

Alibaba Cloud ApsaraVideo for Media Processing

Best Practices

Issue: 20181025

Legal disclaimer

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








1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.
5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade

secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).

6. Please contact Alibaba Cloud directly if you discover any errors in this document.

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 <code>cd /d C:/windows</code> command to enter the Windows system folder.
<i>Italics</i>	It is used for parameters and variables.	<code>bae log list --instanceid Instance_ID</code>
[] or [a b]	It indicates that it is a optional value, and only one item can be selected.	<code>ipconfig [-all -t]</code>
{ } or {a b}	It indicates that it is a required value, and only one item can be selected.	<code>swich {stand / slave}</code>

Contents

Legal disclaimer.....	I
Generic conventions.....	I
1 Transcoding.....	1
1.1 Simple transcoding.....	1
1.2 API transcoding.....	3
1.3 Workflow transcoding.....	4
2 Encryption.....	6
2.1 HLS encryption and play.....	6
3 Upload videos.....	10
4 Watermarks.....	17
5 Screenshots.....	21
6 Splicing and cutting.....	25
6.1 Splicing and simple cutting.....	25
6.2 Opening and ending scenes.....	26
7 Package.....	31
7.1 HLS package.....	31
7.2 DASH packaging.....	39
7.3 Creating an HLS package workflow.....	47

1 Transcoding

1.1 Simple transcoding

Introduction

Transcoding refers to the process to process an OSS input file according to specified parameters, and output the result to specified OSS file. When submitting a [transcoding task](#), note the following objects:

- [Input](#)

Specify an OSS input file.



Note:

The Location of OSS must correspond to the region of MPS. For example, the oss-cn-hangzhou of OSS corresponds to the cn-hangzhou of MPS.

- [Output](#)

Several output objects can be specified for each transcoding task, which includes multiple parameters and subobjects. Three important parameters and subobjects are introduced as follows:

- [Container](#)

The output container type (file format). The video supports mp4, flv, ts, and m3u8, and the audio supports mp3 and mp4.

- [Video](#)

The output video parameter, for example, the codec format, bitrate, width, height, and frame rate.

- [Audio](#)

The output audio parameter, for example, the audio codec format, bitrate, channels, and samplerate.

- [TemplateId](#)

Parameters specified through API have higher priority than parameters set by templates, and overwrite the corresponding parameters set in the templates.

MPS provides [Preset static templates](#).

Also, you can [Custom transcoding template](#).

- PipelineId

Each region provides an MPS queue. You can log on to the [MPS console](#), click **Library** > **Settings** > **MPS queue** to query.

Scenarios

The video in any format is transcoded to MP4 video file with the definition 720P (1280×720), and the audio and video parameters are set as follows:

- Video

H. 264 coder is used, the bitrate is 1500Kbps, the width is 1280, the height is adaptive (to avoid the picture is scaled out of proportion due to a fixed height value), and the frame rate value is 25.

- Audio

AAC encoder is used, the bit rate is 128Kbps, the channels are 2 and the samplerate is 44100.

- Transcoding template

The preset static template “MP4-fluent” is used: S00000001-200010, the video bitrate set in the template is 400Kbps, the audio bit rate is 64Kbps, and the width is 640 (The height is adaptive).

- Output result

The API parameters overwrite the template parameters, therefore, in the output video, the bitrate is 1500Kbps, the width is 1280, the frame rate is 25, and in the output audio, the bitrate is 128Kbps, the channels are 2, and the samplerate is 44100.

Example code

[Simple transcoding-Java](#)

[Simple transcoding-Python](#)

[Simple transcoding-PHP](#)

1.2 API transcoding

Background

When workflows can not meet user requirements, users need to judge their business logic and use the API to submit transcoding tasks. For example, not all videos require transcoding; and different videos need different transcoding configurations.

Advantage

- Customizes business logic and flexibly submit transcoding tasks.
- Supports powerful functions such as transcoding, encapsulation, watermarking, supports HLS-AES128 standard encryption, editing and other functions.
- Supports sending execution information to the specified message queue or message notification upon completion of the transcoding task.
- Supports URL playback.

Restrictions

- A transcoding task generates an output file that allows tasks to be submitted in batch.
- API transcoding supports HLS-AES128 standard encryption. Currently, Alibaba Cloud private encryption is not supported.
- API transcoding supports URL playback, but does not support media ID playback. Users need to associate multiple output of multiple definition in multiple formats to achieve the logic of automatic switching between different definition and supporting multiple formats.

Preparation

- Custom transcoding template (as needed), log on to the [MPS console](#) for configuration.
- Custom watermark template (as needed), log on to the [MPS console](#) for configuration.

Procedure

1. [Upload input files to OSS](#) (Multiple upload options: OSS console, OSS related uploading tools and upload SDK).
2. [Set MPS Queue notification](#).
3. [Submit transcoding task](#).
4. After getting the message, call the “QueryTranscodingJob” interface to query the task execution result and get the output file URL.
- 5.
6. Video playback via URL.

7.

Set up an application to add watermarks to the video

[Java source code download.](#)

1.3 Workflow transcoding

Background

One input file corresponds to multiple output files (different resolutions, different formats, etc.).

A commonly used video processing flow can be created quickly through the console graphic interface.

Advantages

- Simple and easy to use, once video upload has completed, transcoding tasks are automatically triggered.
- Supports multiple functions such as screenshots, transcoding, encapsulation, watermarking, editing and other functions.
- Supports sending workflow execution message to the specified MPS queue or message notification upon beginning and ending the workflow.
- Media Files to offer you audio and video management capabilities. Media ID associates outputs of multiple definition in multiple formats. When using the MediaID for playback, automatic switch among multiple definitions can be achieved , and multiple formats are supported.
- Supports both URL and MediaID playback.

For more information, see [Developing process of workflows](#).

Restrictions

- A workflow can only be configured with only one input path, the workflow processes handle all videos in this path.
- Videos uploaded to the input directory will trigger workflow transcoding. Currently, only simple conditional transcoding scenarios are supported, and some complex business logic cannot be supported.
- The workflow supports Alibaba Cloud private encryption, but HLS-AES128 standard encryption is not yet supported.

Preparation

- [Custom transcoding template \(As needed\)](#).

- [Custom watermark template \(As needed\)](#).

Procedure

1. [Add Input/Output Media Bucket](#).
2. [Create a workflow](#). In the workflow, you can customize parameters for screenshots, transcoding, conversion and encapsulation, watermark, encryption, editing and other functions.
3. Enable the CDN acceleration function (optional): If you need to enable Content Distribution Acceleration for your domain name, see [Domain name management](#).
4. [Upload videos](#): You can use the MPS console or OSS related upload tools to upload a video file. In addition, an upload SDK that covers all platforms is provided. For details, see Upload SDK usage instructions, Upload SDK downloading.
5. [Manage videos](#).
6. Play back videos.

Set up a video transcoding application

[Download JAVA source code](#).

2 Encryption

2.1 HLS encryption and play

Purpose

This document describes the complete procedure of creating HLS standard encryption workflow to play the encrypted video.

For more information about the architecture of HLS standard encryption, see [HLS standard encryption](#).

Procedure

1. Create HLS encryption workflow.

For more information about creating HLS encryption workflow and DEMO code, see [Create HLS standard encryption workflow](#).



Note:

When creating HLS standard workflow, enter `http://127.0.0.1:8888` in the value of the `HLS_KEY_URI` parameter for a test. During playing, the player request the key to this address, and we create a service to distribute key.

2. Upload and encrypt video.

Upload a video by using Media Files in the MPS console. When selecting workflow, select the newly created HLS standard encryption workflow. After uploading, the workflow automatically triggers encryption transcoding. When the video is in the published status, follow these steps.

3. Create local authentication service.

Create a local HTTP service, which serves as authentication service in playing HLS standard encryption video, to issue and verify `MtsHlsUriToken` token.

