Media Processing Center

SDK Reference

Issue	01
Date	2020-03-31

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 SDK Overview

Media Processing Center (MPC) software development kits (SDKs) provide functions such as creating, canceling, and querying transcoding tasks, and creating, deleting, modifying, and querying transcoding templates.

Currently, the Java SDK and Python SDK are provided. If you have requirements for other development languages, you are advised to use **MPC APIs**.

2 SDK Download

You can download the Java SDK and Python SDK from **HUAWEI CLOUD SDKs**. For details about the Java SDK, see **Java SDK**. For details about the Python SDK, see **Python SDK**. If you encounter technical problems during development, **submit a service ticket** for assistance.

3 Java SDK

3.1 Preparations

Preparing a Development Environment

- JDK 1.8 or later has been installed and the environment has been configured. For details about how to install the JDK, see **JDK Installation**.
- Maven has been **downloaded** and installed.
- Development environments such as Eclipse have been prepared.

Installing and Configuring Maven

Download the **settings.xml** file and overwrite the file with the same name in the *Maven installation directory*/**conf/** directory. If you do not want to overwrite the configuration file, perform the following steps to modify the **settings.xml** file:

1. Add the following content to the profiles node:

```
<profile>
  <id>MyProfile</id>
```



```
  <repositories>
    <repository>
      <id>HuaweiCloudSDK</id>
      <url>https://mirrors.huaweicloud.com/repository/maven/huaweicloudsdk/</url>
    <releases>
       <enabled>true</enabled>
    </releases>
    <snapshots>
    <enabled>false</enabled>
    </snapshots>
    </repository>
  </repositories>

  <pluginRepositories>
     <pluginRepository>
        <id>HuaweiCloudSDK</id>
        <url>https://mirrors.huaweicloud.com/repository/maven/huaweicloudsdk/</url>
        <releases>
        <enabled>true</enabled>
        </releases>
        <snapshots>
        <enabled>false</enabled>
        </snapshots>
     </pluginRepository>
 </pluginRepositories>
</profile>
```

2. Add the following information to the mirrors node:

```
<mirror>
  <id>huaweicloud</id>
  <mirrorOf>*,!HuaweiCloudSDK</mirrorOf>
  <url>https://mirrors.huaweicloud.com/repository/maven/</url>
</mirror>
```

3. Add the **activeProfiles** tag to activate the configurations.

```
<activeProfiles>
   <activeProfile>MyProfile</activeProfile>
</activeProfiles>
```

4. Add the SDK dependencies of MPC to the **pom.xml** file. You can view the latest SDK version of MPC at the **Huawei open-source image site**.

```
<dependencies>
 <dependency>
    <groupId>com.huawei.mpc</groupId>
    <artifactId>cloud-java-sdk-mpc</artifactId>
    <version>${mpc-sdk-version}</version>
  </dependency>
</dependencies>
```

3.2 Transcoding

3.2.1 Creating a Transcoding Task

You can create a transcoding task by creating an MpcClient instance and setting related parameters.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

```
MpcConfig mpcConfig = new MpcConfig();
    mpcConfig.setEndPoint(endPoint);// Set the endpoint.
    mpcConfig.setProjectId(projectId);// Set the project ID.
```



```
mpcConfig.setSk(sk);// Set the SK.
mpcConfig.setAk(ak);// Set the AK.
```

Table 3-1 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
   // Create an MpcClient instance.
   MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
   clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
   clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
   clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
   clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
   MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Create a transcoding request.

   A transcoding request includes information about an input file and output file, and transcoding template settings. For details about the parameters, see **Creating a Transcoding Task**.

   ```
   // Create a transcoding request.
   CreateTranscodingRequest createTranscodingRequest = new CreateTranscodingRequest();

   // Set input file path parameters. You can view the path parameters on the HUAWEI CLOUD OBS console.
   ObsObjInfo input  = new ObsObjInfo();
   // Set the bucket name.
   input.setBucket("bucketName");
   // Set the input file path.
   input.setObject("objectKey");
   // Set the region where an input bucket is deployed.
   input.setLocation("cn-north-4");
   createTranscodingRequest.setInput(input);


   // Set output file path parameters. You can view the path parameters on the HUAWEI CLOUD OBS console.
   ObsObjInfo output  = new ObsObjInfo();
   // Set the bucket name.
   output.setBucket("bucketName");
   ```

```
        // Set the output file path.
        output.setObject("path");
        // Set the region where an output bucket is deployed.
        output.setLocation("cn-north-4");
        createTranscodingRequest.setOutput(output);

        // Set the transcoding template ID.
        // The video bitrate, video height, video width, and audio bitrate in transcoding templates can be changed, but other parameters remain the same.
        // A maximum of nine template IDs are supported.
        List<Long> transTempIds = new ArrayList<Long>();
        transTempIds.add(203L);
        transTempIds.add(212L);
        transTempIds.add(210L);
        createTranscodingRequest.setTransTemplateId(transTempIds);
```

Parameter	Type	Description
bucketName	String	OBS bucket name
location	String	Location of the input OBS bucket
path	String	OBS object path, which complies with the OSS Object definition. If this parameter is used for an input, a specific object must be specified. If this parameter is used for an output, only the directory for storing the output needs to be specified.

4. (Optional) Set transcoding parameters.

   You can select either the transcoding parameters or transcoding template ID. If both the transcoding template ID and transcoding parameters are set, the transcoding parameters are used first.

   - Create a transcoding parameter object.

     ```
     List<AvParmeters> avParmetersList = new LinkedList<>();
     AvParmeters transTemplate = new AvParmeters();
     ```

   - Configure common parameters.

     ```
     BaseCommonBean common = new BaseCommonBean();

     // Set the packaging type (DASH+MP4 or HLS+TS).
     // Describe the packaging format based on the bitmap. Each bit represents a packaging format.
     // 00000001: DASH+MP4
     // 00000010: HLS+TS
     // 00000011: HLS+TS and DASH+MP4
     // 00000100: DASH+MP4 (common)
     // 00000101: TS
     common.setPackType(2);

     // HLS segment interval, in seconds. The value ranges from 2 to 10. The default value is 5.
     common.setHlsInterval(5);

     transTemplate.setCommon(common);
     ```

   - Set video parameters. If you want to disable video, set **video** to **null**.

     ```
     BaseVideoBean video = new BaseVideoBean();

     // Set the average output bitrate, in kbit/s. The value ranges from 40 to 30,000 or is 0. The value 0 indicates adaptive bitrate.
     video.setBitrate(1000);
     ```



```
        // Video codec
        // 1: VIDEO_CODEC_H264
        // 2: VIDEO_CODEC_H265
        // 3: VIDEO_CODEC_AVS2
        video.setCodec(1);

        // Set the frame rate (unit: FPS). The value ranges from 1 to 1,000. The value 1, 2, 3, 4, or
5 indicates the adaptive frame rate.
        video.setFrameRate(30);

         // Set the video width (unit: pixel).
        // H.264: The value is 0 (adaptive) or a multiple of 2 from 32 to 4,096.
        // H.265: The value is 0 (adaptive) or a multiple of 2 from 160 to 4,096.
        video.setWidth(270);

         // Set the video height (unit: pixel).
        // H.264: The value is 0 (adaptive) or a multiple of 2 from 32 to 2,880.
        // H.265: The value is 0 (adaptive) or a multiple of 2 from 96 to 2,880.
        video.setHeight(480);

        // Set whether to enable black bar removal.
        // 0: Disable black bar removal.
        // 1: Enable black bar removal and low-complexity algorithms for long videos (>5 minutes).
        // 2: Enable black bar removal and high-complexity algorithms for short videos (≤5
minutes).
        video.setBlackCut(0);

        transTemplate.setVideo(video);
```

- Set audio parameters. If you want to disable audio, set **audio** to **null**.

```
        BaseAudioBean audio = new BaseAudioBean();

        // Audio codec
        // 1: AUDIO_CODECTYPE_AAC (default)
        // 2: AUDIO_CODECTYPE_HEAAC1
        // 3: AUDIO_CODECTYPE_HEAAC2
        // 4: AUDIO_CODECTYPE_MP3
        audio.setCodec(4);

        // Audio bitrate, in kbit/s. The value is 0 (adaptive) or ranges from 8 to 1,000.
        audio.setBitrate(126);

        // Audio sampling rate
        // 1: AUDIO_SAMPLE_AUTO (default value), indicating that the sampling rate of an output
audio is the same as that of an input audio.
        // 2: AUDIO_SAMPLE_22050
        // 3: AUDIO_SAMPLE_32000
        // 4: AUDIO_SAMPLE_44100
        // 5: AUDIO_SAMPLE_48000
        // 6: AUDIO_SAMPLE_96000
        // If audio.setCodec is 4 (MP3), the value of audio.setSampleRate ranges from 1 to 5.
        audio.setSampleRate(5);

        // Number of audio channels
        // 1: AUDIO_CHANNELS_1
        // 2: AUDIO_CHANNELS_2 (default)
        // 3: AUDIO_CHANNELS_4
        // 4: AUDIO_CHANNELS_5
        // 5: AUDIO_CHANNELS_6
        // 6: AUDIO_CHANNELS_8
        // If audio.setCodec is 4 (MP3), the value of audio.setChannels is 1 or 2.
        audio.setChannels(2);

        transTemplate.setAudio(audio);
```

- Add the transcoding parameters to the transcoding request.

```
        avParmetersList.add(transTemplate);
        req.setAvParmeters(avParmetersList);
```

5. Set transcoding-related parameters.

Set the following parameters based on your needs. If you do not need the parameters, skip them.

- Transcoding priority

```
// Set the transcoding priority. Currently, the value can only be 6 or 9.
// 6: medium (default value ); 9: high
createTranscodingRequest.setPriority(6);
```

- Video processing parameters

```
// Set video processing parameters if needed.
VideoProcess videoProcess = new VideoProcess();
// Adaptive resolution. SHORT (default value): Adaptive width; LONG: Adaptive height; NONE: Do not adapt.
videoProcess.setAdaptation(VideoProcess.AdaptationEnum.LONG);
// Clockwise rotation angle of a video. 0: Do not rotate. 1: Rotate a video clockwise by 90 degrees. 2: Rotate a video clockwise by 180 degrees. 3: Rotate a video clockwise by 270 degrees.
videoProcess.setRotate(1);
createTranscodingRequest.setVideoProcess(videoProcess);
```

- Watermark

  Before enabling watermarking, upload the watermark image to an OBS bucket and obtain the permission to access the OBS bucket. For details, see **Authorizing Access to Cloud Resources**.

```
// Set the watermark function.
CreateTranscodingRequest.Watermark watermark = new CreateTranscodingRequest.Watermark();
watermark.setTemplateId("watermark_template_id");
// Set the watermark image address.
ObsObjInfo watermarkInput  = new ObsObjInfo();
// Bucket that stores the watermark image
watermarkInput.setBucket("bucketName");
// Path to the watermark image
watermarkInput.setObject("objectKey");
// Region where the watermark image resides
watermarkInput.setLocation("cn-north-4");
watermark.setInput(watermarkInput);
// Watermark image parameters used for overwriting fields in the template
ImageWatermark imageWatermark = new ImageWatermark();
// Overwrite the dx parameter and other parameters.
imageWatermark.setDx("30");
watermark.setImageWatermark(imageWatermark);
// Base64-coded image content
// For example, if you want to add the text watermark "test text watermark", the value is 5rWL6K+V5paH5a2X5rC05Y2w.
watermark.setTextContext("5rWL6K+V5paH5a2X5rC05Y2w");
TextWatermark textWatermark = new TextWatermark();
// Font color
// Currently, black, blue, white, green, red, yellow, brown, gold, pink, orange, and purple are supported.
// The default color is white.
textWatermark.setFontColor("black");
// Font. Currently, fzyouh and msyh are supported.
// The default font is msyh.
textWatermark.setFontName("fzyouh");
// Font size. Its value ranges from 4 to 120. The default value is 16.
textWatermark.setFontSize(16);
watermark.setTextWatermark(textWatermark);
List<CreateTranscodingRequest.Watermark> watermarks = new ArrayList<CreateTranscodingRequest.Watermark>();
watermarks.add(watermark);
CreateTranscodingRequest.Watermark[] watermarkArray = new CreateTranscodingRequest.Watermark[watermarks.size()];
createTranscodingRequest.setWatermarks(watermarks.toArray(watermarkArray));
```

- Snapshot

  Before enabling the snapshot function, set an OBS bucket for storing snapshots and obtain the permission to access the bucket. For details, see **Authorizing Access to Cloud Resources**.


```
        // Set snapshot parameters.
        CreateTranscodingRequest.Thumbnail thumbnail = new
CreateTranscodingRequest.Thumbnail();
        // Set which format snapshot files are compressed into. In this example, the .tar format is
used.
        thumbnail.setTar(1);
        // Set the location for storing snapshot files. Ensure that MPC has been authorized to
access the bucket that stores snapshot files.
        ObsObjInfo thumbOutput = new ObsObjInfo();
        // OBS bucket name
        thumbOutput.setBucket("obs-bills");
        // Region where the OBS bucket is located. The OBS bucket must be in the same region as
MPC.
        thumbOutput.setLocation("cn-north-4");
        // Storage path in the OBS bucket
        thumbOutput.setObject("output/thumbnail/");
        thumbnail.setThumbOutput(thumbOutput);
        CreateThumbnailRequest.ThumbnailPara thumbnailPara = new
CreateThumbnailRequest.ThumbnailPara();
        // Sampling type. Three options are available: PERCENT, TIME, and DOTS. PERCENT
indicates sampling based on a percentage of the video duration. Time indicates sampling by
interval. DOTS indicating sampling based on a datetime array. Now only TIME and DOTS are
supported.
        thumbnailPara.setType(CreateThumbnailRequest.ThumbnailPara.TypeEnum.TIME);
       // Interval for sampling
        thumbnailPara.setTime(12);
       // Start time if the sampling type is TIME, which is used together with time. The unit is
second. Its value is a number. The default value is 0.
        thumbnailPara.setStartTime(0);
       // Duration if the sampling type is TIME, which is used together with time and start_time.
The unit is second. Its value is a digit greater than or equal to 0. The default value is ToEND.
ToEND indicates sampling lasts until the end of the video.
        thumbnailPara.setDuration(60);
       // Datetime array for capturing snapshots. This parameter is used only when the sampling
type is DOTS.
        List<Integer> dots = new ArrayList<>();
        dots.add(10);
        dots.add(20);
        thumbnailPara.setDots(dots);

        // Set the maximum length.
        thumbnailPara.setMaxLength(480);
        // Set the aspect ratio.
        thumbnailPara.setAspectRatio(0);
        // Set the snapshot file format. The value 0 indicates the default format, and the value 1
indicates the JPG format.
        thumbnailPara.setFormat(1);
        createTranscodingRequest.setThumbnail(thumbnail);
```

- Content review

```
        // Set content review parameters.
        CreateTranscodingRequest.Audit audit = new CreateTranscodingRequest.Audit();
        // 1: review the input file; 2: review the output file.
        audit.setPosition(1);
        // Set the parameter to the name of the channel to be reviewed (starting from 0). If you
have six outputs and want to review the fourth file, set this to 3. If you want to review only the
input file, you do not need to set this parameter.
        audit.setIndex(0);
        createTranscodingRequest.setAudit(audit);
```

NOTICE

- The resolution of input files and output files must be at least 1280x720.
- If you want to review an output file, its resolution must be the same as the resolution of the input file.

