Alibaba Cloud
Object Storage Service

API Reference

Issue: 20181106

Legal disclaimer

Generic conventions

Table -1: Style conventions

Style	Description	Example
⛔	This warning information 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.
⚠	This warning information 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 restore business.
📋	This indicates warning information, supplementary instructions, and other content that the user must understand.	📋 **Note:** Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	📋 **Note:** You can use **Ctrl** + **A** to select all files.
>	Multi-level menu cascade.	**Settings** > **Network** > **Set network type**
Bold	It is used for buttons, menus, page names, and other UI elements.	Click **OK**.
`Courier font`	It is used for commands.	Run the `cd /d C:/windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *`Instance_ID`*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *`[-all|-t]`*
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich` *`{stand	slave}`*

Contents

1 Service operations

2 Bucket operations

2.1 PutBucket

The **`PutBucket`** interface is used to create a bucket (anonymous access is not supported).

The region of the created bucket is consistent with the region of the endpoint from which the request is sent. Once the data center of the bucket is determined, all objects in this bucket are stored in the corresponding region. For more information, see *Regions and endpoints* .

Request syntax

```
PUT / HTTP/1.1
Host: BucketName.oss-cn-hangzhou.aliyuncs.com
Date: GMT Date
x-oss-acl: Permission
Authorization: SignatureValue
<? xml version="1.0" encoding="UTF-8"? >
<CreateBucketConfiguration>
    <StorageClass>Standard</StorageClass>
</CreateBucketConfiguration>
```

Detail analysis

- You can use the `x-oss-acl` header in a Put request to set access permissions for a bucket. Currently, three access permissions are available for a bucket: public-read-write, public-read, and private.
- If the requested bucket already exists, 409 Conflict is returned. Error code: BucketAlre adyExists.
- If the bucket to be created does not conform to the naming conventions, the message of 400 Bad Request is returned. Error code: InvalidBucketName.
- If the information for user authentication is not introduced when you initiate a Put Bucket request, the message of 403 Forbidden is returned. Error code: AccessDenied.
- You can create a maximum of 30 buckets in a region. If the number is exceeded, the message of 400 Bad Request is returned. Error code: TooManyBuckets.
- If no access permission is specified for the created bucket, the `Private` permission applies by default.
- The storage type of a new bucket can be specified. Standard, IA, and Archive are available.
- When creating a bucket, you can specify the data redundancy type for the bucket. The values can be LRS (Locally Redundant Storage) and ZRS (Zone Redundant Storage), in which the LRS is the default value.

Example

Request example:

```
PUT / HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Date: Fri, 24 Feb 2017 03:15:40 GMT
x-oss-acl: private
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:77Dvh5wQgIjWjwO/KyRt8dOPfo
8=
<? xml version="1.0" encoding="UTF-8"? >
<CreateBucketConfiguration>
    <StorageClass>Standard</StorageClass>
</CreateBucketConfiguration>
```

Response example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 24 Feb 2017 03:15:40 GMT
Location: /oss-example
Content-Length: 0
Connection: keep-alive
Server: AliyunOSS
```

2.2 PutBucketLogging

Bucket owners can use **`PutBucketLogging`** to enable the access logging function for their bucket.

When this function is enabled, OSS automatically records the details about the requests to this bucket, and follows the user-specified rules to write the access logs as an object into a user-specified bucket on an hourly basis.

Note:

OSS provides bucket access logs for bucket owners to understand and analyze bucket access behaviors easily. The bucket access logs provided by OSS do not guarantee that every single access record is logged.

Request syntax

```
PUT /? logging HTTP/1.1
Date: GMT Date
Content-Length：ContentLength
Content-Type: application/xml
Authorization: SignatureValue
Host: BucketName.oss-cn-hangzhou.aliyuncs.com
<? xml version="1.0" encoding="UTF-8"? >
<BucketLoggingStatus>
    <LoggingEnabled>
        <TargetBucket>TargetBucket</TargetBucket>
        <TargetPrefix>TargetPrefix</TargetPrefix>
    </LoggingEnabled>
</BucketLoggingStatus>
```

Request elements

Name	Type	Required	Description
`BucketLoggingStatus`	container	Yes	The container for storing access log status information Child element: LoggingEnabled Parent element: none
`LoggingEnabled`	container	No	The container for storing access log information. This element is required only when server access logging is enabled. Child element: TargetBucket, TargetPrefix Parent element: BucketLoggingStatus
`TargetBucket`	character	This element is required when server access logging is enabled	The bucket for storing access logs. Child element: none Parent element: BucketLoggingStatus. LoggingEnabled
`TargetPrefix`	character	No	The prefix of the names of saved access log files. Child element: none Parent element: BucketLoggingStatus. LoggingEnabled

Naming rules for the objects storing access logs

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

In the naming rules, the TargetPrefix is specified by the user; YYYY, mm, DD, HH, MM, and SS give the year, month, day, hour, minutes, and seconds of the creation time in Arabic numerals (note the digits); and UniqueString is the string generated by OSS system. An example for the name of an object actually used to store OSS access logs is given as follows:

```
MyLog-oss-example-2012-09-10-04-00-00-0000
```

In the preceding example, “MyLog-“ is the Object prefix specified by the user; “oss-example” is the name of the origin bucket; “2012-09-10-04-00-00” is the Object creation time (Beijing time); and “0000” is the string generated by OSS system.

Log file format

Name	Example	Description
Remote IP	119.140.142.11	IP address from which the request is initiated (the proxy or user firewall may block this field)
Reserved	-	Reserved field
Reserved	-	Reserved field
Time	[02/May/2012:00:00:04 +0800]	Time when OSS receives the request
Request-URI	“GET /aliyun-logo.png HTTP/1.1“	User-Requested URI (including query-string)
HTTP Status	200	HTTP status code returned by OSS
SentBytes	5576	Traffic that the user downloads from OSS
RequestTime (ms)	71	Time utilized in completing this request (in ms)
Referer	`http://www.aliyun.com/product/oss`	HTTP Referer in the request
User-Agent	curl/7.15.5	HTTP User-Agent header
HostName	oss-example.regionid.example.com	Domain name for access request
Request ID	505B01695037C2AF032593A4	UUID used to uniquely identify this request
LoggingFlag	true	Whether the access logging function is enabled
Requester Aliyun ID	1657136103983691	Alibaba Cloud ID of the requester, “-“ for an anonymous access
Operation	GetObject	Request type
Bucket	oss-example	Name of the bucket requested for access
Key	/aliyun-logo.png	Key of user request
ObjectSize	5576	Object size
Server Cost Time (ms)	17	Time utilized by OSS server to process this request (in ms)
Error Code	NoSuchBucket	Error code returned by OSS
Request Length	302	Length of user request (byte)

Name	Example	Description
UserID	1657136103983691	ID of the bucket owner
Delta DataSize	280	Bucket size variation, "-" for no change
Sync Request	-	Whether this is an origin retrieval request from CND, "-" for no
Reserved	-	Reserved field

Detail analysis

- The source bucket and target bucket must belong to the same user.
- In the preceding request syntax, "BucketName" refers to the bucket for which access logging is enabled; "TargetBucket" refers to the bucket into which access logs are saved; "TargetPrefix" refers to the name prefix of the object storing access logs and can be null.
- The source bucket and target bucket can be the same or different buckets. You can save logs from multiple source buckets to the same target bucket (in this case, we recommend that you assign different values to TargetPrefix).
- To disable the access logging function for a bucket, you only must send an empty BucketLogg ingStatus request. For a detailed method, see the following request example.
- All PUT Bucket Logging requests must be provided with signatures, because the anonymous access is not supported.
- If the initiator of a PUT Bucket Logging request is not the owner of the source bucket (BucketName in the request example), OSS returns error code 403.
- If the source bucket does not exist, OSS returns the error code: NoSuchBucket.
- If the initiator of a PUT Bucket Logging request is not the owner of the target bucket (indicated by TargetBucket in the request example), OSS returns Error 403. If the target bucket does not exist, OSS returns the error code: InvalidTargetBucketForLogging.
- The source bucket and target bucket must belong to the same data center. Otherwise, Error 400 with the error code: InvalidTargetBucketForLogging is returned.
- If a PUT Bucket Logging request has an invalid XML, the error code: MalformedXML is returned.
- The source bucket and target bucket can be the same bucket. You can save the logs of different source buckets into the same target bucket (note that you must set TargetPrefix to different values).
- When the source bucket is deleted, the corresponding logging rules are also deleted.

- OSS generates a bucket access log file every hour. However, all requests during the hour may not be recorded in the log file, but may get recorded in the previous or next log file.
- In the naming rules for log files generated by OSS, “UniqueString” is only a UUID that OSS generates for a file to uniquely identify the file.
- Each time OSS generates a bucket access log file, this is considered a PUT operation and the occupied space is recorded, but the generated traffic is not recorded. After log files are generated, you can operate these log files as common objects.
- OSS ignores all query-string parameters prefixed by “x-“ but such query-string parameters are recorded in access logs. If you want to mark a special request from massive access logs, you can add a query-string parameter prefixed by “x-“ to the URL. For example:

  `http://oss-example.oss-cn-hangzhou.aliyuncs.com/aliyun-logo.png`

  `http://oss-example.regionid.example.com/aliyun-logo.png?x-user=admin`
- When OSS processes the preceding two requests, the results are the same. However, you can search access logs with “x-user=admin” to quickly locate the marked request.
- You may see “-“ in any field of OSS logs. It indicates that data is unknown or the field is invalid for the current request.
- Certain fields are added to the end of OSS log files in future based on the requirements. We recommend that developers to consider compatibility issues when developing log processing tools.
- If you have uploaded the Content-MD5 request header, OSS calculates the body’s Content-MD5 and checks if the two are the same. If the two are different, the error code: InvalidDigest is returned.

Example

Example of a request for enabling bucket access logging:

```
PUT /? logging HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Content-Length: 186
Date: Fri, 04 May 2012 03:21:12 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:KU5h8YMUC78M30dXqf3J
xrTZHiA=
<? xml version="1.0" encoding="UTF-8"? >
<BucketLoggingStatus>
<LoggingEnabled>
<TargetBucket>doc-log</TargetBucket>
<TargetPrefix>MyLog-</TargetPrefix>
</LoggingEnabled>
```

```
</BucketLoggingStatus>
```

Response example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 04 May 2012 03:21:12 GMT
Content-Length: 0
Connection: keep-alive
Server: AliyunOSS
```

Example of a request for disabling bucket access logging:

```
PUT /? logging HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Content-Type: application/xml
Content-Length: 86
Date: Fri, 04 May 2012 04:21:12 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:KU5h8YMUC78M30dXqf3J
xrTZHiA=
<? xml version="1.0" encoding="UTF-8"? >
<BucketLoggingStatus>
</BucketLoggingStatus>
```

Response example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 04 May 2012 04:21:12 GMT
Content-Length: 0
Connection: keep-alive
Server: AliyunOSS
```

2.3 PutBucketLifecycle

The bucket owner can set the lifecycle of a bucket with the **`PutBucketLifecycle`** request. After Lifecycle is enabled, OSS automatically deletes the objects or transitions the objects (to another storage class) corresponding the lifecycle rules on a regular basis.

Request syntax

```
PUT /?lifecycle HTTP/1.1
Date: GMT Date
Content-Length: ContentLength
Content-Type: application/xml
Authorization: SignatureValue
Host: BucketName.oss.aliyuncs.com
<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration>
  <Rule>
    <ID>RuleID</ID>
    <Prefix>Prefix</Prefix>
    <Status>Status</Status>
    <Expiration>
      <Days>Days</Days>
    </Expiration>
    <AbortMultipartUpload>
      <Days>Days</Days>
    </AbortMultipartUpload>
  </Rule>
</LifecycleConfiguration>
```

Request elements

Name	Type	Required?	Description
CreatedBef oreDate	string	One from the two: Days and CreatedBef oreDate	Specify the time before which the rules go into effect. The date must conform to the ISO8601 format and always be UTC 00:00. For example: 2002-10-11T00:00:00.000Z, which means the objects with a last update time before 2002-10-11T00:00:00.000Z are deleted or transitioned to another storage class, and the objects updated after this time (including this time) are not deleted or transitioned. Parent node: Expiration or AbortMulti partUpload
Days	positive integer	One from the two: Days and CreatedBef oreDate	Specify how many days after the last object update until the rules take effect. Parent node: Expiration
Expiration	container	No	Specify the expiration attribute of the object. Sub-node: Days or CreatedBeforeDate Parent node: Rule
AbortMulti partUpload	container	No	Specify the expiration attribute of the unfulfilled Part rules. Sub-node: Days or CreatedBeforeDate Parent node: Rule
ID	string	No	The unique ID of a rule. An ID is composed of 255 bytes at most. When you fail to specify this value or this value is null, OSS generates a unique value for you. Sub-node: none Parent node: Rule
LifecycleC onfiguration	container	Yes	Container used for storing lifecycle configurations, which can hold a maximum of 1000 rules. Sub-node: Rule Parent node: none

Name	Type	Required?	Description
Prefix	string	Yes	Specify the prefix applicable to a rule. Only those objects with a matching prefix can be affected by the rule. It cannot be overlapped. Sub-node: none Parent node: Rule
Rule	container	Yes	Express a rule Sub-nodes: ID, Prefix, Status, Expiration Parent node: LifecycleConfiguration
Status	string	Yes	If this value is Enabled, OSS runs this rule regularly. If this value is Disabled, then OSS ignores this rule. Parent node: Rule Valid value: `Enabled`, `Disabled`
StorageClass	string	Required if parent node transition is set	Specifies the type of target storage that the object is transition to the OSS. Value: `IA`, `Archive` Parent node: Transition
Transition	Container	No	Specifies when the object is transition to the IA or archive storage type during a valid life cycle .

Detail analysis

- Only the bucket owner can initiate a Put Bucket Lifecycle request. Otherwise, the message of 403 Forbidden is returned. Error code: AccessDenied.
- If no lifecycle has been set previously, this operation creates a new lifecycle configuration or overwrites the previous configuration.
- You can also set an expiration time for an object, or for the Part. Here, the Part refers to the unsubmitted parts for multipart upload.

Notes for storage types transition:

- Supports objects in Standard bucket transition to IA and Archive storage type. Standard bucket can simultaneously configure both transition to IA and archive storage type rules for one object . In this case, the time set to transition to archive must be longer than the time to transition to IA . For example, the days set for transition to IA is 30, then it must be greater than 30 days set for transition to archive. Otherwise, the invalidargument error is returned.

- The object setting must have an expiration time greater than the time converted to IA or archive . Otherwise, the invalidArgument error is returned.
- Supports objects transition to archive storage type in IA bucket.
- Archvie bucket creation is not supported.
- IA object convertion is not supported as standard.
- The archive object convertion is not supported for IA or standard.

Examples

Request example:

