLIC_READ_WRITE

For more information about the ACL, see *Access control* in the developer guide.

You can run the following code to configure an ACL for a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
```

```
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout.
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as

networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign a char* to the aos_string_t bucket. */
    aos_str_set(&bucket, bucket_name);
    /* Set the bucket ACL to public read.
    resp_status = oss_put_bucket_acl(oss_client_options, &bucket,
OSS_ACL_PUBLIC_READ, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("set bucket acl succeeded\n");
    } else {
        printf("set bucket acl failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

Obtain an ACL for a bucket

Run the following code to obtain the ACL for a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as

networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t oss_acl;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign a char* to the aos_string_t bucket */ */
    aos_str_set(&bucket, bucket_name);
    /* Obtain the ACL for a bucket. */
```

```
    resp_status = oss_get_bucket_acl(oss_client_options, &bucket, &
oss_acl, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket acl succeeded : %s \n", oss_acl.data);
    } else {
        printf("get bucket acl failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Obtain the region of a bucket

Run the following code to obtain the region (or location) for a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout.
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as

networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
```

```
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t oss_location;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign a char* to the aos_string_t bucket. */
    aos_str_set(&bucket, bucket_name);
    /* Obtain the region of the bucket. */
    resp_status = oss_get_bucket_location(oss_client_options, &bucket,
 &oss_location, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket location succeeded : %s \n", oss_location.
data);
    } else {
        printf("get bucket location failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Obtain bucket information

Run the following code to obtain bucket information:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as

networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
```

```
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    oss_bucket_info_t bucket_info;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign a char* to the aos_string_t bucket. */
    aos_str_set(&bucket, bucket_name);
    /* Obtain the bucket inforamtion. */
    resp_status = oss_get_bucket_info(oss_client_options, &bucket, &
bucket_info, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket info succeeded\n");
        printf("location: %s\n", bucket_info.location.data);
        printf("owner_id: %s\n", bucket_info.owner_id.data);
        printf("owner_name: %s\n", bucket_info.owner_name.data);
        printf("created_date: %s\n", bucket_info.created_date.data);
        printf("location: %s\n", bucket_info.location.data);
    } else {
        printf("get bucket info failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Delete a bucket

Before a bucket is deleted, ensure that all objects in the bucket and fragments that are generated from multipart upload are deleted.

Note:

To delete the fragments that are generated from multipart upload, use oss_list_upload_part to list all fragments, and then use oss_abort_multipart_upload to delete the fragments. For more information, see *Multipart upload*.

Run the following code to delete a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as

networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign a char* to the aos_string_t bucket. */
    aos_str_set(&bucket, bucket_name);
    /* Delete the bucket. */
    resp_status = oss_delete_bucket (oss_client_options, &bucket, &
resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("delete bucket succeeded\n");
    } else {
        printf("delete bucket failed\n");
    }
```

```
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.5.2 Manage lifecycle rules

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**
- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**
- **Status: This lifecyle rule takes effect or not.**

The parts uploaded by uploadPart method also support lifecycle rules. The last modification time of an object is the time the multipart upload event is initiated.

For more information about lifecycle management, see *Manage object lifecycle*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
```

```
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_list_t lifecycle_rule_list;
    /* Create a lifecycle rule for the bucket. */
    aos_str_set(&bucket, bucket_name);
    aos_list_init(&lifecycle_rule_list);
    /* Specify the expiration duration. */
    oss_lifecycle_rule_content_t *rule_content_days = oss_create
_lifecycle_rule_content(pool);
    aos_str_set(&rule_content_days->id, "rule-1");
    aos_str_set(&rule_content_days->prefix, "obsoleted");
    aos_str_set(&rule_content_days->status, "Enabled");
    rule_content_days->days = 3;
    aos_list_add_tail(&rule_content_days->node, &lifecycle_rule_list);
    /* Specify the expiration date. */
    oss_lifecycle_rule_content_t *rule_content_date = oss_create
_lifecycle_rule_content(pool);
    aos_str_set(&rule_content_date->id, "rule-2");
    aos_str_set(&rule_content_date->prefix, "delete");
    aos_str_set(&rule_content_date->status, "Enabled");
    aos_str_set(&rule_content_date->date, "2022-10-11T00:00:00.000Z");
    aos_list_add_tail(&rule_content_date->node, &lifecycle_rule_list);
    /* Configure a lifecycle rule. */
    resp_status = oss_put_bucket_lifecycle(oss_client_options, &bucket
, &lifecycle_rule_list, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put bucket lifecycle succeeded\n");
    } else {
```

```
        printf("put bucket lifecycle failed, code:%d, error_code:%s,
error_msg:%s, request_id:%s\n",
            resp_status->code, resp_status->error_code, resp_status->
error_msg, resp_status->req_id);
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

View lifecycle rules

Run the following code to view lifecycle rules:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
```

```
    aos_list_t lifecycle_rule_list;
    oss_lifecycle_rule_content_t *rule_content;
    char *rule_id;
    char *prefix;
    char *status;
    int days = INT_MAX;
    char* date = "";
    aos_str_set(&bucket, bucket_name);
    /* Obtain and print the lifecycle of the bucket. */
    aos_list_init(&lifecycle_rule_list);
    resp_status = oss_get_bucket_lifecycle(oss_client_options, &bucket
, &lifecycle_rule_list, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket lifecycle succeeded\n");
        aos_list_for_each_entry(oss_lifecycle_rule_content_t,
rule_content, &lifecycle_rule_list, node) {
            rule_id = apr_psprintf(pool, "%.*s", rule_content->id.len
, rule_content->id.data);
            prefix = apr_psprintf(pool, "%.*s", rule_content->prefix.
len, rule_content->prefix.data);
            status = apr_psprintf(pool,"%.*s", rule_content->status.
len, rule_content->status.data);
            date = apr_psprintf(pool, "%.*s", rule_content->date.len,
rule_content->date.data);
            days = rule_content->days;
        }
        printf("rule_id: %s \n", rule_id);
        printf("prefix: %s \n", prefix);
        printf("status: %s \n", status);
        printf("date: %s \n", date);
        printf("days: %d \n", days);
    } else {
        printf("get bucket lifecycle failed, code:%d, error_code:%s,
error_msg:%s, request_id:%s\n",
            resp_status->code, resp_status->error_code, resp_status->
error_msg, resp_status->req_id);
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
```

```
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    /* Delete bucket lifecycle rules. */
    aos_str_set(&bucket, bucket_name);
    resp_status = oss_delete_bucket_lifecycle(oss_client_options, &
bucket, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("delete bucket lifecycle succeeded\n");
    } else {
        printf("delete bucket lifecycle failed, code:%d, error_code:%s
, error_msg:%s, request_id:%s\n",
            resp_status->code, resp_status->error_code, resp_status->
error_msg, resp_status->req_id);
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.5.3 CORS

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* and *PutBucketcors* in OSS Developer Guide.

Configure CORS rules

Run the following code to configure CORS rules for a specified bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_list_t cors_rule_list;
    oss_cors_rule_t *cors_rule1 = NULL, *cors_rule2 = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_list_init(&cors_rule_list);
    cors_rule1 = oss_create_cors_rule(pool);
    aos_list_add_tail(&cors_rule1->node, &cors_rule_list);
    oss_create_sub_cors_rule(pool, &cors_rule1->allowed_origin_list, "
allowed_origin_1_1");
```

```
    oss_create_sub_cors_rule(pool, &cors_rule1->allowed_origin_list, "
allowed_origin_1_1");
    oss_create_sub_cors_rule(pool, &cors_rule1->allowed_method_list, "
PUT");
    oss_create_sub_cors_rule(pool, &cors_rule1->allowed_method_list, "
GET");
    oss_create_sub_cors_rule(pool, &cors_rule1->allowed_head_list, "
Authorization");
    oss_create_sub_cors_rule(pool, &cors_rule1->expose_head_list, "
expose_head_1_1");
    oss_create_sub_cors_rule(pool, &cors_rule1->expose_head_list, "
expose_head_1_1");
    cors_rule2 = oss_create_cors_rule(pool);
    aos_list_add_tail(&cors_rule2->node, &cors_rule_list);
    oss_create_sub_cors_rule(pool, &cors_rule2->allowed_origin_list, "
allowed_origin_2_1");
    oss_create_sub_cors_rule(pool, &cors_rule2->allowed_origin_list, "
allowed_origin_2_2");
    oss_create_sub_cors_rule(pool, &cors_rule2->allowed_method_list, "
PUT");
    oss_create_sub_cors_rule(pool, &cors_rule2->allowed_method_list, "
GET");
    oss_create_sub_cors_rule(pool, &cors_rule2->allowed_head_list, "
Authorization");
    oss_create_sub_cors_rule(pool, &cors_rule2->expose_head_list, "
expose_head_2_1");
    oss_create_sub_cors_rule(pool, &cors_rule2->expose_head_list, "
expose_head_2_2");
    /* Configure the CORS rule. */
    resp_status = oss_put_bucket_cors(oss_client_options, &bucket, &
cors_rule_list, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put bucket cors succeeded\n");
    } else {
        printf("put bucket cors failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Obtain CORS rules

Run the following code to obtain CORS rules:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
```

```
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_list_t cors_rule_list;
    oss_cors_rule_t *cors_rule = NULL;
    oss_sub_cors_rule_t *sub_cors_rule = NULL;
    aos_str_set(&bucket, bucket_name);
    /* Obtain CORS rules. */
    aos_list_init(&cors_rule_list);
    resp_status = oss_get_bucket_cors(oss_client_options, &bucket, &
cors_rule_list, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket cors succeeded\n");
        aos_list_for_each_entry(oss_cors_rule_t, cors_rule, &
cors_rule_list, node) {
            printf("max_age_seconds: %d\n", cors_rule->max_age_seconds
);
            aos_list_for_each_entry(oss_sub_cors_rule_t, sub_cors_rule
, &cors_rule->allowed_origin_list, node) {
                printf("allowed_origin_list: %s \n", sub_cors_rule->
rule.data);
            }
            aos_list_for_each_entry(oss_sub_cors_rule_t, sub_cors_rule
, &cors_rule->allowed_method_list, node) {
                printf("allowed_method_list: %s \n", sub_cors_rule->
rule.data);
            }
            aos_list_for_each_entry(oss_sub_cors_rule_t, sub_cors_rule
, &cors_rule->allowed_head_list, node) {
                printf("allowed_head_list: %s \n", sub_cors_rule->rule
.data);
            }
            aos_list_for_each_entry(oss_sub_cors_rule_t, sub_cors_rule
, &cors_rule->expose_head_list, node) {
                printf("expose_head_list: %s \n", sub_cors_rule->rule.
data);
```

```
            }
        }
    } else {
        printf("get bucket cors failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Delete CORS rules

Run the following code to delete all CORS rules for the specified bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
```

```
    aos_str_set(&bucket, bucket_name);
    /* Delete all CORS rules. */
    resp_status = oss_delete_bucket_cors(oss_client_options, &bucket,
 &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket cors succeeded\n");
    } else {
        printf("get bucket cors failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.5.4 Set logging

You can enable access logs to record bucket access to log files, which are stored in a specified bucket.

The log file format is as follows:

`<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString`

For more information about access log files, see *Set access logging*.

Enable access logging

Run the following code to enable bucket access logging:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *target_bucket_name = "<yourTargetBucketName>";
const char *target_logging_prefix = "<yourTargetPrefix>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the maid method at the program portal to initialize global
 resources such as network, memory, and so on. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
```

```
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    oss_logging_config_content_t *content;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    content = oss_create_logging_rule_content(pool);
    aos_str_set(&content->target_bucket, target_bucket_name);
    aos_str_set(&content->prefix, target_logging_prefix);
    /* Enable bucket access logging. */
    resp_status = oss_put_bucket_logging(oss_client_options, &bucket,
content, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put symlink succeeded\n");
    } else {
        printf("put symlink failed, code:%d, error_code:%s, error_msg:
%s, request_id:%s\n",
            resp_status->code, resp_status->error_code, resp_status->
error_msg, resp_status->req_id);
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

View access logging configurations

Run the following code to view the access logging configurations for a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
```

```
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    oss_logging_config_content_t *content;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    content = oss_create_logging_rule_content(pool);
    /* View bucket access logging configurations. */
    resp_status = oss_get_bucket_logging(oss_client_options, &bucket,
content, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket logging succeeded\n");
    } else {
        printf("get bucket logging failed, code:%d, error_code:%s,
error_msg:%s, request_id:%s\n",
            resp_status->code, resp_status->error_code, resp_status->
error_msg, resp_status->req_id);
    }
    printf("bucket:%s, prefix:%s\n", content->target_bucket.data,
content->prefix.data);
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Disable access logging

Run the following code to disable access logging for a bucket:

```
#include "oss_api.h"
#include "aos_http_io.h"
```

```
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    /* Disable access logging. */
    resp_status = oss_delete_bucket_logging(oss_client_options, &
bucket, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("delete bucket logging succeeded\n");
    } else {
        printf("delete bucket logging failed, code:%d, error_code:%s,
error_msg:%s, request_id:%s\n",
            resp_status->code, resp_status->error_code, resp_status->
error_msg, resp_status->req_id);
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

7.5.5 Hotlink protection

This topic describes how to use hotlink protection.

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49*.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
```

```
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_referer_config_t referer_config;
    aos_str_set(&bucket, bucket_name);
    aos_list_init(&referer_config.referer_list);
    oss_create_and_add_refer(pool, &referer_config, "http://www.aliyun
.com");
    oss_create_and_add_refer(pool, &referer_config, "https://www.
aliyun.com");
    referer_config.allow_empty_referer = 0;
    /* Add the referer field. The referer field allows question marks
 (?) and asterisks (*) for wildcard use. */
    resp_status = oss_put_bucket_referer(oss_client_options, &bucket,
 &referer_config, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put bucket referer succeeded\n");
    } else {
        printf("put bucket referer failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Obtain a referer whiltelist

Run the following code to obtain a referer whiltelist:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
```

```
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_referer_config_t referer_config;
    oss_referer_t *referer;
    aos_str_set(&bucket, bucket_name);
    aos_list_init(&referer_config.referer_list);
    /* Obtain the referer list for a bucket. */
    resp_status = oss_get_bucket_referer(oss_client_options, &bucket,
 &referer_config, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket referer succeeded\n");
        aos_list_for_each_entry(oss_referer_t, referer, &referer_co
nfig.referer_list, node) {
            printf("get referer %s\n", referer->referer.data);
        }
    } else {
        printf("get bucket referer failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
```

```
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_referer_config_t referer_config;
    aos_str_set(&bucket, bucket_name);
    aos_list_init(&referer_config.referer_list);
    referer_config.allow_empty_referer = 1;
    /* You cannot clear a referer whitelist directly. To clear a
referer whitelist, you need to create the rule that allows an empty
referer field and replace the original rule with the new rule. */
    resp_status = oss_put_bucket_referer(oss_client_options, &bucket,
 &referer_config, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("delete bucket referer succeeded\n");
    } else {
        printf("delete bucket referer failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

7.5.6 Static website hosting

You can set your bucket to static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

For more information about static website hosting, see *Configure static website hosting*.

Configure static website hosting

Run the following code to configure static website hosting:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&p, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
```

```
    oss_website_config_t website_config;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&website_config.suffix_str, "index.html");
    aos_str_set(&website_config.key_str, "error.html");
    /* Configure static web hosting: The index page is index.html and
the error page is error.html. */
    resp_status = oss_put_bucket_website(oss_client_options, &bucket,
 &website_config, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put bucket website succeeded\n");
    } else {
        printf("put bucket website failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

View static website hosting configurations

Run the following code to view static website hosting configurations:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
```

```
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_website_config_t website_config;
    aos_str_set(&bucket, bucket_name);
    /* View static web hosting configurations. */
    resp_status = oss_get_bucket_website(oss_client_options, &bucket,
 &website_config, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get bucket website succeeded\n");
        printf("website_config: %s  %s \n", website_config.suffix_str.
data, website_config.key_str.data);
    } else {
        printf("get bucket website failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Delete static website hosting configurations

Run the following code to delete static website hosting configurations:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
```

```
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    /* Delete static website hosting configurations. */
    resp_status = oss_delete_bucket_website(oss_client_options, &
bucket, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("delete bucket website succeeded\n");
    } else {
        printf("delete bucket website failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.6 Upload objects

7.6.1 Overview

In OSS, an object is the basic unit for data operations. OSS C SDK provides the following file upload methods:

- *Simple upload*: **supports the upload of a file up to 5 GB in size. It includes streaming upload and file upload.**
- *Append upload*: **supports the upload of a file up to 5 GB in size.**
- *Resumable upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files. It also supports concurrent upload and you can define the size of each part.**
- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files.**

Note:

For more information about the usage scenarios of each upload method, see "Upload files" in OSS Developer Guide.

During the file upload, you can set *Object Meta*, and check the upload progress in *upload progress bar*. After you complete the file upload, you can perform *upload callback*.

7.6.2 Simple upload

This topic describes how to use simple upload.

For the complete code of simple upload, see *GitHub*.

Upload data from memory to OSS

You can run the following code to upload data from memory to OSS:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *object_content = "More than just cloud.";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
```

```
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_list_t buffer;
    aos_buf_t *content = NULL;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_list_init(&buffer);
    content = aos_buf_pack(oss_client_options->pool, object_content,
strlen(object_content));
    aos_list_add_tail(&content->node, &buffer);
    /* Upload a file. */
    resp_status = oss_put_object_from_buffer(oss_client_options, &
bucket, &object, &buffer, headers, &resp_headers);
    /* Determine whether the upload is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("put object from buffer succeeded\n");
    } else {
        printf("put object from buffer failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Upload local files

You can run the following code to upload a local file to OSS:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
```

```
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /*  Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    /* Upload a file. */
    resp_status = oss_put_object_from_file(oss_client_options, &bucket
, &object, &file, headers, &resp_headers);
    /* Determine whether the upload is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("put object from file succeeded\n");
    } else {
        printf("put object from file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

FAQ: Files with Chinese characters in their names cannot be uploaded.

For example, in Windows, you can upload a file with Chinese characters in its name after encoding its name in UTF-8. The example code is as follows:

```
#include "oss_api.h"
#include "aos_http_io.h"
#include <wchar.h>
const char endpoint = "<yourEndpoint>";
const char access_key_id = "<yourAccessKeyId>";
const char access_key_secret = "<yourAccessKeySecret>";
const char bucket_name = "<yourBucketName>";
const char object_name = "<yourObjectName>";
const char local_filename_ch[] = "c:\\test.txt";
void init_options(oss_request_options_t options)
```

```
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
the CNAME is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}

char* multibyte_to_utf8(const char * ch, char *str, int size)
{
    if (! ch || ! str || size <= 0)
     return NULL;

    int chlen = strlen(ch);
    int multi_byte_cnt = 0;
    for (int i = 0; i < chlen - 1; i++) {
      if ((ch[i] & 0x80) && (ch[i + 1] & 0x80)) {
             i++;
             multi_byte_cnt++;
      }
    }
    if (multi_byte_cnt == 0)
      return ch;

    int len = MultiByteToWideChar(CP_ACP, 0, ch, -1, NULL, 0);
    if (len <= 0)
      return NULL;
    wchar_t *wch = malloc(sizeof(wchar_t) * (len + 1));
    if (! wch)
      return NULL;
    wmemset(wch, 0, len + 1);
    MultiByteToWideChar(CP_ACP, 0, ch, -1, wch, len);

    len = WideCharToMultiByte(CP_UTF8, 0, wch, -1, NULL, 0, NULL, NULL
);
    if ((len <= 0) || (size < len + 1)) {
      free(wch);
      return NULL;
    }
    memset(str, 0, len + 1);
    WideCharToMultiByte(CP_UTF8, 0, wch, -1, str, len, NULL, NULL);
    free(wch);
    return str;
}

int main(int argc, char argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
     exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
```

```
    aos_pool_create(&pool, NULL);
     /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
     aos_string_t file;
     aos_table_t *headers = NULL;
     aos_table_t *resp_headers = NULL;
     aos_status_t *resp_status = NULL;
     aos_str_set(&bucket, bucket_name);
     aos_str_set(&object, object_name);

     /* Encode the file name with multiple Chinese characters in UTF-8.
 */
     char local_filename_buf[1024];
    char * local_filename = multibyte_to_utf8(local_filename_ch,
local_filename_buf, 1024);
    aos_str_set(&file, local_filename);

    /* Upload a file. */
     resp_status = oss_put_object_from_file(oss_client_options, &bucket
, &object, &file, headers, &resp_headers);
    /* Determine whether the upload is successful. */
    if (aos_status_is_ok(resp_status)) {
      printf("put object from file succeeded\n");
     } else {
      printf("put object from file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.6.3 Append upload

You cannot perform copyObject for appended objects.

Append data from memory

Run the following code to append data from memories

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *object_content = "More than just cloud.";
void init_options(oss_request_options_t *options)
{
```

```
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&p, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_list_t buffer;
    int64_t position = 0;
    char *next_append_position = NULL;
    aos_buf_t *content = NULL;
    aos_table_t *headers1 = NULL;
    aos_table_t *headers2 = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    headers1 = aos_table_make(pool, 0);
    /* Obtain the position where the object is appended for the first
time. */
    resp_status = oss_head_object(oss_client_options, &bucket, &object
, headers1, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        next_append_position = (char*)(apr_table_get(resp_headers, "x-
oss-next-append-position"));
        position = atoi(next_append_position);
    }
    /* Append the file. */
    headers2 = aos_table_make(pool, 0);
    aos_list_init(&buffer);
    content = aos_buf_pack(pool, object_content, strlen(object_content
));
    aos_list_add_tail(&content->node, &buffer);
```

```
    resp_status = oss_append_object_from_buffer(oss_client_options, &
bucket, &object, position, &buffer, headers2, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("append object from buffer succeeded\n");
    } else {
        printf("append object from buffer failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Append data from a local file

Run the following code to append data from a local file:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&p, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
```

```
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    int64_t position = 0;
    char *next_append_position = NULL;
    aos_table_t *headers1 = NULL;
    aos_table_t *headers2 = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    /* Obtain the position where the object is appended for the first
time. */
    resp_status = oss_head_object(oss_client_options, &bucket, &object
, headers1, &resp_headers);
    if(aos_status_is_ok(resp_status)) {
        next_append_position = (char*)(apr_table_get(resp_headers, "x-
oss-next-append-position"));
        position = atoi(next_append_position);
    }
    /* Append the file. */
    resp_status = oss_append_object_from_file(oss_client_options, &
bucket, &object, position, &file, headers2, &resp_headers);
    /* Determine whether the append upload is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("append object from file succeeded\n");
    } else {
        printf("append object from file failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

If the file already exists, the following may occur:

- **If the file cannot be uploaded with append upload, an ObjectNotAppendable exception occurs.**
- **If the file can be uploaded with append upload, the file is upload by the Put Object method and overwrites the existing object. The object type is changed to Normal Object. After the Head Object operation is performed, the system returns x-oss-object-type, which indicates the type of the object.**

7.6.4 Resumable upload

In resumable upload, the object is divided into multiple parts, which are then uploaded separately. After all the parts are uploaded, they are merged into a complete object to finish the upload.

Current upload progress is recorded into a checkpoint file during the upload. If a part fails to be uploaded during the process, the object is uploaded from the part recorded in the checkpoint file when the upload restarts, that is, the resumable upload. After the upload is complete, the checkpoint is deleted.

You can use the `oss_resumable_clt_params_t` method to achieve resumable upload. Common parameters used in this method are listed as follows:

Parameter	Description
thread_num	The number of concurrent threads, which is 1 by default.
enable_checkpoint	Whether to enable resumable upload, which is disabled by default.
partSize	The part size, ranging from 100 KB to 5 GB.
checkpoint_path	The path that the checkpoint file is stored, which is {upload_file_path}.cp by default.

You can run the following code to perform resumable upload:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Deternube whether the CNAME is used. 0 indicates that it is not
 used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
```

```
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_list_t resp_body;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_resumable_clt_params_t *clt_params;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    aos_list_init(&resp_body);
    /* Resumable upload. */
    clt_params = oss_create_resumable_clt_params_content(pool, 1024 *
100, 3, AOS_TRUE, NULL);
    resp_status = oss_resumable_upload_file(oss_client_options, &
bucket, &object, &file, headers, NULL, clt_params, NULL, &resp_headers
, &resp_body);
    if (aos_status_is_ok(resp_status)) {
        printf("resumable upload succeeded\n");
    } else {
        printf("resumable upload failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.6.5 Multipart upload

To enable multipart upload, perform the following steps:

1. **Initiate a multipart upload event.**

   You can call ossClient.initiateMultipartUpload to return the globally unique uploadId created in OSS.

2. **Upload parts.**

   You can call ossClient.uploadPart to upload part data.

   Notice:

   - **For parts with a same uploadId, parts are sequenced by their part numbers. If you have uploaded a part and use the same part number to upload another part, the later part will replace the former part.**
   - **OSS places the MD5 value of part data in ETag and returns the MD5 value to the user.**
   - **SDK automatically configures Content-MD5. OSS calculates the MD5 value of uploaded data and compares it with the MDS value calculated by SDK. If the two values vary, the error code of InvalidDigest is returned.**

3. **Complete multipart upload.**

   After you have uploaded all parts, call partossClient.completeMultipartUpload to combine these parts into a complete object.

The following code is used as a complete example that describes the process of multipart upload:

```
#include "oss_api.h"
#include "aos_http_io.h"
#include <sys/stat.h>
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    ptions->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int64_t get_file_size(const char *file_path)
{
    int64_t filesize = -1;
    struct stat statbuff;
    if(stat(file_path, &statbuff) < 0){
        return filesize;
    } else {
```

```
        filesize = statbuff.st_size;
    }
    return filesize;
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration nformation, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    oss_upload_file_t *upload_file = NULL;
    aos_string_t upload_id;
    aos_table_t *headers = NULL;
    aos_table_t *complete_headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_null(&upload_id);
    headers = aos_table_make(p, 1);
    complete_headers = aos_table_make(p, 1);
    int part_num = 1;
    /* Initialize multipart upload and obtain an upload ID (upload_id
). */
    resp_status = oss_init_multipart_upload(oss_client_options, &
bucket, &object, &upload_id, headers, &resp_headers);
    /* Determines whether the slice upload Initialization is
successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("Init multipart upload succeeded, upload_id:%.*s\n",
               upload_id.len, upload_id.data);
    } else {
        printf("Init multipart upload failed, upload_id:%.*s\n",
               upload_id.len, upload_id.data);
    }
    /* Upload the parts. */
    int64_t file_length = 0;
    int64_t pos = 0;
    file_length = get_file_size(local_filename);
    while(pos < file_length) {
        upload_file = oss_create_upload_file(pool);
        aos_str_set(&upload_file->filename, local_filename);
        upload_file->file_pos = pos;
        pos += 100 * 1024;
        upload_file->file_last = pos < file_length ? pos : file_length
;
```

```
        resp_status = oss_upload_part_from_file(oss_client_options, &
bucket, &object, &upload_id, part_num++, upload_file, &resp_headers);
        if (aos_status_is_ok(resp_status)) {
            printf("Multipart upload part from file succeeded\n");
        } else {
            printf("Multipart upload part from file failed\n");
        }
    }
    oss_list_upload_part_params_t *params = NULL;
    aos_list_t complete_part_list;
    /* Obtain uploaded parts. */
    params = oss_create_list_upload_part_params(pool);
    params->max_ret = 1000;
    aos_list_init(&complete_part_list);
    resp_status = oss_list_upload_part(oss_client_options, &bucket, &
object, &upload_id, params, &resp_headers);
    /* Determine whether the part list is successfully obtained. */
    if (aos_status_is_ok(resp_status)) {
        printf("List multipart succeeded\n");
    } else {
        printf("List multipart failed\n");
    }
    oss_complete_part_content_t *complete_part_content = NULL;
    oss_list_part_content_t *part_content = NULL;
    aos_list_for_each_entry(oss_list_part_content_t, part_content, &
params->part_list, node) {
        complete_part_content = oss_create_complete_part_content(pool
);
        aos_str_set(&complete_part_content->part_number, part_content-
>part_number.data);
        aos_str_set(&complete_part_content->etag, part_content->etag.
data);
        aos_list_add_tail(&complete_part_content->node, &complete_p
art_list);
    }
    /* Complete multipart upload. */
    resp_status = oss_complete_multipart_upload(oss_client_options, &
bucket, &object, &upload_id,
            &complete_part_list, complete_headers, &resp_headers);
    /* Determine whether the multipart upload is complete. */
    if (aos_status_is_ok(resp_status)) {
        printf("Complete multipart upload from file succeeded,
upload_id:%.*s\n",
                upload_id.len, upload_id.data);
    } else {
        printf("Complete multipart upload from file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Note:

You must provide all valid PartETags when you call oss_complete_multipart_upload to obtain uploaded parts. The ETags can be obtained in two methods: 1. The ETag of a part is included in the returned result

when the part is uploaded. 2. You can call oss_list_upload_part to obtain the ETag of an uploaded part. In the example, the second method is used.

Cancel a multipart upload event

Run the following code to cancel a multipart upload event:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as

networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not

inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
global configuration

information, such as endpoint, access_key_id, acces_key_secret,
is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t upload_id;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
```

```
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_null(&upload_id);
    /* Initialize multipart upload and obtain an upload ID (upload_id
). */
    resp_status = oss_init_multipart_upload(oss_client_options, &
bucket, &object, &upload_id, headers, &resp_headers);
    /* Determine whether the multipart upload is initialized
successfully. */
    if (aos_status_is_ok(resp_status)) {
        printf("Init multipart upload succeeded, upload_id:%.*s\n",
               upload_id.len, upload_id.data);
    } else {
        printf("Init multipart upload failed, upload_id:%.*s\n",
               upload_id.len, upload_id.data);
    }
    /* Cancel the multipart upload. */
    resp_status = oss_abort_multipart_upload(oss_client_options, &
bucket, &object, &upload_id, &resp_headers);
    /* Determine whether the multipart upload is canceled successfully
. */
    if (aos_status_is_ok(resp_status)) {
        printf("Abort multipart upload succeeded, upload_id::%.*s\n",
               upload_id.len, upload_id.data);
    } else {
        printf("Abort multipart upload failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Note:

If you cancel a multipart upload event, you are not allowed to perform any other operations with this upload_id anymore. The uploaded parts will be deleted.

7.6.6 Upload progress bars

You can use progress bars to indicate the upload or download progress.

For the complete code of upload progress bar, see *GitHub*.

The following code is used as an example to describe how to view progress information with PutObject:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
```

```
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
void percentage(int64_t consumed_bytes, int64_t total_bytes)
{
    assert(total_bytes >= consumed_bytes);
    printf("%%%" APR_INT64_T_FMT "\n", consumed_bytes * 100 /
total_bytes);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_list_t resp_body;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    /* Upload with a progress bar displayed. */
    resp_status = oss_do_put_object_from_file(oss_client_options,
 &bucket, &object, &file, NULL, NULL, percentage, &resp_headers, &
resp_body);
    if (aos_status_is_ok(resp_status)) {
        printf("put object from file succeeded\n");
    } else {
        printf("put object from file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
```

```
    return 0;
}
```

oss_put_object_from_file and oss_put_object_from_buffer do not support progress bars.

7.6.7 Upload callback

Run the following code for upload callback:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *object_content = "More than just cloud.";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    aos_pool_t *p = NULL;
    aos_status_t *s = NULL;
    aos_string_t bucket;
    aos_string_t object;
    aos_table_t *headers = NULL;
    oss_request_options_t *options = NULL;
    aos_table_t *resp_headers = NULL;
    aos_list_t resp_body;
    aos_list_t buffer;
    aos_buf_t *content;
    char *buf = NULL;
    int64_t len = 0;
    int64_t size = 0;
    int64_t pos = 0;
    char b64_buf[1024];
    int b64_len;
    /* Use the JSON format. */
    char *callback =  "{"
        "\"callbackUrl\":\"http://oss-demo.aliyuncs.com:23450\","
        "\"callbackHost\":\"oss-cn-hangzhou.aliyuncs.com\","
        "\"callbackBody\":\"bucket=${bucket}&object=${object}&size=${
size}&mimeType=${mimeType}\","
        "\"callbackBodyType\":\"application/x-www-form-urlencoded\""
        "}";
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
```

```
        exit(1);
    }
    /* Initialize parameters. */
    aos_pool_create(&p, NULL);
    options = oss_request_options_create(p);
    init_options(options);
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_list_init(&resp_body);
    aos_list_init(&buffer);
    content = aos_buf_pack(options->pool, object_content, strlen(
object_content));
    aos_list_add_tail(&content->node, &buffer);
    /* Add callback into the header. */
    b64_len = aos_base64_encode((unsigned char*)callback, strlen(
callback), b64_buf);
    b64_buf[b64_len] = '\0';
    headers = aos_table_make(p, 1);
    apr_table_set(headers, OSS_CALLBACK, b64_buf);
    /* Start upload callback. */
    s = oss_do_put_object_from_buffer(options, &bucket, &object, &
buffer,
        headers, NULL, NULL, &resp_headers, &resp_body);
    if (aos_status_is_ok(s)) {
        printf("put object from buffer succeeded\n");
    } else {
        printf("put object from buffer failed\n");
    }
    /* Obtain the buffer length. */
    len = aos_buf_list_len(&resp_body);
    buf = (char *)aos_pcalloc(p, (apr_size_t)(len + 1));
    buf[len] = '\0';
    /* Copy the content in the buffer to the memory. */
    aos_list_for_each_entry(aos_buf_t, content, &resp_body, node) {
        size = aos_buf_size(content);
        memcpy(buf + pos, content->pos, (size_t)size);
        pos += size;
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(p);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

For more information about upload callback, see *Upload callback*. **For detailed debugging methods and troubleshooting information, see** *Upload callback error and exclusion*.

7.7 Download objects

7.7.1 Overview

For the complete code of object download, see *GitHub*.

OSS C SDK provides the following download methods:

- *Download to your local file*
- *Download to local memory*
- *Range download*
- *Resumable download*

You can view the download progress shown in the *download progress bar*.

7.7.2 Download an object to a local file

For the complete code of downloading OSS object, see *GitHub*.

You can run the following code to download the specified OSS object to a local file.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
```

```
    aos_string_t object;
    aos_string_t file;
    aos_table_t *params;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    params = aos_table_make(pool, 0);
    /* Download the object. A new local file is created if the
specified file does not exist. If the specified local file exists, it
is overwritten. */
    resp_status = oss_get_object_to_file(oss_client_options, &bucket,
 &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("Get object from file succeeded\n");
    } else {
        printf("Get object from file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Note:

Compared to the 1.0.0 version, the params parameter is added to the oss_get_object_to_file interface, and the value of the headers and params parameters can be NULL.

7.7.3 Download to your local memory

For the complete code of object download, see *GitHub*.

Run the following code to download a specified object to local memory:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
```

```
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_list_t buffer;
    aos_buf_t *content = NULL;
    aos_table_t *params = NULL;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    char *buf = NULL;
    int64_t len = 0;
    int64_t size = 0;
    int64_t pos = 0;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_list_init(&buffer);
    /* Download an object to local memory. */
    resp_status = oss_get_object_to_buffer(oss_client_options, &bucket
, &object,
                                    headers, params, &buffer, &resp_heade
rs);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to buffer succeeded\n");
        /* Copy the downloaded content to the buffer. */
        len = aos_buf_list_len(&buffer);
        buf = aos_pcalloc(pool, len + 1);
        buf[len] = '\0';
        aos_list_for_each_entry(aos_buf_t, content, &buffer, node) {
        size = aos_buf_size(content);
            memcpy(buf + pos, content->pos, size);
            pos += size;
        }
    }
    else {
        printf("get object to buffer failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
```

```
    return 0;
}
```

7.7.4 Range download

If you only need part of the data in an object, you can use range downloads to download specified content.

Run the following code for range download:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_list_t buffer;
    aos_buf_t *content = NULL;
    aos_table_t *params = NULL;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
```

```
    char *buf = NULL;
    int64_t len = 0;
    int64_t size = 0;
    int64_t pos = 0;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_list_init(&buffer);
    headers = aos_table_make(pool, 1);
    /* Set the download range, which ranges from the 20th byte to the
100th byte. */
    apr_table_set(headers, "Range", "bytes=20-100");
    /* Download the object to memory. */
    resp_status = oss_get_object_to_buffer(oss_client_options, &bucket
, &object, headers, params, &buffer, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to buffer succeeded\n");
        /* Obtain the buffer length. */
        aos_list_for_each_entry(aos_buf_t, content, &buffer, node) {
            len += aos_buf_size(content);
        }
        buf = aos_pcalloc(pool, (apr_size_t)(len + 1));
        buf[len] = '\0';
        /* Copy the content in the buffer to memory. */
        aos_list_for_each_entry(aos_buf_t, content, &buffer, node) {
            size = aos_buf_size(content);
            memcpy(buf + pos, content->pos, (size_t)size);
            pos += size;
        }
    }
    else {
        printf("get object to buffer failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.7.5 Resumable download

Large-sized objects may fail to be downloaded because of unstable network or other exceptions, and the entire object needs to be downloaded again. However, the object may still fail to be downloaded after multiple attempts. Therefore, OSS provides resumable download.

To enable resumable download, you need to split the object you want to download into multiple parts and download them separately. After you have downloaded all parts, these parts will be combined into a complete object.

You can use `oss_resumable_clt_params_t` to enable resumable download.

`oss_resumable_clt_params_t` contains the following parameters:

Parameter	Description
thread_num	The number of concurrent threads, which is 1 by default.
enable_checkpoint	Determines whether to enable resumable download, which is disabled by default.
partSize	Specifies the size of a part. The value can be between 1 byte to 5 GB and the unit is byte.
checkpoint_path	The path where the checkpoint file is located, which is under {upload_file_path}.cp by default.

This `checkpoint` file stores information about the download progress.If you fail to download a part, the download will continue based on the progress recorded in the `checkpoint` file. After the object is downloaded, the `checkpoint` file will be deleted.

Run the following code for resumable download:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
```

```
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_resumable_clt_params_t *clt_params;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    /* Download the object using resumable download. */
    clt_params = oss_create_resumable_clt_params_content(pool, 1024 *
100, 3, AOS_TRUE, NULL);
    resp_status = oss_resumable_download_file(oss_client_options, &
bucket, &object, &file, headers, NULL, clt_params, NULL, &resp_headers
);
    if (aos_status_is_ok(resp_status)) {
        printf("download succeeded\n");
    } else {
        printf("download failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.7.6 Download progress bars

You can use progress bars to indicate the upload or download progress.

For the complete code of download progress bar, see *GitHub*.

The following code is used as an example to describe how to use the download progress bar.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
const char *download_filename = "<yourDownloadFilename>";
void init_options(oss_request_options_t *options)
{
```

```
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
void percentage(int64_t consumed_bytes, int64_t total_bytes)
{
    assert(total_bytes >= consumed_bytes);
    printf("%%%" APR_INT64_T_FMT "\n", consumed_bytes * 100 /
total_bytes);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *resp_headers = NULL;
    aos_list_t resp_body;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    /* Upload with a progress bar displayed. */
    resp_status = oss_do_put_object_from_file(oss_client_options,
 &bucket, &object, &file, NULL, NULL, percentage, &resp_headers, &
resp_body);
    if (aos_status_is_ok(resp_status)) {
        printf("put object from file succeeded\n");
    } else {
        printf("put object from file failed\n");
    }
    /* Download with a progress bar displayed. */
    aos_pool_create(&pool, NULL);
    oss_client_options = oss_request_options_create(pool);
    init_options(oss_client_options);
    aos_str_set(&file, download_filename);
```

```
    resp_status = oss_do_get_object_to_file(oss_client_options, &
bucket, &object, NULL, NULL, &file, percentage, NULL);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

oss_get_object_from_file and oss_get_object_from_buffer do not support progress bars.

7.8 Manage objects

7.8.1 Overview

You can perform the following operations on objects in a bucket with APIs:

- *Manage ACL for an object*
- *Manage Object Meta*
- *List objects*
- *Copy objects*
- *Delete objects*
- *Restore an archive object*
- *Manage a symbolic link*

7.8.2 Manage ACL for an object

The following table describes the permissions included in the Access Control List (ACL) for an object.

Permission	Description	Value
default	The ACL of an object is the same with that of its bucket.	OSS_ACL_DEFAULT
Private	Only the object owner and authorized users can read and write the object.	OSS_ACL_PRIVATE

Permission	Description	Value
Public read	Only the object owner and authorized users can read and write the object. Other users can only read the object. Authorize this permission with caution.	OSS_ACL_PUBLIC_READ
Public read-write	All users can read and write the object. Authorize this permission with caution.	OSS_ACL_PUBLIC_READ_WRITE

The ACL of objects take precedence over that of buckets. For example, if the ACL of a bucket is private, while the object ACL is public read-write, all users can read and write the object. If an object is not configured with an ACL, its ACL is the same as that of its bucket by default.

Configure an ACL for an object

You can run the following code to configure an ACL for an object:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
```

```
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    oss_acl_e oss_acl = OSS_ACL_PUBLIC_READ;
    /* Configure an ACL for an object. */
    resp_status = oss_put_object_acl(oss_client_options, &bucket, &
object, oss_acl, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put object acl success! \n");
    } else {
        printf("put object acl failed! \n");
    }
    /* Obtain the ACL for an object. */
    aos_string_t oss_acl_string;
    resp_status = oss_get_object_acl(oss_client_options, &bucket, &
object, &oss_acl_string, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object acl success! \n");
        printf("acl: %s \n", oss_acl_string.data);
    } else {
        printf("get object acl failed! \n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.8.3 Manage Object Meta

Object Meta includes HTTP headers and user-defined Object Meta. For more information, see *Object Meta* in OSS Developer Guide.

Obtain Object Meta

After uploading files to OSS or before reading objects from OSS, you can use the `oss_get_object_meta` interface to obtain some of meta information about the objects, such as the length and type. Run the following code to configure and obtain Object Meta:

```
#include "oss_api.h"
```

```
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. The value 0 indicates that
 the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t.
The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes global
configuration information, such as endpoint, access_key_id, acces_key_
secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_table_t *headers;
    aos_list_t buffer;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_buf_t *content = NULL;
    char *content_length_str = NULL;
    char *object_type = NULL;
    char *object_author = NULL;
    int64_t content_length = 0;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    headers = aos_table_make(pool, 2);
    // Configure custom Object Meta. */
    apr_table_set(headers, "Expires", "Fri, 28 Feb 2032 05:38:42 GMT
");
    apr_table_set(headers, "x-oss-meta-author", "oss");
    aos_list_init(&buffer);
    content = aos_buf_pack(oss_client_options->pool, object_content,
strlen(object_content));
```

```
    aos_list_add_tail(&content->node, &buffer);
    /* Set object permissions and upload the object from the cache. */
    resp_status = oss_put_object_from_buffer(oss_client_options, &
bucket, &object,
                &buffer, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put object from buffer with md5 succeeded\n");
    } else {
        printf("put object from buffer with md5 failed\n");
    }
    /* Obtain the object permission. */
    resp_status = oss_get_object_meta(oss_client_options, &bucket, &
object, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        content_length_str = (char*)apr_table_get(resp_headers,
OSS_CONTENT_LENGTH);
        if (content_length_str ! = NULL) {
            content_length = atol(content_length_str);
        }
        object_author = (char*)apr_table_get(resp_headers, OSS_AUTHOR
IZATION);
        object_type = (char*)apr_table_get(resp_headers, OSS_OBJECT
_TYPE);
        printf("get object meta succeeded, object author:%s, object
type:%s, content_length:%ld\n", object_author, object_type, content_le
ngth);
    } else {
        printf("req:%s, get object meta failed\n", resp_status->req_id
);
    }
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Note:

For detailed information about HTTP header, see *RFC2616*.

7.8.4 List objects

For the complete code for listing objects, see *GitHub*.

OSS objects are listed alphabetically. You can use `oss_list_object` to list all objects in a bucket. The following table describes common parameters used to list objects.

Parameter	Description
delimiter	Specifies a delimiter used to group objects. All those objects whose names contain the specified prefix and after which the delimiter occurs for the first time, act as a group of elements, that is, commonPrefixes.
prefix	Specifies the prefix that the listed objects must contain.
max_ret	Specifies the maximum number of objects that can be listed. The default value is 1000.
marker	Specifies the initial object that is listed.

List all objects

You can run the following code to list all objects in the current bucket.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
```

```
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_status_t *resp_status = NULL;
    oss_list_object_params_t *params = NULL;
    oss_list_object_content_t *content = NULL;
    int size = 0;
    char *line = NULL;
    char *prefix = "";
    char *nextMarker = "";
    aos_str_set(&bucket, bucket_name);
    params = oss_create_list_object_params(pool);
    params->max_ret = 100;
    aos_str_set(&params->prefix, prefix);
    aos_str_set(&params->marker, nextMarker);
    printf("Object\tSize\tLastModified\n");
    /* List all objects. */
    do {
        resp_status = oss_list_object(oss_client_options, &bucket,
params, NULL);
        if (! aos_status_is_ok(resp_status))
        {
            printf("list object failed\n");
            break;
        }
        aos_list_for_each_entry(oss_list_object_content_t, content, &
params->object_list, node) {
            ++size;
            line = apr_psprintf(pool, "%.*s\t%.*s\t%.*s\n", content->
key.len, content->key.data,
                content->size.len, content->size.data,
                content->last_modified.len, content->last_modified.
data);
            printf("%s", line);
        }
        nextMarker = apr_psprintf(pool, "%.*s", params->next_marker.
len, params->next_marker.data);
        aos_str_set(&params->marker, nextMarker);
        aos_list_init(&params->object_list);
        aos_list_init(&params->common_prefix_list);
    } while (params->truncated == AOS_TRUE);
    printf("Total %d\n", size);
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Note:

By default, if a bucket contains more than 1,000 objects, only the first 1,000 objects are returned and the truncated parameter in the returned results is true. Additonally, the next_marker is returned to serve as the start point of next read. To

adjust the number of returned objects, you can modify the max_ret parameter or use the marker parameter for multiple reads.

List specified number of objects

You can run the following code to list specified number of objects.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* aos_str_set uses a char* string to initialize the aos_string_t
type.*/
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used.*/
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout.*/
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories.*/
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pool. */ */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Memories for options are allocated by the memory pool and are
released after the memory pool is released. Therefore, you do not need
 to release the memory for options individually. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_status_t *resp_status = NULL;
    oss_list_object_params_t *params = NULL;
    oss_list_object_content_t *content = NULL;
    int size = 0;
    char *line = NULL;
    char *prefix = "";
    char *nextMarker = "";
    aos_str_set(&bucket, bucket_name);
    params = oss_create_list_object_params(pool);
```

```
    /* Set the maximum number of listed objects to 200. */
    params->max_ret = 200;
    aos_str_set(&params->prefix, prefix);
    aos_str_set(&params->marker, nextMarker);
    printf("Object\tSize\tLastModified\n");
    /* List all objects */
    resp_status = oss_list_object(oss_client_options, &bucket, params
, NULL);
    if (! aos_status_is_ok(resp_status))
    {
        printf("list object failed\n");
    }
    aos_list_for_each_entry(oss_list_object_content_t, content, &
params->object_list, node) {
        ++size;
        line = apr_psprintf(pool, "%.*s\t%.*s\t%.*s\n", content->key.
len, content->key.data,
            content->size.len, content->size.data,
            content->last_modified.len, content->last_modified.data);
        printf("%s", line);
    }
    printf("Total %d\n", size);
    /* Release the memory pool after a request is executed, which
means that memories allocated for all resources during the request are
 released. */
    aos_pool_destroy(pool);
    /* Before the project ends, call the aos_http_io_deinitialize
method to release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

List objects with specified prefix

You can run the following code to list objects with specified prefix.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* aos_str_set uses a char* string to initialize the aos_string_t
data type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, including networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
```

```
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
 */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Memories for options are allocated by the memory pool and are
released after the memory pool is released. Therefore, you do not need
 to release the memory for options individually. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_status_t *resp_status = NULL;
    oss_list_object_params_t *params = NULL;
    oss_list_object_content_t *content = NULL;
    int size = 0;
    char *line = NULL;
    /* List objects with specified prefix. */
    char *prefix = "<yourObjectPefix>";
    char *nextMarker = "";
    aos_str_set(&bucket, bucket_name);
    params = oss_create_list_object_params(pool);
    params->max_ret = 100;
    aos_str_set(&params->prefix, prefix);
    aos_str_set(&params->marker, nextMarker);
    printf("Object\tSize\tLastModified\n");
    /* List all objects. */
    do {
        resp_status = oss_list_object(oss_client_options, &bucket,
params, NULL);
        if (! aos_status_is_ok(resp_status))
        {
            printf("list object failed\n");
            break;
        }
        aos_list_for_each_entry(oss_list_object_content_t, content, &
params->object_list, node) {
            ++size;
            line = apr_psprintf(pool, "%.*s\t%.*s\t%.*s\n", content->
key.len, content->key.data,
                content->size.len, content->size.data,
                content->last_modified.len, content->last_modified.
data);
            printf("%s", line);
        }
        nextMarker = apr_psprintf(pool, "%.*s", params->next_marker.
len, params->next_marker.data);
        aos_str_set(&params->marker, nextMarker);
        aos_list_init(&params->object_list);
        aos_list_init(&params->common_prefix_list);
    } while (params->truncated == AOS_TRUE);
    printf("Total %d\n", size);
    /* Release the memory pool after a request is executed, which
means that memories allocated for all resources during the request are
 released. */
    aos_pool_destroy(pool);
```

```
    /* Before the project ends, call the aos_http_io_deinitialize
method to release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Specifies the initial object that is listed.

The value of marker is the name of the initial object. You can run the following code to list objects after marker in alphabetic order.

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* aos_str_set uses a char* string to initialize the aos_string_t
type.*/
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used.*/
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout.*/
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as network and memories.*/
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Memories for options are allocated by the memory pool and are
released after the memory pool is released. Therefore, you do not need
 to release the memory for options individually. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters */
    aos_string_t bucket;
    aos_status_t *resp_status = NULL;
    oss_list_object_params_t *params = NULL;
    oss_list_object_content_t *content = NULL;
    int size = 0;
    char *line = NULL;
    char *prefix = "";
```

```
    Specifies the initial object that is listed. */
    char *nextMarker = "<yourObjectMarker>";
    aos_str_set(&bucket, bucket_name);
    params = oss_create_list_object_params(pool);
    params->max_ret = 100;
    aos_str_set(&params->prefix, prefix);
    aos_str_set(&params->marker, nextMarker);
    printf("Object\tSize\tLastModified\n");
    /* List all objects */
    do {
        resp_status = oss_list_object(oss_client_options, &bucket,
params, NULL);
        if (! aos_status_is_ok(resp_status))
        {
            printf("list object failed\n");
            break;
        }
        aos_list_for_each_entry(oss_list_object_content_t, content, &
params->object_list, node) {
            ++size;
            line = apr_psprintf(pool, "%.*s\t%.*s\t%.*s\n", content->
key.len, content->key.data,
                content->size.len, content->size.data,
                content->last_modified.len, content->last_modified.
data);
            printf("%s", line);
        }
        nextMarker = apr_psprintf(pool, "%.*s", params->next_marker.
len, params->next_marker.data);
        aos_str_set(&params->marker, nextMarker);
        aos_list_init(&params->object_list);
        aos_list_init(&params->common_prefix_list);
    } while (params->truncated == AOS_TRUE);
    printf("Total %d\n", size);
    /* Release the memory pool after a request is executed, which
means that memories allocated for all resources during the request are
 released. */
    aos_pool_destroy(pool);
    /* Before the project ends, call the aos_http_io_deinitialize
method to release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.8.5 Delete objects

Warning:

Delete objects with caution because deleted objects cannot be recovered.

For the complete code of deleting objects, see *GitHub*.

Delete an object

Run the following code to delete a single object:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
```

```
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    /* Assign char* data to the aos_string_t bucket. */
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    /* Delete an object. */
    resp_status = oss_delete_object(oss_client_options, &bucket, &
object, &resp_headers);
    /* Determine whether the object is deleted successfully. */
    if (aos_status_is_ok(resp_status)) {
        printf("delete object succeed\n");
    } else {
        printf("delete object failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

Delete multiple objects

Run the following code to delete multiple objects simultaneously:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name1 = "<yourObjectName1>";
const char *object_name2 = "<yourObjectName2>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object1;
    aos_string_t object2;
    int is_quiet = 1;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object1, object_name1);
    aos_str_set(&object2, object_name2);
    /* Construct the list of objects to be deleted. */
    aos_list_t object_list;
    aos_list_t deleted_object_list;
```

```
    oss_object_key_t *content1;
    oss_object_key_t *content2;
    aos_list_init(&object_list);
    aos_list_init(&deleted_object_list);
    content1 = oss_create_oss_object_key(pool);
    aos_str_set(&content1->key, object_name1);
    aos_list_add_tail(&content1->node, &object_list);
    content2 = oss_create_oss_object_key(pool);
    aos_str_set(&content2->key, object_name2);
    aos_list_add_tail(&content2->node, &object_list);
    /* Delete objects in the list. `is_quiet` is used to determine
whether the result of deletion is returned. */
    resp_status = oss_delete_objects(oss_client_options, &bucket, &
object_list, is_quiet, &resp_headers, &deleted_object_list);
    if (aos_status_is_ok(resp_status)) {
        printf("delete objects succeeded\n");
    } else {
        printf("delete objects failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Delete multiple objects with a specified prefix

You can delete a directory by deleting multiple objects with a specified prefix. For example, you can delete the dir directory by deleting objects with the prefix `dir/`. Run the following code to delete multiple objects with a specified prefix simultaneously:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_prefix = "<yourObjectNamePrefix>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
```

```
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t prefix;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&prefix, object_prefix);
    /* Delete objects with a specified prefix. */
    resp_status = oss_delete_objects_by_prefix(oss_client_options, &
bucket, &prefix);
    if (aos_status_is_ok(resp_status)) {
        printf("delete objects by prefix succeeded\n");
    } else {
        printf("delete objects by prefix failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Warning:

If the prefix is an empty string or null, all objects in the bucket will be deleted. Perform this operation with caution.

7.8.6 Copy objects

Copy a small-sized object

You must note the following limits before copy a small-sized object:

- **The user must have the read and write permission on the source object.**
- **You cannot copy an object from a region to another region. For example, you cannot copy an object from a bucket in Hangzhou region to a bucket in Qingdao.**
- **The object size must be smaller than 1 GB.**

Run the following code to copy a small-sized object:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *source_bucket_name = "<yourSourceBucketName>";
const char *source_object_name = "<yourSourceObjectName>";
const char *dest_bucket_name = "<yourDestBucketName>";
const char *dest_object_name = "<yourDestObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t source_bucket;
    aos_string_t source_object;
    aos_string_t dest_bucket;
    aos_string_t dest_object;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&source_bucket, source_bucket_name);
    aos_str_set(&source_object, source_object_name);
    aos_str_set(&dest_bucket, dest_bucket_name);
    aos_str_set(&dest_object, dest_object_name);
    headers = aos_table_make(pool, 0);
    /* Copy an object. */
```

```
    resp_status = oss_copy_object(oss_client_options, &source_bucket,
 &source_object, &dest_bucket, &dest_object, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("copy object succeeded\n");
    } else {
        printf("copy object failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Copy a large-sized object

For objects larger than 1 GB, we recommend that you should use `oss_upload_part_copy` to perform multipart copy (UploadPartCopy). Run the following code for multipart copy:

```
#include "oss_api.h"
#include "aos_http_io.h"
#include <sys/stat.h>
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *source_bucket_name = "<yourSourceBucketName>";
const char *source_object_name = "<yourSourceObjectName>";
const char *dest_bucket_name = "<yourDestBucketName>";
const char *dest_object_name = "<yourDestObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int64_t get_file_size(const char *file_path)
{
    int64_t filesize = -1;
    struct stat statbuff;
    if(stat(file_path, &statbuff) < 0){
        return filesize;
    } else {
        filesize = statbuff.st_size;
    }
    return filesize;
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
```

```
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t dest_bucket;
    aos_string_t dest_object;
    aos_string_t upload_id;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    oss_list_upload_part_params_t *list_upload_part_params;
    oss_upload_part_copy_params_t *upload_part_copy_params1;
    oss_upload_part_copy_params_t *upload_part_copy_params2;
    oss_list_part_content_t *part_content;
    aos_list_t complete_part_list;
    oss_complete_part_content_t *complete_content;
    aos_table_t *list_part_resp_headers = NULL;
    aos_table_t *complete_resp_headers = NULL;
    int part1 = 1;
    int part2 = 2;
    int64_t range_start1 = 0;
    int64_t range_end1 = 6000000;//not less than 5MB
    int64_t range_start2 = 6000001;
    int64_t range_end2;
    int max_ret = 1000;
    headers = aos_table_make(pool, 0);
    aos_str_set(&dest_bucket, dest_bucket_name);
    aos_str_set(&dest_object, dest_object_name);
    resp_status = oss_init_multipart_upload(oss_client_options, &
dest_bucket, &dest_object, &upload_id, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("init multipart upload succeeded, upload_id is %s\n",
upload_id.data);
    } else {
        printf("init multipart upload failed\n");
    }
    /* Copy the first part of the object. */
    upload_part_copy_params1 = oss_create_upload_part_copy_params(pool
);
    aos_str_set(&upload_part_copy_params1->source_bucket, source_buc
ket_name);
    aos_str_set(&upload_part_copy_params1->source_object, source_obj
ect_name);
    aos_str_set(&upload_part_copy_params1->dest_bucket, dest_bucke
t_name);
    aos_str_set(&upload_part_copy_params1->dest_object, dest_objec
t_name);
    aos_str_set(&upload_part_copy_params1->upload_id, upload_id.data);
    upload_part_copy_params1->part_num = part1;
    upload_part_copy_params1->range_start = range_start1;
```

```
    upload_part_copy_params1->range_end = range_end1;
    headers = aos_table_make(pool, 0);
    resp_status = oss_upload_part_copy(oss_client_options, upload_par
t_copy_params1, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("upload part 1 copy succeeded\n");
    } else {
        printf("upload part 1 copy failed\n");
    }
    /* Copy the second part of the object. */
    range_end2 = get_file_size(local_filename) - 1;
    upload_part_copy_params2 = oss_create_upload_part_copy_params(pool
);
    aos_str_set(&upload_part_copy_params2->source_bucket, source_buc
ket_name);
    aos_str_set(&upload_part_copy_params2->source_object, source_obj
ect_name);
    aos_str_set(&upload_part_copy_params2->dest_bucket, dest_bucke
t_name);
    aos_str_set(&upload_part_copy_params2->dest_object, dest_objec
t_name);
    aos_str_set(&upload_part_copy_params2->upload_id, upload_id.data);
    upload_part_copy_params2->part_num = part2;
    upload_part_copy_params2->range_start = range_start2;
    upload_part_copy_params2->range_end = range_end2;
    headers = aos_table_make(pool, 0);
    resp_status = oss_upload_part_copy(oss_client_options, upload_par
t_copy_params2, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("upload part 2 copy succeeded\n");
    } else {
        printf("upload part 2 copy failed\n");
    }
    /* Lists the parts. */
    headers = aos_table_make(pool, 0);
    list_upload_part_params = oss_create_list_upload_part_params(pool
);
    list_upload_part_params->max_ret = max_ret;
    aos_list_init(&complete_part_list);
    resp_status = oss_list_upload_part(oss_client_options, &dest_bucke
t, &dest_object, &upload_id, list_upload_part_params, &list_part_
resp_headers);
    aos_list_for_each_entry(oss_list_part_content_t, part_content, &
list_upload_part_params->part_list, node) {
       complete_content = oss_create_complete_part_content(pool);
       aos_str_set(&complete_content->part_number, part_content->
part_number.data);
       aos_str_set(&complete_content->etag, part_content->etag.data);
       aos_list_add_tail(&complete_content->node, &complete_part_list
);
    }
    /* Complete the multipart copy. */
    resp_status = oss_complete_multipart_upload(oss_client_options, &
dest_bucket, &dest_object, &upload_id, &complete_part_list, headers, &
complete_resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("complete multipart upload succeeded\n");
    } else {
        printf("complete multipart upload failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
```

```
    aos_http_io_deinitialize();
    return 0;
}
```

7.8.7 Restore an archive object

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

- **An archive object is in the frozen state.**
- **After you submit it for restoration, the server restores the object. The object is in the restoring state.**
- **You can read the object after it is restored. The restored state of the object lasts one day by default. You can prolong this period to a maximum of seven days. Once this period expires, the object returns to the frozen state.**

Run the following code to restore an object:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *object_content = "More than just cloud.";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
```

```
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_list_t buffer;
    oss_acl_e oss_acl = OSS_ACL_PRIVATE;
    aos_buf_t *content = NULL;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    headers = aos_table_make(pool, 0);
    /* Create a bucket of the archive storage class. */
    resp_status = oss_create_bucket_with_storage_class(oss_client
_options, &bucket, oss_acl, OSS_STORAGE_CLASS_ARCHIVE, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("create bucket succeeded\n");
    } else {
        printf("create bucket failed\n");
    }
    aos_list_init(&buffer);
    content = aos_buf_pack(oss_client_options->pool, object_content,
strlen(object_content));
    aos_list_add_tail(&content->node, &buffer);
    /* Upload a file. */
    resp_status = oss_put_object_from_buffer(oss_client_options, &
bucket, &object, &buffer, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put object from buffer succeeded\n");
    } else {
        printf("put object from buffer failed\n");
    }
    /* Restore the object. */
    do {
        headers = aos_table_make(pool, 0);
        resp_status = oss_restore_object(oss_client_options, &bucket,
 &object, headers, &resp_headers);
        printf("restore object resp_status->code: %d \n", resp_status-
>code);
        if (resp_status->code ! = 409) {
            break;
        } else {
            printf("restore object is already in progress, resp_status
->code: %d \n", resp_status->code);
            apr_sleep(5000);
        }
    } while (1);
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

For more information about the archive storage classes, see *Introduction to storage classes*. For more information about related status code, see *RestoreObject*.

7.8.8 Manage a symbolic link

Create a symbolic link

A symbolic link is a special object that maps to an object similar to a shortcut used in Windows. You can configure user-defined object meta for symbolic links.

Run the following code to create a symbolic link:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *link_object_name = "<yourLinkObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
```

```
    aos_string_t object;
    aos_string_t sym_object;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&sym_object, link_object_name);
    resp_status = oss_put_symlink(oss_client_options, &bucket, &
sym_object, &object, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put symlink succeeded\n");
    } else {
        printf("put symlink failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

Obtain object content for a symbolic link

To obtain a symbolic link, you must have the read permission on it. Run the following code to obtain the object content for a symbolic link:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *link_object_name = "<yourLinkObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key_secret
);
    /* Determine whether the CNAME is used. 0 indicates that the CNAME
 is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
```

```
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t link_object;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    char *target_object_name = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&link_object, link_object_name);
    resp_status = oss_get_symlink(oss_client_options, &bucket, &
link_object, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get symlink succeeded\n");
        target_object_name = (char*)(apr_table_get(resp_headers,
OSS_CANNONICALIZED_HEADER_SYMLINK));
        printf("target_object_name: %s \n", target_object_name);
    } else {
        printf("get symlink failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.9 Authorized access

Use STS for temporary access authorization

OSS supports Alibaba Cloud Security Token Service (STS) for temporary access authorization. STS is a web service that provides a temporary access token to a cloud computing user. Through the STS, you can assign a third-party application or a RAM user (you can manage the user ID) an access credential with a custom validity period and permissions. For more information about STS, see *STS introduction*.

STS advantages:

- **Your long-term key (AccessKey) is not exposed to a third-party application. You only need to generate an access token and send the access token to the third-party application. You can customize access permissions and the validity of this token.**
- **You do not need to keep track of permission revocation issues. The access token automatically becomes invalid when it expires.**

For more information about the process of access to OSS with STS, see *RAM and STS scenario practices* in OSS Developer Guide.

Create a signature request with STS

Run the following code to create a signature request with STS:

```
#include "oss_api.h"
#include "aos_http_io.h"

const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *sts_token = "<yourStsToken>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *object_content = "More than just cloud.";

void init_options(oss_request_options_t *options)
{
 options->config = oss_config_create(options->pool);
 /* Use a char* string to initialize the aos_string_t type. */
 aos_str_set(&options->config->endpoint, endpoint);
 aos_str_set(&options->config->access_key_id, access_key_id);
 aos_str_set(&options->config->access_key_secret, access_key_secret);
 aos_str_set(&options->config->sts_token, sts_token);
 /* Determine whether the CNAME is used. 0 indicates that the CNAME is
 not used. */
 options->config->is_cname = 0;
 /* Used to configure network parameters, such as timeout. */
 options->ctl = aos_http_controller_create(options->pool, 0);
}

int main(int argc, char *argv[])
{
 /* Call the aos_http_io_initialize method in main() to initialize
global resources, such as networks and memories. */
 if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
  exit(1);
 }

 /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library. */
 aos_pool_t *pool;
 /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
 aos_pool_create(&pool, NULL);
 /* Create and initialize options. This parameter mainly includes
global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
 oss_request_options_t *oss_client_options;
 /* Allocate memories in the memory pool to options. */
 oss_client_options = oss_request_options_create(pool);
 /* Use oss_client_options to initialize client options. */
 init_options(oss_client_options);

    /*  Initialization parameters. */
 aos_string_t bucket;
 aos_string_t object;
 aos_list_t buffer;
 aos_buf_t *content = NULL;
```

```
    aos_table_t *headers = NULL;
 aos_table_t *resp_headers = NULL;
 aos_status_t *resp_status = NULL;
 /* Assign the char* data to the bucket. */
 aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);

    aos_list_init(&buffer);
    content = aos_buf_pack(oss_client_options->pool, object_content,
strlen(object_content));
    aos_list_add_tail(&content->node, &buffer);

 /* Upload an object. */
    resp_status = oss_put_object_from_buffer(oss_client_options, &
bucket, &object, &buffer, headers, &resp_headers);
    /* Determine whether the upload is successful. */
    if (aos_status_is_ok(resp_status)) {
        printf("put object from buffer succeeded\n");
    } else {
        printf("put object from buffer failed\n");
    }

 /* Release the memory pool, that is, memories allocated to resources
during the request. */
 aos_pool_destroy(pool);

 /* Release allocated global resources. */
 aos_http_io_deinitialize();

 return 0;
}
```

Sign a URL to authorize temporary access

You can provide a signed URL to a visitor for temporary access. When you sign a URL, you can specify the expiration time for a URL to restrict the period of access from visitors.