Java code dependency example:

<https://mvnrepository.com/artifact/com.aliyun/aliyun-java-sdk-core>

<https://mvnrepository.com/artifact/com.aliyun/aliyun-java-sdk-kms>

```
package com.aliyun.smallcode;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.kms.model.v20160120.DecryptRequest;
```

```
import com.aliyuncs.kms.model.v20160120. DecryptResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.sun.net.httpserver.Headers;
import com.sun.net.httpserver.HttpExchange;
import com.sun.net.httpserver.HttpHandler;
import com.sun.net.httpserver.HttpServer;
import com.sun.net.httpserver.spi.HttpServiceProvider;
import org.apache.commons.codec.binary.Base64;
import java.io.IOException;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.InetSocketAddress;
import java.net.URI;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
public class AuthorizationServer {
private static DefaultAcsClient client;
static {
String region = "";
String accessKeyId = "<your-access-key-id>"
String accessKeySecret = "<your-access-key-secret>";
client = new DefaultAcsClient(DefaultProfile.getProfile(region,
accessKeyId, accessKeySecret));
}
public class AuthorizationHandler implements HttpHandler {
public void handle(HttpExchange httpExchange) throws IOException {
String requestMethod = httpExchange.getRequestMethod();
if(requestMethod.equalsIgnoreCase("GET")){
//Get ciphertext and key from URL
String ciphertext = getCiphertext(httpExchange);
if (null == ciphertext)
return;
//decrypt ciphertext from KMS, and Base64 decode
byte[] key = decrypt(ciphertext);
//Set header
setHeader(httpExchange, key);
//Response key
OutputStream responseBody = httpExchange.getResponseBody();
responseBody.write(key);
responseBody.close();
}
}
private void setHeader(HttpExchange httpExchange, byte[] key) throws
IOException {
Headers responseHeaders = httpExchange.getResponseHeaders();
responseHeaders.set("Access-Control-Allow-Origin", "*");
httpExchange.sendResponseHeaders(HttpURLConnection.HTTP_OK, key.
length);
}
private byte[] decrypt(String ciphertext) {
DecryptRequest request = new DecryptRequest();
request.setCiphertextBlob(ciphertext);
request.setProtocol(ProtocolType.HTTPS);
try {
DecryptResponse response = client.getAcsResponse(request);
String plaintext = response.getPlaintext();
//Note: require base64 decode
return Base64.decodeBase64(plaintext);
} catch (ClientException e) {
e.printStackTrace();
return null;
}
}
```

```

    }
    private String getCiphertext(HttpExchange httpExchange) {
        URI uri = httpExchange.getRequestURI();
        String queryString = uri.getQuery();
        String pattern = "Ciphertext=(\\w*)";
        Pattern r = Pattern.compile(pattern);
        Matcher m = r.matcher(queryString);
        if (m.find())
            return m.group(1);
        else {
            System.out.println("Not Found Ciphertext");
            return null;
        }
    }
    private void startService() throws IOException {
        HttpServerProvider provider = HttpServerProvider.provider();
        //listening port 8888 can accept 10 request simultaneously
        HttpServer httpserver = provider.createHttpServer(new InetSocketAddress(8888), 10);
        httpserver.createContext("/", new AuthorizationHandler());
        httpserver.start();
        System.out.println("server started");
    }
    public static void main(String[] args) throws IOException {
        AuthorizationServer server = new AuthorizationServer();
        server.startService();
    }
}

```

Python sample code:

pip install aliyun-python-sdk-core

pip install aliyun-python-sdk-kms

pip install aliyun-python-sdk-mts

```

# -*- coding: UTF-8 -*-
from BaseHTTPServer import BaseHTTPRequestHandler
from aliyunsdkcore.client import AcsClient
from aliyunsdkkms.request.v20160120 import DecryptRequest
import cgi
import json
import base64
import urlparse
client = AcsClient("", "", "");
class AuthorizationHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.check()
        self.set_header()
        ciphertext = self.get_ciphertext()
        plaintext = self.decrypt_ciphertext(ciphertext)
        print plaintext
        key = base64.b64decode(plaintext)
        print key
        self.wfile.write(key)
    def do_POST(self):
        pass

```

```
def check(self):
    #check MtsHlsUriToken, etc.
    pass
def set_header(self):
    self.send_response(200)
    #cors
    self.send_header('Access-Control-Allow-Origin', '*')
    self.end_headers()
def get_cihpertext(self):
    path = urlparse.urlparse(self.path)
    query = urlparse.parse_qs(path.query)
    return query.get('Ciphertext')[0]
def decrypt_cihpertext(self, cipertext):
    request = DecryptRequest.DecryptRequest()
    request.set_CiphertextBlob(cipertext)
    response = client.do_action_with_exception(request)
    jsonResp = json.loads(response)
    return jsonResp["Plaintext"]
if __name__ == '__main__':
    # Start a simple server, and loop forever
    from BaseHTTPServer import HTTPServer
    print "Starting server, use  to stop"
    server = HTTPServer(('127.0.0.1', 8888), AuthorizationHandler)
    server.serve_forever()
```

4. Obtain playback addresses.

You can obtain playback address by multiple ways. For more information, see [Questions about MPS file output](#).

5. Play video.

By using an online player, test the playback of HLS encryption video. For more information, see [Alibaba Cloud player user diagnositc tool](#).

Enter the playback address obtained from step 4 to the dialogue box as shown in the following figure, and click **Play**.



Note:

By using browser DEBUG, the player automatically request authentication server, obtain decryption key and do the playback operation after decryption.

3 Upload videos

Background

The following describes how to quickly build an audio and video file upload service based on the OSS service and MPS's SDK upload.

Advantages

Uploading audio and video files using MPS's SDK offers the following advantages:

- Adds file list management.
- Adds STS Token timeout update function.
- Auto-retry function when network jitter occurs in the process of uploading.
- File resume breakpoint function.
- Workflow that automatically triggers the MPS service.
- Configures media titles, tags, descriptions, categories, cover URLs, and more.



Note:

- Restrictions on resuming HTTP: does not allow cross-lifecycle. JS side page can not be refreshed, closed, and Android/iOS can not close the APP and mobile phone.
- The same local file can only be uploaded once.

Server creation

Consider mobile AK security issues, choose STS to upload files. To learn how using STS increases the security of the upload, see [RAM and STS User Guide](#).

STS Activation Procedure

1. Activate the OSS service, create a bucket, and log on to the [OSS console](#).
2. Find the basic configuration area on the OSS overview page and click the **security token**.
3. Go to the **Security Token Shortcut Configuration** page.
4. Authorize automatically and save parameters in the text boxes. Click **Save AK Info** to close the dialog and complete STS activation.

Build an application server

Configuration of app server sample code

This document provides three development example programs available for download in three languages.

- Java: [Download](#)
- PHP: [Download](#)
- Ruby: [Download](#)

The download for each language pack contains a configuration file: config.json:

```
{
  "AccessKeyID" : "",
  "AccessKeySecret" : "",
  "RoleArn" : "",
  "TokenExpireTime" : "900",
  "PolicyFile": "policy/all_policy.txt"
}
```



Note:

- **AccessKeyID:** Set the parameter value marked 1 in the above diagram.
- **AccessKeySecret:** Set the parameter value marked 2 in the above diagram.
- **RoleArn:** Set the parameter value marked 3 in the above diagram.
- **TokenExpireTime:** Indicates the expiration time of the token obtained by the Android/iOS app. Note: The minimum value is 900s. The default value can be retained.
- **PolicyFile:** Fill in the list of rights to the Token file, the default value can not be modified.

This document has provided three token files defining the most common permissions in the policy directory. They are:

- **all_policy.txt:** Specifies that the token has the authority to create or delete a bucket, to upload or download a file, and to delete a file under this account.
- **bucket_read_policy.txt:** Specifies that the token has read access to the specified bucket under this account.
- **bucket_read_write_policy.txt:** Specifies a token that grants read and write permissions for the specified bucket for this account.

If you want to create a token to grant read and write permissions for the specified bucket, replace \$BUCKET_NAME in the bucket_read_policy.txt and bucket_read_write_policy.txt files with the name of the desired bucket.

- Return format resolution:

```
{
  "status":200,
  "AccessKeyId":"STS. 3pYjsdgdgagdasdg",
  "AccessKeySecret":"rpnwO9kvEgetGdrddgsR2YrTtI",
  "Security":"CAES+wMIARKAAZhjH0EUOIhJMQBMjRywXq7MQ/cjLYg80Aho
1ek0Jm63Xmhr9Oc5s3qaPer8p1YaX1NTDiCFZWfKv1Hf1pQhuxfKBc+mRR9KAbHue
fQh+rdjZqjTF7p2mlwJXP8S6k+G2MpHrUe6TYBkJ43GhhTVFMuM3BZajY3VjZWOXBI
ODRIRlFKZjiEjMzMzE0MjY0NzM5MTE4NjkxMSoLY2xpZGSSDgSDGAGESGTE
TqOio6c2RrLWRlbW8vKgoUYWNzOm9zczoqOio6c2RrLWRlbW9KEDExNDg5Mz
AxMDcyNDY4MThSBTI2ODQyWg9Bc3N1bWVlUm9sZVVzZXJgAGoSMzMzMtQyNj
Q3MzkxMTg2OTExcglzZGstZGVtbzI=",
  "Expiration":"2015-12-12T07:49:09Z",
}
```



Note:

Note (the four variables shown below comprise a token) :

- **status** indicates the result that the app retrieves the token. The app returns a 200 status code for successful retrieval of the token.
 - **AccessKeyId** indicates the AccessKeyId the Android/iOS app obtains when initializing the OSS client.
 - **AccessKeySecret** indicates the AccessKeySecret the Android/iOS app obtains when initializing the OSS client.
 - **SecurityToken** indicates the token the Android/iOS app initializes.
 - **Expiration** indicates the time when the token expires. The Android SDK will automatically determine the validity of the token and retrieve a new one as needed.
- Examples of how to run code:
 - For JAVA (based on Java 1.7), after downloading and unzipping a pack, run this command: java -jar oss-token-server.jar (port). If you run java -jar oss-token-server.jar without specifying a port, the program listens to Port 7080. To change the listening port to 9000, run java -jar app-token-server.jar 9000. Specify the port number as needed.
 - For PHP, after you download and decompress the package, modify the config.json file and run php sts.php directly to generate a token. Then set up the app server at the specified address.