```
PUT /?lifecycle HTTP/1.1
Host: oss-example.oss.aliyuncs.com
Content-Length: 443
Date: Mon, 14 Apr 2014 01:08:38 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:KU5h8YMUC78M30dXqf3J
xrTZHiA=
<?xml version="1.0" encoding="UTF-8"?>
</LifecycleConfiguration>
  <Rule>
    <ID>delete objects and parts after one day</ID>
    <Prefix>logs/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <Days>1</Days>
    </Expiration>
    <AbortMultipartUpload>
      <Days>1</Days>
    </AbortMultipartUpload>
  </Rule>
  <Rule>
    <ID>delete created before date</ID>
    <Prefix>backup/</Prefix>
    <Status>Enabled</Status>
    <Expiration>
      <CreatedBeforeDate>2014-10-11T00:00:00.000Z</CreatedBeforeDate>
    </Expiration>
    <AbortMultipartUpload>
      <CreatedBeforeDate>2014-10-11T00:00:00.000Z</CreatedBeforeDate>
    </AbortMultipartUpload>
  </Rule>
</LifecycleConfiguration>
```

Response example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Thu , 8 Jun 2017 13:08:38 GMT
Content-Length: 0
Connection: keep-alive
```

```
Server: AliyunOSS
```

2.4 GetBucket(List Object)

The **`GetBucket`** operation can be used to list all of the object information in a bucket.

Request syntax

```
GET / HTTP/1.1
Host: BucketName.oss-cn-hangzhou.aliyuncs.com
Date: GMT Date
Authorization: SignatureValue
```

Request parameters

When you initiate a GetBucket (ListObject) request, you can use prefix, marker, delimiter, and max -keys to prescribe a limit to the list to return partial results. Besides, encoding-type can be used to encode the following elements in the returned results: delimiter, marker, prefix, NextMarker, and key.

Name	Data type	Required	Description
`delimiter`	string	No	A character used to group object names. All the names of the objects that contain a specified prefix and after which the delimiter occurs for the first time, act as a group of elements - CommonPrefixes. Default value: None
`marker`	string	No	Sets the returned results to begin from the first entry after the marker in alphabetical order. Default value: None
`max-keys`	string	No	Limits the maximum number of objects returned for one request. If not specified, the default value is 100. The max-keys value cannot exceed 1000. Default value: 100
`prefix`	string	No	Limits that the returned object key must be prefixed accordingly. Note that the keys returned from queries using a prefix still contain the prefix. Default value: None
`encoding-type`	string	No	Specifies the encoding of the returned content and the encoding type. Parameters delimiter, marker, prefix, NextMarker, and key use UTF-8 characters, but the XML 1.0 Standard does not support parsing certain control characters, such as characters with ASCII values ranging from 0 to 10. If some elements in the returned

Name	Data type	Required	Description
			results contain characters that are not supported by the XML 1.0 Standard, encoding-type can be specified to encode these elements, such as delimiter, marker, prefix, NextMarker, and key. Default value: None; Optional value: URL

Response elements

Name	Type	Description
Contents	container	Container used for saving every returned object meta. Parent node: ListBucketResult
CommonPrefixes	string	If the delimiter parameter is specified in the request, the response returned by OSS contains the CommonPrefixes element. This element indicates the set of objects which ends with a delimiter and have a common prefix. Parent node: ListBucketResult
Delimiter	string	A character used to group object names. 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 - CommonPrefixes. Parent node: ListBucketResult
EncodingType	string	Encoding type for the returned results. If encoding-type is specified in a request, the following elements in the returned results are encoded: delimiter, marker, prefix, NextMarker, and key. Parent node: ListBucketResult
DisplayName	string	Name of the object owner. Parent node: ListBucketResult.Contents.Owner
ETag	string	The ETag (entity tag) is created when an object is generated and is used to indicate the content of the object. For an object created by a Put Object request, the value of ETag is the value of MD5 in the content of the object. For an object created in other way, the value of ETag is the UUID in the content of the object. The value of ETag can be used to check whether the content of the object is changed. We recommend that the ETag be used as the MD5 value of the object content to verify data integrity. Parent node: ListBucketResult.Contents

Name	Type	Description
ID	string	User ID of the bucket owner. Parent node: ListBucketResult.Contents.Owner
IsTruncated	enumerated string	Indicates whether all results have been returned; “true” means that not all results are returned this time; “false” means that all results are returned this time. Valid values: `true` and `false` Parent node: ListBucketResult
Key	string	Key of an object Parent node: ListBucketResult.Contents
LastModified	time	The latest modification time of an object. Parent node: ListBucketResult.Contents
ListBucket Result	container	Container for storing the results of the “Get Bucket” request subnodes: Name, Prefix, Marker, MaxKeys, Delimiter, IsTruncated, Nextmarker, and Contents Parent node: None
Marker	string	Marks the origin of the current Get Bucket (List Object) request. Parent node: ListBucketResult
MaxKeys	string	The maximum number of returned results in response to the request. Parent node: ListBucketResult
Name	string	Name of a bucket Parent node: ListBucketResult
Owner	container	Container used for saving the information about the bucket owner. subnodes: DisplayName and ID Parent node: ListBucketResult
Prefix	string	Starting prefix for the current results of query. Parent node: ListBucketResult
Size	string	Number of bytes of the object. Parent node: ListBucketResult.Contents
StorageClass	string	Indicates Object storage type. “Standard”, “IA”, and “Archive” types are available. (Currently, the “Archive” type is only available in some regions.) Parent node: ListBucketResult.Contents

Detail analysis

- The custom meta in the object is not returned during the GetBucket request.
- If the bucket to be accessed does not exist, or if you attempt to access a bucket which cannot be created because of standard naming rules are not followed when naming a bucket, Error 404 Not Found with the error code “NoSuchBucket” is returned.
- If you have no permission to access the bucket, the system returns Error 403 Forbidden with the error code “AccessDenied”.
- If listing cannot be completed at one time because of the max-keys setting, a `<NextMarker>` is appended to the returned result, prompting that this can be taken as a marker for continued listing. The value in NextMarker is still in the list result.
- During a condition query, even if the marker does not exist in the list actually, what is returned is printed starting from the next to what conforms to the marker letter sorting. If the max-keys value is less than 0 or greater than 1000, error 400 Bad Request is returned. The error code is “InvalidArgument”.
- If the prefix, marker, or delimiter parameters do not meet the length requirement, 400 Bad Request is returned. The error code is “InvalidArgument”.
- The prefix and marker parameters are used to achieve display by pages, and the parameter length must be less than 1024 bytes.
- Setting a prefix as the name of a folder lists the files starting with this prefix, recursively returning all files and subfolders in this folder. Additionally, if we set the Delimiter as “/“, the returned values lists the files in the folder and the subfolders are returned in the CommonPref ixes section. Recursive files and folders in the subfolders are not displayed. For example, a bucket has the following three objects: fun/test.jpg, fun/movie/001.avi, and fun/movie/007.avi. If the prefix is set to “fun/“, three objects are returned. If the delimiter is set to “/“ additionally, file “fun/test.jpg” and prefix “fun/movie/“ are returned. That is, the folder logic is achieved.

Scenario example

Four objects are available in the bucket “my_oss” and are named as:

- oss.jpg
- fun/test.jpg
- fun/movie/001.avi
- fun/movie/007.avi

Example

Request example:

```
GET / HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Date: Fri, 24 Feb 2012 08:43:27 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:BC+oQIXVR2/ZghT7cGa0y
kboO4M=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 24 Feb 2012 08:43:27 GMT
Content-Type: application/xml
Content-Length: 1866
Connection: keep-alive
Server: AliyunOSS
<? xml version="1.0" encoding="UTF-8"? >
<ListBucketResult xmlns=”http://doc.oss-cn-hangzhou.aliyuncs.com”>
<Name>oss-example</Name>
<Prefix></Prefix>
<Marker></Marker>
<MaxKeys>100</MaxKeys>
<Delimiter></Delimiter>
    <IsTruncated>false</IsTruncated>
    <Contents>
        <Key>fun/movie/001.avi</Key>
        <LastModified>2012-02-24T08:43:07.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
        <StorageClass>Standard</StorageClass>
        <Owner>
            <ID>00220120222</ID>
            <DisplayName>user-example</DisplayName>
        </Owner>
    </Contents>
    <Contents>
        <Key>fun/movie/007.avi</Key>
        <LastModified>2012-02-24T08:43:27.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
        <StorageClass>Standard</StorageClass>
        <Owner>
            <ID>00220120222</ID>
            <DisplayName>user-example</DisplayName>
        </Owner>
    </Contents>
<Contents>
        <Key>fun/test.jpg</Key>
        <LastModified>2012-02-24T08:42:32.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
        <StorageClass>Standard</StorageClass>
        <Owner>
            <ID>00220120222</ID>
            <DisplayName>user-example</DisplayName>
        </Owner>
    </Contents>
    <Contents>
        <Key>oss.jpg</Key>
        <LastModified>2012-02-24T06:07:48.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
        <StorageClass>Standard</StorageClass>
        <Owner>
            <ID>00220120222</ID>
            <DisplayName>user-example</DisplayName>
        </Owner>
    </Contents>
</ListBucketResult>
```

Example of a request containing the prefix parameter:

```
GET /? prefix=fun HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Date: Fri, 24 Feb 2012 08:43:27 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:BC+oQIXVR2/ZghT7cGa0y
kboO4M=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 24 Feb 2012 08:43:27 GMT
Content-Type: application/xml
Content-Length: 1464
Connection: keep-alive
Server: AliyunOSS
<? xml version="1.0" encoding="UTF-8"? >
<ListBucketResult xmlns=”http://doc.oss-cn-hangzhou.aliyuncs.com”>
<Name>oss-example</Name>
<Prefix>fun</Prefix>
<Marker></Marker>
<MaxKeys>100</MaxKeys>
<Delimiter></Delimiter>
    <IsTruncated>false</IsTruncated>
    <Contents>
        <Key>fun/movie/001.avi</Key>
        <LastModified>2012-02-24T08:43:07.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
        <StorageClass>Standard</StorageClass>
        <Owner>
            <ID>00220120222</ID>
            <DisplayName>user_example</DisplayName>
        </Owner>
    </Contents>
    <Contents>
        <Key>fun/movie/007.avi</Key>
        <LastModified>2012-02-24T08:43:27.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
```

```
         <StorageClass>Standard</StorageClass>
         <Owner>
             <ID>00220120222</ID>
             <DisplayName>user_example</DisplayName>
         </Owner>
     </Contents>
     <Contents>
         <Key>fun/test.jpg</Key>
         <LastModified>2012-02-24T08:42:32.000Z</LastModified>
         <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
         <Type>Normal</Type>
         <Size>344606</Size>
         <StorageClass>Standard</StorageClass>
         <Owner>
             <ID>00220120222</ID>
             <DisplayName>user_example</DisplayName>
         </Owner>
     </Contents>
</ListBucketResult>
```

Example of a request containing parameters prefix and delimiter:

```
GET /? prefix=fun/&delimiter=/ HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Date: Fri, 24 Feb 2012 08:43:27 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:DNrnx7xHk3sgysx7I8U9
I9IY1vY=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 534B371674E88A4D8906008B
Date: Fri, 24 Feb 2012 08:43:27 GMT
Content-Type: application/xml
Content-Length: 712
Connection: keep-alive
Server: AliyunOSS
<? xml version="1.0" encoding="UTF-8"? >
<ListBucketResult xmlns=”http://doc.oss-cn-hangzhou.aliyuncs.com”>
<Name>oss-example</Name>
<Prefix>fun/</Prefix>
<Marker></Marker>
<MaxKeys>100</MaxKeys>
<Delimiter>/</Delimiter>
    <IsTruncated>false</IsTruncated>
    <Contents>
        <Key>fun/test.jpg</Key>
        <LastModified>2012-02-24T08:42:32.000Z</LastModified>
        <ETag>&quot;5B3C1A2E053D763E1B002CC607C5A0FE&quot;</ETag>
        <Type>Normal</Type>
        <Size>344606</Size>
        <StorageClass>Standard</StorageClass>
        <Owner>
            <ID>00220120222</ID>
            <DisplayName>user_example</DisplayName>
        </Owner>
    </Contents>
   <CommonPrefixes>
        <Prefix>fun/movie/</Prefix>
   </CommonPrefixes>
```

```
</ListBucketResult>
```

2.5 GetBucketInfo

`GetBucketInfo` operation is used to view the bucket information.