- **Use a signed URL to upload an object**

  Run the following code to upload an object with a signed URL:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key
_secret);
    /* Determine whether the CNAME is used. 0 indicates that the
CNAME is not used. */
```

```
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to
initialize global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library.
 */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /* Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_http_request_t *req;
    apr_time_t now;
    char *url_str;
    aos_string_t url;
    int64_t expire_time;
    int one_hour = 3600;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    expire_time = now / 1000000 + one_hour;
    headers = aos_table_make(pool, 0);
    req = aos_http_request_create(pool);
    req->method = HTTP_PUT;
    now = apr_time_now();  /* Unit: microseconds */
    expire_time = now / 1000000 + one_hour;
    /* Generate the signed URL. */
    url_str = oss_gen_signed_url(oss_client_options, &bucket, &
object, expire_time, req);
    aos_str_set(&url, url_str);
    printf("Temporary upload URL: %s\n", url_str);
    /* Use the signed URL to upload an object. */
    resp_status = oss_put_object_from_file_by_url(oss_client_options
, &url, &file, headers, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("put objects by signed url succeeded\n");
    } else {
        printf("put objects by signed url failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
```

```
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

- **Use a signed URL to download an object**

  Run the following code to upload an object with a signed URL:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
const char *local_filename = "<yourLocalFilename>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize the aos_string_t type. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key
_secret);
    /* Determine whether the CNAME is used. 0 indicates that the
CNAME is not used. */
    options->config->is_cname = 0;
    /* Used to configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to
initialize global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, which is equivalent to
apr_pool_t. The implementation code is included in the apr library.
 */
    aos_pool_t *pool;
    /* Re-create a new memory pool. The second parameter is NULL,
indicating that it does not inherit from any other memory pools. */
    aos_pool_create(&p, NULL);
    /* Create and initialize options. This parameter mainly includes
 global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Use oss_client_options to initialize client options. */
    init_options(oss_client_options);
    /*  Initialization parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *params = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_http_request_t *req;
    apr_time_t now;
```

```
    char *url_str;
    aos_string_t url;
    int64_t expire_time;
    int one_hour = 3600;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    aos_str_set(&file, local_filename);
    expire_time = now / 1000000 + one_hour;
    headers = aos_table_make(pool, 0);
    params = aos_table_make(pool, 0);
    req = aos_http_request_create(pool);
    req->method = HTTP_GET;
    now = apr_time_now();  /* Unit: microseconds */
    expire_time = now / 1000000 + one_hour;
    /* Generate the signed URL. */
    url_str = oss_gen_signed_url(oss_client_options, &bucket, &
object, expire_time, req);
    aos_str_set(&url, url_str);
    /* Use the signed URL to download an object. */
    resp_status = oss_get_object_to_file_by_url(oss_client_options,
 &url, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object succeeded\n");
    } else {
        printf("get object failed\n");
    }
    /* Release the memory pool, that is, memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

7.10 Image processing

Image Processing (IMG) is a massive, secure, cost-effective and highly reliable image processing service. After source images are uploaded to OSS, you can process images on any Internet device at any anytime, from anywhere through simple RESTful APIs.

For more information about IMG, see *Image Processing*.

For the complete code of IMG, see *GitHub*.

Basic features

OSS offers the following IMG features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*

- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**
- *Image processing*
- *Image processing access rules*: **calls multiple IMG functions.**

Usage

IMG uses standard HTTP GET. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is private read and write, only authorized users are allowed for access.

- **Anonymous access**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Specifies the name of a bucket.
endpoint	Specifies endpoint used to access a region.
object	Specifies the name of an image object.
image	Specifies the reserved identifier for IMG.
action	Specifies the operations on an image, such as scaling, cropping, and rotating .

Parameter	Description
param	Specifies the parameter that indicates the operation on an image.

- **Basic operations**

  For example, scale an image to a width of 100 px. Adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Customized image styles**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=style/<yourStyleName>
```

  - **style: specifies a reserved identifier of a customized image style.**
  - **yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.**

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=style/oss-pic-style-w-100
```

- **Cascade operations**

  Cascade operations allow you to perform multiple operations on an image sequentially.

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss
-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<
yourParamValue2>/...
```

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100/rotate,90
```

- **HTTPS access**

IMG supports access through HTTPS. Example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  Authorized access allows you to customize an image style, HTTPS access, and cascade operations.

  Use the following code to generate a signed URL for IMG:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key
_secret);
    /* Determine whether the CNAME is used. The value 0 indicates
that the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to
initialize global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to apr_pool_t
. The implementation code is included in the apr library. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory pool.
 */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes
global configuration information, such as endpoint, access_key_id,
acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_table_t *params = NULL;
    aos_http_request_t *req;
```

```
    char *url_str;
    apr_time_t now;
    int64_t expire_time;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    /* Start image processing. */
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/resize,m_fixed,w_100,
h_100");
    req = aos_http_request_create(pool);
    req->method = HTTP_GET;
    req->query_params = params;
    /* Specify the expiration time (expire_time) in seconds. */
    now = apr_time_now();
    expire_time = now / 1000000 + 10 * 60;
    /* Generate a signed URL. */
    url_str = oss_gen_signed_url(oss_client_options, &bucket, &
object, expire_time, req);
    printf("url: %s\n", url_str);
    /* Release the memory pool, that is, the memories allocated to
resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

- **Access with SDK**

  You can use SDK to access and process an image object.

  SDK allows you to customize image styles, HTTPS access, and cascade operations

  .

  - **Basic operations**

    Run the following code to perform basic operations on an image:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key
_secret);
    /* Determine whether the CNAME is used. The value 0 indicates
that the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
```

```
    /* Call the aos_http_io_initialize method in main() to
initialize global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to
apr_pool_t. The implementation code is included in the apr library
. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory
pool. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes
global configuration information, such as endpoint, access_key_id
, acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *params = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    /* Scale the image.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/resize,m_fixed,w_100
,h_100");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-resize.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Crop the image.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/crop,w_100,h_100,
x_100,y_100,r_1");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-crop.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Rotate the image.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/rotate,90");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-rotate.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
```

```
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Sharpen the image.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/sharpen,100");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-sharpen.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Add watermarks.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/watermark,text_SGVsb
G8g5Zu-54mH5pyN5YqhIQ");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-watermark.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Convert the image format.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/format,png");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-format.png");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Obtain image information.
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/info");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-info.txt");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
 resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
```

```
}
```

- **Customize an image style**

  Run the following code to customize an image style:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key
_secret);
    /* Determine whether the CNAME is used. The value 0 indicates
that the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to
initialize global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to
apr_pool_t. The implementation code is included in the apr library
. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory
pool. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes
global configuration information, such as endpoint, access_key_id
, acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *params = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    /* Customize the image style. */
    params = aos_table_make(pool, 1);
```

```
    apr_table_set(params, OSS_PROCESS, "style/<yourCustomStyleName
>");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-custom-style.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
 resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

- **Cascade operations**

  Run the following code to perform cascade operations on an image:

```
#include "oss_api.h"
#include "aos_http_io.h"
const char *endpoint = "<yourEndpoint>";
const char *access_key_id = "<yourAccessKeyId>";
const char *access_key_secret = "<yourAccessKeySecret>";
const char *bucket_name = "<yourBucketName>";
const char *object_name = "<yourObjectName>";
void init_options(oss_request_options_t *options)
{
    options->config = oss_config_create(options->pool);
    /* Use a char* string to initialize aos_string_t. */
    aos_str_set(&options->config->endpoint, endpoint);
    aos_str_set(&options->config->access_key_id, access_key_id);
    aos_str_set(&options->config->access_key_secret, access_key
_secret);
    /* Determine whether the CNAME is used. The value 0 indicates
that the CNAME is not used. */
    options->config->is_cname = 0;
    /* Configure network parameters, such as timeout. */
    options->ctl = aos_http_controller_create(options->pool, 0);
}
int main(int argc, char *argv[])
{
    /* Call the aos_http_io_initialize method in main() to
initialize global resources, such as networks and memories. */
    if (aos_http_io_initialize(NULL, 0) ! = AOSE_OK) {
        exit(1);
    }
    /* Memory pool used to manage memories, equivalent to
apr_pool_t. The implementation code is included in the apr library
. */
    aos_pool_t *pool;
    /* Re-create a memory pool. The second parameter is NULL,
indicating that the new pool does not inherit any other memory
pool. */
    aos_pool_create(&pool, NULL);
    /* Create and initialize options. This parameter includes
global configuration information, such as endpoint, access_key_id
, acces_key_secret, is_cname, and curl. */
    oss_request_options_t *oss_client_options;
```

```
    /* Allocate the memories in the memory pool to options. */
    oss_client_options = oss_request_options_create(pool);
    /* Initialize oss_client_options. */
    init_options(oss_client_options);
    /* Initialize parameters. */
    aos_string_t bucket;
    aos_string_t object;
    aos_string_t file;
    aos_table_t *headers = NULL;
    aos_table_t *params = NULL;
    aos_table_t *resp_headers = NULL;
    aos_status_t *resp_status = NULL;
    aos_str_set(&bucket, bucket_name);
    aos_str_set(&object, object_name);
    /* Perform cascade operations. */
    params = aos_table_make(pool, 1);
    apr_table_set(params, OSS_PROCESS, "image/resize,m_fixed,w_100
,h_100");
    /* Download processed images to a local file. */
    aos_str_set(&file, "example-cascade.jpg");
    resp_status = oss_get_object_to_file(oss_client_options, &
bucket, &object, headers, params, &file, &resp_headers);
    if (aos_status_is_ok(resp_status)) {
        printf("get object to file succeeded\n");
    } else {
        printf("get object to file failed\n");
    }
    /* Release the memory pool, that is, the memories allocated to
 resources during the request. */
    aos_pool_destroy(pool);
    /* Release the allocated global resources. */
    aos_http_io_deinitialize();
    return 0;
}
```

IMG tools

You can use the IMG viewer *ImageStyleViever* to directly view the IMG result.

7.11 Error handling

Exception handling

If a request error occurs in the use of OSS C SDK, the corresponding error message is output in aos_status_s. The attributes of aos_status_s include:

Error code	Description	Type
code	The HTTP status code of the error request	Integer
error_code	Error code returned by OSS	Character string
error_msg	Error message returned by OSS	Character string

Error code	Description	Type
req_id	The UUID that identifies the request. You can provide the req_id to OSS development engineers for further help.	Character string

Timeout handling

If the code in the returned aos_status_t is not “2XX”, and the error_code is -992 or -995, it indicates a connection timeout error or a request timeout error occurs. You can try again.

You can use the aos_should_retry(aos_status_t *) in the aos_status.h to determine whether the returned error code requires a retry. If the returned value is 1, you must try again.

Common error codes

Error code	Description	HTTP status code
AccessDenied	Access denied	403
BucketAlreadyExists	The bucket already exists.	409
BucketNotEmpty	The bucket is not empty.	409
EntityTooLarge	The entity size exceeds the maximum limit.	400
EntityTooSmall	The entity size is below the minimum limit.	400
FileGroupTooLarge	The total file group size exceeds the maximum limit.	400
FilePartNotExist	The file part does not exist .	400
FilePartStale	The file part has expired.	400
InvalidArgument	The parameter format is invalid.	400
InvalidAccessKeyId	The AccessKeyId does not exist.	403
InvalidBucketName	The bucket name is invalid.	400

Error code	Description	HTTP status code
InvalidDigest	The digest is invalid.	400
InvalidObjectName	The object name is invalid .	400
InvalidPart	The part is invalid.	400
InvalidPartOrder	The part order is invalid.	400
InvalidTargetBucketForLogging	The buckets that store log files are invalid.	400
InternalError	Internal OSS error	500
MalformedXML	The XML format is invalid.	400
MethodNotAllowed	The method is not supported.	405
MissingArgument	Parameters are not configured.	411
MissingContentLength	The content length is not configured.	411
NoSuchBucket	The bucket does not exist.	404
NoSuchKey	The object does not exist	404
NoSuchUpload	The multipart upload ID does not exist.	404
NotImplemented	The method cannot be processed.	501
PreconditionFailed	Pre-processing error.	412
RequestTimeTooSkewed	The local time set for OSSClient is deviated from the time set for the OSS server by over 15 minutes.	403
RequestTimeout	Request timeout	400
SignatureDoesNotMatch	Signature mismatch	403
TooManyBuckets	The number of buckets exceed the limits,.	400

7.12 FAQ

This topic lists the FAQs about OSS C SDK.

- *How to configure the output of application logs in OSS C SDK.*
- *How to configure parameters for cURL-based communication in OSS C SDK.*
- *How to upload data using callback of cURL in OSS C SDK.*
- *How to upload data using callback of cURL in OSS C SDK.*
- *How to download data using callback of cURL in OSS C SDK.*
- *How to upload and download objects with names containing Chinese characters in OSS C SDK (for Windows).*

8 . NET

8.1 Preface

The OSS C# SDK supports .NET Framework 2.0 and later. This document is written based on OSS C# SDK 2.8.0.

Compatibility

- **For SDK series of 2.x.x**
  - **API: compatible**
  - **Namespace: compatible**
- **For SDK series of 1.0.x**
  - **API: compatible**
  - **Namespace: incompatible. Aliyun.OpenServices.OpenStorageService is changed to Aliyun.OSS**

SDK source code and API documents

For the SDK source code, see *GitHub*. For more information, see *API documents*.

Code samples

OSS SDK for C# provides a variety of code samples for your reference or use. You can obtain the code samples from *GitHub*. The following table describes the content of code samples.

Sample file	Content
PutObjectSample.cs	*Upload objects*
AppendObjectSample.cs	*Append upload*
DoesObjectExistSample.cs	*Determine whether an object exists*
DeleteObjectsSample.cs	*Delete objects*
CopyObjectSample.cs	*Copy objects*
ModifyObjectMetaSample.cs	*Manage Object Meta*
MultipartUploadSample.cs	*Multipart upload*
ResumableSample.cs	*Resumable upload*

Sample file	Content
GetObjectSample.cs	*Download objects*
GetObjectByRangeSample.cs	*Range download*
GetObjectAclSample.cs	*Manage ACL for an object*
SetObjectAclSample.cs	*Manage ACL for an object*
ListObjectsSample.cs	*List objects*
UrlSignatureSample.cs	*Authorized access*
UploadCallbackSample.cs	*Upload callback*
ProgressSample.cs	*Upload progress bars* **and** *Download progress bars*
CNameSample.cs	**Use custome domain name to access OSS (CNAME)**
PostPolicySample.cs	*Form upload*
CreateBucketSample.cs	*Create a bucket*
DeleteBucketSample.cs	*Delete a bucket*
Doesbucketexistsample. CS	*Determine whether a bucket exists*
ListBucketsSample.cs	*List buckets*
SetBucketAclSample.cs	*Configure an ACL for a bucket*
SetBucketLifecycleSample.cs	*Lifecycle management*
SetBucketLoggingSample.cs	*Access log files*
SetBucketRefererSample.cs	*Hotlink protection*
SetBucketWetbsiteSample.cs	*Static website hosting*
SetBucketCorsSample.cs	*CORS*
ImageProcessSample.cs	*Image processing*

8.2 Installation

This topic describes how to install OSS C# SDK.

Preparation

- **Windows**
  - **Applicable to `.NET 2.0` and later.**
  - **Applicable to `Visual Studio 2010` and later.**
- **Linux Mac**

  Applicable to `Mono 3.12` and later.

Download SDK

- *Direct download*
- *Download from GitHub*
- *Download previous versions*

Install SDK

- **Install OSS C# SDK in Windows**
  - **Install SDK through NuGet**
    1. **If your Visual Studio does not have NuGet installed, install *NuGet* first.**
    2. **Create a project or open an existing project in Visual Studio, selectTools > NuGet Package Manager > Management Solution NuGet Package.**
    3. **Search for aliyun.oss.sdk and find Aliyun.OSS.SDK or Aliyun.OSS.SDK.NetCore in the search results. Select the latest version, and click Install.**
  - **Install SDK through DLL reference**
    1. **Download and unzip the C# SDK package.**
    2. **In the Visual Studio, access Solution Resource Manager, select your project, right click Project Name. Select Reference > Add Reference, and then select Browse in the prompt dialog box.**
    3. **Find thedirectory that the SDK package is unzipped to, select `Aliyun.OSS.dll` in the bin directory, and click OK.**
  - **Install SDK through project introduction**

    If you download SDK package or the source code from GitHub and you want to install SDK package using the source code, you can

    1. **right click Solution in Visual Studio and select Add in the dialog box menu to add an existing project.**
    2. **In the prompt dialog box, select `aliyun-oss-sdk.csproj`, and click Open.**
    3. **Right click Your Projects and select Reference > Add Reference. In the prompt dialog box, click the Project tab, select the aliyun-oss-sdk project, and then click OK.**

- **Install OSS C# SDK in Windows**
  - **Install SDK through NuGet**
    1. **In Xamarin, create a project or open an existing project, and selectAdd NuGet Packages from Tool.**
    2. **Search and find aliyun.oss.sdk or or Aliyun.OSS.SDK.NetCore, select the latest version, and click Add Package to add the package to project application.**
  - **Install SDK through DLL reference**
    1. **Download and unzip the C# SDK package.**
    2. **In Xamarin, select your project in Solution, right click Reference, and then select Edit References in the prompt menu.**
    3. **In the Edit References dialog box, select** *.Net Assembly*, **clickBrowse. Find the directory that SDK is unzipped to, select** *Aliyun.OSS.dll* **in the bin directory, and then click Open.**

8.3 Quick start

This topic describes how to use OSS .NET SDK to perform routine operations such as bucket creation, object uploads, and object downloads.

Create a bucket

A bucket is a global namespace in OSS. It is similar to a data container that stores files.

Run the following code to create a bucket:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Create a bucket.
    var bucket = client.CreateBucket(bucketName);
    Console.WriteLine("Create bucket succeeded, {0} ", bucket.Name);
}
catch (Exception ex)
{
    Console.WriteLine("Create bucket failed, {0}", ex.Message);
```

```
}
```

For more information about bucket naming rules, see naming conventions in *Basic concepts*.

For more information about endpoints, see *Regions and endpoints*.

Upload objects

Run the following code to upload a file to OSS:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Upload a file
    var result = client.PutObject(bucketName, objectName, localFilen
ame);
    Console.WriteLine("Put object succeeded, ETag: {0} ", result.ETag
);
}
catch (Exception ex)
{
    Console.WriteLine("Put object failed, {0}", ex.Message);
}
```

For more information, see *Upload an object*.

Download objects

Run the following code to download specified object to a local file:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
var downloadFilename = "<yourDownloadFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    client.PutObject(bucketName, objectName, localFilename);
    // Download the object.
    var result = client.GetObject(bucketName, objectName);
    using (var requestStream = result.Content)
    {
        using (var fs = File.Open(downloadFilename, FileMode.
OpenOrCreate))
```

```
        {
            int length = 4 * 1024;
            var buf = new byte[length];
            do
            {
                length = requestStream.Read(buf, 0, length);
                fs.Write(buf, 0, length);
            } while (length ! = 0);
        }
    }
    Console.WriteLine("Get object succeeded");
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

List objects

Run the following code to list objects in a specified bucket:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var objects = new List<string>();
    ObjectListing result = null;
    String nextmarker = string. empty;
    do
    {
        var listObjectsRequest = new ListObjectsRequest(bucketName)
        {
            Marker = nextMarker,
        };
        // List objects.
        result = client.ListObjects(listObjectsRequest);
        foreach (var summary in result.ObjectSummaries)
        {
            Console.WriteLine(summary.Key);
            objects.Add(summary.Key);
        }
        nextMarker = result.NextMarker;
    } while (result.IsTruncated);
    Console.WriteLine("List objects of bucket:{0} succeeded ",
bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
```

```
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Delete objects

Run the following code to delete a specified object:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Delete an object.
    client.DeleteObject(bucketName, objectName);
    Console.WriteLine("Delete object succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Delete object failed, {0}", ex.Message);
}
```

8.4 Initialization

OSSClient serves as the OSS C# client to manage OSS resources such as buckets and objects.

Create an OSSClient instance

To create an OSSClient instance, you need to specify an endpoint. For more information about endpoints, see *Regions and endpoints* **and** *Bind a custom domain name*.

- **Use an OSS domain to create an OSSClient instance**

  Run the following code to create an OSSClient instance with a domain assigned by OSS:

```
using Aliyun.OSS;

const string accessKeyId = "<yourAccessKeyId>";
const string accessKeySecret = "<yourAccessKeySecret>";
const string endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

// Create a new OSSClient instance with the OSS access address
specified by the user and the AccessKeyId/AccessKeySecret granted by
 Alibaba Cloud.
var ossClient = new OssClient(endpoint, accessKeyId, accessKeySecret
);
```

- **Use a custom domain (CNAME) to create an OSSClient instance**

  Run the following code to create an OSSClient instance with CNAME:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

const string accessKeyId = "<yourAccessKeyId>";
const string accessKeySecret = "<yourAccessKeySecret>";
const string endpoint = "<yourDomain>";

// Create a ClientConfiguration instance. Modify parameters as
required.
var conf = new ClientConfiguration();

// Enable CNAME. CNAME indicates a custom domain bound to a bucket.
conf.IsCname = true;

// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret,
conf);
```

Note:

The ossClient.listBuckets method cannot be used when a CNAME is used.

Configure an OSSClient instance

ClientConfiguration is a configuration class of OSSClient. ClientConfiguration is used to configure parameters such as user agents, host proxies, connection timeout , and the maximum number of connections. You can configure the following parameters.

Parameter	Description	Configuration method
ConnectionLimit	Specifies the maximum number of HTTP connections that can be enabled.	512
MaxErrorRetry	Specifies the maximum number of retry attempts in the case of a request error.	3

Parameter	Description	Configuration method
ConnectionTimeout	Specifies the timeout time in milliseconds when connections are established. The default value is -1, indicating that the connections are not time-out.	-1
IsCname	Specifies whether CNAME can be used as an endpoint.	false
ProgressUpdateInterval	Specifies the update interval of the progress bar, which is calculated by bytes.	8096

A code example is given as follows:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

var conf = new ClientConfiguration();
conf.ConnectionLimit = 512;
conf.MaxErrorRetry = 3;
conf.ConnectionTimeout = 300;

var client = new OssClient(endpoint, accessKeyId, accessKeySecret,
conf);
```

- Data verification

  Run the following code for MD5 data verification:

  ```
  using Aliyun.OSS;
  using Aliyun.OSS.Common;

  var conf = new ClientConfiguration();
  // Enable MD5 verification. Set EnableMD5Check to perform MD5
  verification on the uploaded or downloaded data. MD5 verification is
   disabled by default.
  conf.EnalbeMD5Check = true;

  var client = new OssClient(endpoint, accessKeyId, accessKeySecret,
  conf);
  ```

  Note:

  System performance reduces when MD5 verification is disabled.

- **Proxy network**

  If you use proxy networks, you can configure the following parameters to access OSS:

Parameter	Description	Default value
ProxyHost	**A proxy server, such as `8.8.8.8` or `abc.def.com`.**	**Null**
ProxyPort	**A proxy port, such as `3128` or `8080`.**	**Null**
ProxyUserName	**The proxy service account, which is optional**	**Null**
ProxyPassword	**The proxy service password, which is optional**	**Null**

  An example of proxy network access without using the account and password is as follows:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

var conf = new ClientConfiguration();
conf.ProxyHost = "8.8.8.8";
conf.ProxyPort = 3128;

var client = new OssClient(endpoint, accessKeyId, accessKeySecret,
conf);
```

  An example of proxy network access using the account and password is as follows:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

var conf = new ClientConfiguration();
conf.ProxyHost = "8.8.8.8";
conf.ProxyPort = 3128;
conf.ProxyUserName = "user";
conf.ProxyPassword = "6666";
```

```
var client = new OssClient(endpoint, accessKeyId, accessKeySecret,
conf);
```

8.5 Buckets

8.5.1 Manage a bucket

A bucket serves as a container that stores objects. Objects belong to a bucket.

Create a bucket

For the complete code of creating a bucket, see *GitHub*.

You can run the following code to create a bucket:

```
using Aliyun.OSS;

// Initialize an OSSClient.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);

// Create a new bucket.
public void CreateBucket(string bucketName)
{
    try
    {
        // Create a bucket. The bucketName is globally unique.
        client.CreateBucket(bucketName);
        Console.WriteLine("Create bucket succeeded");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Create bucket failed. {0}", ex.Message);
    }
}
```

List buckets

For the complete code of listing buckets, see *GitHub*.

You can run the following code to list all buckets under an account:

```
using Aliyun.OSS;

// Initialize an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);

// List the information about all buckets under an account.
public void ListBuckets()
{
    try
    {
        var buckets = client.ListBuckets();

        Console.WriteLine("List bucket succeeded");
        foreach (var bucket in buckets)
        {
```

```
            Console.WriteLine("Bucket name: {0}, Location: {1}, Owner: {2
}", bucket.Name, bucket.Location, bucket.Owner);
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("List bucket failed. {0}", ex.Message);
    }
}
```

Determine whether a bucket exists

For the complete code of determining whether a bucket exists, see *GitHub*.

You can run the following code to determine whether a bucket exists:

```
using Aliyun.OSS;

//Initialize an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);

// Determine whether a bucket exists.
public void DoesBucketExist(string bucketName)
{
    try
    {
        var exist = client.DoesBucketExist(bucketName);

        Console.WriteLine("Check object Exist succeeded");
        Console.WriteLine("exist ? {0}", exist);
    }
    catch (Exception ex)
    {
        Console.WriteLine("Check object Exist failed. {0}", ex.Message
);
    }
}
```

Configure an ACL for a bucket

For the complete code of configuring ACL, see *GitHub*.

The ACL of a bucket includes the following permissions.

Permission	Description	Value
Private	**The bucket owner and the authorized users can read and write objects in the bucket. Other users cannot perform any operation on the objects.**	**oss.ACLPrivate**

Permission	Description	Value
Public read	The bucket owner and the authorized users can read and write objects in the bucket. Other users can only read the objects in the bucket. Authorize this permission with caution.	oss.ACLPublicRead
Public read-write	All users can read and write objects in the bucket. Authorize this permission with caution.	oss.ACLPublicReadWrite

You can run the following code to configure an ACL for a bucket:

```
using Aliyun.OSS;

// Initialize an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);

// Configure an access ACL for the bucket
public void SetBucketAcl(string buckteName)
{
    try
    {
        // Set the ACL of the bucket to public read.
        client.SetBucketAcl(bucketName, CannedAccessControlList.
PublicRead);
        Console.WriteLine("Set bucket ACL succeeded");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Set bucket ACL failed. {0}", ex.Message);
    }
}
```

Obtain the ACL for a bucket

For the complete code of obtaining the ACL for a bucket, see *GitHub*.

You can run the following code to obtain the ACL for a bucket:

```
using Aliyun.OSS;

// Initialize an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);

// Obtain the ACL for the bucket.
public void GetBucketAcl(string bucketName)
{
    try
    {
        string bucketName = "your-bucket";
```

```
        var acl = client.GetBucketAcl(bucketName);

        Console.WriteLine("Get bucket ACL success", acl.ACL.ToString
());
    }
    catch (Exception ex)
    {
        Console.WriteLine("Get bucket ACL failed. {0}", ex.Message);
    }
}
```

Delete a bucket

For the complete code of deleting a bucket, see *GitHub*.

Before a bucket is deleted, ensure that all objects in the bucket and fragments that are generated from multipart upload are deleted.

Note:

To delete the fragments that are generated from multipart upload, use *Bucket.ListMultipartUploads* to list all fragments, and then use *Bucket.AbortMultipartUpload* to delete the fragments.

You can run the following code to delete a bucket:

```
using Aliyun.OSS;

// Initialize an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);

// Delete the bucket.
public void DeleteBucket(string bucketName)
{
    try
    {
        client.DeleteBucket(bucketName);

        Console.WriteLine("Delete bucket succeeded");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Delete bucket failed. {0}", ex.Message);
    }
}
```

8.5.2 Manage lifecycle rules

You can use OSS to configure lifecycle rules to reduce storage costs. The lifecycle management rules can be used for automatic deletion of expired objects and fragments, or storage class conversion to Infrequent Access for objects that will expire soon. Each rule includes:

- **Rule ID: It identifies a rule. Ensure that each rule ID in a bucket is unique.**
- **Policy: You can configure a policy by using the following configuration methods. Only one method can be configured for a bucket.**
  - **Configure by Prefix: You can create multiple rules with this method. Ensure that each prefix is unique.**
  - **Configure for Entire Bucket: You can configure only one rule with this method.**
- **Expired: You can configure the following configuration methods:**
  - **Set by Number of Days: It specifies the number of days from the last modificati on time of objects.**
  - **Set by Date: It specifies the date prior to which objects are created. If objects are created prior to the date, these objects are deleted.**
- **Status: This lifecyle rule takes effect or not.**

The parts uploaded by uploadPart method also support lifecycle rules. The last modification time of an object is the time the multipart upload event is initiated.

For more information about lifecycle management, see *Manage object lifecycle*.

For the complete code of lifecycle management, see *GitHub*.

Configure lifecycle rules

Run the following code to configure lifecycle rules:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var setBucketLifecycleRequest = new SetBucketLifecycleRequest(
bucketName);
    // Create the first lifecycle rule that is valid within 3 days.
    LifecycleRule lcr1 = new LifecycleRule()
    {
        ID = "delete obsoleted files",
        Prefix = "obsoleted/",
        Status = RuleStatus.Enabled,
        ExpriationDays = 3
    };
    // Create the second lifecycle rule.
    LifecycleRule lcr2 = new LifecycleRule()
    {
        ID = "delete temporary files",
        Prefix = "temporary/",
        Status = RuleStatus.Enabled,
```

```
        ExpriationDays = 20
    };
    setBucketLifecycleRequest.AddLifecycleRule(lcr1);
    setBucketLifecycleRequest.AddLifecycleRule(lcr2);
    // Configure the lifecycle.
    client.SetBucketLifecycle(setBucketLifecycleRequest);
    Console.WriteLine("Set bucket:{0} Lifecycle succeeded ",
bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

View lifecycle rules

For the complete code of viewing lifecycle rules, see *GitHub*.

Run the following code to view lifecycle rules:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // View the lifecycle rules.
    var rules = client.GetBucketLifecycle(bucketName);
    Console.WriteLine("Get bucket:{0} Lifecycle succeeded ",
bucketName);
    foreach (var rule in rules)
    {
        Console.WriteLine("ID: {0}", rule.ID);
        Console.WriteLine("Prefix: {0}", rule.Prefix);
        Console.WriteLine("Status: {0}", rule.Status);
        if (rule.ExpriationDays.HasValue)
            Console.WriteLine("ExpirationDays: {0}", rule.Expriation
Days);
    }
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
```

```
}
```

Clear lifecycle rules

Run the following code to clear lifecycle rules:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Clear the lifecycle rules
    client.DeleteBucketLifecycle(bucketName);
    Console.WriteLine("Delete bucket:{0} Lifecycle succeeded ",
bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.5.3 Set logging

You can enable access logs to record bucket access to log files, which are stored in a specified bucket.

The log file format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

For more information about access log files, see *Set access logging*. For the complete code of access logging, see *GitHub*.

Enable access logging

Run the following code to enable bucket access logging:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var targetBucketName = "<yourTargetBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
```

```
try
{
    // The targetBucketName parameter is the name of the bucket that
logs are stored. The logging- option indicates the prefix of log files
. The bucketName and targetBucketName can be the same bucket.
    var request = new SetBucketLoggingRequest(bucketName, targetBuck
etName, "logging-");
    // Enable access logging.
    client.SetBucketLogging(request);
    Console.WriteLine("Set bucket:{0} Logging succeeded ", bucketName
);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

View access logging configurations

For the complete code of viewing access logging configurations, see *GitHub*.

Run the following code to view the access logging configurations for a bucket:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // View bucket access logging configurations.
    var result = client.GetBucketLogging(bucketName);
    Console.WriteLine("Get bucket:{0} Logging succeeded, prefix:{1}",
bucketName, result.TargetPrefix);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Disable access logging

For the complete code of disabling access logging, see *GitHub*.

Run the following code to disable access logging for a bucket:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Disable access logging. Existing log files are not affected.
    client.DeleteBucketLogging(bucketName);
    Console.WriteLine("Delete bucket:{0} Logging succeeded ",
bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.5.4 CORS

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* and *PutBucketcors* in OSS Developer Guide.

For the complete code of CORS, see *GitHub*.

Configure CORS rules

For complete code of configuring CORS rules, see *GitHub*.

Run the following code to configure CORS rules for a specified bucket:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Creates an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var request = new SetBucketCorsRequest(bucketName);
    var rule1 = new CORSRule();
```

```
    // Specify the allowed source of the cross-origin access request.
    rule1. AddAllowedOrigin("http://www.a.com");
    // Specify the cross-region request methods (GET, PUT, DELETE,
POST, and HEAD) that are allowed.
    rule1. AddAllowedMethod("POST");
    // AllowedHeaders and ExposeHeaders do not support wildcards.
    rule1. AddAllowedHeader("*");
    // Specify the response header that allows user access from
applications.
    rule1. AddExposeHeader("x-oss-test");
    // A maximum of 10 rules is allowed.
    request.AddCORSRule(rule1);
    var rule2 = new CORSRule();
    // AllowedOrigins and AllowedMethods allow only one wildcard
asterisk (*). Wildcard asterisks (*) indicate that all sources of the
cross-origin requests and operations are allowed.
    rule2. AddAllowedOrigin("http://www.b.com");
    rule2. AddAllowedMethod("GET");
    // Determine whether the header specified in the Access-Control-
Headers in the pre-flight (OPTIONS) requests is allowed.
    rule2. AddExposeHeader("x-oss-test2");
    // Specify the cache time (seconds) for the response of browser
pre-flight (OPTIONS) requests to a specific resource.
    rule2. MaxAgeSeconds = 100;
    request.AddCORSRule(rule2);
    // Configure CORS rules.
    client.SetBucketCors(request);
    Console.WriteLine("Set bucket:{0} Cors succeeded ", bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Obtain CORS rules

For the complete code of obtaining CORS rules, see *GitHub*.

Run the following code to obtain CORS rules:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Obtain CORS rules.
    var result = client.GetBucketCors(bucketName);
    Console.WriteLine("Get bucket:{0} Cors succeeded ", bucketName);
    foreach (var rule in result)
    {
        foreach (var origin in rule.AllowedOrigins)
```

```
        {
            Console.WriteLine("Allowed origin:{0}", origin);
        }
    }
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Delete CORS rules

For the complete code of deleting CORS rules, see *GitHub*.

Run the following code to delete all CORS rules for a specified bucket:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Delete CORS rules.
    client.DeleteBucketCors(bucketName);
    Console.WriteLine("Delete bucket:{0} Cors succeeded ", bucketName
);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0} ; Error info: {1}.
 \nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.5.5 Static website hosting

You can set your bucket configuration to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

For more information about static website hosting, see *Static website hosting*.

Configure static website hosting

Run the following code to configure static website hosting:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Configure static website hosting.
    var request = new SetBucketWebsiteRequest(bucketName, "index.html
", "error.html");
    client.SetBucketWebsite(request);
    Console.WriteLine("Set bucket:{0} Wetbsite succeeded ", bucketName
);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

View static website hosting configurations

Run the following code to view static website hosting configurations:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // View static website hosting configurations.
    var result = client.GetBucketWebsite(bucketName);
    Console.WriteLine("Get bucket:{0} Wetbsite succeeded, index doc:{1
}, error doc:{2}",
                      bucketName, result.IndexDocument, result.
ErrorDocument);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
```

```
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Delete static website hosting configurations

Run the following code to delete static website hosting configurations:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Delete static website hosting configurations.
    client.DeleteBucketWebsite(bucketName);
    Console.WriteLine("Delete bucket:{0} Wetbsite succeeded ",
bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error info: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.5.6 Hotlink protection

This topic describes how to use hotlink protection.

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49*.

For the complete code of configuring hotlink protection, see *GitHub*.

Configure the referer whitelist

Run the following code to configure the referer whitelist:

```
using Aliyun.OSS;
```

```
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var refererList = new List<string>();
    // Add the referer field. The referer field allows question marks
 (?) and asterisks (*) for wildcard use.
    refererList.Add(" http://www.aliyun.com");
    refererList.Add(" http://www. *.com");
    refererList.Add(" http://www.?.aliyuncs.com");
    var srq = new SetBucketRefererRequest(bucketName, refererList);
    // Configure the referer list for a bucket.
    client.SetBucketReferer(srq);
    Console.WriteLine("Set bucket:{0} Referer succeeded ", bucketName
);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Obtain a referer whiltelist

For the complete code of obtaining a referer whitelist, see *GitHub*.

Run the following code to obtain a referer whiltelist:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Obtain the referer list for a bucket.
    var rc = client.GetBucketReferer(bucketName);
    Console.WriteLine("Get bucket:{0} Referer succeeded ", bucketName
);
    Console.WriteLine("allow? " + (rc.AllowEmptyReferer ? "yes" : "no
"));
    if (rc.RefererList.Referers ! = null)
    {
        for (var i = 0; i < rc.RefererList.Referers.Length; i++)
            Console.WriteLine(rc.RefererList.Referers[i]);
    }
    else
    {
        Console.WriteLine("Empty Referer List");
```

```
    }
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
finally
{
    client.SetBucketReferer(new SetBucketRefererRequest(bucketName));
}
```

Clear a referer whitelist

Run the following code to clear a referer whitelist:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // You cannot clear a referer whitelist directly. To clear a
referer whitelist, you need to create the rule that allows an empty
referer field and replace the original rule with the new rule.
    var srq = new SetBucketRefererRequest(bucketName);
    client.SetBucketReferer(srq);
    Console.WriteLine("Set bucket:{0} Referer succeeded ", bucketName
);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.6 Upload objects

8.6.1 Overview

In OSS, an object is the basic unit for data operations. OSS .NET SDK provides the following file upload methods:

- *Simple upload*: supports the upload of a file up to 5 GB in size.
- *Append upload*: supports the upload of a file up to 5 GB in size.
- *Resumable upload*: supports concurrent upload and you can define the size of each part. It applies to the upload of large files. Resumable upload supports the upload of a file up to 48.8 TB in size.
- *Multipart upload*: supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files.

Note:

For more information about the usage scenarios of each upload method, see "Upload files" in OSS Developer Guide.

During the file upload, you can set *Object Meta*, and view the *upload progress*. After you complete the file upload, you can perform *upload callback*.

8.6.2 Simple upload

Upload a string

For the complete code of uploading a string, see *GitHub*.

You can run the following code to upload a string:

```
using System.Text;
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var objectContent = "More than just cloud." ;
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    byte[] binaryData = Encoding.ASCII.GetBytes(objectContent);
    MemoryStream requestContent = new MemoryStream(binaryData);
    // Upload a file.
    client.PutObject(bucketName, objectName, requestContent);
    Console.WriteLine("Put object succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Put object failed, {0}", ex.Message);
}
```

Upload a specified local file

For the complete code of uploading a specified local file, see *Git Hub*.

You can run the following code to upload a specified local file:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Upload a local file.
    client.PutObject(bucketName, objectName, localFilename);
    Console.WriteLine("Put object succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Put object failed, {0}", ex.Message);
}
```

Upload a file with MD5 verification

To ensure the consistency between the data sent by the SDK and the data received by the OSS client, you can add a Content-MD5 value in ObjectMeta. Then OSS checks the received data against the MD5 value.

For the complete code of uploading a file with MD5 verification, see *GitHub*.

```
using Aliyun.OSS;
using Aliyun.OSS.Util;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Calculate the MD5 value.
    string md5;
    using (var fs = File.Open(localFilename, FileMode.Open))
    {
        md5 = OssUtils.ComputeContentMd5(fs, fs.Length);
    }
    var objectMeta = new ObjectMetadata
    {
        ContentMd5 = md5
    };
    // Upload a file.
    client.PutObject(bucketName, objectName, localFilename, objectMeta
);
    Console.WriteLine("Put object succeeded");
}
catch (Exception ex)
{
```

```
    Console.WriteLine("Put object failed, {0}", ex.Message);
}
```

Note:

MD5 verification may cause a reduction in performance.

Asynchronous upload

For the complete code of asynchronous upload, see *GitHub*.

You can run the following code to perform asynchronous upload:

```
using System;
using System.IO;
using System.Threading;

using Aliyun.OSS;
using Aliyun.OSS.Common;

// Upload a file asynchronously.
namespace AsyncPutObject
{
    class Program
    {
        static string endpoint = "<yourEndpoint>";
        static string accessKeyId = "<yourAccessKeyId>";
        static string accessKeySecret = "<yourAccessKeySecret>";
        static string bucketName = "<yourBucketName>";
        static string objectName = "<yourObjectName>";
        static string localFilename = "<yourLocalFilename>";

        static AutoResetEvent _event = new AutoResetEvent(false);

        // Create an OSSClient instance.
        static OssClient client = new OssClient(endpoint, accessKeyId
, accessKeySecret);

        private static void PutObjectCallback(IAsyncResult ar)
        {
            try
            {
                client.EndPutObject(ar);
                Console.WriteLine(ar.AsyncState as string);
                Console.WriteLine("Put object succeeded");
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
            }
            finally
            {
                _event.Set();
            }
        }

        public static void AsyncPutObject()
        {
            try
            {
```

```
                using (var fs = File.Open(localFilename, FileMode.Open
))
                {
                    var metadata = new ObjectMetadata();
     // Add custom meta information.
                    metadata.UserMetadata.Add("mykey1", "myval1");
                    metadata.UserMetadata.Add("mykey2", "myval2");
                    metadata.CacheControl = "No-Cache";
                    metadata.ContentType = "text/html";
                    string result = "Notice user: put object finish";

                    // Upload a file asynchronously.
                    client.BeginPutObject(bucketName, objectName, fs,
metadata, PutObjectCallback, result.ToCharArray());

                    _event.WaitOne();
                }
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0}; Error
info: {1}. \nRequestID:{2}\tHostID:{3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId
);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.
Message);
            }
        }

        static void Main(string[] args)
        {
            Program.AsyncPutObject();
        }
    }
}
```

Note:

When using asynchronous upload, you must implement your own callback handler.

8.6.3 Append upload

For the complete code of append upload, see *GitHub*.

You cannot perform copyObject for appended objects. When you call AppendObject to upload data, if the target object does not exist, an appendable object is created. If the object already exists, data is appened to the object.

Run the following code for append upload:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
```

```
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
// If the object is appended for the first time, the append position
is 0, and the returned value is the position for the next append. The
position for the next append is the length of object before the append
.
long position = 0;
try
{
    var metadata = client.GetObjectMetadata(bucketName, objectName);
    position = metadata.ContentLength;
}
catch (Exception) { }
try
{
    using (var fs = File.Open(localFilename, FileMode.Open))
    {
        var request = new AppendObjectRequest(bucketName, objectName)
        {
            ObjectMetadata = new ObjectMetadata(),
            Content = fs,
            Position = position
        };
        // Append the object.
        var result = client.AppendObject(request);
        // Configure the position where the object is appended
        position = result.NextAppendPosition;
        Console.WriteLine("Append object succeeded, next append
position:{0}", position);
    }
    // Obtain the start position for the next append.
    using (var fs = File.Open(localFilename, FileMode.Open))
    {
        var request = new AppendObjectRequest(bucketName, objectName)
        {
            ObjectMetadata = new ObjectMetadata(),
            Content = fs,
            Position = position
        };
        var result = client.AppendObject(request);
        position = result.NextAppendPosition;
        Console.WriteLine("Append object succeeded, next append
position:{0}", position);
    }
}
catch (Exception ex)
{
    Console.WriteLine("Append object failed, {0}", ex.Message);
}
```

8.6.4 Resumable upload

You can use resumable upload to split a file you want to upload into several parts and upload them simultaneously. After you have uploaded all parts, you can

combine these parts into a complete object. Thus, you can complete uploading the entire object.

Current upload progress is recorded into a checkpoint file during the upload. If a part fails to be uploaded during the process, the object is uploaded from the part recorded in the checkpoint file when the upload restarts, that is, the resumable upload. After the upload is complete, the checkpoint is deleted.

For the complete code of resumable upload, see *GitHub*.

Run the following code for resumable upload:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
string checkpointDir = "<yourCheckpointDir>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    //  Configure parameters with UploadFileRequest.
    UploadObjectRequest request = new UploadObjectRequest(bucketName,
objectName, localFilename)
    {
        // Specify the size of a part you want to upload.
        PartSize = 8 * 1024 * 1024,
        // Specify the number of threads for simultaneous upload.
        ParallelThreadCount = 3,
        // The progress information of a resumable upload is stored
in the checkpointDir directory.  If a part fails to be uploaded, the
progress information is used to continue the upload. If the checkpoint
Dir directory is null, the resumable upload does not take effect, and
the file that failed to be uploaded is uploaded all over again.
        CheckpointDir = checkpointDir,
    };
    // Start resumable upload.
    client.ResumableUploadObject(request);
    Console.WriteLine("Resumable upload object:{0} succeeded",
objectName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
```

```
}
```

8.6.5 Multipart upload

For the complete code of multipart upload, see *GitHub*.

To enable multipart upload, perform the following steps:

1. **Initiate a multipart upload event.**

   You can call OssClient.initiateMultipartUpload to return the globally unique uploadId created in OSS.

2. **Upload parts.**

   You can call OssClient.uploadPart to upload part data.

   Notice:

   - **For parts with a same uploadId, parts are sequenced by their part numbers. If you have uploaded a part and use the same part number to upload another part, the later part will replace the former part.**
   - **OSS places the MD5 value of part data in ETag and returns the MD5 value to the user.**
   - **SDK automatically configures Content-MD5. OSS calculates the MD5 value of uploaded data and compares it with the MD5 value calculated by SDK. If the two values vary, the error code of InvalidDigest is returned.**

3. **Complete multipart upload.**

   After you have uploaded all parts, call partossClient.completeMultipartUpload to combine these parts into a complete object.

The following code is used as a complete example that describes the process of multipart upload:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
// Initiate the multipart upload.
var uploadId = "";
try
{
```

```
    // Specifiy the name and the bucket of the uploaded object. You
can configure ObjectMeta when using InitiateMultipartUploadRequest.
However, you do not need to specify the ContentLength.
    var request = new InitiateMultipartUploadRequest(bucketName,
objectName);
    var result = client.InitiateMultipartUpload(request);
    uploadId = result.UploadId;
    // Print the UploadId
    Console.WriteLine("Init multi part upload succeeded");
    Console.WriteLine("Upload Id:{0}", result.UploadId);
}
catch (Exception ex)
{
    Console.WriteLine("Init multi part upload failed, {0}", ex.Message
);
}
// Calculate the total number of parts.
var partSize = 100 * 1024;
var fi = new FileInfo(localFilename);
var fileSize = fi.Length;
var partCount = fileSize / partSize;
if (fileSize % partSize ! = 0)
{
    partCount++;
}
// Start the multipart upload. partETags is a list of partETags. OSS
 verifies the validity of all parts one by one after it receives the
partETags. After part verification is successful, OSS combines these
parts into a complete object.
var partETags = new List<PartETag>();
try
{
    using (var fs = File.Open(localFilename, FileMode.Open))
    {
        for (var i = 0; i < partCount; i++)
        {
            var skipBytes = (long)partSize * i;
            // Locate to the start position of the upload.
            fs.Seek(skipBytes, 0);
            // Calculate the size of the uploaded part. The size of
the last part is the size of the data remained after being splited by
the part size.
            var size = (partSize < fileSize - skipBytes) ? partSize :
 (fileSize - skipBytes);
            var request = new UploadPartRequest(bucketName, objectName
, uploadId)
            {
                InputStream = fs,
                PartSize = size,
                PartNumber = i + 1
            };
            // Call the UploadPart interface to upload the object. The
 returned results contain the ETag value of the uploaded part.
            var result = client.UploadPart(request);
            partETags.Add(result.PartETag);
            Console.WriteLine("finish {0}/{1}", partETags.Count,
partCount);
        }
        Console.WriteLine("Put multi part upload succeeded");
    }
}
catch (Exception ex)
{
```

```
    Console.WriteLine("Put multi part upload failed, {0}", ex.Message
);
}
// List uploaded parts.
try
{
    var listPartsRequest = new ListPartsRequest(bucketName, objectName
, uploadId);
    var listPartsResult = client.ListParts(listPartsRequest);
    Console.WriteLine("List parts succeeded");
    // Upload each part simultaneously until all parts are uploaded.
    var parts = listPartsResult.Parts;
    foreach (var part in parts)
    {
        Console.WriteLine("partNumber: {0}, ETag: {1}, Size: {2}",
part.PartNumber, part.ETag, part.Size);
    }
}
catch (Exception ex)
{
    Console.WriteLine("List parts failed, {0}", ex.Message);
}
// Complete the multipart upload.
try
{
    var completeMultipartUploadRequest = new CompleteMultipartUpl
oadRequest(bucketName, objectName, uploadId);
    foreach (var partETag in partETags)
    {
        completeMultipartUploadRequest.PartETags.Add(partETag);
    }
    var result = client.CompleteMultipartUpload(completeMultipartUpl
oadRequest);
    Console.WriteLine("complete multi part succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("complete multi part failed, {0}", ex.Message);
}
```

Cancel a multipart upload event

For the complete code of canceling a multiplart upload event, see *GitHub*.

Run the following code to cancel a multipart upload event:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var uploadId = "";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
// Initiate the multipart upload.
try
{
    var request = new InitiateMultipartUploadRequest(bucketName,
objectName);
    var result = client.InitiateMultipartUpload(request);
    uploadId = result.UploadId;
```

```
    // Print the UploadId.
    Console.WriteLine("Init multi part upload succeeded");
    Console.WriteLine("Upload Id:{0}", result.UploadId);
}
catch (Exception ex)
{
    Console.WriteLine("Init multi part upload failed, {0}", ex.Message
);
}
// Cancel the multipart upload.
try
{
    var request = new AbortMultipartUploadRequest(bucketName,
objectName, uploadId);
    client.AbortMultipartUpload(request);
    Console.WriteLine("Abort multi part succeeded,  {0}", uploadId);
}
catch (Exception ex)
{
    Console.WriteLine("Abort multi part failed, {0}", ex.Message);
}
```

List uploaded parts

For the complete code of listing uploaded parts, see *GitHub*.

Run the following code to list uploaded parts:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var uploadId = "<yourUploadId>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    PartListing listPartsResult = null;
    var nextMarker = 0;
    do
    {
        var listPartsRequest = new ListPartsRequest(bucketName,
objectName, uploadId)
        {
            PartNumberMarker = nextMarker,
        };
        // List uploaded parts.
        listPartsResult = client.ListParts(listPartsRequest);
        Console.WriteLine("List parts succeeded");
        // Upload each part simultaneously until all parts are
uploaded.
        var parts = listPartsResult.Parts;
        foreach (var part in parts)
        {
            Console.WriteLine("partNumber: {0}, ETag: {1}, Size: {2
}", part.PartNumber, part.ETag, part.Size);
        }
        nextMarker = listPartsResult.NextPartNumberMarker;
    } while (listPartsResult.IsTruncated);
}
```

```
catch (Exception ex)
{
    Console.WriteLine("List parts failed, {0}", ex.Message);
}
```

List multipart upload events

Call ossClient.listMultipartUploads to list all ongoing part upload events (events that have been initiated but not completed or have been canceled). You can configure the following parameters:

Parameter	Description	Configuration method
prefix	Specifies the prefix that must be included in the returned object name . Note that if you use a prefix for query, the returned object name will contain the prefix.	ListMultipartUploads Request.setPrefix(String prefix)
delimiter	Specifies a delimiter of a forward slash (/) used to group object names . The object between the specified prefix and the first occurrence of a delimiter of a forward slash (/) is commonPref ixes.	ListMultipartUploads Request.setDelimiter(String delimiter)
maxUploads	Specifies the maximum number of part upload events. The maximum value (also default value) you can set is 1,000.	ListMultipartUploads Request.setMaxUploads(Integer maxUploads)
keyMarker	Lists all part upload events with the object whose names start with a letter that comes after the keyMarker value in the alphabetical order. You can use this parameter with the uploadIdMarker parameter to specify the initial position for the specified returned result.	ListMultipartUploads Request.setKeyMarker(String keyMarker)

Parameter	Description	Configuration method
uploadIdMarker	You can use this parameter with the keyMarker parameter to specify the initial position for the specified returned result. If you do not configure keyMarker, the uploadIdMarker parameter is invalid. If you configure keyMarker, the query result contains: · All objects whose names start with a letter that comes after the keyMarker value. · All objects whose names start with a letter that is the same as the keyMarker value in the alphabetical order and the value of uploadId greater than that of uploadIdMarker.	ListMultipartUploadsRequest.setUploadIdMarker(String uploadIdMarker)

For the complete code of listing multiplart upload events, see *GitHub*.

Run the following code to list all multipart upload events:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    MultipartUploadListing multipartUploadListing = null;
    var nextMarker = string.Empty;
    do
    {
        // List multipart upload events.
        var request = new ListMultipartUploadsRequest(bucketName)
        {
            KeyMarker = nextMarker,
        };
        multipartUploadListing = client.ListMultipartUploads(request);
        Console.WriteLine("List multi part succeeded");
```

```
        // List the information about multipart upload events.
        foreach (var mu in multipartUploadListing.MultipartUploads)
        {
            Console.WriteLine("Key: {0},  UploadId: {1}", mu.Key, mu.
UploadId);
        }
        // If the returned result shows that the value of isTruncate
d is false, the values of nextKeyMarker and nextUploadIdMarker are
returned and used as the initial position for the next object reading.
        nextMarker = multipartUploadListing.NextKeyMarker;
    } while (multipartUploadListing.IsTruncated);
}
catch (Exception ex)
{
    Console.WriteLine("List multi part uploads failed, {0}", ex.
Message);
}
```

Specify the prefix and maximum number of returned results

Run the following code to perform multipart upload with a specified prefix and maximum number of returned results:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Define prefix.
var prefix = "<yourObjectPrefix>";
// Set the maximum number of returned result to 100.
var maxUploads = 100;
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    MultipartUploadListing multipartUploadListing = null;
    var nextMarker = string.Empty;
    do
    {
        // List multipart upload events. By default, 1,000 parts are
listed.
        var request = new ListMultipartUploadsRequest(bucketName)
        {
            KeyMarker = nextMarker,
            // Specify the prefix.
            Prefix = prefix,
            // Specify the maximum number of returned result.
            MaxUploads = maxUploads,
        };
        multipartUploadListing = client.ListMultipartUploads(request);
        Console.WriteLine("List multi part succeeded");
        // List the information about each multipart upload event.
        foreach (var mu in multipartUploadListing.MultipartUploads)
        {
            Console.WriteLine("Key: {0},  UploadId: {1}", mu.Key, mu.
UploadId);
        }
        nextMarker = multipartUploadListing.NextKeyMarker;
    } while (multipartUploadListing.IsTruncated);
}
catch (Exception ex)
```

```
{
    Console.WriteLine("List multi part uploads failed, {0}", ex.
Message);
}
```

8.6.6 Upload progress bars

You can use progress bars to indicate the upload or download progress.

For the complete code of progress bars, see *GitHub*.

The following code is used as an example to describe how to view progress information with PutObject:

```
using System;
using System.IO;
using System.Text;
using Aliyun.OSS;
using Aliyun.OSS.Common;
namespace PutObjectProgress
{
    class Program
    {
        static void Main(string[] args)
        {
            Program.PutObjectProgress();
            Console.ReadKey();
        }
        public static void PutObjectProgress()
        {
            var endpoint = "<yourEndpoint>";
            var accessKeyId = "<yourAccessKeyId>";
            var accessKeySecret = "<yourAccessKeySecret>";
            var bucketName = "<yourBucketName>";
            var objectName = "<yourObjectName>";
            var localFilename = "<yourLocalFilename>";
            // Create an OSSClient instance.
            var client = new OssClient(endpoint, accessKeyId,
accessKeySecret);
            // Upload with a progress bar displayed.
            try
            {
                using (var fs = File.Open(localFilename, FileMode.Open
))
                {
                    var putObjectRequest = new PutObjectRequest(
bucketName, objectName, fs);
                    putObjectRequest.StreamTransferProgress +=
streamProgressCallback;
                    client.PutObject(putObjectRequest);
                }
                Console.WriteLine("Put object:{0} succeeded",
objectName);
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0}; Error
info: {1}. \nRequestID: {2}\tHostID: {3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId
);
            }
```

```
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.
Message);
            }
        }
        // Obtain the upload progress.
        private static void streamProgressCallback(object sender,
StreamTransferProgressArgs args)
        {
            System.Console.WriteLine("ProgressCallback - Progress: {0
}%, TotalBytes:{1}, TransferredBytes:{2} ",
                args.TransferredBytes * 100 / args.TotalBytes, args.
TotalBytes, args.TransferredBytes);
        }
    }
}
```

8.6.7 Upload callback

For the complete code of upload callback, see *GitHub*.

Run the following code for upload callback:

```
using System;
using System.IO;
using System.Text;
using Aliyun.OSS;
using Aliyun.OSS.Common;
using Aliyun.OSS.Util;
namespace Callback
{
    class Program
    {
        static void Main(string[] args)
        {
            Program.PutObjectCallback();
            Console.ReadKey();
        }
        public static void PutObjectCallback()
        {
            var endpoint = "<yourEndpoint>";
            var accessKeyId = "<yourAccessKeyId>";
            var accessKeySecret = "<yourAccessKeySecret>";
            var bucketName = "<yourBucketName>";
            var objectName = "<yourObjectName>";
            var localFilename = "<yourLocalFilename>";
            // Specify the IP address of the server you want to send
the callback request to.
            const string callbackUrl = "http://oss-demo.aliyuncs.com:
23450";
            # Configure the value of the body field carried in the
callback request.
            const string callbackBody = "bucket=${bucket}&object=${
object}&etag=${etag}&size=${size}&mimeType=${mimeType}&" +
                                        "my_var1=${x:var1}&my_var2=${x
:var2}";
            // Create an OSSClient instance.
            var client = new OssClient(endpoint, accessKeyId,
accessKeySecret);
            try
            {
```

```
                string responseContent = "";
                var metadata = BuildCallbackMetadata(callbackUrl,
callbackBody);
                using (var fs = File.Open(localFilename, FileMode.Open
))
                {
                    var putObjectRequest = new PutObjectRequest(
bucketName, objectName, fs, metadata);
                    var result = client.PutObject(putObjectRequest);
                    responseContent = GetCallbackResponse(result);
                }
                Console.WriteLine("Put object:{0} succeeded, callback
response content:{1}", objectName, responseContent);
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0}; Error
info: {1}. \nRequestID: {2}\tHostID: {3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId
);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.
Message);
            }
        }
        // Configure upload callback.
        private static ObjectMetadata BuildCallbackMetadata(string
callbackUrl, string callbackBody)
        {
            string callbackHeaderBuilder = new CallbackHeaderBuilder(
callbackUrl, callbackBody). Build();
            string CallbackVariableHeaderBuilder = new CallbackVa
riableHeaderBuilder().
                AddCallbackVariable("x:var1", "x:value1"). AddCallbac
kVariable("x:var2", "x:value2"). Build();
            var metadata = new ObjectMetadata();
            metadata.AddHeader(HttpHeaders.Callback, callbackHe
aderBuilder);
            metadata.AddHeader(HttpHeaders.CallbackVar, CallbackVa
riableHeaderBuilder);
            return metadata;
        }
        // Read the message returned from upload callback.
        private static string GetCallbackResponse(PutObjectResult
putObjectResult)
        {
            string callbackResponse = null;
            using (var stream = putObjectResult.ResponseStream)
            {
                var buffer = new byte[4 * 1024];
                var bytesRead = stream.Read(buffer, 0, buffer.Length);
                callbackResponse = Encoding.Default.GetString(buffer,
0, bytesRead);
            }
            return callbackResponse;
        }
    }
}
```

Note:

For more information about upload callback, see *Upload callback* in the Developer Guide.

8.7 Download objects

8.7.1 Overview

OSS .NET SDK provides the following download methods:

- *Streaming download*
- *Range download*
- *Resumable download*

You can view the download progress shown in *download progress bars*.

8.7.2 Streaming download

If you have a large object to download or it is time-consuming to download the entire object at a time, you can use streaming download. Streaming download enables you to download part of the object each time until you have downloaded the entire object.

For the complete code of streaming download, see *GitHub*.

Run the following code to download a specified OSS object to a stream:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var downloadFilename = "<yourDownloadFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Download an object to a stream. OssObject includes object
information, such as the bucket that contains the object, object name
, metadata, and an input stream.
    var obj = client.GetObject(bucketName, objectName);
    using (var requestStream = obj.Content)
    {
        byte[] buf = new byte[1024];
        var fs = File.Open(downloadFilename, FileMode.OpenOrCreate);
        var len = 0;
        // Use the input stream to read the object content into a file
 or the memory.
        while ((len = requestStream.Read(buf, 0, 1024)) ! = 0)
        {
            fs.Write(buf, 0, len);
```

```
        }
        fs.Close();
    }
    Console.WriteLine("Get object succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Get object failed. {0}", ex.Message);
}
```

8.7.3 Range download

If you only need part of the data in an object, you can use range downloads to download specified content.

For the complete code of range download, see *GitHub*.

Run the following code for range download:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var downloadFilename = "<yourDownloadFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var getObjectRequest = new GetObjectRequest(bucketName, objectName
);
    // Set the range, which is from the 20th byte to the 100th byte.
    getObjectRequest.SetRange(20, 100);
    // Start range download. The setRange of getObjectRequest can be
used to implement multipart download and resumable download.
    var obj = client.GetObject(getObjectRequest);
    // Download the data and write it into a file.
    using (var requestStream = obj.Content)
    {
        byte[] buf = new byte[1024];
        var fs = File.Open(downloadFilename, FileMode.OpenOrCreate);
        var len = 0;
        while ((len = requestStream.Read(buf, 0, 1024)) ! = 0)
        {
            fs.Write(buf, 0, len);
        }
        fs.Close();
    }
    Console.WriteLine("Get object succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Get object failed. {0}", ex.Message);
}
```

You can configure the following parameters of GetObjectRequest.

Parameter	Description
Range	Specifies the transmission range of an object.
ModifiedSinceConstraint	If the specified time is earlier than the actual modification time of the file, the file is transmitted normally. Otherwise, a 304 Not Modified exception is thrown.
UnmodifiedSinceConstraint	If the time passed to the parameter is equal to or later than the actual modification time of the file, the file is transmitted normally Otherwise, a 412 precondition failed exception is thrown.
MatchingETagConstraints	Passes in a group of ETags. If the ETags match the ETag of the file, the file is transmitted normally. Otherwise, a 412 precondition failed exception is thrown.
NonmatchingEtagConstraints	Passes in a group of ETags. If the ETags do not match the ETag of the file, the file is transmitted normally. Otherwise, a 304 Not Modified exception is thrown.
ResponseHeaderOverrides	Customizes some headers in requests returned by OSS.

8.7.4 Resumable download

Large-sized objects may fail to be downloaded because the network is unstable or the Java program exits abnormally and the entire object needs to be downloaded again. However, the object may still fail to be downloaded after multiple attempts. Therefore, OSS provides resumable download.

Run the following code for resumable download:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
var downloadFilename = "<yourDownloadFilename>";
var checkpointDir = "<yourCheckpointDir>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
```

```
{
    // Configure parameters with DownloadObjectRequest.
    DownloadObjectRequest request = new DownloadObjectRequest(
bucketName, objectName, downloadFilename)
    {
        // Specify the size of a part you want to download.
        PartSize = 8 * 1024 * 1024,
        // Specifies the number of concurrent download threads.
        ParallelThreadCount = 3,
        // Specify the checkpointDir to store the download progress
information. If the download fails, it can be continued based on the
 progress information. If the checkpointDir directory is null, the
resumable upload is not enabled, which means that objects that fails
to be downloaded are downloaded all over again.
        CheckpointDir = checkpointDir,
    };
    // Start resumable download.
    client.ResumableDownloadObject(request);
    Console.WriteLine("Resumable download object:{0} succeeded",
objectName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

For more information about resumable download, see *Resumable download* in OSS Developer Guide.

8.7.5 Download progress bars

You can use progress bars to indicate the upload or download progress.

For the complete code of download progress bar, see *GitHub*.

The following code is used as an example to describe how to view progress information with GetObject:

```
using System;
using System.IO;
using Aliyun.OSS;
using Aliyun.OSS.Common;
namespace GetObjectProgress
{
    class Program
    {
        static void Main(string[] args)
        {
            Program.GetObjectProgress();
            Console.ReadKey();
        }
        public static void GetObjectProgress()
        {
            var endpoint = "<yourEndpoint>";
```

```
            var accessKeyId = "<yourAccessKeyId>";
            var accessKeySecret = "<yourAccessKeySecret>";
            var bucketName = "<yourBucketName>";
            var objectName = "<yourObjectName>";
            // Create an OSSClient instance.
            var client = new OssClient(endpoint, accessKeyId,
accessKeySecret);
            try
            {
                var getObjectRequest = new GetObjectRequest(bucketName
, objectName);
                getObjectRequest.StreamTransferProgress += streamProg
ressCallback;
                // Download an object.
                var ossObject = client.GetObject(getObjectRequest);
                using (var stream = ossObject.Content)
                {
                    var buffer = new byte[1024 * 1024];
                    var bytesRead = 0;
                    while ((bytesRead = stream.Read(buffer, 0, buffer.
Length)) > 0)
                    {
                        // Process the read data (detailed code is not
 listed here).
                    }
                }
                Console.WriteLine("Get object:{0} succeeded",
objectName);
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0}; Error
info: {1}. \nRequestID:{2}\tHostID:{3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId
);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.
Message);
            }
        }
        private static void streamProgressCallback(object sender,
StreamTransferProgressArgs args)
        {
            System.Console.WriteLine("ProgressCallback - Progress: {0
}%, TotalBytes:{1}, TransferredBytes:{2} ",
                args.TransferredBytes * 100 / args.TotalBytes, args.
TotalBytes, args.TransferredBytes);
        }
    }
}
```

8.8 Manage objects

8.8.1 Overview

This topic is the overview of object management in OSS .NET SDK.

You can perform the following operations on objects in a bucket with APIs:

- *Determine whether an object exists*
- *Manage ACL for an object*
- *Manage Object Meta*
- *List objects*
- *Delete an object*
- *Copy an object*
- *Restore an archive object*
- *Manage a symbolic link*

8.8.2 Determine whether a specified object exists

Run the following code to determine whether a specified object exists:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";

// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    //  Determine whether a specified object exists.
    var exist = client.DoesObjectExist(bucketName, objectName);
    Console.WriteLine("Object exist ? " + exist);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.8.3 Manage ACL for an object

The following table describes the permissions included in the Access Control List (ACL) for an object.

Permission	Description	Value
default	The ACL of an object is the same with that of its bucket.	CannedAccessControlList.Default
Private	Only the object owner and authorized users can read and write the object.	CannedAccessControlList.Private
Public read	Only the object owner and authorized users can read and write the object. Other users can only read the object. Authorize this permission with caution.	CannedAccessControlList.PublicRead
Public read-write	All users can read and write the object. Authorize this permission with caution.	CannedAccessControlList.PublicReadWrite

The ACL of objects take precedence over that of buckets. For example, if the ACL of a bucket is private, while the object ACL is public read-write, all users can read and write the object. If an object is not configured with an ACL, its ACL is the same as that of its bucket by default.

For the complete code of configuring an ACL for an object, see *GitHub*. For the complete code of obtaining the ACL for an object, see *GitHub*.

You can run the following code to configure an ACL for an object and obtain the ACL for an object:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;

var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";

// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
// Configure an ACL for an object.
try
{
    // Use SetObjectAcl to configure an ACL for an object.
    client.SetObjectAcl(bucketName, objectName, CannedAccessControlL
ist.PublicRead);
    Console.WriteLine("Set Object:{0} ACL succeeded ", objectName);
}
```

```
catch (Exception ex)
{
    Console.WriteLine("Set Object ACL failed with error info: {0}", ex
.Message);
}
// Obtain the ACL for an object.
try
{
    // Use GetObjectAcl to obtain the ACL for an object.
    var result = client.GetObjectAcl(bucketName, objectName);
    Console.WriteLine("Get Object ACL succeeded, Id: {0}  ACL: {1}",
        result.Owner.Id, result.ACL.ToString());
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID: {2}\tHostID: {3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

For details about object ACL, see *Access control*.

8.8.4 Manage Object Meta

Object Meta includes HTTP headers and user-defined Object Meta. For more information, see *Object Meta* **in OSS Developer Guide.**

For the complete code of configuring Object Meta, see *GitHub*. **For the complete code of modifying Object Meta, see** *GitHub*.

Run the following code to configure, modify, and obtain Object Meta:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var localFilename = "<yourLocalFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    using (var fs = File.Open(localFilename, FileMode.Open))
    {
        // Create Object Meta. You can configure HTTP headers with
Object Meta.
        var metadata = new ObjectMetadata()
        {
            // Specify the object type.
            ContentType = "text/html",
            // Configure the expiration time (GMT is used).
            ExpirationTime = DateTime.Parse("2025-10-12T00:00:00.000Z
"),
```

```
        };
        // Configure the length of the object you want to upload.
If the object length is greater than the configured length, only
the configured length of the object is uploaded. If the configured
length is greater than the actual object length, the entire object is
uploaded.
        metadata.ContentLength = fs.Length;
        // Configure the cache action of a webpage when an object is
downloaded.
        metadata.CacheControl = "No-Cache";
        // Set the value of metadata mykey1 to myval1.
        metadata.UserMetadata.Add("mykey1", "myval1");
        // Set the value of metadata mykey2 to myval2.
        metadata.UserMetadata.Add("mykey2", "myval2");
        var saveAsFilename = "Filetest123.txt";
        var contentDisposition = string.Format("attachment;filename*=
utf-8''{0}", HttpUtils.EncodeUri(saveAsFilename, "utf-8"));
        // Provide a default file name when saving the requested
content as a file.
        metadata.ContentDisposition = contentDisposition;
        // Upload a file and configure the Object Meta.
        client.PutObject(bucketName, objectName, fs, metadata);
        Console.WriteLine("Put object succeeded");
        //Obtain Object Meta.
        var oldMeta = client.GetObjectMetadata(bucketName, objectName
);
        // Configure new Object Meta.
        var newMeta = new ObjectMetadata()
        {
            ContentType = "application/octet-stream",
            ExpirationTime = DateTime.Parse("2035-11-11T00:00:00.000Z
"),
            // Specify the encoding format of a download object.
            ContentEncoding = null,
            CacheControl = ""
        };
        // Add custom Object Meta.
        newMeta.UserMetadata.Add("author", "oss");
        newMeta.UserMetadata.Add("flag", "my-flag");
        newMeta.UserMetadata.Add("mykey2", "myval2-modified-value");
        // Use the ModifyObjectMeta method to modify Object Meta.
        client.ModifyObjectMeta(bucketName, objectName, newMeta);
    }
}
catch (Exception ex)
{
    Console.WriteLine("Put object failed, {0}", ex.Message);
}
```

Note:

- **For detailed information about HTTP header, see *RFC2616*.**
- **The size of Object Meta with custom HTTP headers is no larger than 8 KB.**

8.8.5 List objects

For the complete code of listing objects, see *GitHub*.