Use the MPS client SDK

- Client Sample Code

This document provides three development example programs available for download in three languages.

- H5: [Download](#)
- Android: [Download](#)
- iOS: [Download](#)
- SDK core code

JS side

Before using the JS SDK, first open [CORS Access](#) to the OSS Bucket where you want to upload the video. Download JS Demo, open in a browser, the parameters on the page configuration are:

- Configure the “HTTP Address” as the application server address configured above, for example, <http://127.0.0.1:7080/>.
- Configure user Bucket.
- Configure Bucket endpoint.
- Click to select the file, select the file to be uploaded.
- Click the Start Upload button.

```
// Initialize the client
var uploader = new VODUpload({
  // Start upload
  'onUploadstarted': function (uploadInfo) {},
  //File uploaded successfully
  'onUploadSucceed': function (uploadInfo) {console.log("Uploaded successfully");},
  //File upload failed
  'onUploadFailed': function (uploadInfo, code, message) {console.log("File upload failed");},
  // File upload progress, in bytes
  'onUploadProgress': function (uploadInfo, totalSize, uploadedSize) {
    console.log("File upload progress,");
  },
  //Security token timed out
  'onUploadTokenExpired': function (uploadInfo) {console.log("Token timeout");}
});
// Get STS Information
result = httpGet(httpServer);
stsToken = JSON.parse(result);
uploader.init(stsToken.AccessKeyId, stsToken.AccessKeySecret,
stsToken.SecurityToken, stsToken.Expiration);
// Add File
```

```
uploader.addFile(event.target.files[i], endpoint, bucket, object,
userData);
// Start uploading
uploader.startUpload();
```

Android end

Make sure Android has added the following permissions:

```
<uses-permission android:name="android.permission.INTERNET"></uses-
permission>
<uses-permission android:name="android.permission.ACCESS_NET
WORK_STATE"></uses-permission>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE
"></uses-permission>
<uses-permission android:name="android.permission.WRITE_EXTE
RNAL_STORAGE"></uses-permission>
```

Download Android Demo, make the following changes:

- Modify MainActivity inside serverUrl for the application server configuration address, such as <http://192.168.0.2:7080/>.
- Configure user Bucket.
- Configure the endpoint corresponding to the user Bucket.
- Run Demo, click Add File.
- Click Upload to check whether the file has been uploaded successfully under the uploadtest / directory of the OSS Bucket.

Main code:

```
VODUploadClient uploader = new VODUploadClientImpl(getApplica
tionContext());
VODUploadCallback callback = new VODUploadCallback() {
@Override
public void onUploadSucceed(UploadFileInfo info) {}
@Override
public void onUploadFailed(UploadFileInfo info, String code, String
message) {}
@Override
public void onUploadProgress(UploadFileInfo info, long uploadedSize
, long totalSize) {}
@Override
public void onUploadTokenExpired(UploadFileInfo info) {
// Get and update STS token.
uploader.resumeWithToken("", "", "", "");
}
@Override
public void onUploadRetry(UploadFileInfo info, String code, String
message) {}
@Override
public void onUploadRetryResume(UploadFileInfo info) {}
@Override
public boolean onUploadStarted(UploadFileInfo uploadFileInfo) {}
```

```
};
// Get STS token and initialize
uploader.init("", "", "", "", callback);
// Add File
uploader.addFile("", "", "", "");
// Start upload
uploader.start();
```

IOS end

Download iOS Demo, make the following changes:

- Modify the serverUrl in VODUploadDemo.m to configure the address for the application server, such as <http://192.168.0.2:7080/>.
- Configure user Bucket.
- Configure the endpoint corresponding to the user Bucket.
- Run Demo, click Add File.
- Click Upload to check whether the file has been uploaded successfully under the uploadtest / directory of the OSS Bucket.

Main code:

```
// Callback Initialization
OnUploadStartedListener testUploadStartedCallbackFunc = ^(UploadFile
Info* fileInfo) {;;};
OnUploadSucceedListener testSuccessCallbackFunc = ^(NSString*
filePath){;;};
OnUploadFailedListener testFailedCallbackFunc = ^(NSString* filePath
, NSString* code, NSString* message){;;};
OnUploadProgressListener testProgressCallbackFunc = ^(NSString*
filePath, long uploadedSize, long totalSize) {;;};
OnUploadTokenExpiredListener testTokenExpiredCallbackFunc = ^{
// Get and update STS token
[uploader resumeWithToken:
accessKeySecret:
secretToken:
expireTime:]
};
OnUploadRertyListener testUploadRertyListener = ^{;;};
OnUploadRertyResumeListener testUploadRertyResumeListener = ^{;;};
VODUploadListener *listener;
listener = [[VODUploadListener alloc] init];
listener.started = testUploadStartedCallbackFunc;
listener.success = testSuccessCallbackFunc;
listener.failure = testFailedCallbackFunc;
listener.progress = testProgressCallbackFunc;
listener.expire = testTokenExpiredCallbackFunc;
listener.retry = testUploadRertyListener;
listener.retryResume = testUploadRertyResumeListener;
// Get Token
// Upload client Initialization
VODUploadClient *uploader;
[uploader init:
accessKeySecret:
```

```
secretToken:
expireTime:
listener:listener];
// Add File
[uploader addFile:
endpoint:
bucket:
object:];
// Start upload
[uploader start];
```

4 Watermarks

Watermarks refer to the process of adding related information (such as a corporate logo, TV station logo, and user nickname) to a video. Watermarks can highlight brands, protect copyrights, and increase product recognition. MPS supports three watermark types: **image watermarks**, **animated watermarks** and **text watermarks**. You can select a type based on your needs.

Type

- **Image watermarks:** You can use a PNG image as the watermark. The image is displayed at a fixed position on the video. You can specify the display time (from the beginning to the end or in a certain duration).
- **Animated watermark:** You can use an animated graphic in APNG format or a video in MOV format as the watermark. The animation is displayed repeatedly at a fixed position on the video.
- **Text watermark:** You can use texts as the watermark. You can specify the font, size, and color of the texts and add different texts to different videos.

Parameter description

When submitting transcoding tasks (see [Simple transcoding](#)), you can specify a watermark template and material to add watermark information for the output video.

You can specify several [WaterMark](#) objects for each transcoding task. A WaterMark object contains the following parameters:

- WaterMarkTemplateId (watermark template ID)

A watermark template contains common parameters such as Type, ReferPos, Width, Height, Dx, and Dy.

You can create a template on the [Media Processing console](#). For more information, see [Watermark transcoding template](#).



Note:

The parameters in WaterMark objects have a higher priority than the corresponding parameters in the template. The parameters in WaterMark objects overwrite those parameters in the template.

- Type (watermark type)

When adding image or animated watermarks, you can set Type to Image and specify InputFile, which is the OSS file path of the watermark materials.

When adding text watermarks, you can set Type to Image and specify the [TextWaterMark](#) parameter, including text font, size, color, and transparency.

- ReferPos (watermark position)

The reference position where a watermark is displayed. Dx and Dy are calculated based on ReferPos. See [Watermark template configuration](#).

Watermark coordinate description:

- Width, Height, Dx, and Dy

Specify the width, height, horizontal offset, and vertical offset of a watermark. The values can be calculated in the following two ways:

- Absolute value:

Units: pixels. Value range: [8,4096].

- Relative scale:

The width and height relative to the output video resolution. Value range: (0, 1). Up to four digits are allowed after the decimal point, for example, 0.9999.

- Default value:

- The default value is 0 when Dx and Dy are not set.

- When neither Width nor Height is set, the watermark width is 0.12 times the resolution width of the output video. The watermark height scales proportionally according to the aspect ratio of the original watermark image.

- When either Width or Height is set, the unspecified parameter scales proportionally based on the aspect ratio of the original watermark image.

- When both Width and Height are set, the watermark image is set based on the specified values.

- InputFile (input file)

Specifies the OSS file location of an image or animated watermark. Images in PNG format and animations in MOV and APNG formats can be used as watermarks.

**Note:**

The file extension of an animated watermark must be mov or apng in lowercase. File extensions of images are not limited.