The information includes the following:

- Create time
- Internet access endpoint
- Intranet access endpoint
- Bucket owner information
- Bucket ACL (AccessControlList)

Request syntax

```
GET /? bucketInfo HTTP/1.1
Host: BucketName.oss.aliyuncs.com
Date: GMT Date
Authorization: SignatureValue
```

Response elements

Name	Type	Description
`BucketInfo`	Container	The container that saves the bucket information content Sub-node: Bucket node Parent node: none
`Bucket`	Container	The container that saves the bucket specific information Parent node: BucketInfo node
`CreationDate`	time	The creation time of the bucket. Time format: 2013-07-31T10:56:21.000Z Parent node: BucketInfo.Bucket
`ExtranetEndpoint`	string	The Internet domain name that the bucket accesses Parent node: BucketInfo.Bucket
`IntranetEndpoint`	string	The intranet domain name for accessing the bucket from ECS in the same region Parent node: BucketInfo.Bucket
`Location`	string	The region of the data center that the bucket is located in Parent node: BucketInfo.Bucket

Name	Type	Description
`Name`	string	The bucket name Parent node: BucketInfo.Bucket
`Owner`	container	Container used for saving the information about the bucket owner. Parent node: BucketInfo.Bucket
`ID`	string	User ID of the bucket owner. Parent node: BucketInfo.Bucket.Owner
`DisplayName`	string	Name of the bucket owner (the same as the ID currently). Parent node: BucketInfo.Bucket.Owner
`AccessControlList`	container	Container used for storing the ACL information Parent node: BucketInfo.Bucket
`Grant`	enumerative string	ACL permissions of the bucket. Valid values: `private`, `public-read`, and `public-read-write` Parent node: BucketInfo.Bucket.AccessControlList
`DataRedundancyType`	enumerative string	The data redundancy type of the bucket. Valid values: `LRS`and `ZRS` Parent node: BucketInfo.Bucket

Detail analysis

- If the bucket does not exist, error 404 is returned. Error code: NoSuchBucket.
- Only the owner of a bucket can view the information of the bucket. If other users attempt to access the location information, the error 403 Forbidden with the error code: AccessDenied is returned.
- The request can be initiated from any OSS endpoint.

Example

Request example:

```
Get /? bucketInfo HTTP/1.1
Host: oss-example.oss.aliyuncs.com
Date: Sat, 12 Sep 2015 07:51:28 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc: BuG4rRK+zNhH1AcF51
NNHD39zXw=
```

Return example after the bucket information is obtained successfully:

```
HTTP/1.1 200
```

```
x-oss-request-id: 534B371674E88A4D8906008B
Date: Sat, 12 Sep 2015 07:51:28 GMT
Connection: keep-alive
Content-Length: 531
Server: AliyunOSS

<? xml version="1.0" encoding="UTF-8"? >
<BucketInfo>
  <Bucket>
    <CreationDate>2013-07-31T10:56:21.000Z</CreationDate>
    <ExtranetEndpoint>oss-cn-hangzhou.aliyuncs.com</ExtranetEndpoint>
    <IntranetEndpoint>oss-cn-hangzhou-internal.aliyuncs.com</
IntranetEndpoint>
    <Location>oss-cn-hangzhou</Location>
    <Name>oss-example</Name>
    <Owner>
      <DisplayName>username</DisplayName>
      <ID>271834739143143</ID>
    </Owner>
    <AccessControlList>
      <Grant>private</Grant>
    </AccessControlList>
  </Bucket>
</BucketInfo>
```

Return example if the requested bucket information does not exist:

```
HTTP/1.1 404
x-oss-request-id: 534B371674E88A4D8906009B
Date: Sat, 12 Sep 2015 07:51:28 GMT
Connection: keep-alive
Content-Length: 308
Server: AliyunOSS

<? xml version="1.0" encoding="UTF-8"? >
<Error>
  <Code>NoSuchBucket</Code>
  <Message>The specified bucket does not exist.</Message>
  <RequestId>568D547F31243C673BA14274</RequestId>
  <HostId>nosuchbucket.oss.aliyuncs.com</HostId>
  <BucketName>nosuchbucket</BucketName>
</Error>
```

Return example if the requester has no access permission to the bucket information:

```
HTTP/1.1 403
x-oss-request-id: 534B371674E88A4D8906008C
Date: Sat, 12 Sep 2015 07:51:28 GMT
Connection: keep-alive
Content-Length: 209
Server: AliyunOSS

<? xml version="1.0" encoding="UTF-8"? >
<Error>
  <Code>AccessDenied</Code>
  <Message>AccessDenied</Message>
  <RequestId>568D5566F2D0F89F5C0EB66E</RequestId>
  <Hostid> test.oss.aliyuncs.com </hostid>
</Error>
```

3 Object operations

3.1 CopyObject

`CopyObject` is used to copy an object within a bucket or between buckets in the same region.

You can send a PUT request to OSS, and add the element “x-oss-copy-source” to the PUT request header to specify the copy source. OSS automatically determines that this is a Copy Object operation, and directly performs this operation on the server side. If the Copy Object operation is successful, the system returns new object information.

This operation is applicable to a file smaller than 1 GB. To copy a file greater than 1 GB, you must use the Multipart Upload operation. For more information about this operation, see *UploadPartCopy*.

Note:
For the Copy Object operation, the source bucket and the target bucket must be in the same region.

Request syntax

```
PUT /DestObjectName HTTP/1.1
Host: DestBucketName.oss-cn-hangzhou.aliyuncs.com
Date: GMT Date
Authorization: SignatureValue
x-oss-copy-source: /SourceBucketName/SourceObjectName
```

Request header

Name	Type	Description
`x-oss-copy-source`	String	Specifies the copy source address (the requester must have the permission to read the source object). Default: None
`x-oss-copy-source-if-match`	String	If the source object’s ETag value is same as the ETag value provided by the user, a COPY operation is executed, and the code 200 is returned. Otherwise, the system returns the HTTP error code 412 (preprocessing failed). Default: None
`x-oss-copy-source-if-none-match`	String	If the source object’s ETag value is not the same as the ETag value provided by the user, a COPY operation is executed, and the code 200 is returned. Otherwise, the

Name	Type	Description
		system returns the HTTP error code 304 (preprocessing failed). Default: None
`x-oss-copy-source-if-unmodified-since`	String	If the time specified by the received parameter is same as or later than the modification time of the file, the system transfers the file normally, and returns 200 OK; otherwise, the system returns 412 Precondition Failed. Default: None
`x-oss-copy-source-if-modified-since`	String	If the source object has been modified after the time specified by the user, the system performs a COPY operation. Otherwise, the system returns the 304 HTTP error code (preprocessing failed). Default: None
`x-oss-metadata-directive`	String	Valid values include COPY and REPLACE. If this parameter is set to COPY, the system copies meta for the new object from the source object. If this parameter is set to REPLACE, the system ignores all meta values of the source object, and uses the meta value specified in this request. If this parameter is set to a value other than COPY and REPLACE, the system returns the 400 Bad Request message. Note that when the value is COPY, the source object's x-oss-server-side-encryption meta value cannot be copied. Default value: `COPY` Valid values:`COPY` and `REPLACE`
x-oss-server-side-encryption	String	Specifies the server-side entropy encryption algorithm when OSS creates the target object. Valid values: `AES256` or `KMS` **Note:** You must enable the KMS (Key Management Service) on the console to use the KMS encryption algorithm. Otherwise, a KmsServiceNotenabled error code is reported.
`x-oss-object-acl`	String	Specifies the access permission when OSS creates an object. Valid values: `public-read`, `private`, `public-read-write`

Response elements

Name	Type	Description
`CopyObjectResult`	String	Object copying result Default: None
`ETag`	String	ETag value of the new object. Parent element: CopyObjectResult
`LastModified`	String	Last update time of the new object. Parent element: CopyObjectResult

Detail analysis

- You can use the Copy Object operation to modify the meta information of an existing object.
- If the source object address is the same as the target object address in the Copy Object operation, the system directly replaces the meta information in the source object regardless of the value of x-oss-metadata-directive.
- OSS allows the Copy Object request to contain any number of the four pre-judgment headers. For more information about the related logic, see Detail Analysis of Get Object.
- To complete a Copy Object operation, the requester must have the permission to read the source object.
- The source object and the target object must belong to the same data center. Otherwise, the system returns the error code 403 AccessDenied. The error message is Target object does not reside in the same data center as source object.
- In the billing statistics of the Copy Object operation, the number of Get requests increases by 1 in the bucket of the source object, the number of Put requests increases by 1 in the bucket of the target object, and a storage space is added accordingly.
- In the Copy Object operation, all relevant request headers start from x-oss-, and therefore must be added to the signature string.
- If the x-oss-server-side-encryption header is specified in the Copy Object request, and its value (AES256) is valid, the target object is encrypted on the server side after the Copy Object operation is performed no matter whether the source object has been encrypted on the server side. In addition, the Copy Object response header contains x-oss-server-side-encryption, the value of which is set to the encryption algorithm of the target object. When this target object is downloaded, the response header also contains x-oss-server-side-encryption, the value of which is set to the encryption algorithm of this target object. If the x-oss-server-side-encryption request header is not specified in the Copy Object operation, the target object is the data that

is not encrypted on the server side even if the source object has been encrypted on the server side.

- When the x-oss-metadata-directive header in the Copy Object request is set to COPY (default value), the system does not copy the x-oss-server-side-encryption value of the source object. That is, the target object is encrypted on the server side only when x-oss-server-side-encryption is specified accordingly in the Copy Object request.
- When the x-oss-server-side-encryption request header is specified in the COPY operation, and the request value is not AES256, the system returns Error 400 with the error code “InvalidEncryptionAlgorithmError”.
- If the size of the file to be copied is greater than 1 GB, the system returns Error 400 with the error code “EntityTooLarge”.
- This operation cannot be used to copy objects created by Append Object.
- If the file type is symbolic link, copy the symbolic link only.

Example

Request example:

```
PUT /copy_oss.jpg HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Date: Fri, 24 Feb 2012 07:18:48 GMT
x-oss-copy-source: /oss-example/oss.jpg
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:gmnwPKuu20LQEjd+iPkL259A+
n0=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 559CC9BDC755F95A64485981
Content-Type: application/xml
Content-Length: 193
Connection: keep-alive
Date: Fri, 24 Feb 2012 07:18:48 GMT
Server: AliyunOSS
<? xml version="1.0" encoding="UTF-8"? >
<CopyObjectResult xmlns=”http://doc.oss-cn-hangzhou.aliyuncs.com”>
 <LastModified>Fri, 24 Feb 2012 07:18:48 GMT</LastModified>
 <ETag>"5B3C1A2E053D763E1B002CC607C5A0FE"</ETag>
```

```
</CopyObjectResult>
```

3.2 HeadObject

`HeadObject` is used to return the meta information of a certain object without returning the file content.

Request syntax

```
HEAD /ObjectName HTTP/1.1
Host: BucketName/oss-cn-hangzhou.aliyuncs.com
Date: GMT Date
Authorization: SignatureValue
```

Request header

Name	Type	Description
`If-Modified-Since`	String	If the specified time is earlier than the actual modification time, the system returns the 200 OK message and the object metadata; otherwise, the system returns the 304 Not Modified message. Default: None
`If-Unmodified-Since`	String	If the specified time is same as or later than the actual file modification time, the system returns the 200 OK message and the object metadata; otherwise, the system returns the 412 Precondition Failed message. Default: None
`If-Match`	String	If the expected ETag that is introduced matches the ETag of the object, the system returns the 200 OK message and the object metadata; otherwise, the system returns the 412 Precondition Failed message. Default: None

Name	Type	Description
`If-None-Match`	String	If the introduced ETag does not match the ETag of the object, the system returns the 200 OK message and the object metadata; otherwise, the system returns the 304 Not Modified message. Default: None

Detail analysis

- After the Head Object request is sent, no message body is returned even if the system returns the 200 OK message or an error message.
- The If-Modified-Since, If-Unmodified-Since, If-Match, and If-None-Match query conditions can be set in the header of the Head Object request. For the detailed setting rules, see the related fields in the Get Object request. If no modification is made, the system returns the 304 Not Modified message.
- If you upload the user meta prefixed with x-oss-meta- when sending a Put Object request, for example, x-oss-meta-location, the user meta is returned.
- If the file does not exist, the system returns Error 404 Not Found.
- If this object is entropy encrypted on the server, the system returns x-oss-server-side-encryption in the header of the response to the Head Object request. The value of x-oss-server-side-encryption indicates the server-side encryption algorithm of the object.
- If the file type is symbolic link, in the response header, `Content-Length`, `ETag`, and `Content-Md5` are metadata of the target file, `Last-Modified` is the maximum value of the target file and symbolic link, and others are metadata of symbolic links.
- If the file type is symbolic link and the target file does not exist, the system returns Error 404 Not Found. The error code is “SymlinkTargetNotExist”.
- If the file type is symbolic link and the target file type is symbolic link, the system returns Error 400 Bad request. The error code is “InvalidTargetType”.
- If the bucket type is Archive and the Restore request has been submitted, the Restore state of Object is indicated by x-oss-restore in the response header.
  - If the Restore request is not submitted or times out, the field is not returned.
  - If the Restore request has been submitted and does not time out, the value of x-oss-restore returned is ongoing-request=”true”.