Objects are listed alphabetically. You can use ListObjects to list objects in a bucket.

The following table describes the parameters of ListObjects

Parameter	Description
Delimiter	Specifies a delimiter of a forward slash (/) used to group object names . CommonPrefixes is a set of objects which are ended with the delimiter and have the same prefix.
Marker	Specifies the initial object in the list.
MaxKeys	Specifies the maximum number of objects that can be listed. The default value of MaxKeys is 100. The maximum value of MaxKeys is 1,000.
Prefix	Specifies the prefix you configure to list required objects.

Simple list

For the complete code of simple list, see *GitHub*.

Run the following code to list objects in a specified bucket.

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var listObjectsRequest = new ListObjectsRequest(bucketName);
    // Simply list the objects in a specified bucket. 100 records are
returned by default.
    var result = client.ListObjects(listObjectsRequest);
    Console.WriteLine("List objects succeeded");
    foreach (var summary in result.ObjectSummaries)
    {
        Console.WriteLine("File name:{0}", summary.Key);
    }
}
catch (Exception ex)
{
    Console.WriteLine("List objects failed. {0}", ex.Message);
```

```
}
```

List a specified number of objects

Use the following code to list a specified number of objects:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var listObjectsRequest = new ListObjectsRequest(bucketName)
    {
        // A maximum of 200 records can be returned.
        MaxKeys = 200,
    };
    var result = client.ListObjects(listObjectsRequest);
    Console.WriteLine("List objects succeeded");
    foreach (var summary in result.ObjectSummaries)
    {
        Console.WriteLine(summary.Key);
    }
}
catch (Exception ex)
{
    Console.WriteLine("List objects failed, {0}", ex.Message);
}
```

List objects by a specified prefix

Use the following code to list objects with a specified prefix:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var prefix = "<yourObjectPrefix>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var keys = new List<string>();
    ObjectListing result = null;
    string nextMarker = string.Empty;
    do
    {
        var listObjectsRequest = new ListObjectsRequest(bucketName)
        {
            Marker = nextMarker,
            MaxKeys = 100,
            Prefix = prefix,
        };
        result = client.ListObjects(listObjectsRequest);
        foreach (var summary in result.ObjectSummaries)
        {
```

```
            Console.WriteLine(summary.Key);
            keys.Add(summary.Key);
        }
        nextMarker = result.NextMarker;
    } while (result.IsTruncated);
    Console.WriteLine("List objects of bucket:{0} succeeded ",
bucketName);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

List objects by marker

The marker parameter indicates the name of an object from which the listing begins. Run the following code to specify an object (specified with marker) after which the listing begins:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var marker = "<yourObjectMarker>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var keys = new List<string>();
    ObjectListing result = null;
    string nextMarker = marker;
    do
    {
        var listObjectsRequest = new ListObjectsRequest(bucketName)
        // To increase the number of returned objects, you can modify
the MaxKeys parameter or use the Marker parameter for multiple reads.
        {
            Marker = nextMarker,
            MaxKeys = 100,
        };
        result = client.ListObjects(listObjectsRequest);
        foreach (var summary in result.ObjectSummaries)
        {
            Console.WriteLine(summary.Key);
            keys.Add(summary.Key);
        }
        nextMarker = result.NextMarker;
    // If the value of IsTruncated is true, the next read starts from
NextMarker.
    } while (result.IsTruncated);
    Console.WriteLine("List objects of bucket:{0} succeeded ",
bucketName);
}
```

```
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

List objects in an asynchronous mode

For the complete code of listing objects in an asynchronous mode, see *GitHub*.

Run the following code to list objects in an asynchronous mode:

```
using System;
using System.IO;
using System.Threading;
using Aliyun.OSS;
namespace AsyncListObjects
{
    class Program
    {
        static string endpoint = "<yourEndpoint>";
        static string accessKeyId = "<yourAccessKeyId>";
        static string accessKeySecret = "<yourAccessKeySecret>";
        static string bucketName = "<yourBucketName>";
        static AutoResetEvent _event = new AutoResetEvent(false);
        // Create an OSSClient instance.
        static OssClient client = new OssClient(endpoint, accessKeyId
, accessKeySecret);
        static void Main(string[] args)
        {
            Program.AsyncListObjects();
            Console.ReadKey();
        }
        public static void AsyncListObjects()
        {
            try
            {
                var listObjectsRequest = new ListObjectsRequest(
bucketName);
                client.BeginListObjects(listObjectsRequest, ListObject
Callback, null);
                _event.WaitOne();
            }
            catch (Exception ex)
            {
                Console.WriteLine("Async list objects failed. {0}", ex
.Message);
            }
        }
        // The ListObjectCallback method is implemented after an
asynchronous call. Similar interfaces must be used when an asynchrono
us interface is called to list objects.
        private static void ListObjectCallback(IAsyncResult ar)
        {
            try
            {
                var result = client.EndListObjects(ar);
```

```
                foreach (var summary in result.ObjectSummaries)
                {
                    Console.WriteLine ("Object name: 0", summary.Key);
                }
                _event.Set();
                Console.WriteLine("Async list objects succeeded");
            }
            catch (Exception ex)
            {
                Console.WriteLine("Async list objects failed. {0}", ex
.Message);
            }
        }
    }
}
```

List all objects in a bucket

Run the following code to list all objects in a bucket:

```
using Aliyun.OSS;
// Initialize OssClient.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
// List all objects in the bucket.
public void ListObject(string bucketName)
{
    try
    {
        ObjectListing result = null;
        string nextMarker = string.Empty;
        do
        {
            // The number of objects listed in each page is specified
 by the maxKeys parameter. If the object number is larger than the
specified value, other files are listed in another page.
            var listObjectsRequest = new ListObjectsRequest(bucketName
)
            {
                Marker = nextMarker,
                MaxKeys = 100
            };
            result = client.ListObjects(listObjectsRequest);
            Console.WriteLine("File:");
            foreach (var summary in result.ObjectSummaries)
            {
                Console.WriteLine("Name:{0}", summary.Key);
            }
            nextMarker = result.NextMarker;
        } while (result.IsTruncated);
    }
    catch (Exception ex)
    {
        Console.WriteLine("List object failed. {0}", ex.Message);
    }
```

```
}
```

8.8.6 Delete objects

This topic describes how to delete objects.

Warning:

Delete objects with caution because deleted objects cannot be recovered.

For the complete code of deleting objects, see *GitHub*.

Delete an object

Run the following code to delete a single object:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    /// Delete an object.
    client.DeleteObject(bucketName, objectName);
    Console.WriteLine("Delete object succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Delete object failed. {0}", ex.Message);
}
```

Delete multiple objects

You can delete a maximum of 1,000 objects simultaneously. Objects can be deleted in two modes: detail (verbose) and simple (quiet) modes.

- **verbose: returns the list of objects that you have deleted successfully. The default mode is verbose.**
- **quiet: returns the list of objects that you failed to delete.**

Run the following code to delete multiple objects simultaneously:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
```

```
    var keys = new List<string>();
    var listResult = client.ListObjects(bucketName);
    foreach (var summary in listResult.ObjectSummaries)
    {
        keys.Add(summary.Key);
    }
    // If the quietMode is true, the quiet mode is used. If the
quietMode is false, the verbose mode is used. The default mode is
verbose.
    var quietMode = false;
    // The third parameter of the DeleteObjectsRequest specifies the
return mode.
    var request = new DeleteObjectsRequest(bucketName, keys, quietMode
);
    // Delete multiple objects.
    var result = client.DeleteObjects(request);
    if ((! quietMode) && (result.Keys ! = null))
    {
        foreach (var obj in result.Keys)
        {
            Console.WriteLine("Delete successfully : {0} ", obj.Key);
        }
    }
    Console.WriteLine("Delete objects succeeded");
}
catch (Exception ex)
{
    Console.WriteLine("Delete objects failed. {0}", ex.Message);
}
```

8.8.7 Copy objects

Copy a small-sized object

For the complete code of copying a small-sized object, see *GitHub*.

Note the following limits before copying a small-sized object:

- **You must have the read and write permission on the source object.**
- **You cannot copy an object from a region to another region. For example, you cannot copy an object from a bucket in Hangzhou region to a bucket in Qingdao region.**
- **The object size is limited to a maximum of 1 GB.**

Run the following code to copy a small-sized object:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var sourceBucket = "<yourSourceBucketName>";
var sourceObject = "<yourSourceObjectName>";
var targetBucket = "<yourDestBucketName>";
var targetObject = "<yourDestObjectName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
```

```
{
    var metadata = new ObjectMetadata();
    metadata.AddHeader("mk1", "mv1");
    metadata.AddHeader("mk2", "mv2");
    var req = new CopyObjectRequest(sourceBucket, sourceObject,
targetBucket, targetObject)
    {
        // If the value of NewObjectMetadata is null, the COPY mode is
 used (the metadata of the source object is copied). If the value of
NewObjectMetadata is not null, the REPLACE mode is used (the metadata
of the source object is replaced).
        NewObjectMetadata = metadata
    };
    // Copy an object.
    client.CopyObject(req);
    Console.WriteLine("Copy object succeeded");
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}. \
nRequestID: {2} \tHostID: {3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

Copy a large-sized object

For the complete code of copying large-sized objects, see *GitHub*.

- **UploadPartCopy**

  To copy an object larger than 1 GB, you need to use UploadPartCopy.

  1. **Use InitiateMultipartUploadRequest to initiate an UploadPartCopy event.**
  2. **Use the UploadPartCopy method to perfome a multipart copy.**
  3. **Use the CompleteMultipartUpload method to complete the object copy.**

  Run the following code for UploadPartCopy task:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var sourceBucket = "<yourSourceBucketName>";
var sourceObject = "<yourSourceObjectName>";
var targetBucket = "<yourDestBucketName>";
var targetObject = "<yourDestObjectName>";
var uploadId = "";
var partSize = 50 * 1024 * 1024;
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Initiate a copy task. You can use the InitiateMultipartUpl
oadRequest method to specify the metadata information about the
target object.
```

```
    var request = new InitiateMultipartUploadRequest(targetBucket,
targetObject);
    var result = client.InitiateMultipartUpload(request);
    // Print the uploadId.
    uploadId = result.UploadId;
    Console.WriteLine("Init multipart upload succeeded,  Upload Id:
 {0}", result.UploadId);
    // Calculate the number of parts.
    var metadata = client.GetObjectMetadata(sourceBucket, sourceObje
ct);
    var fileSize = metadata.ContentLength;
    var partCount = (int)fileSize / partSize;
    if (fileSize % partSize ! = 0)
    {
        partCount++;
    }
    // Start the UploadPartCopy task.
    var partETags = new List<PartETag>();
    for (var i = 0; i < partCount; i++)
    {
        var skipBytes = (long)partSize * i;
        var size = (partSize < fileSize - skipBytes) ? partSize : (
fileSize - skipBytes);
        // Create UploadPartCopyRequest. You can specify conditions
with UploadPartCopyRequest.
        var uploadPartCopyRequest = new UploadPartCopyRequest(
targetBucket, targetObject, sourceBucket, sourceObject, uploadId)
            {
                PartSize = size,
                PartNumber = i + 1,
                // Beginindex is used to locate the location that
the last UploadPartCopy starts.
                BeginIndex = skipBytes
            };
        // Call the uploadPartCopy method to copy each part.
        var uploadPartCopyResult = client.UploadPartCopy(uploadPart
CopyRequest);
        Console.WriteLine("UploadPartCopy : {0}", i);
        partETags.Add(uploadPartCopyResult.PartETag);
    }
    // Complete the UploadPartCopy task.
    var completeMultipartUploadRequest =
    new CompleteMultipartUploadRequest(targetBucket, targetObject,
uploadId);
    // The partETags refers to the list of partETags saved during
 the multipart upload. After OSS receives the part list submitted
by the user, it verifies the validity of each data part one by one
. After the verification is passed, OSS conbine these parts into a
complete object.
    foreach (var partETag in partETags)
    {
        completeMultipartUploadRequest.PartETags.Add(partETag);
    }
    var completeMultipartUploadResult = client.CompleteMultipartUpl
oad(completeMultipartUploadRequest);
    Console.WriteLine("CompleteMultipartUpload succeeded");
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}.
 \nRequestID: {2} \tHostID: {3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
```

```
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

- **Resumable copy**

  If the copy is interrupted, you can use resumable copy to continue the copy. For the complete code of resumable copy, see *GitHub*.

  Run the following code for resumable copy:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var sourceBucket = "<yourSourceBucketName>";
var sourceObject = "<yourSourceObjectName>";
var targetBucket = "<yourDestBucketName>";
var targetObject = "<yourDestObjectName>";
var checkpointDir = @"<yourCheckpointDir>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var request = new CopyObjectRequest(sourceBucket, sourceObject,
targetBucket, targetObject);
    // The checkpointDir directory stores the intermediate state
information of a resumable upload. The information will be used when
 a failed upload task is resumed. // If the checkpointDir directory
 is null, resumable copy will not take effect, and the object that
previously failed to be copied will be copied all over again.
    client.ResumableCopyObject(request, checkpointDir);
    Console.WriteLine("Resumable copy new object:{0} succeeded",
request.DestinationKey);
}
catch (Exception ex)
{
    Console.WriteLine("Resumable copy new object failed, {0}", ex.
Message);
}
```

8.8.8 Restore an archive object

For the complete code of restoring an archive object, see *GitHub*.

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

- **An archive object is in the frozen state.**
- **After you submit it for restoration, the server restores the object. The object is in the restoring state.**

- **You can read the object after it is restored. The restored state of the object lasts one day by default. You can prolong this period to a maximum of seven days. Once this period expires, the object returns to the frozen state.**

Run the following code to restore an object:

```
using Aliyun.OSS;
using Aliyun.OSS.Model;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var objectContent = "More than just cloud.";
int maxWaitTimeInSeconds = 600;
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Create a bucket of the archive storage class..
    var bucket = client.CreateBucket(bucketName, StorageClass.Archive
);
    Console.WriteLine("Create Archive bucket succeeded, {0} ", bucket.
Name);
}
catch (Exception ex)
{
    Console.WriteLine("Create Archive bucket failed, {0}", ex.Message
);
}
// Upload a file.
try
{
    byte[] binaryData = Encoding.ASCII.GetBytes(objectContent);
    MemoryStream requestContent = new MemoryStream(binaryData);
    client.PutObject(bucketName, objectName, requestContent);
    Console.WriteLine("Put object succeeded, {0}", objectName);
}
catch (Exception ex)
{
    Console.WriteLine("Put object failed, {0}", ex.Message);
}
var metadata = client.GetObjectMetadata(bucketName, objectName);
string storageClass = metadata.HttpMetadata["x-oss-storage-class"] as
string;
if (storageClass ! = "Archive")
{
    Console.WriteLine("StorageClass is {0}", storageClass);
    return;
}
// Restore the object.
RestoreObjectResult result = client.RestoreObject(bucketName,
objectName);
Console.WriteLine("RestoreObject result HttpStatusCode : {0}", result.
HttpStatusCode);
if (result.HttpStatusCode ! = HttpStatusCode.Accepted)
{
    throw new OssException(result.RequestId + ", " + result.HttpStatus
Code + " ,");
}
while (maxWaitTimeInSeconds > 0)
```

```
{
    var meta = client.GetObjectMetadata(bucketName, objectName);
    string restoreStatus = meta.HttpMetadata["x-oss-restore"] as
string;
    if (restoreStatus ! = null && restoreStatus.StartsWith("ongoing-
request=\"false\"", StringComparison.InvariantCultureIgnoreCase))
    {
        break;
    }
    Thread.Sleep(1000);
    maxWaitTimeInSeconds--;
}
if (maxWaitTimeInSeconds == 0)
{
    Console.WriteLine("RestoreObject is timeout. ");
    throw new TimeoutException();
}
else
{
    Console.WriteLine("RestoreObject is successful. ");
}
```

For more information about the archive storage classes, see *Introduction to storage classes*. **For detailed information about related status code, see** *RestoreObject*.

8.8.9 Manage a symbolic link

A symbolic link is a special object that maps to an object similar to a shortcut used in Windows. You can configure user-defined Object Meta for symbolic links. To obtain a symbolic link, you must have the read permission on it.

Run the following code to create and obtain a symbolic link:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var targetObjectName = "<yourTargetObjectName>";
var symlinkObjectName = "<yourSymlinkObjectName>";
var objectContent = "More than just cloud." ;
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Upload the target file.
    byte[] binaryData = Encoding.ASCII.GetBytes(objectContent);
    MemoryStream requestContent = new MemoryStream(binaryData);
    client.PutObject(bucketName, targetObjectName, requestContent);
    // Create a symbolic link.
    client.CreateSymlink(bucketName, symlinkObjectName, targetObje
ctName);
    // Obtain a symbolic link.
    var ossSymlink = client.GetSymlink(bucketName, symlinkObjectName);
    Console.WriteLine("Target object is {0}", ossSymlink.Target);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
```

```
}
```

For more information about creating symbolic links, see *PutSymlink*.

For more information about obtaining symbolic links, see *GetSymlink*.

8.9 Authorized access

Use STS for temporary access authorization

OSS supports Alibaba Cloud Security Token Service (STS) for temporary access authorization. STS is a web service that provides a temporary access token to a cloud computing user. Through the STS, you can assign a third-party application or a RAM user (you can manage the user ID) an access credential with a custom validity period and permissions. For more information about STS, see *STS introduction*.

STS advantages:

- **Your long-term key (AccessKey) is not exposed to a third-party application. You only need to generate an access token and send the access token to the third-party application. You can customize access permissions and the validity of this token.**
- **You do not need to keep track of permission revocation issues. The access token automatically becomes invalid when it expires.**

For more information about the process of access to OSS with STS, see *RAM and STS scenario practices* in OSS Developer Guide.

Run the following code to create a signature request with STS:

```
using Aliyun.OSS;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var securityToken = "<yourSecurityToken>";
// After a user obtains a temporary STS credential, the OSSClient is
generated with the security token and temporary access key (AccessKeyI
D and AccessKeySecret).
// Create an OSSClient instance.
var ossStsClient = new OssClient(endpoint, accessKeyId, accessKeyS
ecret, securityToken);
```

```
// Perform operations on OSS.
```

Sign a URL to authorize temporary access

You can provide a signed URL to a visitor for temporary access. When you sign a URL, you can specify the expiration time for a URL to restrict the period of access from visitors.

For the complete code of signing a URL to authorize temporary access, see *GitHub*.

- **Use a signed URL to upload a file**

  Run the following code to upload a file with a signed URL:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var objectContent = "More than just cloud." ;
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    // Generate a signed URL for upload.
    var generatePresignedUriRequest = new GeneratePresignedUri
Request(bucketName, objectName, SignHttpMethod.Put)
    {
        Expiration = DateTime.Now.AddHours(1),
    };
    var signedUrl = client.GeneratePresignedUri(generatePresignedUri
Request);
    // Upload a file with the signed URL.
    var buffer = Encoding.UTF8. GetBytes(objectContent);
    using (var ms = new MemoryStream(buffer))
    {
        client.PutObject(signedUrl, ms);
    }
    Console.WriteLine("Put object by signatrue succeeded. {0} ",
signedUrl.ToString());
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}.
 \nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

- **Use a signed URL to download an object**

  Run the following code to download an object with a signed URL:

```
using Aliyun.OSS;
```

```
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
var downloadFilename = "<yourDownloadFilename>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var metadata = client.GetObjectMetadata(bucketName, objectName);
    var etag = metadata.ETag;
    // Generate a signed URL for download.
    var req = new GeneratePresignedUriRequest(bucketName, objectName
, SignHttpMethod.Get);
    var uri = client.GeneratePresignedUri(req);
    // Download an object with the signed URL.
    OssObject ossObject = client.GetObject(uri);
    using (var file = File.Open(downloadFilename, FileMode.
OpenOrCreate))
    {
        using (Stream stream = ossObject.Content)
        {
            int length = 4 * 1024;
            var buf = new byte[length];
            do
            {
                length = stream.Read(buf, 0, length);
                file.Write(buf, 0, length);
            } while (length ! = 0);
        }
    }
    Console.WriteLine("Get object by signatrue succeeded. {0} ", uri
.ToString());
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}.
 \nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
}
```

8.10 Image processing

Image Processing (IMG) is a massive, secure, cost-effective and highly reliable image processing service. After source images are uploaded to OSS, you can process images on any Internet device at any anytime, from anywhere through simple RESTful APIs.

For more information about IMG, see *Image Processing*.

Basic features

OSS offers the following IMG features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**
- *Image processing*
- *Image processing access rules*: **calls multiple IMG functions.**

Usage

IMG uses standard HTTP GET. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is private read and write, only authorized users are allowed for access.

- **Anonymous access**

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Specifies the name of a bucket.
endpoint	Specifies endpoint used to access a region.
object	Specifies the name of an image object.
image	Specifies the reserved identifier for IMG.
action	Specifies the operations on an image, such as scaling, cropping, and rotating .

Parameter	Description
param	Specifies the parameter that indicates the operation on an image.

- Basic operations

  For example, scale an image to a width of 100 px. Adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=image/resize,w_100
```

- Customized image styles

  You can use the following format in level-3 domains to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-process=style/<yourStyleName>
```

  - style: specifies a reserved identifier of a customized image style.
  - yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=style/oss-pic-style-w-100
```

- Cascade operations

  Cascade operations allow you to perform multiple operations on an image sequentially.

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<yourParamValue2>/...
```

  Example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=image/resize,w_100/rotate,90
```

- HTTPS access

IMG supports access through HTTPS. Example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  Authorized access allows you to customize an image style, HTTPS access, and cascade operations.

  Run the following code to generate a signed URL for IMG:

```
using Aliyun.OSS;
using Aliyun.OSS.Common;
var endpoint = "<yourEndpoint>";
var accessKeyId = "<yourAccessKeyId>";
var accessKeySecret = "<yourAccessKeySecret>";
var bucketName = "<yourBucketName>";
var objectName = "<yourObjectName>";
// Create an OSSClient instance.
var client = new OssClient(endpoint, accessKeyId, accessKeySecret);
try
{
    var process = "image/resize,m_fixed,w_100,h_100";
    var req = new GeneratePresignedUriRequest(bucketName, objectName
, SignHttpMethod.Get)
    {
        Expiration = DateTime.Now.AddHours(1),
        Process = process
    };
    // Generat a signed URI.
    var uri = client.GeneratePresignedUri(req);
    Console.WriteLine("Generate Presigned Uri:{0} with process:{1}
succeeded ", uri, process);
}
catch (OssException ex)
{
    Console.WriteLine("Failed with error code: {0}; Error info: {1}.
 \nRequestID:{2}\tHostID:{3}",
        ex.ErrorCode, ex.Message, ex.RequestId, ex.HostId);
}
catch (Exception ex)
{
    Console.WriteLine("Failed with error info: {0}", ex.Message);
```

```
}
```

- **Access with SDK**

  You can use SDK to access and process an image object.

  SDK allows you to customize image styles, HTTPS access, and cascade operations .

  - **Basic operations**

    Run the following code to perform basic operations on an image:

```
using System;
using System.IO;
using Aliyun.OSS;
using Aliyun.OSS.Common;
using Aliyun.OSS.Util;
namespace ImageProcess
{
    class Program
    {
        static void Main(string[] args)
        {
            Program.ImageProcess();
            Console.ReadKey();
        }
        public static void ImageProcess()
        {
            var endpoint = "<yourEndpoint>";
            var accessKeyId = "<yourAccessKeyId>";
            var accessKeySecret = "<yourAccessKeySecret>";
            var bucketName = "<yourBucketName>";
            var objectName = "<yourObjectName>";
            var localImageFilename = "<yourLocalImageFilename>";
            var downloadDir = "<yourDownloadDir>";
            // Create an OSSClient instance.
            var client = new OssClient(endpoint, accessKeyId,
accessKeySecret);
            try
            {
                client.PutObject(bucketName, objectName,
localImageFilename);
                // Scale the image.
                var process = "image/resize,m_fixed,w_100,h_100";
                var ossObject = client.GetObject(new GetObjectR
equest(bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-resize.jpg",
ossObject.Content);
                // Crop the image.
                process = "image/crop,w_100,h_100,x_100,y_100,r_1
";
                ossObject = client.GetObject(new GetObjectRequest(
bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-crop.jpg",
ossObject.Content);
                // Rotate the image.
                process = "image/rotate,90";
                ossObject = client.GetObject(new GetObjectRequest(
bucketName, objectName, process));
```

```
                WriteToFile(downloadDir + "/sample-rotate.jpg",
ossObject.Content);
                // Sharpen the image.
                process = "image/sharpen,100";
                ossObject = client.GetObject(new GetObjectRequest(
bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-sharpen.jpg",
ossObject.Content);
                // Add watermarks.
                process = "image/watermark,text_SGVsbG8g5Zu-
54mH5pyN5YqhIQ";
                ossObject = client.GetObject(new GetObjectRequest(
bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-watermark.jpg
", ossObject.Content);
                // Convert the image format.
                process = "image/format,png";
                ossObject = client.GetObject(new GetObjectRequest(
bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-format.png",
ossObject.Content);
                // Obtain image information.
                process = "image/info";
                ossObject = client.GetObject(new GetObjectRequest(
bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-info.txt",
ossObject.Content);
                Console.WriteLine("Get Object:{0} with process:{1
} succeeded ", objectName, process);
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0};
Error info: {1}. \nRequestID:{2}\tHostID:{3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.
HostId);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}",
ex.Message);
            }
        }
        private static void WriteToFile(string filePath, Stream
stream)
        {
            using (var requestStream = stream)
            {
                using (var fs = File.Open(filePath, FileMode.
OpenOrCreate))
                {
                    IoUtils.WriteTo(stream, fs);
                }
            }
        }
    }
}
```

- **Customize an image style**

  Use the following code to customize an image style:

```
using System;
```

```
using System.IO;
using Aliyun.OSS;
using Aliyun.OSS.Common;
using Aliyun.OSS.Util;
namespace ImageProcessCustom
{
    class Program
    {
        static void Main(string[] args)
        {
            Program.ImageProcessCustomStyle();
            Console.ReadKey();
        }
        public static void ImageProcessCustomStyle()
        {
            var endpoint = "<yourEndpoint>";
            var accessKeyId = "<yourAccessKeyId>";
            var accessKeySecret = "<yourAccessKeySecret>";
            var bucketName = "<yourBucketName>";
            var objectName = "<yourObjectName>";
            var localImageFilename = "<yourLocalImageFilename>";
            var downloadDir = "<yourDownloadDir>";
            // Create an OSSClient instance.
            var client = new OssClient(endpoint, accessKeyId,
accessKeySecret);
            try
            {
                client.PutObject(bucketName, objectName,
localImageFilename);
                // Customize the image style: "style/<yourCustom
StyleName>".
                var process = "style/oss-pic-style-w-100";
                var ossObject = client.GetObject(new GetObjectR
equest(bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-style.jpg",
ossObject.Content);
                Console.WriteLine("Get Object:{0} with process:{1
} succeeded ", objectName, process);
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0};
Error info: {1}. \nRequestID:{2}\tHostID:{3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.
HostId);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}",
ex.Message);
            }
        }
        private static void WriteToFile(string filePath, Stream
stream)
        {
            using (var requestStream = stream)
            {
                using (var fs = File.Open(filePath, FileMode.
OpenOrCreate))
                {
                    IoUtils.WriteTo(stream, fs);
                }
            }
        }
```

```
    }
}
```

- **Cascade operations**

  Use the following code to perform cascade operations on an image:

```
using System;
using System.IO;
using Aliyun.OSS;
using Aliyun.OSS.Common;
using Aliyun.OSS.Util;
namespace ImageProcessCascade
{
    class Program
    {
        static void Main(string[] args)
        {
            Program.ImageProcessCascade();
            Console.ReadKey();
        }
        public static void ImageProcessCascade()
        {
            var endpoint = "<yourEndpoint>";
            var accessKeyId = "<yourAccessKeyId>";
            var accessKeySecret = "<yourAccessKeySecret>";
            var bucketName = "<yourBucketName>";
            var objectName = "<yourObjectName>";
            var localImageFilename = "<yourLocalImageFilename>";
            var downloadDir = "<yourDownloadDir>";
            // Create an OSSClient instance.
            var client = new OssClient(endpoint, accessKeyId,
accessKeySecret);
            try
            {
                client.PutObject(bucketName, objectName,
localImageFilename);
                // Perform cascade operations.
                var process = "image/resize,m_fixed,w_100,h_100/
rotate,90";
                var ossObject = client.GetObject(new GetObjectR
equest(bucketName, objectName, process));
                WriteToFile(downloadDir + "/sample-style.jpg",
ossObject.Content);
                Console.WriteLine("Get Object:{0} with process:{1
} succeeded ", objectName, process);
            }
            catch (OssException ex)
            {
                Console.WriteLine("Failed with error code: {0};
Error info: {1}. \nRequestID:{2}\tHostID:{3}",
                    ex.ErrorCode, ex.Message, ex.RequestId, ex.
HostId);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}",
ex.Message);
            }
        }
        private static void WriteToFile(string filePath, Stream
stream)
        {
```

```
            using (var requestStream = stream)
            {
                using (var fs = File.Open(filePath, FileMode.
OpenOrCreate))
                {
                    IoUtils.WriteTo(stream, fs);
                }
            }
        }
    }
}
```

IMG tools

You can use the IMG viewer *ImageStyleViever* to directly view the IMG result.

8.11 Exception handling

OSS .NET SDK has two types of exceptions: ClientException and OSSException. Both share the properties of RuntimeException.

ClientException

The ClientException exception occurs when the client attempts to send requests and when the data is transmitted to OSS. For example, when a request is sent, ClientException may occur due to unavailable network connections. ClientException may occur during file uploads due to I/O exceptions.

OSSException

OSSException indicates a server exception. The exception occurs because a server error message is parsed. OSSException includes the error code and information returned from OSS. It is used to locate and handle problems.

The following table describes common OSSException information.

Parameter	Description
Code	Specifies the error code returned by OSS .
Message	Specifies the detailed error information returned by OSS.
RequestId	Specifies the UUID. It is unique and used to identify a request. If a problem persists, send the RequestId to OSS developers for help.

Parameter	Description
HostId	Identifies an OSS cluster. Ensure that the value of HostId is consistent with the Host (endpoint to access a bucket) used in the request.

OSS error codes

Error code	Description
AccessDenied	Access denied
BucketAlreadyExists	The bucket already exists.
BucketNotEmpty	The bucket is not empty.
EntityTooLarge	The entity size exceeds the maximum limit.
EntityTooSmall	The entity size is below the minimum limit.
FileGroupTooLarge	The total file group size exceeds the maximum limit.
FilePartNotExist	No such part exists.
FilePartStale	The part has expired.
InvalidArgument	The parameter format is invalid.
InvalidAccessKeyId	No such AccessKeyId exists.
InvalidBucketName	The bucket name is invalid.
InvalidDigest	The digest is invalid.
InvalidObjectName	The object name is invalid.
InvalidPart	The part is invalid.
InvalidPartOrder	The part order is invalid.
InvalidTargetBucketForLogging	The buckets that store log files are invalid.
InternalError	Internal OSS error
MalformedXML	The XML format is invalid.
MethodNotAllowed	The method is not allowed.
MissingArgument	Parameters are not configured.
MissingContentLength	The content length is not configured.

Error code	Description
NoSuchBucket	No such bucket exists.
NoSuchKey	No such object exists.
NoSuchUpload	No such uploadId exists.
NotImplemented	The methods are not implemented.
PreconditionFailed	Precondition error
RequestTimeTooSkewed	The local time set for OSSClient is deviated from the time set for the OSS server by over 15 minutes.
RequestTimeout	Request timeout
SignatureDoesNotMatch	Signature mismatch
TooManyBuckets	The number of buckets exceeds the limit.

9 Android

9.1 Preface

This topic is based on OSS Android SDK 2.9.1.

Compatibility

Later versions of OSS Android SDK are backward-compatible with earlier versions.

Source code

For the source code of OSS Android SDK, see *GitHub*.

Code samples

OSS SDK for Android provides a variety of code samples for your reference.

The following table describes the content of code samples.

Sample file	Content
BaseTestCase.java	*Initialization*
OSSPutObjectTest.java	*Upload objects*
OSSGetObjectTest.java	*Download objects*
ManageObjectTest.java	*Manage objects*
OSSBucketTest.java	*Bucket*
ResumableUploadTest.java	*Resumable upload*
CRC64Test.java	*Data security*
ImagePersistTest.java	*Image processing*

9.2 Installation

This topic describes how to install OSS Android SDK.

Environment requirements

- **Android system version: 2.3 or later**
- **You must have an Alibaba Cloud account and activate the OSS service.**

Download OSS Android SDK

- **Add OSS Android SDK.**

  Add the following dependency in Android Studio:

```
dependencies { compile 'com.aliyun.dpa:oss-android-sdk:+' }
```

- **Download OSS Android SDK from *Github*.**

Install OSS Android SDK

- **Method 1 (recommended): Add the following dependency in your Maven project.**

```
dependencies {
 compile 'com.aliyun.dpa:oss-android-sdk:+'
}
```

- **Method 2: Compile the source code into a jar package.**

  1. **Clone the source code of the project and run the `gradle` command to compile the code into a jar package:**

```
# Clone the project.
$ git clone https://github.com/aliyun/aliyun-oss-android-sdk.git

# Enter the directory of the project.
$ cd aliyun-oss-android-sdk/oss-android-sdk/

# Run the package script. Note that jdk 1.7 is required.
$ ../gradlew releaseJar

# Enter the directory where the generated jar package is stored.
$ cd build/libs && ls
```

  2. **Directly introduce the generated jar package.**

     Introduce the following three jar packages to the *libs* directory: aliyun-oss-sdk-android-x.x.x.jar, *okhttp-3.x.x.jar*, and *okio-1.x.x.jar*.

Related settings

- **Permission settings**

  The following permissions are required to use the OSS Android SDK.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NET
WORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE
" />
<uses-permission android:name="android.permission.WRITE_EXTE
RNAL_STORAGE" />
```

```
<uses-permission android:name="android.permission.MOUNT_UNMO
UNT_FILESYSTEMS" />
```

Note:

Ensure the preceding permissions are set in your AndroidManifest.xml file to let SDK work normally.

- **Proguard settings**

  Add the following code to the Proguard settings:

```
-keep class com.alibaba.sdk.android.oss.** { *; }
-dontwarn okio.**
-dontwarn org.apache.commons.codec.binary. **
```

9.3 Initialization

OSSClient is the client of OSS for Android systems. It provides the caller with a series of methods to operate and manage the buckets and objects. Before you use the OSS Android SDK to initiate a request to OSS, you must first initialize an OSSClient instance and complete necessary settings.

Note:

Make sure that the life cycle of the OSSClient is consistent with that of the application. Create a global OSSClient instance when an application starts, and destroy it when the application ends.

Determine an endpoint

An endpoint is the address of Alibaba Cloud OSS in a region. It supports the following two formats:

Endpoint type	Description
OSS region address	Address of the region where an OSS bucket is located. For more information about the endpoints in various regions, see *Regions and endpoints*.
User-defined domain name	Domain name defined by the user, with the CNAME directing to the OSS domain

- **OSS region address**

  You can use one of the following two methods to search for an endpoint mapped to the address of the region where an OSS bucket is located:

  - **You can query the mapping relationship between the endpoint and the region. For more information, see** *Regions and endpoints*.
  - **You can log on to** *Alibaba Cloud OSS Console*, **open the Bucket Overview page, and find the suffix of the bucket domain. For example, the suffix** `oss-cn-hangzhou.aliyuncs.com` **of the bucket domain** `bucket-1.oss-cn-hangzhou.aliyuncs.com` **is the endpoint of the bucket on the Internet.**

- **CNAME**

  You can bind your domain to a bucket through the CNAME and access the objects in the bucket through the domain.

  For example, you want to bind the domain new-image.xxxxx.com to a bucket named "image" in the Shenzhen region: you must contact your domain hosting provider for xxxxx.com to configure a new domain name resolution record used to resolve `http://new-image.xxxxx.com` **to** `http://image.oss-cn-shenzhen.aliyuncs.com`. **The record type is CNAME.**

Configure the endpoint and credential

A mobile client is an untrusted environment. An extremely high risk occurs if you store the `AccessKeyId` **and** `AccessKeySecret` **in a mobile client for signing requests. We recommend you use** `STS Authentication Mode` **or** `Self-signed Mode`. **For more information, see** *Authorized access* **and** *Set up direct data transfer for mobile apps*.

Note:
If you are using STS authentication mode, we recommend you use OSSAuthCredentialsProvider to access the application server. Expired tokens are updated automatically.

For more information about sample projects, see *GitHub*.

An example of EndPoint and CredentialProvider settings is shown as follows:

```
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

String stsServer = "STS application server address, such as http://abc
.com"
```

```
//We recommend you use OSSAuthCredentialsProvider because expired
tokens can be updated quickly.
OSSCredentialProvider credentialProvider = new OSSAuthCredentialsPr
ovider(stsServer);

//If this configuration class is not set, default configuration
applies.
ClientConfiguration conf = new ClientConfiguration();
conf.setConnectionTimeout(15 * 1000); // Connection time-out. The
default value is 15 seconds.
conf.setSocketTimeout(15 * 1000); // Socket time-out. The default
value is 15 seconds.
conf.setMaxConcurrentRequest(5); // The maximum number of concurrent
requests. The default value is 5.
conf.setMaxErrorRetry(2); // The maximum number of retries after a
failure. The default value is 2.

OSS oss = new OSSClient(getApplicationContext(), endpoint, credential
Provider);
```

Note:

For more authentication modes, see *Authorized access*.

Configure the endpoint according to the CNAME

If you have bound the CNAME to the bucket, you can direct the endpoint to the CNAME directly. For example,

```
String endpoint = "http://new-image.xxxxx.com";
...
OSS oss = new OSSClient(getApplicationContext(), endpoint, credential
Provider);
```

Enable logging

Mobile terminals have complicated environments during usage. The OSS SDK may fail to function in a region or during a period of time. To help developers locate issues, the OSS SDK records some logs to a local directory after the log feature is enabled. To enable the log feature, you must initialize the OSSClient before using it, by calling the following method.

```
//The log style.
//You can call the OSSLog.enableLog() method to enable displaying logs
 in the console.
//This setting also supports saving one copy of the log file to the
SD card of the mobile phone at \OSSLog\logs.csv. This setting is not
enabled by default.
//The log records the request data, returned data and exceptions of
OSS operations.
//Such as requestId and response header. The following is a log record
 case.
//android_version: 5.1. The Android version
//mobile_model: XT1085 . The Android phone model
```

```
//network_state: connected. The network status
//network_type: WIFI. The network connection type
//Specific operation information:
//[2017-09-05 16:54:52] - Encounter local exception: //java.lang.
IllegalArgumentException: The bucket name is invalid.
//A bucket name must:
//1) be comprised of lower-case characters, numbers or dash(-);
//2) start with lower case or numbers;
//3) be between 3-63 characters long.
//------>end of log
OSSLog.enableLog(); //Call this method to enable logs.
```

Note:

- **Log files are stored on the SD card at** *\OSSLog\logs.csv*.
- **You can upload the file to the server to further trace the issue.**
- **You can also upload the log files after connecting to Alibaba Cloud Log Service. Go to** *Log Service*.

Set network parameters

You can set detailed ClientConfiguration during the initialization process as follows :

```
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

// We recommend you initialize OSSClient using STS on the mobile
client. For more information, see: [Resource Access Management].
OSSCredentialProvider credentialProvider = new OSSStsTokenCredentia
lProvider("<StsToken.AccessKeyId>", "<StsToken.SecretKeyId>", "<
StsToken.SecurityToken>");


ClientConfiguration conf = new ClientConfiguration();
conf.setConnectionTimeout(15 * 1000); // Connection time-out. The
default value is 15 seconds.
conf.setSocketTimeout(15 * 1000); // Socket time-out. The default
value is 15 seconds.
conf.setMaxConcurrentTaskNum(5); // The maximum number of concurrent
requests. The default value is 5.
conf.setMaxErrorRetry(2); // The maximum number of retries after a
failure. The default value is 2.

OSS oss = new OSSClient(getApplicationContext(), endpoint, credential
Provider, conf);
```

Synchronous and asynchronous interfaces in the SDK

Network requests are not allowed to be processed in the UI thread in mobile development, so most SDK interfaces support synchronous and asynchronous calls to handle requests. After it has been called, the synchronous interface blocks other

requests while waiting for the returned results. The asynchronous interface has to import a callback function that processes the result of requests.

Synchronous interfaces cannot be called on the UI thread. If a ClientException or ServiceException occurs, it is thrown immediately. ClientException refers to local exceptions, including network exceptions, invalid parameters, and so on. ServiceException refers to service exceptions returned by OSS, such as authentica tion failures and server errors, among others.

If an exception occurs while an asynchronous request is being processed, it is handled by a callback function.

When an asynchronous interface is called, the function returns a task. You can cancel a task, wait for it to finish, or get the results directly. For example:

```
OSSAsyncTask task = oss.asyncGetObejct(...);
task.cancel(); // Cancel the task
task.waitUntilFinished(); // Wait until the task is finished
GetObjectResult result = task.getResult(); // Block other requests and
 wait for the returned results
```

Note:

The interface supports both the synchronous and asynchronous call methods. For simplicity, this document only provides examples of how to call important interfaces in synchronous and asynchronous modes. For other interfaces, only asynchronous call examples are provided.

Set whether to enable the DNS settings

```
ClientConfiguration conf = new ClientConfiguration();
conf.setHttpDnsEnable(true);//The default value is "true", meaning
enable. To disable the feature, set the value to "false".
```

Set the custom user-agent

```
ClientConfiguration conf = new ClientConfiguration();
//The set ua is added to the end of the default ua of the SDK by
default. Use the last ua value.
conf.setUserAgentMark("customUserAgent");
```

9.4 Quick start

This topic describes how to use OSS Android SDK to quickly perform common OSS operations, such as creating a bucket, uploading an object, and downloading an object.

You can use the sample project in the following two methods:

- **View the *sample* directory which includes sample projects of uploading local file, downloading objects, resumable upload, and setting callback. For more information, see *Github*.**
- **Run the `git clone` command to clone the *project*.**

Before running the project, you must configure the Config parameter. The sample code of configuring this parameter is as follows:

```
public class Config {

    // To run the sample correctly, the following variables must have
valid values.
    // The endpoint value below is just the example. Please use proper
 value according to your region

    // Specify the endpoint that you want to access.
    public static final String OSS_ENDPOINT = "http://oss-cn-shanghai.
aliyuncs.com";
    // Specify the URL used to test the callback function.
    public static final String OSS_CALLBACK_URL = "http://oss-demo.
aliyuncs.com:23450";
    // Specify the address of the authentication server.
    // You can also start the local STS authentication server by
 running the local authentication script in the sts_local_server
directory of the project.
    public static final String STS_SERVER_URL = "http://****/sts/
getsts";//Specify the URL of the STS authentication server.

    public static final String BUCKET_NAME = "Input the bucket name.";
    public static final String OSS_ACCESS_KEY_ID = "<yourAccessKeyId
>";;
    public static final String OSS_ACCESS_KEY_SECRET = "<yourAccess
KeySecret>";

    public static final int DOWNLOAD_SUC = 1;
    public static final int DOWNLOAD_Fail = 2;
    public static final int UPLOAD_SUC = 3;
    public static final int UPLOAD_Fail = 4;
    public static final int UPLOAD_PROGRESS = 5;
    public static final int LIST_SUC = 6;
    public static final int HEAD_SUC = 7;
    public static final int RESUMABLE_SUC = 8;
    public static final int SIGN_SUC = 9;
    public static final int BUCKET_SUC = 10;
    public static final int GET_STS_SUC = 11;
```

```
    public static final int MULTIPART_SUC = 12;
    public static final int STS_TOKEN_SUC = 13;
    public static final int FAIL = 9999;
    public static final int REQUESTCODE_AUTH = 10111;
    public static final int REQUESTCODE_LOCALPHOTOS = 10112;
}
```

Note:

- **For more information about how to configure the URL of the STS authentication server, see** *Set up direct data transfer for mobile apps*.
- **For more information about the script in the sts_local_server directory, see** *GitHub*.

Create a bucket

A bucket is a container for objects stored in OSS. Every object is contained in a bucket.

Run the following code to create a bucket:

```
CreateBucketRequest createBucketRequest = new CreateBucketRequest("<
bucketName>");
// Set the ACL for the bucket to public-read. The default ACL for a
created bucket is private.
createBucketRequest.setBucketACL(CannedAccessControlList.PublicRead);
// Specify the region where the bucket belongs to.
createBucketRequest.setLocationConstraint("oss-cn-hangzhou");
OSSAsyncTask createTask = oss.asyncCreateBucket(createBucketRequest,
new OSSCompletedCallback<CreateBucketRequest, CreateBucketResult>() {
    @Override
    public void onSuccess(CreateBucketRequest request, CreateBuck
etResult result) {
        Log.d("locationConstraint", request.getLocationConstraint());
        }
    @Override
    public void onFailure(CreateBucketRequest request, ClientException
 clientException, ServiceException serviceException) {
        // Handle the exceptions returned for the request.
        if (clientException != null) {
            // A local exception (such as network exception) occurs.
            clientException.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
```

For the naming conventions for a bucket, see *Basic concepts*. **For more information about how to create a bucket, see** *Manage buckets*.

For more information about the endpoints, see *Regions and endpoints*.

Upload an object

Run the following code to upload a specified local file to OSS:

```
// Construct an upload request.
PutObjectRequest put = new PutObjectRequest("<bucketName>", "<
objectName>", "<uploadFilePath>");

// You can configure the callback function when calling the asynchorou
s interface to upload a file.
put.setProgressCallback(new OSSProgressCallback<PutObjectRequest>() {
    @Override
    public void onProgress(PutObjectRequest request, long currentSize
, long totalSize) {
        Log.d("PutObject", "currentSize: " + currentSize + " totalSize
: " + totalSize);
    }
});

OSSAsyncTask task = oss.asyncPutObject(put, new OSSCompletedCallback<
PutObjectRequest, PutObjectResult>() {
    @Override
    public void onSuccess(PutObjectRequest request, PutObjectResult
result) {
        Log.d("PutObject", "UploadSuccess");
        Log.d("ETag", result.getETag());
        Log.d("RequestId", result.getRequestId());
    }

    @Override
    public void onFailure(PutObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // Handle the exceptions returned for the request.
        if (clientExcepion != null) {
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
// task.cancel(); // You can cancel the upload task.
// task.waitUntilFinished(); // You can also wait until the task is
complete.
```

Download a specified object

Run the following code to download a specified object to the local device.

```
// Construct a download request.
GetObjectRequest get = new GetObjectRequest("<bucketName>", "<
objectName>");
```

```
OSSAsyncTask task = oss.asyncGetObject(get, new OSSCompletedCallback<
GetObjectRequest, GetObjectResult>() {
    @Override
    public void onSuccess(GetObjectRequest request, GetObjectResult
result) {
        // The request is successful.
        Log.d("asyncGetObject", "DownloadSuccess");
        Log.d("Content-Length", "" + result.getContentLength());

        InputStream inputStream = result.getObjectContent();
        byte[] buffer = new byte[2048];
        int len;

        try {
            while ((len = inputStream.read(buffer)) != -1) {
                // You can add your own code here to process the
downloaded data.
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    @Override
 // If the GetObject request is successful, GetObjectResult which
includes an instance of the input stream is returned. Add your own
code to process the input stream.
    public void onFailure(GetObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // Hanlde exceptions returned to the request.
        if (clientExcepion != null) {
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});

// task.cancel(); // You can cancel the download task.
// task.waitUntilFinished(); // You can also wait untile the task is
complete.
```

Note:

You must process the returned input stream by running your own code.

9.5 Buckets

9.5.1 Manage buckets

A bucket serves as a container that stores objects. Objects belong to a bucket.

OSS uses buckets as the namespaces and management objects for advanced features such as billing, permission control, and logging. The bucket name is globally unique throughout OSS and cannot be changed. Every object stored on OSS must be included in a bucket.

Create a bucket

The following code creates a bucket:

```
CreateBucketRequest createBucketRequest = new CreateBucketRequest("<
bucketName>");
createBucketRequest.setBucketACL(CannedAccessControlList.PublicRead
); // Specify the ACL permission of the bucket.
createBucketRequest.setLocationConstraint("oss-cn-hangzhou"); //
Specify the data center that stores the bucket.
OSSAsyncTask createTask = oss.asyncCreateBucket(createBucketRequest,
new OSSCompletedCallback<CreateBucketRequest, CreateBucketResult>() {
    @Override
    public void onSuccess(CreateBucketRequest request, CreateBuck
etResult result) {
        Log.d("locationConstraint", request.getLocationConstraint());
        }
    @Override
    public void onFailure(CreateBucketRequest request, ClientException
 clientException, ServiceException serviceException) {
        // Request exception
        if (clientException ! = null) {
            // Local exception, such as a network exception
            clientException.printStackTrace();
        }
        if (serviceException ! = null) {
            // Service exception
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
```

The preceding code specifies the ACL and data center for the created bucket.

- **Each user can have a maximum 30 buckets.**
- **The name of each bucket is globally unique. If the name of a newly created bucket is the same as an in-service bucket that belongs to another user, the creation fails.**

- **When you create a bucket, you can configure the bucket ACL. If no ACL is configured, by default the bucket permission is private.**
- **The result of successful bucket creation be returned to the data center where the bucket is located.**

Get the bucket ACL permission

The following code gets the ACL of a bucket:

```
GetBucketACLRequest getBucketACLRequest = new GetBucketACLRequest("<
bucketName>");
OSSAsyncTask getBucketAclTask = oss.asyncGetBucketACL(getBucketA
CLRequest, new OSSCompletedCallback<GetBucketACLRequest, GetBucketA
CLResult>() {
    @Override
    public void onSuccess(GetBucketACLRequest request, GetBucketA
CLResult result) {
        Log.d("BucketAcl", result.getBucketACL());
        Log.d("Owner", result.getBucketOwner());
        Log.d("ID", result.getBucketOwnerID());
    }
    @Override
    public void onFailure(GetBucketACLRequest request, ClientException
 clientException, ServiceException serviceException) {
        // Request exception
        if (clientException ! = null) {
            // Local exception, such as a network exception
            clientException.printStackTrace();
        }
        if (serviceException ! = null) {
            // Service exception
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
```

When you use the preceding code to obtain the bucket ACL permission, consider the following points:

- **Currently, three access permissions are available for a bucket: public-read-write , public-read, and private.**
- **Only the bucket owner can use the GetBucketACL interface.**
- **The returned result contains the bucket ACL permission, ID, and name (consistent with the ID) of the bucket owner.**

Delete a bucket

The following code deletes a bucket:

```
DeleteBucketRequest deleteBucketRequest = new DeleteBucketRequest("<
bucketName>");
OSSAsyncTask deleteBucketTask = oss.asyncDeleteBucket(deleteBuck
etRequest, new OSSCompletedCallback<DeleteBucketRequest, DeleteBuck
etResult>() {
    @Override
    public void onSuccess(DeleteBucketRequest request, DeleteBuck
etResult result) {
        Log.d("DeleteBucket", "Success!") ;
    }
    @Override
    public void onFailure(DeleteBucketRequest request, ClientException
 clientException, ServiceException serviceException) {
            // Request exception
        if (clientException ! = null) {
            // Local exception, such as a network exception
            clientException.printStackTrace();
        }
        if (serviceException ! = null) {
            // Service exception
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
});
```

Notice:

- **To prevent accidental deletion, OSS does not allow users to delete a non-empty bucket.**
- **Only the bucket owner has the permission to delete the bucket.**

9.5.2 Hotlink protection

This topic describes how to use hotlink protection.

To configure hotlinking protection, you need to set the following parameters:

- **Referer Whitelist: the referer whitelist that specifies the domains that are allowed to access OSS resources.**
- **Allow Empty Referer: indicates whether the Referer field can be left empty in a request. If the Referer field cannot be left empty, users must include the Referer field in the HTTP or HTTPS request header so as to access OSS resources.**

For more information about hotlink protection, see *Configure hotlink protection*.

Configure hotlink protection

```
PutBucketRefererRequest request = new PutBucketRefererRequest();
request.setBucketName("yourBucketName");
// Add a Referer whitelist. You can use an asterisk (*) or a question
mark (?) as a wildcard character to set the Referer parameter.
ArrayList<String> referers = new ArrayList<String>();
referers.add("http://www.aaa.com");
referers.add("http://www. *.com");
referers.add("http://www.?.com");
request.setReferers(referers);

OSSAsyncTask task = oss.asyncPutBucketReferer(request, new OSSComplet
edCallback<PutBucketRefererRequest, PutBucketRefererResult>() {
    @Override
    public void onSuccess(PutBucketRefererRequest request, PutBucketR
efererResult result) {
        OSSLog.logInfo("code: " + result.getStatusCode());
    }
    @Override
    public void onFailure(PutBucketRefererRequest request, ClientExce
ption clientException, ServiceException serviceException) {
        OSSLog.logError("error: "+serviceException.getRawMessage());
    }
});
task.waitUntilFinished();
```

Obtain hotlink protection information

```
GetBucketRefererRequest request = new GetBucketRefererRequest();
request.setBucketName("yourBucketName");
OSSAsyncTask task = oss.asyncGetBucketReferer(request, new OSSComplet
edCallback<GetBucketRefererRequest, GetBucketRefererResult>() {
    @Override
    public void onSuccess(GetBucketRefererRequest request, GetBucketR
efererResult result) {
        // Obtain the Referer whitelist of the bucket.
        ArrayList<String> list = result.getReferers();
        for (String ref : list){
            OSSLog.logInfo("info: " + ref);
        }
    }
    @Override
    public void onFailure(GetBucketRefererRequest request, ClientExce
ption clientException, ServiceException serviceException) {
        OSSLog.logError("error: "+serviceException.getRawMessage());
    }
});
task.waitUntilFinished();
```

Clear hotlink protection configurations

```
PutBucketRefererRequest request = new PutBucketRefererRequest();
request.setBucketName("yourBucketName");
request.setAllowEmpty(true);
// You cannot clear hotlink protection configurations directly. To
clear hotlink protection configurations, you must create a rule that
allows an empty Referer field to replace the existing rule.
ArrayList<String> referers = new ArrayList<String>();

request.setReferers(referers);
```

```
OSSAsyncTask task = oss.asyncPutBucketReferer(request, new OSSComplet
edCallback<PutBucketRefererRequest, PutBucketRefererResult>() {
    @Override
    public void onSuccess(PutBucketRefererRequest request, PutBucketR
efererResult result) {
        OSSLog.logInfo("code: " + result.getStatusCode());

    }
    @Override
    public void onFailure(PutBucketRefererRequest request, ClientExce
ption clientException, ServiceException serviceException) {
        OSSLog.logError("error: "+serviceException.getRawMessage());

    }
});
task.waitUntilFinished();
```

9.6 Upload objects

9.6.1 Overview

This topic describes the methods provided by OSS Android SDK to upload objects.

In OSS, an object is the basic unit for data operations. OSS Android SDK provides the following object upload methods:

- *Simple upload*: **Supports uploading local files and binary byte[] arrays that are smaller than or equal to 5 GB.**
- *Append upload*: **Supports uploading files that are smaller than or equal to 5 GB.**
- *Resumable upload*: **Supports concurrent resumable upload tasks and custom part size. We recommend you use this method when uploading files that are larger than 5 GB and smaller than or equal to 48.8 TB.**

Note:

For more information about the application scenarios for the preceding upload methods, see Upload objects in the OSS Developer Guide.

When uploading a file, you can view the upload progress through the *progress bar*. **After a file is uploaded, you can perform** *upload callback*.

9.6.2 Simple upload

This topic describes how to upload a file in the simple upload method.

Upload a local file

- **Upload a local file by calling the synchronous interface:**

```
// Construct an upload request.
```

```
PutObjectRequest put = new PutObjectRequest("<bucketName>", "<
objectName>", "<uploadFilePath>");

// The setting of the object metadata is optional.
// ObjectMetadata metadata = new ObjectMetadata();
// metadata.setContentType("application/octet-stream"); // Specify
the content-type.
// metadata.setContentMD5(BinaryUtil.calculateBase64Md5(uploadFile
Path)); // Verify the MD5 value.
// put.setMetadata(metadata);

try {
    PutObjectResult putResult = oss.putObject(put);

    Log.d("PutObject", "UploadSuccess");
    Log.d("ETag", putResult.getETag());
    Log.d("RequestId", putResult.getRequestId());
} catch (ClientException e) {
    // A local exception (such as network exception) occurs.
    e.printStackTrace();
} catch (ServiceException e) {
    // A service error occurs.
    Log.e("RequestId", e.getRequestId());
    Log.e("ErrorCode", e.getErrorCode());
    Log.e("HostId", e.getHostId());
    Log.e("RawMessage", e.getRawMessage());
}
```

Note:

In OSS Android SDK, you can call the synchronous interface only in child threads but not UI treads. Otherwise, an exception occurs. If you want to upload the file in a UI tread, call the asynchronous interface.

- **Upload a local file by calling the asynchronous interface:**

```
// Construct an upload request.
PutObjectRequest put = new PutObjectRequest("<bucketName>", "<
objectName>", "<uploadFilePath>");

// You can set progress callback when uploading a file by calling
the asynchronous interface.
put.setProgressCallback(new OSSProgressCallback<PutObjectRequest>()
 {
    @Override
    public void onProgress(PutObjectRequest request, long currentSiz
e, long totalSize) {
        Log.d("PutObject", "currentSize: " + currentSize + "
totalSize: " + totalSize);
    }
});

OSSAsyncTask task = oss.asyncPutObject(put, new OSSCompletedCallback
<PutObjectRequest, PutObjectResult>() {
    @Override
    public void onSuccess(PutObjectRequest request, PutObjectResult
result) {
        Log.d("PutObject", "UploadSuccess");
        Log.d("ETag", result.getETag());
        Log.d("RequestId", result.getRequestId());
```

```
    }

    @Override
    public void onFailure(PutObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // A request exception occurs.
        if (clientExcepion != null) {
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
// task.cancel(); // You can cancel the task.
// task.waitUntilFinished(); // You can also wait until the task is
complete.
```

Upload a binary byte[] array

Run the following code to upload a binary byte[] array:

```
byte[] uploadData = new byte[100 * 1024];
new Random().nextBytes(uploadData);

// Construct an upload request.
PutObjectRequest put = new PutObjectRequest("<bucketName>", "<
objectName>", uploadData);

try {
    PutObjectResult putResult = oss.putObject(put);

    Log.d("PutObject", "UploadSuccess");
    Log.d("ETag", putResult.getETag());
    Log.d("RequestId", putResult.getRequestId());
} catch (ClientException e) {
    // A local exception (such as network exception) occurs.
    e.printStackTrace();
} catch (ServiceException e) {
    // A service exception occurs.
    Log.e("RequestId", e.getRequestId());
    Log.e("ErrorCode", e.getErrorCode());
    Log.e("HostId", e.getHostId());
    Log.e("RawMessage", e.getRawMessage());
}
```

9.6.3 Append upload

This topic describes how to upload a file in the append upload method, in which the uploaded file is appended to an existing object.

An object created by the AppendObject operation is an appendable object, and an object created by the PutObject operation is a normal object.

Run the following code to upload a file in the append upload method:

```
AppendObjectRequest append = new AppendObjectRequest("<bucketName>",
 "<objectName>", "<uploadFilePath>");

ObjectMetadata metadata = new ObjectMetadata();
metadata.setContentType("application/octet-stream");
append.setMetadata(metadata);

// Specify the position where the uploaded data is appended.
append.setPosition(0);

// Set progress callback.
append.setProgressCallback(new OSSProgressCallback<AppendObjectRequest
>() {
    @Override
    public void onProgress(AppendObjectRequest request, long
currentSize, long totalSize) {
        Log.d("AppendObject", "currentSize: " + currentSize + "
totalSize: " + totalSize);
    }
});
// Perform append upload by calling the asynchronous interface.
OSSAsyncTask task = oss.asyncAppendObject(append, new OSSComplet
edCallback<AppendObjectRequest, AppendObjectResult>() {
    @Override
    public void onSuccess(AppendObjectRequest request, AppendObje
ctResult result) {
        Log.d("AppendObject", "AppendSuccess");
        Log.d("NextPosition", "" + result.getNextPosition());
    }

    @Override
    public void onFailure(AppendObjectRequest request, ClientException
 clientExcepion, ServiceException serviceException) {
        // Handle the exceptions
    }
});
```

Set the `position` parameter properly as follows when uploading a file in the appended upload method.

- **Set the value of `position` to 0 when you create a new appendable object by performing the AppendObject operation.**
- **When you append the uploaded data to an appendable object, set the value of `position` to the current size of the object.**

  You can get the size of an object in the following two methods:

  - **Get the object size from the response to the upload request.**
  - **Get the object size by initiating a HeadObject request.**

9.6.4 Multipart upload

This topic describes how to use multipart upload.

To enable multipart upload, perform the following steps:

1. **Initialize a multipart upload task.**

   You can call oss.initMultipartUpload to obtain the globally unique upload ID generated in OSS.

2. **Upload parts.**

   You can call oss.uploadPart to upload parts.

   Note:

   - **For parts with a same uploadId, parts are sequenced by their part numbers. If you have uploaded a part and use the same part number to upload another part, the later part will replace the former part.**
   - **OSS places the MD5 value of part data in ETag and returns the MD5 value to the user.**
   - **SDK automatically configures Content-MD5. OSS calculates the MD5 value of uploaded data and compares it with the MD5 value calculated by SDK. If the two values vary, the error code of InvalidDigest is returned.**

3. **Complete the multipart upload.**

   After you have uploaded all parts, call CompleteMultipartUpload to combine these parts into a complete object.

The following code is used as a complete example that describes the process of multipart upload:

```
    //Initialize a multipart upload task.
    //InitiateMultipartUploadRequest is used to specify the name of
the object you want to upload and the name of the target bucket.
    InitiateMultipartUploadRequest init = new InitiateMultipartUpl
oadRequest("<bucketName>", "<objectKey>");
    InitiateMultipartUploadResult initResult = oss.initMultipartUpload
(init);
    //The response to initResult contains the upload ID. The upload ID
 is the unique identifier of the multipart upload task.
    String uploadId = initResult.getUploadId();

    //Multipart upload.
    for (int i = 1;i<5;i++){
    byte[] data = new byte[100 * 1024];
    UploadPartRequest uploadPart = new UploadPartRequest();
    uploadPart.setBucketName(BucketName);
    uploadPart.setObjectKey(objectKey);
```

```
    uploadPartRequest.setUploadId(uploadId);
    //The part number corresponding to the parts to be uploaded starts
 from 1.
    uploadPart.setPartNumber(i);
    uploadPart.setPartContent(data);]
    try {
        oss.uploadPart(uploadPart);
    } catch (ServiceException serviceException) {
        OSSLog.logError(serviceException.getErrorCode());
    }
}

    //Complete the multipart upload task.
    CompleteMultipartUploadRequest complete = new CompleteMultipartUpl
oadRequest("<bucketName>", "<objectName>", "<uploadId>", "<partETagLi
st>";

    //Upload callback. You can set the CALLBACK_SERVER parameter to
 complete the multipart upload request. A callback request is sent
to the specified server address after the multipart upload task is
complete. You can view the servercallback result in completeResult.
getServerCallbackReturnBody() of the response.
    complete.setCallbackParam(new HashMap<String, String>() {
        {
            put("callbackUrl", CALLBACK_SERVER); //Set the CALLBACK_S
ERVER parameter to your server address.
            put("callbackBody", "test");
        }
    });
    CompleteMultipartUploadResult completeResult = oss.completeMu
ltipartUpload(complete);
    OSSLog.logError("-------------- serverCallback: " + completeResult
.getServerCallbackReturnBody());
```

The preceding code uses UploadPart to upload each part.

Notice:

- **An upload ID and a part number must be specified for each multipart upload request.**
- **If you use uploadPart, each part must be larger than 100 KB aside from the last part. However, the UploadPart API does not check the uploaded part size immediately. This API checks the part size only when multipart upload is complete.**
- **The part number ranges from 1 to 10,000. If a part number is beyond this range, OSS returns the InvalidArgument error code.**
- **When each part is uploaded, the stream is directed to the starting position of the uploaded part.**
- **After each part is uploaded, OSS returns a response that contains the ETag value of the part. If the ETag value and the MD5 value of the part are the same, you**

must combine the ETag value with the part number into PartETag and save the PartETag for subsequent upload of parts.

List uploaded parts

You can call oss.listParts to obtain all uploaded parts of a multipart upload task.

Use the following code to list uploaded parts:

```
    //List uploaded parts.
    ListPartsRequest listParts = new ListPartsRequest("<bucketName>",
 "<objectName>", "<uploadId>");
    ListPartsResult result = oss.listParts(listParts);

    List<PartETag> partETagList = new ArrayList<PartETag>();
    for (PartSummary part : result.getParts()) {
        partETagList.add(new PartETag(part.getPartNumber(), part.
getETag()));
    }
```

Notice:

If a bucket contains more than 1,000 parts that are uploaded by using multipart upload, OSS returns the first 1,000 parts by default. In the response, the IsTruncated field is set to false and NextPartNumberMarker is returned to indicate the next starting position to continue reading data.

Cancel a multipart upload task

You can call oss.abortMultipartUpload to cancel a multipart upload request mapped to the specified upload ID. After you cancel a multipart upload task, the same upload ID cannot be used to perform operations, and the parts uploaded using that upload ID are deleted.

Use the following code to cancel a multipart upload task:

```
    //Cancel a multipart upload task.
    AbortMultipartUploadRequest abort = new AbortMultipartUpload
Request("<bucketName>", "<objectName>", "<uploadId>");
    AbortMultipartUploadResult abortResult = oss.abortMultipartUpload(
abort);
```

Upload a local file in multipart upload mode

You can use either of the following methods to upload a local file in multipart upload mode through its file path.