- [TextWaterMark](#) (text watermark)

Specifies the detailed parameters of text watermarks.

**Note:**

The reference position and relative scale cannot be set for text watermarks. You can specify the upper-left corner as the reference position and specify Dx and Dy offsets according to the absolute values.

Scenarios

Short videos

In this scenario, you can add an image watermark (product logo) and a text watermark (user ID) to downloaded or shared short videos to protect copyright.

Example:

Audio and video websites

For audio and video websites, you can add brand logos to videos to demonstrate copyright ownership. Additionally, you can also add stickers to videos of entertainment programs to make the programs more interesting or increase advertisement exposure.

Example:

Code examples

When transcoding to a 720P (1280 x 720) in MP4 format, you can add three watermarks at the same time and specify parameters for all three.

- Image watermark

You can specify the upper-right corner as the reference position and set the relative scale of the image watermark width to 0.05 times the resolution width of the output video. The watermark height scales proportionally based on the aspect ratio of the original watermark image.

- Text watermark

You can specify the upper-left corner as the reference position and set the text to `Test text watermark`. You can specify the text font information such as typeface to SimSun, font size

to 16, color to red, and transparency to 50. The text will be displayed in the video based on the specified information.

- Animated watermark

You can specify the lower-left corner as the reference position and add an MOV video as the watermark. You can set the height of the video to 240 pixels. The width of the video scales proportionally according to the aspect ratio of the original watermark video.

Code examples:

- [Watermarks-Java SDK](#)
- [Watermarks-Python SDK](#)
- [Watermarks-PHP SDK](#)

5 Screenshots

Screenshots refer to the process where an image captured at a specified time in a video is saved as an image file.

Type

- Key frame

A key frame has good image quality and can be captured quickly because key frames of videos are independently encoded. Key frames appear in a video at intervals but cannot be captured at a specified time. The system can search for corresponding key frames close to the specified time.

- Normal frame

Compared with a key frame, a normal frame has poorer image quality and takes longer time to capture. However, it can be precisely captured at a specified time.

Parameter description

You need to take note of the following parameters when you input a file:

Input

Specifies the OSS input file of the video that you want to take a screenshot of.



Note:

The location of OSS must correspond to the region of MPS. For example, oss-cn-hangzhou location of OSS corresponds to cn-hangzhou region of MPS.

You need to take note of the following parameters in [SnapshotConfig](#):

- *OutputFile*

Specifies the OSS output file of screenshots. You can either specify a fixed file name of an OSS Object or customize one. For naming rules, see [Screenshot OutputFile](#).

- Time

Specifies the time for a single screenshot or the start time for multiple screenshots. Integer type in milliseconds.

- Interval and Num

Specify the time interval between any two frames (unit: second) and the number of frames to be captured. It involves the following situations:

- When Num is not set, screenshots are taken till the end of the video based on the specified interval.
- When the value of Num is greater than 1, screenshots are taken based on the specified interval till the number of captured screenshots reaches the specified number.
- When the value of Num is 1, a single screenshot is taken asynchronously.
- Width and Height

Specify the width and height (pixels) of one or more output screenshot images.

The width and height are based on the input video.

- When neither Width nor Height is set, the size of the output image is the same as that of the video.
- When either Width or Height is set, the unspecified side is scaled based on the aspect ratio of the video. This avoids image deformation.



Note:

We recommend that you do not randomly set both Width and Height to avoid image distortion.

- When an MP4 video in the portrait mode has a rotation identifier, the screenshot orientation is landscape.
- When an MP4 video in the portrait mode does not have a rotation identifier, the screenshot orientation remains portrait.

- FrameType

Specifies the screenshot type: Key Frame or Normal Frame. The default value is Key Frame.

- TileOutputFile and TileOut

Specify the OSS output file of image sprites and [TileOut](#).

- SubOut and Format

- If you want to use WebVTT thumbnails, set Format to vtt.
- If you want to output the WebVTT thumbnails as image sprites, set Format and SubOut at the same time.

Scenarios

- Single screenshot

Specify a time to take the screenshot of a video at that time.

- Multiple screenshots

Specify a time interval to take multiple screenshots of a video at even intervals. Each screenshot is an image file. This operation is also called batch screenshots or ordinal screenshots.

- Image sprites

You can merge the images of multiple screenshots into one big image sprite. Multiple screenshots can be requested at one time, which reduces the number of image requests and improves client performance.

- WebVTT thumbnails

Standard subtitle format in HTML5 as well as the thumbnail preview format for several H5 players. See [JWPlayer Documents](#).

WebVTT is a file format. A thumbnail can be either multiple images or one big image sprite.

Execution method

See `Task execution and completion` in [Task and MPS Queue](#).

- Synchronization

When the API is called, IDs and results of screenshot jobs are synchronously returned.

Only a single screenshot can be synchronized.

- Asynchronization

When the API is called, only IDs of screenshot jobs are returned. To check the screenshot results, you can use screenshot job IDs or use the Notification service.

Single screenshot, multiple screenshots, image sprites, and WebVTT thumbnails can all be taken asynchronously.

Code examples

You have a 720P (1280x720) video lasting 10 seconds. You can specify the height of screenshots to 360 pixels, the start time to 2 seconds, the interval to 1 second, and the number to

3. Three images are captured at the second, third, and fourth seconds, respectively. The image files are named in succession, such as 00001, 00002, and 00003.

[*Screenshots-Java SDK*](#)

[*Screenshots-Python SDK*](#)

[*Screenshots-PHP SDK*](#)

6 Splicing and cutting

6.1 Splicing and simple cutting

Splicing refers to the process of splicing multiple videos of different formats, encoding, and resolutions and outputting a new video with the same format, encoding methods, and resolution. This function is often used to add fixed titles and credits and splice live broadcast videos.

Cutting refers to the process where you extract a certain clip of a video and output into a new video. This function is often used to extract the highlights or essential contents of videos.

Parameter description

When splicing a video, you need to take note of the following parameters:

Input

Specifies an OSS input file for the title.



Note:

The location of OSS must correspond to the region of MPS. For example, oss-cn-hangzhou location of OSS corresponds to cn-hangzhou region of MPS.

Set the following parameters in *Output*:

- *Video*

Specifies the width, height, and bit rate of the final output video. If the width and height of multiple clips (including titles and credits) are inconsistent with those of the final video, the margins will be automatically filled with black. We recommend that you prepare titles and credits of different widths and heights based on the actual resolutions of different jobs to create a better video.

- *MergeList*

The list order is the splicing order. The last item of the list is the credits. You can splice up to five videos (including the title and credits) into one. You can use the `MergeConfigUrl` parameter to splice more videos together.



Note:

`MergeList` and `MergeConfigUrl` are mutually exclusive. You can specify only one of them.

Each spliced video has three parameters:

— MergeURL

Specifies the OSS URL of a splicing video.

**Note:**

The OSS region of the video that you splice must be consistent with that of the title. Videos from different regions cannot be spliced.

— Start

When you splice videos, you can specify a start point to extract a desired clip for the final video. The default value is 0.

— Duration

When you splice videos, you can specify a duration (with the start point as the value of Start) to extract a desired clip for the final video. The default duration is from Start to the end of the video.

- MergeConfigUrl

Specifies the OSS URL of the configuration file of the videos that you are splicing. The content of the configuration file is a JSON object, with the same value of [MergeList](#).

The list order is the splicing order. The last item of the list is the credits. You can splice up to 100 clips (including the title and credits) into one.

Code examples

You can splice a 720P(1280x720) video with the 480P(640x480) title and credit in MP4 format. The resolution of the output video is 1280x720. When you play the output video, black borders are displayed on the left and right sides of the title and credit. However, the video is normally displayed.

Code example details:

- [Splicing and simple cutting-Java SDK](#)[Panels and simple clips-Java SDK](#)
- [Splicing and simple cutting-Python SDK](#)
- [Splicing and simple cutting-PHP SDK](#)

6.2 Opening and ending scenes

Video opening and ending scenes refer to a special effect of splicing: embedding the scenes in a video as picture-in-picture.

Parameter description

When you embed the opening and ending scenes, you need to take note of the following parameters:

Input

Specifies an OSS input file for the video.



Note:

The location of OSS must correspond to the region of MPS. For example, oss-cn-hangzhou location of OSS corresponds to cn-hangzhou region of MPS.

You need to note the following parameters in *Output*:

- *Video*

Specifies the width, height, and bit rate of the final output video. When the width and height of the video are inconsistent with those of the final output, the video will be stretched automatically. We recommend that you specify either the width or height. The unspecified side will be scaled based on the original aspect ratio of the video.

- *Opening*

The order of the opening scene list is the splicing order. You can add up to two opening scenes .

Each opening scene contains four parameters:

- **OpenUrl**

Specifies the OSS URL of an opening scene.