- If the Restore request has been submitted and completed, the value of x-oss-restore returned is ongoing-request=”false”, expiry-date=”Sun, 16 Apr 2017 08:12:33 GMT”. Where the expiry-date refers to the expiry date of the readable state of the restored file.

Example

Request example:

```
HEAD /oss.jpg HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Date: Fri, 24 Feb 2012 07:32:52 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:JbzF2LxZUtanlJ5dLA092wpDC/
E=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 559CC9BDC755F95A64485981
x-oss-object-type: Normal
x-oss-storage-class: Archive
Date: Fri, 24 Feb 2012 07:32:52 GMT
Last-Modified: Fri, 24 Feb 2012 06:07:48 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 344606
Content-Type: image/jpg
Connection: keep-alive
Server: AliyunOSS
```

Example of a request when the Restore request has been submitted but not completed:

```
HEAD /oss.jpg HTTP/1.1
Host: oss-archive-example.oss-cn-hangzhou.aliyuncs.com
Date: Sat, 15 Apr 2017 07:32:52 GMT
Authorization: OSS e1Unnbm1rgdnpI:KKxkdNrUBu2t1kqlDh0MLbDb99I=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 58F71A164529F18D7F000045
x-oss-object-type: Normal
x-oss-storage-class: Archive
x-oss-restore: ongoing-request="true"
Date: Sat, 15 Apr 2017 07:32:52 GMT
Last-Modified: Sat, 15 Apr 2017 06:07:48 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 344606
Content-Type: image/jpg
Connection: keep-alive
Server: AliyunOSS
```

Example of a request when the Restore request has been submitted and completed:

```
HEAD /oss.jpg HTTP/1.1
Host: oss-archive-example.oss-cn-hangzhou.aliyuncs.com
Date: Sat, 15 Apr 2017 09:35:51 GMT
```

```
Authorization: OSS e1Unnbm1rgdnpI:21qtGJ+ykDVmdu6O6FMJnn+WuBw=
```

Return example:

```
HTTP/1.1 200 OK
x-oss-request-id: 58F725344529F18D7F000055
x-oss-object-type: Normal
x-oss-storage-class: Archive
x-oss-restore: ongoing-request="false", expiry-date="Sun, 16 Apr 2017
08:12:33 GMT"
Date: Sat, 15 Apr 2017 09:35:51 GMT
Last-Modified: Sat, 15 Apr 2017 06:07:48 GMT
ETag: "fba9dede5f27731c9771645a39863328"
Content-Length: 344606
```

3.3 PostObject

The **`PostObject`** operation is used to upload a file to a specified bucket using the HTML form.

As a substitute of Put Object, Post Object makes it possible to upload files to a bucket based on the browser. The message body of Post Object is encoded using multipart/form-data. In the Put Object operation, parameters are transferred through the HTTP request header.

Post object

Request syntax

```
POST / HTTP/1.1
Host: BucketName.oss-cn-hangzhou.aliyuncs.com
User-Agent: browser_data
Content-Length：ContentLength
Content-Type: multipart/form-data; boundary=9431149156168
--9431149156168
Content-Disposition: form-data; name="key"
key
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"
success_redirect
--9431149156168
Content-Disposition: form-data; name="Content-Disposition"
attachment;filename=oss_download.jpg
--9431149156168
Content-Disposition: form-data; name="x-oss-meta-uuid"
myuuid
--9431149156168
Content-Disposition: form-data; name="x-oss-meta-tag"
mytag
--9431149156168
Content-Disposition: form-data; name="OSSAccessKeyId"
access-key-id
--9431149156168
Content-Disposition: form-data; name="policy"
encoded_policy
--9431149156168
Content-Disposition: form-data; name="Signature"
signature
--9431149156168
```

```
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg
file_content
--9431149156168
Content-Disposition: form-data; name="submit"
Upload to OSS
```

Form fields

Name	Type	Description	Required or Optional
`OSSAccessKeyId`	string	Specify the AccessKey ID of the bucket owner. Default value: none Restriction: This form field is required when the bucket does not allow public-read-write , or when the Policy (or Signature) form field is provided.	Conditional
`policy`	string	Specify validity of the form fields in the request. A request that does not contain the Policy form field is treated as an anonymous request, and can only access buckets that allow public-read-write. For more information, see 5.7.4.1 Post policy. Default value: none Restriction: This form field is required when the bucket does not allow public-read-write, or when the OSSAccessKeyId (or Signature) form field is provided.	Conditional
`Signature`	string	Specify the signature information that is	Conditional

Name	Type	Description	Required or Optional
		computed based on the Access Key Secret and Policy. The OSS checks the signature information to verify validity of the Post Object request. For more information, see 5.7.4.2 Post Signature. Default value: none Restriction: This form field is required when the bucket does not allow public-read-write, or when the OSSAccessKeyId (or Policy) form field is provided.	
Cache-Control, **Content-Type**, **Content-Disposition**, **Content-Encoding**, **Expires**	string	REST request headers. For more information, see the related descriptions in Put Object. Default value: none	Optional
file	string	Specify the file or text content. It must be the last field in the form. The browser automatically sets Content-Type based on the file type, and overwrites the user setting. The OSS can only upload one file at a time. Default value: none	Required
key	string	Specify the object name of the uploaded file. If the object name contains forward slashes	Required

Name	Type	Description	Required or Optional
		(/), such as a/b/c/b.jpg, OSS will create the corresponding directory. Default value: none	
`success_action_redirect`	string	Specify the URL to which the client is redirected after successful upload. If this form field is not specified, the returned result is specified by success_action_status. If upload fails, the OSS returns an error code, and the client is not redirected to any URL. Default value: none	Optional
`success_action_status`	string	Specify the status code returned to the client after the previous successful upload if success_action_redirect is not specified. Valid values include 200, 201, and 204 (default). If this field is set to 200 or 204, the OSS returns an empty file and a corresponding status code. If this field is set to 201, the OSS returns an XML file and the 201 status code. If this field is not specified or set to an invalid value, the OSS returns an empty file	

Name	Type	Description	Required or Optional
		and the 204 status code. Default value: none	
`x-oss-meta-*`	string	Specify the user meta value set by the user. The OSS does not check or use this value. Default value: none	Optional
`x-oss-server-side-encryption`	string	Specify the server-side encryption algorithm when the OSS creates an object. Valid value: `AES256`	Optional
`x-oss-object-acl`	string	Specify the access permission when the OSS creates an object. Valid values: `public-read`, `private`, and `public-read-write`	Optional
`x-oss-security-token`	string	If STS temporary authorization is used for this access, you must specify the item to be the SecurityToken value. At the same time, OSSAccessKeyId must use a paired temporary AccessKeyId. The signature calculation is consistent with the general AccessKeyId signature. Default value: none	Optional

Response header

Name	Type	Description
`x-oss-server-side-encryption`	string	If x-oss-server-side-encryption is specified in the request, the response contains this header, which indicates the encryption algorithm used.

Response elements

Name	Type	Description
`PostResponse`	container	Specify the container that saves the result of the Post Object request. Sub-nodes: Bucket, ETag, Key, and Location
`Bucket`	string	Specify the bucket name. Parent node: PostResponse
`ETag`	string	Specify the entity tag (ETag) that is created when an object is generated. For an object created by Post Object, the ETag value is the UUID of the object, and can be used to check whether the content of the object has changed. Parent node: PostResponse
`Location`	string	Specify the URL of the newly created object. Parent node: PostResponse

Detail analysis

- To perform the Post Object operation, you must have the permission to write the bucket. If the bucket allows public-read-write, you can choose not to upload the signature information ; otherwise, signature verification must be performed on the Post Object operation. Unlike Put Object, Post Object uses AccessKeySecret to compute the signature for the policy. The computed signature string is used as the value of the Signature form field. The OSS checks this value to verify validity of the signature.
- No matter whether the bucket allows public-read-write, once any one of the OSSAccessKeyId, Policy, and Signature form fields is uploaded, the remaining two form fields are required. If the remaining two form fields are missing, the OSS returns the error code: InvalidArgument.
- Form encoding submitted by the Post Object operation must be "multipart/form-data". That is, Content-Type in the header must be in the `multipart/form-data; boundary=xxxxxx` format, where boundary is the boundary string.

- The URL of the submitted form can be the domain name of the bucket. It is not necessary to specify the object in the URL. That is, the request line is `POST / HTTP/1.1`, and cannot be written as `POST /ObjectName HTTP/1.1`.
- The policy specifies the valid values of form fields in the Post Object request. The OSS checks validity of the request based on the policy. If the request is invalid, the OSS returns the error code: AccessDenied. When checking validity of the policy, the OSS does not check irrelevant form fields in the policy.
- The form and policy must be encoded with UTF-8. The policy is a JSON text encoded with UTF -8 and Base64.
- The Post Object request can contain extra form fields. The OSS checks validity of these form fields based on the policy.
- If you have uploaded the Content-MD5 request header, the OSS calculates the body's Content -MD5 and check if the two are consistent. If the two are different, the error code InvalidDigest is returned.
- If the Post Object request contains the Header signature or URL signature, the OSS does not check these signatures.
- If the Put Object request carries a form field prefixed with x-oss-meta-, the form field is treated as the user meta, for example, x-oss-meta-location. A single object can have multiple similar parameters, but the total size of all user meta cannot exceed 8 KB.
- The total length of the body in the Post Object request cannot exceed 5 GB. When the file length is too large, the system returns the error code: EntityTooLarge.
- If the x-oss-server-side-encryption header is specified when you upload an object, the value of this header must be set to AES256 or KMS. Otherwise, the system returns 400 and the error code: InvalidEncryptionAlgorithmError. After this header is specified, the response header also contains this header, and the OSS stores the encryption algorithm of the uploaded object . When this object is downloaded, the response header contains x-oss-server-side-encryption, the value of which is set to the encryption algorithm of this object.
- Form fields are not case-sensitive, but their values are case-sensitive.

Examples

- Request example:

```
POST / HTTP/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Content-Length: 344606
Content-Type: multipart/form-data; boundary=9431149156168
--9431149156168
```

```
Content-Disposition: form-data; name="key"
/user/a/objectName.txt
--9431149156168
Content-Disposition: form-data; name="success_action_status"
200
--9431149156168
Content-Disposition: form-data; name="Content-Disposition"
content_disposition
--9431149156168
Content-Disposition: form-data; name="x-oss-meta-uuid"
uuid
--9431149156168
Content-Disposition: form-data; name="x-oss-meta-tag"
metadata
--9431149156168
Content-Disposition: form-data; name="OSSAccessKeyId"
44CF9590006BF252F707
--9431149156168
Content-Disposition: form-data; name="policy"
eyJleHBpcmF0aW9uIjoiMjAxMy0xMi0wMVQxMjowMDowMFoiLCJjb25kaXRp
b25zIjpbWyJjb250ZW50LWxlbmd0aC1yYW5nZSIsIDAsIDEwNDg1NzYwXSx7
ImJ1Y2tldCI6ImFoYWhhIn0sIHsiQSI6ICJhIn0seyJrZXkiOiAiQUJDIn1dfQ==
--9431149156168
Content-Disposition: form-data; name="Signature"
kZoYNv66bsmc10+dcGKw5x2PRrk=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.
txt"
Content-Type: text/plain
abcdefg
--9431149156168
Content-Disposition: form-data; name="submit"
Upload to OSS
--9431149156168--
```

- Response example:

```
HTTP/1.1 200 OK
x-oss-request-id: 61d2042d-1b68-6708-5906-33d81921362e
Date: Fri, 24 Feb 2014 06:03:28 GMT
ETag: 5B3C1A2E053D763E1B002CC607C5A0FE
Connection: keep-alive
Content-Length: 0
Server: AliyunOSS
```

Post Policy

The policy form field requested by POST is used to verify the validity of the request. The policy is a JSON text encoded with UTF-8 and Base64. It states the conditions that a Post Object request must meet. Although the post form field is optional for uploading public-read-write buckets, we strongly suggest using this field to limit POST requests.