- Encryption

```
// Set encryption parameters.
CreateTranscodingRequest.Encryption encryption = new CreateTranscodingRequest.Encryption();
CreateTranscodingRequest.Encryption.Multidrm multidrm = new CreateTranscodingRequest.Encryption.Multidrm();
// Set content_id.
multidrm.setContentId("contentId");
// Define the data stream type. The value can be DASH or HLS.
multidrm.setStreamingMode("HLS");
// Whether to enable audio encryption. The value 0 indicates that audio encryption is disabled and 1 indicates that audio encryption is enabled. The default value is 0.
multidrm.setEncryptAudio(1);
// Define the encryption method. The default value is 16420.
multidrm.setEmi(16420);
multidrm.setDrmList(new String[]{"PLAYREADY"})
encryption.setMultidrm(multidrm);
createTranscodingRequest.setEncryption(encryption);
```

- Subtitle

  Before enabling subtitling, upload a subtitle file to an OBS bucket and obtain the permission to access the bucket. For details, see **Authorizing Access to Cloud Resources**.

```
// Set subtitle parameters.
CreateTranscodingRequest.Subtitle subtitle = new CreateTranscodingRequest.Subtitle();
// Set whether to add subtitles. 0: no. 1: yes.
subtitle.setSubtitleType(1);
// Set the subtitle file address.
ObsObjInfo fileInput = new ObsObjInfo();
fileInput.setBucket("bucketName");
fileInput.setObject("objectKey");
fileInput.setLocation("cn-north-4");
subtitle.setSubtitleFile(fileInput);
createTranscodingRequest.setSubtitle(subtitle);
```

- Image rotation

```
VideoProcess videoProcess = new VideoProcess();
// Clockwise rotation angle of a video. 0: Do not rotate. 1: Rotate a video clockwise by 90 degrees. 2: Rotate a video clockwise by 180 degrees. 3: Rotate a video clockwise by 270 degrees.
videoProcess.setRotate(2);
createTranscodingRequest.setVideoProcess(videoProcess);
```

- Image enhancement

```
QualityEnhance qualityEnhance = new QualityEnhance();
// For media files with no obvious problems, use technologies such as enhancement and sharpening to improve the subjective effect. The parameters such as the resolution and frame rate remain unchanged before and after this operation.
qualityEnhance.setNormalEnhance(QualityEnhance.NormalEnhanceEnum.NORMAL);
createTranscodingRequest.setQualityEnhance(qualityEnhance);
```

- Volume control

```
AudioProcess audioProcess = new AudioProcess();
// Volume adjustment method. auto indicates automatic volume adjustment. dynamic indicates manual volume adjustment. If the value is dynamic, the volume adjustment amplitude needs to be set.
audioProcess.setVolume(AudioProcess.VolumeEnum.DYNAMIC);
// Volume adjustment amplitude if you plan to adjust the volume manually. The value ranges from -15 to 15. The unit is dB.
audioProcess.setVolumeExpr(15);
createTranscodingRequest.setAudioProcess(audioProcess);
```

- Multi-audio

```
// Set multi-audio track parameters.
 MultiAudio multiAudio = new MultiAudio();
 List<AudioFile> audioFiles = new LinkedList<>();
 AudioFile audioFile = new AudioFile();

 ObsObjInfo multiAudioInput = new ObsObjInfo();
// Set the input file path.
 multiAudioInput.setBucket("obs-syg01");
```

```
        multiAudioInput.setLocation("cn-north-7");
        multiAudioInput.setObject("1.mp3");
        audioFile.setInput(multiAudioInput);
        audioFiles.add(audioFile);
        multiAudio.setAudioFiles(audioFiles);
        createTranscodingRequest.setMultiAudio(multiAudio);
```

6. Send a transcoding request.

```
        // Send a transcoding request.
        CreateTranscodingResponse createTranscodingResponse =
mpcClient.createTranscodingTask(createTranscodingRequest);
       // Return a message.
        System.out.println(new Gson().toJson(createTranscodingResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.ObsObjInfo;
import com.huawei.mpc.model.transcoding.CreateTranscodingRequest;
import com.huawei.mpc.model.transcoding.CreateTranscodingResponse;

    // Set the method for constructing MPC configuration items.
    MpcConfig mpcConfig = new MpcConfig();

    // Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
    mpcConfig.setEndPoint(endPoint);

    // Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
    mpcConfig.setProjectId(projectId);

    // Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
    mpcConfig.setSk(sk);

    // Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
    mpcConfig.setAk(ak);

   /*if you need proxy*/
   // Optional. Set the proxy if necessary.
   /*
      ClientConfig clientConfig = new ClientConfig();
      // Set the IP address of the proxy server.
       clientConfig.setProxyHost(proxyHost);
      // Set the port number of the proxy server.
        clientConfig.setProxyPort(Integer.parseInt(proxyPort));
      // Set the username for accessing the proxy server.
       clientConfig.setProxyUserName(proxyUserName);
      // Set the password for accessing the proxy server.
      clientConfig.setProxyPassword(proxyPassword);
     */


    // MPC construction method
    MpcClient mpcClient = new MpcClient(mpcConfig);

    // Set request parameters.
    CreateTranscodingRequest createTranscodingRequest = new CreateTranscodingRequest();

    // Set the storage location of an input file.
    ObsObjInfo input  = new ObsObjInfo();
    input.setBucket("bucketName");
    input.setObject("objectKey");
    input.setLocation("cn-north-4");
    createTranscodingRequest.setInput(input);


   // Set the storage location of an output file.
    ObsObjInfo output  = new ObsObjInfo();
```



```
        output.setBucket("bucketName");
        output.setObject("path");
        output.setLocation("cn-north-4");
        createTranscodingRequest.setOutput(output);

        /*
        // Optional. Set the watermark function if necessary.
        CreateTranscodingRequest.Watermark watermark = new CreateTranscodingRequest.Watermark();
        watermark.setTemplateId("watermark_template_id");
        // Set the watermark image address.
        ObsObjInfo watermarkInput  = new ObsObjInfo();
        watermarkInput.setBucket("bucketName");
        watermarkInput.setObject("objectKey");
        watermarkInput.setLocation("cn-north-4");
        watermark.setInput(watermarkInput);


        List<CreateTranscodingRequest.Watermark> watermarks = new
ArrayList<CreateTranscodingRequest.Watermark>();
        watermarks.add(watermark);

        CreateTranscodingRequest.Watermark[] watermarkArray = new
CreateTranscodingRequest.Watermark[watermarks.size()];
        createTranscodingRequest.setWatermarks(watermarks.toArray(watermarkArray));
        */

       // Set the transcoding template ID.
        List<Long> transTempIds = new ArrayList<Long>();
        transTempIds.add(203L);
        transTempIds.add(212L);
        transTempIds.add(210L);
        createTranscodingRequest.setTransTemplateId(transTempIds);

        /*
        // Set snapshot parameters if the snapshot function is enabled.
        CreateTranscodingRequest.Thumbnail thumbnail = new CreateTranscodingRequest.Thumbnail();
        // Set which format snapshot files are compressed into. In this example, the .tar format is used.
        thumbnail.setTar(1);
        // Set the location for storing snapshot files. Ensure that MPC has been authorized to access the
bucket that stores snapshot files.
        ObsObjInfo thumbOutput = new ObsObjInfo();
        // OBS bucket name
        thumbOutput.setBucket("obs-bills");
        // Region where an OBS bucket is deployed
        thumbOutput.setLocation("cn-north-4");
        // Storage path in the OBS bucket
        thumbOutput.setObject("output/thumbnail/");
        thumbnail.setThumbOutput(thumbOutput);
        CreateThumbnailRequest.ThumbnailPara thumbnailPara = new
CreateThumbnailRequest.ThumbnailPara();
        // Sampling type. Three options are available: PERCENT, TIME, and DOTS. PERCENT indicates
sampling based on a percentage of the video duration. Time indicates sampling by interval. DOTS
indicating sampling based on a datetime array. Now only TIME and DOTS are supported.
        thumbnailPara.setTime(12);
        // Set the maximum length.
        thumbnailPara.setMaxLength(480);
        // Set the aspect ratio.
        thumbnailPara.setAspectRatio(0);
        // Set the snapshot file format. The value 0 indicates the default format, and the value 1 indicates the
JPG format.
        thumbnailPara.setFormat(1);
        createTranscodingRequest.setThumbnail(thumbnail);

        */
        /*
       // Set the transcoding priority. 6: medium (default value ). 9: high.
        createTranscodingRequest.setPriority(6);
        */
```



```
        /*
        // Set content review parameters.
        CreateTranscodingRequest.Audit audit = new CreateTranscodingRequest.Audit();
        // 1: review the input file; 2: review the output file.
        audit.setPosition(1);
        // Transcoding template index. For details, see the Audit structure in the Media Processing Center API Reference.
        audit.setIndex(0);
        createTranscodingRequest.setAudit(audit);
        */
        /*
        // Set subtitle parameters.
        CreateTranscodingRequest.Subtitle subtitle = new CreateTranscodingRequest.Subtitle();
        // Set whether to add subtitles. 0: no. 1: yes.
        subtitle.setSubtitleType(1);
        // Set the subtitle file address.
        ObsObjInfo fileInput = new ObsObjInfo();
        fileInput.setBucket("bucketName");
        fileInput.setObject("objectKey");
        fileInput.setLocation("cn-north-4");
        subtitle.setSubtitleFile(fileInput);
        createTranscodingRequest.setSubtitle(subtitle);
        */

        // Set encryption parameters.
        CreateTranscodingRequest.Encryption encryption = new CreateTranscodingRequest.Encryption();
        CreateTranscodingRequest.Encryption.Multidrm multidrm = new CreateTranscodingRequest.Encryption.Multidrm();
        // Set content_id.
        multidrm.setContentId("contentId");
        // Define the data stream type. The value can be DASH or HLS.
        multidrm.setStreamingMode("HLS");
        // Whether to enable audio encryption. The value 0 indicates that audio encryption is disabled and 1 indicates that audio encryption is enabled. The default value is 0.
        multidrm.setEncryptAudio(1);
        // Define the encryption method. The default value is 16420.
        multidrm.setEmi(16420);
        multidrm.setDrmList(new String[]{"PLAYREADY"})
        encryption.setMultidrm(multidrm);
        createTranscodingRequest.setEncryption(encryption);
        // Set image rotation parameters.
        VideoProcess videoProcess = new VideoProcess();
        // Clockwise rotation angle of a video. 0: Do not rotate. 1: Rotate a video clockwise by 90 degrees. 2: Rotate a video clockwise by 180 degrees. 3: Rotate a video clockwise by 270 degrees.
        videoProcess.setRotate(2);
        createTranscodingRequest.setVideoProcess(videoProcess);

        // Set image enhancement parameters.
        QualityEnhance qualityEnhance = new QualityEnhance();
        // For media files with no obvious problems, use technologies such as enhancement and sharpening to improve the subjective effect. The parameters such as the resolution and frame rate remain unchanged before and after this operation.
        qualityEnhance.setNormalEnhance(QualityEnhance.NormalEnhanceEnum.NORMAL);
        createTranscodingRequest.setQualityEnhance(qualityEnhance);

        // Set volume adjustment parameters.
        AudioProcess audioProcess = new AudioProcess();
        // Volume adjustment method. auto indicates automatic volume adjustment. dynamic indicates manual volume adjustment. If the value is dynamic, the volume adjustment amplitude needs to be set.
        audioProcess.setVolume(AudioProcess.VolumeEnum.DYNAMIC);
        // Volume adjustment amplitude if you plan to adjust the volume manually. The value ranges from -15 to 15. The unit is dB.
        audioProcess.setVolumeExpr(15);
        createTranscodingRequest.setAudioProcess(audioProcess);

         // Send a transcoding request.
         CreateTranscodingResponse createTranscodingResponse = mpcClient.createTranscodingTask(createTranscodingRequest);
```



```
// Return a message.
System.out.println(new Gson().toJson(createTranscodingResponse));
```

3.2.2 Canceling a Transcoding Task

Notes

- To cancel a transcoding task, you need to provide the task ID.
- The task to be canceled must be a task in the task queue. A transcoding task that has started or completed cannot be canceled.
- For details about error handling, see **Error Codes**.

Setting the Parameters for Canceling Transcoding

```
// Set the task ID.
String taskId="taskId";
// Send a request for canceling the transcoding task.
BaseResponse baseResponse=mpcClient.deleteTranscodingTask(taskId);
// Return the result.
System.out.println(new Gson().toJson(baseResponse));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and
authorization.
* 3. Set request parameters for canceling a transcoding task.
* 4. Send a request.
* 5. Return the result.
*/

import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

    // Set the method for constructing MPC configuration items.
     MpcConfig mpcconfig = new MpcConfig();
     // Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
      mpcConfig.setEndPoint(endPoint);

      // Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
      mpcConfig.setProjectId(projectId);

      // Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
      mpcConfig.setSk(sk);

      // Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
      mpcConfig.setAk(ak);

     /*if you need proxy*/
    // Optional. Set the proxy if necessary.
     /*
        ClientConfig clientConfig = new ClientConfig();
       // Set the IP address of the proxy server.
         clientConfig.setProxyHost(proxyHost);
       // Set the port number of the proxy server.
          clientConfig.setProxyPort(Integer.parseInt(proxyPort));
        // Set the username for accessing the proxy server.
         clientConfig.setProxyUserName(proxyUserName);
       // Set the password for accessing the proxy server.
         clientConfig.setProxyPassword(proxyPassword);
```

```
     */

    // MPC construction method
     MpcClient mpcClient = new MpcClient(mpcconfig);
    /*
      * proxy provided
       * MpcClient mpcClient=new MpcClient(mpcconfig, clientConfig);
       */

     // Set the task ID.
     String taskId="taskId";
    // Send a request for canceling the transcoding task.
     BaseResponse baseResponse=mpcClient.deleteTranscodingTask(taskId);
    // Return the result.
    System.out.println(new Gson().toJson(baseResponse));
```

3.2.3 Querying Transcoding Tasks

You can query details about one or more transcoding tasks by task ID, task status, page number, start time, and end time.

In the query results, if the page number and maximum number of displayed records are not specified, but the number of records is greater than 10, 10 records are displayed by default on each page.

For details about the search criteria and search result parameters, see the API for **querying transcoding tasks**.

Querying a Transcoding Task

```
String taskId="593";
// Set the task ID.
queryTranscodingRequest.setTaskId(taskId);
// Send a query request.
QueryTranscodingResponse
queryTranscodingResponse=mpcClient.queryTranscodingTask(queryTranscodingRequest);
System.out.println(new Gson().toJson(queryTranscodingResponse));
```

Querying Transcoding Tasks

```
// A maximum of 10 transcoding task IDs can be queried at a time.
String[] taskIds={"593", "595", "594","597"};
// Set task IDs.
queryTranscodingRequest.setTaskIdArray(taskIds);
// Send a query request.
QueryTranscodingResponse
queryTranscodingResponse=mpcClient.queryTranscodingTask(queryTranscodingRequest);
System.out.println(new Gson().toJson(queryTranscodingResponse));
```

Query by Task Status

```
String status="CANCELED";
// Set the task status.
queryTranscodingRequest.setStatus(status);
// Send a query request.
QueryTranscodingResponse
queryTranscodingResponse=mpcClient.queryTranscodingTask(queryTranscodingRequest);
System.out.println(new Gson().toJson(queryTranscodingResponse));
```

Query by Start Time and End Time

```
// Query based on the start time and end time of a transcoding task.
// Set the start time.
queryTranscodingRequest.setStartTime("20180116125625");
// Set the end time.
```



```
queryTranscodingRequest.setEndTime("20180118132810");
// Send a query request.
QueryTranscodingResponse
queryTranscodingResponse=mpcClient.queryTranscodingTask(queryTranscodingRequest);
System.out.println(new Gson().toJson(queryTranscodingResponse));
```

Query by Page Number

```
// Query based on the page number and number of records on each page.
// Set the page number.
queryTranscodingRequest.setPage(0);
// Set the number of records on each page.
queryTranscodingRequest.setSize(4);
// Send a query request.
QueryTranscodingResponse
queryTranscodingResponse=mpcClient.queryTranscodingTask(queryTranscodingRequest);
System.out.println(new Gson().toJson(queryTranscodingResponse));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC.
* 3. Set request parameters.
* 4. Send a request for querying a transcoding task.
* 5. Return the results.
*/

import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.transcoding.QueryTrancodingResponse;
import com.huawei.mpc.model.transcoding.QueryTranscodingRequest;

// Set the method for constructing MPC configuration items.
MpcConfig mpcconfig = new MpcConfig();
// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// MPC construction method
MpcClient mpcClient = new MpcClient(mpcconfig);
/*
* proxy provided
// MPC construction method. If a proxy needs to be configured, use this construction method.
* MpcClient mpcClient=new MpcClient(mpcconfig, clientConfig);
```



```
*/

// Set task parameters.
QueryTranscodingRequest queryTranscodingRequest=new QueryTranscodingRequest();
/**
* Single task query
*/
// Query a single transcoding task by task ID.
/*String taskId="";
// Set the task ID.
queryTranscodingRequest.setTaskId(taskId);*/

/**
* Multitasking query
*/
// A maximum of 10 transcoding task IDs can be queried at a time.
String[] taskIds={"593", "595", "594","597"};
// Set task IDs.
queryTranscodingRequest.setTaskIdArray(taskIds);

/**
* According to the status query
*/
/*String status="CANCELED";
// Set the task status.
queryTranscodingRequest.setStatus(status);*/

/**
* According to the page&size query,If you do not provide default display 10 data
*/
// Query based on the page number and number of records on each page.
// Set the page number.
/*queryTranscodingRequest.setPage(0);
// Set the number of records on each page.
queryTranscodingRequest.setSize(4);*/

/**
* According to the startTime&endTime query
*/
// Query by start time and end time
// Set the start time.
/*queryTranscodingRequest.setStartTime("20180116125625");
// Set the end time.
queryTranscodingRequest.setEndTime("20180118132810");*/

// Send a query request.
QueryTranscodingResponse
queryTranscodingResponse=mpcClient.queryTranscodingTask(queryTranscodingRequest);

/*response*/
// Return the query results.
System.out.println(new Gson().toJson(queryTranscodingResponse));
```

3.3 Snapshot Capturing

3.3.1 Creating a Snapshot Task

Prerequisites

- You have bought Object Storage Service (OBS) resources and uploaded an input file to an OBS bucket which is in the same region (for example, CN North-Beijing4) as MPC by referring to **Uploading Media Files**.
- MPC has been authorized to access OBS resources. For details, see **Authorizing Access to Cloud Resources**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
       mpcConfig.setEndPoint(endPoint);// Set the endpoint.
       mpcConfig.setProjectId(projectId);// Set the project ID.
       mpcConfig.setSk(sk);// Set the SK.
       mpcConfig.setAk(ak);// Set the AK.
   ```

   Table 3-2 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
       // Create an MpcClient instance.
       MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
         clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
         clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
         clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy
   server.
         clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
    MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Create a request for creating a snapshot task.

   The request contains the input file and output file paths. For details about the parameters, see **Creating a Snapshot Task**.

   ```
   // Set request parameters.
   CreateThumbnailRequest createThumbnailRequest = new CreateThumbnailRequest();

   // Set the storage location of an input file.
   ObsObjInfo input  = new ObsObjInfo();
   // OBS bucket name
   input.setBucket("bucketName");
   // OBS path
   input.setObject("objectKey");
   // Region where an OBS bucket is deployed
   input.setLocation("cn-north-4");
   ```

```
createThumbnailRequest.setInput(input);

// Set the storage location of an output file.
ObsObjInfo output  = new ObsObjInfo();
output.setBucket("bucketName");
output.setObject("path");
output.setLocation("cn-north-4");
createThumbnailRequest.setOutput(output);
```

4. Set snapshot parameters.