Note:

The OSS region of an opening scene must be consistent with that of the video you plan to splice it with. Videos from different regions cannot be spliced.

- **Start**

Specifies the time when the opening scene starts to play after the start of the video. The default value is 0 seconds.

- **Width**

Specifies the width of an opening scene. Two special values of the parameter:

- `-1` indicates that the width is equal to that of the opening scene source;
- `full` indicates that the video screen is filled up.
- You can specify other values. Value range: (0, 4096].

**Note:**

The scene is aligned to the center of the video. We recommend that you do not set the width of an opening scene to be greater than that of the video to avoid any problems.

— Height

Specifies the height of an opening scene. Two special values of the parameter:

- `-1` indicates that the height is equal to that of the opening scene source;
- `full` indicates that the video screen is filled up.
- You can specify other values. Value range: (0, 4096].

**Note:**

The scene is aligned to the center of the video. We recommend that you do not set the height of an opening scene to be greater than that of the video to avoid any problems.

- *[TailSlate](#)*

The order of the ending scene order list is the splicing order. You can add up to two ending scenes.

Each ending scene contains the following parameters:

— TailUrl

Specifies the OSS URL of an ending scene.

**Note:**

The OSS region of an ending scene must be consistent with that of the video you plan to splice it with. Videos from different regions cannot be spliced.

— Width

Specifies the width of an ending scene. Two special values of the parameter:

- `-1` indicates that the width is equal to that of the ending scene source;
- `full` indicates that the video screen is filled up.
- You can specify other values. Value range: (0, 4096].

**Note:**

The scene is aligned to the center of the video. We recommend that you do not set the width of an ending scene to be greater than that of the video to avoid any problems.

— Height

Specifies the height of an ending scene. Two special values of the parameter:

- `-1` indicates that the height equals to that of the ending scene source;
- `full` indicates that the video screen is filled up.
- You can specify other values. Value range: (0, 4096].

**Note:**

The scene is aligned to the center of the video. We recommend that you do not set the height of an ending scene to be greater than that of the video to avoid any problems.

— BlendDuration

The duration of transition from the video to the ending scene. The effect of transition is fade-in and fade-out: The last frame of the video and the ending scene start playing at the same time. The last frame of the video gradually fades out while the ending scene fades in. The default value is 0 seconds.

— IsMergeAudio

Specifies whether to splice the audio in the ending scene or not.

— BgColor

Specifies the background color that fills up the margin when the width or height is smaller than that of the video.

Code example

You can splice a 720P (1280x720) video with the 480P (640x480) opening and ending scenes in MP4 format. You can set the start time of the opening scene to 2 seconds, the transition time of the ending scene to 3 seconds, and the background color to `Black`. The opening scene starts to play two seconds into the video. The opening scene is centered in the video and plays simultaneously with the video. The ending scene fades in and out at the end of the video.

Code example:

- [Opening and ending scenes-Java SDK](#)

- [*Opening and ending scene-Python SDK*](#)
- [*Opening and ending scene-PHP SDK*](#)

7 Package

7.1 HLS package

Introduction

HLS package refers to the process in which multi-subtitle, multi-track and multi-bitstream are integrated into a Master Playlist file. The process includes creating HLS package workflow and calling AddMedia interface to specify video and the ID of the HLS package workflow for video processing.

1. When you use [AddMediaWorkflow](#) interface to add workflow, pay attention to the following objects:

- Topology

Topology refers to the business processing procedure, Directed Acyclic Graph (DAG).

- Activity

Activity refers to the processing nodes which constitute the topology. While creating HLS package workflow, pay attention to the following activities:

— [PackageConfig](#)

Specify HLS package configuration, and configure the output location for the Master Playlist file

.

- The front node allows: Start.
- The back node allows: SubtitleGroup, AudioGroup, and Transcode (only video).

— [SubtitleGroup](#)

Specify the subtitle group ID

.

- The front node allows: PackageConfig.
- The back node allows: Transcode (only subtitle).

— [AudioGroup](#)

Specify the audio group ID

.

- The front node allows: PackageConfig.
- The back node allows: Transcode (only audio).

— *Transcode*

Extract video streams/audio stream/subtitle stream.

.

- The front node allows: PackageConfig, SubtitleGroup, and AudioGroup.
- The back node allows: GenerateMasterPlayList.

— *GenerateMasterPlayList*

HLS package generation activity specifies multi-ratestream configuration, audio group and subtitle group

.

- The front node allows: Transcode.
- The back node allows: Report.

• Dependencies

Dependencies refer to the edges of the topology, indicating the dependency between activities.

2. When you use the *AddMedia* interface to add media, pay attention to the following aspects:

- Specify Media workflow ID.
- If subtitle extraction exists, you can configure in the way that the subtitle file address overwrites the WebVTTSubtitleURL parameter in the Transcode activity, and only subtitle files of WebVTT are supported.
- Set the Workflow triggering mode as NotInAuto.

Scenarios

The mxf format of the source file, also supports such formats as mp4, flv and m3u8(ts), extracts three audio tracks, two video streams and two groups of WebVTT subtitles from the source file, and then combine and package into a Master Playlist:

Configure HLS package output location and name of Master Playlist.

- Configure Bucket.
- Configure Location.

- Configure the name of Master Playlist.
- The activity is defined as follows:

```
{
  "Parameters" : {
    "Output" : "{ \"Bucket\": \"processedmediafile\", \"Location\": \"oss-cn-hangzhou\", \"MasterPlaylistName\": \"{MediaId}/{RunId}/hls/master.m3u8\" }"
  },
  "Type" : "PackageConfig"
}
```

- Output configures the storage location and name of Master Playlist. For more information, see [Parameters supported by the PackageConfig activity](#).
- Type specifies the activity type as PackageConfig.

Audio group

- Configure the audio group ID, wherein two audio streams belongs to the same audio group.
- The activity is defined as follows:

```
{
  "Parameters" : {
    "GroupId" : "audios"
  },
  "Type" : "AudioGroup"
}
```

- GroupId: Specify the audio group Id as audios.
- Type: Specify the type as AudioGroup activity.

Audio extraction

- Extract audio streams from mxf source file, and the video streams must be removed.
- Output audio parameters:
 - Codec: AAC
 - SampleRate : 48000 Hz
 - Format : Stereo
- The activity is defined as follows:

```
{
  "Name" : "audio-extract-1",
  "Parameters" : {
    "Outputs" : "[ { \"TemplateId\": \"S00000001-100020\", \"AudioStreamMap\": \"0:a:0\", \"Video\": { \"Remove\": \"true\" } } ]",

```

```
"ExtXMedia" : "{ \"URI\" : \"sd/audio-en.m3u8\", \"Name\" : \"audio-en\", \"Language\" : \"en-US\" }"
}
```

- [Preset static templates](#) ID: S00000001-100020 indicates that the audio output is m3u8(ts), and the audio bitrate configured in the preset templates is 80kbps.
- `AudioStreamMap`: Audio stream sequence number. For more information, see [Output](#).
- Remove the video streams from the output. For more information, see [Video](#).
- [ExtXMedia](#) defines Media Playlist, and URI specifies the name of Media Playlist.
- Type is configured as Transcode, which is transcode activity.

Video extraction

- Extract video streams from the mxf source file, and the audio streams must be removed.
- The activity is defined as follows:

```
{
  "Name" : "video-extract",
  "Parameters" : {
    "Outputs" : "[ { \"TemplateId\" : \"1fe5393bdb7b2b883f0a0fc91e81344a\", \"Audio\" : { \"Remove\" : \"true\" } } ]",
    "MultiBitrateVideoStream" : "{ \"URI\" : \"sd/video1.m3u8\" }"
  },
  "Type" : "Transcode"
}
```

- Custom transcoding template ID: 1fe5393bdb7b2b883f0a0fc91e81344a, you can log on to the [MPS console](#), and configure the video transcoding parameter in **Settings > Transcoding Templates**:
 - Codec : H. 264
 - Resolution : 384x216
 - Profile : Main
 - Bitrate : 240 Kbps
 - Fps : 25
 - PixelFormat : YUV420P Max GOP size : 1 segment length (4 seconds)
 - Output format: m-3u8
- Remove audio streams from the output. For more information [Audio](#).
- [MultiBitrateVideoStream](#) defines the multi-bitrate video streams in Master Playlist, and URI specifies the name of Media Playlist.
- Type is configured as Transcode, which is transcode activity.

Subtitle group

- Configure subtitle group ID, wherein two subtitle streams belong to the same subtitle group.
- The activity is defined as follows:

```
{
  "Parameters" : {
    "GroupId" : "subtitles"
  },
  "Type" : "SubtitleGroup"
}
```

- **GroupId**: Specify the audio group Id as subtitles.
- **Type**: Specify the type as SubtitleGroup activity.