Policy example

```
{ "expiration": "2014-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "johnsmith" },
    ["starts-with", "$key", "user/eric/"]
    ]
}
```

In the Post Object request, the policy must contain expiration and conditions.

Expiration

Expiration specifies the expiration time of the policy, which is expressed in ISO8601 GMT. For example, "2014-12-01T12:00:00.000Z" means that the Post Object request must be sent before 12:00 on December 1, 2014.

Conditions

Conditions is a list that specifies the valid values of form fields in the Post Object request. Note: The value of a form field is extended after the OSS checks the policy. Therefore, the valid value of the form field set in the policy is equivalent to the value of the form field before extension. The following table lists the conditions supported by the policy:

Name	Description
`content-length-range`	Specify the acceptable maximum and minimum sizes of the uploaded file. This condition supports the content-length-range match mode.
`Cache-Control`, `Content-Type`, `Content-Disposition`, `Content-Encoding`, `Expires`	HTTP request headers. This condition supports the exact match and starts-with match modes.
`key`	Specify the object name of the uploaded file. This condition supports the exact match and starts-with match modes.
`success_action_redirect`	Specify the URL to which the client is redirected after successful upload. This condition supports the exact match and starts-with match modes.
`success_action_status`	Specify the status code returned after successful upload if success_action_redirect is not specified. This condition supports the exact match and starts-with match modes.
`x-oss-meta-*`	Specify the user meta set by the user. This condition supports the exact match and starts-with match modes.

If the Post Object request contains other form fields, these extra form fields can be added to Conditions of the policy. The OSS does not check validity of the form fields that are not contained in the conditions.

Condition match modes

Condition match modes	Description
Exact match	The value of a form field must be exactly the same as the value declared in the conditions. For example, if the value of the key form field must be a, the conditions must be: {“key”: “a”}, or: [“eq”, “$key”, “a”]
Starts With	The value of a form field must start with the specified value. For example, if the value of key must start with /user/user1, the conditions must be: [“starts-with”, “$key”, “/user/user1”]
Specified file size	Specify the maximum and minimum sizes of the files that can be uploaded. For example , if the acceptable file size is 1–10 bytes, the conditions must be: ["content-length-range", 1, 10]

Escape characters

In the policy form field of the Post Object request, $ is used to indicate a variable. Therefore, to describe $, the escape character must be used. In addition, some characters in JSON strings are escaped. The following chart describes characters in the JSON string of the policy form field of a Post Object request.

Escape characters	Description
\/	Slash
\	Backslash
\”	Double quotation marks
\$	Dollar sign
Space	Space
\f	Form feed
\n	Newline
\r	Carriage return

Escape characters	Description
\t	Horizontal tab
\uxxxx	Unicode character

Post Signature

For a verified Post Object request, the HTML form must contain policy and signature. Policy specifies which values are acceptable in the request. The procedure for computing signature is as follows:

1. Create a UTF-8 encoded policy.
2. Encode the policy with Base64. The encoding result is the value of the policy form field, and this value is used as the string to be signed.
3. Use AccessKeySecret to sign the string. The signing method is the same as the computing method of the signature in the Header, that is, replacing the string to be signed with the policy form field.

3.4 Callback

To perform a callback, you only need to attach the relevant callback parameters to the request sent to OSS.

APIs that currently support callbacks are PutObject, PostObject, and CompleteMultipartUpload.

Construct the callback parameter

The callback parameter is composed of a JSON string encoded in Base64. It is critical that you specify the request callback server URL (callbackUrl) and callback content (callbackBody). Detailed JSON fields are as follows:

Field	Meaning	Required?
callbackUrl	• After a file is uploaded successfully, OSS sends a callback request to this URL. The request method is POST and the body is the content specified for callbackBody. Under normal circumstances, if this URL must respond to “HTTP/1.1 200 OK”, the	Yes

Field	Meaning	Required?
	response body must be in the JSON format and the response header Content-Length must be a valid value and not exceeding 3 MB. • This function allows users to set up to 5 URLs, separated by “;”. OSS sends requests one by one until the first successful response is returned. • If no URL is configured or the value is null, it is regarded that callback is not configured. • HTTPS addresses are supported. • To make sure that Chinese characters are correctly processed, the callbackUrl must be encoded. For example, `http://example.com/Chinese.php?key=value&Chinese Name=Chinese Value` needs to be encoded into `http://example.com/%E4%B8%AD%E6%96%87.php?key=value&%E4%B8%AD%E6%96%87%E5%90%8D%E7%A7%B0=%E4%B8%AD%E6%96%87%E5%80%BC.`	
callbackHost	• The host header value for initiating callback requests. It is valid only when the callbackUrl is set. • If no callbackHost is set, the URL in callbackUrl is resolved and the host	No

Field	Meaning	Required?
	generated after resolving is entered in callbackHost.	
callbackBody	• The value of the request body when a callback is initiated, for example, key=$(key)&etag=$(etag)&my_var=$(x:my_var). • It supports OSS system variables, custom variables, and constants. The supported system variables are described in the following table. Custom variables are supported by transmission through callback-var in PutObject and CompleteMultipart. In Post Object operations, each variable is transmitted through a form field.	Yes
callbackBodyType	• The Content-Type of the callback requests initiated. It supports application/x-www-form-urlencoded and application/json, and the former is the default value. • If the Content-Type is set to application/x-www-form-urlencoded, the variables in callbackBody are replaced by URL encoded values. If the Content-Type is set to application/json, these variables are replaced according to the JSON format.	No

JSON string examples are as follows:

```
{
"callbackUrl":"121.101.166.30/test.php",
"callbackHost":"oss-cn-hangzhou.aliyuncs.com",
"callbackBody":"{\"mimeType\":${mimeType},\"size\":${size}}",
"callbackBodyType":"application/json"
}
```

```
{
"callbackUrl":"121.43.113.8:23456/index.html",
"callbackBody":"bucket=${bucket}&object=${object}&etag=${etag}&size
=${size}&mimeType=${mimeType}&imageInfo.height=${imageInfo.height}&
imageInfo.width=${imageInfo.width}&imageInfo.format=${imageInfo.format
}&my_var=${x:my_var}"
}
```

Here, the system variables that can be set for callbackBody include the following. In specific, the imageInfo is for the image format. It must be left empty for a non-image format:

System variable	Meaning
bucket	bucket
object	object
etag	The file's etag, that is, the etag field returned to the user.
size	The object size. During the CompleteMultipartUpload operation, this is the size of the whole object.
mimeType	The resource type. For jpeg images, the resource type is image/jpeg
imageInfo.height	The image height
imageInfo.width	The image width
imageInfo.format	The image format, such as jpg and png

Custom parameters

You can use the callback-var parameter to configure custom parameters.

Custom parameters are a map of key-values. You can configure the required parameters to the map. When initiating a POST callback request, OSS puts these parameters and the system parameters described in the preceding section in the body of the POST request, so that these parameters can be easily obtained by the callback recipient.

You can construct custom parameters in the same way as constructing the callback parameter. The custom parameters can also be transmitted in the JSON format. The JSON string is a map containing key-values of all custom parameters.

Note:

It must be particularly noted that, the keys of the custom parameters must start with x: and be in the lower case. Otherwise, OSS returns an error.

Assume that you must set two custom parameters x:var1 and x:var2, and the values of the two parameters are value1 and value2 respectively, the JSON format constructed is as follows:

```
{
"x:var1":"value1",
"x:var2":"Value2"
}
```

Construct callback requests

After the callback and callback-var parameters are constructed, you can transmit the parameters to OSS with three methods. The callback parameter is required, and the callback-var parameter is optional. If you configure no custom parameter, the callback-var field does not need to be added. The aforesaid three methods are as follows:

- Including parameters in the URL.
- Including parameters in the header.
- Using form fields to include parameters in the body of a POST request.

  Note:

  You can only use this method to specify the callback parameter when using POST to upload an object.

The three methods are alternative; otherwise, OSS returns an InvalidArgument error.

To include a parameter in OSS request, first you must use Base64 to encode the preceding constructed JSON string, and include the string in OSS request using the methods described as follows:

- To include parameters in the URL, use 'callback=[CallBack]' or 'callback-var=[CallBackVar]' as a URL parameter to send it with the request. When CanonicalizedResource of the signature is calculated, callback, or callback-var is taken into consideration as a sub-resource.
- To include parameters in the header, use 'x-oss-callback=[CallBack]' or 'x-oss-callback-var=[CallBackVar]' as a head to send it with the request. When CanonicalizedOSSHeaders of

the signature is calculated, x-oss-callback-var and x-oss-callback are taken into consideration. An example is provided as follows:

```
PUT /test.txt HTTP/1.1
Host: callback-test.oss-test.aliyun-inc.com
Accept-ncoding: identity
Content-Length: 5
x-oss-callback-var: eyJ4Om15X3ZhciI6ImZvci1jYWxsYmFjay10ZXN0In0=
User-Agent: aliyun-sdk-python/0.4.0 (Linux/2.6.32-220.23.2.ali1089.
el5.x86_64/x86_64;2.5.4)
x-oss-callback: eyJjYWxsYmFja1VybCI6IjEyMS40My4xMTMuODoyMzQ1Ni9pbm
RleC5odG1sIiwgICJjYWxsYmFja0JvZHkiOiJidWNrZXQ9JHtidWNrZXR9Jm
9iamVjdD0ke29iamVjdH0mZXRhZz0ke2V0YWd9JnNpemU9JHtzaXplfSZtaW
1lVHlwZT0ke21pbWVUeXBlfSZpbWFnZUluZm8uaGVpZ2h0PSR7aW1hZ2VJbm
ZvLmhlaWdodH0maW1hZ2VJbmZvLndpZHRoPSR7aW1hZ2VJbmZvLndpZHRofS
ZpbWFnZUluZm8uZm9ybWF0PSR7aW1hZ2VJbmZvLmZvcm1hdH0mbXlfdmFyPS
R7eDpteV92YXJ9In0=
Host: callback-test.oss-test.aliyun-inc.com
Expect: 100-Continue
Date: Mon, 14 Sep 2015 12:37:27 GMT
Content-Type: text/plain
Authorization: OSS mlepou3zr4u7b14:5a74vhd4UXpmyuudV14Kaen5cY4=
Test
```

- It is slightly complicated to include the callback parameter when POST is used to upload an object, because the callback parameter must be included using an independent form field. See the following example:

```
--9431149156168
Content-Disposition: form-data; name="callback"
eyJjYWxsYmFja1VybCI6IjEwLjEwMS4xNjYuMzA6ODA4My9jYWxsYmFjay5w
aHAiLCJjYWxsYmFja0hvc3QiOiIxMC4xMDEuMTY2LjMwIiwiY2FsbGJhY2tC
b2R5IjoiZmlsZW5hbWU9JChmaWxlbmFtZSkmdGFibGU9JHt4OnRhYmxlfSIs
ImNhbGxiYWNrQm9keVR5cGUiOiJhcHBsaWNhdGlvbi94LXd3dy1mb3JtLXVy
bGVuY29kZWQifQ==
```

If custom parameters are used, you cannot directly include the callback-var parameter in the form field. Each custom parameter must be included using an independent form field. For example, if the JSON of a custom parameter is:

```

{
"x:var1":"value1",
"x:var2":"value2"
}
```

The form field of the POST request are as follows:

```
--9431149156168
Content-Disposition: form-data; name="callback"
eyJjYWxsYmFja1VybCI6IjEwLjEwMS4xNjYuMzA6ODA4My9jYWxsYmFjay5w
aHAiLCJjYWxsYmFja0hvc3QiOiIxMC4xMDEuMTY2LjMwIiwiY2FsbGJhY2tC
b2R5IjoiZmlsZW5hbWU9JChmaWxlbmFtZSkmdGFibGU9JHt4OnRhYmxlfSIs
ImNhbGxiYWNrQm9keVR5cGUiOiJhcHBsaWNhdGlvbi94LXd3dy1mb3JtLXVy
bGVuY29kZWQifQ==
--9431149156168
Content-Disposition: form-data; name="x:var1"
value1
--9431149156168
Content-Disposition: form-data; name="x:var2"
value2
```

At the same time, you can add callback conditions in the policy (if callback is not added, upload verification is not performed on this parameter). For example:

```
{ "expiration": "2014-12-01T12:00:00.000Z",
  "conditions": [
    {"bucket": "johnsmith" },
    {"callback": "eyJjYWxsYmFja1VybCI6IjEwLjEwMS4xNjYuMzA6ODA4My9jYW
xsYmFjay5waHAiLCJjYWxsYmFja0hvc3QiOiIxMC4xMDEuMTY2LjMwIiwiY2
FsbGJhY2tCb2R5IjoiZmlsZW5hbWU9JChmaWxlbmFtZSkiLCJjYWxsYmFja0
JvZHlUeXBlIjoiYXBwbGljYXRpb24veC13d3ctZm9ybS11cmxlbmNvZGVkIn0="},
    ["starts-with", "$key", "user/eric/"],
  ]
}
```