```
// Set snapshot parameters.
CreateThumbnailRequest.ThumbnailPara thumbnailPara = new
CreateThumbnailRequest.ThumbnailPara();

// Sampling type. Three options are available: PERCENT, TIME, and DOTS. PERCENT indicates sampling based on a percentage of the video duration. Time indicates sampling by interval. DOTS indicating sampling based on a datetime array. Now only TIME and DOTS are supported.
thumbnailPara.setType(CreateThumbnailRequest.ThumbnailPara.TypeEnum.TIME);
// Interval for sampling
thumbnailPara.setTime(12);
// Start time if the sampling type is TIME, which is used together with time. The unit is second. Its value is a number. The default value is 0.
thumbnailPara.setStartTime(0);
// Duration if the sampling type is TIME, which is used together with time and start_time. The unit is second. Its value is a digit greater than or equal to 0. The default value is ToEND. ToEND indicates sampling lasts until the end of the video.
thumbnailPara.setDuration(60);
// Datetime array for capturing snapshots. This parameter is used only when the sampling type is DOTS.
List<Integer> dots = new ArrayList<>();
dots.add(10);
dots.add(20);
thumbnailPara.setDots(dots);
// Set the snapshot width. Its value ranges from 380 to 3,840. The snapshot height is scaled based on the ratio between the size and the input video pixel.
thumbnailPara.setMaxLength(480);
// Set the aspect ratio (min = 0, max = 1).
thumbnailPara.setAspectRatio(0);
// Set the snapshot file format. The value 0 indicates the default format, and the value 1 indicates the JPG format.
thumbnailPara.setFormat(1);
createThumbnailRequest.setThumbnailPara(thumbnailPara);
```

Note: A snapshot file is named based on the timestamp. Capture the first and last frames. The middle part is captured by interval. For example, if a video lasts 20s and the snapshot interval is 11s, the generated snapshot files are named as 0.jpg, 11.jpg, and 20.jpg.

5. Send a request and return a message.

```
// Send a request to MPC.
CreateThumbnailResponse createThumbnailResponse =
mpcClient.createThumbnailsTask(createThumbnailRequest);

// Return a message.
System.out.println(new Gson().toJson(createThumbnailResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.ObsObjInfo;
import com.huawei.mpc.model.thumbnail.CreateThumbnailRequest;
import com.huawei.mpc.model.thumbnail.CreateThumbnailResponse;
import java.util.ArrayList;
import java.util.List;
// Set the method for constructing MPC configuration items.
MpcConfig mpcConfig = new MpcConfig();
```



```
// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// Set request parameters.
CreateThumbnailRequest createThumbnailRequest = new CreateThumbnailRequest();

// Set the storage location of an input file.
ObsObjInfo input  = new ObsObjInfo();
// OBS bucket name
input.setBucket("bucketName");
// OBS path
input.setObject("objectKey");
// Region where an OBS bucket is deployed
input.setLocation("cn-north-4");
createThumbnailRequest.setInput(input);

// Set the storage location of an output file.
ObsObjInfo output  = new ObsObjInfo();
output.setBucket("bucketName");
output.setObject("path");
output.setLocation("cn-north-4");
createThumbnailRequest.setOutput(output);

// Set snapshot parameters.
CreateThumbnailRequest.ThumbnailPara thumbnailPara = new CreateThumbnailRequest.ThumbnailPara();
// Sampling type. Three options are available: PERCENT, TIME, and DOTS. PERCENT indicates sampling
based on a percentage of the video duration. Time indicates sampling by interval. DOTS indicating
sampling based on a datetime array. Now only TIME and DOTS are supported.
thumbnailPara.setTime(12);
// Set the maximum length. Its value ranges from 380 to 3,840.
thumbnailPara.setMaxLength(480);
// Set the aspect ratio (min = 0, max = 1).
thumbnailPara.setAspectRatio(0);
// Set the snapshot file format. The value 0 indicates the default format, and the value 1 indicates the JPG
format.
thumbnailPara.setFormat(1);
createThumbnailRequest.setThumbnailPara(thumbnailPara);

// Send a request to MPC.
CreateThumbnailResponse createThumbnailResponse =
mpcClient.createThumbnailsTask(createThumbnailRequest);

// Return a message.
System.out.println(new Gson().toJson(createThumbnailResponse));
```


3.3.2 Canceling a Snapshot Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be a task in the task queue. A snapshot task that has been started or completed cannot be deleted.

Core Code

```
// Send a request to MPC.
BaseResponse baseResponse = mpcClient.deleteThumbnailTask("23456");

// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

// Set the method for constructing MPC configuration items.
MpcConfig mpcConfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// MPC construction method
MpcClient mpcClient = new MpcClient(mpcConfig);

// Send a request to MPC.
BaseResponse baseResponse = mpcClient.deleteThumbnailTask("23456");

// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

3.3.3 Querying Snapshot Tasks

Notes

- You can query snapshot tasks by task ID, task status, time range, page number, or compound query.
- In the query results, if the page number and maximum number of displayed records are not specified, but the number of records is greater than 10, 10 records are displayed by default on each page.

Query by Task ID

```
// A maximum of 10 task IDs are supported.
queryThumbTaskRequest.setTaskId(new String[]{"1234","2345","3456"});

// Send a request to MPC.
QueryThumbTaskResponse queryThumbTaskResponse =
mpcClient.queryThumbnailsTask(queryThumbTaskRequest);
// Return a message.
System.out.println(new Gson().toJson(queryThumbTaskResponse));
```

Query by Page Number

```
// Set the page number and number of records on each page.
queryThumbTaskRequest.setPage(1);
queryThumbTaskRequest.setSize(10);
// Send a request to MPC.
QueryThumbTaskResponse queryThumbTaskResponse =
mpcClient.queryThumbnailsTask(queryThumbTaskRequest);
// Return a message.
System.out.println(new Gson().toJson(queryThumbTaskResponse));
```

Query by Time Range

```
// Set the start time and end time.
queryThumbTaskRequest.setStartTime("20180520131400");
queryThumbTaskRequest.setEndTime("20180520141300");
// Send a request to MPC.
QueryThumbTaskResponse queryThumbTaskResponse =
mpcClient.queryThumbnailsTask(queryThumbTaskRequest);
// Return a message.
System.out.println(new Gson().toJson(queryThumbTaskResponse));
```

Query by Task Status

```
// Set the task status.
// queryThumbTaskRequest.setStatus(QueryThumbTaskRequest.STATUS_SUCCEEDED);
// Send a request to MPC.
QueryThumbTaskResponse queryThumbTaskResponse =
mpcClient.queryThumbnailsTask(queryThumbTaskRequest);
// Return a message.
System.out.println(new Gson().toJson(queryThumbTaskResponse));
```

Compound Query

```
// Set the following parameters:
queryThumbTaskRequest.setPage(1);
queryThumbTaskRequest.setSize(10);
queryThumbTaskRequest.setStartTime("20180520131400");
queryThumbTaskRequest.setEndTime("20180520141300");
queryThumbTaskRequest.setStatus(QueryThumbTaskRequest.STATUS_SUCCEEDED);
// Send a request to MPC.
QueryThumbTaskResponse queryThumbTaskResponse =
```



```
mpcClient.queryThumbnailsTask(queryThumbTaskRequest);
// Return a message.
System.out.println(new Gson().toJson(queryThumbTaskResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.thumbnail.QueryThumbTaskRequest;
import com.huawei.mpc.model.thumbnail.QueryThumbTaskResponse;
// Set the method for constructing MPC configuration items.
MpcConfig mpcConfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// MPC construction method
MpcClient mpcClient = new MpcClient(mpcConfig);

// Set request parameters.
QueryThumbTaskRequest queryThumbTaskRequest = new QueryThumbTaskRequest();

// 1. Query by task ID. A maximum of 10 task IDs are supported.
queryThumbTaskRequest.setTaskId(new String[]{"1234","2345","3456"});

// 2. Query by page number
// queryThumbTaskRequest.setPage(1);
// queryThumbTaskRequest.setSize(10);

// 3. Query by time range
// queryThumbTaskRequest.setStartTime("20180520131400");
// queryThumbTaskRequest.setEndTime("20180520141300");

// 4. Query by task status
// queryThumbTaskRequest.setStatus(QueryThumbTaskRequest.STATUS_SUCCEEDED);

// 5. Compound query
// queryThumbTaskRequest.setPage(1);
// queryThumbTaskRequest.setSize(10);
// queryThumbTaskRequest.setStartTime("20180520131400");
// queryThumbTaskRequest.setEndTime("20180520141300");
// queryThumbTaskRequest.setStatus(QueryThumbTaskRequest.STATUS_SUCCEEDED);

// Send a request to MPC.
QueryThumbTaskResponse queryThumbTaskResponse =
```



```
mpcClient.queryThumbnailsTask(queryThumbTaskRequest);
// Return a message.
System.out.println(new Gson().toJson(queryThumbTaskResponse));
```

3.4 Encryption

3.4.1 Creating an Encryption Task

You can create an encryption task by creating an MpcClient instance and setting related parameters.

Prerequisites

- You have bought Object Storage Service (OBS) resources and uploaded an input file to an OBS bucket which is in the same region (for example, CN North-Beijing4) as MPC by referring to **Uploading Media Files**.
- MPC has been authorized to access OBS resources. For details, see **Authorizing Access to Cloud Resources**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
   mpcConfig.setEndPoint(endPoint);// Set the endpoint.
   mpcConfig.setProjectId(projectId);// Set the project ID.
   mpcConfig.setSk(sk);// Set the SK.
   mpcConfig.setAk(ak);// Set the AK.
   ```

   Table 3-3 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
   // Create an MpcClient instance.
    MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

```
// Configure the proxy.
ClientConfig clientConfig = new ClientConfig();
      clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
      clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
      clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
      clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
// Use the constructor to initialize MpcClient.
 MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
```

3. Create an encryption request.

   The request includes the input file, output file, and encryption parameter settings.

```
      CreateEncryptRequest createEncryptRequest = new CreateEncryptRequest();
      ObsObjInfo input = new ObsObjInfo();
      // Set the OBS bucket for storing an input file.
      input.setBucket("obs-bills");
      // Set the region where the OBS bucket is located. You can view it on the OBS console.
      input.setLocation("cn-north-4");
      // Set the input file path.
      input.setObject("bills/index.m3u8");
      createEncryptRequest.setInput(input);

      ObsObjInfo output = new ObsObjInfo();
      // Set the OBS bucket for storing an input file.
      output.setBucket("obs-bills");
      // Set the region where the OBS bucket is located. You can view it on the OBS console.
      output.setLocation("cn-north-4");
      // Set the output file path.
      output.setObject("output/");
      createEncryptRequest.setOutput(output);

     // Set encryption parameters.
      CreateEncryptRequest.Encryption encryption = new CreateEncryptRequest.Encryption();

     /* CreateEncryptRequest.Encryption.Multidrm multidrm = new CreateEncryptRequest.Encryption.Multidrm();
      * // Set content_id.
      * multidrm.setContentId("contentId");
      * // Define the data stream type. The value can be DASH or HLS.
      * multidrm.setStreamingMode("streamingMode");
      * // Whether to enable audio encryption. The value 0 indicates that audio encryption is disabled and 1 indicates that audio encryption is enabled. The default value is 0.
      * multidrm.setEncryptAudio(1);
      * // Define the encryption method. The default value is 16420.
      * multidrm.setEmi(16420);
      * encryption.setMultidrm(multidrm);
       */

      CreateEncryptRequest.Encryption.HlsEncrypt hlsEncrypt = new CreateEncryptRequest.Encryption.HlsEncrypt();

      // Set encryption algorithms.
      hlsEncrypt.setAlgorithm("AES-128-CBC");

      // Set the key.
      hlsEncrypt.setKey("EpFDp4T1yyVA7YTB3QFNcw==");

      // Set the initialization vector.
      hlsEncrypt.setIv("nsADTFTiLVj7VCZ76HkhEw==");

      // Set the URL to obtain the key.
      hlsEncrypt.setUrl("http://10.154.193.244:16951/test/9b429fd8be7e47d89d6a6e1d34541136/d90e2dbacdb1619461d4012945c46175/asset/get_key");
      encryption.setHlsEncrypt(hlsEncrypt);
```



```
createEncryptRequest.setEncryption(encryption);
CreateEncryptResponse response = mpcClient.createEncryptTask(createEncryptRequest);
```

Parameter	Type	Description
bucketName	String	OBS bucket name
location	String	Location of the input OBS bucket
path	String	OBS object path, which complies with the OSS Object definition. If this parameter is used for an input, a specific object must be specified. If this parameter is used for an output, only the directory for storing the output needs to be specified.

4. (Mandatory) Set encryption parameters.