- **Method 1**

```
ObjectMetadata meta = new ObjectMetadata();
        //Configure object metadata.
```

```
        meta.setHeader("x-oss-object-acl", "public-read-write");
        MultipartUploadRequest rq = new MultipartUploadRequest("<
bucketName>", "<objkey>",
                "<filepath>", meta);
        //Set PartSize. The default value of PartSize is 256 KB, and
 the minimum value is 100 KB.
        rq.setPartSize(1024 * 1024);
        rq.setProgressCallback(new OSSProgressCallback<MultipartU
ploadRequest>() {
            @Override
            public void onProgress(MultipartUploadRequest request,
long currentSize, long totalSize) {
                OSSLog.logDebug("[testMultipartUpload] - " +
currentSize + " " + totalSize, false);
            }
        });

        CompleteMultipartUploadResult result = oss.multipartUpload(
rq);

        //Check the integrity of the uploaded objects.
        OSSTestUtils.checkFileMd5(oss, "<bucketName>" "<objkey>", "<
filepath>");
```

- **Method 2**

```
MultipartUploadRequest request = new MultipartUploadRequest("<
bucketName>", "<objkey>",
                "<filepath>");

    request.setProgressCallback(new OSSProgressCallback<MultipartU
ploadRequest>() {

        @Override
        public void onProgress(MultipartUploadRequest request, long
currentSize, long totalSize) {
            OSSLog.logDebug("[testMultipartUpload] - " + currentSize
 + " " + totalSize, false);
        }
    });

    OSSAsyncTask task = oss.asyncMultipartUpload(request, new
OSSCompletedCallback<MultipartUploadRequest, CompleteMultipartUpl
oadResult>() {
    @Override
    public void onSuccess(MultipartUploadRequest request, CompleteMu
ltipartUploadResult result) {
        OSSLog.logInfo(result.getServerCallbackReturnBody());
    }

    @Override
    public void onFailure(MultipartUploadRequest request, ClientExce
ption clientException, ServiceException serviceException) {
        OSSLog.logError(serviceException.getRawMessage());

    }
});

   // Thread.sleep(100);
   //Cancel the multipart upload task.
   // task.cancel();

    task.waitUntilFinished();
```

```
    OSSTestUtils.checkFileMd5(oss, "<bucketName>" "<objkey>", "<
filepath>");
```

9.6.5 Resumable upload

This topic describes how to perform resumable uploads.

In a wireless network, it takes a long time to upload a large file. The upload may fail due to poor network connection or network switching. In this case, the entire file needs to be uploaded again. OSS Android SDK provides the resumable upload feature to address this problem.

Notice:

- **Currently, resumable upload is applicable only to local files.**
- **We recommend you do not use resumable upload when uploading small-sized and medium-sized files from a mobile device. Resumable upload adopts the multipart upload method, which initiates multiple network requests for uploading a single file, reducing the upload efficiency.**

Before uploading a file to OSS using the resumable upload method, you can specify the folder used to store resumable upload records.

- **The resuamble upload method is applied only in the current upload task. In scenarios where the folder used to store resumable upload records is not specified, if a part of a large file fails to be uploaded due to network problems, it takes a large amount of time and traffic to upload the entire file again.**
- **If you specify the folder used to store resumable upload records, a failed resumable upload task can be restarted from the position recorded.**

When a resumable upload task fails and is not resumed for a long time, the uploaded parts become useless and occupy the storage space of OSS. In this case, you can set lifecycle rules for the bucket to clear useless parts in a timely manner. For more information, see *Lifecycle Management*.

Note:

- **Resumable upload is performed based on the following APIs: InitMultipartUpload, UploadPart, ListParts, CompleteMultipartUpload, and AbortMultipartUpload. If you use the STS authentication mode to perform resumable upload tasks, make sure that you can access the preceding APIs.**

- **Resumable upload supports the upload callback function, which is used in the same way as it is used in normal upload tasks.**
- **The MD5 value of each part is verified in a resumable upload task by default. Therefore, you do not have to specify the `Content-Md5` header in the request.**

- **The following code uses a calling method in which resumable upload records are not permanently stored on the local device:**

```
// Create a resumable upload request.
ResumableUploadRequest request = new ResumableUploadRequest("<
bucketName>", "<objectKey>", "<uploadFilePath>");
// Configure the progress callback function for the upload task.
request.setProgressCallback(new OSSProgressCallback<ResumableU
ploadRequest>() {
 @Override
 public void onProgress(ResumableUploadRequest request, long
currentSize, long totalSize) {
  Log.d("resumableUpload", "currentSize: " + currentSize + "
totalSize: " + totalSize);
 }
});
// Call the asynchronously resumable upload interface.
OSSAsyncTask resumableTask = oss.asyncResumableUpload(request, new
OSSCompletedCallback<ResumableUploadRequest, ResumableUploadResult
>() {
 @Override
 public void onSuccess(ResumableUploadRequest request, ResumableU
ploadResult result) {
  Log.d("resumableUpload", "success!") ;
 }

 @Override
 public void onFailure(ResumableUploadRequest request, ClientExce
ption clientExcepion, ServiceException serviceException) {
  // Handle exceptions.
 }
});

// resumableTask.waitUntilFinished(); // Wait until the task is
complete.
```

- **The following code uses a calling method in which resumable upload records are permanently stored on the local device:**

```
String recordDirectory = Environment.getExternalStorageDirectory().
getAbsolutePath() + "/oss_record/";

File recordDir = new File(recordDirectory);

// The directory must exist. Otherwise, it will be automatically
created.
if (! recordDir.exists()) {
 recordDir.mkdirs();
}

// Create a resumable upload request with a parameter indicating the
 absolute path of the folder for storing resumable upload records.
```

```
ResumableUploadRequest request = new ResumableUploadRequest("<
bucketName>", "<objectKey>", "<uploadFilePath>", recordDirectory);
// Set the progress callback function for the upload task..
request.setProgressCallback(new OSSProgressCallback<ResumableU
ploadRequest>() {
 @Override
 public void onProgress(ResumableUploadRequest request, long
currentSize, long totalSize) {
  Log.d("resumableUpload", "currentSize: " + currentSize + "
totalSize: " + totalSize);
 }
});


OSSAsyncTask resumableTask = oss.asyncResumableUpload(request, new
OSSCompletedCallback<ResumableUploadRequest, ResumableUploadResult
>() {
 @Override
 public void onSuccess(ResumableUploadRequest request, ResumableU
ploadResult result) {
  Log.d("resumableUpload", "success!") ;
 }

 @Override
 public void onFailure(ResumableUploadRequest request, ClientExce
ption clientExcepion, ServiceException serviceException) {
  //Handle exceptions.
 }
});

// resumableTask.waitUntilFinished();
```

- **The following code implements a resumable upload task:**

```
// Specify whether to delete the settings of resumable upload
records when the OSSAsyncTask cancel() method is called
String recordDirectory = Environment.getExternalStorageDirectory().
getAbsolutePath() + "/oss_record/";

File recordDir = new File(recordDirectory);

// The directory must exist. Otherwise, it will be automatically
created.
if (! recordDir.exists()) {
 recordDir.mkdirs();
}

// Create a resumable upload request with a parameter indicating the
 absolute path of the folder for storing resumable upload records
ResumableUploadRequest request = new ResumableUploadRequest("<
bucketName>", "<objectKey>", "<uploadFilePath>", recordDirectory);
// If it is set to false, the resumable upload records are not
deleted when the upload task is canceled. If you do not set it, the
 default value is true, indicating that the resumable upload records
 are deleted after the task is canceled and the files are uploaded
again during the next upload.
request.setDeleteUploadOnCancelling(false);
// Set the progress callback function for the upload task.
request.setProgressCallback(new OSSProgressCallback<ResumableU
ploadRequest>() {
 @Override
 public void onProgress(ResumableUploadRequest request, long
currentSize, long totalSize) {
```

```
  Log.d("resumableUpload", "currentSize: " + currentSize + "
totalSize: " + totalSize);
 }
});


OSSAsyncTask resumableTask = oss.asyncResumableUpload(request, new
OSSCompletedCallback<ResumableUploadRequest, ResumableUploadResult
>() {
 @Override
 public void onSuccess(ResumableUploadRequest request, ResumableU
ploadResult result) {
  Log.d("resumableUpload", "success!") ;
 }

 @Override
 public void onFailure(ResumableUploadRequest request, ClientExce
ption clientExcepion, ServiceException serviceException) {
  //Handle exceptions.
 }
});

// resumableTask.waitUntilFinished();
```

9.6.6 Progress bar

This topic describes how to use the progress bar to indicate the upload progress.

The following code shows how to call the asyncPutObject method to use the progress bar.

```
// Construct an upload request.
PutObjectRequest put = new PutObjectRequest("<bucketName>", "<
objectName>", "<uploadFilePath>");

// Configure the callback function to display the progress bar.
put.setProgressCallback(new OSSProgressCallback<PutObjectRequest>() {
    @Override
    public void onProgress(PutObjectRequest request, long currentSize
, long totalSize) {
        Log.d("PutObject", "currentSize: " + currentSize + " totalSize
: " + totalSize);
    }
});

// Upload the file asynchronously.
OSSAsyncTask task = oss.asyncPutObject(put, new OSSCompletedCallback<
PutObjectRequest, PutObjectResult>() {
    @Override
    public void onSuccess(PutObjectRequest request, PutObjectResult
result) {
        Log.d("PutObject", "UploadSuccess");
        Log.d("ETag", result.getETag());
        Log.d("RequestId", result.getRequestId());
    }

    @Override
    public void onFailure(PutObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // A request exception occurs.
        if (clientExcepion != null) {
```

```
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
// task.cancel(); // You can cancel the upload task.
// task.waitUntilFinished(); // Wait until the upload task is complete
.
```

9.6.7 Upload callback

This topic describes how to use the upload callback function.

After a file is uploaded to OSS, OSS can return the callback information to the application server. To get the callback information, you can add callback-related parameters in the upload request. Note that it takes more time for the client to wait for the response than in simple upload tasks if the callback function is enabled because OSS takes time to respond the callback request.

The following code enables the upload callback function:

```
PutObjectRequest put = new PutObjectRequest(testBucket, testObject,
uploadFilePath);

put.setCallbackParam(new HashMap<String, String>() {
 {
  put("callbackUrl", "110.75.82.106/callback");
        put("callbackHost", "oss-cn-hangzhou.aliyuncs.com");
        put("callbackBodyType", "application/json");
  put("callbackBody", "{\"mimeType\":${mimeType},\"size\":${size}}");
 }
});

OSSAsyncTask task = oss.asyncPutObject(put, new OSSCompletedCallback<
PutObjectRequest, PutObjectResult>() {
 @Override
 public void onSuccess(PutObjectRequest request, PutObjectResult
result) {
  Log.d("PutObject", "UploadSuccess");

  // This parameter has value only when the servercallback parameter
is set.
  String serverCallbackReturnJson = result.getServerCallbackReturnBody
();

  Log.d("servercallback", serverCallbackReturnJson);
 }

 @Override
 public void onFailure(PutObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
```

```
  // Handle exceptions.
 }
});
```

If you want to configure custom parameters in the callback information, see the following code:

```
put.setCallbackParam(new HashMap<String, String>() {
    {
        put("callbackUrl", "http://182.92.192.125/leibin/notify.php");
        put("callbackHost", "oss-cn-hangzhou.aliyuncs.com");
        put("callbackBodyType", "application/json");
        put("callbackBody", "{\"object\":${object},\"size\":${size},\"
my_var1\":${x:var1},\"my_var2\":${x:var2}}");
    }
});

put.setCallbackVars(new HashMap<String, String>() {
    {
        put("x:var1", "value1");
        put("x:var2", "value2");
    }
});
```

For more information about the callback function, see *Callback*.

9.7 Download objects

9.7.1 Overview

OSS Android SDK provides various methods for you to download objects, including:

- *Streaming download*
- *Range download*
- *Resumable download*

9.7.2 Streaming download

If you have a large object to download or it is time-consuming to download the entire object at a time, you can use the streaming download method. Streaming download enables you to download a part of an object each time until you download the entire object.

You can obtain the input stream of an object if you download the object in the streaming download method. You must have the write permission on the object to perform streaming download.

Run the following code to perform a streaming download task synchronously:

```
// Construct an object download request.
```

```
GetObjectRequest get = new GetObjectRequest("<bucketName>", "<
objectKey>");

// Configure the download progress callback function.
get.setProgressListener(new OSSProgressCallback<GetObjectRequest>() {
            @Override
            public void onProgress(GetObjectRequest request, long
currentSize, long totalSize) {
                OSSLog.logDebug("getobj_progress: " + currentSize+"
total_size: " + totalSize, false);
            }
        });

try {
 // Execute the download request synchronously and return the result.
 GetObjectResult getResult = oss.getObject(get);

 Log.d("Content-Length", "" + getResult.getContentLength());

 // Obtain the input stream of the object.
 InputStream inputStream = getResult.getObjectContent();

 byte[] buffer = new byte[2048];
 int len;

 while ((len = inputStream.read(buffer)) != -1) {
  // Process the downloaded data. For example, display images or write
 data to files.
 }

 // You can view the metadata of the download object.
 ObjectMetadata metadata = getResult.getMetadata();
 Log.d("ContentType", metadata.getContentType());


} catch (ClientException e) {
 // Catch local exceptions, such as network exceptions.
 e.printStackTrace();
} catch (ServiceException e) {
 // Catch service exceptions.
 Log.e("RequestId", e.getRequestId());
 Log.e("ErrorCode", e.getErrorCode());
 Log.e("HostId", e.getHostId());
 Log.e("RawMessage", e.getRawMessage());
} catch (IOException e) {
 e.printStackTrace();
}
```

Run the following code to perform a streaming download tasks asynchronously:

```
GetObjectRequest get = new GetObjectRequest("<bucketName>", "<
objectKey>");
// Configure the download progress callback function.
get.setProgressListener(new OSSProgressCallback<GetObjectRequest>() {
            @Override
            public void onProgress(GetObjectRequest request, long
currentSize, long totalSize) {
                OSSLog.logDebug("getobj_progress: " + currentSize+"
total_size: " + totalSize, false);
            }
        });
OSSAsyncTask task = oss.asyncGetObject(get, new OSSCompletedCallback<
GetObjectRequest, GetObjectResult>() {
```

```
 @Override
 public void onSuccess(GetObjectRequest request, GetObjectResult
result) {
  // The request is successful.
  InputStream inputStream = result.getObjectContent();

  byte[] buffer = new byte[2048];
  int len;

  try {
   while ((len = inputStream.read(buffer)) != -1) {
    // Process the downloaded data.
   }
  } catch (IOException e) {
   e.printStackTrace();
  }
 }

 @Override
 public void onFailure(GetObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
  // Handle the exceptions returned for the request.
  if (clientExcepion != null) {
   // A local exception (such as network exception) occurs.
   clientExcepion.printStackTrace();
  }
  if (serviceException != null) {
   // A service exception occurs.
   Log.e("ErrorCode", serviceException.getErrorCode());
   Log.e("RequestId", serviceException.getRequestId());
   Log.e("HostId", serviceException.getHostId());
   Log.e("RawMessage", serviceException.getRawMessage());
  }
 }
});
```

9.7.3 Range download

If you only need a part of the data in an object, you can use the range download method to download specified content.

Run the following code to perform a range download task:

```
GetObjectRequest get = new GetObjectRequest("<bucketName>", "<
objectKey>");

// Set the download range.
get.setRange(new Range(0, 99)); // Download data between the starting
 byte value of 0 and the maximum byte value 99 inclusive (that is, a
total range of 100 bytes).
// get.setRange(new Range(100, Range.INFINITE)); // Download data
between the starting byte value of 100 and the end of the object.
OSSAsyncTask task = oss.asyncGetObject(get, new OSSCompletedCallback<
GetObjectRequest, GetObjectResult>() {
 @Override
 public void onSuccess(GetObjectRequest request, GetObjectResult
result) {
  // The request is successful.
  InputStream inputStream = result.getObjectContent();

  byte[] buffer = new byte[2048];
```

```
  int len;

  try {
   while ((len = inputStream.read(buffer)) != -1) {
    // Process the downloaded data.
   }
  } catch (IOException e) {
   e.printStackTrace();
  }
 }

 @Override
 public void onFailure(GetObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
  // Handle the exceptions returned for the request.
  if (clientExcepion != null) {
   // A local exception (such as network exception) occurs.
   clientExcepion.printStackTrace();
  }
  if (serviceException != null) {
   // A service exception occurs.
   Log.e("ErrorCode", serviceException.getErrorCode());
   Log.e("RequestId", serviceException.getRequestId());
   Log.e("HostId", serviceException.getHostId());
   Log.e("RawMessage", serviceException.getRawMessage());
  }
 }
});
```

9.7.4 Download an object to a local file

This topic describes how to download an object to a local file.

Run the following code to download a specified object to a local file:

```
public void testDownloadObjectToFile() throws Exception {
        // Upload a file.
        PutObjectRequest put = new PutObjectRequest(mBucketName, "
file1m",
                OSSTestConfig.FILE_DIR + "file1m");
        OSSTestConfig.TestPutCallback putCallback = new OSSTestConfig.
TestPutCallback();
        String srcFileBase64Md5 = BinaryUtil.toBase64String(BinaryUtil
.calculateMd5(OSSTestConfig.FILE_DIR + "file1m"));
        OSSAsyncTask putTask = oss.asyncPutObject(put, putCallback);
        putTask.waitUntilFinished();
        assertEquals(200, putCallback.result.getStatusCode());

        // Download an object to a local file.
        GetObjectRequest get = new GetObjectRequest(mBucketName, "
file1m");
        OSSTestConfig.TestGetCallback getCallback = new OSSTestConfig.
TestGetCallback();
        OSSAsyncTask getTask = oss.asyncGetObject(get, getCallback);
        getTask.waitUntilFinished();
        assertEquals(200, getCallback.result.getStatusCode());
        long length = getCallback.result.getContentLength();
        byte[] buffer = new byte[(int) length];
        int readCount = 0;
        while (readCount < length) {
            readCount += getCallback.result.getObjectContent().read(
buffer, readCount, (int) length - readCount);
```

```
        }
        try {
            FileOutputStream fout = new FileOutputStream(OSSTestConfig
.FILE_DIR + "download_file1m");
            fout.write(buffer);
            fout.close();
        } catch (Exception e) {
            OSSLog.logInfo(e.toString());
        }
        String downloadFileBase64Md5 = BinaryUtil.toBase64String(
BinaryUtil.calculateMd5(OSSTestConfig.FILE_DIR + "download_file1m"));
        assertEquals(srcFileBase64Md5, downloadFileBase64Md5);

    }
```

9.7.5 Conditional download

This topic describes how to download objects by setting conditions.

You can specify one or more conditions when downloading objects to determine that the object can be downloaded if the conditions are met, and an error code is returned and the object cannot be downloaded if the conditions are not met. The following table describes the conditions that you can specify.

Parameter	Description
`If-Modified-Since`	**Indicates that if the specified time is earlier than the time when the object is modified, the object can be downloaded. Otherwise, a 304 Not Modified error is returned.**
`If-Unmodified-Since`	**Indicates that if the specified time is later than or equal to the time when the object is modified, the object can be downloaded. Otherwise, a 412 Precondition Failed error is returned.**
`If-Match`	**Indicates that if the specified ETag matches that of an object, the object can be downloaded. Otherwise, a 412 Precondition Failed error is returned.**
`If-None-Match`	**Indicates that if the specified ETag does not match that of an object, the object can be downloaded. Otherwise, a 304 Not Modified error is returned.**

`If-Modified-Since` **and** `If-Unmodified-Since` **can be configured as download conditions at the same time.** `If-Match` **and** `If-None-Match` **can be configured as download conditions at the same time.**

You can use ossClient.getObjectMeta to obtain the ETag of an object.

Run the following code to download an object with conditions:

```
public void testGetObjectWithHeaders() throws Exception {
        GetObjectRequest request = new GetObjectRequest(bucketName,
objectName);
        request.setRequestHeaders(...);


        OSSAsyncTask task = oss.asyncGetObject(request, getCallback);
        task.waitUntilFinished();

        GetObjectRequest rq = getCallback.request;
        GetObjectResult result = getCallback.result;

    }
```

9.7.6 Resumable download

In a resumable download, if the download process is interrupted, it can be restarted from where it stopped, which saves time and traffic.

For example, when you download a video using an application on your mobile phone, the download process is interrupted by default if the network mode of the mobile phone switches from Wi-Fi networks to the cellular data network. When the network mode switches back to Wi-Fi networks, you can manually restart the download process from where it stopped, that is, perform a resumable download.

Detail analysis

The If-Range HTTP request header field specifies the condition that needs to be met to make the Range header field functional.

- **The Range header field takes effect only when the condition included in the If -Range header field is met. In this case, the OSS server returns the 206 Partial Content code and the content specified by the Range header field.**
- **If the condition included in the If-Range header field is not met, the OSS server returns a 200 OK status code and the complete resources that are requested.**
- **Both the Last-Modified and ETag values can be used for verification. However, the two values cannot be used at the same time.**

The If-Range header field is used in resumable downloads to verify whether the resources downloaded in the last paused download are changed.

When the `If-Range` header field is used, the client only needs to initiate one request. However, when `If-Unmodified-Since` or `If-Match` is used, if the specified condition is not met, a 412 pre-condition check failure status code is returned, and the client must initiate another request to obtain the resources.

Implementation

For more information about how to perform resumable downloads on OSS objects in Android systems, see the following implementation code in iOS. If you want to perform resumable downloads, you can implement the method by yourself or download open source third-party frameworks.

You can run the following code to perform resumable downloads on OSS objects in iOS.

```
//1. Use SDK to get the download URL of the object.
String signedURLString = ossClient.presignConstrainedObjectURL(bucket
, object, expires);

//2. Add a download task.

        mDownloadManager = DownloadManager.getInstance();
        mDownloadManager.add(signedURLString, new DownloadListner() {
            @Override
            public void onFinished() {
                Toast.makeText(MainActivity.this, "Download complete
!", Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onProgress(float progress) {
                pb_progress1.setProgress((int) (progress * 100));
                tv_progress1.setText(String.format("%. 2f", progress
 * 100) + "%");
            }

            @Override
            public void onPause() {
                Toast.makeText(MainActivity.this, "Download paused!",
Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onCancel() {
                tv_progress1.setText("0%");
                pb_progress1.setProgress(0);
                btn_download1.setText("Download");
                Toast.makeText(MainActivity.this, "Download canceled
!", Toast.LENGTH_SHORT).show();
            }
        });
```

```
//3. Start the download.
        mDownloadManager.download(signedURLString);

//4. Pause the download.
        mDownloadManager.cancel(signedURLString);

//5. Continue to download.
        mDownloadManager.download(signedURLString);
```

Note:

Currently, OSS servers do not support the `If-Range` field. Therefore, you cannot use `NSURLSessionDownloadTask` to perform resumable downloads in Android systems. When `NSURLSessionDownloadTask` is used to perform resumable downloads, the `If-Range` header field is sent to the OSS server to verify whether the downloaded object is changed.

9.8 Manage objects

9.8.1 Overview

You can manage objects in a bucket by calling APIs to perform the following operations:

- *Determine whether an object exists*
- *Obtain Object Meta*
- *List objects*
- *Copy objects*
- *Delete objects*

9.8.2 Determine whether a specified object exists

OSS provides a synchronous interface for you to check whether a specified object exists in a bucket.

Run the following code to determine whether a specified object exists:

```
try {
    if (oss.doesObjectExist("<bucketName>", "<objectKey>")) {
        Log.d("doesObjectExist", "object exist.");
    } else {
        Log.d("doesObjectExist", "object does not exist.");
    }
} catch (ClientException e) {
    // Catch local exceptions, such as network exceptions.
    e.printStackTrace();
} catch (ServiceException e) {
    // Catch service exceptions.
```

```
    Log.e("ErrorCode", e.getErrorCode());
    Log.e("RequestId", e.getRequestId());
    Log.e("HostId", e.getHostId());
    Log.e("RawMessage", e.getRawMessage());
}
```

9.8.3 Copy an object

You can copy an object from a bucket (source bucket) to another bucket (target bucket) in the same region.

Run the following code to copy an object.

```
// Construct a copy request.
CopyObjectRequest copyObjectRequest = new CopyObjectRequest("<
srcBucketName>", "<srcObjectKey>",
        "<destBucketName>", "<destObjectKey>");

// (Optional) Configure the metadata of the object to be copied.
// ObjectMetadata objectMetadata = new ObjectMetadata();
// objectMetadata.setContentType("application/octet-stream");
// copyObjectRequest.setNewObjectMetadata(objectMetadata);

// Copy the object by calling the asynchronous interface.
OSSAsyncTask copyTask = oss.asyncCopyObject(copyObjectRequest, new
OSSCompletedCallback<CopyObjectRequest, CopyObjectResult>() {
    @Override
    public void onSuccess(CopyObjectRequest request, CopyObjectResult
result) {
            Log.d("copyObject", "copy success!");
    }

    @Override
    public void onFailure(CopyObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // Handle the exceptions returned for the request.
        if (clientExcepion != null) {
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
```

Note:

- **The source object and the target object must be in the same region.**
- **If the path of the source object is the same as that of the target bucket, you can modify the metadata of the existing target bucket.**

- **You can only copy objects that is smaller or equal to 1 GB. For objects larger than 1 GB, you must copy them by performing other operations, such as MultipartUpload.**

9.8.4 List objects

This topic describes how to list all objects or objects with a specified prefix in a bucket.

List all objects in a bucket

Run the following code to list all objects in a bucket:

```
ListObjectsRequest listObjects = new ListObjectsRequest("<bucketName
>");

// Configure the callback information returned when the request is
successful and fails. Send the list request by calling the asynchrono
us interface.
OSSAsyncTask task = oss.asyncListObjects(listObjects, new OSSComplet
edCallback<ListObjectsRequest, ListObjectsResult>() {
 @Override
 public void onSuccess(ListObjectsRequest request, ListObjectsResult
result) {
  Log.d("AyncListObjects", "Success!");
  for (int i = 0; i < result.getObjectSummaries().size(); i++) {
   Log.d("AyncListObjects", "object: " + result.getObjectSummaries().
get(i).getKey() + " "
     + result.getObjectSummaries().get(i).getETag() + " "
     + result.getObjectSummaries().get(i).getLastModified());
  }
 }

 @Override
 public void onFailure(ListObjectsRequest request, ClientException
clientExcepion, ServiceException serviceException) {
  // Handle the exceptions returned for the request.
  if (clientExcepion != null) {
   // A local exception (such as network exception) occurs.
   clientExcepion.printStackTrace();
  }
  if (serviceException != null) {
   // A service exception occurs.
   Log.e("ErrorCode", serviceException.getErrorCode());
   Log.e("RequestId", serviceException.getRequestId());
   Log.e("HostId", serviceException.getHostId());
   Log.e("RawMessage", serviceException.getRawMessage());
  }
 }
});
// Wait until the task is complete.
```

```
task.waitUntilFinished();
```

List objects with a specified prefix in a bucket

Run the following code to list all object prefixed with "file" in a bucket:

```
 ListObjectsRequest listObjects = new ListObjectsRequest("<bucketName
>");

// Set the prefix.
listObjects.setPrefix("file");

// Configure the callback information returned when the request is
successful and fails. Send the list request by calling the asynchrono
us interface.
OSSAsyncTask task = oss.asyncListObjects(listObjects, new OSSComplet
edCallback<ListObjectsRequest, ListObjectsResult>() {
 @Override
 public void onSuccess(ListObjectsRequest request, ListObjectsResult
result) {
  Log.d("AyncListObjects", "Success!");
  for (int i = 0; i < result.getObjectSummaries().size(); i++) {
   Log.d("AyncListObjects", "object: " + result.getObjectSummaries().
get(i).getKey() + " "
     + result.getObjectSummaries().get(i).getETag() + " "
     + result.getObjectSummaries().get(i).getLastModified());
  }
 }

 @Override
 public void onFailure(ListObjectsRequest request, ClientException
clientExcepion, ServiceException serviceException) {
  // Handle the exceptions returned for the request.
  if (clientExcepion != null) {
   // A local exception (such as network exception) occurs.
   clientExcepion.printStackTrace();
  }
  if (serviceException != null) {
   // A service exception occurs.
   Log.e("ErrorCode", serviceException.getErrorCode());
   Log.e("RequestId", serviceException.getRequestId());
   Log.e("HostId", serviceException.getHostId());
   Log.e("RawMessage", serviceException.getRawMessage());
  }
 }
});
task.waitUntilFinished();
```

The following table describes the parameters in the preceding code.

Parameter	Description
delimiter	**Specifies the character used to group object names. All objects whose name includes the specified prefix and the delimiter character for the first time are classified into a group of elements: CommonPrefixes.**

Parameter	Description
marker	Indicates that objects with names after the marker in alphabetical order are returned.
maxkeys	Specifies the maximum number of the returned objects for the current request. The value of this parameter cannot be larger than 1,000. Default value: 100
prefix	Indicates that the keys of the returned objects must be prefixed with the value of this parameter. **Note:** If you include the prefix in the query request, it is also included in the returned key.

9.8.5 Delete an object

This topic describes how to delete an object.

Warning:

Delete an object with caution because a deleted object cannot be recovered.

Run the following code to delete an object:

```
// Create a delete request
DeleteObjectRequest delete = new DeleteObjectRequest("<bucketName>",
 "<objectKey>");
// Delete an object by calling the asynchronous interface.
OSSAsyncTask deleteTask = oss.asyncDeleteObject(delete, new OSSComplet
edCallback<DeleteObjectRequest, DeleteObjectResult>() {
    @Override
    public void onSuccess(DeleteObjectRequest request, DeleteObje
ctResult result) {
        Log.d("asyncCopyAndDelObject", "success!");
    }

    @Override
    public void onFailure(DeleteObjectRequest request, ClientException
 clientExcepion, ServiceException serviceException) {
        // Handle the exceptions returned for the request.
        if (clientExcepion != null) {
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
```

```
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});
```

Notice:

To call DeleteObject to delete an object, you must have the write permission on the bucket that stores the object.

9.8.6 Set the Content-Type of an object

In Web services, the Content-Type of an object specifies the type of the object and determines the method and encoding used to read the object.

In some cases, you must specify the Content-Type of an object to be uploaded. Otherwise, the object cannot be read in the appropriate method. If you do not specify the Content-Type when using OSS Android SDK to upload an object, the SDK automatically specifies it according to the suffix of the object.

```
// Construct an upload request.
PutObjectRequest put = new PutObjectRequest("<bucketName>", "<
objectKey>", "<uploadFilePath>");

ObjectMetadata metadata = new ObjectMetadata();
// Set the Content-Type of the object to be uploaded.
metadata.setContentType("application/octet-stream");
// Customize the user metadata.
metadata.addUserMetadata("x-oss-meta-name1", "value1");
put.setMetadata(metadata);

OSSAsyncTask task = oss.asyncPutObject(put, new OSSCompletedCallback<
PutObjectRequest, PutObjectResult>() {
    ...
});
```

9.8.7 Obtain the metadata of an object

You can call the HeadObject method to obtain only the metadata but not the content of an object.

Run the following code to obtain the metadata of an object:

```
// Create a request to obtain the metadata of an object by calling the
 synchronous interface.
HeadObjectRequest head = new HeadObjectRequest("<bucketName>", "<
objectKey>");

// Obtain the metadata of an object.
```

```
OSSAsyncTask task = oss.asyncHeadObject(head, new OSSCompletedCallback
<HeadObjectRequest, HeadObjectResult>() {
    @Override
    public void onSuccess(HeadObjectRequest request, HeadObjectResult
result) {

 // Obtain the size of the object.
        Log.d("headObject", "object Size: " + result.getMetadata().
getContentLength());
 // Obtain the type of the object.
        Log.d("headObject", "object Content Type: " + result.
getMetadata().getContentType());
    }

    @Override
    public void onFailure(HeadObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // Handle the exceptions returned for the request.
        if (clientExcepion != null) {
            // A local exception (such as network exception) occurs.
            clientExcepion.printStackTrace();
        }
        if (serviceException != null) {
            // A service exception occurs.
            Log.e("ErrorCode", serviceException.getErrorCode());
            Log.e("RequestId", serviceException.getRequestId());
            Log.e("HostId", serviceException.getHostId());
            Log.e("RawMessage", serviceException.getRawMessage());
        }
    }
});

// task.waitUntilFinished(); // Wait until the task is complete.
```

9.9 Data security

OSS Android SDK provides methods to verify data integrity, which ensure data security when you uploading, downloading, and copying objects.

Due to the complex network environment for mobile devices, errors may occur when data is transmitted between the client and the OSS server. Therefore, OSS Android SDK provides two methods to verify data integrity: CRC end-to-end verification and MD5 verification.

- **CRC verification**

  If you enable CRC verification when reading the download stream, data integrity is automatically verified after the stream is completely read.

  Run the following code to enable CRC verification:

  ```
  GetObjectRequest request = new GetObjectRequest(OSSTestConfig.
  ANDROID_TEST_BUCKET, testFile);
  // Enable CRC verification.
  request.setCRC64(OSSRequest.CRC64Config.YES);
  //....
  ```

```
try{
 GetObjectResult result = oss.getObject(request);
 InputStream in = result.getObjectContent();
 ByteArrayOutputStream output = new ByteArrayOutputStream();
    byte[] buffer = new byte[BUFFER_SIZE];
    int len;
    while ((len = in.read(buffer)) > -1) {
            output.write(buffer, 0, len);
    }
    output.flush();
 in.close();
}catch(ClientException e){
 //...
}catch(InconsistentException e){
    //....
}
```

- **MD5 verification**

  To check whether the object uploaded in the multipart upload method is the same as the local file, you can upload the parts with their Content-MD5 values. The OSS server checks the MD5 values of the uploaded parts and determines the upload task is successful only when the MD5 values are the same as the specified Content-MD5 values, ensuring that the uploaded object is the same as the local file.

  Run the following code to enable MD5 verification:

```
UploadPartRequest uploadPart = new UploadPartRequest("<bucketName>",
 "<objectKey>", uploadId, currentIndex);
// Configure the part content.
uploadPart.setPartContent(partData);
// Configure the MD5 value.
```

```
uploadPart.setMd5Digest(BinaryUtil.calculateBase64Md5(data));
```

9.10 Sign a URL

SDK supports the signing of the URL with a validity period or a public resource URL so that the URL can be transferred to a third party for authorized access.

Sign a private resource URL with a validity period

For a bucket or an object not assigned the public-read permission, the following interface can be called to obtain a signed URL to access the bucket or object:

```
String url = oss.presignConstrainedObjectURL("<bucketName>", "<
objectKey>", 30 * 60);
```

Sign a public resource URL

For a bucket or an object with the public-read permission, the following interface can be called to obtain a public URL to access the bucket or object:

```
String url = oss.presignPublicObjectURL("<bucketName>", "<objectKey
>");
```

9.11 Authorized access

OSS Android SDK provides two authentication modes to ensure data security of mobile devices: STS authentication mode and self-signed mode.

STS authentication mode

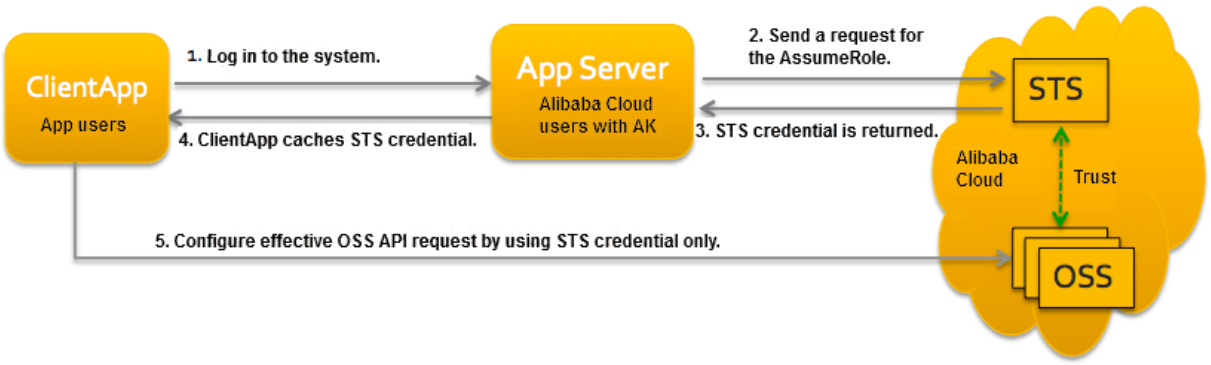
You can temporarily grant OSS access to a user by using the Alibaba Cloud Security Token Service (STS). Alibaba Cloud STS is a web service providing temporary access tokens for cloud computing users. With the STS, you can assign a third-party application or federated user (you can manage the user ID) an access credential with a custom validity period and permissions. For more information about STS, see *Introduction*.

The STS authentication mode has the following advantages:

- **You do not need to disclose your long-term key (AccessKey) to a third-party application. You only need to generate an access token and send the access token to the third-party application. You can customize the access permissions and the validity of this token.**

- You do not need to revoke permissions. The access token becomes invalid automatically when it expires.

For example, the interaction process in an application is shown in the following figure:



The solution is as follows:

1. Log on to the application. You can manage your application ID. You can customize the ID management system or use an external web account or OpenID . For each valid app user, the AppServer can precisely define the minimum access permission.
2. The AppServer requests a security token (SecurityToken) from the STS. Before the STS is called, the AppServer needs to determine the minimum access permission (in the policy syntax) of the application user and the expiration time of the authorization. Then, you can obtain the security token by calling the STS AssumeRole API. For more information about the roles and usage, see *Role Management* in the RAM User Guide.
3. The STS returns a valid access credential (FederationToken in the application) to the AppServer. The access credential contains a SecurityToken, a temporary pair of access keys (an AccessKeyID and an AccessKeySecret), and the expiration time .
4. The AppServer returns the FederationToken to the ClientApp. The ClientApp can cache this credential. When the credential becomes invalid, the ClientApp needs to request a new valid access credential from the AppServer. For example, if the access credential is valid for one hour, the ClientApp can request the AppServer to update the access credential every 30 minutes.

5. The ClientApp uses the locally cached FederationToken to request Alibaba Cloud service APIs. The cloud service detects the STS access credential, verifies the credential with the STS service, and responds to the user requests accordingly.

For more information about the process of accessing OSS resources through STS, see *Access OSS with a temporary access token provided by STS* in the OSS Developer Guide.

For more information about STS security tokens, see *Role Management* in the RAM User Guide. You can call the *AssumeRole* API of STS to obtain the valid access credential. You can also directly use STS SDKs to call this method. For more information, see *#unique_433*.

Note:
To use the STS authentication mode, you must first activate the Alibaba Cloud RAM service.

For more information about STS usage, see *STS user guide*.

For more information about OSS authentication policy configuration, see *Overview*.

- Set the StsToken directly

  You can get a pair of the StsToken in the application by a certain method (such as network requests from your business server) and then use it to initialize the SDK. If this method is adopted, you must follow the expiration time of the STSToken. When the STSToken is about to expire, you must take the initiative to update the new STSToken for the SDK.

  The sample code for initialization is as follows:

```
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

OSSCredentialProvider credentialProvider = new OSSStsTokenCredentia
lProvider("<StsToken.AccessKeyId>", "<StsToken.SecretKeyId>", "<
StsToken.SecurityToken>");
```

```
OSS oss = new OSSClient(getApplicationContext(), endpoint,
credentialProvider);
```

When you notice that the token is about to expire, you can rebuild a new OSSClient or update the CredentialProvider by the following method:

```
oss.updateCredentialProvider(new OSSStsTokenCredentialProvider
("<StsToken.AccessKeyId>", "<StsToken.SecretKeyId>", "<StsToken.
SecurityToken>"));
```

- **Implement the callback to get the StsToken**

  If you want the SDK to automatically update the Token, you must implement the callback in the SDK application, which gets a FederationToken (STSToken) and returns the token. The SDK uses the token for signing. When the token needs to be updated, the SDK calls the callback to get the token.

```
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

OSSCredentialProvider credentialProvider = new OSSFederationCredent
ialProvider() {

    @Override
 public OSSFederationToken getFederationToken() {
  // Here you must implement a method to get a FederationToken and
construct it into the OSSFederationToken object to return it.
     // If the token is not obtained, "nil" is returned.

     OSSFederationToken * token;
     // The following are some codes that can be used to obtain the
token, for example from your server.
     ...
     return token;
 }
};

OSS oss = new OSSClient(getApplicationContext(), endpoint,
credentialProvider);
```

Note:

Additionally, if you have obtained all the fields required by the token by other means, you can use the callback to directly return the token. In this case, you must manually update the token, and then reset the OSSCredentialProvider of the OSSClient instance.

Suppose the URL of the server you set up is http://localhost:8080/distribute-token.json. If you access the URL, the returned data is shown as follows:

```
{
 "StatusCode": 200,
 "AccessKeyId":"STS.iA645eTOXEqP3cg3VeHf",
 "AccessKeySecret":"rV3VQrpFQ4BsyHSAvi5NVLpPIVffDJv4LojUBZCf",
```

```
 "Expiration":"2015-11-03T09:52:59Z",
 "SecurityToken":"CAES7QIIARKAAZPlqaN9ILiQZPS+JDkS/GSZN45RLx4YS/
p3OgaUC+oJl3XSlbJ7StKpQ...."}
```

You can use the following codes to implement the `OSSFederationCredentialProvider` instance:

```
OSSCredentialProvider credetialProvider = new OSSFederationCredent
ialProvider() {
 @Override
 public OSSFederationToken getFederationToken() {
  try {
   URL stsUrl = new URL("http://localhost:8080/distribute-token.json
");
   HttpURLConnection conn = (HttpURLConnection) stsUrl.openConnec
tion();
   InputStream input = conn.getInputStream();
   String jsonText = IOUtils.readStreamAsString(input, OSSConstants.
DEFAULT_CHARSET_NAME);
   JSONObject jsonObjs = new JSONObject(jsonText);
   String ak = jsonObjs.getString("AccessKeyId");
   String sk = jsonObjs.getString("AccessKeySecret");
   String token = jsonObjs.getString("SecurityToken");
   String expiration = jsonObjs.getString("Expiration");
   return new OSSFederationToken(ak, sk, token, expiration);
  } catch (Exception e) {
   e.printStackTrace();
  }
  return null;
 }
};
```

- **Self-signed mode**

  The code is as follows:

  Notice:

  For the STS authentication mode and self-signed mode, make sure the results are returned when you call the implemented callback function. Therefore, if you must obtain the token and signature from the business server with a network request, we recommend that you call the synchronous APIs of the network library. The callback is run in the child thread of the request from the SDKs, which does not block the main thread.

Self-signed mode

You can store the AccessKeyID and AccessKeySecret to your business server and then obtain them by implementing the callback in the SDK. Post the signature string to be signed to the server, sign the string using the OSS-specified signature algorithm on the business server, and transfer the signed string to the callback function, which returns the string to the server.

For more information about the signature algorithm, see *Add a signature to the header*.

The content is a string merged according to various request parameters. The algorithm is:

```
signature = "OSS " + AccessKeyId + ":" + base64(hmac-sha1(AccessKeyS
ecret, content))
```

The code is as follows:

```
String endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

credentialProvider = new OSSCustomSignerCredentialProvider() {
 @Override
 public String signContent(String content) {
  // Here you must sign a string using the OSS-specified signature
algorithm, merge the AccessKeyID with the signed string, and return
the string.
     // Generally, post the string to your business server and then
return the signature.
     // If the signing fails, "nil" is returned after the error is
described.

     // The following is a demo using a local algorithm for signing.
     return "OSS " + AccessKeyId + ":" + base64(hmac-sha1(AccessKeyS
ecret, content));
 }
};

OSS oss = new OSSClient(getApplicationContext(), endpoint, credential
Provider);
```

Notice:

For the STS authentication mode and self-signed mode, make sure the results are returned when you call the implemented callback function. Therefore, if you must obtain the token and signature from the business server with a network request, we recommend that you call the synchronous APIs of the network library. The callback is run in the child thread of the request from the SDKs, which does not block the main thread.

9.12 Image processing

Image Processing (IMG) is a massive, secure, cost-effective and highly reliable image processing service.

After source images are uploaded to OSS, you can process the images on any Internet device at any anytime, from anywhere through simple `RESTful` APIs. You

can establish your own image processing services based on the image processing service provided by OSS.

Features

For more information about the features of OSS image processing service, see *OSS image processing***.**

OSS image processing service **provides the following features:**

- *Resize images*
- *Crop images*
- *Rotate images*
- *Apply effects*
- *Convert format*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**

Usage

- **Anonymous access**

```
String url = oss.presignPublicObjectURL(testBucket, testObject);
  OSSLog.logDebug("signPublicURL", "get url: " + url);
Add the x-oss-process:operation parameter into the generated URL, in
 which operation indicates the operation performed on the image.
```

- **Authorized access**

  When processing images by using OSS Android SDK, call the `request.setxOssProcess()` **method to configure image processing parameters, as shown in the following code:**

```
// Construct the image download request.
GetObjectRequest get = new GetObjectRequest("<bucketName>", "example
.jpg");

// Process the image.
request.setxOssProcess("image/resize,m_fixed,w_100,h_100");

OSSAsyncTask task = oss.asyncGetObject(get, new OSSCompletedCallback
<GetObjectRequest, GetObjectResult>() {
    @Override
    public void onSuccess(GetObjectRequest request, GetObjectResult
result) {
        // The request is successful.
        InputStream inputStream = result.getObjectContent();

        byte[] buffer = new byte[2048];
        int len;
```

```
        try {
            while ((len = inputStream.read(buffer)) != -1) {
                // Process the downloaded data.
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    @Override
    public void onFailure(GetObjectRequest request, ClientException
clientExcepion, ServiceException serviceException) {
        // Handle the exceptions. You can add your own code.
    }
});
```

Note:

To perform other operations on the image, replace the parameters of `request.setxOssProcess()`.

- **Access through SDK**

```
GetObjectRequest request = new GetObjectRequest("bucket-name", "
image-name");
request.setxOssProcess("image/resize,m_lfit,w_100,h_100");  //
Configure image processing parameters.
OSSAsyncTask task = ossClient.asyncGetObject(request, getCallback);
```

Save image processing result

Run the following code to save the image processing results:

```
OSSImagePersistRequest *request = [OSSImagePersistRequest new];
request.fromBucket = _privateBucketName;
request.fromObject = OSS_IMAGE_KEY;
request.toBucket = _privateBucketName;
request.toObject = @"image_persist_key";
// Scale the image.
request.action = @"image/resize,w_100";

[[[ossClient imageActionPersist:request] continueWithBlock:^id
_Nullable(OSSTask * _Nonnull task) {

    return nil;
```

```
}] waitUntilFinished];
```

9.13 Exception handling

OSS Android SDK may be subject to two types of exceptions: ClientException and ServiceException, which are as follows:

ClientException

ClientException indicates an exception inside SDK, such as an incorrect parameter, an unreachable network, or an active cancellation.

ServiceException

OSSException indicates a server error, which is from parsing a server error message. OSSException generally has the following components:

- **Code: The error code OSS returns to users.**
- **Message: The detailed error message provided by OSS.**
- **RequestId: The UUID that uniquely identifies the request. When you cannot solve the problem, you can seek help from OSS development engineers by providing this RequestId.**
- **HostId: Marking the accessed OSS cluster.**
- **rawMessage The original body text of an HTTP response.**

The following are common OSS exceptions:

Error code	Description
AccessDenied	Access denied
BucketAlreadyExists	The bucket already exists.
BucketNotEmpty	The bucket is not empty.
EntityTooLarge	The entity is too large.
EntityTooSmall	The entity is too small.
FileGroupTooLarge	The object group is too large.
FilePartNotExist	An object part does not exist.
FilePartStale	An object part has expired.
InvalidArgument	Parameter format error.
InvalidAccessKeyId	The AccessKeyID does not exist.
InvalidBucketName	The bucket name is invalid.

Error code	Description
InvalidDigest	The digest is invalid.
InvalidObjectName	The object name is invalid.
InvalidPart	A part is invalid.
InvalidPartOrder	The part order is invalid.
InvalidTargetBucketForLogging	The logging operation has an invalid target bucket.
InternalError	Internal OSS error.
MalformedXML	Invalid XML format.
MethodNotAllowed	The method is not supported.
MissingArgument	A parameter is missing.
MissingContentLength	The content length is missing.
NoSuchBucket	The bucket does not exist.
NoSuchKey	The object does not exist.
NoSuchUpload	The Multipart Upload ID does not exist.
NotImplemented	The method cannot be processed.
PreconditionFailed	Pre-processing error.
RequestTimeTooSkewed	The request initiation time exceeds the server time by 15 minutes.
RequestTimeout	Request timed out.
SignatureDoesNotMatch	Signature error.
TooManyBuckets	The user’s bucket quantity exceeds the limit.

10 iOS

10.1 Preface

This topic is based on OSS iOS SDK 8.0.

Overview

This topic assumes that you have activated OSS and created an AccessKeyID and AccessKeySecret. In OSS iOS documentation, ID indicates the AccessKeyID and KEY indicates the AccessKeySecret.

If you have not activated or do not know about OSS, log on to the *OSS Product Homepage* for more help.

Environment requirements

- **iOS V8.0 and later**
- **You must have registered an Alibaba Cloud account and activated OSS.**

SDK source code and API documents

For the source code of the SDK, see *GitHub*. For more information, see *OSS iOS SDK*.

Download the SDK

- *Download from GitHub*
- **Pod dependency: pod 'AliyunOSSiOS'**
- **demo URL:** *https://github.com/aliyun/aliyun-oss-ios-sdk*

10.2 Installation

This topic describes how to install the OSS iOS SDK.

Environment requirements

- **iOS: 8.0 or later**
- **MacOS: 10.10 or later**

Directly import the framework

For more information about how to obtain the OSS iOS SDK framework, see *GitHub*.

In Xcode, drag the framework and drop it to your target area. In the dialog box that appears, select Copy items if needed.

Pod dependency

If your project uses Cocoapods to manage dependencies, add the following dependency to the Podfile. In this case, you do not need to import the OSS iOS SDK framework.

```
pod 'AliyunOSSiOS'
```

For more information about the Cocoapods management tool, see *CocoaPods installation and use tutorial*.

Note:
You can choose either of the preceding methods.

Import the header file to your project

```
#import <AliyunOSSiOS/OSSService.h>
```

Notice:
After you import the framework, add `-ObjC` to `Other Linker Flags` of `Build Settings` in your project. If the `-force_load` option is configured for your project, add `-force_load <framework path>/AliyunOSSiOS`.

Compatible with IPv6-only networks

Domain name resolution in wireless networks is subject to hijackings. To address this issue, the OSS mobile SDK imports HTTPDNS for domain name resolution and directly uses IP addresses to request the OSS server. In an IPv6-only network, compatibility issues may occur. The App Store has officially issued the review requirements for applications, requiring applications to be IPv6-only network compatible. To meet this requirement, the SDK starts to be compatible from V2.5.0. In the later version, aside from -ObjC settings, two system libraries must be imported:

```
libresolv.tbd
CoreTelephony.framework
SystemConfiguration.framework
```

About the ATS policy of Apple

During WWDC 2016, Apple announced that all the applications in Apple App Store must implement App Transport Security (ATS) from January 1, 2017. In other words, all newly submitted applications are not allowed to use `NSAllowsArbitraryLoads` to bypass the ATS limitation. Additionally, the application submitters must ensure that all network requests from the application are encrypted through HTTPS. Otherwise, the application may fail the review.

The OSS iOS SDK V`2.6.0` and later provide the support for this ATS policy. The SDK sends only HTTPS requests. The SDK supports the prefix of `https://` in the `Endpoint` field value. You need only to set the correct HTTPS `Endpoint` field. In this way, all network requests can meet the requirements.

Notice:

- **When you set the `Endpoint` field, you must use the URL that is prefixed with `https://`.**
- **Ensure that the application initiates only HTTPS requests when implementing callbacks such as signing and obtaining STS tokens.**

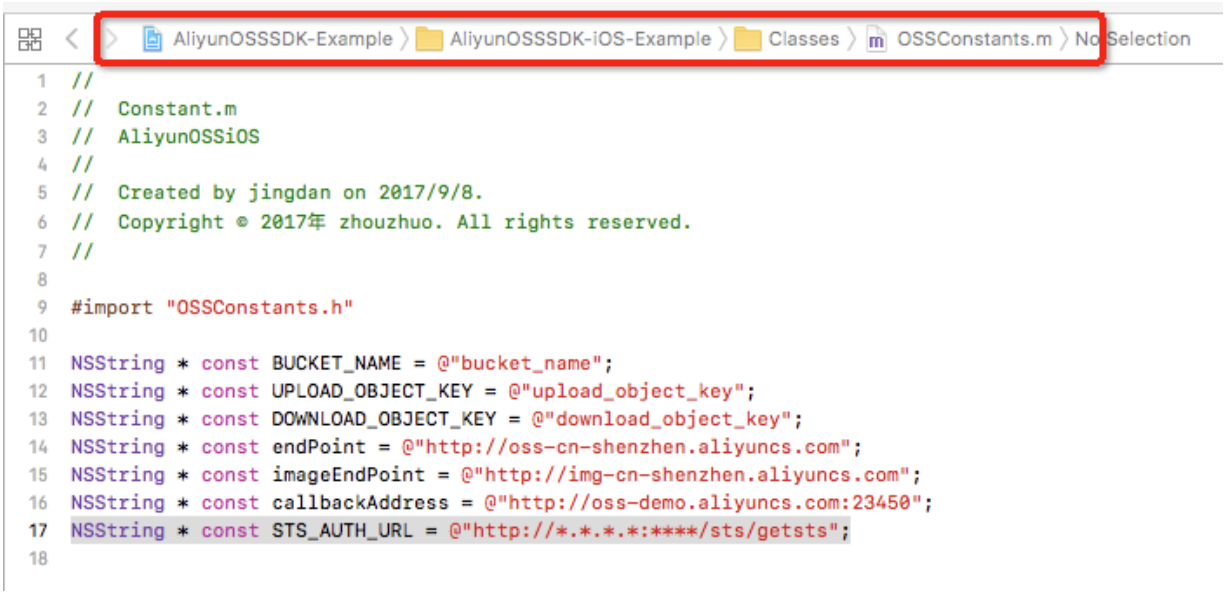
10.3 Quick start

This topic describes how to get started with the iOS SDK.

For more information, see the following examples and cases of this project:

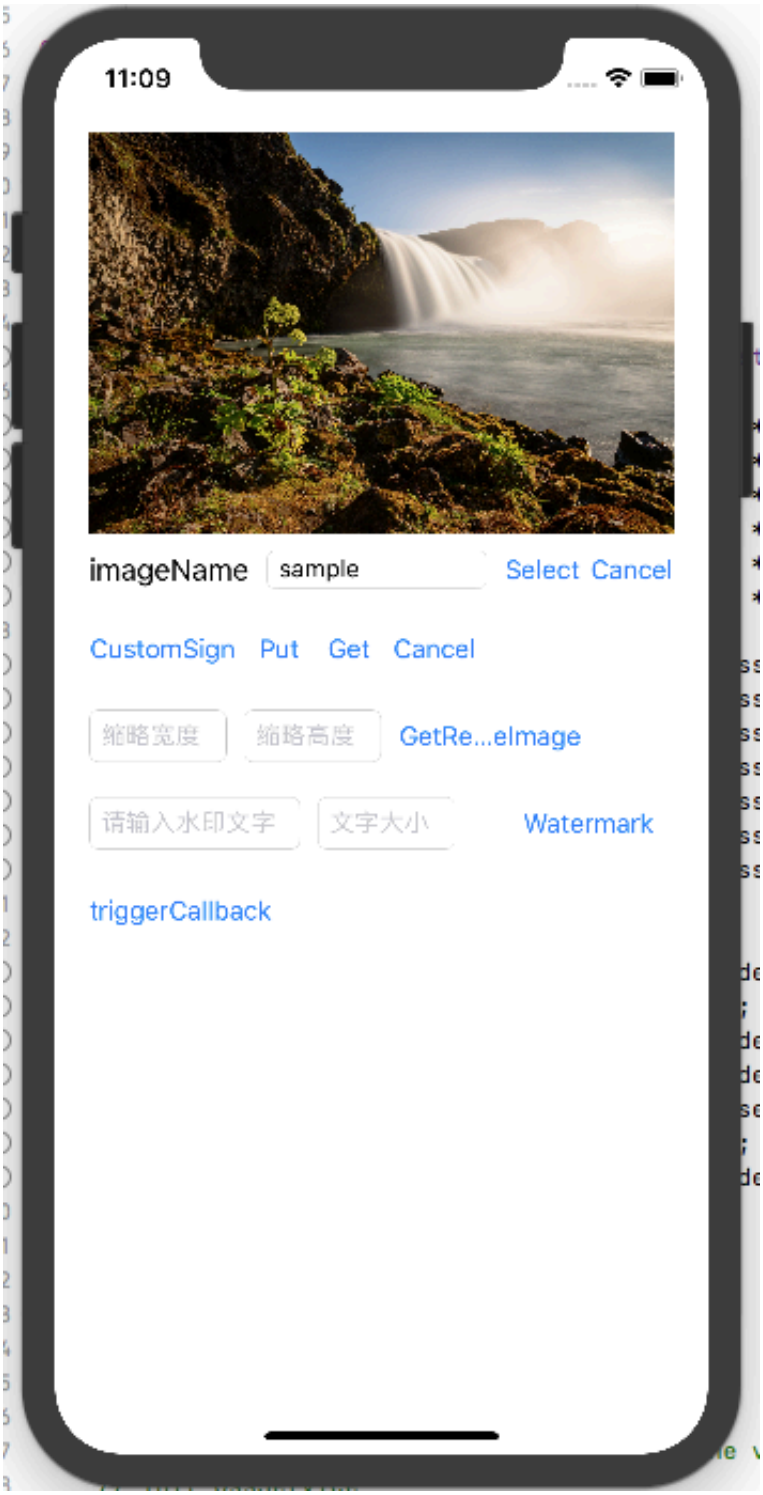
- *iOS demo*
- *Mac demo*
- *Swift demo*
- *Test cases*

You can also directly run the git clone *https://github.com/aliyun/aliyun-oss-ios-sdk* command and configure the following parameters:




```
1  //
2  //  Constant.m
3  //  AliyunOSSiOS
4  //
5  //  Created by jingdan on 2017/9/8.
6  //  Copyright © 2017年 zhouzhuo. All rights reserved.
7  //
8
9  #import "OSSConstants.h"
10
11 NSString * const BUCKET_NAME = @"bucket_name";
12 NSString * const UPLOAD_OBJECT_KEY = @"upload_object_key";
13 NSString * const DOWNLOAD_OBJECT_KEY = @"download_object_key";
14 NSString * const endPoint = @"http://oss-cn-shenzhen.aliyuncs.com";
15 NSString * const imageEndPoint = @"http://img-cn-shenzhen.aliyuncs.com";
16 NSString * const callbackAddress = @"http://oss-demo.aliyuncs.com:23450";
17 NSString * const STS_AUTH_URL = @"http://*.*.*.*:****/sts/getsts";
18
```

Run the project demo as follows:



Examples

The following code shows you how to upload and download files on the iOS and Mac platforms.

- **iOS platform**

  1. **Add the reference.**

     ```
     #import <AliyunOSSiOS/OSSService.h>
     ```

  2. **Initialize the OSSClient instance.**

     To initialize an OSSClient instance, you must configure the Endpoint field, authentication mode, and Client parameters. Three authentication modes are available: plaintext setting, self-signed, and STS authentication. For more information about how to use the STS authentication mode and learn about the basic knowledge of RAM, see *Authorized access*. The following code assumes that you have activated RAM and have basic knowledge of it, such as how to obtain the AccessKey ID, AccessKey Secret, and role ID of a RAM user.

     Set AccessKeyId, SecretKeyId, and RoleArn parameters, as shown in *Script files*. You can use the Python SDK to start a local HTTP service. Use the client code to access the local service and obtain the StsToken.AccessKeyId, StsToken.SecretKeyId, and StsToken.SecurityToken field values.

     For more information, see the "*STS instructions*" section in the sample.

     ```
     NSString *endpoint = "https://oss-cn-hangzhou.aliyuncs.com";

     // We recommend that you use STS to initialize the OSSClient
     instance if you use the mobile device. For more information, see
     the "STS instructions" section in the sample, as shown in https://
     github.com/aliyun/aliyun-oss-ios-sdk/tree/master/DemoByOC.
     id<OSSCredentialProvider> credential = [[OSSStsTokenCredentia
     lProvider alloc] initWithAccessKeyId:@"AccessKeyId" secretKeyId:@"
     AccessKeySecret" securityToken:@"SecurityToken"];

     client = [[OSSClient alloc] initWithEndpoint:endpoint credential
     Provider:credential];
     ```

     The use of the OSSClient instance to initiate upload and download requests is thread-safe. You can run multiple tasks concurrently.

  3. **Upload a file.**

     Assume that you have created a bucket in the console. An `OSSTask` is returned for each SDK-relevant operation. You can configure a continuation for the task to be completed asynchronously or call `waitUntilFinished` to wait for the task to be complete.

     ```
     OSSPutObjectRequest * put = [OSSPutObjectRequest new];
     put.bucketName = @"<bucketName>";
     ```

```
put.objectKey = @"<objectKey>";
put.uploadingData = <NSData *>; // Directly upload NSData.
put.uploadProgress = ^(int64_t bytesSent, int64_t totalByteSent,
int64_t totalBytesExpectedToSend) {
    NSLog(@"%lld, %lld, %lld", bytesSent, totalByteSent,
totalBytesExpectedToSend);
};
OSSTask * putTask = [client putObject:put];
[putTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"upload object success!") ;
    } else {
        NSLog(@"upload object failed, error: %@" , task.error);
    }
    return nil;
}];
// Wait until the task is complete.
// [putTask waitUntilFinished];
```

4. **Download the specified object.**

   Use the following code to download the specified object as `NSData`:

```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
request.bucketName = @"<bucketName>";
request.objectKey = @"<objectKey>";
request.downloadProgress = ^(int64_t bytesWritten, int64_t
totalBytesWritten, int64_t totalBytesExpectedToWrite) {
    NSLog(@"%lld, %lld, %lld", bytesWritten, totalBytesWritten,
totalBytesExpectedToWrite);
};
OSSTask * getTask = [client getObject:request];
[getTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"download object success!") ;
        OSSGetObjectResult * getResult = task.result;
        NSLog(@"download result: %@", getResult.downloadedData);
    } else {
        NSLog(@"download object failed, error: %@" ,task.error);
    }
    return nil;
}];
// Wait until the task is complete.
// [task waitUntilFinished];
```

- **MAC platform**

  Aside from the import method, all operations on the MAC platform are the same as those on the iOS platform.

```
import <AliyunOSSOSX/AliyunOSSiOS.h>
```

10.4 Initialization

The OSSClient is the iOS client of OSS. It provides the caller with a series of methods to operate and manage buckets and objects. Before you use the OSS iOS SDK to

initiate a request to OSS, you need to first initialize an OSSClient instance and complete necessary settings.

Note:

Make sure that the lifecycle of the OSSClient is consistent with that of the application. When an application is started, create an OSSClient instance. When the application is ended, release the instance.

Determine an endpoint

An endpoint is the URL of Alibaba Cloud OSS in a region. It supports the following two formats:

Endpoint type	Description
OSS endpoint	The URL of the region where the bucket resides. For more information about the endpoints of each region, see *Regions and endpoints*.
Custom domain	The domain defined by the user. CNAME allows you to customize the OSS domain .

- OSS endpoints

  When you use the endpoint of a bucket, you can query the endpoint in either of the following ways:

  - Query the relationship between the endpoint and region, as shown in *Regions and endpoints*.
  - You can log on to the *OSS console*. In the left-side navigation pane, locate the Buckets section, enter bucket-1 in the search box and click the search icon. Click the bucket name to go to the Overview tab. In the Domain Names section, the suffix of `bucket-1.oss-cn-hangzhou.aliyuncs.com` is `oss-cn-hangzhou.aliyuncs.com`, which is the endpoint for Internet access.

- **Custom domains**

  Use CNAME to bind a custom domain to a bucket, and then use the domain to access objects in the bucket.

  For example, to bind the new-image.xxxxx.com domain to the bucket named image in China (Shenzhen), follow these steps: Contact your hosting provider for xxxxx.com to configure a new domain name resolution record. This record is used to resolve `http://new-image.xxxxx.com` to `http://image.oss-cn-shenzhen.aliyuncs.com`. The record type is CNAME.

Configure an endpoint and credential

A mobile device is an untrusted environment. It is highly risky to store `AccessKeyId` and `AccessKeySecret` in a mobile device to sign requests. We recommend that you use `STS Authentication Mode` or `Self-signed Mode`. For more information, see *Authorized access* and *Direct transmission through a mobile device*.

> **Note:**
> **If the STS authentication mode is used, we recommend that you use OSSAuthCredentialsProvider to directly access STS. The token is automatically updated after it expires.**

The following example shows you how to configure the Endpoint and Credential Provider fields:

```
NSString *endpoint = "http://oss-cn-hangzhou.aliyuncs.com";

// Use the AccessKey ID and AccessKey Secret that are issued by
Alibaba Cloud to construct a credential provider.
// We recommend that you use the plaintext AccessKey pair only for
 tests. For more information about authentication modes, see the "
Authorized access" section.
id<OSSCredentialProvider> credential = [[OSSPlainTextAKSKPair
CredentialProvider alloc] initWithPlainTextAccessKey:@"<your
accessKeyId>" secretKey:@"<your accessKeySecret>"];

client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential];
```

Set the Endpoint field to CNAME

If you bind CNAME to a bucket, you can set the Endpoint field to CNAME.

```
NSString *endpoint = "http://new-image.xxxxx.com";
```

```
id<OSSCredentialProvider> credential = [[OSSPlainTextAKSKPair
CredentialProvider alloc] initWithPlainTextAccessKey:@"<your
accessKeyId>" secretKey:@"<your accessKeySecret>"];

client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential];
```

Notice:

After Apple implemented the ATS standard, all `Endpoint` URLs must be HTTPS URLs, while domains set by `CNAME` do not support certificate settings. Therefore, you cannot use `CNAME` to set the `Endpoint` field.

Enable access logging

Mobile devices are used in various environments. The OSS SDK cannot provide normal services in some regions or in a certain period of time. To further locate problems developers encounter, the OSS SDK records the log information locally after access logging is enabled. Before you use the OSSClient, initialize the OSSClient instance and call the following method to enable access logging:

```
//The log entry style.
//2017/10/25 11:05:43:863  [Debug]: the 17th: <NSThread: 0x7f809910
8580>{number = 3, name = (null)}
//2017/10/25 11:05:43:863  [Debug]: the 15th:<NSThread: 0x7f80976052c0
>
//2017/10/25 11:05:43:863  [Debug]: ----------TestDebug------------
[OSSLog enableLog];//Run this method to enable access logging.
```

Note:

- **Logs are stored in the Caches/OSSLogs folder of the sandbox.**
- **You can upload logs to the server to further track problems.**

You can also access Alibaba Cloud *Log Service* to upload logs.