Initiate callback requests

If the file is uploaded successfully, OSS uses the POST method to send the specific content to the application server based on the callback parameter and the custom parameters (the callback-var parameter) in the user’s request.

```
POST /index.html HTTP/1.0
Host: 121.43.113.8
Connection: close
Content-Length: 0
Content-Type: application/x-www-form-urlencoded
User-Agent: ehttp-client/0.0.1
bucket=callback-test&object=test.txt&etag=D8E8FCA2DC0F896FD7CB
4CB0031BA249&size=5&mimeType=text%2Fplain&imageInfo.height=&imageInfo.
width=&imageInfo.format=&x:var1=for-callback-test
```

Return callback results

For example, the application server returns the following request for response:

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.3 Python/2.7.6
Date: Mon, 14 Sep 2015 12:37:27 GMT
Content-Type: application/json
Content-Length: 9
{"a":"b"}
```

Return upload results

The following content is sent to the client:

```
HTTP/1.1 200 OK
Date: Mon, 14 Sep 2015 12:37:27 GMT
Content-Type: application/json
```

```
Content-Length: 9
Connection: keep-alive
ETag: "D8E8FCA2DC0F896FD7CB4CB0031BA249"
Server: AliyunOSS
x-oss-bucket-version: 1442231779
x-oss-request-id: 55F6BF87207FB30F2640C548
{"a":"b"}
```

It must be noted that, in the case of requests such as CompleteMultipartUpload, the returned request body includes content (for example, information in XMl format). After using the upload callback function, the original body content is overwritten, such as '"a":"b"'. Take this into consideration for judgment and processing.

Callback signature

When the callback parameter is set, OSS sends the POST callback request to the user's application server based on the callbackUrl set by the user. After receiving the callback request, if you expect the application server to check whether the callback request is initiated by OSS, you can include a signature in the callback request to verify the OSS identity.

- Generate signatures

  The signature occurs at the OSS side, and is signed using the RSA Asymmetric Encryption. You can encrypt the signature using a private key as follows:

  ```
  authorization = base64_encode(rsa_sign(private_key, url_decode(path)
   + query_string + ‘\n’ + body, md5))
  ```

  Instructions: The private_key indicates a private key which is only known to OSS. The path indicates the resource path of the callback request. The query_string indicates a query string . The body indicates the message body of the callback. The signature thus consists of the following steps:

  — Obtain the string to be signed: The resource path URL is decoded, added by the initial query string, a carriage return and the callback message body.
  — RSA signature: Use a private key to sign the expected string. The hashing function for signature is MD5.
  — Use Base64 to encode the signed result to get the final signature. Put the signature in the authorization header of the callback request.

  An example is provided as follows:

  ```
  POST /index.php? id=1&index=2 HTTP/1.0
  Host: 121.43.113.8
  Connection: close
  Content-Length: 18
  ```

```
authorization: kKQeGTRccDKyHB3H9vF+xYMSrmhMZjzzl2/kdD1ktNVgb
WEfYTQG0G2SU/RaHBovRCE8OkQDjC3uG33esH2txA==
Content-Type: application/x-www-form-urlencoded
User-Agent: ehttp-client/0.0.1
x-oss-pub-key-url: aHR0cDovL2dvc3NwdWJsaWMuYWxpY2RuLmNvbS9j
YWxsYmFja19wdWJfa2V5X3YxLnBlbQ==
bucket=yonghu-test
```

The path is `/index.php`, query_string is `? id=1&index=2`, the body is `bucket=yonghu-test`, and the final signature result is `kKQeGTRccDKyHB3H9vF+xYMSrmhMZjzzl2/kdD1ktNVgbWEfYTQG0G2SU/RaHBovRCE8OkQDjC3uG33esH2txA==`.

- Verify signature

  Signature verification is an inverse process of signature. The signature is verified by the application server, and the process is as follows:

```
Result = rsa_verify(public_key, md5(url_decode(path) + query_string
 + ‘\n’ + body), base64_decode(authorization))
```

The fields have the same meanings as described during the signature process. The public_key indicates a public key. The authorization indicates the signature in the callback header. The signature verification consists of the following steps:

1. The x-oss-pub-key-url header of the callback request stores the Base64-encoded URL of the public key. The header must be decoded with Base64 to obtain the public key as follows:

   ```
   public_key = urlopen(base64_decode(x-oss-pub-key-url header))
   ```

   It must be noted that, the value of the `x-oss-pub-key-url` header must start with `http://gosspublic.alicdn.com/` or `https://gosspublic.alicdn.com/`, so as to make sure that the public key is provided by OSS.
2. Obtain the Base64-decoded signature

   ```
   signature = base64_decode(Value of the authorization header)
   ```
3. Obtain the string to be signed the same way as described in the signature process.

   ```
   sign_str = url_decode(path) + query_string + ‘\n’ + body
   ```
4. Verify the signature

   ```
   result = rsa_verify(public_key, md5(sign_str), signature)
   ```

The preceding sample is used as an example:

1. Obtain the URL of the public key, that is, with Base64 decoding the `aHR0cDovL2dvc3NwdWJsaWMuYWxpY2RuLmNvbS9jYWxsYmFja19wdWJfa2V5X3YxLnBlbQ==` to `http://gosspublic.alicdn.com/callback_pub_key_v1.pem`.
2. The signature header `kKQeGTRccDKyHB3H9vF+xYMSrmhMZjzzl2/kdD1ktNVgbWEfYTQG0G2SU/RaHBovRCE8OkQDjC3uG33esH2txA==` is decoded with Base64 (The decoded result cannot be displayed because it is a nonprintable string).
3. Obtain the string to be singed, that is, url_decode("index.php") + "?id=1&index=2" + "\n" + " bucket=yonghu-test" . Then perform the MD5 check.
4. Verify the signature

- Application server example

  Python is used as an example to demonstrate how an application server verifies a signature. In this example, the M2Crypto library must be installed.

```
import httplib
import base64
import md5
import urllib2
from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
from M2Crypto import RSA
from M2Crypto import BIO
def get_local_ip():
    try:
        csock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        csock.connect(('8.8.8.8', 80))
        (addr, port) = csock.getsockname()
        csock.close()
        return addr
    except socket.error:
        return ""
class MyHTTPRequestHandler(BaseHTTPRequestHandler):

    def log_message(self, format, *args):
        return

    def do_POST(self):
        #get public key
        pub_key_url = ''
        try:
            pub_key_url_base64 = self.headers['x-oss-pub-key-url']
            pub_key_url = pub_key_url_base64.decode('base64')
            if not pub_key_url.startswith("http://gosspublic.alicdn.
com/") and not pub_key_url.startswith("https://gosspublic.alicdn.com
/"):
                self.send_response(400)
                self.end_headers()
                return
            url_reader = urllib2.urlopen(pub_key_url)
            #you can cache it
            pub_key = url_reader.read()
        except:
            print 'pub_key_url : ' + pub_key_url
```

```
                print 'Get pub key failed!'
                self.send_response(400)
                self.end_headers()
                return
            #get authorization
            authorization_base64 = self.headers['authorization']
            authorization = authorization_base64.decode('base64')
            #get callback body
            content_length = self.headers['content-length']
            callback_body = self.rfile.read(int(content_length))
            #compose authorization string
            auth_str = ''
            pos = self.path.find('?')
            if -1 == pos:
                auth_str = urllib2.unquote(self.path) + '\n' +
callback_body
            else:
                auth_str = urllib2.unquote(self.path[0:pos]) + self.path
[pos:] + '\n' + callback_body
            print auth_str
            #verify authorization
            auth_md5 = md5.new(auth_str).digest()
            bio = BIO.MemoryBuffer(pub_key)
            rsa_pub = RSA.load_pub_key_bio(bio)
            try:
                result = rsa_pub.verify(auth_md5, authorization, 'md5')
            except:
                result = False
            if not result:
                print 'Authorization verify failed!'
                print 'Public key : %s' % (pub_key)
                print 'Auth string : %s' % (auth_str)
                self.send_response(400)
                self.end_headers()
                return
            #do something accoding to callback_body
            #response to OSS
            resp_body = '{"Status":"OK"}'
            self.send_response(200)
            self.send_header('Content-Type', 'application/json')
            self.send_header('Content-Length', str(len(resp_body)))
            self.end_headers()
            self.wfile.write(resp_body)
class MyHTTPServer(HTTPServer):
    def __init__(self, host, port):
        HTTPServer.__init__(self, (host, port), MyHTTPRequestHandler
)
if '__main__' == __name__:
    server_ip = get_local_ip()
server_port = 23451
server = MyHTTPServer(server_ip, server_port)
server.serve_forever()
```

Application servers implemented in other languages are as follows:

Java version:

- Download address: *click here*.

- Running method: Extract the package and run `java -jar oss-callback-server-demo.jar 9000` (9000 is the port number and can be designated as needed)

PHP version:

- Download address: *click here*
- Running method: Deploy the program to an Apache environment. The characteristics of the PHP language determine that the environment is depended on to retrieve some headers. You may see the example to make modifications to your own environment.

Python version:

- Download address: *click here*
- Running method: Extract the package and directly run `python callback_app_server.py`. You must install RSA dependencies to run this program.

C # version:

- Download address: *click here*
- Running method: Extract the package and see `README.md`.

.NET version:

- Download address: *click here*
- Running method: Extract the package and see `README.md`.

Go version:

- Download address: *click here*
- Running method: Extract the package and see `README.md`.

Ruby version:

- Download address: *click here*
- Running method: ruby aliyun_oss_callback_server.rb

Special instructions

- If the input callback parameter or callback-var parameter is invalid, a 400 error is returned, with the error code of “InvalidArgument”. Invalid situations include the following:
  - In the PutObject() and CompleteMultipartUpload() interfaces, the callback(x-oss-callback) or callback-var(x-oss-callback-var) parameters are input at the same time to the URL and header fields.

    - The callback or callback-var parameter is too long (over 5KB). PostObject() is not subject to this restriction because callback-var parameter is not used, and this is true for the following as well.
    - Callback or callback-var is not Base64 encoded.
    - After Base64 decoding, the callback or callback-var parameter is not in a valid JSON format.
    - After callback parameter resolution, the callbackUrl field contains more than 5 URLs, or the input port in the URL is invalid, such as `{"callbackUrl":"10.101.166.30:test", " callbackBody":"test"}`
    - After callback parameter resolution, the callbackBody field is blank.
    - After callback parameter resolution, the callbackBodyType field value is not “application/x-www-form-urlencoded” or “application/json”.
    - After callback parameter resolution, the callbackBody field contains invalid formats of variables. The valid format is `${var}`
    - After callback-var parameter resolution, the format is not the expected JSON format. The expected format is:`{"x:var1":"value1","x:var2":"value2"...}`
- If a callback fails, the system returns a 203 error, with the error code “CallbackFailed”. A callback failure only indicates that OSS did not receive the expected callback response (for example, the response from the application server was not in the JSON format), not that the application server did not receive the callback request. In addition, by this time, the file has been successfully uploaded to OSS.
- The response returned by the application server to OSS must contain the Content-Length header, and the size of the body cannot exceed 1 MB.

Regions used in Callback

Currently, callback only supports the following regions: China North 2 (Beijing), China East 1 (Hangzhou), China North 1 (Qingdao), China East 2 (Shanghai), Shanghai Financial Cloud, China South 1 (Shenzhen), Hong Kong, China North 5 (huhehaote), China North 3 (zhangjiakou), Middle East 1 (Dubai), Asia Pacific NE 1 (Tokyo), EU Central 1 (Frankfurt), Asia Pacific SE 1 (Singapore), US East 1 (Virginia), US West 1 (Silicon Valley), Asia Pacific SE 2 (Sydney) and Asia Pacific SE 3 (Kuala Lumpur).

3.5 SelectObject (in beta phase)

Introduction

Object Storage Service (OSS) built on Alibaba Cloud's Apsara distributed system is a massive, secure, and highly reliable cloud storage solution that offers low cost storage accessible anywhere in the world. OSS possesses excellent scaling abilities for storage capacity and processes, and supports RESTful APIs. Not only can OSS store media files, but it can also be utilized as a data warehouse for massive data file storage. OSS can seamlessly integrate with Hadoop 3.0, and services that are run on EMR (such as Spark/Hive/Presto, MaxCompute, HybridDB and the newly-released Data Lake Analytics) support data processing and retrieval directly from OSS.

However, the current GetObject interface provided by OSS determines that the big data platform can only download all OSS data locally and then for analysis and filter. A lot of bandwidth and client resources are wasted in querying scenarios.

To address this problem, the SelectObject interface is provided. This method allows big data platforms to access OSS to perform basic filtering on data through conditions and Projection, and return useful data only to the big data platform. In this way, the bandwidth and the amount of data processed at the client-side is greatly reduced, making OSS-based data warehousing and data analysis a highly attractive option.

SelectObject is now in beta phase, and provides Java and Python SDKs. SelectObject supports CSV files of RFC 4180 standard to be encoded as UTF-8 (including Class CSV files such as TSV, row and column separators of the file and customizable Quote characters). SelectObject supports files in standard and low frequency access storage types, and encrypted files, which are fully managed by OSS (or CMK managed by KMS).

The supported SQL syntax is as follows:

- SQL statements: Select From Where
- Data Type: String, Int (64bit), float (64bit), Timestamp, and Boolean
- Operation: Logical condition (AND, OR, NOT), Arithmetic Expression (+-*/%), Comparison operation (>,=, <, >=, <=, ! =), and String operation (LIKE, ||)

The sharding mechanism of SelectObject is similar to the shard download mechanism of GetObject, and includes two sharding methods: sharding by row and sharding by Split. Sharding by row is a common method, but it results in uneven load balancing of sparse data. Sharding by Split is more efficient than sharding by row as a Split contains multiple rows of data, and the data size of each Split is roughly equal, which enables better load balancing performance. Additionally

, byte-based sharding (provided by GetObject) may corrupt data. Therefore, sharding by Split is recommended for CSV data.

CSV data in OSS is String type by default. Users can use CAST function to convert data. For example, the following SQL query converts _1 and _2 into Int and compares them.