```
// Set encryption parameters.
CreateTranscodingRequest.Encryption encryption = new CreateTranscodingRequest.Encryption();
CreateTranscodingRequest.Encryption.Multidrm multidrm = new
CreateTranscodingRequest.Encryption.Multidrm();
// Set content_id.
multidrm.setContentId("contentId");
// Define the data stream type. The value can be DASH or HLS.
multidrm.setStreamingMode("HLS");
// Whether to enable audio encryption. The value 0 indicates that audio encryption is disabled
and 1 indicates that audio encryption is enabled. The default value is 0.
multidrm.setEncryptAudio(1);
// Define the encryption method. The default value is 16420.
multidrm.setEmi(16420);
multidrm.setDrmList(new String[]{"PLAYREADY"})
encryption.setMultidrm(multidrm);
createTranscodingRequest.setEncryption(encryption);
```

Full Code

```
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;
import com.huawei.mpc.model.ObsObjInfo;
import com.huawei.mpc.model.encrypt.CreateEncryptRequest;
import com.huawei.mpc.model.encrypt.CreateEncryptResponse;
import com.huawei.mpc.model.encrypt.QueryEncryptRequest;
import com.huawei.mpc.model.encrypt.QueryEncryptResponse;
importjava.util.ArrayList;
importjava.util.List;

    MpcConfig mpcConfig = new MpcConfig();

    // Authentication method 1: AK/SK authentication (default method). For details about how to obtain
the AK/SK pair, see section "Obtaining Key Parameters."
    // mpcConfig.setAk("YS0JVWUSSQ23QL2PBMVH");
    // mpcConfig.setSk("HCxKJ2plAR8h5kmP3jOeaXo9sXAb1ROeOMcKOnkR");

    // Set the language. Chinese and English are supported. Chinese is the default language.
    // mpcConfig.setLanguage(MpcConfig.LanguageEnum.EN_US.getName());

    // Authentication method 2: obtain temporary authorization and then perform authentication.
    /*
     * mpcConfig.setAk("");
```



```
        * mpcConfig.setSk("");
        * mpcConfig.setSecurityToken("");
        */

       // Authentication method 3: authentication using the username and password (DomainName is the same as the username by default). For details, see section "Obtaining Key Parameters."
       mpcConfig.setUsername("VOD_wwx530051");
       mpcConfig.setPassword("Huawei@123");
       mpcConfig.setDomainName("VOD_wwx530051");
       mpcConfig.setIamEndPoint("192.144.1.37:31943");
       // mpcConfig.setIamEndPoint("192.145.63.209:31943");

       // Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
       mpcConfig.setEndPoint("mts.cn-north-4.myhuaweicloud.com");
       // Set the project ID. For details, see section "Obtaining Key Parameters."
       mpcConfig.setProjectId("14ce1d4437164aba8b364ce15866154e");
       mpcClient = new MpcClient(mpcConfig);

       CreateEncryptRequest createEncryptRequest = new CreateEncryptRequest();
       ObsObjInfo input = new ObsObjInfo();
       // Set the OBS bucket for storing an input file.
       input.setBucket("obs-bills");
       // Set the region where the OBS bucket is located. You can view it on the OBS console.
       input.setLocation("cn-north-4");
       // Set the input file path.
       input.setObject("bills/1080P_2.mp4");
       createEncryptRequest.setInput(input);

       ObsObjInfo output = new ObsObjInfo();
       // Set the OBS bucket for storing an input file.
       output.setBucket("obs-bills");
       // Set the region where the OBS bucket is located. You can view it on the OBS console.
       output.setLocation("cn-north-4");
       // Set the output file path.
       output.setObject("output/");
       createEncryptRequest.setOutput(output);

      // Set encryption parameters.
       CreateEncryptRequest.Encryption encryption = new CreateEncryptRequest.Encryption();

       /*
        * CreateEncryptRequest.Encryption.Multidrm multidrm = new CreateEncryptRequest.Encryption.Multidrm();
        * // Set content_id.
        *  multidrm.setContentId("contentId");
       * // Define the data stream type. The value can be DASH or HLS.
        * multidrm.setStreamingMode("streamingMode");
        * // Whether to enable audio encryption. The value 0 indicates that audio encryption is disabled and 1 indicates that audio encryption is enabled. The default value is 0.
        * multidrm.setEncryptAudio(1);
        * // Define the encryption method.
        *  multidrm.setEmi(16420);
        *  encryption.setMultidrm(multidrm);
        */

       CreateEncryptRequest.Encryption.HlsEncrypt hlsEncrypt = new CreateEncryptRequest.Encryption.HlsEncrypt();

       // Set encryption algorithms.
       hlsEncrypt.setAlgorithm("AES-128-CBC");

       // Set the key.
       hlsEncrypt.setKey("EpFDp4T1yyVA7YTB3QFNcw==");

       // Set the initialization vector.
       hlsEncrypt.setIv("nsADTFTiLVj7VCZ76HkhEw==");

       // Set the URL to obtain the key.
       hlsEncrypt.setUrl("http://10.154.193.244:16951/test/9b429fd8be7e47d89d6a6e1d34541136/
```



```
d90e2dbacdb1619461d4012945c46175/asset/get_key");
        encryption.setHlsEncrypt(hlsEncrypt);

        createEncryptRequest.setEncryption(encryption);
        CreateEncryptResponse response = mpcClient.createEncryptTask(createEncryptRequest);
```

3.4.2 Canceling an Encryption Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be a task in the task queue. An encryption task that has been started or completed cannot be deleted.

Core Code

```
// Send a request to MPC.
BaseResponse baseResponse = mpcClient.deleteEncryptTask("23456");

// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

// Set the method for constructing MPC configuration items.
MpcConfig mpcConfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// Create an MpcClient instance.
MpcClient mpcClient = new MpcClient(mpcConfig);

// Send a request to MPC.
BaseResponse baseResponse = mpcClient.deleteEncryptTask("23456");

// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```


3.4.3 Querying Encryption Tasks

Notes

- You can query encryption tasks by task ID, task status, time range, page number, or compound query.
- In the query results, if the page number and maximum number of displayed records are not specified, but the number of records is greater than 10, 10 records are displayed by default on each page.

Query by Task ID

```
// A maximum of 10 task IDs are supported.
QueryEncryptRequest queryEncryptRequest = new QueryEncryptRequest();
queryEncryptRequest .setTaskIds(new String[]{"1234","2345","3456"});

// Send a request to MPC.
QueryEncryptResponse rsp = mpcClient.queryEncryptTask(queryEncryptRequest);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Query by Page Number

```
// Set the page number and number of records on each page.
QueryEncryptRequest queryEncryptRequest = new QueryEncryptRequest();
queryEncryptRequest.setPage(1);
queryEncryptRequest.setSize(10);
// Send a request to MPC.
QueryEncryptResponse rsp = mpcClient.queryEncryptTask(queryEncryptRequest);
// Return a message.
System.out.println(new Gson().toJson(rsp ));
```

Query by Time Range

```
// Set the start time and end time.
QueryEncryptRequest queryEncryptRequest = new QueryEncryptRequest();
queryEncryptRequest.setStartTime("20180520131400");
queryEncryptRequest.setEndTime("20180520141300");
// Send a request to MPC.
QueryEncryptResponse rsp = mpcClient.queryEncryptTask(queryEncryptRequest);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Query by Task Status

```
// Set the task status.
QueryEncryptRequest queryEncryptRequest = new QueryEncryptRequest();
queryEncryptRequest.setStatus(QueryEncryptRequest.STATUS_SUCCEEDED);
// Send a request to MPC.
QueryEncryptResponse rsp = mpcClient.queryEncryptTask(queryEncryptRequest);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Compound Query

```
// Set the following parameters:
QueryEncryptRequest queryEncryptRequest = new QueryEncryptRequest();
queryEncryptRequest.setPage(1);
queryEncryptRequest.setSize(10);
queryEncryptRequest.setStartTime("20180520131400");
quequeryEncryptRequest.setEndTime("20180520141300");
queryEncryptRequest.setStatus(QueryEncryptRequest.STATUS_SUCCEEDED);
// Send a request to MPC.
```

```
QueryEncryptResponse rsp = mpcClient.queryEncryptTask(queryEncryptRequest);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.encrypt.QueryEncryptRequest;
import com.huawei.mpc.model.encrypt.QueryEncryptResponse;
// Set the method for constructing MPC configuration items.
MpcConfig mpcConfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/


// MPC construction method
MpcClient mpcClient = new MpcClient(mpcConfig);


// Set request parameters.
QueryEncryptRequest queryEncryptRequest = new QueryEncryptRequest();

// 1. Query by task ID. A maximum of 10 task IDs are supported.
queryEncryptRequest.setTaskIds(new String[]{"1234","2345","3456"});

// 2. Query by page number
// queryEncryptRequest.setPage(1);
// queryEncryptRequest.setSize(10);

// 3. Query by time range
// queryEncryptRequest.setStartTime("20180520131400");
// queryEncryptRequest.setEndTime("20180520141300");

// 4. Query by task status
// queryEncryptRequest.setStatus(QueryEncryptRequest.STATUS_SUCCEEDED);

// 5. Compound query
// queryEncryptRequest.setPage(1);
// queryEncryptRequest.setSize(10);
// queryEncryptRequest.setStartTime("20180520131400");
// queryEncryptRequest.setEndTime("20180520141300");
// queryEncryptRequest.setStatus(QueryEncryptRequest.STATUS_SUCCEEDED);
```



```
// Send a request to MPC.
QueryEncryptResponse rsp = mpcClient.queryEncryptTask(queryEncryptRequest);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

3.5 Animated GIF Management

3.5.1 Creating an Animated GIF Task

You can create an animated GIF task by creating an MpcClient instance and setting related parameters. The task is used to convert a video to animated GIF.

Prerequisites

- You have bought Object Storage Service (OBS) resources and uploaded an input file to an OBS bucket which is in the same region (for example, CN North-Beijing4) as MPC by referring to **Uploading Media Files**.
- MPC has been authorized to access OBS resources. For details, see **Authorizing Access to Cloud Resources**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
       mpcConfig.setEndPoint(endPoint);// Set the endpoint.
       mpcConfig.setProjectId(projectId);// Set the project ID.
       mpcConfig.setSk(sk);// Set the SK.
       mpcConfig.setAk(ak);// Set the AK.
   ```

   Table 3-4 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
   // Create an MpcClient instance.
   MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

```
// Configure the proxy.
ClientConfig clientConfig = new ClientConfig();
      clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
      clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
      clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy
server.
      clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
// Use the constructor to initialize MpcClient.
 MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
```

3. Create an animated GIF task.

   Set the parameters such as the input video file, output file path, output image frame rate, and output image width and height.

```
      CreateAnimatedGraphicsTaskReq req = new CreateAnimatedGraphicsTaskReq();

      // Set the input file path.
      ObsObjInfo input = new ObsObjInfo();
      input.setBucket("obs-test01");
      input.setLocation("cn-north-4");
      input.setObject("volume_auto_test.mp4");
      req.setInput(input);

     // Set the output file path.
      ObsObjInfo output = new ObsObjInfo();
      output.setBucket("obs-test01");
      output.setLocation("cn-north-4");
      output.setObject("/e7404a3fa1b547919d659b0cbfa268c1/animate-sdk-test/");
      output.setFileName("my-test.gif");
      req.setOutput(output);

      // Set output parameters.
      AnimatedGraphicsOutputParam outputParam = new AnimatedGraphicsOutputParam();
      outputParam.setStart(1_000);
      outputParam.setEnd(5_000);
      outputParam.setFormat(AnimatedGraphicsOutputParam.FormatEnum.GIF);
      outputParam.setFrameRate(15);
      outputParam.setWidth(1024);
      outputParam.setHeight(768);
      req.setOutputParam(outputParam);

    // Send a request.
      CommonCreateTaskRsp rsp = mpcClient.createAnimatedGraphicsTask(req);
     // Print the result.
      System.out.println(gson.toJson(rsp));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.CommonCreateTaskRsp;
import com.huawei.mpc.model.ObsObjInfo;
import com.huawei.mpc.model.animated.AnimatedGraphicsOutputParam;
import com.huawei.mpc.model.animated.CreateAnimatedGraphicsTaskReq;
Gson gson = new Gson();

// Initialize the client.
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com:443");
mpcConfig.setProjectId("Your project ID");
mpcConfig.setAk("*************");
mpcConfig.setSk("******************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

CreateAnimatedGraphicsTaskReq req = new CreateAnimatedGraphicsTaskReq();
```



```
// Set the input file path.
ObsObjInfo input = new ObsObjInfo();
input.setBucket("obs-test01");
input.setLocation("cn-north-4");
input.setObject("volume_auto_test.mp4");
req.setInput(input);

// Set the output file path.
ObsObjInfo output = new ObsObjInfo();
output.setBucket("obs-test01");
output.setLocation("cn-north-4");
output.setObject("/e7404a3fa1b547919d659b0cbfa268c1/animate-sdk-test/");
output.setBucket("my-test.gif");
req.setOutput(output);

// Set output parameters.
AnimatedGraphicsOutputParam outputParam = new AnimatedGraphicsOutputParam();
outputParam.setStart(1_000);
outputParam.setEnd(5_000);
outputParam.setFormat(AnimatedGraphicsOutputParam.FormatEnum.GIF);
outputParam.setFrameRate(15);
outputParam.setWidth(1024);
outputParam.setHeight(768);
req.setOutputParam(outputParam);

// Send a request.
CommonCreateTaskRsp rsp = mpcClient.createAnimatedGraphicsTask(req);
// Print the result.
System.out.println(gson.toJson(rsp));
```

3.5.2 Querying Animated GIF Tasks

Notes

- You can query animated GIF tasks by task ID, task status, time range, page number, or compound query.
- In the query results, if the page number and maximum number of displayed records are not specified, but the number of records is greater than 10, 10 records are displayed by default on each page.

Query by Task ID

```
// A maximum of 10 task IDs are supported.
QueryAnimatedGraphicsTaskReq queryAnimatedGraphicsTaskReq = new QueryAnimatedGraphicsTaskReq();
LinkedList<String> taskIds = new LinkedList<>();
taskIds.add("1234");
queryAnimatedGraphicsTaskReq.setTaskId(taskIds);

// Send a request to MPC.
QueryAnimatedGraphicsTaskRsp rsp =
mpcClient.queryAnimatedGraphicsTask(queryAnimatedGraphicsTaskReq);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Query by Page Number

```
// Set the page number and number of records on each page.
QueryAnimatedGraphicsTaskReq queryAnimatedGraphicsTaskReq = new QueryAnimatedGraphicsTaskReq();
queryAnimatedGraphicsTaskReq .setPage(1);
queryAnimatedGraphicsTaskReq .setSize(10);
// Send a request to MPC.
QueryAnimatedGraphicsTaskRsp rsp =
mpcClient.queryAnimatedGraphicsTask(queryAnimatedGraphicsTaskReq);
// Return a message.
System.out.println(new Gson().toJson(rsp ));
```


Query by Time Range

```
// Set the start time and end time.
QueryAnimatedGraphicsTaskReq queryAnimatedGraphicsTaskReq = new QueryAnimatedGraphicsTaskReq();
queryAnimatedGraphicsTaskReq .setStartTime("20180520131400");
queryAnimatedGraphicsTaskReq .setEndTime("20180520141300");
// Send a request to MPC.
QueryAnimatedGraphicsTaskRsp rsp =
mpcClient.queryAnimatedGraphicsTask(queryAnimatedGraphicsTaskReq);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Query by Task Status

```
// Set the task status.
QueryAnimatedGraphicsTaskReq queryAnimatedGraphicsTaskReq = new QueryAnimatedGraphicsTaskReq();
queryAnimatedGraphicsTaskReq.setStatus(CommonTask.STATUS_SUCCEED);
// Send a request to MPC.
queryAnimatedGraphicsTaskReq rsp =
mpcClient.queryAnimatedGraphicsTask(queryAnimatedGraphicsTaskReq);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Compound Query

```
// Set the following parameters:
QueryAnimatedGraphicsTaskReq queryAnimatedGraphicsTaskReq = new QueryAnimatedGraphicsTaskReq();
queryAnimatedGraphicsTaskReq .setPage(1);
queryAnimatedGraphicsTaskReq .setSize(10);
queryAnimatedGraphicsTaskReq .setStartTime("20180520131400");
queryAnimatedGraphicsTaskReq .setEndTime("20180520141300");
queryAnimatedGraphicsTaskReq.setStatus(CommonTask.STATUS_SUCCEED);
// Send a request to MPC.
queryAnimatedGraphicsTaskReq rsp =
mpcClient.queryAnimatedGraphicsTask(queryAnimatedGraphicsTaskReq);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Full Code

```
package com.huawei.mpc;

import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.CommonTask;
import com.huawei.mpc.model.animated.QueryAnimatedGraphicsTaskReq;
import com.huawei.mpc.model.animated.QueryAnimatedGraphicsTaskRsp;

import java.util.LinkedList;

public class AnimatedQueryTest {
    public void queryAnimatedTask() {
        Gson gson = new Gson();

       // Initialize the client.
        MpcConfig mpcConfig = new MpcConfig();
        mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com");
        mpcConfig.setProjectId("Your project ID");
        mpcConfig.setAk("*************");
        mpcConfig.setSk("******************************");

        MpcClient mpcClient = new MpcClient(mpcConfig);

        // Create a query instance.
        QueryAnimatedGraphicsTaskReq queryAnimatedGraphicsTaskReq = new
QueryAnimatedGraphicsTaskReq();
        LinkedList<String> taskIds = new LinkedList<>();
```



```
        taskIds.add("1234");
        queryAnimatedGraphicsTaskReq.setStartTime("20180520131400");
        queryAnimatedGraphicsTaskReq.setEndTime("20180520141300");
        queryAnimatedGraphicsTaskReq.setPage(1);
        queryAnimatedGraphicsTaskReq.setSize(10);
        queryAnimatedGraphicsTaskReq.setTaskId(taskIds);
        queryAnimatedGraphicsTaskReq.setStatus(CommonTask.STATUS_SUCCEED);

        QueryAnimatedGraphicsTaskRsp queryResponse =
mpcClient.queryAnimatedGraphicsTask(queryAnimatedGraphicsTaskReq);
       // Print the query results.
        System.out.println(gson.toJson(queryResponse));
    }
}
```

3.5.3 Canceling an Animated GIF Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be a task in the task queue. An animated GIF task that has been started or completed cannot be deleted.

Core Code

```
String taskId = "1234";
BaseResponse deleteResponse = mpcClient.deleteAnimatedGraphicsTask(taskId);
System.out.println(gson.toJson(deleteResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;
import com.huawei.mpc.model.CommonTask;
import com.huawei.mpc.model.animated.QueryAnimatedGraphicsTaskReq;
import com.huawei.mpc.model.animated.QueryAnimatedGraphicsTaskRsp;

import java.util.LinkedList;Gson gson = new Gson();

// Initialize the client.
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com");
mpcConfig.setProjectId("Your project ID");
mpcConfig.setAk("**************");
mpcConfig.setSk("******************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

String taskId = "1234";
BaseResponse deleteResponse = mpcClient.deleteAnimatedGraphicsTask(taskId);
System.out.println(gson.toJson(deleteResponse));
```

3.6 Video Parsing

3.6.1 Creating a Video Parsing Task

You can create a video parsing task by creating an MpcClient instance and setting related parameters. The task is used to parse video metadata.

Prerequisites

- You have bought Object Storage Service (OBS) resources and uploaded an input file to an OBS bucket which is in the same region (for example, CN North-Beijing4) as MPC by referring to **Uploading Media Files**.
- MPC has been authorized to access OBS resources. For details, see **Authorizing Access to Cloud Resources**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
       mpcConfig.setEndPoint(endPoint);// Set the endpoint.
       mpcConfig.setProjectId(projectId);// Set the project ID.
       mpcConfig.setSk(sk);// Set the SK.
       mpcConfig.setAk(ak);// Set the AK.
   ```

   Table 3-5 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
       // Create an MpcClient instance.
       MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
         clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
         clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
         clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
         clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
    MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Create a video parsing task.