Configure ClientConfiguration

Use the following code to configure ClientConfiguration in detail during the initialization process:

```
NSString *endpoint = "https://oss-cn-hangzhou.aliyuncs.com";

// We recommend that you use the plaintext AccessKey pair only for
 tests. For more information about authentication modes, see the "
Authorized access" section.
id<OSSCredentialProvider> credential = [[OSSPlainTextAKSKPair
CredentialProvider alloc] initWithPlainTextAccessKey:@"<your
accessKeyId>" secretKey:@"<your accessKeySecret>"];
```

```
OSSClientConfiguration * conf = [OSSClientConfiguration new];
conf.maxRetryCount = 3; // The number of retry attempts after the
network request encounters an exception and fails.
conf.timeoutIntervalForRequest = 30; // The timeout period of a
network request.
conf.timeoutIntervalForResource = 24 * 60 * 60; // The maximum timeout
 period for resource transmission.

client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential clientConfiguration:conf];
```

OSSTask

An OSSTask is immediately returned for all operations that call APIs. Example:

```
OSSTask * task = [client getObject:get];
```

To implement an asynchronous callback, you can configure a continuation for the task. Example:

```
[task continueWithBlock: ^(OSSTask *task) {
    // do something
    ...
    return nil;
}];
```

You can also wait until the task is complete (synchronous wait). Example:

```
[task waitUntilFinished];
```

10.5 Manage a bucket

A bucket serves as a container that stores objects. Objects belong to the bucket they are created in or uploaded to.

Create a bucket

Use the following code to create a bucket:

```
OSSCreateBucketRequest * create = [OSSCreateBucketRequest new];
create.bucketName = @"<bucketName>";
create.xOssACL = @"public-read";
create.location = @"oss-cn-hangzhou";

OSSTask * createTask = [client createBucket:create];

[createTask continueWithBlock:^id(OSSTask *task) {
 if (! task.error) {
  NSLog(@"create bucket success!") ;
 } else {
  NSLog(@"create bucket failed, error: %@", task.error);
 }
 return nil;
```

```
}];
```

Note:

- **Each user can create a maximum of 30 buckets.**
- **The name of each bucket must be globally unique. In other words, a user cannot create a bucket with the same name as existing buckets. In this case, the bucket fails to be created.**
- **When you create a bucket, you can configure the bucket ACL. If no ACL is configured, the bucket ACL is Private.**
- **The region where the bucket belongs to is returned if the bucket is created.**

List buckets

Use the following code to list all buckets owned by the requester:

```
OSSGetServiceRequest * getService = [OSSGetServiceRequest new];
OSSTask * getServiceTask = [client getService:getService];
[getServiceTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        OSSGetServiceResult * result = task.result;
        NSLog(@"buckets: %@", result.buckets);
        NSLog(@"owner: %@, %@", result.ownerId, result.ownerDispName);
        [result.buckets enumerateObjectsUsingBlock:^(id  _Nonnull obj
, NSUInteger idx, BOOL * _Nonnull stop) {
            NSDictionary * bucketInfo = obj;
            NSLog(@"BucketName: %@", [bucketInfo objectForKey:@"Name
"]);
            NSLog(@"CreationDate: %@", [bucketInfo objectForKey:@"
CreationDate"]);
            NSLog(@"Location: %@", [bucketInfo objectForKey:@"Location
"]);
        }];
    }
    return nil;
}];
```

Note:

Anonymous users do not have the permission to list buckets.

List objects in a bucket

Use the following code to list objects in a bucket:

```
OSSGetBucketRequest * getBucket = [OSSGetBucketRequest new];
getBucket.bucketName = @"<bucketName>";
// getBucket.marker = @"";
// getBucket.prefix = @"";
// getBucket.delimiter = @"";
OSSTask * getBucketTask = [client getBucket:getBucket];
[getBucketTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
```

```
        OSSGetBucketResult * result = task.result;
        NSLog(@"get bucket success!") ;
        for (NSDictionary * objectInfo in result.contents) {
            NSLog(@"list object: %@", objectInfo);
        }
    } else {
        NSLog(@"get bucket failed, error: %@", task.error);
    }
    return nil;
}];
```

Note:

- **To list the objects in the bucket, you must have the operation permission on the bucket.**
- **You can set the prefix, marker, delimiter, and max-keys parameters to list desired results only.**

Delete a bucket

Use the following code to delete a bucket:

```
OSSDeleteBucketRequest * delete = [OSSDeleteBucketRequest new];
delete.bucketName = @"<bucketName>";
OSSTask * deleteTask = [client deleteBucket:delete];
[deleteTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"delete bucket success!") ;
    } else {
        NSLog(@"delete bucket failed, error: %@", task.error);
    }
    return nil;
}];
```

- **Only the bucket owner has the permission to delete the bucket.**
- **To prevent unexpected data loss, OSS does not allow bucket owners to delete non-empty buckets.**

10.6 Upload objects

10.6.1 Overview

This document describes the file upload methods provided by the OSS iOS SDK.

Objects are the basic data units for user operations in OSS. The OSS iOS SDK provides the following object upload methods:

- *Simple upload*: **supports the upload of a file up to 5 GB in size. It includes local file upload and upload from the memory.**

- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files.**
- *Append upload*: **supports the upload of a file up to 5 GB in size.**
- *Resumable upload*: **supports the upload of a file up to 48.8 TB in size. It applies to the upload of large files. It supports concurrent upload and you can define the size of each part.**

> **Note:**
> **For more information about the scenarios of each upload method, see the "Upload files" section in OSS Developer Guide.**

After you complete the file upload, you can perform upload callback, as shown in *Upload callback*.

10.6.2 Simple upload

This topic describes how to use simple upload.

Upload a file from memory or upload a local file

You can upload OSSData directly or use NSURL to upload a file:

```
OSSPutObjectRequest * put = [OSSPutObjectRequest new];
// Configure required fields.
put.bucketName = @"<bucketName>";
put.objectKey = @"<objectKey>";
put.uploadingFileURL = [NSURL fileURLWithPath:@"<filepath>"];
// put.uploadingData = <NSData *>; // Directly upload NSData.
// Configure optional fields.
put.uploadProgress = ^(int64_t bytesSent, int64_t totalByteSent,
int64_t totalBytesExpectedToSend) {
    // The length of the segment being uploaded, the total length of
 the segments uploaded, and the total length of the segments to be
uploaded.
    NSLog(@"%lld, %lld, %lld", bytesSent, totalByteSent, totalBytes
ExpectedToSend);
};
// For more information about the descriptions of optional fields, see
 https://docs.aliyun.com/#/pub/oss/api-reference/object&PutObject.
// put.contentType = @"";
// put.contentMd5 = @"";
// put.contentEncoding = @"";
// put.contentDisposition = @"";
// put.objectMeta = [NSMutableDictionary dictionaryWithObjectsAndKeys
:@"value1", @"x-oss-meta-name1", nil]; // You can configure Object
Meta or other HTTP headers for the upload.
OSSTask * putTask = [client putObject:put];
[putTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"upload object success!") ;
    } else {
        NSLog(@"upload object failed, error: %@" , task.error);
    }
```

```
    return nil;
}];
// [putTask waitUntilFinished];
// [put cancel];
```

Upload a file to a directory

OSS does not use folders. All elements are stored as objects. To simulate a folder in the OSS console, you can create an object whose name ends with a forward slash (/). This object can be uploaded and downloaded. The OSS console displays an object whose name ends with a forward slash (/) as a folder.

When you upload a file, if you write ObjectKey as `"folder/subfolder/file"`, you are simulating the upload of the file to the `file` object in the `folder/subfolder/` directory. Note that the default path is the root directory, and does not need to start with a forward slash (/).

Configure the Content-Type field and enable MD5 verification during an upload

You can explicitly specify the Content-Type field during the upload. If it is not specified, the SDK automatically determines its value based on the file name or the uploaded ObjectKey field value. Additionally, if the Content-MD5 field is set when an object is uploaded, OSS uses the Content-MD5 field value to check whether the received object content is consistent with the sent object content. The SDK provides methods to calculate the Base64 and MD5 values.

```
OSSPutObjectRequest * put = [OSSPutObjectRequest new];
// Configure required fields.
put.bucketName = @"<bucketName>";
put.objectKey = @"<objectKey>";
put.uploadingFileURL = [NSURL fileURLWithPath:@"<filepath>"];
// put.uploadingData = <NSData *>; // Directly upload NSData.
// Optional. Set the Content-Type field.
put.contentType = @"application/octet-stream";
// Optional. Set the MD5 verification.
put.contentMd5 = [OSSUtil base64Md5ForFilePath:@"<filePath>"]; // Set
the MD5 value of the file path.
// put.contentMd5 = [OSSUtil base64Md5ForData:<NSData *>]; // Set the
MD5 value for the binary data.
// Optional. Set the upload progress.
put.uploadProgress = ^(int64_t bytesSent, int64_t totalByteSent,
int64_t totalBytesExpectedToSend) {
    // The length of the segment being uploaded, the total length of
 the segments uploaded, and the total length of the segments to be
uploaded.
    NSLog(@"%lld, %lld, %lld", bytesSent, totalByteSent, totalBytes
ExpectedToSend);
};
OSSTask * putTask = [client putObject:put];
[putTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"upload object success!") ;
```

```
    } else {
        NSLog(@"upload object failed, error: %@" , task.error);
    }
    return nil;
}];
// [putTask waitUntilFinished];
// [put cancel];
```

10.6.3 Multipart upload

This topic describes how to use multipart upload.

Initialize a multipart upload task

Use the following code to initialize a multipart upload task:

```
__block NSString * uploadId = nil;
__block NSMutableArray * partInfos = [NSMutableArray new];
NSString * uploadToBucket = @"<bucketName>";
NSString * uploadObjectkey = @"<objectKey>";
OSSInitMultipartUploadRequest * init = [OSSInitMultipartUploadRequest
new];
init.bucketName = uploadToBucket;
init.objectKey = uploadObjectkey;
// init.contentType = @"application/octet-stream";
OSSTask * initTask = [client multipartUploadInit:init];
[initTask waitUntilFinished];
if (! initTask.error) {
    OSSInitMultipartUploadResult * result = initTask.result;
    uploadId = result.uploadId;
} else {
    NSLog(@"multipart upload failed, error: %@", initTask.error);
    return;
}
```

Note:

- **OSSInitMultipartUploadRequest is used to specify the name of the object you want to upload and the name of the bucket to which the object belongs.**
- **The response to multipartUploadInit contains the upload ID. The upload ID is the unique identifier of the multipart upload task.**

Upload parts

Use the following code to upload parts:

```
NSString * filePath = [docDir stringByAppendingPathComponent:@"***"];
//The object you want to upload.
int chuckCount = *;
//The number of parts.
uint64_t offset = fileSie/chuckCount;
//The size of each part.
for (int i = 1; i <= chuckCount; i++) {
 OSSUploadPartRequest * uploadPart = [OSSUploadPartRequest new];
 uploadPart.bucketName = uploadToBucket;
 uploadPart.objectkey = uploadObjectkey;
```

```
 uploadPart.uploadId = uploadId;
 uploadPart.partNumber = i; // part number start from 1

 NSFileHandle* readHandle = [NSFileHandle fileHandleForReadingAtPath:
filePath];
        [readHandle seekToFileOffset:offset * (i -1)];

        NSData* data = [readHandle readDataOfLength:offset];
        uploadPart.uploadPartData = data;

 OSSTask * uploadPartTask = [client uploadPart:uploadPart];

 [uploadPartTask waitUntilFinished];

 if (! uploadPartTask.error) {
  OSSUploadPartResult * result = uploadPartTask.result;
  uint64_t fileSize = [[[NSFileManager defaultManager] attributes
OfItemAtPath:uploadPart.uploadPartFileURL.absoluteString error:nil]
fileSize];
  [partInfos addObject:[OSSPartInfo partInfoWithPartNum:i eTag:result.
eTag size:fileSize]];
 } else {
  NSLog(@"upload part error: %@", uploadPartTask.error);
  return;
 }
}
```

The preceding code uses uploadPart to upload each part.

Notice:

- **An upload ID and a part number must be specified for each multipart upload request.**
- **If you use uploadPart, each part must be larger than 100 KB aside from the last part. However, the UploadPart API does not check the uploaded part size immediately. This API checks the part size only when multipart upload is complete.**
- **The part number ranges from 1 to 10,000. If a part number is beyond this range, OSS returns the InvalidArgument error code.**
- **When each part is uploaded, the stream is directed to the starting position of the uploaded part.**
- **After each part is uploaded, OSS returns a response that contains the ETag value of the part. If the ETag value and the MD5 value of the part are the same, you must combine the ETag value with the part number into PartETag and save the PartETag for subsequent upload of parts.**

Complete the multipart upload

In the following code, the partInfos field value indicates the list of PartETags saved during the multipart upload. OSS verifies the validity of each part after OSS receives the partInfos field value. When all data parts are verified to be valid, OSS combines them into a complete object.

```
OSSCompleteMultipartUploadRequest * complete = [OSSCompleteMultipart
UploadRequest new];
complete.bucketName = uploadToBucket;
complete.objectKey = uploadObjectkey;
complete.uploadId = uploadId;
complete.partInfos = partInfos;

OSSTask * completeTask = [client completeMultipartUpload:complete];

[[completeTask continueWithBlock:^id(OSSTask *task) {
 if (! task.error) {
  OSSCompleteMultipartUploadResult * result = task.result;
  // ...
 } else {
  // ...
 }
 return nil;
}] waitUntilFinished];
```

You can set the Server Callback parameter to complete the multipart upload request. A callback request is sent to the specified server address after the multipart upload task is complete. You can view the servercallback result in result. serverReturnJsonString of the response.

```
OSSCompleteMultipartUploadRequest * complete = [OSSCompleteMultipart
UploadRequest new];
complete.bucketName = @"<bucketName>";
complete.objectKey = @"<objectKey>";
complete.uploadId = uploadId;
complete.partInfos = partInfos;
complete.callbackParam = @{
                           @"callbackUrl": @"<server address>",
                           @"callbackBody": @"<test>"
                           };
complete.callbackVar = @{
                         @"var1": @"value1",
                         @"var2": @"value2"
                         };
OSSTask * completeTask = [client completeMultipartUpload:complete];
[[completeTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        OSSCompleteMultipartUploadResult * result = task.result;
        NSLog(@"server call back return : %@", result.serverRetu
rnJsonString);
    } else {
        // ...
    }
    return nil;
```

```
}] waitUntilFinished];
```

Note:

To check whether the object uploaded to OSS is consistent with the local file, attach the Content-MD5 field value to the upload request. The OSS server helps you with the MD5 verification. The upload can succeed only when the MD5 value of the object received by the OSS server is the same as the Content-MD5 field value. This method can ensure the consistency of uploaded object.

Use the following code to configure MD5 verification:

```
OSSUploadPartRequest * uploadPart = [OSSUploadPartRequest new];
uploadPart.bucketName = TEST_BUCKET;
uploadPart.uploadId = uploadId;
....
uploadPart.contentMd5 = [OSSUtil fileMD5String:filepath];
```

List parts

Use the following code to call listParts to obtain all uploaded parts of an upload event:

```
OSSListPartsRequest * listParts = [OSSListPartsRequest new];
listParts.bucketName = @"<bucketName>";
listParts.objectKey = @"<objectkey>";
listParts.uploadId = @"<uploadid>";

OSSTask * listPartTask = [client listParts:listParts];

[listPartTask continueWithBlock:^id(OSSTask *task) {
 if (! task.error) {
  NSLog(@"list part result success!") ;
  OSSListPartsResult * listPartResult = task.result;
  for (NSDictionary * partInfo in listPartResult.parts) {
   NSLog(@"each part: %@", partInfo);
  }
 } else {
  NSLog(@"list part result error: %@", task.error);
 }
 return nil;
}];
```

Notice:

If a bucket contains more than 1,000 parts that are uploaded by using multipart upload, OSS returns the first 1,000 parts by default. In the response, the IsTruncated field is set to false and NextPartNumberMarker is returned to indicate the next starting position to continue reading data.

Cancel a multipart upload event

Use the following code to cancel a multipart upload request that is mapped to the specified upload ID:

```
OSSAbortMultipartUploadRequest * abort = [OSSAbortMultipartUpl
oadRequest new];
abort.bucketName = @"<bucketName>";
abort.objectKey = @"<objectKey>";
abort.uploadId = uploadId;

OSSTask * abortTask = [client abortMultipartUpload:abort];

[abortTask waitUntilFinished];

if (! abortTask.error) {
 OSSAbortMultipartUploadResult * result = abortTask.result;
 uploadId = result.uploadId;
} else {
 NSLog(@"multipart upload failed, error: %@", abortTask.error);
 return;
}
```

10.6.4 Append upload

This topic describes how to use append upload.

You can use AppendObject to append objects. The object that is created by using AppendObject is an Appendable object, while the object that is uploaded by using PutObject is a Normal object.

Use the following code for append upload:

```
OSSAppendObjectRequest * append = [OSSAppendObjectRequest new];
// Configure required fields.
append.bucketName = @"<bucketName>";
append.objectKey = @"<objectKey>";
append.appendPosition = 0; // Specify the starting position to append
the object.
NSString * docDir = [self getDocumentDirectory];
append.uploadingFileURL = [NSURL fileURLWithPath:@"<filepath>"];
// Configure optional fields.
append.uploadProgress = ^(int64_t bytesSent, int64_t totalByteSent,
int64_t totalBytesExpectedToSend) {
    NSLog(@"%lld, %lld, %lld", bytesSent, totalByteSent, totalBytes
ExpectedToSend);
};
// For more information about the descriptions of optional fields, see
 https://docs.aliyun.com/#/pub/oss/api-reference/object&AppendObject.
// append.contentType = @"";
// append.contentMd5 = @"";
// append.contentEncoding = @"";
// append.contentDisposition = @"";
OSSTask * appendTask = [client appendObject:append];
[appendTask continueWithBlock:^id(OSSTask *task) {
    NSLog(@"objectKey: %@", append.objectKey);
    if (! task.error) {
        NSLog(@"append object success!") ;
```

```
        OSSAppendObjectResult * result = task.result;
        NSString * etag = result.eTag;
        long nextPosition = result.xOssNextAppendPosition;
    } else {
        NSLog(@"append object failed, error: %@" , task.error);
    }
    return nil;
}];
```

10.6.5 Resumable upload

This topic describes how to use resumable upload.

In a wireless network environment, it takes a long time to upload a large file. The upload may fail due to poor network connectivity or network switches. In this case, the entire file needs to be uploaded again. To address this problem, the SDK provides resumable upload.

Notice:

- **Currently, this feature allows you to upload only local files.**
- **Resumable upload is inapplicable to small- and medium-sized files from a mobile device. Resumable upload adopts the multipart upload method. This method is inefficient because it initiates multiple network requests for uploading a single file.**

Before you upload a file, you can specify the folder for the file that stores resumable upload records. Resumable upload takes effect only for the current upload task. If you do not specify the folder for the file that stores resumable upload records, a part of a large file may fail to be uploaded due to network errors. To upload the entire file, large amounts of time and traffic are required to make retry attempts. If you specify the folder for the file that stores resumable upload records and the upload fails, the reupload starts from the recorded position. For more information, see the subsequent example.

When a resumable upload task fails and is not resumed for a long time, the uploaded parts may become unnecessary in OSS. In this case, you can set the lifecycle rule for the bucket to clear parts periodically. For more information, see *Lifecycle management*.

To manage parts, if the current task is cancelled during resumable upload, the parts already uploaded to the server are cleared synchronously by default. If you want to save the resumable upload record, you must specify a folder of a file that

stores the resumable upload record and modify the deleteUploadIdOnCancelling parameter. If the resumable upload record is saved on the local device for a long time but the parts uploaded to the server are deleted based on the lifecycle rule configured for the bucket, the records on the server and the mobile device may be inconsistent.

Note:

- **Resumable upload depends on `InitMultipartUpload/UploadPart/ListParts/CompleteMultipartUpload/AbortMultipartUpload`. If you use the STS authentication mode, add the permissions required to use the APIs.**
- **Resumable upload supports post-upload callback notifications that can be used in the same way as the preceding typical upload callback notifications.**
- **MD5 verification is automatically enabled for each multipart upload task during resumable upload. You do not need to set the `Content-Md5` field in the request header.**

- **The invocation method (not specified by default) to store resumable upload records permanently in the local device:**

```
OSSResumableUploadRequest * resumableUpload = [OSSResumableUploadRe
quest new];
resumableUpload.bucketName = OSS_BUCKET_PRIVATE;
//...
NSString *cachesDir = [NSSearchPathForDirectoriesInDomains(
NSCachesDirectory, NSUserDomainMask, YES) firstObject];
resumableUpload.recordDirectoryPath = cachesDir;
```

- **Implementation of resumable upload**

```
// Obtain an upload ID to upload the file. If the task fails and can
 be resumed, you can use the same upload ID to upload the same file
to the same bucket.
OSSResumableUploadRequest * resumableUpload = [OSSResumableUploadRe
quest new];
resumableUpload.bucketName = <bucketName>;
resumableUpload.objectKey = <objectKey>;
resumableUpload.partSize = 1024 * 1024;
resumableUpload.uploadProgress = ^(int64_t bytesSent, int64_t
totalByteSent, int64_t totalBytesExpectedToSend) {
    NSLog(@"%lld, %lld, %lld", bytesSent, totalByteSent, totalBytes
ExpectedToSend);
};
NSString *cachesDir = [NSSearchPathForDirectoriesInDomains(
NSCachesDirectory, NSUserDomainMask, YES) firstObject];
//Configure the checkpoint file.
resumableUpload.recordDirectoryPath = cachesDir;
//Set the resumableUpload.deleteUploadIdOnCancelling field to No.
When a task is cancelled, the checkpoint file is not deleted. If
you retain default value Yes, the checkpoint file is deleted when a
```

```
 task is cancelled, and the entire file needs to be uploaded again
when the task restarts.
resumableUpload.deleteUploadIdOnCancelling = NO;

resumableUpload.uploadingFileURL = [NSURL fileURLWithPath:<your file
 path>];
OSSTask * resumeTask = [client resumableUpload:resumableUpload];
[resumeTask continueWithBlock:^id(OSSTask *task) {
    if (task.error) {
        NSLog(@"error: %@", task.error);
        if ([task.error.domain isEqualToString:OSSClientErrorDomain]
 && task.error.code == OSSClientErrorCodeCannotResumeUpload) {
            // The task cannot be resumed. You must obtain a new
upload ID to upload the file.
        }
    } else {
        NSLog(@"Upload file success");
    }
    return nil;
}];

// [resumeTask waitUntilFinished];

// [resumableUpload cancel];
```

10.6.6 Upload callback

This topic describes how to use upload callback.

When an upload is complete, the OSS instance can send a callback to the applicatio n server. To implement the callback, you need only to add the relevant callback parameters to the request that is sent to OSS. Compared with simple upload, the upload callback notification mechanism requires the client to wait for a longer time to process a callback request and response.

For more information, see *Callback* **.**

Use the following code for upload callback:

```
OSSPutObjectRequest * request = [OSSPutObjectRequest new];
request.bucketName = @"<bucketName>";
request.objectKey = @"<objectKey>";
request.uploadingFileURL = [NSURL fileURLWithPath:@<filepath>"];
// Set callback parameters.
request.callbackParam = @{
                          @"callbackUrl": @"<your server callback
address>",
                          @"callbackBody": @"<your callback body>"
                          };
// Set custom variables.
request.callbackVar = @{
                        @"<var1>": @"<value1>",
                        @"<var2>": @"<value2>"
                        };
request.uploadProgress = ^(int64_t bytesSent, int64_t totalByteSent,
int64_t totalBytesExpectedToSend) {
    NSLog(@"%lld, %lld, %lld", bytesSent, totalByteSent, totalBytes
ExpectedToSend);
```

```
};
OSSTask * task = [client putObject:request];
[task continueWithBlock:^id(OSSTask *task) {
    if (task.error) {
        OSSLogError(@"%@", task.error);
    } else {
        OSSPutObjectResult * result = task.result;
        NSLog(@"Result - requestId: %@, headerFields: %@, servercall
back: %@",
              result.requestId,
              result.httpResponseHeaderFields,
              result.serverReturnJsonString);
    }
    return nil;
}];
```

10.7 Download objects

10.7.1 Overview

This document describes how to use the OSS iOS SDK to download objects.

The OSS iOS SDK provides the following download methods:

- *Simple download*
- *Streaming download*
- *Range download*
- *Resumable download*

10.7.2 Simple download

This topic describes how to use simple download.

You can download an object to a local file or as NSData:

```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];

// Configure required fields.
request.bucketName = @"<bucketName>";
request.objectKey = @"<objectKey>";

// Configure optional fields.
request.downloadProgress = ^(int64_t bytesWritten, int64_t totalBytes
Written, int64_t totalBytesExpectedToWrite) {
    // The length of the segment being downloaded, the total length of
 the segments downloaded, and the total length of the segments to be
downloaded.
    NSLog(@"%lld, %lld, %lld", bytesWritten, totalBytesWritten,
totalBytesExpectedToWrite);
};
// request.range = [[OSSRange alloc] initWithStart:0 withEnd:99]; //
Download the foremost 99 Bytes.
// request.downloadToFileURL = [NSURL fileURLWithPath:@"<filepath
>"]; // Specify the destination path if you need to directly download
the object to a file.
```

```
OSSTask * getTask = [client getObject:request];

[getTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"download object success!") ;
        OSSGetObjectResult * getResult = task.result;
        NSLog(@"download result: %@", getResult.dowloadedData);
    } else {
        NSLog(@"download object failed, error: %@" ,task.error);
    }
    return nil;
}];

// [getTask waitUntilFinished];

// [request cancel];
```

10.7.3 Streaming download

If you have a large object to download or it is time-consuming to download the entire object at a time, you can use streaming download. Streaming download enables you to download part of the object content each time until you download the entire object.

The OSS iOS SDK provides no streaming download APIs. Instead, it provides the multipart callback function similar to `didRecieveData` in the `NSURLSession` library.

Note: The download result does not contain the actual data if the multipart callback is configured.

Use the following code for streaming download:

```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
// Configure required fields.
request.bucketName = @"<bucketName>";
request.objectKey = @"<objectKey>";
// Configure the multipart callback function.
request.onRecieveData = ^(NSData * data) {
    NSLog(@"Recieve data, length: %ld", [data length]);
};
OSSTask * getTask = [client getObject:request];
[getTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"download object success!") ;
    } else {
        NSLog(@"download object failed, error: %@" ,task.error);
    }
    return nil;
}];
// [getTask waitUntilFinished];
```

```
// [request cancel];
```

10.7.4 Range download

If you only need part of the data in an object, you can use range download to download specified content.

Range download is suitable for downloading large objects. If the Range parameter is specified in the request header, the response contains the length of the entire object and the range of data.

Run the following code for range download:

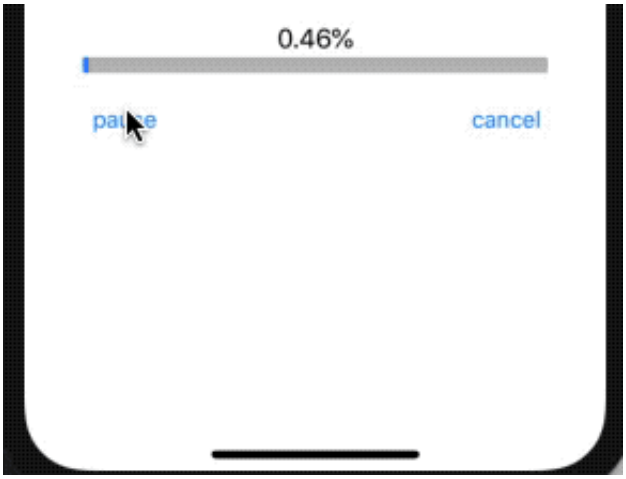
```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
request.bucketName = @"<bucketName>;
request.objectKey = @"<objectKey>";
request.range = [[OSSRange alloc] initWithStart:1 withEnd:99]; //
bytes=1-99
// request.range = [[OSSRange alloc] initWithStart:-1 withEnd:99]; //
bytes=-99
// request.range = [[OSSRange alloc] initWithStart:10 withEnd:-1]; //
bytes=10-
request.downloadProgress = ^(int64_t bytesWritten, int64_t totalBytes
Written, int64_t totalBytesExpectedToWrite) {
    NSLog(@"%lld, %lld, %lld", bytesWritten, totalBytesWritten,
totalBytesExpectedToWrite);
};
OSSTask * getTask = [client getObject:request];
[getTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"download object success!") ;
        OSSGetObjectResult * getResult = task.result;
        NSLog(@"download result: %@", getResult.dowloadedData);
    } else {
        NSLog(@"download object failed, error: %@" ,task.error);
    }
    return nil;
}];
// [getTask waitUntilFinished];
// [request cancel];
```

10.7.5 Resumable download

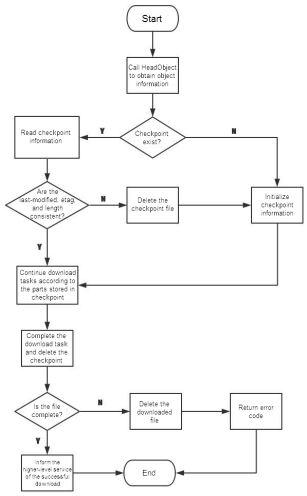
When a client downloads resources over a network, the download is interrupted for some reason. In this case, you can use resumable download to continue downloading the uncompleted parts. This method saves time and traffic.

Scenario: When you use a video application to download a video on a mobile phone, and the network switches from Wi-Fi to Cellular during the download, the application automatically interrupts the download. When you switch back to Wi-Fi , you can manually restart the download task to resume the download.

The following figure shows the effect of resumable download.



The following figure shows the process of resumable download.



Detail analysis

- **The Range field is added to the HTTP 1.1 header to specify the scope within which to obtain data. The formats of the Range field value are as follows:**
  - `Range: bytes=100-`**: starts from the 101st Byte until the download is complete.**
  - `Range: bytes=100-200`**: starts the download from the 101st Byte and stops the download at the 201st Byte. The Range field value is counted from 0 by default. This method is typically used for multipart transmission of large objects, such as videos.**
  - `Range: bytes =-100`**: The start position is not specified. The server downloads the last 100 Bytes, instead of the 100 Bytes starting from 0.**
  - `Range: bytes=0-100, 200-300`**: downloads content within multiple ranges at the same time. This is untypical.**

- **During resumable download, the server requires to use the `If-Match` value in the header to verify whether the object on the server changes. The `If-Match` field value corresponds to the `ETag` value.**
- **When the client initiates a request, the `Range` and `If-Match` field values are included in the request. The OSS server verifies the ETag value that corresponds to the If-Match field value. If the received ETag value does not match the ETag value saved in OSS, OSS returns the 412 precondition status code.**
- **The OSS server currently supports the following fields for GetObject: `Range`, `If-Match`, `If-None-Match`, `If-Modified-Since`, `If-Unmodified-Since`. Therefore, you can use resumable download to download resources from OSS on your mobile device.**

Specific implementation

Example:

Note:

The following code is for reference only. We recommend that you do not use it in production projects. To download the code, you can download it by yourself or use third-party open-source download software.

```
#import "DownloadService.h"
#import "OSSTestMacros.h"

@implementation DownloadRequest

@end

@implementation Checkpoint

- (instancetype)copyWithZone:(NSZone *)zone {
    Checkpoint *other = [[[self class] allocWithZone:zone] init];

    other.etag = self.etag;
    other.totalExpectedLength = self.totalExpectedLength;

    return other;
}

@end


@interface DownloadService()<NSURLSessionTaskDelegate, NSURLSessi
onDataDelegate>

@property (nonatomic, strong) NSURLSession *session;         //The
network session.
@property (nonatomic, strong) NSURLSessionDataTask *dataTask;   //The
data request task.
```

```
@property (nonatomic, copy) DownloadFailureBlock failure;    //The
request error.
@property (nonatomic, copy) DownloadSuccessBlock success;    //The
request succeeds.
@property (nonatomic, copy) DownloadProgressBlock progress;  //The
download progress.
@property (nonatomic, copy) Checkpoint *checkpoint;        //The
checkpoint file.
@property (nonatomic, copy) NSString *requestURLString;    //The
object resource URL used in a download request.
@property (nonatomic, copy) NSString *headURLString;       //The
object resource URL used in a HEAD request.
@property (nonatomic, copy) NSString *targetPath;     //The path to
the object.
@property (nonatomic, assign) unsigned long long totalReceivedContent
Length; //The size of the downloaded content.
@property (nonatomic, strong) dispatch_semaphore_t semaphore;

@end

@implementation DownloadService

- (instancetype)init
{
    self = [super init];
    if (self) {
        NSURLSessionConfiguration *conf = [NSURLSessionConfiguration
defaultSessionConfiguration];
        conf.timeoutIntervalForRequest = 15;

        NSOperationQueue *processQueue = [NSOperationQueue new];
        _session = [NSURLSession sessionWithConfiguration:conf
delegate:self delegateQueue:processQueue];
        _semaphore = dispatch_semaphore_create(0);
        _checkpoint = [[Checkpoint alloc] init];
    }
    return self;
}

+ (instancetype)downloadServiceWithRequest:(DownloadRequest *)request
 {
    DownloadService *service = [[DownloadService alloc] init];
    if (service) {
        service.failure = request.failure;
        service.success = request.success;
        service.requestURLString = request.sourceURLString;
        service.headURLString = request.headURLString;
        service.targetPath = request.downloadFilePath;
        service.progress = request.downloadProgress;
        if (request.checkpoint) {
            service.checkpoint = request.checkpoint;
        }
    }
    return service;
}

/**
 * Use HeadObject to obtain Object Meta. Compare the ETag value saved
 in OSS with that saved in the local checkpoint file. The result is
returned.
 */
- (BOOL)getFileInfo {
    __block BOOL resumable = NO;
    NSURL *url = [NSURL URLWithString:self.headURLString];
```

```
    NSMutableURLRequest *request = [[NSMutableURLRequest alloc]
initWithURL:url];
    [request setHTTPMethod:@"HEAD"];

    NSURLSessionDataTask *task = [[NSURLSession sharedSession]
dataTaskWithRequest:request completionHandler:^(NSData * _Nullable
data, NSURLResponse * _Nullable response, NSError * _Nullable error) {
        if (error) {
            cNSLog(@"Failed to obtain Object Meta. error: %@", error);
        } else {
            NSHTTPURLResponse *httpResponse = (NSHTTPURLResponse *)
response;
            NSString *etag = [httpResponse.allHeaderFields objectForK
ey:@"Etag"];
            if ([self.checkpoint.etag isEqualToString:etag]) {
                resumable = YES;
            } else {
                resumable = NO;
            }
        }
        dispatch_semaphore_signal(self.semaphore);
    }];
    [task resume];

    dispatch_semaphore_wait(self.semaphore, DISPATCH_TIME_FOREVER);
    return resumable;
}

/**
 * Obtain the size of the local file.
 */
- (unsigned long long)fileSizeAtPath:(NSString *)filePath {
    unsigned long long fileSize = 0;
    NSFileManager *dfm = [NSFileManager defaultManager];
    if ([dfm fileExistsAtPath:filePath]) {
        NSError *error = nil;
        NSDictionary *attributes = [dfm attributesOfItemAtPath:
filePath error:&error];
        if (! error && attributes) {
            fileSize = attributes.fileSize;
        } else if (error) {
            NSLog(@"error: %@", error);
        }
    }

    return fileSize;
}

- (void)resume {
    NSURL *url = [NSURL URLWithString:self.requestURLString];
    NSMutableURLRequest *request = [[NSMutableURLRequest alloc]
initWithURL:url];
    [request setHTTPMethod:@"GET"];

    BOOL resumable = [self getFileInfo];    // If the resumable field
value is NO, resumable upload is unfeasible. Otherwise, continue the
resumable upload logic.
    if (resumable) {
        self.totalReceivedContentLength = [self fileSizeAtPath:self.
targetPath];
        NSString *requestRange = [NSString stringWithFormat:@"bytes=%
llu-", self.totalReceivedContentLength];
        [request setValue:requestRange forHTTPHeaderField:@"Range"];
    } else {
```

```
        self.totalReceivedContentLength = 0;
    }

    if (self.totalReceivedContentLength == 0) {
        [[NSFileManager defaultManager] createFileAtPath:self.
targetPath contents:nil attributes:nil];
    }

    self.dataTask = [self.session dataTaskWithRequest:request];
    [self.dataTask resume];
}

- (void)pause {
    [self.dataTask cancel];
    self.dataTask = nil;
}

- (void)cancel {
    [self.dataTask cancel];
    self.dataTask = nil;
    [self removeFileAtPath: self.targetPath];
}

- (void)removeFileAtPath:(NSString *)filePath {
    NSError *error = nil;
    [[NSFileManager defaultManager] removeItemAtPath:self.targetPath
error:&error];
    if (error) {
        NSLog(@"remove file with error : %@", error);
    }
}

#pragma mark - NSURLSessionDataDelegate

- (void)URLSession:(NSURLSession *)session task:(NSURLSessionTask *)
task
didCompleteWithError:(nullable NSError *)error {
    NSHTTPURLResponse *httpResponse = (NSHTTPURLResponse *)task.
response;
    if ([httpResponse isKindOfClass:[NSHTTPURLResponse class]]) {
        if (httpResponse.statusCode == 200) {
            self.checkpoint.etag = [[httpResponse allHeaderFields]
objectForKey:@"Etag"];
            self.checkpoint.totalExpectedLength = httpResponse.
expectedContentLength;
        } else if (httpResponse.statusCode == 206) {
            self.checkpoint.etag = [[httpResponse allHeaderFields]
objectForKey:@"Etag"];
            self.checkpoint.totalExpectedLength = self.totalRecei
vedContentLength + httpResponse.expectedContentLength;
        }
    }

    if (error) {
        if (self.failure) {
            NSMutableDictionary *userInfo = [NSMutableDictionary
dictionaryWithDictionary:error.userInfo];
            [userInfo oss_setObject:self.checkpoint forKey:@"
checkpoint"];

            NSError *tError = [NSError errorWithDomain:error.domain
code:error.code userInfo:userInfo];
            self.failure(tError);
        }
```

```
    } else if (self.success) {
        self.success(@{@"status": @"success"});
    }
}

- (void)URLSession:(NSURLSession *)session dataTask:(NSURLSessi
onDataTask *)dataTask didReceiveResponse:(NSURLResponse *)response
 completionHandler:(void (^)(NSURLSessionResponseDisposition))
completionHandler
{
    NSHTTPURLResponse *httpResponse = (NSHTTPURLResponse *)dataTask.
response;
    if ([httpResponse isKindOfClass:[NSHTTPURLResponse class]]) {
        if (httpResponse.statusCode == 200) {
            self.checkpoint.totalExpectedLength = httpResponse.
expectedContentLength;
        } else if (httpResponse.statusCode == 206) {
            self.checkpoint.totalExpectedLength = self.totalRecei
vedContentLength +  httpResponse.expectedContentLength;
        }
    }

    completionHandler(NSURLSessionResponseAllow);
}

- (void)URLSession:(NSURLSession *)session dataTask:(NSURLSessi
onDataTask *)dataTask didReceiveData:(NSData *)data {

    NSFileHandle *fileHandle = [NSFileHandle fileHandleForWriting
AtPath:self.targetPath];
    [fileHandle seekToEndOfFile];
    [fileHandle writeData:data];
    [fileHandle closeFile];

    self.totalReceivedContentLength += data.length;
    if (self.progress) {
        self.progress(data.length, self.totalReceivedContentLength,
self.checkpoint.totalExpectedLength);
    }
}

@end
```

The preceding sample code shows the process logic when data is received over the network. The fields in the code are described as follows:

- **DownloadService: the core of the download logic.**
- **URLSession:dataTask:didReceiveData: The received network data is appended to the object. The download progress record is updated.**
- **URLSession:task:didCompleteWithError: determines whether the download task is complete, and then returns the result to the upper layer business.**
- **URLSession:dataTask:didReceiveResponse:completionHandler: processes object -relevant information, such as the ETag value for the precheck of resumable download and the content-length field value to calculate the download progress.**

DownloadRequest is defined as follows:

```
#import <Foundation/Foundation.h>

typedef void(^DownloadProgressBlock)(int64_t bytesReceived, int64_t
totalBytesReceived, int64_t totalBytesExpectToReceived);
typedef void(^DownloadFailureBlock)(NSError *error);
typedef void(^DownloadSuccessBlock)(NSDictionary *result);

@interface Checkpoint : NSObject<NSCopying>

@property (nonatomic, copy) NSString *etag;     // The ETag value of
the resource.
@property (nonatomic, assign) unsigned long long totalExpectedLength;
   //The total size of the object.

@end

@interface DownloadRequest : NSObject

@property (nonatomic, copy) NSString *sourceURLString;      // The URL
 for the download.

@property (nonatomic, copy) NSString *headURLString;        // The URL
 used to obtain the original information of the object.

@property (nonatomic, copy) NSString *downloadFilePath;     // The
local path to the object.

@property (nonatomic, copy) DownloadProgressBlock downloadProgress
; // The download progress.

@property (nonatomic, copy) DownloadFailureBlock failure;   // The
callback that is sent after the download fails.

@property (nonatomic, copy) DownloadSuccessBlock success;   // The
callback that is sent after the download succeeds.

@property (nonatomic, copy) Checkpoint *checkpoint;         // The
checkpoint file that stores the ETag value of the object.

@end


@interface DownloadService : NSObject

+ (instancetype)downloadServiceWithRequest:(DownloadRequest *)request;

/**
 * Start the download.
 */
- (void)resume;

/**
 * Pause the download.
 */
- (void)pause;

/**
 * Cancel the download.
 */
- (void)cancel;
```

```
@end
```

The invocation code for the upper-layer business is as follows:

```
- (void)initDownloadURLs {
    OSSPlainTextAKSKPairCredentialProvider *pCredential = [[OSSPlainTe
xtAKSKPairCredentialProvider alloc] initWithPlainTextAccessKey:
OSS_ACCESSKEY_ID secretKey:OSS_SECRETKEY_ID];
    _mClient = [[OSSClient alloc] initWithEndpoint:OSS_ENDPOINT
credentialProvider:pCredential];

    // Generate the signed URL for the GET request.
    OSSTask *downloadURLTask = [_mClient presignConstrainURLW
ithBucketName:@"aliyun-dhc-shanghai" withObjectKey:OSS_DOWNLO
AD_FILE_NAME withExpirationInterval:1800];
    [downloadURLTask waitUntilFinished];
    _downloadURLString = downloadURLTask.result;

    // Generate the signed URL for the HEAD request.
    OSSTask *headURLTask = [_mClient presignConstrainURLWithBucketN
ame:@"aliyun-dhc-shanghai" withObjectKey:OSS_DOWNLOAD_FILE_NAME
httpMethod:@"HEAD" withExpirationInterval:1800 withParameters:nil];
    [headURLTask waitUntilFinished];

    _headURLString = headURLTask.result;
}

- (IBAction)resumeDownloadClicked:(id)sender {
    _downloadRequest = [DownloadRequest new];
    _downloadRequest.sourceURLString = _downloadURLString;       //
Set the resource URL.
    _downloadRequest.headURLString = _headURLString;
    NSString *documentPath = NSSearchPathForDirectoriesInDomains(
NSDocumentDirectory, NSUserDomainMask, YES).firstObject;
    _downloadRequest.downloadFilePath = [documentPath stringByAp
pendingPathComponent:OSS_DOWNLOAD_FILE_NAME];   //Set the local path
to the object you want to download.

    __weak typeof(self) wSelf = self;
    _downloadRequest.downloadProgress = ^(int64_t bytesReceived,
int64_t totalBytesReceived, int64_t totalBytesExpectToReceived) {
        // totalBytesReceived is the number of Bytes already cached by
 the client. totalBytesExpectToReceived is the total number of Bytes
that need to be downloaded.
        dispatch_async(dispatch_get_main_queue(), ^{
            __strong typeof(self) sSelf = wSelf;
            CGFloat fProgress = totalBytesReceived * 1.f / totalBytes
ExpectToReceived;
            sSelf.progressLab.text = [NSString stringWithFormat:@"%.
2f%%", fProgress * 100];
            sSelf.progressBar.progress = fProgress;
        });
    };
    _downloadRequest.failure = ^(NSError *error) {
        __strong typeof(self) sSelf = wSelf;
        sSelf.checkpoint = error.userInfo[@"checkpoint"];
    };
    _downloadRequest.success = ^(NSDictionary *result) {
        NSLog(@"The download succeeds.");
    };
    _downloadRequest.checkpoint = self.checkpoint;

    NSString *titleText = [[_downloadButton titleLabel] text];
```

```
    if ([titleText isEqualToString:@"download"]) {
        [_downloadButton setTitle:@"pause" forState: UIControlS
tateNormal];
        _downloadService = [DownloadService downloadServiceWithRequest
:_downloadRequest];
        [_downloadService resume];
    } else {
        [_downloadButton setTitle:@"download" forState: UIControlS
tateNormal];
        [_downloadService pause];
    }
}

- (IBAction)cancelDownloadClicked:(id)sender {
    [_downloadButton setTitle:@"download" forState: UIControlS
tateNormal];
    [_downloadService cancel];
}
```

Note:

The checkpoint file can be obtained from the failure callback when the download is paused or cancelled. When you restart the download, you can upload the checkpoint file to DownloadRequest. Then, DownloadService uses the checkpoint file to verify the consistency.

10.8 Manage objects

10.8.1 Overview

You can use APIs to perform the following operations on objects in a bucket:

- *Determine whether the specified object exists*
- *Obtain Object Meta*
- *List objects*
- *Copy an object*
- *Delete objects*

10.8.2 Determine whether the specified object exists

The OSS iOS SDK provides a convenient synchronous API to check whether the specified object exists in the bucket.

Use the following code to determine whether an object exists:

```
NSError * error = nil;
BOOL isExist = [client doesObjectExistInBucket:TEST_BUCKET withObject
Key:@"file1m" withError:&error];
if (! error) {
    if(isExist) {
```

```
        NSLog(@"File exists.");
    } else {
        NSLog(@"File not exists.");
    }
} else {
    NSLog(@"Error!") ;
}
```

10.8.3 Obtain object metadata

The HeadObject method can obtain only the object metadata but not the content of the object.

Note:

- **The object metadata includes HTTP headers and user metadata. For more information, see *Manage object metadata* in OSS Developer Guide.**
- **You cannot set or modify the object metadata by using the iOS SDK.**

The following code provides an example on how to obtain the object metadata:

```
OSSHeadObjectRequest * request = [OSSHeadObjectRequest new];
request.bucketName = @"<bucketName>;
//objectKey indicates the name of the object, that is, the complete
path of the object. objectKey must include the file extension of the
object. For example, set objectKey to abc/efg/123.jpg.
request.objectKey = @"<objectKey>";

OSSTask * headTask = [client headObject:request];

[headTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"head object success!") ;
        OSSHeadObjectResult * result = task.result;
        NSLog(@"header fields: %@", result.httpResponseHeaderFields);
        for (NSString * key in result.objectMeta) {
            NSLog(@"ObjectMeta: %@ - %@", key, [result.objectMeta
objectForKey:key]);
        }
    } else {
        NSLog(@"head object failed, error: %@" ,task.error);
    }
    return nil;
}];
```

10.8.4 List objects

This topic describes how to list all objects in a bucket.

Use the following code to list all objects in a bucket:

```
OSSGetBucketRequest * getBucket = [OSSGetBucketRequest new];
getBucket.bucketName = @"<bucketName>";
```

```
// Set optional parameters. For more information, see https://docs.
aliyun.com/#/pub/oss/api-reference/bucket&GetBucket.
// getBucket.marker = @"";
// getBucket.prefix = @"";
// getBucket.delimiter = @"";

OSSTask * getBucketTask = [client getBucket:getBucket];

[getBucketTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        OSSGetBucketResult * result = task.result;
        NSLog(@"get bucket success!") ;
        for (NSDictionary * objectInfo in result.contents) {
            NSLog(@"list object: %@", objectInfo);
        }
    } else {
        NSLog(@"get bucket failed, error: %@", task.error);
    }
    return nil;
}];
```

The following table lists the configurable parameters and their descriptions returned by the list operation:

Name	Description
delimiter	**The delimiter used to separate object names. CommonPrefixes specify a set of substrings that start with a specified prefix and end with the first specified delimiter.**
marker	**The initial object in the list.**
maxkeys	**The maximum number of objects that can be listed. The default value is 100. The maximum value is 1,000.**
prefix	**The prefix that must be included in the returned objects. Note that if you use a prefix to make a query, the returned key values still contain the prefix.**

10.8.5 Copy an object

You can copy an object from a bucket (source bucket) to another bucket (destination bucket) within the same region.

Use the following code to copy an object:

```
OSSCopyObjectRequest * copy = [OSSCopyObjectRequest new];
copy.bucketName = @<bucketName>;
copy.objectKey = @"<objectKey>";
copy.sourceCopyFrom = [NSString stringWithFormat:@"/%@/%@", @"<
bucketName>, @"<objectKey_copyFrom>"];
```

```
OSSTask * task = [client copyObject:copy];

[task continueWithBlock:^id(OSSTask *task) {
 if (! task.error) {
        // ...
    }
    return nil;s
}];
```

Notice:

- **The source and destination objects must belong to the same region.**
- **During the copy operation, if the source and destination objects have the same IP address, you can modify the Object Meta of the existing objects.**
- **You cannot use CopyObject to copy an object whose size exceeds 1 GB. To copy an object larger than 1 GB, use multipart upload.**

10.8.6 Delete an object

This topic describes how to delete an object.

Warning:

Deleted objects cannot be recovered. Use caution when deleting objects.

Use the following code to delete an object:

```
OSSDeleteObjectRequest * delete = [OSSDeleteObjectRequest new];
delete.bucketName = @"<bucketName>";
delete.objectKey = @"<objectKey>";

OSSTask * deleteTask = [client deleteObject:delete];

[deleteTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        // ...
    }
    return nil;
}];

// [deleteTask waitUntilFinished];
```

Note:

To delete the object, you must have the write permission on the bucket where the object resides.

10.9 Data security

The iOS SDK provides data integrity checks to secure data during uploads and downloads.

Data integrity checks during file uploads

Due to the complexities in mobile network environments, an error can occur when data is transmitted between the client and the server. The OSS iOS SDK provides end-to-end data integrity checks based on MD5 and CRC64 values.

- **MD5 verification**

  You must provide the Content-MD5 value of the file during the upload. The OSS server performs the MD5 verification after the file is uploaded. The upload can succeed only when the MD5 value of the file received by the OSS server is equal to the MD5 value provided before the upload. In this way, the integrity of uploaded data is guaranteed.

```
OSSPutObjectRequest * request = [OSSPutObjectRequest new];
request.bucketName = BUCKET_NAME;
...
request.contentMd5 = [OSSUtil fileMD5String:filepath];
```

- **CRC**

  Compared with MD5 values, CRC64 values can be calculated during file uploads.

```
// Construct an upload request.
OSSPutObjectRequest * request = [OSSPutObjectRequest new];
request.bucketName = OSS_BUCKET_PRIVATE;
///....
request.crcFlag = OSSRequestCRCOpen;
// After the CRC is enabled, If data inconsistency is found during
the upload, OSSClientErrorCodeInvalidCRC is thrown.
OSSTask * task = [_client putObject:request];
[[task continueWithBlock:^id(OSSTask *task) {
    // If the CRC fails, an error is reported.
    XCTAssertNil(task.error);
    return nil;
}] waitUntilFinished];
```

Data integrity checks during file downloads

The OSS iOS SDK provides end-to-end data integrity checks based on CRC64 values.

When you read a downloaded data stream, the data integrity check is automatically performed if the CRC is enabled.

Use the following code to enable the CRC:

```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
request.bucketName = ... ;
// Enable the CRC.
request.crcFlag = OSSRequestCRCOpen;

OSSTask * task = [testProxyClient getObject:request];

[[task continueWithBlock:^id(OSSTask *task) {
 // If the CRC is enabled and a data error occurs during transmission
, OSSClientErrorCodeInvalidCRC is thrown.
    XCTAssertNil(task.error);
    return nil;
}] waitUntilFinished];

Note: If the onReceiveData block field is set, you must check the
consistency of CRC64 values after the CRC is enabled. Example:
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
request.bucketName = ....
request.crcFlag = OSSRequestCRCOpen;
....

__block uint64_t localCrc64 = 0;
 // If the onReceiveData block field is set,
NSMutableData *receivedData = [NSMutableData data];
request.onRecieveData = ^(NSData *data) {
 if (data)
 {
  NSMutableData *mutableData = [data mutableCopy];
     void *bytes = mutableData.mutableBytes;
     localCrc64 = [OSSUtil crc64ecma:localCrc64 buffer:bytes length:
data.length];
     [receivedData appendData:data];
    }
};

__block uint64_t remoteCrc64 = 0;
OSSTask * task = [_client getObject:request];
[[task continueWithBlock:^id(OSSTask *task) {
 XCTAssertNil(task.error);
    OSSGetObjectResult *result = task.result;
    if (result.remoteCRC64ecma)
 {
     NSScanner *scanner = [NSScanner scannerWithString:result.
remoteCRC64ecma];
  [scanner scanUnsignedLongLong:&remoteCrc64];
        if (remoteCrc64 == localCrc64)
        {
         NSLog(@"CRC64 verification succeeded!"); ;
        }
  else
        {
         NSLog(@"CRC64 verification failed!"); ;
        }
   }
   return nil;
```

```
}] waitUntilFinished];
```

10.10 Signed URLs

The SDK supports the signed URLs with a validity period or public URLs so that the URL can be forwarded to a third party for authorized access.

Generated a signed private resource URL with a validity period

If the bucket or object ACL is Private, you must call the following API to obtain the signed URL:

```
NSString * constrainURL = nil;

// Generate the signed URL.
OSSTask * task = [client presignConstrainURLWithBucketName:@"<bucket
name>"
                                                withObjectKey:@"<object
key>"
                                        withExpirationInterval: 30 * 60];
if (! task.error) {
    constrainURL = task.result;
} else {
    NSLog(@"error: %@", task.error);
}
```

Generated a signed public URL

If the bucket or object ACL is Public Read, you must call the following API to obtain the URL of the publicly accessible object:

```
NSString * publicURL = nil;

// Generate the signed URL.
task = [client presignPublicURLWithBucketName:@"<bucket name>"
        withObjectKey:@"<object key>"];
if (! task.error) {
 publicURL = task.result;
} else {
 NSLog(@"sign url error: %@", task.error);
}
```

10.11 Authorized access

The iOS SDK provides STS authentication and self-signed modes to secure mobile devices.

STS authentication mode

You can use Alibaba Cloud STS to authorize temporary access to OSS. STS is a Web service that provides a temporary access token to a cloud computing user. You can

use STS to assign a third-party application or a federated user (you can manage the user ID) an access credential with a custom validity period and permissions . The access credential in the application is called a federation token. The third-party application or federated user can use the access credential to directly call the Alibaba Cloud service APIs or use the Alibaba Cloud service SDKs to access the cloud service APIs.

- STS benefits:
  - You do not need to expose your long-term key (AccessKey) to a third-party application. You need only to generate an access token and send the access token to the third-party application. You can customize the access permission s and validity period of this token.
  - The access token automatically becomes invalid after expiration.
- How to use STS
  1. Log on to AppServer as an application user.

     You can manage your application ID. You can customize the ID management system or use an external Web account or open ID. For each valid application user, AppServer can precisely define their minimum access permissions.
  2. AppServer requests a security token from STS.
     a. Before calling STS, AppServer needs to determine the minimum access permissions (described in the policy syntax) of the application user and the expiration time of the authorization.
     b. Then, you can call the *AssumeRole* of STS to obtain the security token. For more information about how to manage and use roles, see *Role management* in RAM User Guide.
  3. STS returns a valid access credential to AppServer. The access credential contains a security token, a temporary AccessKey pair (AccessKey ID and AccessKey Secret), and the expiration time.
  4. AppServer returns the access credential to ClientApp.

     ClientApp caches this credential. When the credential becomes invalid, ClientApp must request a new valid access credential from AppServer. For

example, if the access credential is valid for one hour, ClientApp can request AppServer to update the access credential every 30 minutes.

5. ClientApp uses the locally cached access credential to request Alibaba Cloud service APIs. Cloud services detect the STS access credential. STS verifies the credential, and correctly responds to user requests.

Notice:

To use this authentication mode, you need to first activate Alibaba Cloud RAM, as shown in *Alibaba Cloud RAM*.

- **Set an STS token directly**

  Before you log on to ClientApp, you can use a certain method (such as sending network requests from your AppServer) to obtain an STS token. Then, use the token to initialize the SDK. If this method is adopted, you must take note of the validity period of the STS token. When the STS token is about to expire, you must manually update the STS token for the SDK.

  The initialization code is as follows:

```
id<OSSCredentialProvider> credential = [[OSSStsTokenCredentia
lProvider alloc] initWithAccessKeyId:@"<StsToken.AccessKeyId>"
secretKeyId:@"<StsToken.SecretKeyId>" securityToken:@"<StsToken.
SecurityToken>"];
client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential];
```

  When you notice that the token is about to expire, you can rebuild a new OSSClient or use the following method to update the CredentialProvider field value:

```
id<OSSCredentialProvider> credential = [[OSSStsTokenCredentia
lProvider alloc] initWithAccessKeyId:@"<StsToken.AccessKeyId>"
secretKeyId:@"<StsToken.SecretKeyId>" securityToken:@"<StsToken.
SecurityToken>"];
client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential];
```

- **Implement a callback to obtain and update an STS token**

  If you want the SDK to automatically help you manage token updates, you must configure the SDK to obtain the token. To implement a callback, you must obtain a federation token (STS token) and send the token to the SDK. The

SDK signs the token. When the token needs to be updated, the SDK calls the callback to obtain the token, as shown in the following figure.

```
id<OSSCredentialProvider> credential = [[OSSFederationCredent
ialProvider alloc] initWithFederationTokenGetter:^OSSFederat
ionToken * {
    // Here you need to obtain a federation token, construct it
into the OSSFederationToken object, and send the object to the SDK
.
    // If the token fails to be obtained for some reason, nil is
returned.
      OSSFederationToken * token;
    // The following code is used to obtain the token from
AppServer:
    ...
    return token;
}];
client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential];
```

Note:

Additionally, if you use other means to obtain all fields required by the token, you can directly send the token to the SDK and implement the callback. In this case, you need to manually update the token and then reset the value of OSSCredentialProvider of the OSSClient instance.

Example:

Assume that your server address is http://localhost:8080/distribute-token.json . The following data is returned after the address is accessed:

```
{
    "StatusCode": 200,
    "AccessKeyId":"STS.iA645eTOXEqP3cg3****",
    "AccessKeySecret":"rV3VQrpFQ4BsyHSAvi5NVLpPIVffDJv4Loj****",
    "Expiration":"2015-11-03T09:52:59Z",
    "SecurityToken":"CAES7QIIARKAAZPlqaN9ILiQZPS+JDkS/GSZN45RLx4YS
/p3OgaUC+oJl3XSlbJ7StKpQ...."}
```

You can use the following code to implement the `OSSFederationCredentialProvider` instance:

```
id<OSSCredentialProvider> credential2 = [[OSSFederationCredent
ialProvider alloc] initWithFederationTokenGetter:^OSSFederat
ionToken * {
    // Construct a request to access your AppServer.
    NSURL * url = [NSURL URLWithString:@"http://localhost:8080/
distribute-token.json"];
    NSURLRequest * request = [NSURLRequest requestWithURL:url];
    OSSTaskCompletionSource * tcs = [OSSTaskCompletionSource
taskCompletionSource];
    NSURLSession * session = [NSURLSession sharedSession];
```

```
    //Send the request.
    NSURLSessionTask * sessionTask = [session dataTaskWithRequest:
request
                                                  completionHandler
:^(NSData *data, NSURLResponse *response, NSError *error) {
                                                      if (error) {
                                                          [tcs
setError:error];
                                                          return;
                                                      }
                                                      [tcs setResult
:data];
                                                  }];
    [sessionTask resume];
    // Wait until the response is returned.
    [tcs.task waitUntilFinished];
    // Parse the result.
    if (tcs.task.error) {
        NSLog(@"get token error: %@", tcs.task.error);
        return nil;
    } else {
        // The returned data is in the JSON format and you must
parse it to obtain all fields of the token.
        NSDictionary * object = [NSJSONSerialization JSONObject
WithData:tcs.task.result

options:kNilOptions

error:nil];
        OSSFederationToken * token = [OSSFederationToken new];
        token.tAccessKey = [object objectForKey:@"AccessKeyId"];
        token.tSecretKey = [object objectForKey:@"AccessKeySecret
"];
        token.tToken = [object objectForKey:@"SecurityToken"];
        token.expirationTimeInGMTFormat = [object objectForKey:@"
Expiration"];
        NSLog(@"get token: %@", token);
        return token;
    }
}];
```

Self-signed mode

```
id<OSSCredentialProvider> credential = [[OSSCustomSignerCrede
ntialProvider alloc] initWithImplementedSigner:^NSString *(NSString *
contentToSign, NSError *__autoreleasing *error) {
    // Here you need to use the signature algorithm specified by OSS
to sign a string, add the AccessKey ID to the signed string, and send
the signed string to AppServer.
    // Typically, you can upload the string to AppServer. AppServer
returns the signature.
    // If the string fails to be signed, error information is recorded
, and nil is returned.
NSString *signature = [OSSUtil calBase64Sha1WithData:contentToSign
withSecret:@"<your accessKeySecret>"]; // The ossutil tool in the SDK
 is used to complete the local signature. We recommend that you use
AppServer to implement the remote signature.
    if (signature ! = nil) {
        *error = nil;
    } else {
        *error = [NSError errorWithDomain:@"<your domain>" code:-1001
userInfo:@"<your error info>"];
        return nil;
```

```
    }
    return [NSString stringWithFormat:@"OSS %@:%@", @"<your accessKeyI
d>", signature];
}];

client = [[OSSClient alloc] initWithEndpoint:endpoint credential
Provider:credential];
```

Notice:

For STS authentication and self-singed modes, you must ensure that the response is correct after you implement the callback. Therefore, if you send a network request to AppServer to obtain the token and signature, we recommend that you call the synchronous APIs of the network library or perform the conversion from async to sync. The callback is run in the subthread of the request from the SDK, which does not block the main thread.

10.12 IMG

Image Processing (IMG) is a secure, cost-effective, and highly reliable image processing service provided by OSS. IMG is capable of processing large amounts of images. After uploading source images to OSS, you can use Representational State Transfer (`RESTful`) APIs to process the images anytime, anywhere, and on any Internet device. You can create your own IMG service in OSS.

For more information about IMG, see *OSS Image Processing Guide*.

Basic features

IMG has the following features:

- *Resize images*
- *Crop images*
- *Rotate images*
- *Apply effects*
- *Convert formats*
- *Add watermarks*: **allows you to add image, text, or image-text watermarks to another image.**

How to use IMG

IMG uses the standard HTTP GET request. You can configure IMG parameters in QueryString of a URL.

If the ACL of an image object is Private, only authorized users are allowed access.

- **Anonymous access**

```
OSSTask * task = [_client presignPublicURLWithBucketName:_publicBuc
ketName
            withObjectKey:@"hasky.jpeg"
           withParameters:@{@"x-oss-process": @"image/resize,w_50
"}];
```

- **Authorized access**

  To use IMG in the SDK, you can call the `request.setxOssProcess()` method to set IMG parameters when downloading an image. Example:

```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
request.bucketName = @"<bucketName>";
request.objectKey = @"example.jpg";
// Process the image.
request.xOssProcess = @"image/resize,m_lfit,w_100,h_100";
OSSTask * getTask = [client getObject:request];
[getTask continueWithBlock:^id(OSSTask *task) {
    if (! task.error) {
        NSLog(@"download image success!") ;
        OSSGetObjectResult * getResult = task.result;
        NSLog(@"download image data: %@", getResult.dowloadedData);
    } else {
        NSLog(@"download object failed, error: %@" ,task.error);
    }
    return nil;
}];
// [getTask waitUntilFinished];
// [request cancel];
```

  Note:

  To perform other processing operations on the image, you need only to replace relevant parameters in `request.setxOssProcess()`.

- **Access with the SDK**

```
OSSGetObjectRequest * request = [OSSGetObjectRequest new];
request.bucketName = @"bucket-name";    // Set the bucket name.
request.objectKey = @"image-name";      // Set the image name.
request.xOssProcess = @"image/resize,m_lfit,w_100,h_100";   //
Process the image.
OSSTask * task = [ossClient getObject:request];
```

Implement persistent storage for a processed image

Use the following code to store a processed image in a bucket for persistent storage:

```
OSSImagePersistRequest *request = [OSSImagePersistRequest new];
request.fromBucket = _privateBucketName;
request.fromObject = OSS_IMAGE_KEY;
request.toBucket = _privateBucketName;
request.toObject = @"image_persist_key";
```

```
Request. action = @ "image/resize, w_100"; // Process the image.
//request.action = @"resize,w_100";

[[[ossClient imageActionPersist:request] continueWithBlock:^id
_Nullable(OSSTask * _Nonnull task) {

 return nil;
}] waitUntilFinished];
```

10.13 Exception handling

There are two types of exceptions in the iOS SDK: ClientError and ServerError.

ClientError indicates parameter and network errors. ServerError indicates abnormal responses from the OSS server.

Error type	Error domain	Code	UserInfo
ClientError	com.aliyun.oss.clientError	0	OSSClientErrorCodeNetworkingFailWithResponseCode0, indicating that the connection is abnormal.
		1	OSSClientErrorCodeSignFailed, indicating that the signature failure occurs.
		2	OSSClientErrorCodeFileCantWrite, indicating that the object fails to be written to the server.
		3	OSSClientErrorCodeInvalidArgument, indicating that the parameter is invalid.
		4	OSSClientErrorCodeNilUploadid, indicating that the upload ID for the resumable upload task fails to be obtained.
		5	OSSClientErrorCodeTaskCancelled, indicating that the task is cancelled.
		6	OSSClientErrorCodeNetworkError, indicating that the network is abnormal.
		7	OSSClientErrorCodeCannotResumeUpload, indicating that the resumable upload task fails.

Error type	Error domain	Code	UserInfo
		8	OSSClientErrorCodeNotKnown, indicating that an unknown error occurs.
ServerError	com.aliyun.oss.serverError	(-1 * httpResponseCode)	The directory obtained by parsing the corresponding XML file.

11 Node. js

11.1 Installation

This topic describes how to install Node.js. Demos in the official documents are based on SDK 6.X. For SDKs of earlier versions, see *Document for 5.X SDK*. For more information about upgrade your SDK to 6.X, see *Upgrade document*.

- *GitHub*
- *API document*
- *ChangeLog*

Environment preparations

- **Prerequisites**
  - **Activate OSS service. If you haven't activated or do not know about Alibaba Cloud OSS, log on to the OSS product page for more information.**
  - **The Alibaba Cloud account AccessKey gives you permission to access all APIs, thus, we recommend that you follow Alibaba Cloud Best Practices for creating an AccessKeyId and an AccessKeySecret. If you deploy the service on the server, you can use an RAM sub-account or STS for API access or daily O&M management and control. If you deploy the service on the client, we recommend that you use STS for API access. For more information, see *Access control*.**
- **Environment requirements**

  The OSS JavaScript SDK is built on the *Node.js* environment.

Usage methods

The OSS JavaScript SDK supports both synchronous and asynchronous use.

- **Synchronous mode: Synchronize the asynchronous mode based on `async` and `wait`.**
- **Asynchronous mode: Similar to the callback mode. The API returns the *Promise*, uses the `.then()` method to process the returned result and uses the `.catch()` method to process errors.**

The `new OSS()` method is used to create a client in both the synchronous mode and asynchronous mode.

Next is to name examples to upload an object first and then download the object respectively.

- **Synchronous mode**

```
let client = new OSS(...);
async function put () {
  try {
    // object indicates the uploaded object name, localfile
indicates a local file or file path.
    let r1 = await client.put('object','localfile');
    console.log('put success: %j', r1);
    let r2 = await client.get('object');
    console.log('get success: %j', r2);
  } catch(e) {
    console.error('error: %j', err);
  }
}
Put ();
```

- **Asynchronous mode**

```
let client = new OSS(...);

// object indicates the uploaded object name, localfile indicates a
local file or file path.
client.put('object', 'localfile').then(function (r1) {
  console.log('put success: %j', r1);
  return client.get('object');
}).then(function (r2) {
  console.log('get success: %j', r2);
}).catch(function (err) {
  console.error('error: %j', err);
});
```

Installation

- **Use Node.js**

  Supported Node.js versions:

  Node.js later than 8.0.0 is supported. Use ali-oss 4.x in Node.js environment earlier than 8.0.0.