```
Select * from OSSObject where cast (_1 as int) > cast(_2 as int)
```

Furthermore, SelectObject supports implicit conversion in WHERE condition, such as the first and the second columns in the following statement will be converted to Int:

```
Select _1 from ossobject where _1 + _2 > 100
```

Description of RESTful API

Execute the SQL statement on the target CSV files and the execution results will be returned. At the same time, the command will automatically save the metadata information of the CSV files, such as the total number of rows and columns.

The API returns a 206 response when the SQL statement is executed correctly. If the SQL statement is incorrect or does not match the CSV files, a 400 error response will be returned.

Request syntax

```
POST /object? x-oss-process=csv/select HTTP/1.1
HOST: BucketName.oss-cn-hangzhou.aliyuncs.com
Date: time GMT
Content-Length: ContentLength
Content-MD5: MD5Value
Authorization: Signature

<? xml version="1.0" encoding="UTF-8"? >
<SelectRequest>
 Base64 encode (select * From ossobject where)
 <InputSerialization>
  <CompressionType>None</CompressionType>
  <CSV>
   <FileHeaderInfo>NONE|IGNORE|USE</FileHeaderInfo>
   <RecordDelimiter>base64 encode</RecordDelimiter>
   <FieldDelimiter>base64 encode</FieldDelimiter>
   <QuoteCharacter>base64 encode</QuoteCharacter>
   <CommentCharacter>base64 encode</CommentCharacter>
   <Range>line-range=start-end|split-range=start-end</Range>
  </CSV>
  </InputSerialization>
  <OutputSerialization>
        <CSV>
   <RecordDelimiter>base64 encode</RecordDelimiter>
   <FieldDelimiter>base64 encode</FieldDelimiter>
   <KeepAllColumns>false|true</KeepAllColumns>
  </CSV>
 <OutputRawData>false|true</OutputRawData>
  </OutputSerialization>
```

```
</SelectRequest>
```

Name	Type	Description
SelectRequest	Container	The container for storing Select requests Child node: Expression, InputSerialization, OutputSerialization Parent node: None
Expression	String	The SQL statement encoded in Base64 Child nodes: None Parent node: SelectRequest
InputSerialization	Container	Input serialized parameters (optional) Child node: CompressionType, CSV Parent node: SelectRequest
OutputSerialization	Container	Output serialized parameters (optional) Child node: CSV, OutputRawData Parent node: SelectRequest
CSV(InputSerialization)	Container	Input CSV-formatted parameters (optional) Child node: FileHeaderInfo, RecordDelimiter, FieldDelimiter, QuoteCharacter, CommentCharacter, Range Parent node: InputSerialization
CSV(OutputSerialization)	Container	Output CSV-formatted parameters (optional) Child node: RecordDelimiter, FieldDelimiter, KeepAllColumns Parent node: OutputSerialization
OutputRawData	bool, default: false	Specifies output data as raw data, not Frame-based data (optional) Child node: None Parent node: OutputSerialization
CompressionType	Enumeration	Specifies file compression types. It can only be None as file compression is currently not supported Child node: None Parent node: InputSerialization
FileHeaderInfo	Enumeration	Specifies CSV files header information (optional) Value: • Use: The CSV file contains header information, and the CSV column name can be used as the column name in the Select.

Name	Type	Description
		• Ignore: The CSV file contains header information, but the CSV column name can not be used as the column name in the Select. • None: The CSV file contains no header information, and the value can be default. Child node: None Parent node: CSV(input)
RecordDelimiter	String	Specifies line breaks for a CSV, encoded in Base64. The default value is\n (optional). The value before decoding is at most two characters, expressed as an ANSI character. \n used in Java indicates a line break. Child node: None Parent node: CSV (input, output)
FieldDelimiter	String	Specifies the CSV column separator, encoded in Base64. The default is , (optional) The value before decoding must be expressed as an ANSI character , used in Java indicates a comma. Child Node: None Parent node: CSV (input and output)
QuoteCharacter	String	Specifies the quote character of the CSV, encoded in Base64. The default value is \″ (optional). Inside the CSV quotes, the column separator is treated as a normal character. The value before encoding must be expressed as an ANSI character, such as \″in Java indicates quotation marks. Child node: None Parent node: CSV (input)
CommentCharacter	String	Specifies the CSV comment character, encoded in Base464. The default value is # (optional)
Range	String	Specifies the scope of the query file (optional). Two formats are supported: • Query by row: line-range=start-end • Query by Split: split-range=start-end

Name	Type	Description
		Both start and end are inclusive. The format is the same as the range parameter in range get. Child node: None Parent node: CSV (input)
KeepAllColumns	Bool	Specifies the location in the response result that contains all of the CSV columns (optional, and default value is false). However, only the columns in the select statement contain values , otherwise they are empty. The data of each row in the response result will be sorted in ascending order of CSV columns. Take the following statement as example: `select _5, _1 from ossobject.` If the value of KeepAllColumn is true, with six columns of data in total, the returned data is as follows: Value of 1st column ...Value of 5th column,\n Child node: None Parent node: CSV(output)

Response results

The request results are returned as a Frame. The format of each frame is as follows, where checksum is CRC32:

Frame-Type | Payload Length | Header Checksum | Payload | Payload Checksum

<---4 bytes--><---4 bytes----------><-------4 bytes-------><variable><----4bytes---------->

There are three different frame types, which are as follows:

Name	Frame-Type Value	Payload format	Description
Data Frame	version | 8388609 <--1 byte><--3 bytes>	scanned size | data <-8 bytes----------><---variable-> The scanned size is the size of the scanned data, and the data is the data returned from the query.	Data Frame is used to return the query data and report its current progress at the same time.

Name	Frame-Type Value	Payload format	Description
Continuous Frame	version | 8388612 <--1 byte><--3 bytes->	scanned size <----8 bytes-->	Continuous Frame is used to report current progress and maintain HTTP connections. If the query does not return data within 5s, a Continuous Frame will be returned.
End Frame	version | 8388613	Offset | total scanned bytes | http status code | error message <--8bytes-><--8bytes --------------><----4 bytes--------><-variable ------> Where offset is the final location offset after scanning and total scanned bytes is the total bytes of all scanned data. http status code is the final processing result. and error message is the error message itself.	The reason it returns the status code is that when the SelectObject is streamed, only the first block is processed when the Response Header is sent. If the first block of data and SQL match, the Status in the Response Header is a 206 response, but if the following data is illegal, the Status in the Header cannot be changed, and the final Status and Error message includes only the End Frame. Therefore, the client should treat it as the final result.

Example request

```
POST /oss-select/bigcsv_normal.csv? x-oss-process=csv%2Fselect HTTP/1.
1
Date: Fri, 25 May 2018 22:11:39 GMT
Content-Type:
Authorization: OSS LTAIJPXxMLocA0fD:FC/9JRbBGRw4o2QqdaL246Pxuvk=
User-Agent: aliyun-sdk-dotnet/2.8.0.0(windows 16.7/16.7.0.0/x86;4.0.
30319.42000)
Content-Length: 748
Expect: 100-continue
Connection: keep-alive
Host: host name
```

```
<? xml version="1.0"? >
<SelectRequest>
 <Expression>c2VsZWN0IGNvdW50KCopIGZyb20gb3Nzb2JqZWN0IHdoZXJlIF
80ID4gNDU=
 </Expression>
 <InputSerialization>
  <Compression>None</Compression>
  <CSV>
   <FileHeaderInfo>Ignore</FileHeaderInfo>
   <RecordDelimiter>Cg==</RecordDelimiter>
   <FieldDelimiter>LA==</FieldDelimiter>
   <QuoteCharacter>Ig==</QuoteCharacter>
   <Comments>Iw==</Comments>
  </CSV>
 </InputSerialization>
 <OutputSerialization>
  <CSV>
   <RecordDelimiter>Cg==</RecordDelimiter>
   <FieldDelimiter>LA==</FieldDelimiter>
   <QuoteCharacter>Ig==</QuoteCharacter>
   <KeepAllColumns>false</KeepAllColumns>
 </CSV>
 <OutputRawData>false</OutputRawData>
 </OutputSerialization>
</SelectRequest>
```

SQL statement regex

`SELECT select-list from OSSObject where_opt limit_opt`

The keywords SELECT, OSSOBJECT and WHERE cannot be changed.

```
select_list: column name

| column index (for example, _1, _2)

| function(column index | column name)

| select_list AS alias
```

The supported functions are AVG, SUM, MAX, MIN, COUNT, and CAST (type conversion function). Only * can be used after COUNT.

```
Where_opt:
| WHERE expr
expr:
| literal value
| column name
| column index
| expr op expr
| expr OR expr
| expr AND expr
| expr IS NULL
| expr IS NOT NULL
| expr IN (value1, value2,….)
| expr NOT in (value1, value2,…)
| expr between value1 and value2
| NOT (expr)
| expr op expr
```

```
 | (expr)
 | cast (column index or column name or literal as INT|DOUBLE|DATETIME)
```

op: includes > < >= <= ! = =, LIKE，+-*/%, and connection string ||.

cast: Cast can only be one type for the same column.

`limit_opt:`

| limit Integer

Mixing of aggregations and limit

`Select avg(cast(_1 as int)) from ossobject limit 100`

In the preceding statement, the AVG value of the first column in the first 100 rows is calculated . This statement is different from what MYSQL outputs, as aggregation in SelectObject always returns only one row of data, so there is no need to limit its output volumes. Therefore, the limit in SelectObject will be executed before the Aggregate function.
SQL statement restrictions are as follows:

- Only UTF-8 encoded text files are supported. Uncompressed GZIP text files can be processed . Support for processing compressed files is coming soon.
- Only single file queries are supported, not JOIN, ORDER BY, GROUP BY, and HAVING
- Not contains aggregation conditions in WHERE statement. For example, where max(cast(age as int)) > 100 is not allowed.
- Up to 1000 columns are supported and the maximum column name is 1024.
- Up to 5% wildcards are supported in the LIKE statement. * and % are equivalent, representing 0 or multiple arbitrary characters.
- Up to 1024 constant items are supported in the IN statement.
- The Projection after Select can be a column name, a column index (_1, _2, etc.), an aggregate function, or a CAST function. Other expressions are not supported Like select _ 1 + _2 from ossobject is not allowed.
- The length of maximum row and column are both 256 KB.

CreateSelectObjectMeta

CreateSelectObjectMeta API is used to obtain information about the target CSV file, such as the total number of rows, the total number of columns, and the number of Splits. If the information does not exist in the file, the whole CSV file is scanned for the preceding information. If the API executes correctly, a 200 response is returned. If the target CSV file is illegal or the specified delimiter does not match the target CSV file, a 400 error response is returned.

Request elements

Name	Type	Description
CsvMetaRequest	Container	Saves the container that created Select Meta requests Child node: Expression, InputSerialization, OutputSerialization Parent node: None
InputSerialization	Container	Inputs serialized parameters (optional) Child node: CompressionType, CSV Parent node: CsvMetaRequest
OverwriteIfExists	Bool	Recalculates SelectMeta to overwrite existing data (optional, the default value is false. If Select Meta already exists, then Select Meta is returned.) Child node: None Parent node: CsvMetaRequest
CompressionType	Enumeration	Specifies file compression types. It can only be None as file compression is currently not supported. Child node: None Parent node: InputSerialization
RecordDelimiter	String	Specifies line breaks for a CSV, encoded in Base64. The default value is \n (optional). The value before decoding is at most two characters, expressed as an ANSI character. \n used in Java indicates a line break. Child node: None Parent node: CSV
FieldDelimiter	String	Specifies the CSV column separator, encoded in Base64. The default value is ， (optional). The value before decoding must be expressed as an ANSI character. ， used in Java indicates a comma. Child node: None Parent node: CSV (input and output)
QuoteCharacter	String	Specifies the CSV quote character, encoded in Base64. The default value is \″ (optional). Line breaks in quotation marks in CSV, column separators will be treated as normal characters. The value before decoding must be

Name	Type	Description
		expressed as an ANSI character. \” used in Java indicates a comma. Child node: None Parent node: CSV (input)
CSV	Container	Specifies CSV input format Child node: RecordDelimiter, FieldDelimiter, QuoteCharacter Parent node: InputSerialization

Response Body: empty

Response Header :

- x-oss-select-csv-lines: total number of rows
- x-oss-select-csv-columns: total number of columns
- x-oss-select-csv-splits: total number of Splits
- content-length: file content length

Note:

X-OSS-select-CSV-columns refers to the number of columns in the first row, assuming that the data in the first row is correct.

Example request