   For a video parsing task, you need to set input video file parameters. If necessary, you can save the metadata file to a specified path.

```
CreateExtractTaskReq req = new CreateExtractTaskReq();

// Set the input file path.
ObsObjInfo input = new ObsObjInfo();
input.setBucket("obs-test");
input.setLocation("cn-north-4");
input.setObject("test.flv");
req.setInput(input);

// Set the output file path.
ObsObjInfo output = new ObsObjInfo();
output.setBucket("obs-test");
output.setLocation("cn-north-7");
output.setObject("/test-output/");
output.setFileName("my-test.txt");
req.setOutput(output);

// Send a request.
CommonCreateTaskRsp rsp = mpcClient.createExtractTask(req);
// Print the result.
System.out.println(gson.toJson(rsp));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.CommonCreateTaskRsp;
import com.huawei.mpc.model.ObsObjInfo;
import com.huawei.mpc.model.extract.CreateExtractTaskReq;

Gson gson = new Gson();

MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com:443");
mpcConfig.setProjectId("ca225055625d48f6888f60730fb82657");
mpcConfig.setAk("Your project ID");
mpcConfig.setSk("****************************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

CreateExtractTaskReq req = new CreateExtractTaskReq();

// Set the input file path.
ObsObjInfo input = new ObsObjInfo();
input.setBucket("obs-test");
input.setLocation("cn-north-7");
input.setObject("test.flv");
req.setInput(input);

// Set the output file path.
ObsObjInfo output = new ObsObjInfo();
output.setBucket("obs-test");
output.setLocation("cn-north-4");
output.setObject("/test-output/");
output.setFileName("my-test.txt");
req.setOutput(output);

// Send a request.
CommonCreateTaskRsp rsp = mpcClient.createExtractTask(req);
// Print the result.
System.out.println(gson.toJson(rsp));
```


3.6.2 Querying Video Parsing Tasks

Notes

- You can query video parsing tasks by task ID, task status, time range, page number, or compound query.
- In the query results, if the page number and maximum number of displayed records are not specified, but the number of records is greater than 10, 10 records are displayed by default on each page.

Query by Task ID

```
// A maximum of 10 task IDs are supported.
QueryExtractTaskReq queryExtractReq = new QueryExtractTaskReq();
LinkedList<String> taskIds = new LinkedList<>();
taskIds.add("2288");
queryExtractReq.setTaskId(taskIds);

// Send a request to MPC.
QueryExtractTaskRsp queryResponse = mpcClient.queryExtractTask(queryExtractReq);
// Return a message.
System.out.println(new Gson().toJson(queryResponse));
```

Query by Page Number

```
// Set the page number and number of records on each page.
QueryExtractTaskReq queryExtractReq = new QueryExtractTaskReq();
queryExtractReq.setPage(1);
queryExtractReq.setSize(10);
// Send a request to MPC.
QueryExtractTaskRsp queryResponse = mpcClient.queryExtractTask(queryExtractReq);
// Return a message.
System.out.println(new Gson().toJson(queryResponse));
```

Query by Time Range

```
// Set the start time and end time.
QueryExtractTaskReq queryExtractReq = new QueryExtractTaskReq();
queryExtractReq.setStartTime("20180520131400");
queryExtractReq.setEndTime("20180520141300");

// Send a request to MPC.
QueryExtractTaskRsp queryResponse = mpcClient.queryExtractTask(queryExtractReq);
// Return a message.
System.out.println(new Gson().toJson(queryResponse));
```

Query by Task Status

```
// Set the task status.
QueryExtractTaskReq queryExtractReq = new QueryExtractTaskReq();
queryExtractReq.setStatus(CommonTask.STATUS_SUCCEED);
// Send a request to MPC.
QueryExtractTaskRsp queryResponse = mpcClient.queryExtractTask(queryExtractReq);
// Return a message.
System.out.println(new Gson().toJson(queryResponse));
```

Compound Query

```
// Set the following parameters:
QueryExtractTaskReq queryExtractReq = new QueryExtractTaskReq();
queryExtractReq.setStartTime("20180520131400");
queryExtractReq.setEndTime("20180520141300");
queryExtractReq.setPage(1);
```



```
queryExtractReq.setSize(10);
queryExtractReq.setTaskId(taskIds);
queryExtractReq.setStatus(CommonTask.STATUS_SUCCEED);
// Send a request to MPC.
QueryExtractTaskRsp queryResponse = mpcClient.queryExtractTask(queryExtractReq);
// Return a message.
System.out.println(new Gson().toJson(rsp));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.CommonTask;
import com.huawei.mpc.model.extract.QueryExtractTaskReq;
import com.huawei.mpc.model.extract.QueryExtractTaskRsp;

import java.util.LinkedList;

Gson gson = new Gson();

// Initialize the client.
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com:443");
mpcConfig.setProjectId("ca225055625d48f6888f60730fb82657");
mpcConfig.setAk("Your project ID");
mpcConfig.setSk("****************************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

// Query by task ID.
// QueryExtractTaskReq queryExtractReq = new QueryExtractTaskReq();
// LinkedList<String> taskIds = new LinkedList<>();
// taskIds.add("1234");

// Compound query
queryExtractReq.setStartTime("20180520131400");
queryExtractReq.setEndTime("20180520141300");
queryExtractReq.setPage(1);
queryExtractReq.setSize(10);
queryExtractReq.setTaskId(taskIds);
queryExtractReq.setStatus(CommonTask.STATUS_SUCCEED);

QueryExtractTaskRsp queryResponse = mpcClient.queryExtractTask(queryExtractReq);
// Print the query results.
System.out.println(gson.toJson(queryResponse));
```

3.6.3 Canceling a Video Parsing Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be a task in the task queue. A video parsing task that has been started or completed cannot be deleted.

Core Code

```
String taskId = "1234";
BaseResponse deleteResponse = mpcClient.deleteExtractTask(taskId);
System.out.println(gson.toJson(deleteResponse));
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
```



```
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

Gson gson = new Gson();

// Initialize the client.
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com");
mpcConfig.setProjectId("ca225055625d48f6888f60730fb82657");
mpcConfig.setAk("Your project ID");
mpcConfig.setSk("****************************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

String taskId = "1234";
BaseResponse deleteResponse = mpcClient.deleteExtractTask(taskId);
System.out.println(gson.toJson(deleteResponse));
```

3.7 Packaging

3.7.1 Creating a Packaging Task

You can create a video packaging task by creating an MpcClient instance and setting related parameters. The task is used to wrap the compressed video in a packetized format.

Prerequisites

- You have bought Object Storage Service (OBS) resources and uploaded an input file to an OBS bucket which is in the same region (for example, CN North-Beijing4) as MPC by referring to **Uploading Media Files**.
- MPC has been authorized to access OBS resources. For details, see **Authorizing Access to Cloud Resources**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
       mpcConfig.setEndPoint(endPoint);// Set the endpoint.
       mpcConfig.setProjectId(projectId);// Set the project ID.
       mpcConfig.setSk(sk);// Set the SK.
       mpcConfig.setAk(ak);// Set the AK.
   ```

   Table 3-6 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.

Parameter	Type	Description
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
   // Create an MpcClient instance.
   MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
         clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
         clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
         clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
         clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
    MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Create a packaging task.

   ```
   CreateRemuxTaskReq req = new CreateRemuxTaskReq();

   ObsObjInfo input = new ObsObjInfo();
   input.setBucket("obs-gg");
   input.setLocation("cn-north-7");
   input.setObject("problem/ts_14M_1080p.ts");
   req.setInput(input);

   ObsObjInfo output = new ObsObjInfo();
   output.setBucket("obs-gg");
   output.setLocation("cn-north-7");
   output.setObject("output/ts");
   req.setOutput(output);

   RemuxOutputParam outputParam = new RemuxOutputParam();
   // Set the output format.
   outputParam.setFormat(RemuxOutputParam.FormatEnum.MP4);
   // Set the segment duration.
   outputParam.setSegmentDuration(7);
   req.setOutputParam(outputParam);

   CreateRemuxTaskResponse rsp = mpcClient.creatRemuxTask(req);
   System.out.println("craete: " + gson.toJson(rsp) + "\n");
   ```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.CommonCreateTaskRsp;
import com.huawei.mpc.model.ObsObjInfo;
import com.huawei.mpc.model.remux.*;

Gson gson = new Gson();
```



```
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com:443");
mpcConfig.setProjectId("ca225055625d48f6888f60730fb82657");
mpcConfig.setAk("Your project ID");
mpcConfig.setSk("****************************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

CreateRemuxTaskReq req = new CreateRemuxTaskReq();

ObsObjInfo input = new ObsObjInfo();
input.setBucket("obs-gg");
input.setLocation("cn-north-7");
input.setObject("problem/ts_14M_1080p.ts");
req.setInput(input);

ObsObjInfo output = new ObsObjInfo();
output.setBucket("obs-gg");
output.setLocation("cn-north-7");
output.setObject("output/ts");
req.setOutput(output);

RemuxOutputParam outputParam = new RemuxOutputParam();
// Set the output format.
outputParam.setFormat(RemuxOutputParam.FormatEnum.MP4);
// Set the segment duration.
outputParam.setSegmentDuration(7);
req.setOutputParam(outputParam);

CreateRemuxTaskResponse rsp = mpcClient.creatRemuxTask(req);
System.out.println("craete: " + gson.toJson(rsp) + "\n");
```

3.7.2 Querying Packaging Tasks

Notes

- You can query packaging tasks by task ID, task status, time range, page number, or compound query.
- In the query results, if the page number and maximum number of displayed records are not specified, but the number of records is greater than 10, 10 records are displayed by default on each page.

Query by Task ID

```
// Query transcoding tasks.
QueryRemuxTaskRequest queryRemuxTaskRequest = new QueryRemuxTaskRequest();
LinkedList<String> taskIds = new LinkedList<>();
taskIds.add("2288");

queryRemuxTaskRequest.setTaskId(taskIds.toArray(new String[0]));

// Send a request to MPC.
QueryRemuxTaskResponse queryResponse = mpcClient.queryRemuxTask(queryRemuxTaskRequest);
// Return a message.
System.out.println("query: " + gson.toJson(queryResponse)  + "\n");
```

Query by Page Number

```
// Set the page number and number of records on each page.

QueryRemuxTaskRequest queryRemuxTaskRequest = new QueryRemuxTaskRequest();
queryRemuxTaskRequest.setPage(1);
queryRemuxTaskRequest.setSize(10);

// Send a request to MPC.
```



```
QueryRemuxTaskResponse queryResponse = mpcClient.queryRemuxTask(queryRemuxTaskRequest);
// Return a message.
System.out.println("query: " + gson.toJson(queryResponse)  + "\n");
```

Query by Time Range

```
// Set the start time and end time.
QueryRemuxTaskRequest queryRemuxTaskRequest = new QueryRemuxTaskRequest();
queryRemuxTaskRequest.setStartTime("20180520131400");
queryRemuxTaskRequest.setEndTime("20180520141300");

// Send a request to MPC.
QueryRemuxTaskResponse queryResponse = mpcClient.queryRemuxTask(queryRemuxTaskRequest);
// Return a message.
System.out.println("query: " + gson.toJson(queryResponse)  + "\n");
```

Query by Task Status

```
// Set the task status.
QueryRemuxTaskRequest queryRemuxTaskRequest = new QueryRemuxTaskRequest();
queryRemuxTaskRequest.setStatus(CommonTask.STATUS_SUCCEED);
// Send a request to MPC.
QueryRemuxTaskResponse queryResponse = mpcClient.queryRemuxTask(queryRemuxTaskRequest);
// Return a message.
System.out.println("query: " + gson.toJson(queryResponse)  + "\n");
```

Compound Query

```
// Set the following parameters:
QueryRemuxTaskRequest queryRemuxTaskRequest = new QueryRemuxTaskRequest();
queryRemuxTaskRequest.setStartTime("20180520131400");
queryRemuxTaskRequest.setEndTime("20180520141300");
queryRemuxTaskRequest.setPage(1);
queryRemuxTaskRequest.setSize(10);
queryRemuxTaskRequest.setTaskId(taskIds.toArray(new String[0]);
queryRemuxTaskRequest.setStatus(CommonTask.STATUS_SUCCEED);
// Send a request to MPC.
QueryRemuxTaskResponse queryResponse = mpcClient.queryRemuxTask(queryRemuxTaskRequest);
// Return a message.
System.out.println("query: " + gson.toJson(queryResponse)  + "\n");
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.CommonTask;
import com.huawei.mpc.model.remux.*;

import java.util.LinkedList;
import java.util.ArrayList;

Gson gson = new Gson();

// Initialize the client.
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com:443");
mpcConfig.setProjectId("ca225055625d48f6888f60730fb82657");
mpcConfig.setAk("Your project ID");
mpcConfig.setSk("****************************************");

MpcClient mpcClient = new MpcClient(mpcConfig);

// Query by task ID.
// QueryRemuxTaskRequest queryRemuxTaskRequest= new QueryRemuxTaskRequest();
// LinkedList<String> taskIds = new LinkedList<>();
// taskIds.add("1234");
```



```
// Compound query
queryRemuxTaskRequest.setStartTime("20180520131400");
queryRemuxTaskRequest.setEndTime("20180520141300");
queryRemuxTaskRequest.setPage(1);
queryRemuxTaskRequest.setSize(10);
queryRemuxTaskRequest.setTaskId(taskIds.toArray(new String[0])
);
queryRemuxTaskRequest.setStatus(CommonTask.STATUS_SUCCEED);

QueryRemuxTaskResponse queryResponse = mpcClient.queryRemuxTask(queryRemuxTaskRequest);
// Print the query results.
System.out.println("query: " + gson.toJson(queryResponse)  + "\n");
```

3.7.3 Canceling a Packaging Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be a task in the task queue. A packaging task that has been started or completed cannot be deleted.

Core Code

```
String taskId = "1234";
BaseResponse deleteResponse = mpcClient.cancelRemuxTask(taskId);
System.out.println("cancel: " + gson.toJson(deleteResponse)  + "\n");
```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

Gson gson = new Gson();

// Initialize the client.
MpcConfig mpcConfig = new MpcConfig();
mpcConfig.setEndPoint("mpc.cn-north-4.myhuaweicloud.com");
mpcConfig.setProjectId("ca225055625d48f6888f60730fb82657");
mpcConfig.setAk("Your project ID");
mpcConfig.setSk("****************************************");

MpcClient mpcClient = new MpcClient(mpcConfig);
String taskId = "1234";
BaseResponse deleteResponse = mpcClient.cancelRemuxTask(taskId);
System.out.println("cancel: " + gson.toJson(deleteResponse)  + "\n");
```

3.8 Transcoding Templates

3.8.1 Creating a Transcoding Template

You can use the SDK to create a transcoding template. For details about template parameters, see the API for **creating a transcoding template**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.