Subtitle extraction

- Upload subtitles in the WebVtt format to OSS.
- The activity is defined as follows:

```
{
  "Name" : "subtitle-extract-1",
  "Parameters" : {
    "WebVTTSubtitleURL" : "http://mts-video.oss-cn-hangzhou.aliyun-inc.com/ShawshankRedemption.vtt",
    "ExtXMedia" : "{ \"URI\" : \"zh/subtitle1-cn.m3u8\", \"Name\" : \"subtitle-cn\", \"Language\" : \"cn\" }"
  },
  "Type" : "Transcode"
}
```

- [WebVTTSubtitleURL](#) specified the subtitle address. The subtitle address is overwritten dynamically while calling [AddMedia](#). For more information about this parameter, see [OverrideParams](#).
- [ExtXMedia](#) defines Media Playlist, and URI specifies the name of Media Playlist.
- Type is configured as Transcode, which is transcode activity.

Master Playlist output

- By means of audio, video and subtitle extraction, all the resources after extraction conversion are packaged into a Master Playlist.
- The activity is defined as follows:

```
{
  "Parameters" : {
```

```
"MasterPlayList" : "{\\"MultiBitrateVideoStreams\\": [{\\"RefActivityName\\": \\"video-extract\\",\\"ExtXStreamInfo\\": {\\"BandWidth\\": \\"1110000\\",\\"Audio\\": \\"audios\\",\\"Subtitles\\": \\"subtitles\\"}}]}",
"Type" : "GenerateMasterPlayList"
}
```

- **MasterPlayList** defines Master Playlist.
- **MultiBitrateVideoStreams** refers to multi-bitrate video streams group.
- RefActivityName specifies the activity name of video streams.
- **ExtXStreamInfo** defines the attributes of the multi-bitrate video streams, Audio specifies the audio group, and Subtitles specifies the subtitle group.
- Type is set as GenerateMasterPlayList, that is generating Master Playlist activity.

Topology:

Complete scenario example shown in topology:

```
{
  "Activities" : {
    "package-node" : {
      "Name" : "package-node",
      "Parameters" : {
        "Output" : "{\\"Bucket\\": \\"processedmediafile\\",\\"Location\\": \\"oss-cn-hangzhou\\",\\"MasterPlayListName\\": \\"{MediaId}/{RunId}/hls/master.m3u8\\"}"
      },
      "Type" : "PackageConfig"
    },
    "audioGroupNode" : {
      "Name" : "audioGroupNode",
      "Parameters" : {
        "GroupId" : "audios"
      },
      "Type" : "AudioGroup"
    },
    "subtitleGroupNode" : {
      "Name" : "subtitleGroupNode",
      "Parameters" : {
        "GroupId" : "subtitles"
      },
      "Type" : "SubtitleGroup"
    },
    "video-extract-1" : {
      "Name" : "video-extract-1",
      "Parameters" : {
        "Outputs" : "[{\\"TemplateId\\":\\"1fe5393bdb7b2b883f0a0fc91e81344a\\",\\"Audio\\":{\\"Remove\\":\\"true\\"}}]",
        "MultiBitrateVideoStream" : "{\\"URI\\": \\"sd/video1.m3u8\\"}"
      },
      "Type" : "Transcode"
    },
    "video-extract-2" : {
```

```

"Name" : "video-extract-1",
"Parameters" : {
  "Outputs" : "[{\\"TemplateId\\":\\"1fe5393bdb7b2b883f0a0fc91e81344b\\",\\"
Audio\\":{\\"Remove\\":\\"true\\"}}]",
  "MultiBitrateVideoStream" : "{\\"URI\\": \\"sd/video2.m3u8\\"}"
},
"Type" : "Transcode"
},
"audio-extract-1" : {
  "Name" : "audio-extract-1",
  "Parameters" : {
    "Outputs" : "[{\\"TemplateId\\":\\"S00000001-100020\\",\\"AudioStreamMap\\":
\\"0:a:0\\"}]",
    "ExtXMedia" : "{\\"URI\\": \\"sd/audio-en-1.m3u8\\",\\"Name\\": \\"audio-en
\\",\\"Language\\": \\"en-US\\"}"
  },
  "Type" : "Transcode"
},
"audio-extract-2" : {
  "Name" : "audio-extract-2",
  "Parameters" : {
    "Outputs" : "[{\\"TemplateId\\":\\"S00000001-100020\\",\\"AudioStreamMap\\":
\\"0:a:1\\"}]",
    "ExtXMedia" : "{\\"URI\\": \\"sd/audio-cn.m3u8\\",\\"Name\\": \\"audio-cn\\",
\\"Language\\": \\"cn\\"}"
  },
  "Type" : "Transcode"
},
"audio-extract-3" : {
  "Name" : "audio-extract-3",
  "Parameters" : {
    "Outputs" : "[{\\"TemplateId\\":\\"S00000001-100020\\",\\"AudioStreamMap\\":
\\"0:a:2\\"}]",
    "ExtXMedia" : "{\\"URI\\": \\"sd/audio-de.m3u8\\",\\"Name\\": \\"audio-de\\",
\\"Language\\": \\"de\\"}"
  },
  "Type" : "Transcode"
},
"subtitle-extract-1" : {
  "Name" : "subtitle-extract-1",
  "Parameters" : {
    "WebVTTSubtitleURL" : "http://mts-video-daily-bucket.oss-test.aliyun-
inc.com/1.vtt",
    "ExtXMedia" : "{\\"URI\\": \\"zh/subtitle1-cn.m3u8\\",\\"Name\\": \\"subtitle
-cn\\",\\"Language\\": \\"cn\\"}"
  },
  "Type" : "Transcode"
},
"subtitle-extract-2" : {
  "Name" : "subtitle-extract-2",
  "Parameters" : {
    "WebVTTSubtitleURL" : "http://mts-video.oss-cn-hangzhou.aliyun-inc.com
/ShawshankRedemption.vtt",
    "ExtXMedia" : "{\\"URI\\": \\"zh/subtitle1-en.m3u8\\",\\"Name\\": \\"subtitle
-en\\",\\"Language\\": \\"en-US\\"}"
  },
  "Type" : "Transcode"
},
"masterPlayListGenerate" : {
  "Name" : "masterPlayListGenerate",
  "Parameters" : {

```

```

"MasterPlayList" : "{ \"MultiBitrateVideoStreams\": [{ \"RefActivityName
\": \"video-extract-1\", \"ExtXStreamInfo\": { \"BandWidth\": \"1110000
\", \"Audio\": \"audios\", \"Subtitles\": \"subtitles\" } }, { \"RefActivit
yName\": \"video-extract-2\", \"ExtXStreamInfo\": { \"BandWidth\": \"
5000000\", \"Audio\": \"audios\", \"Subtitles\": \"subtitles\" } } ] }"
},
"Type" : "GenerateMasterPlayList"
},
"activityEnd" : {
"Name" : "activityEnd",
"Parameters" : {
"PublishType" : "Manual"
},
"Type" : "Report"
},
"activityStart" : {
"Name" : "activityStart",
"Parameters" : {
"PipelineId" : "900ededca77641ecbecd4f44cc3a2965",
"Role" : "AliyunMTSDefaultRole",
"InputFile" : "{ \"Bucket\": \"videouploaded\", \"Location\": \"oss-cn-
hangzhou\", \"ObjectPrefix\": \"uploaded/\" }"
},
"Type" : "Start"
},
},
"Dependencies" : {
"video-extract-1" : [ "masterPlayListGenerate" ],
"video-extract-2" : [ "masterPlayListGenerate" ],
"audio-extract-1" : [ "masterPlayListGenerate" ],
"audio-extract-2" : [ "masterPlayListGenerate" ],
"audio-extract-3" : [ "masterPlayListGenerate" ],
"subtitle-extract-1" : [ "masterPlayListGenerate" ],
"subtitle-extract-2" : [ "masterPlayListGenerate" ],
"package-node" : [ "video-extract-1", "video-extract-2", "subtitleGr
oupNode", "audioGroupNode" ],
"audioGroupNode" : [ "audio-extract-1", "audio-extract-2", "audio-
extract-3" ],
"subtitleGroupNode" : [ "subtitle-extract-1", "subtitle-extract-2" ],
"masterPlayListGenerate" : [ "activityEnd" ],
"activityEnd" : [ ],
"activityStart" : [ "package-node" ]
}
}
}

```

Code example

1. Create HLS package workflow

[Create workflow-Java](#)

[Create workflow-Python](#)

[Create workflow-PHP](#)

2. Add media

[Add media-Java](#)

[Add media-Python](#)

[Add media-PHP](#)

7.2 DASH packaging

Description

Dynamic Adaptive Streaming Over HTTP (DASH) packaging allows you to package one or more video streams at different bit rates, subtitles in different languages, and audio tracks into a Master Playlist file. The process includes creating a DASH packaging workflow and calling the AddMedia operation by specifying the media and the workflow ID.