  Use *npm* to install the SDK:

```
npm install ali-oss
```

  Use it in your project:

```
let OSS = require('ali-oss');

let client = new OSS({
```

```
  region: '<oss region>',
  //The Alibaba Cloud account AccessKey gives you permission to
 access all APIs, so we recommend to follow Alibaba Cloud Best
Practices for deploying the service on the server using an RAM sub-
account or STS, or deploying the service on the client using STS.
  accessKeyId: '<Your accessKeyId>',
  accessKeySecret: '<Your accessKeySecret>',
  bucket: '<Your bucket name>'
});
```

- **If you face any network problems when using npm, you can use the npm image provided by Taobao: c***npm*.

11.2 Options

This topic describes how to configure options.

OSS(options)

- **[accessKeyId] {String}: accessKeyId you create on Alibaba Cloud console website**
- **[accessKeySecret] {String}: accessKeySecret you create**
- **[bucket] {String}: the default bucket you want to access. If you do not have any bucket, use putBucket() to create one first.**
- **[endpoint] {String}: oss region domain. It takes priority over region.**
- **[region] {String}: the bucket data region location. The default value is oss-cn-hangzhou.**
- **[internal] {Boolean}: access OSS with Alibaba Cloud internal network or not. The default value is false. If your servers are running on Alibaba Cloud too, you can set it to true to reduce costs.**
- **[secure] {Boolean}: instruct OSS client to use HTTPS (secure: true) or HTTP (secure: false) protocol. For more information, click *here*.**
- **[timeout] {String|Number}: instance level timeout for all operations. The default value is 60s.**

```
var oss = require('ali-oss');

var store = oss({
  accessKeyId: 'your access key',
  accessKeySecret: 'your access secret',
  bucket: 'your bucket name',
  region: 'oss-cn-hangzhou'
```

```
});
```

11.3 Quick start

This topic describes how to quickly perform OSS operations (such as view the bucket list or upload objects) in the Node.js environment.

Note:

To facilitate changes, a file named *app.js* is created to demonstrate the code used to perform the following operations.

View the bucket list

Add the following code to the end of *app.js*, and then call the `listBuckets` interface to view the bucket list.

```
async function listBuckets () {
  try {
    let result = await client.listBuckets();
  } catch(err) {
    console.log(err)
  }
}

listBuckets();
```

You can run `node app.js` to view the result.

For more information about bucket-related APIs, see *GitHub*.

View the object list

Modify the `app.js` file as follows and call the `list` interface to view the object list.

```
client.useBucket('Your bucket name');
async function list () {
  try {
    let result = await client.list({
      'max-keys': 5
    })
    console.log(result)
  } catch (err) {
    console.log (err)
  }
}
list();
```

You can run `node app.js` to view the result.

Upload an object

Modify the `app.js` file as follows and call the `put` interface to upload a single object.

```
client.useBucket('Your bucket name');

async function put () {
  try {
    let result = await client.put('object-name', 'local file');
    console.log(result);
   } catch (err) {
     console.log (err);
   }
}

put();
```

Download an object

Modify the `app.js` file as follows and call the `get` interface to download a single object.

```
async function get () {
  try {
    let result = await client.get('object-name');
    console.log(result);
  } catch (err) {
    console.log (err);
  }
}

get();
```

Delete an object

Modify the `app.js` file and call the `delete` interface to delete an object.

```
async function delete () {
  try {
    let result = await client.delete('object-name');
    console.log(result);
  } catch (err) {
    console.log (err);
  }
}

delete();
```

For more information about object-related interfaces, see *GitHub*.

11.4 Initialization

This topic describes how to initialize the Node.js SDK.

Create an object named `app.js` and write the following content:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>'
});
```

The `region` field specifies the region that is set when you apply for OSS, such as `oss-cn-hangzhou`. For the complete list of regions, see *Endpoints and regions*.

Note:

If the desired endpoint is not included in the preceding list, you can use the following parameters to specify the endpoint:

- **internal: You must set the `region` field if the internal parameter is used. If you set `internal` to `true`, you can access the intranet nodes.**
- **secure: You must set the `region` field if the secure parameter is used. If you set `secure` to `true`, use HTTPS for access.**
- **endpoint: If you specify the `endpoint` field, such as `http://oss-cn-hangzhou.aliyuncs.com`, you do not need to set the `region` field. The `endpoint` field value can be a domain that uses HTTPS or an IP address.**
- **cname: You must set the `endpoint` field if the cname parameter is used. If you set `cname` to `true`, the `endpoint` field value is used as the custom domain that is bound to a bucket.**
- **bucket: If the `bucket` field value is not specified, you must first call `useBucket` to perform operations on the object that belongs to the bucket. You need to call this API only once.**
- **timeout: specifies the timeout period to access OSS APIs. The default value is 60 seconds.**

11.5 Buckets

11.5.1 Manage a bucket

A bucket is a namespace on OSS and a management entity for billing, access control, logging, and other advanced features.

View all buckets

You can use the `listBuckets` interface to list all the buckets belonging to the current user. You can specify the `prefix` parameter to list all buckets whose names contain the specified prefix.

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>'
});

async function listBuckets() {
  try {
    const result = await client.listBuckets();
    console.log(result);
    const result2 = await client.listBuckets({
      prefix: 'prefix',
    });
    console.log(result);
  } catch (err) {
    console.log(err);
  }
}

listBuckets();
```

Create a bucket

You can create a bucket using the `putBucket` interface. You require to specify the bucket name:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>'
});

// putBucket
async function putBucket() {
  try {
    const result = await client.putBucket('your bucket name');
    console.log(result);
  } catch (err) {
```

```
    console.log(err);
  }
}

putBucket();
```

Notice:

- **For bucket naming conventions, see *OSS Basic Concepts*.**
- **This is because the name of a bucket is globally unique, make sure your bucket name does not match with the other users' .**

Delete a bucket

You can use the `deleteBucket` interface to delete a bucket. You require to specify the bucket name:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>'
});

async function deleteBucket() {
  try {
    const result = await client.deleteBucket('your bucket name');
    console.log(result);
  } catch (err) {
    console.log(err);
  }
}

deleteBucket();
```

Notice:

- **If the bucket to be deleted has objects in it, you must delete the objects first.**
- **If the bucket to be deleted has unfinished upload requests, you must cancel the requests first using the `listUploads` and `abortMultipartUpload` interfaces.**

Bucket access permissions

You can set the ACL policy of a bucket to allow or prohibit anonymous reads/writes to the bucket. For more information on ACL, see *ACL*.

- **Obtain the ACL of a bucket**

  You can use `getBucketACL` to view the ACL policy of a bucket:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>'
});

async function getBucketACL() {
  try {
    const result = await client.getBucketACL('luozhang002');
    console.log(result);
  } catch (err) {
    console.log(err);
  }
}

getBucketACL();
```

- **Set the ACL policy of a bucket**

  You can use `putBucketACL` to configure the ACL policy of a bucket:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>'
});

async function putBucketACL() {
  try {
    const result = await client.putBucketACL('bucket name', 'public-
read');
    console.log(result);
  } catch (err) {
    console.log(err);
  }
}

putBucketACL();
```

11.5.2 CORS

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to different regions. OSS provides CORS operations to facilitate cross-origin access control.

Note:

For more information, see *#unique_56* in OSS Developer Guide and visit *GitHub*.

Configure CORS rules

The following code provides an example on how to configure CORS rules for a specific bucket:

```
let oss = require('ali-oss');

const client = oss({
  accessKeyId: 'your access key',
  accessKeySecret: 'your access secret',
  bucket: 'your bucket name',
  region: 'oss-cn-hangzhou'
})
client.putBucketCORS('hello', [
  {
    allowedOrigin: '*',
    allowedMethod: [
      'GET',
      'HEAD',
    ],
  }
]).then((result) => {});
```

For more information about how to configure CORS rules, see *#unique_57*.

Query CORS rules

The following code provides an example on how to query CORS rules configured for a specific bucket:

```
let OSS = require('ali-oss');
const client = new OSS({
  region: 'your region',
  accessKeyId: 'your accessKeyId',
  accessKeySecret: 'your accessKeySecret',
})
client.getBucketCORS('bucketName').then((res) => {
  console.log(res);
}).catch(e => {
  console.log(e)
})
```

For more information about how to query CORS rules, see *#unique_476*.

Delete CORS rules

The following code provides an example on how to delete the CORS rules configured for a specific bucket:

```
let OSS = require('ali-oss');
const client = new OSS({
  region: 'your region',
  accessKeyId: 'your accessKeyId',
  accessKeySecret: 'your accessKeySecret',
})
client.deleteBucketCORS('bucketName').then((res) => {
  console.log(res);
```

```
}).catch(e => {
  console.log(e)
})
```

For more information about how to delete CORS rules, see *#unique_477*.

11.5.3 Set ACLs

This topic describes how to set ACLs.

OSS allows you to set ACLs for buckets and objects respectively. This helps to easily control external access to your resources. You can set three ACLs for a bucket:

- **public-read-write: Anonymous users are allowed to create/retrieve/delete objects in the bucket.**
- **public-read: Anonymous users are allowed to retrieve objects in the bucket.**
- **private: Anonymous users are not allowed to access objects in the bucket. Signature is required for all accesses.**

When a bucket is created, the private ACL applies by default. You can use `putBucketACL` **to set bucket ACLs and use** `getBucketACL` **to get the bucket ACL.**

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function bucketACL () {
  try {
    let result = await client.getBucketACL('bucket-name');
    console.log(result);

    let result = await client.putBucketACL('bucket-name', 'acl');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

bucketACL();
```

You can set four ACLs for an object.

- **default: The object inherits the ACL for the bucket it belongs to, that is, the ACL for the object is the same as that for the bucket where the object is stored.**
- **public-read-write: Anonymous users are allowed to read/write the object.**
- **public-read: Anonymous users are allowed to read the object.**

- **private: Anonymous users are not allowed to access objects in the bucket. Signature is required for all accesses.**

When an object is created, the default ACL applies by default. You can use `putACL` to set the object ACL.

```
let OSS = require('ali-oss')
let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});
async function bucketACL () {
  try {
     let result = await client.getACL('my-object');
     console.log(result.acl); // default
     await client.putACL('my-object', 'public-read');
     result = yield client.getACL('my-object');
     console.log(result.acl); // public-read
  } catch (e) {
    console.log(e);
  }
}
```

Notice:

1. **If an object is configured with an ACL policy (not default), the object ACL takes priority during permission authentication when it is accessed. The bucket ACL is ignored.**
2. **If anonymous access is allowed (public-read or public-read-write is configured for the object), you can directly access the object using a browser.**

   ```
   For example: http://bucket-name.oss-cn-hangzhou.aliyuncs.com/
   object.jpg.
   ```

For more information about ACL, see *Access Control*.

11.5.4 Manage lifecycle rules

OSS allows you to set lifecycle rules for buckets to automatically remove expired objects and save the storage space.

You can configure lifecycle rules for OSS buckets or objects to automatically delete expired objects and parts or change the storage class of expired objects to IA or Archive, saving storage costs. A lifecycle rule includes the following elements:

- **Rule ID: Identifies a lifecycle rule. A rule ID is unique in a bucket.**

- **Strategy: You can set one of the following two application strategies for a rule. The strategy for all rules set for a bucket must be the same.**
  - **The rule applies to objects with specified prefixes. You can create multiple rules with this strategy. Prefixes specified in different rules must be different.**
  - **The rule applies to the entire bucket. You can create only one rule with this strategy.**
- **Expiration period:**
  - **Specifies the expiration period (N days) of a lifecycle rule. An object expires N days after it is modified for the last time.**
  - **Specifies an expiration time. Objects modified before the time expire.**
- **Status: Indicates whether the rule is enabled or disabled.**

For more information, see *Lifecycle rules*.

Configure lifecycle rules

You can use `putBucketLifecycle` **to configure lifecycle rules.**

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function putBucketLifecycle () {
  try {
    let result = yield client.putBucketLifecycle('bucket-name', [
    {
      id: 'rule1',
      status: 'Enabled',
      prefix: 'foo/',
      days: 3
    },
    {
      id: 'rule2',
      status: 'Disabled',
      date: '2022-10-11T00:00:00.000Z'
    }
  ]);
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

putBucketLifecycle();
```

View lifecycle rules

You can use `getBucketLifecycle` to view lifecycle rules:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function getBucketLifecycle () {
  try {
    let result = await client.getBucketLifecycle('bucket-name');
  console.log(result);
  } catch (e) {
    console.log(e);
  }
}

getBucketLifecycle();
```

Clear lifecycle rules

You can use `deleteBucketLifecycle` to set a null Rule array to clear the lifecycle rules:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function deleteBucketLifecycle () {
  try {
    let result = await client.deleteBucketLifecycle('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

deleteBucketLifecycle();
```

11.5.5 Set logging

OSS allows you to configure bucket access logging to store the bucket access logs in a specified bucket for future analysis. The logs are stored as objects in the specified bucket.

The object name format is as follows:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

The `TargetPrefix` needs to be specified by the user during configuration. For more information about access logging, see *Bucket Access Logging*.

Enable bucket logging

Use `putBucketLogging` to enable bucket logging:

```
let OSS = require('ali-oss')
let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});
async function putBucketLogging () {
  try {
     let result = await client.putBucketLogging('bucket-name', 'logs
/');
     console.log(result)
  } catch (e) {
    console.log(e)
  }
}
putBucketLogging();
```

View bucket logging settings

Use `getBucketLogging` to view bucket logging settings:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function getBucketLogging() {
  try {
     let result = await client.getBucketLogging('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
```

```
  }
})

getBucketLogging();
```

Disable bucket logging

Use `deleteBucketLogging` to disable bucket logging:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function deleteBucketLogging () {
  try {
    let result = await client.deleteBucketLogging('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }

deleteBucketLogging();
```

11.5.6 Hotlink protection

This topic describes how to use hotlink protection.

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about hotlink protection, see *#unique_49*.

Configure a referer whitelist

Use `putBucketReferer` to set a referer whitelist:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});
```

```
async function putBucketReferer () {
  try {
  let result = await client.putBucketReferer('bucket-name', true, [
    'my-domain.com',
    '*.example.com'
  ]);
  console.log(result);
  } catch (e) {
    console.log(e);
  }
 }

putBucketReferer();
```

View a referer whitelist

Use `getBucketReferer` to view a referer whitelist:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function getBucketReferer () {
  try {
    let result = await client.getBucketReferer('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

getBucketReferer();
```

Clear a referer whitelist

Use `deleteBucketReferer` to clear a referer whitelist:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function deleteBucketReferer () {
  try {
    let result = await client.deleteBucketReferer('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
```

```
}

deleteBucketReferer();
```

11.5.7 Static website hosting

You can set your bucket to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

As mentioned in *CNAME*, OSS allows you to direct your domain name to the OSS service address. By this means, OSS bucket is accessed when you access your website. You must specify the names of the objects in the bucket corresponding to the homepage (index page) and error page of the website respectively.

For more information, see *Static Website Hosting*.

Set website hosting

Use `putBucketWebsite` to set website hosting:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function putBucketWebsite () {
  try {
    let result = await client.putBucketWebsite('bucket-name', {
    index: 'index.html',
    error: 'error.html'
  });
   console.log(result);
  } catch (e) {
    console.log(e);
  }
}

putBucketWebsite();
```

View website hosting

Use `getBucketWebsite` to view website hosting:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
```

```
  bucket: '<Your bucket name>'
});

async function getBucketWebsite () {
  try {
    let result = await client.getBucketWebsite('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

getBucketWebsite();
```

Clear website hosting

Use `deleteBucketWebsite` to clear website hosting:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function deleteBucketWebsite() {
  try {
    let result = await client.deleteBucketWebsite('bucket-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

deleteBucketWebsite();
```

11.5.8 CNAME

This topic describes how to bind a custom domain name.

OSS supports binding custom domain names to OSS, so that you can seamlessly migrate data storage to OSS. For example, if your domain name is "my-domain.com" and all your images were previously located at `http://img.my-domain.com/x.jpg`, after you've moved your storage to OSS and linked your custom domain name, the images can still be accessed at the original address.

- **Activate OSS and create a bucket.**
- **Modify the DNS configuration, add a CNAME record, and direct img.my-domain.com to an endpoint of OSS (for example, my-bucket.oss-cn-hangzhou.aliyuncs.com).**
- **On the *OSS console*, bind img.my-domain.com to the created bucket.**

- **Upload images to the created bucket in OSS.**

By following the preceding steps, you can use the original address *http://img.my-domain.com/xx.jpg* **to access the images on OSS. For more information on how to bind custom domain names, see** *Bind Custom Domain Names*.

The SDK supports using a CNAME as the endpoint. Therefore, set the `cname` **parameter to true.**

```
let OSS = require('ali-oss')

let client = new OSS({
  endpoint: '<Your endpoint>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  cname: true
});

client.useBucket('my-bucket')
```

Notice:

The list_buckets interface is unavailable when a CNAME is used as the endpoint, because the custom domain name has been bound to a specific bucket.

11.6 Upload objects

11.6.1 Overview

Objects are the basic unit for data operations in OSS. The OSS Node.js SDK provides the following methods to upload objects:

- *Local file upload*: **supports the upload of a file up to 5 GB in size.**
- *Upload from local memory*: **supports the upload of a file up to 5 GB in size.**
- *Streaming upload*: **supports any file sizes.**
- *Multipart upload*: **supports the upload of a file up to 48.8 TB in size. This method is suitable for the upload of large files.**
- *Resumable upload*: **supports the upload of a file up to 48.8 TB in size. This method is suitable for the upload of large files. It supports concurrent upload and you can define the size of each part.**

Note:

For more information about the scenarios of each upload method, see the "Upload files" section in OSS Developer Guide.

11.6.2 Upload a local file

This topic describes how to upload a local file to OSS.

Note:

The following sample code uses the catch syntax. Learn ES6 Promise and ES6 Async/Await on your own. For more information about how to use the SDK, see *Installation*.

You can use `put` to upload a local file to OSS:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function put () {
  try {
    let result = await client.put('object-name', 'local-file');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

put();
```

11.6.3 Upload from local memory

This document describes how to upload content from local memory to OSS.

Note:

The following sample code uses the catch syntax. Learn ES6 Promise and ES6 Async/Await on your own. For more information about how to use the SDK, see *Installation*.

You can use `put` to upload content from local memory to OSS:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
```

```
  bucket: 'Your bucket name'
});

async function putBuffer () {
  try {
    let result = await client.put('object-name', new Buffer('hello
world'));
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

putBuffer();
```

11.6.4 Streaming upload

This topic describes how to perform streaming upload.

Note:

The following sample code uses the catch syntax. Learn ES6 Promise and ES6 Async/Await on your own. For more information about how to use the SDK, see *Installation*.

You can use `putStream` **to upload the content of a stream. You can set** `stream` **to specify the types of** `Readable Stream`**, such as file stream and network stream. When you use** `putStream`**, the SDK initiates an HTTP PUT request for** `chunked encoding`**. If** `contentLength` **is specified in** `options`**,** `chunked encoding` **is not used.**

```
let OSS = require('ali-oss');
let fs = require('fs');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function putStream () {
  try {
  // use 'chunked encoding'
  let stream = fs.createReadStream('local-file');
  let result = await client.putStream('object-name', stream);
  console.log(result);

  // don't use 'chunked encoding'
  let stream = fs.createReadStream('local-file');
  let size = fs.statSync('local-file').size;
  let result = await client.putStream(
    'object-name', stream, {contentLength: size});
  console.log(result);
  } catch (e) {
    console.log(e)
  }
```

```
}

putStream();
```

11.6.5 Multipart upload

This topic describes how to use multipart upload.

Note:

The following sample code uses the catch syntax. Learn ES6 Promise and ES6 Async/Await on your own. For more information about how to use the SDK, see *Installation*.

To upload a large file, you can use `multipartUpload`. The advantage of multipart upload is to divide the upload request of a large file into multiple small requests and run each small request. This method guarantees that when some of the requests fail, you need only to upload the failed parts instead of the entire file. We recommend that you use multipart upload for files that are larger than 100 MB.

If `ConnectionTimeoutError` occurs during the use of multipartUpload, the business side must process the timeout logic on its own. You can scale in the part size, prolong the timeout period, retry the request, or analyze the specific cause by capturing `ConnectionTimeoutError`.

Relevant parameters are described as follows:

- **name {String}: specifies the object name.**
- **file {String|File}: specifies the file path or the HTML5 file.**

- [options] {Object}: specifies optional parameters.
  - [checkpoint] {Object}: specifies the checkpoint information. If this parameter is set, the upload starts from the recorded checkpoint information. If this parameter is not set, the entire file needs to be uploaded.
  - [parallel] {Number}: specifies the number of parts that are to be uploaded simultaneously.
  - [partSize] {Number}: specifies the size of each part.
  - [progress] {Function}: specifies the async function mode. The callback function involves three parameters:
    - (percentage {Number}: specifies the percentage of the upload progress (or a decimal from 0 to 1).
    - checkpoint {Object}: specifies the checkpoint information.
    - res {Object}): specifies the response to a part upload request.
  - [meta] {Object}: specifies the Object Meta you can set in the header. The prefix of Object Meta in the header is `x-oss-meta-`.
  - [mime] {String}: specifies the `Content-Type` value you can define in the header.
  - [headers] {Object}: specifies other field information in the header. For more information, see *RFC 2616*.
    - 'Cache-Control': specifies the cache mechanism that is added to HTTP request or response headers. The cache mechanism is implemented through commands. Example: `Cache-Control: public, no-cache`.
    - 'Content-Disposition': specifies the form in which the reply content is displayed. The reply content can be displayed on a Web page, or downloaded and saved locally as an attachment. Example: `Content-Disposition: somename`.
    - 'Content-Encoding': compresses the data of a specific medium type, such as `Content-Encoding: gzip`.
    - 'Expires': specifies the validity period, such as `Expires: 3600000`.

Use the following code for multipart upload:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
```

```
  bucket: 'Your bucket name'
});

async function multipartUpload () {
  try {
    let result = await client.multipartUpload('object-name', 'local-
file', {
    progress,
 meta: {
   year: 2017,
   people: 'test'
 }
  });
  console.log(result);
  let head = await client.head('object-name');
  console.log(head);
  } catch (e) {
   // Catch the timeout exception.
 if (e.code === 'ConnectionTimeoutError') {
   console.log("Woops,timeout exception!") ;
   // do ConnectionTimeoutError operation
 }
    console.log(e)
  }
}
```

`progress` **is a progress upload callback function that is used to obtain the upload progress.** `progress` **can be an async function:**

```
const progress = async function (p) {
  console.log(p);
};
```

`meta` **specifies Object Meta. You can use head to obtain Object Meta.**

11.6.6 Append upload

This topic describes how to append data from a local file.

You can use AppendObject to append content to appendable objects that have been uploaded.

Note:

- **Only objects that are created through append upload can have data appended to them. Therefore, objects created through simple upload, form upload, multipart upload, or resumable upload cannot have data appended to them.**
- **You cannot perform CopyObject for appendable objects.**

For more information about append upload, see *#unique_62* **in the OSS Developer Guide.**

Run the following code for append upload:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>',
});

async function append () {
  try {
    let result = await client.append('object-name', 'local-file');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

append();
```

Note:

`[position] {String}` **specifies the position calculated based on the content length of the object uploaded last time. For more information about how to configure other parameters, visit** *GitHub*.

11.6.7 Resumable upload

You can use resumable upload to split a file you want to upload into several parts and upload them simultaneously. After all parts are uploaded, you can combine these parts into a complete object. Thus, you can complete uploading the entire object.

Note:

The following sample code uses the catch syntax. Learn ES6 Promise and ES6 Async/Await on your own. For more information about how to use the SDK, see *Installation*.

For more information about resumable upload, see the "*Resumable upload***" section in OSS Developer Guide.**

Multipart upload provides `progress` **to facilitate making upload progress callback requests. The SDK uses the upload progress and the checkpoint information as parameters in the callback requests. To implement resumable upload, you can save the checkpoint information during the upload process. When an error occurs, you**

can pass the saved checkpoint information as a parameter to `multipartUpload`. The upload continues from where it failed earlier.

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

let checkpoint;
async function resumeUpload() {
  // Set the number of retry attempts to 5.
  for (let i = 0; i < 5; i++) {
    try {
      const result = await client.multipartUpload('object-name',
filePath, {
        checkpoint,
        async progress(percentage, cpt) {
          checkpoint = cpt;
        },
      });
      console.log(result);
      break; // break if success
    } catch (e) {
      console.log(e);
    }
  }
}

resumeUpload();
```

The preceding code saves the checkpoint information in the variable. If the program stops responding, the checkpoint information is lost. We recommend that you save the checkpoint information in a file. After the program restarts, the checkpoint information can be read from the file.

11.7 Download objects

11.7.1 Overview

The OSS Node.js SDK provides the following methods to download an object:

- *Download to a local file*
- *Download to local memory*
- *Streaming download*
- *HTTP-based download*

11.7.2 Download to a local file

This topic describes how to download an object from OSS to a local file.

You can use `get` to download an object from OSS to a local file:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function get () {
  try {
    let result = await client.get('object-name', 'local-file');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

get();
```

11.7.3 Download to local memory

This topic describes how to download an object to local memory.

You can use `get` to download an object from OSS to local memory:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function getBuffer () {
  try {
    let result = await client.get('object-name');
    console.log(result.content);
  } catch (e) {
    console.log(e);
  }
}

getBuffer();
```

11.7.4 Streaming download

If you have a large object to download or it is time-consuming to download the entire object at a time, you can use streaming download. Streaming download

enables you to download part of the object content each time until you download the entire object.

When you use `getStream` to download an object, `Readable Stream` is returned and used to stream the object content.

```
let OSS = require('ali-oss');
let fs = require('fs');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function getStream () {
  try {
    let result = await client.getStream('object-name');
    console.log(result);
    let writeStream = fs.createWriteStream('local-file');
    result.stream.pipe(writeStream);
  } catch (e) {
    console.log(e);
  }
}

getStream()
```

11.7.5 HTTP-based download

Aside from using the OSS Node.js SDK, you can directly use HTTP to download objects stored in OSS.

HTTP-based download supports browsers and command line interface (CLI) tools such as `wget` and `curl`. The object URL is generated through the SDK. Use `signatureUrl` to generate an HTTP-based address that can be used for downloads. The validity period of the URL is half an hour by default.

Use the following code to download an object from OSS over HTTP:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

let url = client.signatureUrl('object-name');
console.log(url);

let url = client.signatureUrl('object-name', {expires: 3600});
console.log(url);
```

```
// Generate the signed URL for PUT.
let url = client.signatureUrl('object-name', {method: 'PUT'});
console.log(url);
```

11.8 Manage objects

11.8.1 Overview

You can use APIs to perform

the following operations on objects in a bucket:

- *Determine whether an object exists*
- *Manage Object Meta*
- *List objects*
- *Copy an object*
- *Delete objects*

11.8.2 Determine whether an object exists

This topic describes how to determine whether an object exists.

Use `get` to determine whether an object exists:

```
client.get(object).then((result) => {
      if (result.res.status == 200) {
        return true
      }
  }).catch((e)=> {
    if (e.code == 'NoSuchKey') {
      return false
    }
  })
```

Note:

For more information about the use of `get`, see *Download an object*.

11.8.3 Manage Object Meta

Object Meta contains HTTP headers and User Meta. For more information, see the "*Object Meta*" section in OSS Developer Guide.

When you upload a file to OSS, you can specify the information of certain properties in addition to the content. The property information is called Object Meta. Object Meta is stored together with its object during uploads, and returned with the object when the object is downloaded.

Object Meta is stored in HTTP headers during object uploads or downloads. Therefore, Object Meta cannot contain complex characters in accordance with HTTP.

Note:

Object Meta can only contain simple characters in the ASCII format and cannot contain line breaks. The total size of Object Meta cannot exceed 8 KB.

You can set `meta` to specify Object Meta when you use `put`, `putStream`, or `multipartUpload`.

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function put () {
  try {
    let result = await client.put('object-name', 'local-file', {
    meta: {
      year: 2016,
      people: 'mary'
    }
  });
  console.log(result);
  } catch (e) {
    console.log(e);
  }
}

put();
```

You can also use `putMeta` to update Object Meta:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function putMeta () {
  try {
    let result = await client.putMeta('object-name', {
    meta: {
      year: 2015,
      people: 'mary'
    }
  });
  console.log(result);
```

```
  } catch (e) {
    console.log(e);
  }
}

putMeta();
```

11.8.4 List all objects in a bucket

This topic describes how to list all objects in a bucket.

Use `list` to list all objects in the current bucket. Parameters are described as follows:

- **prefix: only lists objects with the specified prefix.**
- **marker: only lists objects whose names come after the marker value.**
- **delimiter: separates object names. CommonPrefixes specify a set of substrings that start with a specified prefix and end with the first specified delimiter.**
- **max-keys: the maximum number of objects to be returned.**

Use the following code to list all objects in a bucket:

```
let OSS = require('ali-oss');
let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});
async function list () {
  {
    // If no parameter is specified, a maximum of 1,000 objects can be
 returned by default.
    let result = await client.list();
    console.log(result);
    // Continue to list objects based on the nextMarker field value.
    if (result.isTruncated) {
      let result = await client.list({
        marker: result.nextMarker
      });
    }
  // List all objects that are prefixed with "my-."
  let result = await client.list({
     prefix: 'my-'
  });
  console.log(result);
  // List all objects whose names are prefixed with "my-" and come
after the "my-object" marker.
  let result = await client.list({
     prefix: 'my-',
     marker: 'my-object'
  });
  console.log(result);
  } catch (e) {
    console.log(e);
  }
}
```

```
list();
```

11.8.5 Copy an object

This topic describes how to copy an object.

You can use `copy` to copy an object in either of the following situations:

- **The source and destination objects are in the same bucket.**
- **The source and destination objects are in different buckets within the same region. The source object name must be in the format of '/Bucket/object.'**

Additionally, you can process Object Meta as follows when you copy an object:

- **If `meta` is not specified, the Object Meta of the source object is copied.**
- **If `meta` is specified, the specified Object Meta overwrites the Object Meta of the source object.**

Use the following code to copy an object:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function copy () {
  try {
     // Copy the object from a bucket to another bucket.
    let result = await client.copy('to', '/from-bucket/from');
    console.log(result);

    // Copy Object Meta.
    let result = await client.copy('to', 'from');
    console.log(result);

    // Overwrite Object Meta.
    let result = await client.copy('to', 'from', {
      meta: {
        year: 2015,
        people: 'mary'
      }
    });
    console.log(result);
  } catch (e) {
    console.log(e);
  }
```

```
}
```

11.8.6 Delete objects

This topic describes how to delete objects.

Delete an object

Use `delete` to delete an object:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function delete () {
  try {
    let result = await client.delete('object-name');
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

delete();
```

Delete multiple objects

Use `deleteMulti` to delete multiple objects and use `quiet` to specify whether to return the deletion results:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function deleteMulti () {
  try {
    let result = await client.deleteMulti(['obj-1', 'obj-2', 'obj-3
']);
    console.log(result);
    let result = await client.deleteMulti(['obj-1', 'obj-2', 'obj-3'],
 {
    quiet: true
  });
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}
```

```
deleteMulti();
```

11.8.7 Restore an Archive object

This topic describes how to restore an Archive object.

You must restore an archive object before you read it. Do not call restoreObject for non-archive objects.

The state conversion process of an archive object is as follows:

1. **An archive object is in the frozen state.**
2. **After you submit it for restoration, the server restores the object. The object is in the restoring state. It takes about 1 minute to restore the object. However, a restoration task may take 4 hours to complete at most.**
3. **After being restored, the object is in the restored state. The restored state of the object lasts for 24 hours by default. If you call RestoreObject again during this period, the period of the restored state is automatically prolonged for 24 hours. You can call RestoreObject to restore an Archive object for 7 times in a restoratio n task so that the restored state of the object can be kept for 7 days in maximum.**
4. **Once the period of the restored state expires, the object returns to the frozen state.**

The following code provides an example of how to restore an Archive object:

Note:
For more information about how to restore an Archive object, visit *GitHub*.

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>',
});

client.restore('objectName').then((res) => {
    console.log(res);
}).catch(err => {
    console.log(err);
})
```

For more information about the Archive storage class, see *Introduction to storage classes*.

11.9 Authorized access

This topic describes how to authorize temporary access to OSS.

Use STS for temporary access authorization

You can use Alibaba Cloud STS to authorize temporary access to OSS. STS is a Web service that provides a temporary access token to a cloud computing user. You can use STS to assign a third-party application or a federated user (you can manage the user ID) an access credential with a custom validity period and permissions. For more information, see *STS introduction*.

STS benefits:

- **You do not need to expose your long-term key (AccessKey) to a third-party application. You need only to generate an access token and send the access token to the third-party application. You can customize the access permissions and validity period of this token.**
- **The access token automatically expires after the validity period.**

For more information about how to use STS to access OSS, see the "*Use STS to authorize temporary access to OSS*" section in OSS Developer Guide.

To use STS to access OSS, you must set the `stsToken` parameter. Example:

```
let OSS = require('ali-oss');
let STS = OSS.STS;
let sts = new STS({
  accessKeyId: '<The AccessKey ID of the RAM user>',
  accessKeySecret: '<The AccessKey Secret of the RAM user>'
});
async function assumeRole () {
  try {
    let token = await sts.assumeRole(
    '<role-arn>', '<policy>', '<expiration>', '<session-name>');
    let client = new OSS({
      region: '<region>',
      accessKeyId: token.credentials.AccessKeyId,
      accessKeySecret: token.credentials.AccessKeySecret,
      stsToken: token.credentials.SecurityToken,
      bucket: '<bucket-name>'
    });
  } catch (e) {
    console.log(e);
  }
}
assumeRole();
```

You can customize an STS policy when applying for a temporary token from STS. The temporary permission you apply for is `owned by your role and specified`

by the policy at the same time. **The following example specifies the STS policy used to apply for the read-only permission on** `my-bucket`**, and sets the validity period of the temporary token to 15 minutes:**

```
let OSS = require('ali-oss');
let STS = OSS.STS;
let sts = new STS({
  accessKeyId: '<The AccessKey ID of the RAM user>',
  accessKeySecret: '<The AccessKey Secret of the RAM user>'
});
let policy = {
  "Statement": [
    {
      "Action": [
        "oss:Get*"
      ],
      "Effect": "Allow",
      "Resource": ["acs:oss:*:*:my-bucket/*"]
    }
  ],
  "Version": "1"
};
async function assumeRole () {
  try {
    let token = await sts.assumeRole(
    '<role-arn>', policy, 15 * 60, '<session-name>');
    let client = new OSS({
      region: '<region>',
      accessKeyId: token.credentials.AccessKeyId,
      accessKeySecret: token.credentials.AccessKeySecret,
      stsToken: token.credentials.SecurityToken,
      bucket: '<bucket-name>'
    });
  } catch (e) {
    console.log(e);
  }
}
assumeRole();
```

Use a signed URL to authorize temporary access

- **Generate a signed URL**

  You can provide a signed URL to authorize temporary access for a visitor. When a signed URL is generated, you can specify the validity period for the URL to restrict the period of access from visitors.

- **Generate a signed URL for an object**

  Use the following code to generate a signed URL for an object:

  Note:

name {String} indicates the name of the object stored in OSS. [expires] {Number} indicates the validity period of the URL. The default value is 1,800 seconds. For more information about other parameters, see *GitHub*.

```
let OSS = require('ali-oss');
let store = new OSS({
    bucket: '<your bucket>',
    region: '<your region>',
    accessKeyId: '<your accessKeyId>',
    accessKeySecret: '<your accessKeySecret>'
})
const url = store.signatureUrl('ossdemo.txt');
console.log(url);
// --------------------------------------------------
const url = store.signatureUrl('ossdemo.txt', {
  expires: 3600,
  method: 'PUT'
});
console.log(url);

//  put object with signatureUrl
// --------------------------------------------------

const url = store.signatureUrl('ossdemo.txt', {
  expires: 3600,
  method: 'PUT',
  'Content-Type': 'text/plain; charset=UTF-8',
});
console.log(url);

// --------------------------------------------------
const url = store.signatureUrl('ossdemo.txt', {
  expires: 3600,
  response: {
    'content-type': 'text/custom',
    'content-disposition': 'attachment'
  }
});
console.log(url);

// put operation
```

- **Generated a signed URL for IMG**

```
let OSS = require('ali-oss');
let store = new OSS({
    bucket: '<your bucket>',
    region: '<your region>',
    accessKeyId: '<your accessKeyId>',
    accessKeySecret: '<your accessKeySecret>'
})
const url = store.signatureUrl('ossdemo.png', {
  process: 'image/resize,w_200'
});
console.log(url);
// --------------------------------------------------
const url = store.signatureUrl('ossdemo.png', {
  expires: 3600,
  process: 'image/resize,w_200'
});
```

```
console.log(url);
```

11.10 Image processing

Alibaba Cloud OSS provides image processing service featuring massive processing power, security, low costs, and high reliability. By uploading and storing source images in OSS, you can process images anytime, anywhere, and on any Internet device through a simple RESTful interface.

Image processing basic features

Image processing provides the following features:

- *Retrieve dominant image tones*
- *Format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**
- *Customize image processing styles*
- **Calling multiple image processing features through** *Image service access rules*.

Usage of image processing

Image processing uses the standard HTTP GET request for access. All processing parameters are encoded in the QueyString in the URL.

If the ACL for an image is private read and write, only authorized users are allowed can access the image.

- **Anonymous access**

  If the access permission of the image object is `public read`, as shown in the following table, you can access image processing anonymously.

Bucket ACL	Object ACL
Public-read or Public-read-write	Default

Bucket ACL	Object ACL
Any ACL	Public-read or Public-read-write

Access image processing anonymously using the third-level domain in the following format:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Name of the user’s bucket
endpoint	The access domain name for the data center of the user’s bucket
object	The image object uploaded to the OSS
image	The identifier reserved by the image processing
action	Operations on images, such as scaling, cropping, and rotating

Parameter	Description
param	Corresponding to the operations on the image

- Basic operations

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- Custom style.

  Access image processing anonymously using the third-level domain in the following format:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=style/<yourStyleName>
```

  - Style: The identifier reserved by the stem for the user’s custom style
  - yourStyleName: Name of the custom style, that is, the rule name of the console-defined style

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=style/oss-pic-style-w-100
```

- Through Image Service access rules,

  you can implement multiple operations on an image. The format is as follows:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss
-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<
yourParamValue2>/...
```

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100/rotate,90
```

- Image processing also supports HTTPS access.

  For example:

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

· Authorized access

For objects with private permissions, as shown in the following table, you must be authorized to access image processing:

Bucket ACL	Object ACL
private	default
Any ACL	private

The image processing URL code generated with signature is as follows:

```
let OSS = require('ali-oss');
let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.aliyuncs
.com>'
});
// Expired for 10 minutes. The image processing style is: "image/
resize,w_300".
let signUrl = client.signatureUrl('example.jpg', {expires: 600, '
process' : 'image/resize,w_300'});
console.log("signUrl="+signUrl);
```

Note:

- Authorized access supports `custom styles`, `HTTPS`, and `cascading processing`.
- Expiration time (expires) is measured in seconds.

- SDK access

  You can directly use SDK to access and process image objects with any permissions.

  Image object processing through SDK supports custom styles, HTTPS, and Image Service access rules.

  - Basic operations

    Run the following code to perform basic operations on an image:

```
let OSS = require('ali-oss');
let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.
aliyuncs.com>'
});
// Scale the image.
```

```
async function scale () {
  try {
    let result = await client.get('example.jpg', './example-resize
.jpg', { process: 'image/resize,m_fixed,w_100,h_100'});
  } catch (e) {
    console.log(e);
  }
}
// Crop the image.
async function cut () {
  try {
     let result = await client.get('example.jpg', './example-crop.
jpg', { process: 'image/crop,w_100,h_100,x_100,y_100,r_1'});
  } catch (e) {
    console.log(e)
  }
}
// Rotate the image.
async function rotate () {
  try {
    let result = await client.get('example.jpg', './example-rotate
.jpg', { process: 'image/rotate,90'});
  } catch (e) {
    console.log(e);
  }
}
// Sharpen the image.
async function sharpen () {
  try {
    let result = yield client.get('example.jpg', './example-
sharpen.jpg', { process: 'image/sharpen,100'});
  } catch (e) {
    console.log(e);
  }
}
// Add watermarks.
async function watermark () {
  try {
    let result = yield client.get('example.jpg', './example-
watermark.jpg', { process: 'image/watermark,text_SGVsbG8g5Zu-
54mH5pyN5YqhIQ'});
  } catch (e) {
    console.log(e);
  }
}
// Convert the image format.
async function format () {
  try {
    let result = await client.get('example.jpg', './example-format
.jpg', { process: 'image/format,png'});
  } catch (e) {
    console.log(e);
  }
}
// Obtain image information.
async function info () {
  try {
    let result = await client.get('example.jpg', './example-info.
txt', {process: 'image/info'});
  } catch (e) {
    console.log(e);
  }
```

```
}
```

- **Custom styles**

> **Note:**
> **You must first log on to the OSS console to add the custom style 'example-style'.**

You can run the following code to customize an image style:

```
let OSS = require('ali-oss');
let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.
aliyuncs.com>'
});
// Custom styles
async function style () {
  try {
    let result = await client.get('example.jpg', './example-custom
-style.jpg', {process: 'style/example-sytle"'});
  } catch (e) {
    console.log(e);
  }
}
style();
```

- **Cascading processing**

Run the following code to perform cascade operations on an image:

```
let OSS = require('ali-oss');
let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.
aliyuncs.com>'
});
// Cascading processing
async function cascade () {
  try {
    let result = await client.get('example.jpg', './example-
cascade.jpg', {process: 'image/resize,m_fixed,w_100,h_100/rotate,
90'});
  } catch (e) {
    console.log(e);
  }
}
```

```
cascade();
```

11.11 Exception handling

If a request error occurs when you use OSS Node.js SDK usage, an exception is thrown.Detailed information about the error is recorded in the log (`oss_sdk.log` by default).

ClientError

ClientErrors indicate exceptions occurred inside the SDK, for example, parameter setting errors and errors occurred when objects are modified in resumable upload or download tasks.

ServerError

ServerErrors are the parsing results of the error messages generated by the server and indicate errors occurred on the server. ServerErrors include the following attributes:

- **status: Indicates the HTTP status code of the error request.**
- **code: Indicates the OSS error code.**
- **message: Indicates the OSS error message.**
- **requestId: Indicates the UUID that uniquely identifies the request. Seek help from OSS development engineers by providing this requestId if necessary.**

For more information about common error messages in OSS, see *OSS Error Response*.

Debugging

If ClientErrors or ServerErrors occur when you use OSS Node.js SDK, enable debugging by setting the `DEBUG` environment variable as follows:

```
DEBUG=ali-oss node app.js
```

In a browser environment, set the `localStorage.debug` variable in the console to enable debugging:

```
localStorage.debug = 'ali-oss'
```

11.12 FAQ

This topic describes common problems that may occur in the use of OSS Node.js SDK and the solutions for the problems.

How to enable HTTPS access

To enable HTTPS access, specify the value of the `secure` parameter to `true` when initializing the SDK.

How to obtain the upload progress

You can specify the `progress` parameter to obtain the upload progress when performing *multipart upload* operations.

How to obtain the download progress

The download progress is calculated based on the size of the download traffic in OSS Node.js SDK.

How to upload Base64-encoded images

Convert the images to File objects and upload them to the OSS server by calling APIs .

```
function dataURLtoFile(dataurl, filename) {
   let arr = dataurl.split(','), mime = arr[0].match(/:(. *?) ;/)[1],
     bstr = atob(arr[1]), n = bstr.length, u8arr = new Uint8Array(n);
   while(n--){
     u8arr[n] = bstr.charCodeAt(n);
   }
   return new File([u8arr], filename, {type:mime});
 }

 let file = dataURLtoFile('<base64 content>', '');

 client.multipartUpload('<oss file name>', file).then( (res)=> {
   console.log(res)
 }).catch((err) => {
   console.log(err)
 });
```

How to upload a file to a specified directory

You only need to add the prefix of the directory to the name of the object that you want to upload. For more information, see: *Comparison of OSS and file system*.

```
let OSS = require('ali-oss')
let client = new OSS({
  region: '<Your region>',
```

```
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

client.multipartUpload('base-dir/' +'object-name', 'local-file', {
    progress: async function (p) {
      console.log('Progress: ' + p);
    }
  });
  console.log(result);
}).then((res) => {
  console.log(res)
}). catch((err) => {
  console.log(err);
});
```

How to obtain the signed URL of an object

You can call the `signatureUrl` method to obtain the signed URL used to download an object. For more information, see the *signatureUrl* part in the OSS SDK document.

Common errors

- *Exception handling*
- *#unique_172*

12 Browser.js

12.1 Installation

This topic describes how to install OSS Browser.js SDK.

Download the SDK

Demos in official documents are based on SDK 6.X. For SDKs earlier than 6.X, see *Documents for 5.X SDK*. For more information about how to upgrade your SDK to 6.X, see *Upgrade document*.

- *GitHub*
- *API documentation*
- *ChangeLog*

Environment preparations

- **Requirements**
  - **You must have activated Alibaba Cloud OSS. If you do not know about Alibaba Cloud OSS, log on to the OSS Product Homepage. For more information, see *Product Overview*.**
  - **You must have created an AccessKeyID and an AccessKeySecret. Since the Alibaba Cloud account AccessKey gives you permission to access all APIs, we recommend that you follow the Alibaba Cloud Security Best Practices. If you deploy the service on the server, you can use an RAM sub-account or STS for API access or daily O&M management and control. If you deploy the service on the client, we recommend you use STS for API access. For more information, see Access control.**

- **Environment requirements**

  The OSS BrowserJS-SDK is built on the *Node.js* environment. It generates codes that apply to browsers through *Browserify* and *Babel*.

  However, because of the special natures of the browser environment, some features cannot be used:

  - **Stream upload: The chunked encoding cannot be set in browsers. The multipart upload can be used instead.**
  - **Operations on local files: You cannot directly operate on local file systems in the browser. You can download the file using a signed URL.**
  - **Bucket-related operations (listBuckets/BucketACL/BucketLogging). Since OSS does not support bucket-related cross-origin requests currently, you can perform bucket-related operations in the console.**

Usage

The OSS BrowserJS-SDK supports both synchronous and asynchronous usages.

- **Synchronous mode: Synchronize the asynchronous mode based on `async/await` in the ES7 standard.**
- **Asynchronous mode: Similar to callback. The API returns *Promise*. The `.then()` method is used to process the returned result and the `.catch()` method is used to process errors.**

`new OSS()` is used to create a client both in the synchronous mode and asynchronous mode.

Here are the examples of uploading a file and then downloading the file immediatel y in the two modes:

- **Synchronous mode**

```
// Create an OSS client instance. For detailed parameters, see the
configuration items described in the documents.
let client = new OSS(...);

async function put () {
  try {
    // object indicates the name of the uploaded object, which can
be customized.
    // file indicates the file that needs to be upload in the
browser. The HTML and Blob formats are supported.
    let r1 = await client.put('object', file);
    console.log('put success: %j', r1);
    let r2 = await client.get('object');
```

```
    console.log('get success: %j', r2);
  } catch (e) {
    console.error('error: %j', e);
  }
}

put();
```

- **Asynchronous mode**

```
// Create an OSS client instance. For detailed parameters, see the
configuration items described in the documents.
let client = new OSS(...);

// object indicates the name of the uploaded object, which can be
customized.
// file indicates the file that needs to be upload in the browser.
The HTML and Blob formats are supported.
client.put('object', file).then(function (r1) {
  console.log('put success: %j', r1);
  return client.get('object');
}).then(function (r2) {
  console.log('get success: %j', r2);
}).catch(function (err) {
  console.error('error: %j', err);
});
```

Installation

- **Usage in browsers**

  Supported browsers:

  - **IE (>=10) and Edge**
  - **Mainstream Chrome/Firefox/Safari versions**
  - **Mainstream Android/iOS/WindowsPhone versions**

  Browser.js SDK can be installed in the following methods:

  - **Browser introduction:**

    Notice:
    Some browsers do not support promise, so a promise compatibility library must be introduced. For example, promise-polyfill must be introduced for IE10 and IE11.

```
<!-- Introduce online resources -->
<script src="http://gosspublic.alicdn.com/aliyun-oss-sdk-x.x.x.min
.js"></script>
```

```
-- Introduce local resources -->
```

```
<script src="./aliyun-oss-sdk-x.x.x.min.js"></script>
```

Note:

- **x.x.x represents the version number. For more information about the specific version, see *here*.**
- **The introduction of online resources depends on the stability of the CDN server. Users are recommended to introduce resources offline, that is, through local resources or self-built.**

You can use the `OSS.Wrapper` object in the code:

```
<script type="text/javascript">
  let client = new OSS({
    region: '<oss region>',
 //Since the Alibaba Cloud account AccessKey gives you permission
to access all APIs, we recommend that you access an API using STS
by following the Alibaba Cloud Security Best Practices.
    accessKeyId: '<Your accessKeyId(STS)>',
    accessKeySecret: '<Your accessKeySecret(STS)>',
    stsToken: '<Your securityToken(STS)>',
    bucket: '<Your bucket name>'
  });
</script>
```

- **Use *npm* to install the SDK:**

```
npm install ali-oss
```

After the installation is completed, you can reference it using "import" or "require". The `require` mode is not supported by browsers. Therefore, you must use packaging tools, such as `webpack` and `browserify` in the development environment.

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<oss region>',
  //Since the Alibaba Cloud account AccessKey gives you permission
 to access all APIs, we recommend that you follow the Alibaba
Cloud Security Best Practices. If you deploy the service on the
 server, use an RAM sub-account or STS, and if you deploy the
service on the client, use STS.
  accessKeyId: '<Your accessKeyId>',
  accessKeySecret: '<Your accessKeySecret>',
  bucket: '<Your bucket name>'
```

```
});
```

- **Use *Bower*:**

  For web projects, you can install the SDK through the *Bower*:

  ```
  bower install ali-oss
  ```

  Then you can add the `<script>` tag in the webpage:

  ```
  <script src="bower_components/ali-oss/dist/aliyun-oss-sdk.min.js
  "></script>
  ```

12.2 Quick start

This topic describes how to quickly access the OSS service in the BrowserJS-SDK, including uploading/downloading files and viewing file lists.

Notice:

Since the Alibaba Cloud account AccessKey gives you permission to access all APIs, we recommend that you not use the AK that can cause configuration leak by following the Alibaba Cloud Security Best Practices. Therefore, you must create and use STS (temporary account permission) to authorize the client.

For more information about browser usage, see *Browser Application*.

For demo project, see *example*.

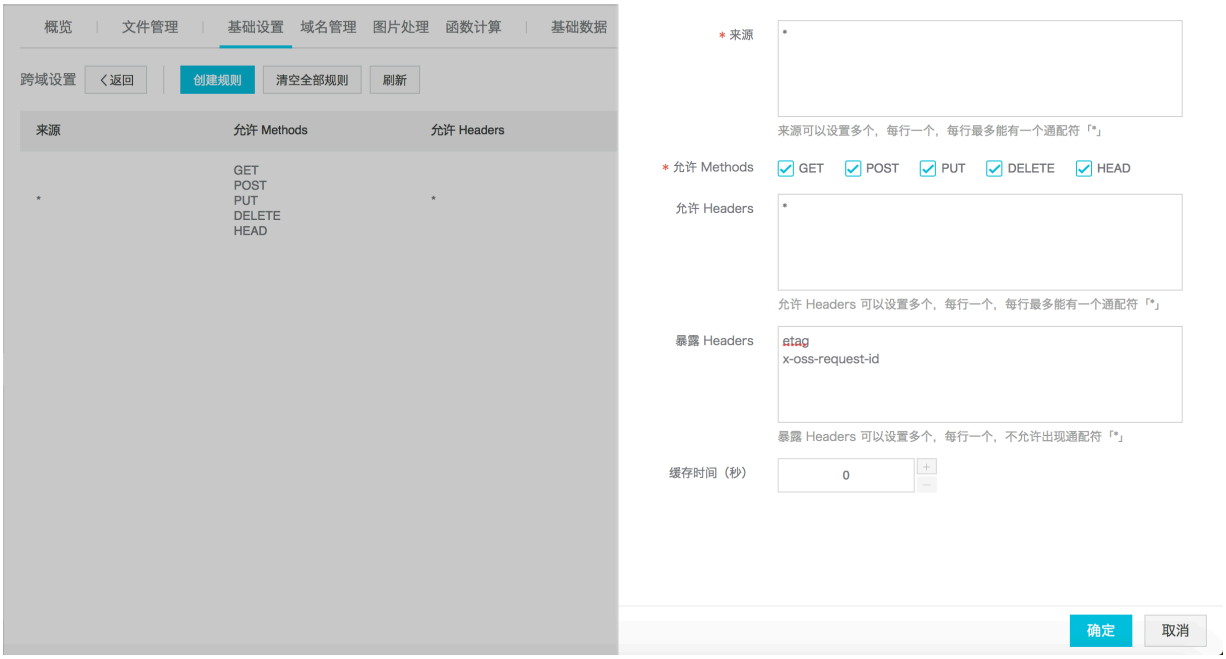
Bucket settings

The *CORS settings* of the bucket must be enabled to access OSS directly in a browser:

- **Set allowed origins to `*`**
- **Set allowed methods to `PUT, GET, POST, DELETE, and HEAD`.**
- **Set allowed headers to `*`.**
- **Set expose headers to `ETag` `x-oss-request-id`.**

Notice:

Set the CORS rule as the first option.



Use the SDK

Currently, browsers do not support direct bucket-related operations (for example, list buckets, and get/set bucket logging/referer/website/cors), but support object-related operations (for example, upload/download files, and view file lists).

- **Include the SDK**

  First, include the following tags in the `<head>` of the HTML file:

  ```
  <script src="http://gosspublic.alicdn.com/aliyun-oss-sdk-x.x.x.min.
  js"></script>
  ```

  For more introduction methods, see *Installation*.

  Create a client using `new OSS.Wapper()`. OSS.Wrapper provides an asynchronous API similar to the callback method to process the returned results in `.then()` and process errors in `.catch()`.

- **Build an STS Server and obtain temporary authorization information from the client**

  1. **Activate the STS service for your account. If the service has been activated, you can skip this step. To activate the STS service, see *STS Activation*. For more**

information on how to configure OSS authorization policy, see *RAM and STS authorization policy configuration*.

2. Apply the exclusive STS signature service through the STS SDK or API. To help you quickly get started with the STS service, we provide demo programs in multiple languages:
   - **NodeJS:** *Download address*
   - **PHP:** *Download address*
   - **Java:** *Download address*
   - **Ruby:** *Download address*

   Note:
   The demo program is only used as reference for STS feature implementation. To implement STS features in the production environment, you must develop codes required for authentication and other functions.

3. Initiate requests to the STS service built in the frontend (browser) to obtain the STS temporary authorization information.

```
<script type="text/javascript">
  //OSS. urlib is the logic of sending requests encapsulated
within the SDK. Developers can use any library that sends requests
 to send requests to 'sts-server '.
  OSS.urllib.request("http://your_sts_server/", {method: 'GET'}, (
err, response) => {
      if (err) {
     return alert(err);
      }
      try {
        result = JSON.parse(response);
      } catch (e) {
          errmsg = 'parse sts response info error: ' + e.message;
          return alert(errmsg);
      }
     // console.log(result)
   })
</script>
```

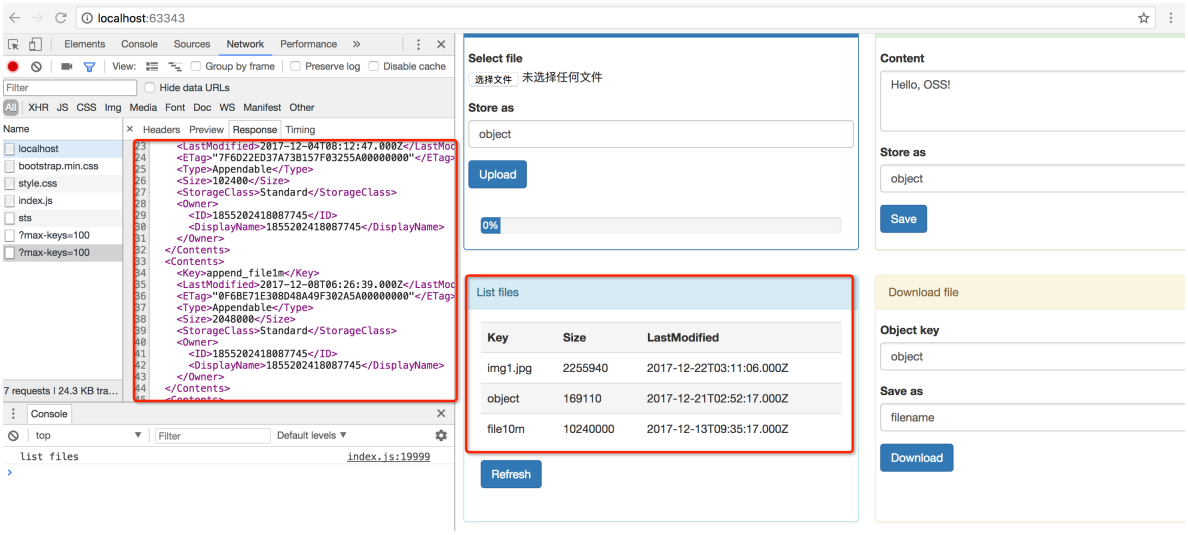
- **View the object list**

```
<script type="text/javascript">
OSS.urllib.request("http://your_sts_server/", {method: 'GET'}, (err
, response) => {
  if (err) {
 return alert(err);
  }

  try {
    result = JSON.parse(response);
  } catch (e) {
    return alert('parse sts response info error: ' + e.message);
```

```
  }

  let client = new OSS({
    accessKeyId: result.AccessKeyId,
 accessKeySecret: result.AccessKeySecret,
 stsToken: result.SecurityToken,
 endpoint: '<oss endpoint>',
 bucket: '<Your bucket name>'
  });

  client.list({
    'max-keys': 10
  }).then(function (result) {
    console.log(result);
  }).catch(function (err) {
    console.log(err);
  });
});
</script>
```

The parameter `region` refers to the region used when you applied for OSS, such as `oss-cn-hangzhou`. The completeregion list can be viewed in *OSS Nodes*.

Open the HTML file in the browser, and open the Chrome Developer Console to view the result in List files.



Notice:

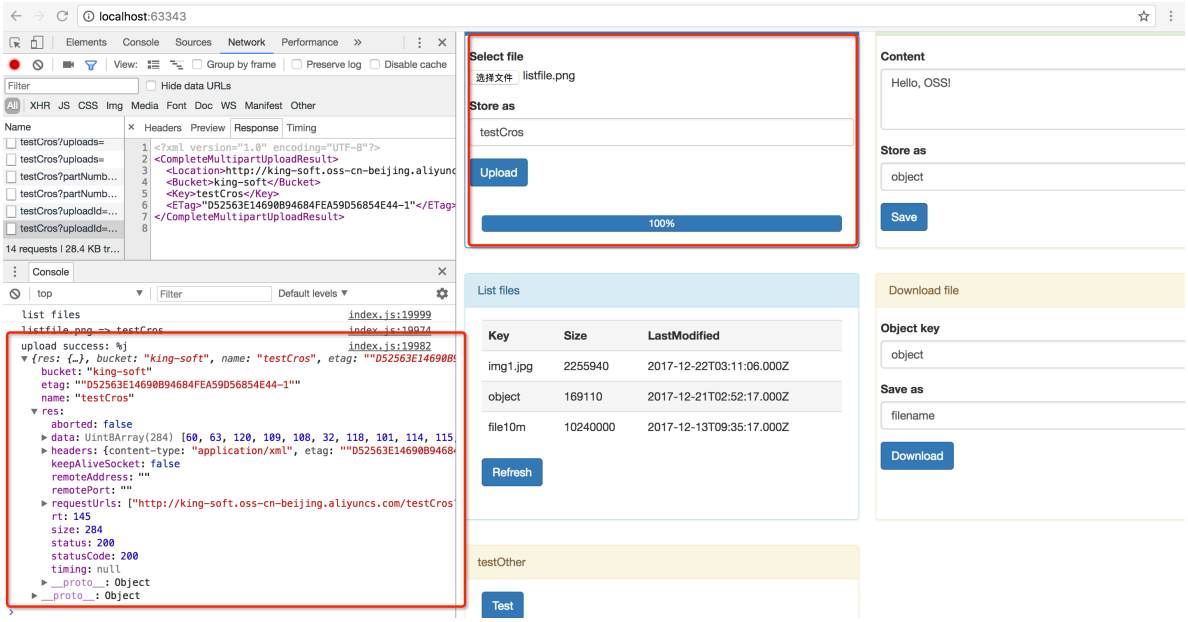
A browser is an untrusted environment. An extremely high risk occurs if you store the AccessKeyID and AccessKeySecret in a browser. We recommend that you only use plaintext settings for testing purposes. `STS authentication mode` or `self-signed mode` is recommended for business applications. For more information, see *Resource Access Management* and *Direct Transfer on a Mobile Device*.

- **Upload an object**

  We use the `multipartUpload` API to upload objects. For more information about `multipartUpload`, see *Multipart upload*.

```
<body>
  <input type="file" id="file" />
  <script type="text/javascript">
    document.getElementById('file').addEventListener('change',
function (e) {
      let file = e.target.files[0];
      let storeAs = 'upload-file';
      console.log(file.name + ' => ' + storeAs);
   //OSS. urlib is the logic of sending requests encapsulated within
 the SDK. Developers can use any library that sends requests to send
 requests to 'sts-server '.
   OSS.urllib.request("http://your_sts_server/", {method: 'GET'}, (
err, response) => {
    if (err) {
   return alert(err);
    }
    try {
   result = JSON.parse(response);
    } catch (e) {
   return alert('parse sts response info error: ' + e.message);
    }
    let client = new OSS({
   accessKeyId: result.AccessKeyId,
   accessKeySecret: result.AccessKeySecret,
   stsToken: result.SecurityToken,
   endpoint: '<oss endpoint>',
   bucket: '<Your bucket name>'
    });
          //storeAs indicates the uploaded object name and file
indicates the uploaded file.
    client.multipartUpload(storeAs, file).then(function (result) {
   console.log(result);
    }).catch(function (err) {
   console.log(err);
    });
  });
 });
  </script>
```

```
</body>
```

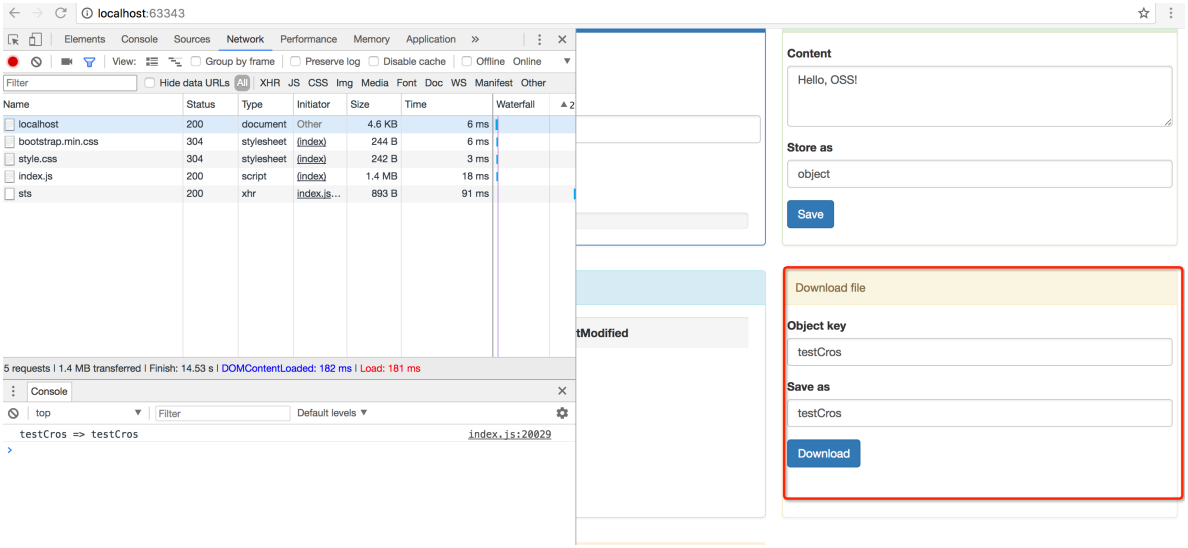


- **Download objects**

  You cannot directly operate on files in the browser. Therefore, the download link of a file is generated using the signed URL method, andyou can download the file by clicking the link.

```
<body>
  <input type="button" id="download" value="Download" />
  <script type="text/javascript">
    document.getElementById('download').addEventListener('click',
function (e) {
      let objectKey = 'test/download_file';
      let saveAs = 'download_file';
      console.log(objectKey + ' => ' + saveAs);
   //OSS. urlib is the logic of sending requests encapsulated within
 the SDK. Developers can use any library that sends requests to send
 requests to 'sts-server '.
   OSS.urllib.request("http://your_sts_server/", {method: 'GET'}, (
err, response) => {
    if (err) {
   return alert(err);
    }
    try {
   result = JSON.parse(response);
    } catch (e) {
   return alert('parse sts response info error: ' + e.message);
    }
    let client = new OSS({
   accessKeyId: result.AccessKeyId,
   accessKeySecret: result.AccessKeySecret,
   stsToken: result.SecurityToken,
   endpoint: '<oss endpoint>',
   bucket: '<Your bucket name>'
    });
    let result = client.signatureUrl(objectKey, {
   expires: 3600,
   response: {
```

```
      'content-disposition': 'attachment; filename="' + saveAs + '"'
    }
     });
     console.log(result);
     window.location = result;
   });
     });
   </script>
</body>
```



12.3 Options

This topic describs how to configure options.

OSS(options) introduction

- **[accessKeyId] {String}: AccessKey you create on Alibaba Cloud console website**
- **[accessKeySecret] {String}: access secret you create**
- **[stsToken] {String}: use temporary authorization method. For more information, see *Use STS to access OSS*.**
- **[bucket] {String}: the bucket you create on the console.**
- **[endpoint] {String}: oss region domain.**
- **[region] {String}: the bucket data region location. The default region is oss-cn-hangzhou.**
- **[internal] {Boolean}: access OSS with Alibaba Cloud internal network or not, default is false. If your servers are running on Alibaba Cloud too, you can set true to save lot of money.**
- **[secure] {Boolean} instruct OSS client to use HTTPS (secure: true) or HTTP (secure: false) protocol. For more information, click *here*.**

- **[timeout] {String|Number} instance level time-out for all operations. The default value is 60s.**

Example:

```
// Assume that an OSS object has been introduced to the browser
environment in the 'script' or 'npm' method.
let store = new OSS({
  accessKeyId: 'your access key',
  accessKeySecret: 'your access secret',
  bucket: 'your bucket name',
  region: 'oss-cn-hangzhou'
});
```

12.4 Upload objects

You can upload a file to OSS using any of the following methods:

- **Upload Blob data**
- **Resumable upload**

Upload Blob Data

Binary Large Object (Blob) indicates a large object of the binary type. The concept of blob is used in some databases. For example, the BLOB type used in MySQL indicates a container for binary data.

You can also use the `put` interface to upload content in a Blob to OSS:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function putBlob () {
  try {
    let result = await client.put('object-key', new Blob(['content'],
{ type: 'text/plain' }));
    console.log(result);
  } catch (e) {
    conosle.log(e);
  }
}
putBlob();
```

Resumable upload

When the file to be uploaded is large, you can use the multipartUpload interface for multipart upload. Multipartupload is to divide a large request into multiple

small requests for execution. As a result, when some of the requests fail,you do not need to upload the entire file again, but only to upload the failed parts. Generally for a file larger than 100 MB, we recommend that you use the preceding multipart upload approach and create a new OSS instance before each multipart upload.