```
MpcConfig mpcConfig = new MpcConfig();
    mpcConfig.setEndPoint(endPoint);// Set the endpoint.
    mpcConfig.setProjectId(projectId);// Set the project ID.
    mpcConfig.setSk(sk);// Set the SK.
    mpcConfig.setAk(ak);// Set the AK.
```

Table 3-7 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
   // Create an MpcClient instance.
   MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
       clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
       clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
       clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
       clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
    MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Send a request.

   ```
   // Set request parameters.
   TransTemplate transTemplate=new TransTemplate();
   // Set the transcoding template name.
   transTemplate.setTemplateName("MP4_12222ddd2122");
   ```

4. Set template parameters.

   a. Video parameters

   ```
   TransTemplate.VideoBean videoBean=new TransTemplate.VideoBean();

   // Set the video codec. 1 is H.264 and 2 is H.265.
   videoBean.setCodec(1);


   // Set the average output bitrate (unit: kbit/s). The value is 0 or an integer ranging from 40 to 30,000. If this is 0, the average output bit rate is an adaptive value.
   videoBean.setBitrate(6000);

   /*
   * Encoding profile
   * H264_BASE = 1,
   ```



```
* H264_MAIN = 2,
* H264_HIGH = 3,
* H265_MAIN = 4,
*/
// Set the encoding profile. The recommended value is 3.
videoBean.setProfile(3);

// Set the encoding level. The value ranges from 1 to 15. The default value is 15.
videoBean.setLevel(15);

/*
* Coding quality level (RMS range)
* HSPEED2 = 1,(only for h.265, h.265 default)
* HSPEED = 2,(only for h.265)
* NORMAL = 3,(h264 / h.265 available, h.264 default)
*/
// Set the encoding quality.
videoBean.setPreset(3);

// Set the maximum number of reference frames (unit: frame). For H.264, the value ranges from 1 to 8. The default value is 4. For H.265, the value is fixed at 4.
videoBean.setRefFramesCount(4);


// Set the maximum I-frame interval (unit: second). The value ranges from 2 to 5. The default value is 5.
videoBean.setMaxIframesInterval(5);

// Set the maximum B-frame interval (unit: frame). The value ranges from 0 to 8 for H.264 and is 4 by default. The value is fixed at 7 for H.265.
videoBean.setbramesCount(4);

// Set the frame rate (unit: FPS). The value is 0 or an integer ranging from 5 to 30. If the frame rate is not in this range, the frame rate is automatically changed to 0. If you configure a frame rate higher than the frame rate of your input file, the frame rate is automatically changed to the frame rate of the input file.
videoBean.setFrameRate(0);

/*
* Set the video width (unit: pixel).
* H.264: The value is 0 or a multiple of 2 from 32 to 4,096.
* H.265: The value must be a multiple of 2 from 160 to 4,096.
*/
videoBean.setWidth(1920);

/*
 * Set the video height (unit: pixel).
 * H.264: The value is 0 or a multiple of 2 from 32 to 2,880.
 * H.265: The value is 0 or a multiple of 2 from 96 to 2,880.
 * If this parameter is set to 0, the video height is an adaptive value.
*/
videoBean.setHeight(1080);

/*
 * Whether to enable black bar removal
 * 0: Disable black bar removal.
 * 1: Enable black bar removal and low-complexity algorithms for long videos (>5 minutes).
 * 2: Enable black bar removal and high-complexity algorithms for short videos (≤5 minutes).
 */
// Set whether to enable black bar removal. The default value is 0.
videoBean.setBlackCut(0);

// Set video parameters.
transTemplate.setVideo(videoBean);
```

b. Audio parameters

```
// Set audio parameters.
TransTemplate.AudioBean audioBean=new TransTemplate.AudioBean();

/*
```



```
 * Audio bitrate, in kbit/s.
 * The value is 0 or ranges from 8 to 1,000.
*/
audioBean.setBitrate(128);

/*
 * Number of audio channels
 * AUDIO_CHANNELS_1=1,
 * AUDIO_CHANNELS_2=2,
 */
audioBean.setChannels(2);

/*
 * Audio codec
 * AAC : 1 (default)
 * HEAAC1 : 2
 * HEAAC2 : 3
 * MP3 : 4
*/
audioBean.setCodec(1);

/*
 * Audio sampling rate
 * AUDIO_SAMPLE_AUTO=1 (default)
 * AUDIO_SAMPLE_22050=2,
 * AUDIO_SAMPLE_32000=3,
 * AUDIO_SAMPLE_44100=4,
 * AUDIO_SAMPLE_48000=5,
 * AUDIO_SAMPLE_96000=6,
*/
audioBean.setSampleRate(4);

// Set audio parameters.
transTemplate.setAudio(audioBean);
```

c. Common parameters

```
// Set common parameters.
TransTemplate.CommonBean commonBean=new TransTemplate.CommonBean();

/*
 * DASH interval, in seconds
 * The value ranges from 2 to 10. The default value is 5.
 */
commonBean.setDashInterval(5);

/*
 * HLS segment interval, in seconds
 * The value ranges from 2 to 10. The default value is 5.
 */
commonBean.setHlsInterval(5);

/*
 * Packaging type.
* Available options:
 * 1: HLS
 * 2: DASH
 * 3: HLS+DASH
 * 4: MP4
 * 5: MP3
 * 6: ADTS
 * If pack_type is set to 5 or 6, do not set video parameters.
 */
commonBean.setPackType(1);

/*
* Whether to enable low bitrate HD
 * false: disabled (default)
 * true: enabled
 */
commonBean.setPvc(false);
```



```
// Set common parameters.
transTemplate.setCommon(commonBean);
```

5. Send a request for creating a transcoding template and return a message.

```
// Send a request.
CreateTemplateResponse createTemplateResponse= mpcClient.createTemplate(transTemplate);
// Return a message.
System.out.println(new Gson().toJson(createTemplateResponse));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and authorization.
* 3. Set request parameters for creating a transcoding template, including the video and audio parameters.
* 4: Send a request.
* 5. Return the result.
*/

import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.template.CreateTemplateResponse;
import com.huawei.mpc.model.template.TransTemplate;

// Set the method for constructing MPC configuration items.
MpcConfig mpcconfig = new MpcConfig();
// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// MPC construction method
MpcClient mpcClient = new MpcClient(mpcconfig);

// Set request parameters.
TransTemplate transTemplate=new TransTemplate();
/*
* Transcoding template name(Length <= 256)
*/
// Set the transcoding template name.
transTemplate.setTemplateName("MP4_12222ddd2122");

/*
* Transcoding template id(String type)
```



```
*/
// Set the transcoding template ID.
transTemplate.setTenantId("123456");

/*VideoBean*/
// Set video parameters.
TransTemplate.VideoBean videoBean=new TransTemplate.VideoBean();

/*
* Video encoding format (RMS range) [1,2] The default 1
* H264 = 1,H265 = 2
*/
// Set the video codec. 1 is H.264 and 2 is H.265.
videoBean.setCodec(1);

/*
* Video Bit rate is within [40,30000]
*/
// Set the average output bitrate (unit: kbit/s).
videoBean.setBitrate(6000);

/*
* Coding grade (RMS range)
* H264_BASE = 1,
* H264_MAIN = 2,
* H264_HIGH = 3,
* H265_MAIN = 4,
*/
// Set the encoding profile.
videoBean.setProfile(3);

/*
* The encoding level (Valid Range) [1,15], default 15
*/
// Set the encoding level.
videoBean.setLevel(15);

/*
* Coding quality level (RMS range)
* HSPEED2 = 1,(only for h.265, h.265 default)
* HSPEED = 2,(only for h.265)
* NORMAL = 3,(h264 / h.265 available, h.264 default)
*/
// Set the encoding quality.
videoBean.setPreset(3);

/*
* Maximum reference frame number (unit: frame)
* H264: Range [1,8], default 4
* H265: fixed value 4
*/
// Set the maximum number of reference frames (unit: frame).
videoBean.setRefFramesCount(4);

/*
* I frame maximum interval (unit: second)
* Range [2,5], default: 5
*/
// Set the maximum I-frame interval (unit: second).
videoBean.setMaxIframesInterval(5);

/*
* Maximum B frame interval (unit: frame)
* H264: Range [0,8], default 4
* H265: fixed value 7
*/
// Set the maximum B-frame interval (unit: frame).
videoBean.setbramesCount(4);
```



```
/*
* Frame rate (unit: frame per second) (Valid range)
* FRAMERATE_AUTO = 1,
* FRAMERATE_10 = 2,
* FRAMERATE_15 = 3,
* FRAMERATE_2397 = 4, // 23.97 fps
* FRAMERATE_24 = 5,
* FRAMERATE_25 = 6,
* FRAMERATE_2997 = 7, // 29.97 fps
* FRAMERATE_30 = 8,
* FRAMERATE_50 = 9,
* FRAMERATE_60 = 10
*/
// Set the frame rate (unit: FPS).
videoBean.setFrameRate(1);

/*
* Video width (in pixels)
* H264: Range [32,4096], must be a multiple of 2
* H265: Range [32,2880], must be a multiple of 4
*/
// Set the video width (unit: pixel).
videoBean.setWidth(1920);

/*
* Video height (in pixels)
* H264: Range [32,2880], must be a multiple of 2
* H265: Range [96,2880], must be a multiple of 4
*/
// Set the video height (unit: pixel).
videoBean.setHeight(1080);

/*
 * Video blackCut
 * 0: off (the current default off),1 or 2: open
 */
// Set whether to enable black bar removal.
videoBean.setBlackCut(0);

/*AudioBean*/
// Set audio parameters.
TransTemplate.AudioBean audioBean=new TransTemplate.AudioBean();

/*
* Audio Bit rate is within [64,320]
*/
// Set the audio bitrate (unit: kbit/s).
audioBean.setBitrate(128);

/*
* The number of channels (RMS range) is 1 or 2
*/
// Set the number of audio channels.
audioBean.setChannels(2);

/*
* Audio encoding format (RMS range) defaults to 1 Optional 2 or 3
*/
// Set the audio codec.
audioBean.setCodec(1);

/*
* Audio sampling rate (RMS range) Default is 1 Optional 2,3,4,5,6
*/
// Set the audio sampling rate.
audioBean.setSampleRate(4);

/*CommonBean*/
// Set common parameters.
```



```
TransTemplate.CommonBean commonBean=new TransTemplate.CommonBean();

/*
* DASH interval (in seconds)
* Range: [2,10], default 5
*/
// Set the DASH interval (unit: second).
commonBean.setDashInterval(5);

/*
* HLS shading interval (in seconds)
* Range: [2,10], default 5
*/
// Set the HLS segment interval (unit: second).
commonBean.setHlsInterval(5);

/*
* Package Type (DASH + MP4, HLS + TS, MP4)
* Bitmap to describe the package format, each bit represents a package format,currently supports two:
* 1 means DASH + MP4 format
* 2 indicates HLS + TS format
* 3 means HLS + TS and DASH + MP4 format
* 4 means MP4 format (note that here refers to MP4 is a video and audio Segment MP4 large files
*/
// Set the packaging type.
commonBean.setPackType(1);

/*
* PVC switch
* 0: off (the current default off),1: open
*/
// Whether to enable low bitrate HD.
commonBean.setPvc(false);



// Set video parameters.
transTemplate.setVideo(videoBean);
// Set audio parameters.
transTemplate.setAudio(audioBean);
// Set common parameters.
transTemplate.setCommon(commonBean);

// Send a request.
CreateTemplateResponse createTemplateResponse= mpcClient.createTemplate(transTemplate);
// Return a message.
System.out.println(new Gson().toJson(createTemplateResponse));
```

3.8.2 Deleting a Transcoding Template

You can delete a custom transcoding template based on its ID.

Core Code

```
// Set the template ID and send a request for deleting the transcoding template.
BaseResponse baseResponse = mpcClient.deleteTemplate(181);
// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and
authorization.
* 3. Set request parameters for deleting a transcoding template.
* 4: Send a request.
```



```
* 5. Return the result.
*/

import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

// Set the method for constructing MPC configuration items.
MpcConfig mpcconfig = new MpcConfig();
// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// MPC construction method
MpcClient mpcClient = new MpcClient(mpcconfig);
/*
* proxy provided
// MPC construction method. If a proxy needs to be configured, use this construction method.
* MpcClient mpcClient=new MpcClient(mpcconfig, clientConfig);
*/

// Set the template ID and send a request for deleting the transcoding template.
BaseResponse baseResponse = mpcClient.deleteTemplate(181);
// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

3.8.3 Modifying a Transcoding Template

You can reset template parameters to modify a transcoding template.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
       mpcConfig.setEndPoint(endPoint);// Set the endpoint.
       mpcConfig.setProjectId(projectId);// Set the project ID.
       mpcConfig.setSk(sk);// Set the SK.
       mpcConfig.setAk(ak);// Set the AK.
   ```

Table 3-8 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. The endpoint of MPC is mts.*cn-north-4*.myhuaweicloud.com. Replace *cn-north-4* with the value of region in the address box of the browser after you select MPC on the management console.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
   // Create an MpcClient instance.
   MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
         clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
         clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
         clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
         clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
    MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Send a request.

   ```
   // Set request parameters.
   TransTemplate transTemplate=new TransTemplate();
   // Set the template name.
   transTemplate.setTemplateName("MP4_12222ddd2122");
   // Set the template ID.
   transTemplate.setTenantId("123456");
   ```

4. // Set template parameters.

   a. Video parameters

      ```
      TransTemplate.VideoBean videoBean=new TransTemplate.VideoBean();

      // Set the video codec. 1 is H.264 and 2 is H.265.
      videoBean.setCodec(1);


      // Set the average output bitrate (unit: kbit/s). The value is 0 or an integer ranging from 40 to 30,000. If this is 0, the average output bit rate is an adaptive value.
      videoBean.setBitrate(6000);

      /*
      * Encoding profile
      * H264_BASE = 1,
      * H264_MAIN = 2,
      * H264_HIGH = 3,
      * H265_MAIN = 4,
      ```



```
*/
// Set the encoding profile. The recommended value is 3.
videoBean.setProfile(3);

// Set the encoding level. The value ranges from 1 to 15. The default value is 15.
videoBean.setLevel(15);

/*
* Coding quality level (RMS range)
* HSPEED2 = 1,(only for h.265, h.265 default)
* HSPEED = 2,(only for h.265)
* NORMAL = 3,(h264 / h.265 available, h.264 default)
*/
// Set the encoding quality.
videoBean.setPreset(3);

// Set the maximum number of reference frames (unit: frame). For H.264, the value ranges from 1 to 8. The default value is 4. For H.265, the value is fixed at 4.
videoBean.setRefFramesCount(4);


// Set the maximum I-frame interval (unit: second). The value ranges from 2 to 5. The default value is 5.
videoBean.setMaxIframesInterval(5);

// Set the maximum B-frame interval (unit: frame). The value ranges from 0 to 8 for H.264 and is 4 by default. The value is fixed at 7 for H.265.
videoBean.setbramesCount(4);

// Set the frame rate (unit: FPS). The value is 0 or an integer ranging from 5 to 30. If the frame rate is not in this range, the frame rate is automatically changed to 0. If you configure a frame rate higher than the frame rate of your input file, the frame rate is automatically changed to the frame rate of the input file.
videoBean.setFrameRate(0);

/*
* Set the video width (unit: pixel).
* H.264: The value is 0 or a multiple of 2 from 32 to 4,096.
* H.265: The value must be a multiple of 2 from 160 to 4,096.
*/
videoBean.setWidth(1920);

/*
 * Set the video height (unit: pixel).
 * H.264: The value is 0 or a multiple of 2 from 32 to 2,880.
 * H.265: The value is 0 or a multiple of 2 from 96 to 2,880.
 * If this parameter is set to 0, the video height is an adaptive value.
*/
videoBean.setHeight(1080);

/*
 * Whether to enable black bar removal
 * 0: Disable black bar removal.
 * 1: Enable black bar removal and low-complexity algorithms for long videos (>5 minutes).
 * 2: Enable black bar removal and high-complexity algorithms for short videos (≤5 minutes).
 */
// Set whether to enable black bar removal. The default value is 0.
videoBean.setBlackCut(0);

// Set video parameters.
transTemplate.setVideo(videoBean);
```

b. Audio parameters

```
// Set audio parameters.
TransTemplate.AudioBean audioBean=new TransTemplate.AudioBean();

/*
 * Audio bitrate, in kbit/s.
 * The value is 0 or ranges from 8 to 1,000.
*/
```



```
audioBean.setBitrate(128);

/*
 * Number of audio channels
 * AUDIO_CHANNELS_1=1,
 * AUDIO_CHANNELS_2=2,
 */
audioBean.setChannels(2);

/*
 * Audio codec
 * AAC : 1 (default)
 * HEAAC1 : 2
 * HEAAC2 : 3
 * MP3 : 4
*/
audioBean.setCodec(1);

/*
 * Audio sampling rate
 * AUDIO_SAMPLE_AUTO=1 (default)
 * AUDIO_SAMPLE_22050=2,
 * AUDIO_SAMPLE_32000=3,
 * AUDIO_SAMPLE_44100=4,
 * AUDIO_SAMPLE_48000=5,
 * AUDIO_SAMPLE_96000=6,
*/
audioBean.setSampleRate(4);

// Set audio parameters.
transTemplate.setAudio(audioBean);
```

c. Common parameters

```
// Set common parameters.
TransTemplate.CommonBean commonBean=new TransTemplate.CommonBean();

/*
 * DASH interval, in seconds
 * The value ranges from 2 to 10. The default value is 5.
 */
commonBean.setDashInterval(5);

/*
 * HLS segment interval, in seconds
 * The value ranges from 2 to 10. The default value is 5.
 */
commonBean.setHlsInterval(5);

/*
 * Packaging type.
* Available options:
 * 1: HLS
 * 2: DASH
 * 3: HLS+DASH
 * 4: MP4
 * 5: MP3
 * 6: ADTS
 * If pack_type is set to 5 or 6, do not set video parameters.
 */
commonBean.setPackType(1);

/*
* Whether to enable low bitrate HD
 * false: disabled (default)
 * true: enabled
 */
commonBean.setPvc(false);
```



```
// Set common parameters.
transTemplate.setCommon(commonBean);
```

5. Send a request for modifying a transcoding template and return a message.