1. To create a new workflow, call the [AddMediaWorkflow](#) operation.

- Topology

The business process that can be defined by users using directed acyclic graphs (DAGs).

- Activity

The type of processing nodes that constitute the topology. When you create a DASH packaging workflow, pay attention to the following activity types.

— [PackageConfig](#)

PackageConfig nodes are used to specify a location for the output Master Playlist file.

Topological sort:

- PackageConfig nodes can follow the Start node.
- PackageConfig nodes can precede SubtitleGroup, AudioGroup, or VideoGroup nodes

— [SubtitleGroup](#)

SubtitleGroup nodes are used to specify the ID and language of each subtitle group.

Topological sort:

- SubtitleGroup nodes can follow PackageConfig nodes.
- SubtitleGroup nodes can precede Transcode nodes (for subtitles only).

— [AudioGroup](#)

AudioGroup nodes are used to specify the ID and language of each audio group.

Topological sort:

- AudioGroup nodes can follow PackageConfig nodes.
- AudioGroup nodes can precede Transcode nodes (for audio only).

— VideoGroup

VideoGroup nodes are used to specify the ID of each video group.

Topological sort:

- VideoGroup nodes can follow PackageConfig nodes.
- VideoGroup nodes can precede Transcode nodes (for video only).

— Transcode

Transcode nodes are used to extract video, audio, or subtitle streams.

Topological sort:

- Transcode nodes can follow SubtitleGroup, AudioGroup, or VideoGroup nodes.
- Transcode nodes can precede GenerateMasterPlayList nodes.

— GenerateMasterPlayList

GenerateMasterPlayList nodes are used to generate a Master Playlist file.

Topological sort:

- GenerateMasterPlayList nodes can follow Transcode nodes.
- GenerateMasterPlayList nodes can precede Report nodes.

• Dependencies

The edges in the topology, which indicates the dependency between activities.

2. To add media to MPS, call the [AddMedia](#) operation.

- You must specify the ID of the media workflow.
- If you need to extract subtitles, you can configure the OverrideParams parameter to override the default URL of the subtitle files in the Transcode activity. For example, `{"subtitleTransNode":{"InputConfig":{"Format":"stl","InputFile":{"URL":"http://subtitleBucket.oss-cn-hangzhou.aliyuncs.com/package/subtitle/CENG.stl"}}}}`. In this example, subtitleTransNode is the node indicating subtitle extraction.
- Set TriggerMode to NotInAuto.

Scenario

Assume that you need to extract two video streams which contain three audio streams and two WebVTT subtitle streams from an mxf source file, and package them into a Master Playlist file.

The source file also supports mp4, flv and m3u8 (ts) formats.

Configure the output location and the name of the Master Playlist file.

- Specify Bucket in which the Master Playlist file will be stored.
- Specify Location for the Master Playlist file.
- Specify the name of the Master Playlist file.
- The activity is defined as follows:

```
{
  "Parameters" : {
    "Output" : "{ \"Bucket\": \"processedmediafile\", \"Location\": \"oss-cn-hangzhou\", \"MasterPlayListName\": \"{MediaId}/{RunId}/dash/master.mpd\" }"
  },
  "Type" : "PackageConfig"
}
```

- Output indicates the output location and the name of the Master Playlist file. For more information, see [Parameters supported for the PackageConfig activity](#).
- Set Type to PackageConfig.

Group audio streams

- The activity is defined as follows:

```
"audio-cn-group" : {
  "Name" : "audio-cn-group",
  "Parameters" : {
    "AdaptationSet" : "{ \"Lang\": \"chinese\", \"Group\": \"AudioGroupChinese\" }"
  },
  "Type" : "AudioGroup"
}
```

- Group: Specify the audio group name as AudioGroupChinese.
- Type: Specify the type as AudioGroup.

Extract audio streams

- To extract audio streams from the mxf source file, you must remove the video streams.
- Output audio parameters:

- The activity is defined as follows:

```
"audioCNTransNode" : {
  "Name" : "audioCNTransNode",
  "Parameters" : {
    "Outputs" : "[{\\"TemplateId\\":\\"S000000001-100020\\",\\"AudioStreamMap\\":\\"0:a:0\\",\\"Video\\":{\\"Remove\\":\\"true\\"}}]",
    "Representation" : "{\\"Id\\":\\"chinese128k\\",\\"URI\\":\\"audiocn/cn-abc.mpd\\"}"
  },
  "Type" : "Transcode"
}
```

- URI: The location for the output audio streams.
- AudioStreamMap: The sequence number of the audio stream. For more information, see [Output](#).
- Remove the video streams from the output. For more information, see [Video](#).
- Set Type to Transcode, which indicates transcoding activities.

Group Video streams

```
"video-group" : {
  "Name" : "video-group",
  "Parameters" : {
    "AdaptationSet" : "{\\"Group\\":\\"VideoGroup\\"}"
  },
  "Type" : "VideoGroup"
}
```

Extract video streams

- To extract video streams from the mxf source file, you must remove the audio streams.
- The activity is defined as follows:

```
"videoTransSD" : {
  "Name" : "videoTransSD",
  "Parameters" : {
    "Outputs" : "[{\\"TemplateId\\":\\"d861b90f6c0aed8f81095e5c5b857cba\\",\\"Audio\\":{\\"Remove\\":\\"true\\"}}]",
    "Representation" : "{\\"Id\\":\\"476pSD\\",\\"URI\\":\\"videoSD/xx.mpd\\"}"
  },
  "Type" : "Transcode"
}
```

- In this example, the ID of the user-defined transcoding template is d861b90f6c0aed8f81095e5c5b857cba. You can call the corresponding API operation to create a transcoding template, and the container format is mpd.
- Remove audio streams from the output. For more information, see [Audio](#).

- URI: The name and location of the output video streams.
- Set Type to Transcode, which indicates transcoding activities.

Group subtitle streams

- Specify the ID of the subtitle group.
- The activity is defined as follows:

```
"subtitle-cn-group" : {
  "Name" : "subtitle-cn-group",
  "Parameters" : {
    "AdaptationSet" : "{ \"Lang\" : \"Chinese\", \"Group\" : \"SubtitleEN
Group\" }"
  },
  "Type" : "SubtitleGroup"
}
```

- Group: Specify the subtitle group name as SubtitleENGroup.
- Lang: Specify a language for the subtitle group.
- Type: Specify the type as SubtitleGroup.

Extract subtitle streams

- Upload STL, TTML, and WebVTT subtitles to OSS.
- The activity is defined as follows:

```
"subtitleCNNode" : {
  "Name" : "subtitleCNNode",
  "Parameters" : {
    "InputConfig" : "{ \"Format\" : \"vtt\", \"InputFile\" : { \"URL\" : \"http
://bucketname.oss-cn-hangzhou.aliyuncs.com/test/Audio-SiHD.chs.vtt
\" } }",
    "Representation" : "{ \"Id\" : \"subtitle-chinese\", \"URI\" : \"subtitle
/cn-xx.vtt\" }"
  },
  "Type" : "Transcode"
}
```

- InputConfig specifies the URL of the subtitle. The URL can be overridden by configuring the OverrideParams parameter when you call the [AddMedia](#) operation.
- URI: The location for the output subtitle streams.
- Set Type to Transcode, which indicates transcoding activities.

Output a Master Playlist file

- Package all the output audio, video, and subtitle streams into a Master Playlist file.

- The activity is defined as follows:

```
{
  "Parameters" : {
  },
  "Type" : "GenerateMasterPlayList"
}
```

— Set Type to GenerateMasterPlayList, which indicates generating a Master Playlist file.