If a `ConnectionTimeoutError` error occurs when you use the multipartUpload interface, you must construct your own method to handle time-out issues. You can reduce the part size, increase the time-out period and allowed retry times, or catch the `ConnectionTimeoutError` error and return it to the user. For more information, see *network error handling*.

For the usage of multipartUpload API, see *MultipartUpload*.

Related parameters:

- **name {String}: Object name**
- **file {File}: HTML5 Web File or Blob data**

- **[options] {Object}: Additional parameters**
  - **[checkpoint] {Object}: Endpoint checkpoint used in resumable upload. If this parameter is set, the upload starts from the endpoint. If it is not set, the upload restarts.**
    - **file {File}: File object selected by the user. Users must manually set this parameter if the browser is restarted.**
    - **name {String}: Uploaded object key**
    - **fileSize {Number}: File size**
    - **partSize {Number}: Part size**
    - **uploadId {String}: Upload Id**
    - **doneParts {Array}: Array of uploaded parts, including the following objects:**
      - **number {Number}: Part number**
      - **etag {String}: Part etag**
  - **[parallel] {Number}: Number of parts uploaded simultaneously**
  - **[partSize] {Number}: Part size**
  - **[progress] {Function}: A generator function or a thunk. The callback function contains the following three parameters: `function`, `async`, and `promise`.**
    - **(percentage {Number}: Percetage of upload progress (a decimal range from 0 to 1)**
    - **checkpoint {Object}: Endpoint checkpoint**
    - **res {Object}): Response returned after a single part is successfully uploaded**
  - **[meta] {Object}: Header meta inforamtion defined by users with a prefix `x-oss-meta-`**
  - **[mime] {String}: Custom `Content-Type header`**
  - **[headers] {Object}: Extra headers. See *RFC 2616* for more information.**
    - **‘Cache-Control’: General header used to implement cache mechanisms by specifying a command in HTTP requests and responses. For example: `Cache-Control: public, no-cache`**
    - **‘Content-Disposition’: Used to indicate the disposition form of a response, which can be an internal reference (a part of a webpage or a page)**

or an attachment downloaded and saved locally. For example: `Content-Disposition: somename`

- **‘Content-Encoding’ : Used to compress data of specific media type. For example:** `Content-Encoding: gzip`
- **‘Expires’ : Expiration time. For example:** `Expires: 3600000`

- **[callback] {Object}: Callback settings For more information, see *Callback*.**
  - **url {String}: The address of the callback server communicated with the OSS server. It corresponds to callbackUrl in the CallBack parameter. Required**
  - **body {String}: The value of callback request. It is in the JSON format and corresponds to callbackBody in the Callback parameter. Required**
  - **[host] {String}: The value of the Host header in the callback request. It corresponds to callbackHost inthe CallBack parameter.**
  - **[contentType] {String}: The Content-Type of the callback request. It corresponds to callbackBodyType in the CallBack parameter.**
  - **[customValue] {Object}: Custom parameters in the callback request. It corresponds to callback-var in the CallBack parameter.**

Example:

1. **The upload progress is recorded. When the upload is initiated again, the recorded checkpoint parameter is passed in.**
2. **We recommend that you create a new OSS instance for each multipart upload task.**

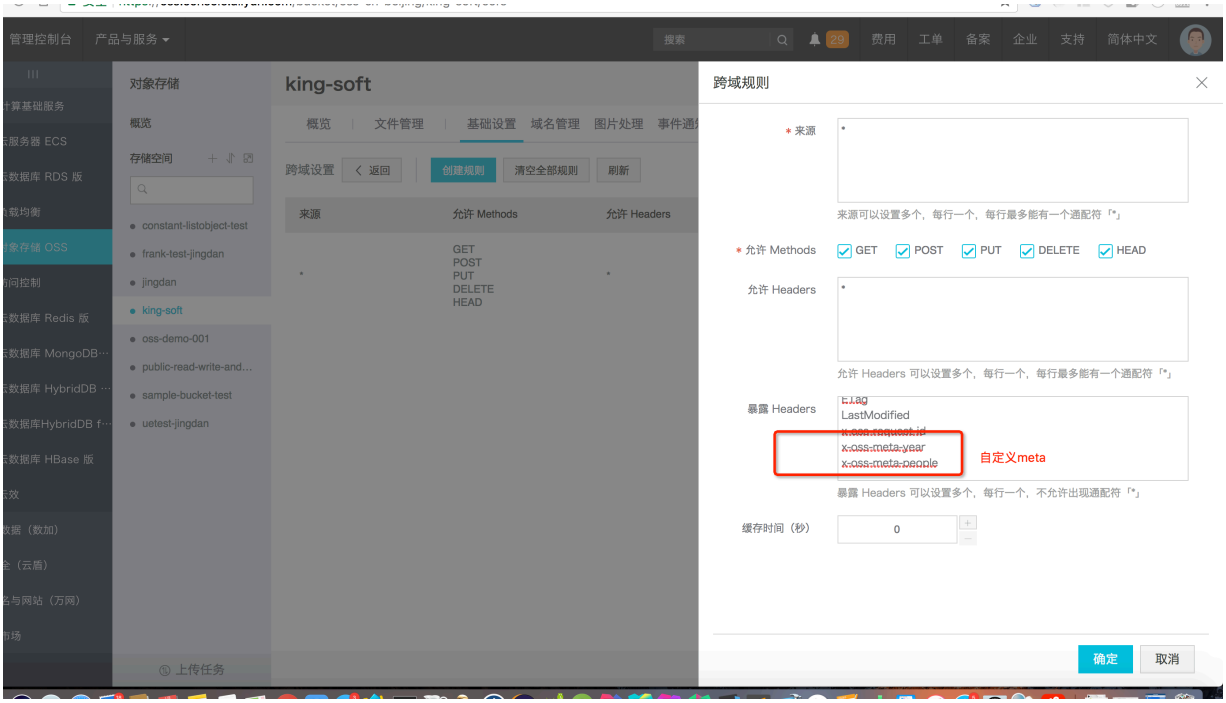
```
let OSS = require('ali-oss')

let ossConfig = {
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
}

let client = new OSS(ossConfig);

let tempCheckpoint;

// Define the upload method
async function multipartUpload () {
  try {
    let result = await client.multipartUpload('object-key', 'local-
file', {
      progress: async function (p, checkpoint) {
        // Record the checkpoint. If you close the browser and
continue the upload after restarting the browser, the upload cannot be
 continued. For more information, see the description of file objects.
```

```
        tempCheckpoint = checkpoint;
      }
      meta: { year: 2017, people: 'test' },
      mime: 'image/jpeg'
   })
  } catch(e){
    console.log(e);
  }
}

// Start uploading
multipartUpload();

// Pause multipart upload
client.cancel();

// Resume multipart upload
let resumeclient = new OSS(ossConfig);
async function resumeUpload () {
  try {
    let result = await resumeclient.multipartUpload('object-key', '
local-file', {
 progress: async function (p, checkpoint) {
          tempCheckpoint = checkpoint;
        },
        checkpoint: tempCheckpoint
        meta: { year: 2017, people: 'test' },
        mime: 'image/jpeg'
  })
  } catch (e) {
    console.log(e);
  }
}

resumeUpload();
```

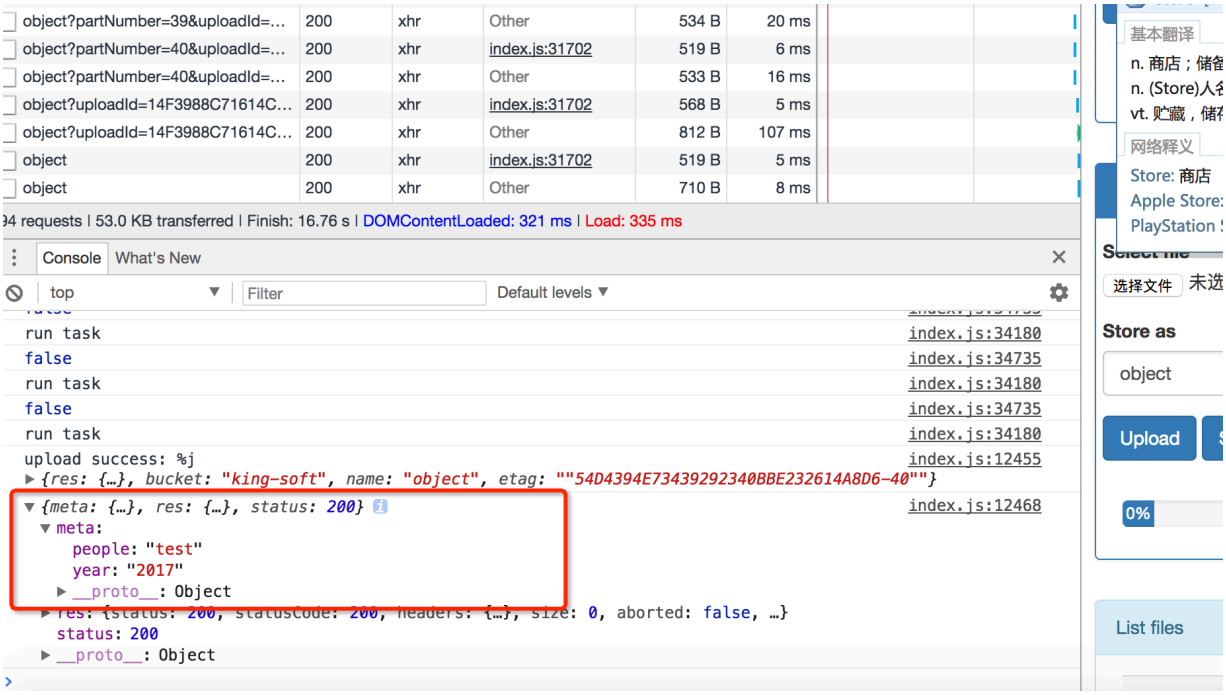
The preceding `progress` parameter is a progress callback function used to get the upload progress.

```
const progress = async function progress(p, checkpoint) {
  console.log(p)
};
```

The preceding `meta` parameter is a user-defined metadata. You can obtain the meta value of the object using the head API, but the meta header must be set properly in the Exposed Headers of the Cross-region Settings in the console, as shown in the following figure:



Returned results of successful requests are as follows:



You can add `callback` in the option parameter to send callback information.

```
callback: {
  url: 'http://oss-demo.aliyuncs.com:23450',
  host: 'oss-cn-hangzhou.aliyuncs.com',
  /* eslint no-template-curly-in-string: [0] */
  body: 'bucket=${bucket}&object=${object}&var1=${x:var1}',
  contentType: 'application/x-www-form-urlencoded',
  customValue: {
    var1: 'value1',
    var2: 'value2',
  },
```

```
},
```

12.5 Download objects

You can download an object from OSS using any of the following methods:

HTTP download (using a browser)

You can use the `signatureUrl` method in the browser to generate the downloadable HTTP address.The validity period of the URL is half an hour by default:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

let url = client.signatureUrl('object-key');
console.log(url);

let url = client.signatureUrl('object-key', {expires: 3600});
console.log(url);
```

12.6 Manage objects

A bucket may have a lot of objects. SDK provides a series of interfaces to help in managing the objects easily.

View all objects

The following code uses `list` to enumerate all objects in the current bucket. Main parameters are described as follows:

- **Prefix: Specifies only list objects with specific prefixes.**
- **Marker: Specifies list objects whose names are greater than a specified marker.**
- **Delimiter: Obtains the common prefix of a group of objects.**
- **Max-keys: Specifies the maximum number of objects returned.**

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});
```

```
async function list (dir) {
  try {
    // Without any parameters. A maximum of 1,000 objects can be
returned by default.
 let result = await client.list();
 console.log(result);

    // Continue to list objects based on nextMarker
    if (result.isTruncated) {
      let result = await client.list({ marker : result.nextMarker });
 }

    // List all objects whose names contain the "my-" prefix
    let result = await client.list({
      prefix: 'my-'
    });
    console.log(result);

    // List all objects whose names are greater than "my-object"
marker and contain the "my-" prefix
    let result = await client.list({
      prefix: 'my-',
      marker: 'my-object'
    });
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

list();
```

Simulate the directory structure

OSS is an object-based storage service. It has no directories. All objects stored in a bucket are uniquely identified by the key of the object. These objects does not follow any hierarchical structure amongst themselves. Such structure allows OSS to store objects efficiently. However, you may want to put different types of objects under different "directories" to manage objects in a conventional way. OSS provides a "Public Prefix" feature that helps to create a stimulated directory structure. For more information about the public prefix concept, see *View the object list*.

Assume that the bucket already contains the following objects:

```
foo/x
foo/y
foo/bar/a
foo/bar/b
foo/hello/C/1
foo/hello/C/2
...
```

```
foo/hello/C/9999
```

Next we implement a function called `listDir` to enlist all the objects and subdirectories under the specified directory:

```
let OSS = require('ali-oss');

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function listDir (dir) {
   try {
     let result = await client.list({
    prefix: dir,
    delimiter: '/'
   });

   result.prefixes.forEach(function (subDir) {
      console.log('SubDir: %s', subDir);
   });
   result.objects.forEach(function (obj) {
      console.log(Object: %s', obj.name);
   });
  } catch (e) {
    console.log(e);
  }
}
```

The execution results are as follows:

```
> listDir('foo/')
=> SubDir: foo/bar/
   SubDir: foo/hello/
   Object: foo/x
   Object: foo/y

> listDir('foo/bar/')
=> Object: foo/bar/a
   Object: foo/bar/b

> listDir('foo/hello/C/')
=> Object: foo/hello/C/1
   Object: foo/hello/C/2
   ...
   Object: foo/hello/C/9999
```

Object Meta

When uploading an object to OSS, except for the object content, you have to specify certain attribute information for the object. This attribute information is called meta information. The metadata is stored together with the object during uploading.

This is because the object meta information is stored in HTTP headers during object uploading/downloading, the meta information cannot contain any complex characters according to HTTP.

Note:
Therefore, the meta information must only contain simple printable ASCII characters and cannot contain line breaks. The maximum size of all meta information cannot exceed 8KB.

You can specify the object meta information by specifying the `meta` parameter when using the `put`, putStream and `multipartUpload`:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function multipartUploadWithMeta () {
  try {
    let result = await client.multipartUpload('object-key', 'local-
file', {
 meta: {
    year: 2017,
    people: 'mary'
     }
   });
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

multipartUploadWithMeta();
```

Delete an object

You can use `delete` to delete the object:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function delete () {
  try {
    let result = await client.delete('object-key');
 console.log(result);
```

```
  } catch (e) {
    console.log(e);
  }
}

delete();
```

Delete multiple objects

The following code uses `deleteMulti` to delete multiple objects. You can set the `quiet` parameter to determine whether to return the deletion results:

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

async function deleteMulti () {
  try {
    let result = await client.deleteMulti(['obj-1', 'obj-2', 'obj-3
']);
    console.log(result);

    let  result = await client.deleteMulti(['obj-1', 'obj-2', 'obj-3
'],{
    quiet: true
  });
    console.log(result);
  } catch (e) {
    console.log(e);
  }
}

deleteMulti();
```

12.7 CNAME

This topic describes how to bind a custom domain name.

OSS supports binding custom domain names to OSS to seamlessly migrate data storage to the OSS. For example, your domain name is my-domain.com, and all your previous image resources are in a format similar to `http://img.my-domain.com/x.jpg`. After you migrate the image storage to OSS, you can still access the images with the original address by binding your custom domain name to OSS:

- **Activate OSS and create a bucket.**

- Modify the DNS configuration, add a CNAME record, and direct img.my-domain.com to an endpoint of OSS (for example, my-bucket.oss-cn-hangzhou.aliyuncs.com).
- On *OSS console*, bind img.my-domain.com to the created bucket.
- Upload images to the created bucket in OSS.

Follow the preceding steps to use the original address `http://img.my-domain.com/xx.jpg` to access images on OSS. For more information, see *Bind custom domain names (CNAME)*.

SDK supports the use of a CNAME as the endpoint. To do this, set the `:cname` parameter to true. For example:

```
let OSS = require('ali-oss')

let client = new OSS({
  endpoint: '<Your endpoint>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  cname: true
});
```

Notice:

The list_buckets interface is unavailable when CNAME is used because the custom domain name is bound already to a specific bucket.

12.8 Authorized access

This topic describes how to authorize access to OSS.

Use STS for temporary access authorization

You can use Alibaba Cloud Security Token Service (STS) to authorize temporary access to OSS. STS is a Web service that provides a temporary access token to a cloud computing user. You can use STS to grant a third-party application or a RAM user (whose user ID is managed by yourself) an access credential with customized validity period and permissions. For more information about STS, see *STS introduction*.

STS has the following advantages:

- **You can generate an access token and send it to the third-party application for authorization without exposing your long-term key (AccessKey). You can customize the access permissions and validity period of the token.**

- **The access token automatically becomes invalid after expiration.**

For more information about how to use STS to access OSS, see *Access control* in OSS Developer Guide.

Run the following code to create a signed request by using STS:

```
const express = require('express');
const { STS } = require('ali-oss');
const fs = require('fs');

const app = express();
const path = require('path');
const conf = require('./config');

app.get('/sts', (req, res) => {
  console.log(conf);
  let policy;
  if (conf.PolicyFile) {
    policy = fs.readFileSync(path.resolve(__dirname, conf.PolicyFile
)).toString('utf-8');
  }

  const client = new STS({
    accessKeyId: conf.AccessKeyId,
    accessKeySecret: conf.AccessKeySecret
  });

  client.assumeRole(conf.RoleArn, policy, conf.TokenExpireTime).then((
result) => {
    console.log(result);
    res.set('Access-Control-Allow-Origin', '*');
    res.set('Access-Control-Allow-METHOD', 'GET');
    res.json({
      AccessKeyId: result.credentials.AccessKeyId,
      AccessKeySecret: result.credentials.AccessKeySecret,
      SecurityToken: result.credentials.SecurityToken,
      Expiration: result.credentials.Expiration
    });
  }).catch((err) => {
    console.log(err);
    res.status(400).json(err.message);
  });
});

app.use('/static', express.static('public'));
app.get('/', (req, res) => {
  res.sendFile(path.resolve(__dirname, '../index.html'));
});

app.listen(9000, () => {
  console.log('App started.');
```

```
});
```

Sign a URL to authorize temporary access

- **Sign a URL**

  You can provide a signed URL to a visitor for temporary access. When you sign a URL, you can specify the expiration time of the URL to restrict the period of access from visitors.

- **Sign a URL for an object**

  Note:

  `name {String}` **indicates the name of the target object.** `[expires] {Number}` **indicates the expiration period of the URL, which is 1,800 seconds by default. For more information about other parameters, see** *Github*.

  Run the following code to sign a URL for an object:

```
const url = store.signatureUrl('ossdemo.txt');
console.log(url);
// --------------------------------------------------
const url = store.signatureUrl('ossdemo.txt', {
  expires: 3600,
  method: 'PUT'
});
console.log(url);

//  put object with signatureUrl
// --------------------------------------------------

const url = store.signatureUrl('ossdemo.txt', {
  expires: 3600,
  method: 'PUT',
  'Content-Type': 'text/plain; charset=UTF-8',
});
console.log(url);

// --------------------------------------------------
const url = store.signatureUrl('ossdemo.txt', {
  expires: 3600,
  response: {
    'content-type': 'text/custom',
    'content-disposition': 'attachment'
  }
});
console.log(url);

// put operation
```

- **Run the following code to sign a URL for image processing:**

```
const url = store.signatureUrl('ossdemo.png', {
  process: 'image/resize,w_200'
});
console.log(url);
```

```
// ------------------------------------------------
const url = store.signatureUrl('ossdemo.png', {
  expires: 3600,
  process: 'image/resize,w_200'
});
console.log(url);
```

12.9 Set ACLs

OSS allows you to set access permissions for objects. This helps to easily control external access to your resources.

You can set four ACLs for an object:

- **default: The object inherits the ACL for the bucket it belongs to, that is, the ACL for the object is the same as that for the bucket where the object is stored.**
- **public-read-write: Anonymous users are allowed to read/write the object.**
- **public-read: Anonymous users are allowed to read the object.**
- **private: Anonymous users are not allowed to access objects in the bucket. Signature is required for all accesses.**

When an object is created, the default permission applies by default. You can use `putACL` to set the ACL for the object.

```
let OSS = require('ali-oss')

let client = new OSS({
  region: '<Your region>'
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: '<Your bucket name>'
});

async function getACL () {
  try {
    let result = await client.getACL('my-object');
    console.log(result.acl); // default

    await client.putACL('my-object', 'public-read');
    let result = await client.getACL('my-object');
    console.log(result.acl); // public-read
  } catch (e) {
    console.log(e);
  }
}

getACL();
```

Note:

1. **If an object is configured with an ACL policy (not default), the object ACL takes priority during permission authentication when it is accessed. The bucket ACL is ignored.**
2. **If anonymous access is allowed (public-read or public-read-write is configured for the object), you can directly access the object using a browser.**

```
 For example: http://bucket-name.oss-cn-hangzhou.aliyuncs.com/
object.jpg.
```

For more information about ACL, see *Access Control*.

12.10 Image processing

OSS Image Service (IMG) is an image processing service that features massive capacity, high security, low costs, and high reliability. By uploading and storing original images in OSS, you can process images anytime, anywhere, on any Internet device through a simple RESTful API.IMG offers image processing APIs.

For more information, see *Image processing*.

Basic features of Image Service

OSS IMG provides the following features:

- *Obtain image information*
- *Image format conversion*
- *Resize images*
- *Incircle*
- *Adaptive orientation*
- *Brightness*
- *Add watermarks*: **allows you to add images, texts, or image-text watermarks to another image.**
- *Customize image processing styles*
- **Calling multiple image processing features through** *Cascading Processing*

Usage of Image Service

Image Service uses the standard HTTP GET request for access. All processing parameters are encoded in the QueyString in the URL.

If the ACL for an image is private read and write, only authorized users are allowed can access the image.

- Anonymous access

  If the access permission of the image object is `public read`, as shown in the following table, you can access Image Service anonymously.

Bucket ACL	Object ACL
Public-read or Public-read-write	Default
Any ACL	Public-read or Public-read-write

Access Image Service anonymously using the third-level domain in the following format:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-
process=image/<yourAction>,<yourParamValue>
```

Parameter	Description
bucket	Name of the user’s bucket
endpoint	The access domain name for the data center of the user’s bucket
object	The image object uploaded by the user to the OSS
image	The identifier reserved by Image Service
action	Operations by the user on images, such as scaling, cropping, and rotating

Parameter	Description
param	Parameters corresponding to the operations by the user on images

- Basic operations

  For example, scale an image to a width of 100 px and adjust the height based on the ratio.

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=image/resize,w_100
```

- Customize image styles

  You can use a level-3 domain name in the following format to access a processed image:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-process=style/<yourStyleName>
```

  - style: The identifier reserved by the system for the user’s custom style
  - yourStyleName: specifies the name of a custom image style. It is the name specified in the rule that is created on the OSS console.

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=style/oss-pic-style-w-100
```

- Cascade operations

  With the cascading processing, you can implement multiple operations on an image. The format is as follows:

```
http://<yourBucketName>.<yourEndpoint>/<yourObjectName>?x-oss-process=image/<yourAction1>,<yourParamValue1>/<yourAction2>,<yourParamValue2>/...
```

  For example:

```
http://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-process=image/resize,w_100/rotate,90
```

- Image Service also supports HTTPS access.

For example,

```
https://image-demo.oss-cn-hangzhou.aliyuncs.com/example.jpg?x-oss-
process=image/resize,w_100
```

- **Authorized access**

  For objects with private permissions, as shown in the following table, you must be authorized to access Image Service.

Bucket ACL	Object ACL
private	default
Any ACL	private

The Image Service URL code generated with signature is as follows:

```
let OSS = require('ali-oss');
let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for exmaple, http://oss-cn-hangzhou.aliyuncs
.com>'
});
// Expire after 10 minutes. The image processing style is: "image/
resize,w_300".
let signUrl = client.signatureUrl('example.jpg', {expires: 600, '
process' : 'image/resize,w_300'});
console.log("signUrl="+signUrl);
```

Note:

- **Authorized access supports** `custom styles`, `HTTPS`, **and** `cascading processing`.
- **Expiration time (expires) is measured in seconds.**

- **SDK access**

  You can directly use `SDK` **to access and process image objects with any ACLs.**

Note:

The image object processing using SDKs supports `custom styles`, `HTTPS`, and `cascading processing`.

- **Basic operations**

  Run the following code to perform basic operations on an image:

```
let OSS = require('ali-oss');

let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.
aliyuncs.com>'
});

// Scale the image.
async function scale () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/resize,m_fixed,w_100,h_100'})
  } catch (e) {
    console.log(e);
  }
}

scale();

// Crop the image.
async function crop () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/crop,w_100,h_100,x_100,y_100,r_1'})
  } catch (e) {
    console.log(e);
  }
}

crop();

// Rotate the image.
async function rotate() {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/rotate,90'})
  } catch (e) {
    console.log(e);
  }
}

rotate();

// Sharpen the image.
async function sharpen () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/sharpen,100'})
  } catch (e) {
    console.log(e);
  }
```

```
}

sharpen();

// Add watermarks.
async function watermark () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/watermark,text_SGVsbG8g5Zu-54mH5pyN5YqhIQ
'})
  } catch (e) {
    console.log(e);
  }
}

watermark();

// Convert the image format.
async function format () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/format,png'})
  } catch (e) {
    console.log(e);
  }
}

format();

// Obtain image information.
async function info () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/info})
  } catch (e) {
    console.log(e)
  }
}

info();
```

- **Customize image styles**

  You can run the following code to customize an image style:

```
let OSS = require('ali-oss');

let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.
aliyuncs.com>'
});

// Custom styles
async function style () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'style/example-sytle'});
  } catch (e) {
    console.log(e);
```

```
  }
}

style();
```

Note:
You must first log on to the OSS console to add the custom style 'example-style'.

- **Cascade operations**

  Run the following code to perform cascade operations on an image:

```
let OSS = require('ali-oss');

let client = new OSS({
  accessKeyId: '<accessKeyId>',
  accessKeySecret: '<accessKeySecret>',
  bucket: '<bucketName>',
  endpoint: '<endpoint, for example, http://oss-cn-hangzhou.
aliyuncs.com>'
});

// Cascading processing
async function resize () {
  try {
    let result = await client.signatureUrl('example.jpg', {expires
: 3600, process: 'image/resize,m_fixed,w_100,h_100/rotate,90'})
  } catch (e) {
    console.log(e);
  }
}

resize();
```

12.11 Browser

This topic describes the application of browsers.

Supported browser

- **IE (>=10) and Edge**
- **Mainstream Chrome/Firefox/Safari versions**
- **Mainstream Android/iOS/WindowsPhone versions**

Notice:
For IE, you must introduce the promise library before introducing the SDK:

Support for browsers of earlier versions

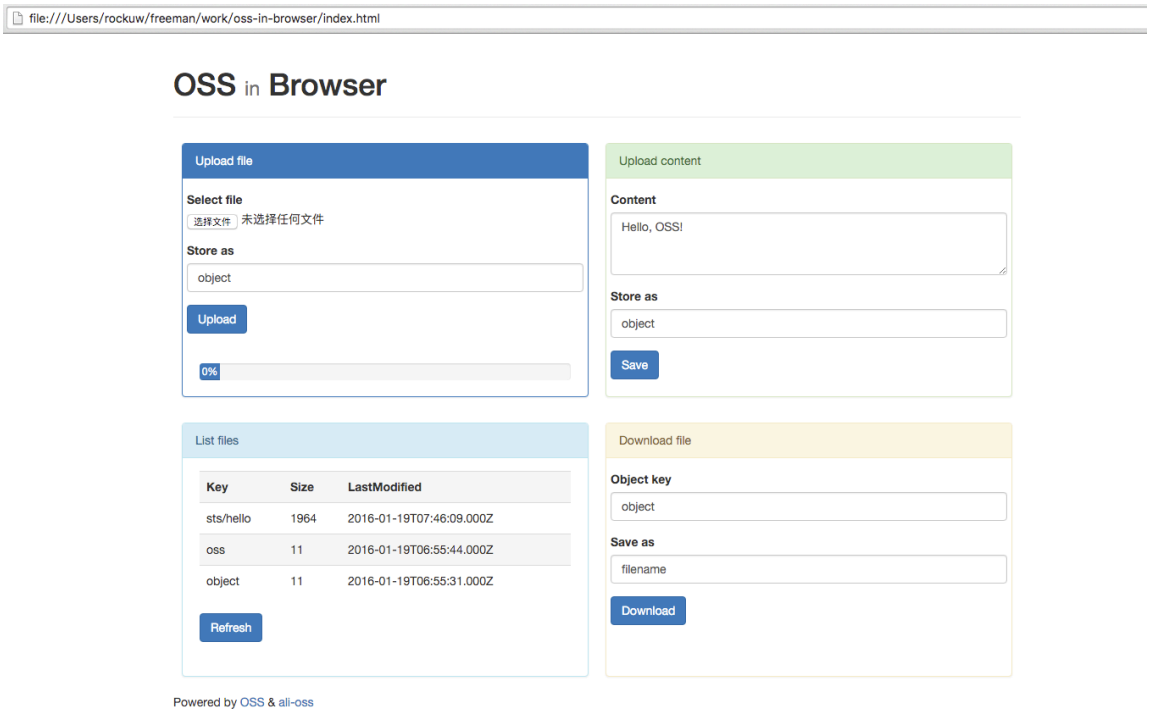
The OSS SDK perform operations through File APIs so errors may occur in browsers of earlier versions. You can use third-party plug-ins to implement OSS APIs to perform operations, such as upload and download.

- **Settings**
  - **Bucket settings**

    To access OSS through browser, you must set the *CORS* of the bucket as follows:

    - **Set `Source` to `*`.**
    - **Set `Method` to `GET, POST, PUT, DELETE, HEAD`.**
    - **Set `Allowed Headers` to `*`.**
    - **Set `Expose Header` to `ETag`.**

    Notice:
    Set the preceding CORS rule to the first option.



  - **STS settings**

    To prevent exposing the AccessKeyId and AccessKeySecret in the webpage, Alibaba Cloud adopts STS for temporary authorization. The authorization server is maintained by users. Only authenticated users are allowed to

apply for a temporary permission from the authorization server. For more information on how to set RAM users and STS policies, see Overview. For information about how to establish your own authorization servers, see *Node.js STS AppServer*.

- For example:

  Here, we use SDK to develop a webpage app which offers four features:

  - Upload objects
  - Upload text
  - List objects
  - Download objects

  For complete code, see *oss-in-browser*. The expected result is as follows:

  Let's take upload objects as an example in the following text:

  1. Create a webpage

     - An object choice box to select the object to be uploaded
     - A textbox to input the name of the object to be uploaded to the OSS.
     - A button to start the upload.
     - A progress bar to display the upload progress

     The following code creates four controls. `class="xxx"` serves to apply the *Bootstrap* style and can be ignored here.

```
<div class="form-group">
  <label>Select file</label>
  <input type="file" id="file" />
</div>
<div class="form-group">
  <label>Store as</label>
  <input type="text" class="form-control" id="object-key-file"
value="object" />
</div>
<div class="form-group">
  <input type="button" class="btn btn-primary" id="file-button"
value="Upload" />
</div>
<div class="form-group">
  <div class="progress">
    <div id="progress-bar"
         class="progress-bar"
         role="progressbar"
         aria-valuenow="0"
         aria-valuemin="0"
         aria-valuemax="100" style="min-width: 2em;">
      0%
    </div>
```

```
  </div>
</div>
```

2. **Add a style**

   You can add some styles to the webpage to enhance its look. Here, we use *Bootstrap*. Add the style in the `<head>` tag as follows:

```
<head>
  <title>OSS in Browser</title>
  <link rel="stylesheet" href="bootstrap.min.css" />
</head>
```

   `bootstrap.min.css` can be downloaded from the official website of Bootstrap.

3. **Add a script**

   Before adding a script, the webpage is static and does not respond to any click . We can add some JavaScript so that it can upload/download/list objects.

   First, introduce the SDK in the `<head>` tag:

```
<head>
  <title>OSS in Browser</title>
  <link rel="stylesheet" href="bootstrap.min.css" />
  <script type="text/javascript" src="http://gosspublic.alicdn.com
/aliyun-oss-sdk-x.x.x.min.js"></script>
  <script type="text/javascript" src="app.js"></script>
</head>
```

   `app.js` contains the code that implements the object uploading. The content is as follows:

```
const appServer = 'http://localhost:3000';
const bucket = 'js-sdk-bucket-sts';
const region = 'oss-cn-hangzhou';

const urllib = OSS.urllib;

const applyTokenDo = function (func) {
  const url = appServer;
  return urllib.request(url, {
    method: 'GET'
  }).then(function (result) {
    const creds = JSON.parse(result.data);
    const client = new OSS({
      region: region,
      accessKeyId: creds.AccessKeyId,
      accessKeySecret: creds.AccessKeySecret,
      stsToken: creds.SecurityToken,
      bucket: bucket
    });

    return func(client);
```

```
  });
};

let currentCheckpoint;
const progress = async function progress(p, checkpoint) {
  currentCheckpoint = checkpoint;
  const bar = document.getElementById('progress-bar');
  bar.style.width = `${Math.floor(p * 100)}%`;
  bar.innerHTML = `${Math.floor(p * 100)}%`;
};

let uploadFileClient;
const uploadFile = function (client) {
  if (! uploadFileClient || Object.keys(uploadFileClient).length
 === 0) {
    uploadFileClient = client;
  }

  const file = document.getElementById('file').files[0];
  const key = document.getElementById('object-key-file').value.
trim() || 'object';
  console.log(`${file.name} => ${key}`);

  const options = {
    progress,
    partSize: 100 * 1024,
    meta: {
      year: 2017,
      people: 'test',
    },
  };

  return client.multipartUpload(key, file, options).then( (res) =>
 {
    console.log('upload success: %j', res);
    currentCheckpoint = null;
    uploadFileClient = null;
  }).catch((err) => {
    if (uploadFileClient && uploadFileClient.isCancel()) {
      console.log('stop-upload!') ;
    } else {
      console.error(err);
    }
  });
};

window.onload = function () {
  document.getElementById('file-button').onclick = function () {
    applyTokenDo(uploadFile);
  }
};
```

The following are the steps to uploading an object:

a. **Apply for a temporary permission from the authorization server. The authorization server is a service built by the** `website developer` **to provide temporary authorization for** `end users`**. Developers can provide different permissions for different users during authorization (see** *Overview*). **For**

authorization server information, see *GitHub*. To make it more convenient, the authorization server in the example returns the temporary credential to the user directly.

b. Create an OSS client using a temporary key.

c. Upload the selected files through the `multipartUpload` interface and set the progress bar through the `progress` parameter.

d. Other features

For text content uploading, object list retrieval, and object downloading features, see the example code: *OSS in Browser*.

12.12 Exception handling

If a request error occurs during SDK usage, an exception is thrown

containing the following information:

- status: the HTTP status code of the error request
- code: OSS error code
- message: OSS error message
- requestId: the UUID that uniquely identifies the request. Seek help from OSS development engineers by providing this RequestId if necessary.

For more information about common error messages in OSS, see *OSS Error Response*.

Debugging

In a browser environment, set the `localStorage.debug` variable in the console to enable debugging:

```
localStorage.debug = 'ali-oss'
```

12.13 FAQ

This topic describes common questions and answers.

How to call STS

A browser is an untrusted environment. It brings an extremely high risk if you store an AccessKeyID and AccessKeySecret in a browser.

for OSS interface calls in a browser environment.

After obtaining the STS token, you can initialize the SDK.

```
<script type="text/javascript">
  $.ajax("http://your_sts_server/",{method: 'GET'},function (err,
result) {
    let client = new OSS({
      accessKeyId: result.AccessKeyId,
   accessKeySecret: result.AccessKeySecret,
   stsToken: result.SecurityToken,
   endpoint: '<oss endpoint>',
   bucket: '<Your bucket name>'
    });
  });
</script>
```

How to enable HTTPS access

When you initialize the SDK, you can input the following parameters:

- **region: This parameter is the region used when you apply for the OSS service, such as `oss-cn-hangzhou`. Complete region lists can be viewed in *Regions and endpoints*.**
- **internal: Used in combination with `region`. If `internal` is set to `true`, visit the intranet node.**
- **secure: Used in combination with `region`. If `secure` is set to `true`, use the HTTPS for access.**
- **endpoint: For example, `http://oss-cn-hangzhou.aliyuncs.com`. If `endpoint` is specified, the `region` is ignored. You can specify the HTTPS or IP address when using `endpoint`.**

How to solve the cross-domain issues when using a browser

Set CORS attributes of buckets before using SDK in a browser. For more information, see bucket settings in *Quick start*.

How to set the user-defined data (meta), file type (mime), and request header (header) of the file to be uploaded

For more information, see *Multipart upload*.

Instruction for resumable upload on browser side

Save the checkpoint to a browser localStorage. You can directly use the checkpoint parameter next time to achieve resumable upload.

How to obtain the upload progress

You can obtain upload process when using multipart upload. For more information, see *Upload objects*.

How to obtain the download progress

The browser does not provide download progress. You can call the `signatureUrl` method to obtain the download address. For more information, see *Download objects*.

How to upload files to a specified directory

Add the specified directory prefix to the object name to be uploaded. For more information, see *Comparison between OSS and file system*.

```
let OSS = require('ali-oss')
let client = new OSS({
  region: '<Your region>',
  accessKeyId: '<Your AccessKeyId>',
  accessKeySecret: '<Your AccessKeySecret>',
  bucket: 'Your bucket name'
});

client.multipartUpload('base-dir/' +'object-key', 'local-file', {
    progress: async function (p) {
      console.log('Progress: ' + p);
    }
  });
  console.log(result);
}).catch((err) => {
  console.log(err);
});
```

How to upload a Base64-encoded image

Base64 first transcodes images into specified formats, and then calls OSS Upload interface to upload the images. For more information, see *Github examples*.

```
/**
 * base64 to file
 * @param dataurl   base64 content
 * @param filename  set up a meaningful suffix, or you can set mime
type in options
 * @returns {File|*}
 */
const dataURLtoFile = function dataURLtoFile(dataurl, filename) {
  const arr = dataurl.split(',');
  const mime = arr[0].match(/:(. *?) ;/)[1];
  const bstr = atob(arr[1]);
  let n = bstr.length;
  const u8arr = new Uint8Array(n);
  while (n--) {
    u8arr[n] = bstr.charCodeAt(n);
  }
```

```
  return new Blob([u8arr], { type: mime });// if env support File,
also can use this: return new File([u8arr], filename, { type: mime });
};

// client indicates OSS client instances
const uploadBase64Img = function uploadBase64Img(client) {
  // base64 format content
  const base64Content = 'data:image:xxxxxxxxxxxx';
  const filename =  'img.png';
  const imgfile = dataURLtoFile(base64Content, filename);
  //key indicates uploaded object key and imgFile indicates the
returned images after processing by dataURLtoFile
  client.multipartUpload(key, imgfile).then((res) => {
    console.log('upload success: %j', res);
  }).catch((err) => {
    console.error(err);
  });
};
```

How to limit the size of the uploaded file

The size (in bytes) of the uploaded file can be obtained in the browser according to document.getElementById(“file”).files[0].size. See the web post request.

How to obtain the signed URL of an object

You can call the `signatureUrl` method to obtain the download address. For more information, see *Download objects*.

How to use a signed URL generated by SDK to upload resources

Signed URLs are often used to authorize third parties to download and upload resources. For the download operations, see the answer for the next question. The signatureUrl API provided by SDK is used to return a signed URL and users can use this URL to upload or download resources directly. For more information about how to upload resources using signed URLs, see *Upload resources using signed URLs*.

How to upload resources to OSS servers using Form-based File Upload

For more information, see *Overview of direct transfer on Web client*.

Common errors

- *JS-SDK exception*
- *OSS error response*

13 Ruby

13.1 Installation

This topic describes how to install Ruby SDK. Install SDK with `gem` directly:

- *GitHub*
- *API documentation*
- *ChangeLog*

Prerequisites

- **You must have an active Alibaba Cloud OSS account. You must create an AccessKey ID and an AccessKey Secret.**
- **If you want more information about OSS service and want to activate the OSS service, log on to *OSS Product Homepage* to learn more.**
- **If you have not created an AccessKey ID and an AccessKey Secret, go to *Alibaba Cloud Access Key Management* to create an AccessKey.**

Installation

Install SDK with `gem` directly:

```
gem install aliyun-sdk
```

If the website https://rubygems.org is inaccessible, use the image source provided by Taobao.

```
gem install aliyun-sdk --clear-sources --source https://ruby.taobao.
org
```

You can also install SDK by using *Bundler*. Initially, add the following content to the `Gemfile` of your app:

```
gem 'aliyun-sdk', '~> 0.3.0'
```

Run the following command:

```
# (Optional). Use the image source provided by Taobao.
bundle config mirror.https://rubygems.org https://ruby.taobao.org
```

```
bundle install
```

Note:
https://ruby.taobao.org is the complete image of rubygems.org and automatically synchronizes with the official source. You can access https://ruby.taobao.org if rubygems.org is inaccessible.

Dependencies

- **Ruby 1.9.3 and later**
- **Windows, Linux, or OS X system that supports Ruby**

Notice:

- **Some gems on which the SDK depends are local extensions, and you must install ruby-dev to compile locally extended gems.**
- **The environment for running the SDK-dependent gem (nokogiri) for processing XML must have the zlib library.**

Linux

The following method is used to install the preceding dependencies through Ubuntu in Linux:

```
sudo apt-get install ruby ruby-dev zlib1g-dev
```

The installation methods for various Linux releases (each with a package management tool) are similar.

Windows

1. **Access *RubyInstaller* to download RubyInstaller. Double-click the installer to install the tool. Select "Add Ruby Executables to Your PATH" during the installation process.**

   Note:
   Download RubyInstaller 2.1 or an earlier version. RubyInstaller 2.2 cannot be installed due to *Issues*.

2. **Access *RubyInstaller* to download the development kit. Make sure you select the right version. The downloaded kit is a compressed package. First create a folder**

named `RubyDev` in the C:\ root directory, and decompress the package to the folder.

3. Press the shortcut key `Windows+R`, input `cmd`, and press Enter to open the CLI. Then run the following commands:

```
cd C:\RubyDev
ruby dk.rb init
ruby dk.rb install
```

If “config.yml configuration error” is displayed during the last step “install”, use the text editor to open the `C:\RubyDev\config.yml` file and modify the file content as follows:

```
---
- C:/Ruby21
```

Save the config.yml file and continue `ruby dk.rb install`. “Ruby21” is the Ruby installation directory, which must be specified according to the actual name. Close the CLI once the operation is completed.

4. Press the shortcut key `Windows+R`, input `cmd`, and press Enter to open the CLI. Run the following commands:

```
gem install aliyun-sdk
```

When installation is completed, input `irb` to open the Ruby interactive CLI, and input `require 'aliyun/oss'`. The SDK is successfully installed if “true” is displayed.

OS X

1. OS X has Ruby installed by default. To facilitate proper usage and management, we recommend you to install a development version.
2. Run `xcode-select --install` on your terminal to install “Xcode command line tools”. If installation fails, you can manually download and install the tools (see the steps to download).
3. Download Xcode command line tools from *Apple’s Developer Center* using your Apple ID. Make sure that your selected version matches your system version. After download, double-click the dmg file to load the file. Double-click the installation

program in the dialog box to install the tools. Enter your system password during the installation process.

4. Run the following command on your terminal to install Brew:

```
ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/
install/master/install)"
```

5. Run the following command on your terminal to install Ruby:

```
brew install ruby
exec $SHELL -l
```

6. Run the following command on your terminal to install SDK:

```
gem install aliyun-sdk
```

7. Run the following command on your terminal to check the installation status of SDK.

```
irb
> require 'aliyun/oss'
=> true
```

13.2 Quick start

This topic describes how to use the OSS Ruby SDK to access OSS services to view the bucket list and object list, to upload, download, and delete the objects.

For usage convenience, the following operations are performed in `irb` which is Ruby’s interactive command-line interface (CLI).

Initialize the client

Input the following content on the CLI and press the Enter:

```
irb
```

Enter the Ruby interactive CLI mode. Introduce the SDK package with `require`:

```
> require 'aliyun/oss'
=> true
```

Note:

In the following demonstration, the > symbol introduces the command that the user enters, and the => symbol introduces the content returned by a program.

Run the following command to create an OSSClient:

```
> client = Aliyun::OSS::Client.new(
>   endpoint: 'endpoint',
>   access_key_id: 'AccessKeyId',
>   access_key_secret: 'AccessKeySecret')
=> #<Aliyun::OSS::Client...
```

Replace the parameters with the actual endpoint, AccessKeyID, and AccessKeyS ecret.

View the bucket list

Run the following command to view the bucket list:

```
> buckets = client.list_buckets
=> #<Enumerator...
> buckets.each { |b| puts b.name }
=> bucket-1
=> bucket-2
=> ...
```

If the bucket list is empty, run the following command to create a bucket:

```
> client.create_bucket('my-bucket')
=> true
```

Note:

- **For bucket naming conventions, see *Basic OSS Concepts*.**
- **The bucket name cannot be the same with existing bucket names of other users in OSS. To create a bucket successfully, select a unique bucket name.**

View the object list

You can run the following command to view the objects in a bucket:

```
> bucket = client.get_bucket('my-bucket')
=> #<Aliyun::OSS::Bucket...
> objects = bucket.list_objects
=> #<Enumerator...
> objects.each { |obj| puts obj.key }
=> object-1
=> object-2
```

```
=> ...
```

Upload an object

You can run the following command to upload a file to a bucket:

```
> bucket.put_object('my-object', :file => 'local-file')
=> true
```

Specifically, `local-file` indicates the path to the local file to be uploaded. When the file is uploaded, you can view it using `list_objects`:

```
> objects = bucket.list_objects
=> #<Enumerator...
> objects.each { |obj| puts obj.key }
=> my-object
=> ...
```

Download an object

You can run the following command to download an object from a bucket:

```
> bucket.get_object('my-object', :file => 'local-file')
=> #<Aliyun::OSS::Object...
```

Specifically, `local-file` is the path to the downloaded object. After the object is downloaded, you can open the object to view its content.

Delete an object

You can run the following command to delete an object from a bucket:

```
> bucket.delete_object('my-object')
=> true
```

After deletion, you can run `list_objects` to check whether the object has been successfully deleted from the bucket.

```
> objects = bucket.list_objects
=> #<Enumerator...
> objects.each { |obj| puts obj.key }
=> object-1
=> ...
```

Learn more

- *Manage a bucket*
- *Upload objects*
- *Download objects*
- *Manage objects*

- *Integration with Rails*
- *CNAME*
- *Use STS to access OSS*
- *Set access permissions*
- *Manage lifecycle rules*
- *Set logging*
- *Static website hosting*
- *Anti-leech*
- *CORS*
- *Exception handling*

13.3 Buckets

13.3.1 Manage buckets

A bucket is a namespace on the OSS and a management entity for billing, access control, logging, and other advanced features.

View all buckets

The `Client#list_buckets` interface is used to list all buckets of the current user. Set the `:prefix` parameter to list only the buckets with the specific prefix in the name.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

buckets = client.list_buckets
buckets.each { |b| puts b.name }

buckets = client.list_buckets(:prefix => 'my-')
buckets.each { |b| puts b.name }
```

Create a bucket

The following code uses the `Client#create_bucket` interface to create a bucket. Specify the name of the bucket to be created:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
```

```
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

client.create_bucket('my-bucket')
```

Note:

- **For bucket naming conventions, see *Basic OSS Concepts*.**
- **The bucket name has to be globally unique. Make sure your bucket name is different from other users' .**

Delete a bucket

The following code uses the `Client#delete_bucket` interface to delete a bucket. Specify the name of the bucket to be deleted:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

client.delete_bucket('my-bucket')
```

Note:

- **If the bucket to be deleted has objects in it, delete the objects first.**
- **If the bucket has an incomplete upload request, use `list_uploads` and `abort_upload` to cancel the request before you delete the bucket. For the usage, see *API Documentation*.**

Check whether a bucket exists

The following code uses the `Client#bucket_exists` interface to check whether a specified bucket of the current user exists:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

puts client.bucket_exists?(' my-bucket')
```

Bucket access permissions

You can set the ACL policy of a bucket to allow or prohibit anonymous reads/writes to the bucket. For more information on ACL, see *ACL*.

- **Obtain the ACL of a bucket**

  The following code uses `Bucket#acl` to view the ACL of a bucket:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret
')

bucket = client.get_bucket('my-bucket')
puts bucket.acl
```

- **Set the ACL policy of a bucket**

  The following code uses `Bucket#acl=` to configure the ACL of a bucket:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret
')

bucket = client.get_bucket('my-bucket')
bucket.acl = Aliyun::OSS::ACL::PUBLIC_READ
puts bucket.acl
```

13.3.2 Set ACLs

OSS allows you to set ACLs for buckets and objects respectively. This helps you to conveniently control external access to your resources.

You can set three ACLs for a bucket:

- **public-read-write: Anonymous users are allowed to create/retrieve/delete objects in the bucket.**
- **public-read: Anonymous users are allowed to retrieve objects in the bucket.**
- **private: Anonymous users are not allowed to access objects in the bucket. Signature is required for all accesses.**

When a bucket is created, the private permission applies by default. You can use `bucket.acl=` to set the ACL of the bucket.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
puts bucket.acl
```

You can set four ACLs for an object.

- **default: The object inherits the ACL for the bucket it belongs to, that is, the ACL for the object is same as that for the bucket where the object is stored.**
- **public-read-write: Anonymous users are allowed to read/write the object.**
- **public-read: Anonymous users are allowed to read the object.**
- **private: Anonymous users are not allowed to access objects in the bucket. Signature is required for all accesses.**

When an object is created, the default permission applies by default. You can use `bucket.set_object_acl` to configure the ACL of the object.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

acl = bucket.get_object_acl('my-object')
puts acl # default
bucket.set_object_acl('my-object', Aliyun::OSS::ACL::PUBLIC_READ)
acl = bucket.get_object_acl('my-object')
puts acl # public-read
```

Note:

- **If an object is configured with an ACL policy (not default), the object ACL takes priority during permission authentication when the object is accessed. The bucket ACL is ignored.**
- **If anonymous access is allowed (public-read or public-read-write is configured for the object), you can directly access the object using a browser.**

  ```
  For example: http://bucket-name.oss-cn-hangzhou.aliyuncs.com/
  object.jpg
  ```

- **A bucket or an object with the public ACL can be accessed by an anonymous client:**

  ```
  require 'aliyun/oss'

  # If access_key_id and access_key_secret are not specified, an
  anonymous client will be created. The client can access only
  # the buckets and objects with the public ACL.
  client = Aliyun::OSS::Client.new(endpoint: 'endpoint')
  ```

```
bucket = client.get_bucket('my-bucket')

bucket.get_object('my-object', :file => 'local_file')
```

- **For more information about ACL, see** *Access Control*.

13.3.3 Manage lifecycle rules

OSS allows you to set lifecycle rules for buckets to automatically remove expired objects and save the storage space.

You can configure lifecycle rules for OSS buckets or objects to automatically delete expired objects and parts or change the storage class of expired objects to IA or Archive, saving storage costs. A lifecycle rule includes the following elements:

- **Rule ID: Identifies a lifecycle rule. A rule ID is unique in a bucket.**
- **Strategy: You can set one of the following two application strategies for a rule. The strategy for all rules set for a bucket must be the same.**
  - **The rule applies to objects with specified prefixes. You can create multiple rules with this strategy. Prefixes specified in different rules must be different.**
  - **The rule applies to the entire bucket. You can create only one rule with this strategy.**
- **Expiration period:**
  - **Specifies the expiration period (N days) of a lifecycle rule. An object expires N days after it is modified for the last time.**
  - **Specifies an expiration time. Objects modified before the time expire.**
- **Status: Indicates whether the rule is enabled or disabled.**

For more information, see *Lifecycle rules*.

Configure lifecycle rules

You can use `Bucket#lifecycle=` **to configure lifecycle rules:**

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket.lifecycle = [
  LifeCycleRule.new(
    :id => 'rule1', :enabled => true, :prefix => 'foo/', :expiry => 3
),
  LifeCycleRule.new(
```

```
    :id => 'rule2', :enabled => false, :prefix => 'bar/', :expiry =>
Date.new(2016, 1, 1))
]
```

View lifecycle rules

You can use `Bucket#lifecycle` to view lifecycle rules:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

rules = bucket.lifecycle
puts rules
```

Clear lifecycle rules

You can use `Bucket#lifecycle=` to configure an empty rule array to clear lifecycle rules:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket.lifecycle = []
```

13.3.4 Hotlink protection

OSS is a Pay-As-You-Go service. To prevent users' data on the OSS from being leeched, OSS supports hotlink protection based on the referer field in the HTTP header.

To prevent your data on OSS from being leeched, OSS supports hotlink protection through the referer field settings in the HTTP header, including the following parameters:

- **Referer whitelist: Used to allow access only for specified domains to OSS data.**
- **Empty referer: Determines whether the referer can be empty. If it is not allowed , only requests with the referer filed in their HTTP or HTTPS headers can access OSS data.**

For more information about the OSS hotlink protection features, see *#unique_49*.

Configure a referer whitelist

The following code uses `Bucket#referer=` to configure a referer whitelist:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.referer = BucketReferer.new(
  allow_empty: true, whitelist: ['my-domain.com', '*.example.com'])
```

View a referer whitelist

The following code uses `Bucket#referer` to set a referer whitelist:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
ref = bucket.referer
puts ref.to_s
```

Clear a referer whitelist

The following code uses `Bucket#referer=` to clear a referer whitelist:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.referer = BucketReferer.new(allow_empty: true, whitelist: [])
```

13.3.5 Set logging

OSS allows you to set access logging for a bucket. After the configuration, accesses to the bucket are logged.

The logs are then stored in a specified bucket on OSS in the format of:

```
<TargetPrefix><SourceBucket>-YYYY-mm-DD-HH-MM-SS-UniqueString
```

You must specify the `TargetPrefix`. A logging rule consists the following:

- **enable: Whether to enable access logging.**
- **target_bucket: The bucket where logs are stored.**
- **target_prefix: The prefix of the log object**

For more information, see *Set access logging*.

Enable bucket logging

Use `Bucket#logging=` to enable logging:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

bucket.logging = BucketLogging.new(
  enable: true, target_bucket: 'logging_bucket', target_prefix: 'my-
log')
```

View bucket logging settings

Use `Bucket#logging` to view logging settings:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
log = bucket.logging
puts log.to_s
```

Disable bucket logging

Use `Bucket#logging=` to disable logging.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
```

```
bucket.logging = BucketLogging.new(enable: false)
```

13.3.6 CORS

This topic describes how to perform CORS.

Cross-origin resource sharing (CORS) allows web applications to access resources that belong to another region. OSS provides CORS APIs for convenient cross-origin access control.

For more information, see *Cross-origin resource sharing* in OSS Developer Guide.

OSS CORS settings contain one or more CORS rules. Each CORS rule includes the following parameters:

- **allowed_origins: The origins allowed for cross-origin requests, for example, www.my-domain.com, *.**
- **allowed_methods: The HTTP methods (PUT/POST/GET/DELETE/HEAD) allowed for cross-origin requests.**
- **allowed_headers: The headers allowed in a prefetch command (OPTIONS), for example, x-oss-test, *.**
- **expose_headers: The headers allowed for the user to access in the application.**
- **max_age_seconds: The cache time for the returned result of a browser prefetch (OPTIONS) request to a specific resource.**

Configure CORS rules

The following code uses `Bucket#cors=` to configure a CORS rule:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.cors = [
    CORSRule.new(
      :allowed_origins => ['aliyun.com', 'http://www.taobao.com'],
      :allowed_methods => ['PUT', 'POST', 'GET'],
      :allowed_headers => ['Authorization'],
      :expose_headers => ['x-oss-test'],
      :max_age_seconds => 100)
```

```
]
```

View CORS rules

The following code uses `Bucket#cors` to display CORS rules:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
cors = bucket.cors
puts cors.map(&:to_s)
```

Clear CORS rules

The following code uses `Bucket#cors=` to clear CORS rules:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.cors = []
```

13.3.7 Static website hosting

You can set your bucket to the static website hosting mode. After the configuration takes effect, you can access this static website with the bucket domain and be redirected to a specified index page or error page.

In *CNAME*, we mentioned that the OSS allows you to direct your domain name to the OSS service address. In this way, the OSS bucket is accessed when you access your website. You must specify the names of the objects in the bucket corresponding to the homepage (index) and error page (error) of the website respectively.

For more information, see *Static website hosting*.

Configure website hosting

Use `Bucket#website=` to configure hosted pages:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
```

```
bucket.website = BucketWebsite.new(index: 'index.html', error: 'error.
html')
```

View website hosting

Use `Bucket#website` to display hosted pages:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
web = bucket.website
puts web.to_s
```

Clear website hosting

Use `Bucket#website=` to clear hosted pages:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.website = BucketWebsite.new(enable: false)
```

13.3.8 CNAME

This topic describes how to bind a custom domain name.

OSS supports binding custom domain names to OSS to seamlessly migrate data storage to the OSS. For example, your domain name is my-domain.com, and all your previous image resources are in a format similar to `http://img.my-domain.com/x.jpg`. After you migrate the image storage to OSS, you can still access the images with the original address by binding your custom domain name to OSS:

1. **Activate OSS and create a bucket**
2. **Bind img.my-domain.com to the created bucket.**
3. **Upload images to the created bucket in OSS.**
4. **Modify the DNS configuration, add a CNAME record, and direct img.my-domain.com to an endpoint of OSS (for example, my-bucket.oss-cn-hangzhou.aliyuncs.com)**

Follow the preceding steps to use the original address `http://img.my-domain.com/xx.jpg` to access images on OSS. For more information, see *Bind custom domain names (CNAME)*.

SDK supports the use of a CNAME as the endpoint. To do this, set the `:cname` parameter to true. For example:

```
require 'aliyun/oss'

include Aliyun::OSS

client = Client.new(
  endpoint: 'ENDPOINT',
  access_key_id: 'ACCESS_KEY_ID',
  access_key_secret: 'ACCESS_KEY_SECRET',
  cname: true)

bucket = client.get_bucket('my-bucket')
```

Note:

The list_buckets interface is unavailable when CNAME is used, because the custom domain name is bound already to a specific bucket.

13.4 Upload objects

This topic describes how to upload objects.

The OSS Ruby SDK provides rich interfaces for object upload. You can upload an object from OSS through any of the following methods:

- Upload a local file to the OSS
- Stream upload
- Resumable upload
- Append upload
- Upload callback

Upload a local file

The following code uses the `Bucket#put_object` interface with the `:file` parameter specified to upload a local file to OSS:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
```

```
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.put_object('my-object', :file => 'local-file')
```

Stream upload

During uploading a large file, we often need to upload the file in a streaming way instead of processing and uploading all the content at the same time. We want to upload a part of content at a time. In particular, when the content to be uploaded comes from the Internet and cannot be retrieved at the same time, stream upload is the only option left. Through`Bucket # put_object`interface and specify`block` parameter to upload the stream-generated content to the OSS:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.put_object('my-object') do |stream|
  100.times { |i| stream << i.to_s }
end
```

Resumable upload

If the network jitters or the program crashes when a large object is being uploaded, the whole upload operation fails. You have to re-upload the objects, wasting resources. When the network is unstable, you may need to make multiple retries. The `Bucket#resumable_upload` interface can be used to achieve resumable upload. The interface has the following parameters:

- **key: The name of the object on OSS mapped to the uploaded file.**
- **file: The local path of the file to be uploaded.**

- opts: An optional parameter. It mainly includes:
  - :cpt_file: The path of the checkpoint file. If not specified, the path of the checkpoint object is by default the `file.cpt` under the same directory of the local file. The `file` is the name of the local file.
  - :disable_cpt: If this parameter is set to true, the upload progress is not recorded during upload, and resumable upload is not performed when upload fails.
  - :part_size: The size of each part. The default size is 4 MB.
  - &block: If a block function is imported when the Bucket#resumable_upload interface is called, the upload progress is handled by the block function.

For more information about the parameters, see *API documentation*.

The principle of resumable upload is to divide the object to be uploaded into multiple parts and upload them separately. When all the parts are uploaded, the upload of the entire object is completed. The progress of the current upload is recorded during the uploading process(in the checkpoint object). If the upload of any part fails during the process, the next upload attempt starts from the recorded position in the checkpoint object. This requires that the same checkpoint object with the previous upload must be used in the next call. When the upload is completed, the checkpoint object is deleted.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.resumable_upload('my-object', 'local-file') do |p|
  puts "Progress: #{p}"
end

bucket.resumable_upload(
  'my-object', 'local-file',
  :part_size => 100 * 1024, :cpt_file => '/tmp/x.cpt') { |p|
  puts "Progress: #{p}"
}
```

Note:

- The SDK records the upload intermediate states in the cpt object. Therefore, make sure that you have the write permission on the cpt object.

- The cpt object records the intermediate state information of the upload and has a self-checking function. You cannot edit the object. Upload fails if the cpt object is corrupted. When the upload is completed, the checkpoint object is deleted.
- Upload fails if the local file is changed during the upload process.

Append upload

OSS supports the appendable object type. The `Bucket#append_object` interface is used to upload appendable objects. When calling this method, you must specify the append position of the object. If the object is newly created, the append position is 0. If the object already exists,the append position must be set to the object length before the append.

- If the object to be appended does not exist, calling `append_object` creates an appendable object.
- If the object to be appended exists, calling `append_object` appends the content to the end of the object.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
# Create an appendable object.
bucket.append_object('my-object', 0) {}

# Append content to the end of the object.
next_pos = bucket.append_object('my-object', 0) do |stream|
  100.times { |i| stream << i.to_s }
end
next_pos = bucket.append_object('my-object', next_pos, :file => 'local
-file-1')
next_pos = bucket.append_object('my-object', next_pos, :file => 'local
-file-2')
```

Note:

- You can append content to appendable object (objects created with `append_object`) only.
- You cannot copy appendable objects.

Upload callback

You can specify an “upload callback” when uploading a file. After the file is successfully uploaded to OSS, OSS initiates an HTTP POST request to the server

address that you have provided. The request is equal to a notification mechanism. You can perform operations according to the received callback results. For more information about upload callback, see *OSS Upload Callback*.

Currently, only the `put_object` and `resumable_upload` interfaces of the OSS support upload callback.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

callback = Aliyun::OSS::Callback.new(
  url: 'http://10.101.168.94:1234/callback',
  query: {user: 'put_object'},
  body: 'bucket=${bucket}&object=${object}'
)

begin
  bucket.put_object('files/hello', file: '/tmp/x', callback: callback)
rescue Aliyun::OSS::CallbackError => e
  puts "Callback failed: #{e.message}"
end
```

In the preceding code, a file is uploaded through the `put_object` interface with an upload callback specified. The callback body contains the bucket and object information for this upload. Upon receiving the callback, the application server determines that the file has been successfully uploaded to the OSS.

The usage of the r`esumable_upload` interface is similar:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

callback = Aliyun::OSS::Callback.new(
  url: 'http://10.101.168.94:1234/callback',
  query: {user: 'put_object'},
  body: 'bucket=${bucket}&object=${object}'
)

begin
  bucket.resumable_upload('files/hello', '/tmp/x', callback: callback)
rescue Aliyun::OSS::CallbackError => e
  puts "Callback failed: #{e.message}"
```

```
end
```

Note:

- **The callback URL must not contain the query string which must be specified in the `:query` parameter.**
- **In the event that the file is successfully uploaded but callback execution fails, the client throws `CallbackError`. To ignore the error, you must explicitly catch the exception.**
- **For detailed examples, see *callback.rb*.**
- **For servers that support callback, see *callback_server.rb*.**

13.5 Download objects

This topic describes how to download an object from OSS to a local file.

The OSS Ruby SDK provides rich interfaces for object download. You can download an object from OSS through any of the following methods:

- **Download to a local file**
- **Stream download**
- **Resumable download**
- **HTTP download (by a browser)**

Download an object to a local file

The following code uses the `Bucket#get_object` interface with the `:file` parameter specified to download an object as a local file:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.get_object('my-object', :file => 'local-file')
```

Stream download

When uploading a large file, we often need to upload the file in a streaming way instead of processing and uploading all the content at the same time. We want to upload a part of content at a time. The following code uses the `Bucket#get_object`

interface with the `block` parameter specified to process downloaded content through streaming:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.get_object('my-object') do |chunk|
  # handle_data(chunk)
  puts "Got a chunk, size: #{chunk.size}."
end
```

Resumable download

If the network jitters or the program crashes when a large object is being downloaded, the whole download operation fails. You have to re-download the objects, wasting resources. When the network is unstable, you may need to make multiple retries. The `Bucket#resumable_download` interface can be used to achieve resumable download. The interface has the following parameters:

- **key: The name of the object to be downloaded.**
- **file: The local path to the downloaded object.**
- **opts: An optional parameter. It mainly includes:**
  - **:cpt_file: The path of the checkpoint file. If not specified, the path of the checkpoint object is by default the `file.cpt` under the same directory of the local `file`. The file is the name of the local file.**
  - **:disable_cpt: If this parameter is set to true, the download progress is not recorded during download, and resumable upload is not performed when upload fails.**
  - **:part_size: The size of each part. The default size is 10 MB.**
  - **&block: If a block function is imported when the Bucket#resumable_ download interface is called, the download progress is handled by the block function.**

For more information about the parameters, see *API Documentation*.

The principle is to divide the object to be downloaded into multiple parts and download them separately. When all the parts are downloaded, the download of the entire object is completed. The progress of the current download task and downloaded parts is recorded (recorded in the checkpoint file) and downloaded

parts (saved as `file.part.N`, among which `file` is the name of the object downloaded to a local path). If one part fails to be downloaded, the re-download starts from the recorded position in the checkpoint object. This requires that the same checkpoint object with the previous download must be used in the next call. The file.part.N files and checkpoint object is deleted when download is completed.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
bucket.resumable_download('my-object', 'local-file') do |p|
  puts "Progress: #{p}"
end

bucket.resumable_download(
  'my-object', 'local-file',
  :part_size => 100 * 1024, :cpt_file => '/tmp/x.cpt') { |p|
  puts "Progress: #{p}"
}
```

Note:

- The SDK records the download intermediate states in the cpt object; therefore, make sure that you have the write permission on the cpt object.
- The SDK saves downloaded parts as file.part.N files; therefore, make sure that you have the file creation permission on the directory where the file.part.N files are stored.
- The cpt object records the intermediate state information of the download and has a self-checking function. You cannot edit the object. Download fails if the cpt object is corrupted.
- If the object to be downloaded is changed during download (the ETag is changed) or a part is missing or modified, an error is reported.

HTTP download

Objects stored in OSS can be downloaded through HTTP directly without using the SDK. HTTP download supports browsers and command-line interface (CLI) tools (such as `wget` and `curl`). The URL of the object to be downloaded is generated by the SDK. The `Bucket#object_url` method is used to generate an HTTP download address which supports the following parameters:

- key: The name of the object to be downloaded.

- **sign: Whether to generate a signed URL. An object with the public-read or public-read-write permission can be accessed without a signed URL. An object with the private permission must be accessed with a signed URL.**
- **expiry: The validity period of the URL. The default value is 60 seconds.**

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
# Generate a URL. By default, the generated URL is a signed URL valid
for 60 seconds.
puts bucket.object_url('my-object')
# http://my-bucket.oss-cn-hangzhou.aliyuncs.com/my-object?Expires=
1448349966&OSSAccessKeyId=5vxxxx&Signature=aM2HpBLeMq1aec6JCd7BBAKYiwI
%3D

# Generate an unsigned URL
puts bucket.object_url('my-object', false)
# http://my-bucket.oss-cn-hangzhou.aliyuncs.com/my-object

# Set the URL validity period to one hour (3,600 seconds)
puts bucket.object_url('my-object', true, 3600)
```

13.6 Manage objects

A bucket may contain a lot of objects. The SDK provides a series of interfaces for you to conveniently manage the objects.