```
// Send a request.
BaseResponse baseResponse = mpcClient.modifyTemplate(transTemplate);
// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

Sample Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and
authorization.
* 3. Set request parameters for modifying a transcoding template, including the video and audio parameters.
* 4. Send a request.
* 5. Return the result.
*/
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;
import com.huawei.mpc.model.template.TransTemplate;


// Set the method for constructing MPC configuration items.
MpcConfig mpcconfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/


// MPC construction method
MpcClient mpcClient = new MpcClient(mpcconfig);
// Set template parameters.
TransTemplate transTemplate=new TransTemplate();

/*
* Transcoding template name(Length <= 256)
*/
// Set the transcoding template name.
transTemplate.setTemplateName("MP4_12222ddd2122");

/*
* Transcoding template id(Integer type)
```



```
*/
// Set the transcoding template ID.
transTemplate.setTemplateId(551);

/*VideoBean*/
// Set video parameters.
TransTemplate.VideoBean videoBean=new TransTemplate.VideoBean();

/*
* Video encoding format (RMS range) [1,2] The default 1
* H264 = 1,H265 = 2
*/
// Set the video codec.
videoBean.setCodec(1);

/*
* Video Bit rate is within [40,30000]
*/
// Set the average output bitrate (unit: kbit/s).
videoBean.setBitrate(6000);

/*
* Coding grade (RMS range)
* H264_BASE = 1,
* H264_MAIN = 2,
* H264_HIGH = 3,
* H265_MAIN = 4,
*/
// Set the encoding profile.
videoBean.setProfile(3);

/*
* The encoding level (Valid Range) [1,15], default 15
*/
// Set the encoding level.
videoBean.setLevel(15);

/*
* Coding quality level (RMS range)
* HSPEED2 = 1,(only for h.265, h.265 default)
* HSPEED = 2,(only for h.265)
* NORMAL = 3,(h264 / h.265 available, h.264 default)
*/
// Set the encoding quality.
videoBean.setPreset(3);

/*
* Maximum reference frame number (unit: frame)
* H264: Range [1,8], default 4
* H265: fixed value 4
*/
// Set the maximum number of reference frames (unit: frame).
videoBean.setRefFramesCount(4);

/*
* I frame maximum interval (unit: second)
* Range [2,5], default: 5
*/
// Set the maximum I-frame interval (unit: second).
videoBean.setMaxIFramesInterval(5);

/*
* Maximum B frame interval (unit: frame)
* H264: Range [0,8], default 4
* H265: fixed value 7
*/
// Set the maximum B-frame interval (unit: frame).
videoBean.setbFramesCount(4);
```



```
/*
* Frame rate (unit: frame per second) (Valid range)
* FRAMERATE_AUTO = 1,
* FRAMERATE_10 = 2,
* FRAMERATE_15 = 3,
* FRAMERATE_2397 = 4, // 23.97 fps
* FRAMERATE_24 = 5,
* FRAMERATE_25 = 6,
* FRAMERATE_2997 = 7, // 29.97 fps
* FRAMERATE_30 = 8,
* FRAMERATE_50 = 9,
* FRAMERATE_60 = 10
*/
// Set the frame rate (unit: FPS).
videoBean.setFrameRate(1);

/*
* Video width (in pixels)
* H264: Range [32,4096], must be a multiple of 2
* H265: Range [160,4096], must be a multiple of 4
*/
// Set the video width (unit: pixel).
videoBean.setWidth(1920);

/*
* Video height (in pixels)
* H264: Range [32,2880], must be a multiple of 2
* H265: Range [96,2880], must be a multiple of 4
*/
// Set the video height (unit: pixel).
videoBean.setHeight(1080);

/*AudioBean*/
// Set audio parameters.
TransTemplate.AudioBean audioBean=new TransTemplate.AudioBean();

/*
* Audio Bit rate is within [64,320]
*/
// Set the audio bitrate (unit: kbit/s).
audioBean.setBitrate(128);

/*
* The number of channels (RMS range) is 1 or 2
*/
// Set the number of audio channels.
audioBean.setChannels(2);

/*
* Audio encoding format (RMS range) defaults to 1 Optional 2 or 3
*/
// Set the audio codec.
audioBean.setCodec(1);

/*
* Audio sampling rate (RMS range) Default is 1 Optional 2,3,4,5,6
*/
// Set the audio sampling rate.
audioBean.setSampleRate(4);

/*CommonBean*/
// Set common parameters.
TransTemplate.CommonBean commonBean=new TransTemplate.CommonBean();

/*
* DASH interval (in seconds)
* Range: [2,10], default 5
*/
// Set the DASH interval (unit: second).
```



```
commonBean.setDashInterval(5);

/*
* HLS shading interval (in seconds)
* Range: [2,10], default 5
*/
// Set the HLS segment interval (unit: second).
commonBean.setHlsInterval(5);

/*
* Package Type (DASH + MP4, HLS + TS, MP4)
* Bitmap to describe the package format, each bit represents a package format,currently supports two:
* 1 means DASH + MP4 format
* 2 indicates HLS + TS format
* 3 means HLS + TS and DASH + MP4 format
* 4 means MP4 format (note that here refers to MP4 is a video and audio Segment MP4 large files
*/
// Set the packaging type.
commonBean.setPackType(1);

/*
* PVC switch
* false: off (the current default off),true: open
*/
// Set whether to enable low bitrate HD.
commonBean.setPvc(false);

/*
* QDS switch
* false: off (the current default off),true: open
*/
// QDS switch
commonBean.setQds(false);

// Set video parameters.
transTemplate.setVideo(videoBean);
// Set audio parameters.
transTemplate.setAudio(audioBean);
// Set common parameters.
transTemplate.setCommon(commonBean);

// Send a request.
BaseResponse baseResponse = mpcClient.modifyTemplate(transTemplate);
// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

3.8.4 Querying Transcoding Templates

You can query custom transcoding templates and system presets by template ID or page number. For details, see the API for **querying transcoding templates**.

Notes

- You can query one or more custom transcoding templates by template ID. A maximum of 10 transcoding templates can be queried at a time.
- You can query templates based on the page number and number of records on each page.

Core Code

```
// Set template parameters.
QueryTemplateReq queryTemplateReq = new QueryTemplateReq();
// Set the template ID. A maximum of 10 transcoding templates can be queried.
queryTemplateReq.setTemplateIds(new Integer[]{215, 156});

// Send a request.
```



```
QueryTemplateResponse queryTemplateResponse = mpcClient.queryTemplate(queryTemplateReq);
// Return the query results.
System.out.println(new Gson().toJson(queryTemplateResponse));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and
authorization.
* 3. Set request parameters for querying a transcoding template.
* 4: Send a request.
* 5. Return the results.
*/
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.template.QueryTemplateReq;
import com.huawei.mpc.model.template.QueryTemplateResponse;


// Set the method for constructing MPC configuration items.
MpcConfig mpcconfig = new MpcConfig();
// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

// MPC construction method
MpcClient mpcClient = new MpcClient(mpcconfig);
/*
* proxy provided
// MPC construction method. If a proxy needs to be configured, use this construction method.
* MpcClient mpcClient=new MpcClient(mpcconfig, clientConfig);
*/

// Set template parameters.
QueryTemplateReq queryTemplateReq = new QueryTemplateReq();
// Set the template ID. A maximum of 10 transcoding templates can be queried.
queryTemplateReq.setTemplateIds(new Integer[]{215, 156});

// Send a request.
QueryTemplateResponse queryTemplateResponse = mpcClient.queryTemplate(queryTemplateReq);
// Return the query results.
System.out.println(new Gson().toJson(queryTemplateResponse));
```


3.9 Watermarking

3.9.1 Creating a Watermark Template

A watermark template is used to add watermarks to transcoded videos. For details, see **Creating a Watermark Template**.

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

   ```
   MpcConfig mpcConfig = new MpcConfig();
       mpcConfig.setEndPoint(endPoint);// Set the endpoint.
       mpcConfig.setProjectId(projectId);// Set the project ID.
       mpcConfig.setSk(sk);// Set the SK.
       mpcConfig.setAk(ak);// Set the AK.
   ```

   Table 3-9 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

   ```
       // Create an MpcClient instance.
       MpcClient mpcClient = new MpcClient(mpcConfig);
   ```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

   ```
   // Configure the proxy.
   ClientConfig clientConfig = new ClientConfig();
         clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
         clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
         clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
         clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
   // Use the constructor to initialize MpcClient.
    MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
   ```

3. Send a request.

```
        CreateWatermarkTemplateRequest createWatermarkTemplateRequest = new
CreateWatermarkTemplateRequest();
```

4. Set template parameters.

```
// Set the watermark template name.
        createWatermarkTemplateRequest.setTemplateName("template_name");

        // Set the watermark type.
        createWatermarkTemplateRequest.setType("Image");

        // Set how the image watermark is processed.
        createWatermarkTemplateRequest.setImageProcess("Original");

       // Set the watermark image width.
        createWatermarkTemplateRequest.setWidth("20");

       // Set the watermark image height.
        createWatermarkTemplateRequest.setHeight("30");

        // Set the horizontal offset of the watermark image relative to the output video. The default
value is 0.
        createWatermarkTemplateRequest.setDx("20");

        // Set the vertical offset of the watermark image relative to the output video.
        createWatermarkTemplateRequest.setDy("30");

        // Set the watermark position. The options are TopRight, TopLeft, BottomRight, and BottomLeft.
        createWatermarkTemplateRequest.setReferpos("TopRight");

        // Set the watermark start time.
        createWatermarkTemplateRequest.setTimelineStart("10");

        // Set the watermark duration. The default value is ToEND.
        createWatermarkTemplateRequest.setTimelineDuration("ToEND");
```

5. Send a request for creating a watermark template and return a message.

```
        // Send a request to MPC.
        CreateWatermarkTemplateRespons createWatermarkTemplateRespons =
mpcClient.createWatermarkTemplate(createWatermarkTemplateRequest);

        // Return a message.
        System.out.println(new Gson().toJson(createWatermarkTemplateRespons));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and
authorization.
* 3. Set request parameters for creating a watermark template, including the watermark template name
and watermark type.
* 4. Send a request.
* 5. Return the result.
*/
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.watermark.CreateWatermarkTemplateRequest;
import com.huawei.mpc.model.watermark.CreateWatermarkTemplateRespons;

// Set the method for constructing MPC configuration items.
MpcConfig mpcConfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);
```



```
// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/

        // MPC construction method
        MpcClient mpcClient = new MpcClient(mpcConfig);
       // If a proxy needs to be set, use this constructor to initialize MpcClient.
    // MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);

        // Set request parameters.
        CreateWatermarkTemplateRequest createWatermarkTemplateRequest = new
CreateWatermarkTemplateRequest();

        // Set the watermark template name.
        createWatermarkTemplateRequest.setTemplateName("template_name");

       // Set the watermark type.
        createWatermarkTemplateRequest.setType("Image");

       // Set how the image watermark is processed.
        createWatermarkTemplateRequest.setImageProcess("Original");

       // Set the watermark image width.
        createWatermarkTemplateRequest.setWidth("20");

       // Set the watermark image height.
        createWatermarkTemplateRequest.setHeight("30");

        // Set the horizontal offset of the watermark image relative to the output video. The default value is 0.
        createWatermarkTemplateRequest.setDx("20");

       // Set the vertical offset of the watermark image relative to the output video.
        createWatermarkTemplateRequest.setDy("30");

       // Set the watermark position. The options are TopRight, TopLeft, BottomRight, and BottomLeft.
        createWatermarkTemplateRequest.setReferpos("TopRight");

       // Set the watermark start time.
        createWatermarkTemplateRequest.setTimelineStart("10");

       // Set the watermark duration. The default value is ToEND.
        createWatermarkTemplateRequest.setTimelineDuration("ToEND");

        // Send a request to MPC.
        CreateWatermarkTemplateRespons createWatermarkTemplateRespons =
mpcClient.createWatermarkTemplate(createWatermarkTemplateRequest);

        // Return a message.
        System.out.println(new Gson().toJson(createWatermarkTemplateRespons));
```


3.9.2 Modifying a Watermark Template

Core Code

1. Create MPC configuration items.

   These configuration items are used for MPC to obtain authorization. **Table 1** describes the parameters.

```
MpcConfig mpcConfig = new MpcConfig();
    mpcConfig.setEndPoint(endPoint);// Set the endpoint.
    mpcConfig.setProjectId(projectId);// Set the project ID.
    mpcConfig.setSk(sk);// Set the SK.
    mpcConfig.setAk(ak);// Set the AK.
```

   Table 3-10 MPC parameters

Parameter	Type	Description
endPoint	String	Endpoint. The endpoint of MPC is mts.*cn-north-4*.myhuaweicloud.com. Replace *cn-north-4* with the value of region in the address box of the browser after you select MPC on the management console.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.

2. Create an MpcClient instance.

   If no proxy server is configured, you can directly create an MpcClient instance.

```
    // Create an MpcClient instance.
    MpcClient mpcClient = new MpcClient(mpcConfig);
```

   If a proxy server needs to be configured, set proxy parameters and then transfer the proxy as a constructor to the MpcClient instance.

```
// Configure the proxy.
ClientConfig clientConfig = new ClientConfig();
      clientConfig.setProxyHost(proxyHost);// Set the IP address of the proxy server.
      clientConfig.setProxyPort(Integer.parseInt(proxyPort));// Set the port number of the proxy server.
      clientConfig.setProxyUserName(proxyUserName);// Set the username for accessing the proxy server.
      clientConfig.setProxyPassword(proxyPassword);// Set the password for accessing the proxy server.
// Use the constructor to initialize MpcClient.
 MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);
```

3. Send a request.

```
      // Set request parameters.
      UpdateWatermarkTemplateRequest updateWatermarkTemplateRequest = new UpdateWatermarkTemplateRequest();
```

4. Set template parameters.

```
// Set the watermark template ID.
      updateWatermarkTemplateRequest.setTemplateId("template_id");
```

```
        // Set the watermark template name.
        updateWatermarkTemplateRequest.setTemplateName("template_name");

       // Set the watermark type.
        updateWatermarkTemplateRequest.setType("Image");

       // Set how the image watermark is processed.
        updateWatermarkTemplateRequest.setImageProcess("Original");

       // Set the watermark image width.
        updateWatermarkTemplateRequest.setWidth("20");

       // Set the watermark image height.
        updateWatermarkTemplateRequest.setHeight("30");

       // Set the horizontal offset of the watermark image relative to the output video. The default
value is 0.
        updateWatermarkTemplateRequest.setDx("20");

       // Set the vertical offset of the watermark image relative to the output video.
        updateWatermarkTemplateRequest.setDy("30");

       // Set the watermark position. The options are TopRight, TopLeft, BottomRight, and BottomLeft.
        updateWatermarkTemplateRequest.setReferpos("TopRight");

       // Set the watermark start time.
        updateWatermarkTemplateRequest.setTimelineStart("10");

       // Set the watermark duration. The default value is ToEND.
        updateWatermarkTemplateRequest.setTimelineDuration("ToEND");
```

5. Send a request for modifying a watermark template and return a message.

```
        // Send a request to MPC.
        BaseResponse baseResponse =
mpcClient.updateWatermarkTemplate(updateWatermarkTemplateRequest);

       // Return a message.
        System.out.println(new Gson().toJson(baseResponse));
```

Full Code

```
/*
* Service process:
* 1. Import the SDK package MPCSDK.jar.
* 2. Set configuration items, including the endpoint, AK, SK, and project ID, for accessing MPC and
authorization.
* 3. Set request parameters for modifying a watermark template, including the template ID, template name,
and watermark type.
* 4. Send a request.
* 5. Return the result.
*/
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;
import com.huawei.mpc.model.watermark.UpdateWatermarkTemplateRequest;
      // Set the method for constructing MPC configuration items.
       MpcConfig mpcConfig = new MpcConfig();

     // Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
       mpcConfig.setEndPoint(endPoint);

      // Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
       mpcConfig.setProjectId(projectId);

      // Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
       mpcConfig.setSk(sk);

      // Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
```



```
        mpcConfig.setAk(ak);

       /*if you need proxy*/
      // Optional. Set the proxy if necessary.
       /*
          ClientConfig clientConfig = new ClientConfig();
         // Set the IP address of the proxy server.
           clientConfig.setProxyHost(proxyHost);
         // Set the port number of the proxy server.
            clientConfig.setProxyPort(Integer.parseInt(proxyPort));
          // Set the username for accessing the proxy server.
           clientConfig.setProxyUserName(proxyUserName);
         // Set the password for accessing the proxy server.
          clientConfig.setProxyPassword(proxyPassword);
         */

       // MPC construction method
        MpcClient mpcClient = new MpcClient(mpcConfig);
       // If a proxy needs to be set, use this constructor to initialize MpcClient.
    // MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);

       // Set request parameters.
        UpdateWatermarkTemplateRequest updateWatermarkTemplateRequest = new
UpdateWatermarkTemplateRequest();

        // Set the watermark template ID.
        updateWatermarkTemplateRequest.setTemplateId("template_id");

       // Set the watermark template name.
        updateWatermarkTemplateRequest.setTemplateName("template_name");

       // Set the watermark type.
        updateWatermarkTemplateRequest.setType("Image");

       // Set how the image watermark is processed.
        updateWatermarkTemplateRequest.setImageProcess("Original");

       // Set the watermark image width.
        updateWatermarkTemplateRequest.setWidth("20");

       // Set the watermark image height.
        updateWatermarkTemplateRequest.setHeight("30");

       // Set the horizontal offset of the watermark image relative to the output video. The default value is 0.
        updateWatermarkTemplateRequest.setDx("20");

       // Set the vertical offset of the watermark image relative to the output video.
        updateWatermarkTemplateRequest.setDy("30");

       // Set the watermark position. The options are TopRight, TopLeft, BottomRight, and BottomLeft.
        updateWatermarkTemplateRequest.setReferpos("TopRight");

       // Set the watermark start time.
        updateWatermarkTemplateRequest.setTimelineStart("10");

       // Set the watermark duration. The default value is ToEND.
        updateWatermarkTemplateRequest.setTimelineDuration("ToEND");

       // Send a request to MPC.
        BaseResponse baseResponse =
mpcClient.updateWatermarkTemplate(updateWatermarkTemplateRequest);

       // Return a message.
        System.out.println(new Gson().toJson(baseResponse));
```


3.9.3 Querying Watermark Templates

Notes

- You can query watermark templates by template ID or page number.
- A maximum of 10 template IDs can be queried at a time.
- The page number and maximum number of templates on each page must be specified for pagination query. If no parameters are specified, **page** is **0** and **size** is **10**.