The topology is as follows:

The sample scenario shown in topology:

```
{
  "Activities": {
    "act-package": {
      "Name": "act-package",
      "Parameters": {
        "Output": "{\\"Bucket\\": \\"outputbucketname\\",\\"Location\\": \\"oss-cn-hangzhou\\",\\"MasterPlayListName\\": \\"dashpackage/{MediaId}/{RunId}/master.mpd\\"}",
        "Protocol": "dash"
      },
      "Type" : "PackageConfig"
    },
    "video-group": {
      "Name": "video-group",
      "Parameters": {
        "AdaptationSet": "{\\"Group\\":\\"VideoGroup\\"}"
      },
      "Type": "VideoGroup"
    },
    "audio-en-group": {
      "Name": "audio-en-group",
      "Parameters": {
        "AdaptationSet": "{\\"Lang\\":\\"english\\", \\"Group\\":\\"AudioGroupEnglish\\"}"
      },
      "Type" : "AudioGroup"
    },
    "audio-cn-group": {
      "Name": "audio-cn-group",
      "Parameters": {
        "AdaptationSet": "{\\"Lang\\":\\"chinese\\", \\"Group\\":\\"AudioGroupChinese\\"}"
      },
      "Type" : "AudioGroup"
    },
    "subtitle-en-group": {
      "Name": "subtitle-en-group",
      "Parameters": {
        "AdaptationSet": "{\\"Lang\\":\\"english\\", \\"Group\\":\\"SubtitleENGGroup\\"}"
      },
      "Type" : "SubtitleGroup"
    }
  }
}
```

```

},
"subtitle-cn-group": {
  "Name": "subtitle-cn-group",
  "Parameters": {
    "AdaptationSet": "{ \"Lang\": \"chinese\", \"Group\": \"SubtitleCNGroup\" }"
  },
  "Type": "SubtitleGroup"
},
"videoTransLD": {
  "Name": "videoTransLD",
  "Parameters": {
    "Outputs": "[ { \"TemplateId\": \"d053297fc44f9dd6becd4a98d1c42f50\", \"Audio\": { \"Remove\": \"true\" } } ]",
    "Representation": "{ \"Id\": \"270pLD\", \"URI\": \"videoLD/xx.mpd\" }"
  },
  "Type": "Transcode"
},
"videoTransSD": {
  "Name": "videoTransSD",
  "Parameters": {
    "Outputs": "[ { \"TemplateId\": \"d861b90f6c0aed8f81095e5c5b857cba\", \"Audio\": { \"Remove\": \"true\" } } ]",
    "Representation": "{ \"Id\": \"480pSD\", \"URI\": \"videoSD/xx.mpd\" }"
  },
  "Type": "Transcode"
},
"videoTransHD": {
  "Name": "videoTransHD",
  "Parameters": {
    "Outputs": "[ { \"TemplateId\": \"117b3ae88efbc97df372cfd9a0e1ff4c\", \"Audio\": { \"Remove\": \"true\" } } ]",
    "Representation": "{ \"Id\": \"720pHD\", \"URI\": \"videoHD/xx.mpd\" }"
  },
  "Type": "Transcode"
},
"audioCNTransNode": {
  "Name": "audioCNTransNode",
  "Parameters": {
    "Outputs": "[ { \"TemplateId\": \"d053297fc44f9dd6becd4a98d1c42f50\", \"AudioStreamMap\": \"0:a:0\", \"Video\": { \"Remove\": \"true\" } } ]",
    "Representation": "{ \"Id\": \"chinese128k\", \"URI\": \"audiocn/cn-abc.mpd\" }"
  },
  "Type": "Transcode"
},
"audioENTransNode": {
  "Name": "audioENTransNode",
  "Parameters": {
    "Outputs": "[ { \"TemplateId\": \"d053297fc44f9dd6becd4a98d1c42f50\", \"AudioStreamMap\": \"0:a:1\", \"Video\": { \"Remove\": \"true\" } } ]",
    "Representation": "{ \"Id\": \"english128k\", \"URI\": \"audioen/en-abc.mpd\" }"
  },
  "Type": "Transcode"
},
"subtitleENNode": {
  "Name": "subtitleENNode",
  "Parameters": {
    "InputConfig": "{ \"Format\": \"vtt\", \"InputFile\": { \"URL\": \"http://bucketname.oss-cn-hangzhou.aliyuncs.com/dashpackage/subtitle/Subtitle.EN.vtt\" } }",

```

```

"Representation": "{ \"Id\": \"subtitle-english\", \"URI\": \"subtitle/en-xx.vtt\" }",
},
"Type": "Transcode"
},
"subtitleCNNode": {
  "Name": "subtitleCNNode",
  "Parameters": {
    "InputConfig": "{ \"Format\": \"vtt\", \"InputFile\": { \"URL\": \"http://bucketname.oss-cn-hangzhou.aliyuncs.com/dashpackage/subtitle/Subtitle.CN.vtt\" } }",
    "Representation": "{ \"Id\": \"subtitle-chinese\", \"URI\": \"subtitle/cn-xx.vtt\" }"
  },
  "Type": "Transcode"
},
"act-report": {
  "Name": "act-report",
  "Parameters": {
    "PublishType": "Auto"
  },
  "Type": "Report"
},
"act-start": {
  "Name": "act-start",
  "Parameters": {
    "PipelineId": "cc7fcef2562e4abc9332d491f93399d2",
    "InputFile": "{ \"Bucket\": \"inputbucketname\", \"Location\": \"oss-cn-hangzhou\", \"ObjectPrefix\": \"package/dash/\" }"
  },
  "Type": "Start"
},
"generateMasterPlayListAct": {
  "Name": "generateMasterPlayListAct",
  "Parameters": {},
  "Type": "GenerateMasterPlayList"
},
"Dependencies": {
  "audio-en-group": ["audioENTransNode"],
  "video-group": ["videoTransLD", "videoTransSD", "videoTransHD"],
  "audio-cn-group": ["audioCNTransNode"],
  "audioCNTransNode": ["generateMasterPlayListAct"],
  "subtitleENNode": ["generateMasterPlayListAct"],
  "act-package": ["audio-en-group", "audio-cn-group", "subtitle-cn-group", "subtitle-en-group", "video-group"],
  "act-report": [],
  "videoTransSD": ["generateMasterPlayListAct"],
  "videoTransHD": ["generateMasterPlayListAct"],
  "subtitle-en-group": ["subtitleENNode"],
  "subtitle-cn-group": ["subtitleCNNode"],
  "subtitleCNNode": ["generateMasterPlayListAct"],
  "act-start": ["act-package"],
  "videoTransLD": ["generateMasterPlayListAct"],
  "generateMasterPlayListAct": ["act-report"],
  "audioENTransNode": ["generateMasterPlayListAct"]
}

```

```
}
```

Sample code

1. Create an HLS packaging workflow

[Create a workflow - Java](#)

[Create a workflow - Python](#)

[Create a workflow - PHP](#)

2. Add media to MPS library

[Add media to MPS library - Java](#)

[Add media to MPS library - Python](#)

[Add media to MPS library - PHP](#)

7.3 Creating an HLS package workflow

Scenario

Pack three video streams respectively in 480P, 720P and 1080P, and output one file, so that you can switch to the most appropriate video stream based on network bandwidth.

Procedure

1. Log on to the [MPS console](#).
2. Select the region.
3. Click **Library > Library Settings**.
4. Click **Workflows > Create Workflow**.
5. Click the icon at the right side of **Input** to add **Output Container** node.
6. Click the icon at the right side of **Config** to add three **Extract Video** nodes.
7. Configure the **Input** node.
 - a. Click the icon at the right side of the **Input** node.
 - b. In **Input**, click **Select** at the right side of **Input Path**.

**Note:**

Input path is a storage location in OSS. The Input path must exist in OSS.

- c. In **OSS File Manager**, select the bucket name and click **OK**.
 - d. **Message Type** is optional. You can select **MNS Queue** or **Notification** and set an instance.
8. Configure the **Config.** node.
- a. Modify **Name**, or you can keep the default name.
 - b. Click the icon at the right side of **Config.** node to configure.
 - c. In **Config.**, click **Select** at the right side of **Output Location**.

**Note:**

Output Location is a storage location in OSS and is a file name. To avoid workflows overwrite output files when executing tasks, you can combine the following built-in UC variable parameters in the system:

- {RunId}: The workflow execution ID,
- {ObjectPrefix}: The path of the original file not including Bucket information,
- {FileName}: The name of the original file not including the extension name,
- {ExtName}: The extension name of the original file.

- d. In **OSS File Manager**, select the bucket name and click **OK**.

**Note:**

The output bucket and the input bucket cannot be the same.

The **Config.** node is configured successfully.

9. Configure the **Extract Video** node.
- a. Click the icon at the right side of the **Extract Video** node.
 - b. Modify **Name**, or you can keep the default name.
 - c. In **Extract Video > Basic Settings**, click **Select** at the right side of **Template**.
 - d. Select the **template** and click **OK**.

e. Configure the **Resource Path**.

We recommend that you use the default resource path. You can also modify the path based on your needs. Note that if the **Output Location** of the **Config.** node is `a/b/c.m3u8`, the **Resource Path** of the Extract Video node is `d/e/f.m3u8`, then the actual storage position of the extracted file is `a/b/d/e/f.m3u8`.

f. In **Audio**, select **Keep**.



Note:

Configure the three **Extract Video** nodes respectively according to the previous procedure, and the transcoding templates correspond to the videos in 480P, 720P and 1080P respectively.

10. Configure the **Generate** node.

- a. Click the icon at the right side of **Generate** to configure.
- b. You can modify the value of **Bandwidth** based on your needs.

11. Click **OK**, and the nodes configuration is completed.

12. Click **Next**.

The workflow is created successfully.

13. Submit the task.

HLS package workflow is not triggered by default. You can use the `AddMedia` interface to specify video and HLS package workflow ID to process videos.