```
POST /oss-select/bigcsv_normal.csv? x-oss-process=csv%2Fmeta HTTP/1.1
Date: Fri, 25 May 2018 23:06:41 GMT
Content-Type:
Authorization: OSS LTAIJPXxMLocA0fD:2WF2l6zozf+hzTj9OSXPDklQCvE=
User-Agent: aliyun-sdk-dotnet/2.8.0.0(windows 16.7/16.7.0.0/x86;4.0.
30319.42000)
Content-Length: 309
Expect: 100-continue
Connection: keep-alive
Host: Host

<? xml version="1.0"? >
<CsvMetaRequest>
 <InputSerialization>
  <CSV>
   <RecordDelimiter>Cg==</RecordDelimiter>
   <FieldDelimiter>LA==</FieldDelimiter>
   <QuoteCharacter>Ig==</QuoteCharacter>
  </CSV>
 </InputSerialization>
 <OverwriteIfExisting>false</OverwriteIfExisting>
```

```
</CsvMetaRequest>
```

Response code

```
HTTP/1.1 200 OK
Server: AliyunOSS
Date: Fri, 25 May 2018 23:06:42 GMT
Content-Type: application/vnd.ms-excel
Content-Length: 0
Connection: close
x-oss-request-id: 5B089702461FB4C07B000C75
x-oss-location: oss-cn-hangzhou-a
x-oss-access-id: LTAIJPXxMLocA0fD
x-oss-sign-type: NormalSign
x-oss-object-name: bigcsv_normal.csv
Accept-Ranges: bytes
ETag: "3E1372A912B4BC86E8A51234AEC0CA0C-400"
Last-Modified: Wed, 09 May 2018 00:22:32 GMT
x-oss-object-type: Multipart
x-oss-bucket-storage-type: standard
x-oss-hash-crc64ecma: 741622077104416154
x-oss-storage-class: Standard
**x-oss-select-csv-rows: 54000049**
**x-oss-select-csv-columns: 4**
**x-oss-select-csv-splits: 960**
```

Python SDK

```
import os
import oss2

def select_call_back(consumed_bytes, total_bytes = None):
 print('Consumed Bytes:' + str(consumed_bytes) + '\n')

# First, initialize the information such as AccessKeyId, AccessKeyS
ecret, and Endpoint.
# Obtain the information through environment variables or replace the
information such as“<yourAccessKeyId>” with the real AccessKeyId, and
so on.
#
# Use Hangzhou region as an example. Endpoint can be:
# http://oss-cn-hangzhou.aliyuncs.com
# https://oss-cn-hangzhou.aliyuncs.com

access_key_id = os.getenv('OSS_TEST_ACCESS_KEY_ID', '<yourAccessKeyId
>')
access_key_secret = os.getenv('OSS_TEST_ACCESS_KEY_SECRET', '<
yourAccessKeySecret>')
bucket_name = os.getenv('OSS_TEST_BUCKET', '<yourBucket>')
endpoint = os.getenv('OSS_TEST_ENDPOINT', '<yourEndpoint>')

# Create a bucket instance, all object-related methods need to be
called through the bucket instance.
bucket = oss2. Bucket(oss2. Auth(access_key_id, access_key_secret),
endpoint, bucket_name)
key = 'python_select.csv'
content = 'Tom Hanks,USA,45\r\n'*1024
filename = 'python_select.csv'
# Upload files
bucket.put_object(key, content)
```

```
csv_meta_params = {'CsvHeaderInfo': 'None',
'RecordDelimiter': '\r\n'}
select_csv_params = {'CsvHeaderInfo': 'None',
'RecordDelimiter': '\r\n',
'LineRange': (500, 1000)}

csv_header = bucket.create_select_object_meta(key, csv_meta_params)
print(csv_header.csv_rows)
Print(csv_header.csv _ splits)
result = bucket.select_object(key, "select * from ossobject where _3
 > 44 limit 100000", select_call_back, select_csv_params)
content_got = b''
for chunk in result:
 content_got += chunk
print(content_got)

result = bucket.select_object_to_file(key, filename,
"select * from ossobject where _3 > 44 limit 100000", select_call_back
, select_csv_params)

bucket.delete_object(key)
```

Java SDK

```
package samples;

import com.aliyun.oss.event.ProgressEvent;
import com.aliyun.oss.event.ProgressListener;
import com.aliyun.oss.model.*;
import com.aliyun.oss.OSS;
Import com. aliyun. OSS;

import java.io.BufferedOutputStream;
import java.io.ByteArrayInputStream;
import java.io.FileOutputStream;

/**
 * Examples of create select object metadata and select object.
 *
 */
public class SelectObjectSample {
    private static String endpoint = "<endpoint, http://oss-cn-
hangzhou.aliyuncs.com>";
    private static String accessKeyId = "<accessKeyId>";
    private static String accessKeySecret = "<accessKeySecret>";
    private static String bucketName = "<bucketName>";
    private static String key = "<objectKey>";

    public static void main(String[] args) throws Exception {
        OSS client = new OSSClientBuilder().build(endpoint, accessKeyI
d, accessKeySecret);
        String content = "name,school,company,age\r\n" +
                "Lora Francis,School A,Staples Inc,27\r\n" +
                "Eleanor Little,School B,\"Conectiv, Inc\",43\r\n" +
                "Rosie Hughes,School C,Western Gas Resources Inc,44\r\
n" +
                "Lawrence Ross,School D,MetLife Inc.,24";

        client.putObject(bucketName, key, new ByteArrayInputStream(
content.getBytes()));
```

```
        SelectObjectMetadata selectObjectMetadata = client.createSele
ctObjectMetadata(
                new CreateSelectObjectMetadataRequest(bucketName, key)
                        .withInputSerialization(
                                new InputSerialization().withCsvInp
utFormat(
                                        new CSVFormat().withHeaderInfo
(CSVFormat.Header.Use).withRecordDelimiter("\r\n"))));
        System.out.println(selectObjectMetadata.getCsvObjectMetadata
().getTotalLines());
        System. Out. println (selectobjectmetadata. getcsvobje
ctmetadata (). getshares ());

        SelectObjectRequest selectObjectRequest =
                new SelectObjectRequest(bucketName, key)
                        .withInputSerialization(
                                new InputSerialization().withCsvInp
utFormat(
                                        new CSVFormat().withHeaderInfo
(CSVFormat.Header.Use).withRecordDelimiter("\r\n")))
                        .withOutputSerialization(new OutputSeri
alization().withCsvOutputFormat(new CSVFormat()));
        selectObjectRequest.setExpression("select * from ossobject
where _4 > 40");
        OSSObject ossObject = client.selectObject(selectObjectRequest
);
        // read object content from ossObject
        BufferedOutputStream outputStream = new BufferedOutputStream(
new FileOutputStream("result.data"));
        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = ossObject.getObjectContent().read(buffer
)) ! = -1) {
            outputStream.write(buffer, 0, bytesRead);
        }
        outputStream.close();
    }
}
```

Best practices

If you want to perform Shard-Query on a massive file, we recommend that you:

1. Call the Create Select Object Meta API to get the total number of Splits for the file. If the file needs to call the SelectObject API, this API makes asynchronous calls before the query, which reduces scan time.
2. Select the appropriate concurrency n based on client-side resources, and divide the total number of Splits by the concurrency n to get the number of Splits that each shard query should contain.
3. Perform the Shard-Query in a form of split-range=1-20 in request body.
4. Merge the results if required.

Use SelectObject with Normal type files. Files of Multipart and Appendable types are not recommended due to poor performance caused by differences in their internal structure.

4 Access control

5 Multipart upload operations

5.1 UploadPartCopy

`UploadPartCopy` uploads a part by copying data from an existing object.

You can add an x-oss-copy-source header in the Upload Part request to call the Upload Part Copy interface. When copying a file larger than 1 GB, you must use the Upload Part Copy method. For the Upload Part Copy operation, the source bucket and the target bucket must be in the same region. If you want to copy a file that is less than 1 GB by a single operation, you can use the Copy Object method.

Request syntax

```
PUT /ObjectName? partNumber=PartNumber&uploadId=UploadId HTTP/1.1
Host: BucketName.oss-cn-hangzhou.aliyuncs.com
Date: GMT Date
Content-Length: Size
Authorization: SignatureValue
x-oss-copy-source: /SourceBucketName/SourceObjectName
x-oss-copy-source-range:bytes=first-last
```

Request header

Except the common request header, other headers in the Upload Part Copy request are used to specify the address of the copied source object and copying range.

Name	Type	Description
`x-oss-copy-source`	String	Specifies the copy source address (the requester must have the permission to read the source object). Default: None
`x-oss-copy-source-range`	Integer	Copying range of the copied source object. For example, if the range is set to bytes = 0-9, the system transfers byte 0 to byte 9. This request header is not required when the entire source object is copied. Default: None

The following request header is used for the source objects specified by x-oss-copy-source.

Name	Type	Description
`x-oss-copy-source-if-match`	String	If the ETag value of the source object is equal to the ETag value provided by the user, the system performs the Copy Object operation; otherwise, the system returns the 412 Precondition Failed error. Default: None
`x-oss-copy-source-if-none-match`	String	If the source object has not been modified since the time specified by the user, the system performs the Copy Object operation; otherwise, the system returns the 412 Precondition Failed error. Default: None
`x-oss-copy-source-if-unmodified-since`	String	If the time specified by the received parameter is the same as or later than the modification time of the file, the system transfers the file normally, and returns 200 OK; otherwise, the system returns the 412 Precondition Failed error. Default: None
`x-oss-copy-source-if-modified-since`	String	If the source object has been modified since the time specified by the user, the system performs the Copy Object operation; otherwise, the system returns the 412 Precondition Failed error. Default: None

Response elements

Name	Type	Description
`x-oss-copy-source-if-match`	String	If the ETag value of the source object is equal to the ETag value provided by the user, the system performs the Copy Object operation; otherwise, the system returns the 412 Precondition Failed error. Default: None
`x-oss-copy-source-if-none-match`	String	If the source object has not been modified since the time specified by the user, the system performs the Copy Object operation; otherwise, the system returns the 412 Precondition Failed error. Default: None
`x-oss-copy-source-if-unmodified-since`	String	If the time specified by the received parameter is the same as or later than the modification time of the file, the system transfers the file normally, and returns 200 OK; otherwise, the system returns the 412 Precondition Failed error. Default: None
`x-oss-copy-source-if-modified-since`	String	If the source object has been modified since the time specified by the user, the system performs the Copy Object operation; otherwise, the system returns the 412 Precondition Failed error. Default: None

Detail analysis

- Before calling the InitiateMultipartUpload interface to upload a part of data, you must call this interface to obtain an Upload ID issued by the OSS server.

- In the Multipart Upload mode, besides the last part, all other parts must be larger than 100 KB . However, the Upload Part interface does not immediately verify the size of the uploaded part (because it cannot immediately determine which part is the last one). It verifies the size of the uploaded part only when Multipart Upload is completed.
- If the x-oss-copy-source-range request header is not specified, the entire source object is copied. If the request header is specified, the returned message includes the length of the entire file and the COPY range. For example, if the returned message is Content-Range: bytes 0-9/44, which means that the length of the entire file is 44, and the COPY range is 0 to 9. If the specified range does not conform to the range rules, OSS copies the entire file and does not contain Content-Range in the result.
- If the x-oss-server-side-encryption request header is specified when the InitiateMultipartUpload interface is called, OSS encrypts the uploaded part and return the x-oss-server-side-encryption header in the Upload Part response header. The value of x-oss-server-side-encryption indicates the server-side encryption algorithm used for this part. For more information, see the InitiateMultipartUpload API.
- This operation cannot be used to copy objects created by Append Object.
- If the bucket type is Archive, you cannot call this interface; otherwise, the system returns Error 400 with the error code “OperationNotSupported”.

Example

Request example:

```
PUT /multipart.data? Partnumber = 1 & sealadid = porterhttp/1.1
Host: oss-example.oss-cn-hangzhou.aliyuncs.com
Content-Length：6291456
Date: Wed, 22 Feb 2012 08:32:21 GMT
Authorization: OSS qn6qrrqxo2oawuk53otfjbyc:J/lICfXEvPmmSW86bBAfMm
UmWjI=
x-oss-copy-source: /oss-example/ src-object
x-oss-copy-source-range:bytes=100-6291756
```

Response example:

```
HTTP/1.1 200 OK
Server: AliyunOSS
Connection: keep-alive
x-oss-request-id: 3e6aba62-1eae-d246-6118-8ff42cd0c21a
Date: Thu, 17 Jul 2014 06:27:54 GMT'
<? xml version="1.0" encoding="UTF-8"? >
<CopyPartResult xmlns=”http://doc.oss-cn-hangzhou.aliyuncs.com”>
    <LastModified>2014-07-17T06:27:54.000Z </LastModified>
    <ETag>"5B3C1A2E053D763E1B002CC607C5A0FE"</ETag>
</CopyPartResult>
```