View all objects

The following code uses `Bucket#list_objects` to list all objects in the current bucket. Main parameters are described as follows:

- **prefix: Specify to only list objects with specific prefixes.**
- **marker: Specify to list objects whose names are greater than a specified marker.**
- **delimiter: Obtain the common prefix of objects**

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
# List all objects
objects = bucket.list_objects
objects.each { |o| puts o.key }

# List all objects with the "my-" prefix in the name
objects = bucket.list_objects(:prefix => 'my-')
```

```
objects.each { |o| puts o.key }

# List all objects that follow the "my-object" marker and whose names
contain the "my-" prefix
objects = bucket.list_objects(:prefix => 'my-', :marker => 'my-object
')
objects.each { |o| puts o.key }
```

Simulate the directory structure

OSS is an object-based storage service. It has no directory. All objects stored in a bucket are uniquely identified by the key of the object. No hierarchical structure is established among these objects. Such a structure allows OSS to efficiently store objects. However, you may want to put different types of objects under different "directories" to manage objects in a conventional way. OSS provides a "Public Prefix" feature for you to conveniently create a stimulated directory structure. For more information about the concept of a public prefix, see *View Object List*.

Suppose that the bucket already contains the following objects:

```
foo/x
foo/y
foo/bar/a
foo/bar/b
foo/hello/C/1
foo/hello/C/2
...
foo/hello/C/9999
```

Next we will implement a function named `list_dir` to list all objects and subdirectories in a specified directory:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

def list_dir(dir)
  objects = bucket.list_objects(:prefix => dir, :delimiter => '/')
  objects.each do |obj|
    if obj.is_a?( OSS::Object) # object
      puts "Object: #{obj.key}"
    else # common prefix
      puts "SubDir: #{obj}"
    end
  end
end
```

The execution results are as follows:

```
> list_dir('foo/')
```

```
=> SubDir: foo/bar/
   SubDir: foo/hello/
   Object: foo/x
   Object: foo/y

> list_dir('foo/bar/')
=> Object: foo/bar/a
   Object: foo/bar/b

> list_dir('foo/hello/C/')
=> Object: foo/hello/C/1
   Object: foo/hello/C/2
   ...
   Object: foo/hello/C/9999
```

Object Meta

When uploading an object to OSS, except for the object content, you can specify certain attribute information for the object. The attribute information is called Object Meta. The metadata is stored together with the object during uploading, and returned together with the object during downloading.

In the SDK, the metadata of an object is expressed by a `Hash`, and the object key and value are both of the `String` type, and only contain simple printable ASCII characters and cannot contain line breaks. The total size of all metadata cannot exceed 8 KB.

When using `Bucket#put_object`, `Bucket#append_object`, and `Bucket#resumable_upload`, you can specify the Object Meta by specifying the : `metas` parameter.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

bucket.put_object(
  'my-object-1',
  :file => 'local-file',
  :metas => {'year' => '2016', 'people' => 'mary'})

bucket.append_object(
  'my-object-2', 0,
  :file => 'local-file',
  :metas => {'year' => '2016', 'people' => 'mary'})

bucket.resumable_upload(
  'my-object',
  'local-file',
```

```
  :metas => {'year' => '2016', 'people' => 'mary'})
```

Run the `Bucket#update_object_metas` command to update Object Meta.

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

bucket.update_object_metas('my-object', {'year' => '2017'})
```

Copy objects

The following code uses `Bucket#copy_object` to copy an object. When an object is copied, two methods can be used to process the Object Meta. The method used is specified by the `:meta_directive` parameter:

- **Method 1: Stay consistent with the source object metadata through the COPY operation.**
- **Method 2: Overwrite the source object metadata with the new metadata.**

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

# Copy the source object metadata
bucket.copy_object(
  'my-object', 'copy-object',
  :meta_directive => Aliyun::OSS::MetaDirective::COPY)

# Overwrite the source object metadata
bucket.copy_object(
  'my-object', 'copy-object',
  :metas => {'year' => '2017'},
  :meta_directive => Aliyun::OSS::MetaDirective::REPLACE)
```

Delete an object

The following code uses `Bucket#delete_object` to delete an object:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')
```

```
bucket.delete_object('my-object')
```

Delete objects in batches

The following code uses `Bucket#batch_delete_object` to delete objects in batches. You can set the `:quiet` parameter to determine whether to return the deletion results:

```
require 'aliyun/oss'

client = Aliyun::OSS::Client.new(
  endpoint: 'endpoint',
  access_key_id: 'AccessKeyId', access_key_secret: 'AccessKeySecret')

bucket = client.get_bucket('my-bucket')

objs = ['my-object-1', 'my-object-2']
result = bucket.batch_delete_object(objs)
puts result #['my-object-1', 'my-object-2']. Names of successfully
deleted objects are returned by default.

objs = ['my-object-3', 'my-object-4']
result = bucket.batch_delete_object(objs, :quiet => true)
puts result #[]. The deletion result is not returned.
```

13.7 Use STS to access OSS

OSS can temporarily grant authorization for access through the Alibaba Cloud STS service.

For more information about STS, see *Concepts*.

Follow these steps to use the STS:

1. **Create a RAM user in the console of the official website. For more information, see *Overview*.**
2. **Create an STS role in the console and grant permission to the role of the RAM user. For more information, see *Overview*.**
3. **Use the AccessKeyID/AccessKeySecret of the RAM user to apply for a temporary token from STS.**
4. **Use the authentication information in the temporary token to create an OSS client.**
5. **Use the OSS client to access OSS.**

You must set the `:sts_token` parameter to access OSS through STS. See the following example:

```
require 'aliyun/sts'
```

```
require 'aliyun/oss'

sts = Aliyun::STS::Client.new(
  access_key_id: '<AccessKeyId of the RAM user>',
  access_key_secret: '<AccessKeySecret of the RAM user>')

token = sts.assume_role('<role-arn>', '<session-name>')

client = Aliyun::OSS::Client.new(
  endpoint: '<endpoint>',
  access_key_id: token.access_key_id,
  access_key_secret: token.access_key_secret,
  sts_token: token.security_token)

bucket = client.get_bucket('my-bucket')
```

You can customize an STS policy when applying for a temporary token from STS. The requested temporary permission is the `intersection of the permission assigned to the role and the permission specified by the STS policy`. The following code applies for the read-only permission on `my-bucket` using a specified STS policy and sets the temporary token validity period to 15 minutes:

```
require 'aliyun/sts'
require 'aliyun/oss'

sts = Aliyun::STS::Client.new(
  access_key_id = '<The AccessKeyId of the RAM user>'
  access_key_secret = '<The AccessKeySecret of the RAM user>'

policy = Aliyun::STS::Policy.new
policy.allow(['oss:Get*'], ['acs:oss:*:*:my-bucket/*'])

token = sts.assume_role('<role arc>', '<session name>', policy, 15 *
60)

client = Aliyun::OSS::Client.new(
  endpoint: 'ENDPOINT',
  access_key_id: token.access_key_id,
  access_key_secret: token.access_key_secret,
  sts_token: token.security_token)

bucket = client.get_bucket('my-bucket')
```

For detailed usage and parameters, see *API documentation*.

13.8 Exception handling

When a request error occurs during SDK usage, an exception is thrown, and the error details are logged (saved to the program runtime directory oss_sdk.log by default).

The two types of OSS Ruby SDK exceptions are: ClientError and ServerError and both are the subclasses of RuntimeError.

ClientError

ClientError is an internal SDK error, such as incorrect parameter setting or object modification during the resumable upload or download.

ServerError

ServerError indicates a server error, generated by parsing a server error message. ServerError has the following attributes:

- **http_code: The HTTP status code of the error request.**
- **error_code: OSS error code.**
- **message: OSS error message.**
- **request_id: The UUID that uniquely identifies the request. When you cannot solve the problem, you can seek help from OSS development engineers by providing this RequestId.**

For more information, see *OSS error response*.

13.9 Application of Rails

This topic describes the application of Rails.

Integration with Rails

To use the OSS Ruby SDK in Rails, add the following dependency to `Gemfile`:

```
gem 'aliyun-sdk', '~> 0.3.0
```

Then introduce the dependency when you use OSS:

```
require 'aliyun/oss'
```

The "rails/ " directory of the SDK provides helper code to facilitate your operation.

The following describes how to use the SDK to implement a simple OSS object manager (oss-manager) which provides the following functions:

- **List all the buckets of a user.**
- **List all the objects in a bucket by level of directory.**
- **Upload objects.**
- **Download objects.**

The integration process is described as follows:

1. **Create a project**

   Install Rails and create a Rails application oss-manager:

   ```
   gem install rails
   rails new oss-manager
   ```

   We recommend, you manage your project code in GitHub:

   ```
   cd oss-manager
   git init
   git add .
   git commit -m "init project"
   ```

2. **Add the SDK dependency**

   Add the SDK dependency to `oss-manager/Gemfile`:

   ```
   gem 'aliyun-sdk', '~> 0.3.0'
   ```

   Run the following command in `oss-manager/`:

   ```
   bundle install
   ```

   Save the change in this step:

   ```
   git add .
   git commit -m "add aliyun-sdk dependency"
   ```

3. **Initialize an OSSClient**

   To avoid initializing an OSSClient instance each time you use it in the project, you can add an initialization file to your project.

   ```
   # oss-manager/config/initializers/aliyun_oss_init.rb
   require 'aliyun/oss'

   module OSS
     def self.client
       unless @client
         Aliyun::Common::Logging.set_log_file('./log/oss_sdk.log')

         @client = Aliyun::OSS::Client.new(
           endpoint:
             Rails.application.secrets.aliyun_oss['endpoint'],
           access_key_id:
             Rails.application.secrets.aliyun_oss['access_key_id'],
           access_key_secret:
             Rails.application.secrets.aliyun_oss['access_key_secret']
         )
       end

       @client
     end
   ```

```
end
```

The preceding code can be found in the “rails/ “ directory of the SDK. The initialization file simplifies the use of the OSSClient in your project:

```
buckets = OSS.client.list_buckets
```

The endpoint, AccessKeyID, and AccessKeySecret are stored in `oss-manager/conf/secrets.yml`. For example:

```
development:
  secret_key_base: xxxx
  aliyun_oss:
    endpoint: xxxx
    access_key_id: aaaa
    access_key_secret: bbbb
```

Save the code:

```
git add .
git commit -m "add aliyun-sdk initializer"
```

4. **Implement the “List Buckets” feature**

   Firstly, use Rails to generate a controller for bucket management:

   ```
   rails g controller buckets index
   ```

   The following objects are generated in `oss-manager`:

   - **app/controller/buckets_controller.rb: The bucket-related logic code**
   - **app/views/buckets/index.html.erb: The bucket-related demonstration code**
   - **app/helpers/buckets_helper.rb: Some helper functions**

   Edit buckets_controller.rb. Call the OSSClient to save the `list_buckets` results to the `@buckets` variable:

   ```
   class BucketsController < ApplicationController
     def index
       @buckets = OSS.client.list_buckets
     end
   end
   ```

   Edit views/buckets/index.html.erb to display the bucket list:

   ```
   <h1>Buckets</h1>
   <table class="table table-striped">
     <tr>
       <th>Name</th>
       <th>Location</th>
       <th>CreationTime</th>
     </tr>
   ```

```
  <% @buckets.each do |bucket| %>
  <tr>
    <td><%= link_to bucket.name, bucket_objects_path(bucket.name)
 %></td>
    <td><%= bucket.location %></td>
    <td><%= bucket.creation_time.localtime.to_s %></td>
  </tr>
  <% end %>
</table>
```

The `bucket_objects_path` is a helper function is stored inapp/helpers/buckets_helper.rb:

```
module BucketsHelper
  def bucket_objects_path(bucket_name)
    "/buckets/#{bucket_name}/objects"
  end
end
```

This is how all the buckets are listed. Before you implement the function, configure a Rails route so that the correct logic can be called after an address is entered in the address bar of the browser. Edit `config/routes.rb` and add the following:

```
resources :buckets do
  resources :objects
end
```

Now, input `rails s` in oss-manager/ to start the Rails server, and input `http://localhost:3000/buckets/` in the address bar of the browser. The bucket list is displayed.

Finally, save the code.

```
git add .
git commit -m "add list buckets feature"
```

5. **Implement the “List Objects” feature**

   First generate a controller for object management:

   ```
   rails g controller objects index
   ```

   Edit app/controllers/objects_controller.rb as follows:

   ```
   class ObjectsController < ApplicationController
     def index
       @bucket_name = params[:bucket_id]
       @prefix = params[:prefix]
       @bucket = OSS.client.get_bucket(@bucket_name)
       @objects = @bucket.list_objects(:prefix => @prefix, :delimiter
    => '/')
   ```

```
  end
end
```

The preceding code obtains the bucket name from the URL parameters first. An additional prefix is required to list objects by directory level. Call the list_objects interface of the OSSClient to obtain the object list. Note that the preceding procedure provide a list of the objects which are delimited by the slash (/) and whose names contain a specified prefix. This also aims to display the object list by directory level. See *Manage objects*.

Next, edit app/views/objects/index.html.erb as follows:

```
<h1>Objects in <%= @bucket_name %></h1>
<p> <%= link_to 'Upload file', new_object_path(@bucket_name, @prefix
) %></p>
<table class="table table-striped">
  <tr>
    <th>Key</th>
    <th>Type</th>
    <th>Size</th>
    <th>LastModified</th>
  </tr>

  <tr>
    <td><%= link_to '../', with_prefix(upper_dir(@prefix)) %></td>
    <td>Directory</td>
    <td>N/A</td>
    <td>N/A</td>
  </tr>

  <% @objects.each do |object| %>
  <tr>
    <% if object.is_a?( Aliyun::OSS::Object) %>
    <td><%= link_to remove_prefix(object.key, @prefix),
            @bucket.object_url(object.key) %></td>
    <td><%= object.type %></td>
    <td><%= number_to_human_size(object.size) %></td>
    <td><%= object.last_modified.localtime.to_s %></td>
    <% else  %>
    <td><%= link_to remove_prefix(object, @prefix), with_prefix(
object) %></td>
    <td>Directory</td>
    <td>N/A</td>
    <td>N/A</td>
    <% end  %>
  </tr>
  <% end %>
```

```
</table>
```

In the preceding code, the main logic for listing objects according to the directory structure is as follows:

a. “../ “ always appears at the beginning to indicate the upper-level directory.

b. Objects with a common prefix in their names are displayed as directories.

c. Objects are displayed as files.

The preceding code uses `with_prefix`, `remove_prefix`, and other helper functions which are defined in app/helpers/objects_helper.rb:

```
module ObjectsHelper
  def with_prefix(prefix)
    "? prefix=#{prefix}"
  end

  def remove_prefix(key, prefix)
    key.sub(/^#{prefix}/, '')
  end

  def upper_dir(dir)
    dir.sub(/[^\/]+\/$/, '') if dir
  end

  def new_object_path(bucket_name, prefix = nil)
    "/buckets/#{bucket_name}/objects/new/#{with_prefix(prefix)}"
  end

  def objects_path(bucket_name, prefix = nil)
    "/buckets/#{bucket_name}/objects/#{with_prefix(prefix)}"
  end
end
```

Once you run the preceding code, run `rails s`, and enter `http://localhost:3000/buckets/my-bucket/objects/` in the address bar of the browser. The object list is displayed.

Save the code as usual:

```
git add .
git commit -m "add list objects feature"
```

6. **Download objects**

Note the the preceding code adds a link to each listed object:

```
<td><%= link_to remove_prefix(object.key, @prefix),
        @bucket.object_url(object.key) %></td>
```

The `Bucket#object_url` method is used to generate a temporary URL for an object. For more information, see *Download objects*.

7. **Upload files**

   You can use either of the following methods to upload files with a server app such as Rails:

   a. **Upload a file to the Rails server that uploads the object to the OSS. In this method, the Rails server works as a transit server by copying the object to the OSS. The upload process is inefficient.**
   b. **Upload a file to the OSS directly with the form and temporary credentials generated by the Rails server.**

   The first method is relatively simple, similar to the common file upload method. The following describes how to use the second method to upload a file:

   Add a `#new` method to app/controllers/objects_controller.rb to generate an upload form.

```
def new
  @bucket_name = params[:bucket_id]
  @prefix = params[:prefix]
  @bucket = OSS.client.get_bucket(@bucket_name)
  @options = {
    :prefix => @prefix,
    :redirect => 'http://localhost:3000/buckets/'
  }
end
```

   Then edit app/views/objects/new.html.erb as follows:

```
<h2>Upload object</h2>

<%= upload_form(@bucket, @options) do %>
  <table class="table table-striped">
    <tr>
      <td><label>Bucket:</label></td>
      <td><%= @bucket.name  %></td>
    </tr>
    <tr>
      <td><label>Prefix:</label></td>
      <td><%= @prefix  %></td>
    </tr>

    <tr>
      <td><label>Select file:</label></td>
      <td><input type="file" name="file" style="display:inline" /></
td>
    </tr>

    <tr>
      <td colspan="2">
        <input type="submit" class="btn btn-default" value="Upload
" />
        <span>  </span>
        <%= link_to 'Back', objects_path(@bucket_name, @prefix) %>
      </td>
```

```
    </tr>
  </table>
<% end %>
```

Specifically, the `upload_form` is a helper function provided by SDK to generate an upload form. The function is stored in rails/aliyun_oss_helper.rb of the SDK. You must copy the function to the app/helpers/ directory, run `rails s`, and input the `http://localhost:3000/buckets/my-bucket/objects/new` in the address bar of the browser. The file is uploaded.

Do not forget to save the code:

```
git add .
git commit -m "add upload object feature"
```

8. **Add a style**

   You can add some styles (CSS) to refine the page outlook.

   Download *bootstrap*. Unzip the package and then copy `bootstrap.min.css` to `app/assets/stylesheets/`.

   Open `app/views/layouts/application.html.erb` and modify the `yield` line as follows:

```
  <div id="main">
    <%= yield %>
  </div>
```

   In this way, each page is added with a `<div>` with main as the ID. Then modify `app/assets/stylesheets/application.css` and add the following content:

```
body {
    text-align: center;
}

div#main {
    text-align: left;
    width: 1024px;
    margin: 0 auto;
}
```

   The preceding modification displays the body content of a webpage in a centered layout. The new style makes the page layout more elegant.

   So far, a simple demo has been completed. For a complete demo, see *Alibaba Cloud OSS Rails Demo*.

14 Media-C

14.1 Preface

Introduction

You can store the video captured by the camera to the on-cloud OSS. However, following are a few limitations:

- **Owing to the high risk of the AccessKey ID and AccessKeySecret leak, both cannot be stored permanently on the device.**
- **The device only allows file uploads and downloads and does not allow administrative permissions such as deletion or configuration changes.**
- **Webpages are available to manage your own videos.**
- **Precise control over the device permission is provided.**
- **The permission for the device has a validity period and thus, the device cannot be held permanently.**
- **The video files output by the camera is expected to be viewed directly through the HLS protocol.**

To provide solutions to the preceding problems, we have launched the OSS MEDIA C SDK built, based on the OSS C SDK version that offers better and convenient solutions for the audio and video industry.

- **This article describes how to install and use the OSS MEDIA C SDK and is applicable to the OSS MEDIA C SDK V2.0.1.**
- **This article assumes that you have activated Alibaba Cloud OSS and created an AccessKey ID and AccessKeySecret.**
- **If you have not activated or do not know about Alibaba Cloud OSS, log on to the *OSS Product Homepage* for more information.**
- **If you have not created an AccessKey ID and AccessKeySecret, go to *Alibaba Cloud Access Key Management* to create the AccessKey.**

Download the SDK

- **Linux (02/03/2016) V2.0.2**
  - **SDK package:** *aliyun-media-c-sdk-2.0.2.tar.gz*
  - **Source code:** *GitHub*
  - **Version iterations:** *Releases*

  Note:
  The OSS MEDIA C SDK does not support Windows.
- **Compatibility**

  SDKs of 1.x.x versions are not compatible with the list-related interfaces after the interfaces change, but are compatible with the other interfaces.

14.2 Installation

This topic describes how to install Media-C SDK.

Version dependency

- **Linux**

  OSS C SDK = 3.x.x
- **Windows**

  Not supported

Installation in Linux

- **Firstly, install the OSS C SDK. For installation steps, see** *Install the C SDK*.
- **You can** *Download SDK package* **or download** *Source Code*. **The downloaded package includes src, sample, and test directories and the CMakeList.txt file.**
- **Install to the system directory**
  - **If the OSS C SDK and its dependencies are all installed under the system directory (/usr/local/ or /usr/) and you want to install the OSS MEDIA C SDK to the system directory too, you can run the following command to compile and install the SDK:**

```
cmake .
make
make install
```

```
```

- After the preceding command is successfully run, the OSS MEDIA C SDK is installed to the /usr/local/ directory.

· Install to the custom directory (install the dependency package to the system directory)

- If the OSS C SDK and its dependencies are all installed under the system directory (/usr/local/ or /usr/), but you want to install the OSS MEDIA C SDK to a custom directory such as /home/user/aliyun/oss/install/, you can run the following command to compile and install the SDK:

```
    cmake . -DCMAKE_INSTALL_PREFIX=/home/user/aliyun/oss/install/
usr/local/
    make
    make install
```

- Once you run the preceding command, the OSS MEDIA C SDK is installed to the /home/user/aliyun/oss/install/usr/local/ directory.

· Install to the custom directory (install the dependency package to the custom directory)

- If the OSS C SDK or some dependency packages are installed to a custom directory, you cannot find the header files and library files of these packages by default, when you compile the OSS MEDIA C SDK. You must specify the path when you run the cmake command. For example, if you have installed the OSS C SDK to the /home/user/aliyun/oss/install/ directory, run the following command to compile and install the SDK:

```
    cmake . -DCMAKE_INSTALL_PREFIX=/home/user/aliyun/oss/install/
usr/local/ -DOSS_C_SDK_INCLUDE_DIR=/home/user/aliyun/oss/install
/usr/local/include/ -DOSS_C_SDK_LIBRARY=/home/user/aliyun/oss/
install/usr/local/lib/liboss_c_sdk.so
    make
    make install
```

- Once you run the preceding command, the OSS MEDIA C SDK is installed to the /home/user/aliyun/oss/install/usr/local/ directory.
- Names of other related parameters for other dependency packages include: APR_UTIL_LIBRARY, APR_LIBRARY, CURL_LIBRARY, APR_INCLUDEDIRS, APU_INCLUDEDIRS, OSS_C_SDK_INCLUDE_DIR, and CURL_INCLUDEDIRS.

- **Only compile and install the client SDK**
  - **The client SDK and the server SDK are installed at the same time by default. If you only need to compile and install the client SDK, run the following command:**

```
cmake . -DONLY_BUILD_CLIENT=ON
make
make install
```

  - **If you only need to compile and install the server, modify ONLY_BUILD _CLIENT to ONLY_BUILD_SERVER.**
  - **The testing example are compiled only when the client and server SDKs are compiled at the same time.**
- **Other compilation and installation methods and issues**
  - **Compile mode: Currently four types are supported, namely Debug, Release, MinSizeRef, and RelWithDebInfo. You can specify a compilation type using the parameter -DCMAKE_BUILD_TYPE. For example, if you want to use Debug mode for compilation, add the parameter -DCMAKE_BUILD_TYPE=Debug: cmake . -DCMAKE_BUILD_TYPE=Debug. The Release mode is adopted by default.**
    - **Debug: No code optimization is performed. GDB is supported. This mode is generally used for program debugging.**
    - **Release: More advanced optimization. This mode is generally applicable to the production environment.**
    - **MinSizeRef: This mode generates the smallest library file and is generally used in the embedded environment.**
    - **RelWithDebInfo: More advanced optimization approach. This mode carries debugging information and is generally used for the production environment.**
  - **If the "Targets may link only to libraries. CMake is dropping the item" warning is prompted during cmake execution, the cause is that the specified**

library path is incorrect. The library path must be specified to *.so, such as /path/to/xxx.so.

- If you want to use the static library of the OSS C SDK, specify the -DOSS_C_SDK_LIBRARY=/home/user/aliyun/oss/install/usr/local/lib/liboss_c_sdk_static.a during cmake execution. Other libraries are similar.
- If "CMake Error: The following variables are used in this project, but they are set to NOTFOUND." is prompted when you run the cmake, this is because the corresponding library cannot be found in the default path and must be specified. See `Install to a custom directory`.

14.3 Initialization

This topic describes how to initialize the Media-C SDK environment.

Determine an endpoint

An endpoint is the domain of Alibaba Cloud OSS in a region. It currently supports two formats.

Endpoint type	Description
OSS region address	Address of the region where an OSS bucket is located. For more information about the endpoints in various regions, see *Regions and endpoints*.
User-defined domain name	Domain name defined by the user, with the CNAME directing to the OSS domain.

- OSS region address

  You can use any of the following methods to search for an endpoint mapped to the address of the region where an OSS bucket is located:

  - You can query the mapping relationship between the endpoint and the region. For more information, see *Regions and endpoints*.
  - You can log on to *Alibaba Cloud OSS Console*, open the Bucket Overview page, and find the suffix of the bucket domain. For example, the suffix `oss-cn-hangzhou.aliyuncs.com` of the bucket domain `bucket-1.oss-cn-hangzhou.aliyuncs.com` is the endpoint of the bucket on the Internet.

Configure an AccessKey

To access Alibaba Cloud OSS, you must have a valid pair of AccessKeys (including an AccessKeyID and AccessKeySecret) for signature authentication. Do the steps in the following links to obtain the AccessKeys:

- *Register an Alibaba Cloud account*
- *Apply for an AccessKey*

Use the RAM and STS services

Only temporary authorization is required for uploading files to or downloading files from the device terminal (such as a mobile phone, tablet, or camera), instead of permanent authorization. In this scenario, the RAM and STS services are used.

1. **Log on to the *RAM console* and activate the RAM service.**
2. **Create a role. You can grant the AliyunOSSFullAccess and AliyunSTSAssumeRoleAccess permissions to this role.**
3. **After successful creation, an ARN in the role details in a format similar to acs:ram:xxxx:role/yyyy. This is the role_arn. It is used to get a temporary token in subsequent steps.**

Initialization

- **For initialization of the client, see *Client operations*.**
- **For initialization of the server, see *Server operations*.**

14.4 Client operations

OSS MEDIA C SDK is divided into three parts: the client, the server, and the HLS. We mainly cover operations related to the client. For other operations, see the following documents:

Interface

Operation interfaces related with the client are all placed in oss_media_file.h. The following interfaces are currently provided:

- **oss_media_file_open**
- **oss_media_file_stat**
- **oss_media_file_tell**
- **oss_media_file_seek**

- **oss_media_file_read**
- **oss_media_file_write**
- **oss_media_file_close**

Next, the functions and use notes for various interfaces are provided in detail.

Basic structs

```
/**
 * Metadata of the OSS MEDIA FILE, including file length, location,
and type
 */
typedef struct {
    int64_t length;
    int64_t pos;
    char    *type;
} oss_media_file_stat_t;


/**
 * Attributes of the OSS MEDIA FILE
 */
typedef struct oss_media_file_s {
    void   *ipc;
    char   *endpoint;
    int8_t is_cname;
    char   *bucket_name;
    char   *object_key;
    char   *access_key_id;
    char   *access_key_secret;
    char   *token;
    char   *mode;
    oss_media_file_stat_t _stat;

    time_t expiration;
    auth_fn_t auth_func;
} oss_media_file_t;
```

Note:

- **type: The file type: normal or appendable.**
- **ipc: Set the unique identifier of the device. This attribute can be used in auth func as per requirement, and ignored if it is not used.**
- **endpoint: Such as oss-cn-hangzhou.aliyuncs.com.**
- **is_cname: Whether CNAME has been enabled.**
- **bucket_name: Name of the OSS bucket.**
- **object_key: Name of the OSS object.**
- **access_key_id: The AccessKey ID provided by Alibaba Cloud for access control . Here two modes are available: 1. The AccessKey ID of the master account or**

the subaccount. In this case, the token that follows must be set to NULL. 2. The temporary AccessKey ID obtained through get_token.

- **access_key_secret: The AccessKey Secret provided by Alibaba Cloud for access control. The two available modes are: 1. The permanent AccessKey Secret of the master account or the subaccount. In this case, the token that follows must be set to NULL. 2. The temporary AccessKey Secret obtained through get_token. The access_key_id and access_key_secret must use the same mode at the same time.**
- **token: If the permanent AccessKey ID and AccessKey Secret of the master account or the subaccount are used at the same time, this attribute must be set to NULL. If you use a terminal device to upload or download resources, the app server needs to grant the temporary access permission to the terminal device. You can get the temporary access_key_id, access_key_secret and token through the get_token on the server. If you set token to a non-NULL value, the system expects you to use a temporary AccessKey ID, a temporary AccessKey Secret and a temporary token. If you do not want to use a temporary token, please set this attribute to NULL.**
- **expiration: The time when the authorization fails. After the first authorization, the next authorization is performed only after the expiration time has passed.**
- **auth: The authorization function. You must implement a function and grant values to the host, bucket, and token values within the function.**
- **mode: The read-write mode.**

Initialization

```
/**
 *  @brief  Initialize the OSS media
 *  @note   This interface should be called first when the program
starts to initialize the OSS MEDIA C SDK.
 *  @return:
 *      If 0 is returned, it indicates the operation was successful.
 *      Otherwise, it indicates an error has occured. Possible causes
include: insufficient memory, or the APR or CURL versions are too low.
 */
int oss_media_init(aos_log_level_e log_level);
```

Note:

For the complete code, see *GitHub*.

Destruction

```
/**
```

```
 *  @brief  Destroy the OSS meida
 *  @note   This interface should be called at last when the program
ends to destroy the OSS MEDIA C SDK.
 */
void oss_media_destroy();
```

Note:

For the complete code, see *GitHub*.

Open an object

```
/**
 *  @brief  Open an OSS object
 *  @param[in]  Name of the bucket that stores objects in bucket_name
oss
 *  @param[in]  Name of the object in object_key  oss
 *  @param[in]  mode:
 *      'r': Read mode
 *      'w': Overwrite mode
 *      'a': Append write mode
 *      notes: Not allowed to be used in combination
 *  @param[in]  auth_func    The authorization function to set
access_key_id/access_key_secret
 *  @return:
 *      If NULL is returned, it indicates success. Otherwise, it
indicates failure.
 */
oss_media_file_t* oss_media_file_open(char *bucket_name,
                                      char *object_key,
                                      char *mode,
                                      auth_fn_t auth_func);
```

Note:

- **mode: Read-only, overwrite, and append write modes are supported. Combination mode is not supported.**
  - **Read mode: Once you open the object, part of the object content is read-only once. Multiple reads are generally required to read the full content.**
  - **Overwrite mode: After you open the object, the object content is written at a time, in full. Multiple writes are supported, but the latest content overwrites the earlier content.**
  - **Append write: After the object is opened, you can write content to the object in the append mode multiple times.**
- **For the complete code, see *GitHub*.**

Close an object

```
/**
 *  @brief  Close the OSS object
```

```
 */
void oss_media_file_close(oss_media_file_t *file);
```

Note:

For the complete code, see *GitHub*.

Write an object

```
/**
 *  @brief  Write an object to the OSS.
 *  @return:
 *      If the request is successful, the size of the written data
will be returned. If "-1" is returned, it indicates the data write
failed.
 */
int64_t oss_media_file_write(oss_media_file_t *file, const void *buf,
int64_t nbyte);
```

Example project:

```
/* Authorization function */
static void auth_func(oss_media_file_t *file) {
    file->endpoint = "your endpoint";
    file->is_cname = 0;
    file->access_key_id = "Access key ID or temporary access key ID
provided by Alibaba Cloud";
    file->access_key_secret = "Access key secret or temporary access
key secret provided by Alibaba Cloud";
    file->token = "Get the temporary token through the get_token
interface";

    /* Effective time of this authorization */
    file->expiration = time(NULL) + 300;
}

static void write_file() {
    oss_clean(g_filename);

    int64_t write_size;
    oss_media_file_t *file;
    char *bucket_name = "<your bucket name>";
    char *key = "<your object key>";
    char *content = "aliyun oss media c sdk";

    /* Open an object */
    file = oss_media_file_open(bucket_name, key, "w", auth_func);
    if (! file) {
        printf("open media file failed\n");
        return;
    }

    /* Write an object */
    write_size = oss_media_file_write(file, content, strlen(content));
    if (-1 ! = write_size) {
         printf("write %" PRId64 " bytes succeeded\n", write_size);
    } else {
        oss_media_file_close(file);
        printf("write failed\n");
        return;
```

```
    }

    /* Close the object and release the resource */
    oss_media_file_close(file);
}
```

Note:

- **In the example, the `overwrite ("w")` mode is used to open an object. If multiple writes are performed, the later write overwrites the previous write. If you want to activate append write, specify the `append write ("a")` mode for opening the object.**
- **For the complete code, see *GitHub*.**

Read an object

```
/**
 *  @brief  Read the data of a fixed number of nbytes.
 *  @note   The size of buf should be greater than or equal to nbyte
 + 1.
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may have
 occurred. Possible causes of the failure include: the object was not
opened in the "read-only" mode, the object cannot connect to the OSS,
or the object has no permission to read the OSS.
 */
int64_t oss_media_file_read(oss_media_file_t *file, void *buf, int64_t
 nbyte);
```

Example project:

```
    int ntotal, nread, nbuf = 16;
    char buf[nbuf];
    char *content;
    char *bucket_name = "<your bucket name>";
    char *key = "<your object key>";

    oss_media_file_t *file;

    /* Open the object */
    file = oss_media_file_open(bucket_name, key, "r", auth_func);
    if (! file) {
        printf("open media file failed\n");
        return;
    }

    /* Read the object */
    content = malloc(stat.length + 1);
    ntotal = 0;
    while ((nread = oss_media_file_read(file, buf, nbuf)) > 0) {
        memcpy(content + ntotal, buf, nread);
        ntotal += nread;
    }
    content[ntotal] = '\0';
```

```
    /* Close the object */
    oss_media_file_close(file);
    free(content);

    printf("oss media c sdk read object succeeded\n");
}
```

Note:

For the complete code, see *GitHub*.

Manage objects

OSS MEDIA FILE also supports tell, seek, and stat operations.

```
/**
 *  @brief  Get the location of the OSS MEDIA FILE
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may have
 occurred. Possible causes of the failure include: the object was not
opened in the "read-only" mode, the object cannot connect to the OSS,
or the object has no permission to read the OSS.
 */
int64_t oss_media_file_tell(oss_media_file_t *file);

/**
 *  @brief  Set the OSS MEDIA FILE pointer to the specified location.
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may have
 occurred. Possible causes of the failure include: the object was not
opened in the "read-only" mode, the object cannot connect to the OSS,
or the object has no permission to read the OSS.
 */
int64_t oss_media_file_seek(oss_media_file_t *file, int64_t offset);

/**
 *  @brief Get the metadata of the OSS MEDIA FILE.
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may have
 occurred. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission to read the OSS.
 */
int oss_media_file_stat(oss_media_file_t *file, oss_media_file_stat_t
 *stat);
```

Example project:

```
static void seek_tell_stat_file() {
    int ntotal, nread, nbuf = 16;
    char buf[nbuf];
    char *bucket_name = "<your bucket name>";
    char *key = "<your object key>";

    oss_media_file_t *file;

    /* Open the object */
```

```
    file = oss_media_file_open(bucket_name, key, "r", auth_func);
    if (! file) {
        printf("open media file failed\n");
        return;
    }

    /* Get the meta information of the object */
    oss_media_file_stat_t stat;
    if (0 ! = oss_media_file_stat(file, &stat)) {
        oss_media_file_close(file);
        oss_media_file_free(file);

        printf("stat media file[%s] failed\n", file->object_key);
        return;
    }

    printf("file [name=%s, length=%ld, type=%s]",
           file->object_key, stat.length, stat.type);

    /* tell */
    printf("file [position=%" PRId64 "]", oss_media_file_tell(file));

    /* seek */
    oss_media_file_seek(file, stat.length / 2);

    /* Close the object */
    oss_media_file_close(file);

    printf("oss media c sdk seek object succeeded\n");
}
```

Note:

For the complete code, see *GitHub*.

14.5 Server operations

OSS MEDIA C SDK has three parts: the client, the server, and the HLS. Next we introduce the operations related to the server. For other operations, see the following chapters.

Interface

Operation interfaces related to the server are all placed in oss_media.h. Currently the following interfaces are provided:

- **oss_media_create_bucket**
- **oss_media_delete_bucket**
- **oss_media_create_bucket_lifecycle**
- **oss_media_get_bucket_lifecycle**
- **oss_media_delete_bucket_lifecycle**
- **oss_media_delete_file**

- **oss_media_list_files**
- **oss_media_get_token**
- **oss_media_get_token_from_policy**

The Functions and notes of usage of various interfaces are as follows:

Basic structs

```
typedef struct oss_media_config_s {
    char *endpoint;
    int is_cname;
    char *access_key_id;
    char *access_key_secret;
    char *role_arn;
} oss_media_config_t;

typedef struct oss_media_files_s {
    char *path;
    char *marker;
    int  max_size;

    char *next_marker;
    int  size;
    char **file_names;
} oss_media_files_t;
```

Note:

- **endpoint: For example, oss-cn-hangzhou.aliyuncs.com.**
- **is_cname: Whether CNAME has been enabled.**
- **access_key_id: The AccessKey ID for access control provided by Alibaba Cloud.**
- **access_key_secret: The AccessKey Secret for access control provided by Alibaba Cloud.**
- **role_arn: The ARN of the role created in Alibaba Cloud RAM. The value is available in RAM console > Manage Roles > click the name of a role > Basic Information > ARN. The value format is similar to: acs:ram::xxxxxx:role/yyyy . If no role is available, you must create a new role and grant the AliyunOSSF ullAccess and AliyunSTSAssumeRoleAccess permissions to the role.**
- **marker: Set the returned results to begin from the first entry after the Marker in alphabetical order.**
- **max_size: Set the maximum number returned. The value cannot exceed 1,000.**
- **next_marker: The start position of the next execution.**

Initialization

```
/**
 *  @brief  Initialize the OSS media
 *  @note   This interface must be called first when the program
starts to initialize the OSS MEDIA C SDK.
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, it indicates that an error has occured. The
possible causes of the failure include: insufficient memory, or the
APR or CURL versions are too low.
 */
int oss_media_init();
```

Note:

For complete code, see *GitHub*.

Destruction

```
/**
 *  @brief  Destroy the OSS meida
 *  @note   This interface must be called at last when the program
ends to destroy the OSS MEDIA C SDK.
 */
void oss_media_destroy();
```

Note:

For the complete code, see *GitHub*.

Create a bucket

```
/**
 *  @brief  Create a new bucket
 *  @paramp[in] oss_media_acl_t
 *     OSS_ACL_PUBLIC_READ     Public read and private write
 *     OSS_ACL_PUBLIC_READ  Public read and private write
 *     OSS_ACL_PUBLIC_READ_WRITE Public read-write
 * @ Return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission.
 */
int oss_media_create_bucket(oss_media_config_t *config, const char *
bucket_name, oss_media_acl_t acl);
```

Example project:

```
static void init_media_config(oss_media_config_t *config) {
    config->endpoint = "your endpoint";
    config->access_key_id = "The access key ID provided by Alibaba
Cloud";
    config->access_key_secret = "The access key secret provided by
Alibaba Cloud";
```

```
    config->role_arn = "The role ARN provided by Alibaba Cloud RAM";
    config->is_cname = 0;
}

void create_bucket() {
    int ret;
    char *bucket_name;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Create the bucket */
    ret = oss_media_create_bucket(&config, bucket_name, OSS_ACL_PR
IVATE);

    if (0 == ret) {
        printf("create bucket[%s] succeeded.\n", bucket_name);
    } else {
        printf("create bucket[%s] failed.\n", bucket_name);
    }
}
```

Note:

For the complete code, see *GitHub*.

Delete a bucket

```
/*
 *  @brief  Delete the bucket
 * @return
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission.
 */
int oss_media_delete_bucket(oss_media_config_t *config, const char *
bucket_name);
```

Example project:

```
void delete_bucket() {
    int ret;
    char *bucket_name;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Delete the bucket */
    ret = oss_media_delete_bucket(&config, bucket_name);

    if (0 == ret) {
        printf("delete bucket[%s] succeeded.\n", bucket_name);
    } else {
        printf("delete bucket[%s] failed.\n", bucket_name);
```

```
    }
}
```

Note:

For the complete code, see *GitHub*.

Create lifecycle rules for a bucket

```
/**
 *  @brief  Create lifecycle rules for the bucket.
 *  @note   These rules can control the time for automatically
deleting objects.
 * @return
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission.
 */
int oss_media_create_bucket_lifecycle(oss_media_config_t *config,
const char *bucket_name, oss_media_lifecycle_rules_t *rules);
```

Example project:

```
void create_bucket_lifecycle() {
    int ret;
    char *bucket_name;
    oss_media_lifecycle_rules_t *rules;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Create lifecycle rules */
    rules = oss_media_create_lifecycle_rules(2);
    oss_media_lifecycle_rule_t rule1;
    rule1.name = "example-1";
    rule1.path = "/example/1";
    rule1.status = "Enabled";
    rule1.days = 1;
    oss_media_lifecycle_rule_t rule2;
    rule2.name = "example-2";
    rule2.path = "/example/2";
    rule2.status = "Disabled";
    rule2.days = 2;

    rules->rules[0] = &rule1;
    rules->rules[1] = &rule2;

    /* Set lifecycle rules of the bucket */
    ret = oss_media_create_bucket_lifecycle(&config,
            bucket_name, rules);

    if (0 == ret) {
        printf("create bucket[%s] lifecycle succeeded.\n", bucket_name
);
    } else {
        printf("create bucket[%s] lifecycle failed.\n", bucket_name);
    }
```

```
    /* Release the resources. */
    oss_media_free_lifecycle_rules(rules);
}
```

Note:

Example code can be found at *GitHub*.

Get lifecycle rules for the bucket

```
/**
 *  @brief  Get lifecycle rules for the bucket
 * @return
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may have
 occurred. Possible causes of the failure include: the object cannot
connect to the OSS.
 */
int oss_media_get_bucket_lifecycle(oss_media_config_t *config, const
char *bucket_name, oss_media_lifecycle_rules_t *rules);
```

Example project:

```
void get_bucket_lifecycle() {
    int ret, i;
    char *bucket_name;
    oss_media_lifecycle_rules_t *rules;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Get lifecycle rules */
    rules = oss_media_create_lifecycle_rules(0);
    ret = oss_media_get_bucket_lifecycle(&config, bucket_name, rules);

    if (0 == ret) {
        printf("get bucket[%s] lifecycle succeeded.\n", bucket_name);
    } else {
        printf("get bucket[%s] lifecycle failed.\n", bucket_name);
    }

    for (i = 0; i < rules->size; i++) {
        printf(">>>> rule: [name:%s, path:%s, status=%s, days=%d]\n",
               rules->rules[i]->name, rules->rules[i]->path,
               rules->rules[i]->status, rules->rules[i]->days);
    }

    /* Release the resources. */
    oss_media_free_lifecycle_rules(rules);
}
```

Note:

For the complete code, see *GitHub*.

Delete lifecycle rules for a bucket

```
/**
 *  @brief  Delete the bucket lifecycle rules
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission.
 */
int oss_media_delete_bucket_lifecycle(oss_media_config_t *config,
const char *bucket_name);
```

Example project:

```
void delete_bucket_lifecycle()
{
    int ret;
    char *bucket_name;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Delete lifecycle rules */
    ret = oss_media_delete_bucket_lifecycle(&config, bucket_name);

    if (0 == ret) {
        printf("delete bucket[%s] lifecycle succeeded.\n", bucket_name
);
    } else {
        printf("delete bucket[%s] lifecycle failed.\n", bucket_name);
    }
}
```

Note:

For the complete code, see *GitHub*.

Delete an object

```
/**
 *  @brief  Delete a specific object in the bucket
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission.
 */
int oss_media_delete_file(oss_media_config_t *config, const char *
bucket_name, const char *key);
```

Example project:

```
void delete_file() {
    int ret;
    oss_media_config_t config;
    char *file;
```

```
    char *bucket_name;

    /* Initialize the variable */
    file = "oss_media_file";
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Delete an object */
    ret = oss_media_delete_file(&config, bucket_name, file);

    if (0 == ret) {
        printf("delete file[%s] succeeded.\n", file);
    } else {
        printf("delete file[%s] lifecycle failed.\n", file);
    }
}
```

Note:

For the complete code, see *GitHub*.

List objects

```
/**
 *  @brief  List objects in a specific bucket
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the OSS, or the object has no permission.
 */
int oss_media_list_files(oss_media_config_t *config, const char *
bucket_name, oss_media_files_t *files);
```

Example project:

```
void list_files() {
    int ret, i;
    char *bucket_name;
    oss_media_files_t *files;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>>";
    init_media_config(&config);

    files = oss_media_create_files();
    files->max_size = 50;

    /* List objects */
    ret = oss_media_list_files(&config, bucket_name, files);

    if (0 == ret) {
        printf("list files succeeded.\n");
    } else {
        printf("list files lifecycle failed.\n");
    }

    for (i = 0; i < files->size; i++) {
        printf(">>>>file name: %s\n", files->file_names[i]);
```

```
    }

    /* Release the resources */
    oss_media_free_files(files);
}
```

Note:

For the complete code, see *GitHub*.

Get the temporary token

```
/*
 *  @brief  Get the temporary token
 *  @param[in]:
      mode:
 *      'r': Read mode
 *      'w': Overwrite mode
 *      'a': Append write mode
 *    expiration: The valid period of the temporary token. The value
ranges from 15 minutes to one hour. The unit is seconds.
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the Alibaba Cloud STS, the RAM or STS is not activated, no
role is created and the parameter is invalid.
 */
int oss_media_get_token(oss_media_config_t *config,
                        const char *bucket_name,
                        const char *path,
                        const char *mode,
                        int64_t expiration,
                        oss_media_token_t *token);
```

Example project:

```
void get_token() {
    int ret;
    char *bucket_name;
    oss_media_token_t token;
    oss_media_config_t config;

    /* Initialize the variable */
    bucket_name = "<your bucket name>";
    init_media_config(&config);

    /* Get the temporary token */
    ret = oss_media_get_token(&config, bucket_name, "/*", "rwa",
                              3600, &token);

    if (0 == ret) {
        printf("get token succeeded, access_key_id=%s, access_key
_secret=%s, token=%s\n",
               token.tmpAccessKeyId, token.tmpAccessKeySecret, token.
securityToken);
    } else {
        printf("get token failed.\n");
    }
```

```
}
```

Note:

For the complete code, see *GitHub*.

Get the token through a custom policy

```
/**
 *  @brief  Get the token through the specific custom policy
 *  @param[in]:
 *      expiration:  The valid period of the temporary token. The
value ranges from 15 minutes to one hour. The unit is second.
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error may
 occur. Possible causes of the failure include: the object cannot
connect to the Alibaba Cloud STS, the RAM or STS is not activated, no
role is created and the parameter is invalid.
 */
int oss_media_get_token_from_policy(oss_media_config_t *config,
                                    const char *policy,
                                    int64_t expiration,
                                    oss_media_token_t *token);
```

Example project:

```
void get_token_from_policy() {
    int ret;
    oss_media_token_t token;
    char *policy;
    oss_media_config_t config;

    /* Initialize the variable */
    init_media_config(&config);

    /* Set the custom policy */
    policy = "{\"Version\":\"1\",\"Statement\":[{\"Effect\":\"Allow\",
 "
             "\"Action\":\"*\", \"Resource\":\"*\"}]}";

    /* Get the token */
    ret = oss_media_get_token_from_policy(&config, policy, 3600, &
token);

    if (0 == ret) {
        printf("get token succeeded, access_key_id=%s, access_key
_secret=%s, token=%s\n",
               token.tmpAccessKeyId, token.tmpAccessKeySecret, token.
securityToken);
    } else {
        printf("get token failed.\n");
    }
}
```

Note:

For the complete code, see *GitHub*.

14.6 HLS basic interfaces

The OSS MEDIA C SDK client supports packing the received H.264 or AAC format data into TS or M3U8 format and then write the data to OSS. You can view the video and listen to the audio through the corresponding M3U8 address.

Interface

HLS-related basic interfaces are all located in the oss_media_hls.h. Currently the provided interfaces include:

- **oss_media_hls_open**
- **oss_media_hls_write_frame**
- **oss_media_hls_begin_m3u8**
- **oss_media_hls_write_m3u8**
- **oss_media_hls_end_m3u8**
- **oss_media_hls_flush**
- **oss_media_hls_close**

The functions and notes of usage of various interfaces are as follows:

- **Basic structs**

```
/**
 *  OSS MEDIA HLS FRAME metadata
 */
typedef struct oss_media_hls_frame_s {
    stream_type_t stream_type;
    frame_type_t frame_type;
    uint64_t pts;
    uint64_t dts;
    uint32_t continuity_counter;
    uint8_t  key:1;
    uint8_t *pos;
    uint8_t *end;
} oss_media_hls_frame_t;

/**
 *  OSS MEDIA HLS descriptive information
 */
typedef struct oss_media_hls_options_s {
    uint16_t video_pid;
    uint16_t audio_pid;
    uint32_t hls_delay_ms;
    uint8_t encrypt:1;
    char    key[OSS_MEDIA_HLS_ENCRYPT_KEY_SIZE];
    file_handler_fn_t handler_func;
    uint16_t pat_interval_frame_count;
} oss_media_hls_options_t;

/**
 *  OSS MEDIA HLS FILE descriptive information
```

```
 */
typedef struct oss_media_hls_file_s {
    oss_media_file_t *file;
    oss_media_hls_buf_t *buffer;
    oss_media_hls_options_t options;
    int64_t frame_count;
} oss_media_hls_file_t;
```

Note:

- **stream_type: Stream type. Currently the st_h264 and st_aac types are supported.**
- **frame_type: Frame type. Currently the ft_non_idr, ft_idr, ft_sei, ft_sps, ft_pps and ft_aud are supported.**
- **pts: The display timestamp.**
- **dts: The decoding timestamp.**
- **continuity_counter: The incremental counter. The value ranges from 0 to 15, but the starting value is not necessarily 0. The values must be consecutive.**
- **key: Whether or not it is the key frame.**
- **pos: The starting position (inclusive) of the current frame data.**
- **end: The ending position (exclusive) of the current frame data.**
- **video_pid: The video PID.**
- **audio_pid: The audio PID.**
- **hls_delay_ms: The display latency (milliseconds).**
- **encrypt: Whether to use AES-128 encryption. Currently AES-128 encryption is not supported.**
- **key: The key when encryption is enabled. Currently this feature is not supported.**
- **handler_func: The callback function of the object operation.**
- **pat_interval_frame_count: The number of frames between every two inserted PAT or every two inserted MAT tables.**

- **Open an HLS file**

```
/**
 *  @brief  Open an OSS HLS file
 *  @param[in]  Name of the bucket that stores objects in bucket_nam
e oss
 *  @param[in]  Name of the object in object_key  oss
 *  @param[in]  auth_func   The authorization function to set
access_key_id/access_key_secret
 * @return
 *      If NULL is returned, it indicates success. Otherwise, it
indicates failure.
```

```
 */
oss_media_hls_file_t* oss_media_hls_open(char *bucket_name, char *
object_key, auth_fn_t auth_func);
```

Note:

For the complete code, see *GitHub*.

- **Close an HLS file**

```
/**
 *  @brief  Close the OSS HLS file
 */
int oss_media_hls_close(oss_media_hls_file_t *file);
```

Note:

For the complete code, see *GitHub*.

- **Write an HLS file**

```
/**
 *  @brief  Write a frame of data in H.264 or AAC format to the OSS
 *  @param[in]  frame         A frame of data in H.264 or AAC format
 *  @param[out] file          hls file
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, it indicates an error has occurred.
 */
int oss_media_hls_write_frame(oss_media_hls_frame_t *frame,
oss_media_hls_file_t *file);
```

Example project:

```
static void write_frame(oss_media_hls_file_t *file) {
    oss_media_hls_frame_t frame;
    FILE    *file_h264;
    uint8_t *buf_h264;
    int     len_h264, i;
    int     cur_pos = -1;
    int     last_pos = -1;
    int     video_frame_rate = 30;
    int     max_size = 10 * 1024 * 1024;
    char    *h264_file_name = "/path/to/example.h264";

    /* Read the H.264 file */
    buf_h264 = calloc(max_size, 1);
    file_h264 = fopen(h264_file_name, "r");
    len_h264 = fread(buf_h264, 1, max_size, file_h264);

    /* Initialize the frame struct */
    frame.stream_type = st_h264;
    frame.pts = 0;
    frame.continuity_counter = 1;
    frame.key = 1;

    /* Tranverse the H.264 data and extract the data of every frame
, and then write the data into the OSS */
    for (i = 0; i < len_h264; i++) {
```

```
        /*  Determine whether the current location is the starting
point of the data of the next frame, that is, the end of the current
 frame */
        if ((buf_h264[i] & 0x0F)==0x00 && buf_h264[i+1]==0x00
            && buf_h264[i+2]==0x00 && buf_h264[i+3]==0x01)
        {
            cur_pos = i;
        }

 /* If the data of a complete frame can be obtained, call the
interface to convert the data into the HLS format and write the data
 into the OSS */
        if (last_pos ! = -1 && cur_pos > last_pos) {
            frame.pts += 90000 / video_frame_rate;
            frame.dts = frame.pts;

            frame.pos = buf_h264 + last_pos;
            frame.end = buf_h264 + cur_pos;

            oss_media_hls_write_frame(&frame, file);
        }

        last_pos = cur_pos;
    }

    /* Close the object and release the resource */
    fclose(file_h264);
    free(buf_h264);
}
```

Note:

- **For the complete code, see *GitHub*.**
- **If Access Unit Delimiter NALs (00 00 00 01 09 xx) is missing in the H.264 data, this NAL is required. Otherwise the audio or video cannot be played on iPad, iPhone, or Safari.**
- **H.264 frames are separated through 0xX0, 0x00, 0x00, and 0x01. AAC frames are separated through 0xFF and 0x0X.**
- **When the current frame is a key frame, frame. The key needs to be set to 1.**

- **Write an M3U8 object**

```
/**
 *  @brief  Write the header data of the M3U8 object
 *  @param[in]  max_duration  The maximum duration of the TS object
 *  @param[in]  sequence      The starting number of the TS object
 *  @param[out] file          M3U8 file
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error has
occured.
 */
void oss_media_hls_begin_m3u8(int32_t max_duration,
                              int32_t sequence,
                              oss_media_hls_file_t *file);
```

```
/**
 *  @brief  Write the data of the M3U8 object
 *  @param[in]  size          Number of the M3U8 items
 *  @param[in]  m3u8          Details of the M3U8 item
 *  @param[out] file          M3U8 file
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, if "-1" is returned, it indicates an error has
occured.
 */
int oss_media_hls_write_m3u8(int size,
                             oss_media_hls_m3u8_info_t m3u8[],
                             oss_media_hls_file_t *file);

/**
 *  @brief  Write data such as the M3U8 object terminator
 *  @param[out] file          M3U8 file
 */
void oss_media_hls_end_m3u8(oss_media_hls_file_t *file);
```

Example project:

```
static void write_m3u8() {
    char *bucket_name;
    char *key;
    oss_media_hls_file_t *file;

    bucket_name = "<your bucket name>";
    key = "<your m3u8 file name>";

    /* Open an HLS object to write the M3U8-format data. The object
name must end with ".m3u8" */
    file = oss_media_hls_open(bucket_name, key, auth_func);
    if (file == NULL) {
        printf("open m3u8 file[%s] failed.", key);
        return;
    }

    /* Construct the information of three TS objects */
    oss_media_hls_m3u8_info_t m3u8[3];
    m3u8[0].duration = 9;
    memcpy(m3u8[0].url, "video-0.ts", strlen("video-0.ts"));
    m3u8[1].duration = 10;
    memcpy(m3u8[1].url, "video-1.ts", strlen("video-1.ts"));

    /* Write to the M3U8 object
    oss_media_hls_begin_m3u8(10, 0, file);
    oss_media_hls_write_m3u8(2, m3u8, file);
    oss_media_hls_end_m3u8(file);

    /* Close the HLS object */
    oss_media_hls_close(file);

    printf("write m3u8 to oss file succeeded\n");
}
```

Note:

- **M3U8 V3 is currently used.**

- In a recording scenario, the oss_media_hls_end_m3u8(file) interface is called at the end of the recording to write a terminator. Otherwise the video cannot be played. In a live video scenario, this interface cannot be called.
- For the complete example code, see *GitHub*.
- You can view the effect in the example project.
- You can watch the video through *VLC Player*, and through Safari directly on iPhone, iPad, and Mac devices.

14.7 HLS encapsulated interfaces

The OSS MEDIA C SDK client can pack the received H.264 or AAC format data into the TS or M3U8 format and then write the data into the OSS. Apart from basic interfaces, the SDK also provides encapsulated recording and live video interfaces.

Interface

HLS-related encapsulated interfaces are all located in the `oss_media_hls_stream.h`. Currently the following interfaces are provided:

- **oss_media_hls_stream_open**
- **oss_media_hls_stream_write**
- **oss_media_hls_stream_close**

The functions and notes of usage of various interfaces are mentioned as follows:

- **Basic structure**

```
/**
 * Information about OSS MEDIA HLS STREAM OPTIONS
 */
typedef struct oss_media_hls_stream_options_s {
    int8_t is_live;
    char *bucket_name;
    char *ts_name_prefix;
    char *m3u8_name;
    int32_t video_frame_rate;
    int32_t audio_sample_rate;
    int32_t hls_time;
    int32_t hls_list_size;
} oss_media_hls_stream_options_t;

/**
 * Information about OSS MEDIA HLS STREAM
 */
typedef struct oss_media_hls_stream_s {
    const oss_media_hls_stream_options_t *options;
    oss_media_hls_file_t *ts_file;
    oss_media_hls_file_t *m3u8_file;
    oss_media_hls_frame_t *video_frame;
```

```
    oss_media_hls_frame_t *audio_frame;
    oss_media_hls_m3u8_info_t *m3u8_infos;
    int32_t ts_file_index;
    int64_t current_file_begin_pts;
    int32_t has_aud;
    aos_pool_t *pool;
} oss_media_hls_stream_t;
```

Note:

- **is_live: Indicates whether the video is in live mode. When the video is in live mode, the M3U8 object only contains the information of the last few TS objects.**
- **bucket_name: Indicates the names of the buckets that store the HLS files.**
- **ts_name_prefix: Indicates the prefix of the TS object name.**
- **m3u8_name: Indicates the name of the M3U8 object.**
- **video_frame_rate: Indicates the frame rate of the video data.**
- **audio_sample_rate: Indicates the sample rate of the audio data.**
- **hls_time: Indiicates the maximum duration of each TS object.**
- **hls_list_size: Indicates the number of TS objects retained in the M3U8 object in live video mode.**

- **Open an HLS stream object**

```
/**
 *  @brief  Open an OSS HLS file
 *  @param[in]  auth_func   The authorization function to set
access_key_id/access_key_secret
 *  @param[in]  options     Configuration information
 *  @return:
 *      If NULL is returned, it indicates success. Otherwise, it
indicates failure.
 */
oss_media_hls_stream_t* oss_media_hls_stream_open(auth_fn_t
auth_func,
                         const oss_media_hls_stream_options_t *
options);
```

Note:

- **For the complete example code, see** *GitHub*
- .

- **Close an HLS stream object**

```
/**
 *  @brief  Close the HLS stream object
```

```
 */
int oss_media_hls_stream_close(oss_media_hls_stream_t *stream);
```

Note:

For the complete example code, see *GitHub*.

- **Write data into an HLS stream object**

```
/**
 *  @brief  Write the video and audio data
 *  @param[in]  video_buf Video data
 *  @param[in]  video_len Video data length. It can be 0
 *  @param[in]  audio_buf Audio data
 *  @param[in]  audio_len Audio data length. It can be 0
 *  @param[in]  stream    HLS stream
 *  @return:
 *      If 0 is returned, it indicates the operation is successful.
 *      Otherwise, it indicates an error has occurred
 */
int oss_media_hls_stream_write(uint8_t *video_buf,
                               uint64_t video_len,
                               uint8_t *audio_buf,
                               uint64_t audio_len,
                               oss_media_hls_stream_t *stream);
```

Example project:

```
static void write_video_audio_vod() {
    int ret;
    int max_size = 10 * 1024 * 1024;
    FILE *h264_file, *aac_file;
    uint8_t *h264_buf, *aac_buf;
    int h264_len, aac_len;

    oss_media_hls_stream_options_t options;
    oss_media_hls_stream_t *stream;

    /* Set the HLS stream parameter value */
    options.is_live = 0;
    options.bucket_name = "<your buckete name>";
    options.ts_name_prefix = "vod/video_audio/test";
    options.m3u8_name = "vod/video_audio/vod.m3u8";
    options.video_frame_rate = 30;
    options.audio_sample_rate = 24000;
    options.hls_time = 5;

    /* Open the HLS stream */
    stream = oss_media_hls_stream_open(auth_func, &options);
    if (stream == NULL) {
        printf("open hls stream failed.\n");
        return;
    }

    /* Create two buffers to store the audio and video data */
    h264_buf = malloc(max_size);
    aac_buf = malloc(max_size);
```

```
    /* Read a piece of video data and audio data and call the
interface to write data to the OSS */
    {
        h264_file = fopen("/path/to/video/1.h264", "r");
        h264_len = fread(h264_buf, 1, max_size, h264_file);
        fclose(h264_file);

        aac_file = fopen("/path/to/audio/1.aac", "r");
        aac_len = fread(aac_buf, 1, max_size, aac_file);
        fclose(aac_file);

        ret = oss_media_hls_stream_write(h264_buf, h264_len,
                aac_buf, aac_len, stream);
        if (ret ! = 0) {
            printf("write vod stream failed.\n");
            return;
        }
    }

    /* Read a piece of video data and audio data again and call the
interface to write data to the OSS */
    {
        h264_file = fopen("/path/to/video/2.h264", "r");
        h264_len = fread(h264_buf, 1, max_size, h264_file);
        fclose(h264_file);

        aac_file = fopen("/path/to/audio/1.aac", "r");
        aac_len = fread(aac_buf, 1, max_size, aac_file);
        fclose(aac_file);

        ret = oss_media_hls_stream_write(h264_buf, h264_len,
                aac_buf, aac_len, stream);
        if (ret ! = 0) {
            printf("write vod stream failed.\n");
            return;
        }
    }

    /* After the data write, close the HLS stream */
    ret = oss_media_hls_stream_close(stream);
    if (ret ! = 0) {
        printf("close vod stream failed.\n");
        return;
    }

    /* Release the resources */
    free(h264_buf);
    free(aac_buf);

    printf("convert H. 264 and aac to HLS vod succeeded\n");
}
```

Note:

- **Currently the recording and live video interfaces support video-only, audio-only and audio-video modes.**
- **For the complete example code, see *GitHub*.**

- **Recording and live video interfaces are still relatively basic. If you need advanced features, use the basic interfaces to simulate the two interfaces to implement advanced custom features by yourself.**
- **You can view the effect in the example project.**

14.8 Scenarios

The previous two sections described relative operations on the *client* and on the *server*. This section explains how to connect the client and server for use. We recommend, that you to read previous two sections before you proceed with this section.

Video monitoring

In the preface, we have introduced the devices like network cameras and its functions and usage are described as follows:

- **Procedure**

  This function involves the three roles such as the network camera, the app server , and Alibaba Cloud Object Storage Service (OSS). The network camera uses the client part of the OSS MEDIA C SDK and the app server uses the server part of the OSS C MEDIA SDK.

  Following is the procedure to upload a video clip captured by a network camera:

  1. **To upload a video clip on the OSS, the network camera sends a network request to the app server to get authorization.**
  2. **After the app server receives the request, it performs a check to confirm that the upload permission can be granted to the network camera, and then requests a token with a specific validity period and with only the upload permission from Alibaba Cloud through the get_token interface of the OSS MEDIA C SDK.**
  3. **After Alibaba Cloud receives the request from the app server to get a token, it checks the user configuration. If the check verifies that the request can be allowed, it issues a temporary token (including the temporary AccessKey ID , the temporary AccessKey Secret, and a temporary STS token) with only the**

upload permission to the OSS and only validity for a specified duration and sends the token to the app server.

4. The app server forwards the token to the network camera once the temporary token is received.
5. After the network camera gets the token, you can upload video files to the OSS through the oss_media_write interface of the OSS MEDIA C SDK client.
6. You can also use the server part of the C SDK or use Java, C#, Go, Python, PHP , Ruby SDKs on the app server to implement an HTTP service. This helps the other users to view and manage various video files on the webpage.

- Example code

A simple example project simulating operations on the client and on the server is as follows:

```
char* global_temp_access_key_id = NULL;
char* global_temp_access_key_secret = NULL;
char* global_temp_token = NULL;

/* Authorization function */
static void auth_func(oss_media_file_t *file) {
    file->endpoint = "your endpoint";
    file->is_cname = 0;
    file->access_key_id = global_temp_access_key_id;
    file->access_key_secret = global_temp_access_key_secret;
    file->token = global_temp_token;

    /* Effective time of this authorization */
    file->expiration = time(NULL) + 300;
}

/* Simulate sending a token to the client from the server */
static void send_token_to_client(oss_media_token_t token) {
    global_temp_access_key_id = token.tmpAccessKeyId;
    global_temp_access_key_secret = token.tmpAccessKeySecret;
    global_temp_token = token.securityToken;
}

void get_and_use_token() {
    oss_media_token_t token;

    /* Server logic: Get a temporary token from Alibaba Cloud and
send it to the client */
    {
        int ret;
        char *policy = NULL;
        oss_media_config_t config;

        policy = "{\n"
                 "\"Statement\": [\n"
                 "{"
                 "\"Action\": \"oss:*\",\n"
                 "\"Effect\": \"Allow\",\n"
                 "\"Resource\": \"*\"\n"
                 "}\n"
```

```
                "],\n"
                "\"Version\": \"1\"\n"
                "}\n";

        init_media_config(&config);

    /* Request a temporary authorization token from Alibaba Cloud */
        ret = oss_media_get_token_from_policy(&config, policy,
                17 * 60, &token);

    if (ret ! = 0) {
       printf ("Get token failed.");
       return;
    }

    /* Simulate sending the temporary token to the client */
        send_token_to_client(token);
    }

    /* Client logic: Use the temporary token to operate on files
after the temporary token is obtained from the server */
    {
        int ret;
        int64_t write_size = 0;
        oss_media_file_t *file = NULL;
        char *content = NULL;
    char *bucket_name;
    char *object_key;
        oss_media_file_stat_t stat;

        content = "hello oss media file\n";
    bucket_name = "<your bucket name>";
    object_key = "key";

        /* Open the object */
        file = oss_media_file_open(bucket_name, object_key, "w",
auth_func);
        if (file ! = NULL) {
        printf ("open file failed.");
        return;
    }

        /* Write an object */
        write_size = oss_media_file_write(file, content, strlen(
content));
        if (write_size ! = strlen(content)) {
        printf ("write file failed.");
        return;
    }

        /* Close the object and release the resource */
        oss_media_file_close(file);
    }
}
```

Note:

- **Log on to the *RAM console*. Click Roles, from the left-side navigation pane. Locate the role and click Manage. Go to the Basic Information page, the policy is available.**
- **If precise control of permissions is not required, you can use the simpler oss_media_get_token interface. The path parameter can be /*, and the mode parameter can be rwa.**
- **For more information, see the *RAM and STS Guide*.**