Core Code

1. Create a query request.
   ```
   // Create a request for querying a watermark template.
   QueryWatermarkTemplateRequest queryWatermarkTemplateRequest = new
   QueryWatermarkTemplateRequest();
   ```
2. Set query parameters.
   a. Query by template ID
      ```
      String[] templateIdArray = new String[]{"template_id1", "template_id2", "template_id3"};
      queryWatermarkTemplateRequest.setTemplateIds(templateIdArray);
      ```
   b. Query by page number
      ```
      // 2. Set the page number and number of records on each page.
      queryWatermarkTemplateRequest.setPage(2);
      queryWatermarkTemplateRequest.setSize(10);
      ```
3. Send a query request and return a message.
   ```
   // Send a request to MPC.
   QueryWatermarkTemplateResponse queryWatermarkTemplateResponse =
   mpcClient.queryWatermarkTemplate(queryWatermarkTemplateRequest);

   // Return a message.
   System.out.println(new Gson().toJson(queryWatermarkTemplateResponse));
   ```

Full Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.watermark.QueryWatermarkTemplateRequest;
import com.huawei.mpc.model.watermark.QueryWatermarkTemplateResponse;

     // Set the method for constructing MPC configuration items.
      MpcConfig mpcConfig = new MpcConfig();

    // Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
      mpcConfig.setEndPoint(endPoint);

     // Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
      mpcConfig.setProjectId(projectId);

     // Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
      mpcConfig.setSk(sk);

     // Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
      mpcConfig.setAk(ak);

      /*if you need proxy*/
     // Optional. Set the proxy if necessary.
      /*
         ClientConfig clientConfig = new ClientConfig();
        // Set the IP address of the proxy server.
```



```
        clientConfig.setProxyHost(proxyHost);
       // Set the port number of the proxy server.
         clientConfig.setProxyPort(Integer.parseInt(proxyPort));
        // Set the username for accessing the proxy server.
         clientConfig.setProxyUserName(proxyUserName);
       // Set the password for accessing the proxy server.
        clientConfig.setProxyPassword(proxyPassword);
       */

      // MPC construction method
       MpcClient mpcClient = new MpcClient(mpcConfig);

      // If a proxy needs to be set, use this constructor to initialize MpcClient.
       //MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);

      // Set request parameters.
       QueryWatermarkTemplateRequest queryWatermarkTemplateRequest = new
QueryWatermarkTemplateRequest();

       // 1. Query by template ID
       String[] templateIdArray = new String[]{"template_id1", "template_id2", "template_id3"};
       queryWatermarkTemplateRequest.setTemplateIds(templateIdArray);

       // 2. Set the page number and number of records on each page.
       // queryWatermarkTemplateRequest.setPage(2);
       // queryWatermarkTemplateRequest.setSize(10);

      // Send a request to MPC.
       QueryWatermarkTemplateResponse queryWatermarkTemplateResponse =
mpcClient.queryWatermarkTemplate(queryWatermarkTemplateRequest);

      // Return a message.
       System.out.println(new Gson().toJson(queryWatermarkTemplateResponse));
```

3.9.4 Deleting a Watermark Template

This section describes how to delete a watermark template based on its ID.

Core Code

```
// Send a request to MPC.
BaseResponse baseResponse = mpcClient.deleteWatermarkTemplate("template_id");

// Return a message.
System.out.println(new Gson().toJson(baseResponse));
```

Sample Code

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.BaseResponse;

// Set the method for constructing MPC configuration items.
       MpcConfig mpcConfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
       mpcConfig.setEndPoint(endPoint);

     // Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
       mpcConfig.setProjectId(projectId);

     // Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
       mpcConfig.setSk(sk);

     // Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
       mpcConfig.setAk(ak);
```



```
/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
   ClientConfig clientConfig = new ClientConfig();
  // Set the IP address of the proxy server.
    clientConfig.setProxyHost(proxyHost);
 // Set the port number of the proxy server.
     clientConfig.setProxyPort(Integer.parseInt(proxyPort));
   // Set the username for accessing the proxy server.
    clientConfig.setProxyUserName(proxyUserName);
  // Set the password for accessing the proxy server.
   clientConfig.setProxyPassword(proxyPassword);
  */

 // MPC construction method
  MpcClient mpcClient = new MpcClient(mpcConfig);

 // If a proxy needs to be set, use this constructor to initialize MpcClient.
  //MpcClient mpcClient =new MpcClient(mpcConfig, clientConfig);

  // Send a request to MPC.
  BaseResponse baseResponse = mpcClient.deleteWatermarkTemplate("template_id");

 // Return a message.
  System.out.println(new Gson().toJson(baseResponse));
```

3.10 Troubleshooting

If a service exception occurs, locate the fault using the following method.

Connection Timeout

Generally, the service address (endpoint) is incorrect or the network is disconnected. In this case, check the service address, network status, and proxy settings.

Error Handling

When the Java SDK is used, if an error occurs on the server or SDK, the SDK returns the corresponding error information. Error information includes an error code and an error message.

- Error code: Find the error description and error type by referring to **A.2 Error Codes**.
- Error information: helps quickly locate and rectify faults.

You can add the logic for fixing errors during development.

```
// If an error message is returned:
if(BaseResponse.FAIL == createTrancodingResponse.getStatus())
{
    // Add your own logic for fixing errors.
    // For example, print error details.
    System.out.println("ErrorCode ="+ rsp.getErrorCode());
    System.out.println("ErrorMsg ="+ rsp.getErrorMsg());
}
// If a success message is returned:
else
{
    // Return the result.
    System.out.println(new Gson().toJson(createTrancodingResponse));
}
```

Example Code

If you need to fix SDK errors, compile code by referring to the following example:

```
import com.google.gson.Gson;
import com.huawei.mpc.client.ClientConfig;
import com.huawei.mpc.client.MpcClient;
import com.huawei.mpc.client.MpcConfig;
import com.huawei.mpc.model.transcoding.CreateTrancodingResponse;
import com.huawei.mpc.model.transcoding.CreateTranscodingRequest;

// Set the method for constructing MPC configuration items.
MpcConfig mpcconfig = new MpcConfig();

// Mandatory. Set the endpoint of MPC. For details, see section "Obtaining Key Parameters."
mpcConfig.setEndPoint(endPoint);
// Mandatory. Set the user's project ID. For details, see section "Obtaining Key Parameters."
mpcConfig.setProjectId(projectId);

// Mandatory. Set the SK. For details, see section "Obtaining Key Parameters."
mpcConfig.setSk(sk);

// Mandatory. Set the AK. For details, see section "Obtaining Key Parameters."
mpcConfig.setAk(ak);

/*if you need proxy*/
// Optional. Set the proxy if necessary.
/*
ClientConfig clientConfig = new ClientConfig();
// Set the IP address of the proxy server.
clientConfig.setProxyHost(proxyHost);
// Set the port number of the proxy server.
clientConfig.setProxyPort(Integer.parseInt(proxyPort));
// Set the username for accessing the proxy server.
clientConfig.setProxyUserName(proxyUserName);
// Set the password for accessing the proxy server.
clientConfig.setProxyPassword(proxyPassword);
*/
// MPC construction method
MpcClient mpcClient = new MpcClient(mpcconfig);
/*
* proxy provided
// MPC construction method. If a proxy needs to be configured, use this construction method.
* MpcClient mpcClient=new MpcClient(mpcconfig, clientConfig);
*/

// Set request parameters.
CreateTranscodingRequest createTranscodingRequest = new CreateTranscodingRequest();
// Set the storage location of an input file.
CreateTranscodingRequest.ObsObjInfo input = new CreateTranscodingRequest.ObsObjInfo();

// Set the OBS bucket name.
input.setBucket("OBS bucket name");
// Set the region where the input OBS bucket is deployed.
input.setLocation("cn-north-4");
// Set the input file path.
input.setObject("Path");
createTranscodingRequest.setInput(input);

// Set the location of an output file.
CreateTranscodingRequest.ObsObjInfo output = new CreateTranscodingRequest.ObsObjInfo();
// Set the OBS bucket name.
output.setBucket("OBS bucket name");
// Set the region where the output OBS bucket is deployed.
output.setLocation("cn-north-4");
// Set the output file path.
output.setObject("Path");
createTranscodingRequest.setOutput(output);
```



```
/*
* Transcoding template ID, aln array, each transcoding output corresponds to a transcoding configuration
template ID example:trans_template_id : [203,212,210]
*/
// Template IDs
int[] transTempIds = {203, 212, 210};
// Set the transcoding template ID.
createTranscodingRequest.setThumb_template_id(transTempIds);
// Send a transcoding request.
CreateTrancodingResponse
createTrancodingResponse=mpcClient.createTranscodingTask(createTranscodingRequest);

// If an error message is returned:
if(BaseResponse.FAIL == createTrancodingResponse.getStatus())
{
    // Add your own logic for fixing errors.
    // For example, print error details.
    System.out.println("ErrorCode ="+ rsp.getErrorCode());
    System.out.println("ErrorMsg ="+ rsp.getErrorMsg());
}
// If a success message is returned:
else
{
    // Return the result.
    System.out.println(new Gson().toJson(createTrancodingResponse));
}
```

3.11 Mappings Between MPC SDK & APIs

Table 3-11 Mapping between MPC SDK and APIs

API	Request URI	Description
createTranscoding-Task	POST /v1/{project_id}/transcodings	Creating a Transcoding Task
deleteTranscoding-Task	DELETE /v1/{project_id}/transcodings{?task_id}	Deleting a Transcoding Task
queryTranscoding-Task	GET /v1/{project_id}/transcodings{?task_id}	Querying Transcoding Tasks
createTemplate	POST /v1/{project_id}/template/transcodings	Creating a Transcoding Template
deleteTemplate	DELETE /v1/{project_id}/template/transcodings{?temp_id}	Deleting a Transcoding Template

API	Request URI	Description
modifyTemplate	PUT /v1/{project_id}/template/transcodings	Modifying a Transcoding Template
queryTemplate	GET /v1/{project_id}/template/transcodings{?temp_id}	Querying Transcoding Templates
createThumbnail-sTask	POST /v1/{project_id}/thumbnails	Creating a Snapshot Task
deleteThumbnail-Task	DELETE /v1/{project_id}/thumbnails{?task_id}	Canceling a Snapshot Task
queryThumbnailsTask	GET /v1/{project_id}/thumbnails{?task_id,start_time,end_time,status,page,size}	Querying Snapshot Tasks
createWatermark-Template	POST /v1/{project_id}/template/watermark	Creating a Watermark Template
updateWatermark-Template	PUT /v1/{project_id}/template/watermark	Modifying a Watermark Template
queryWatermarkTemplate	GET /v1/{project_id}/template/watermark{?template_id,page,size}	Querying Watermark Templates
deleteWatermark-Template	DELETE /v1/{project_id}/template/watermark{?template_id}	Deleting a Watermark Template

API	Request URI	Description
createEncryptRequest	POST /v1/{project_id}/encryptions	Creating an Encryption Task
deleteEncryptTask	DELETE /v1/{project_id}/encryptions/?task_id={task_id}	Canceling an Encryption Task
queryEncryptRequest	GET /v1/{project_id}/encryptions{?task_id,start_time,end_time,status,page,size}	Querying Encryption Tasks
createAnimatedGraphicsTask	POST /v1/{project_id}/animated-graphics	Creating an Animated GIF Task
queryAnimatedGraphicsTask	GET /v1/{project_id}/animated-graphics{?task_id,start_time,end_time,status,page,size}	Querying Animated GIF Tasks
deleteAnimatedGraphicsTask	DELETE /v1/{project_id}/animated-graphics?task_id={task_id}	Canceling an Animated GIF Task
createExtractTask	POST /v1/{project_id}/animated-graphics	Creating a Video Parsing Task
queryExtractTask	GET /v1/{project_id}/animated-graphics{?task_id,start_time,end_time,status,page,size}	Querying Video Parsing Tasks
deleteExtractTask	DELETE /v1/{project_id}/animated-graphics{?task_id}	Canceling a Video Parsing Task
creatRemuxTask	POST /v1/{project_id}/extract-metadata	Creating a Packaging Task

API	Request URI	Description
queryRemuxTask	GET /v1/{project_id}/template/extract-metadata{?task_id,start_time,end_time,status,page,size}	Querying Packaging Tasks
cancelRemuxTask	DELETE /v1/{project_id}/extract-metadata{?task_id}	Canceling a Packaging Task

4 Python SDK

Preparing a Development Environment

- JDK 1.8 or later has been installed and the environment has been configured. For details about how to install the JDK, see **JDK Installation**.
- Python 3.7 or later has been installed. If not, download it from the **Python official website**.
- PIP has been installed. If not, download it from the **PIP official website**.
- A development environment such as JetBrains PyCharm 2018.1.4 x64 has been prepared.

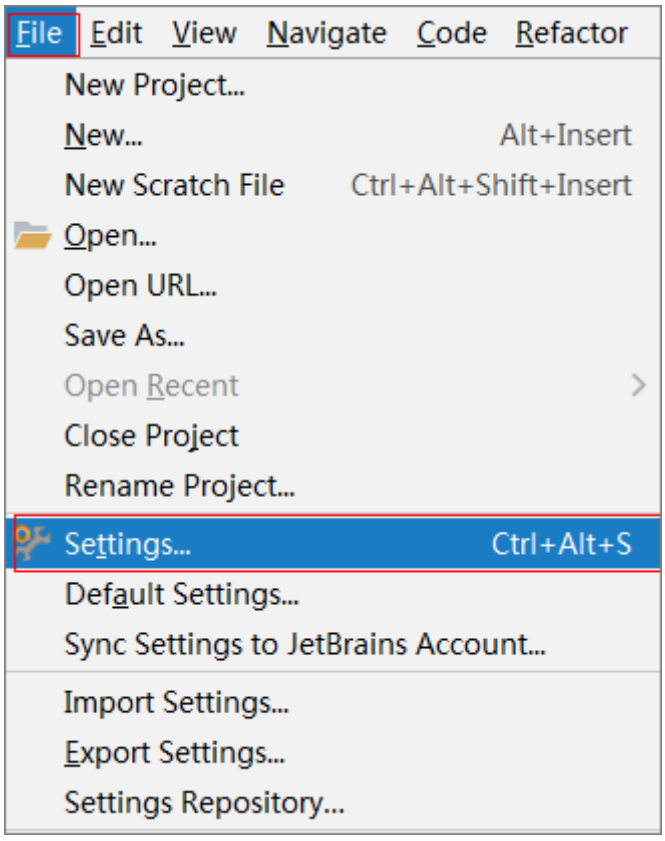
Integrating SDK

Step 1 Decompress the downloaded mpc-python-sdk package to your local PC.

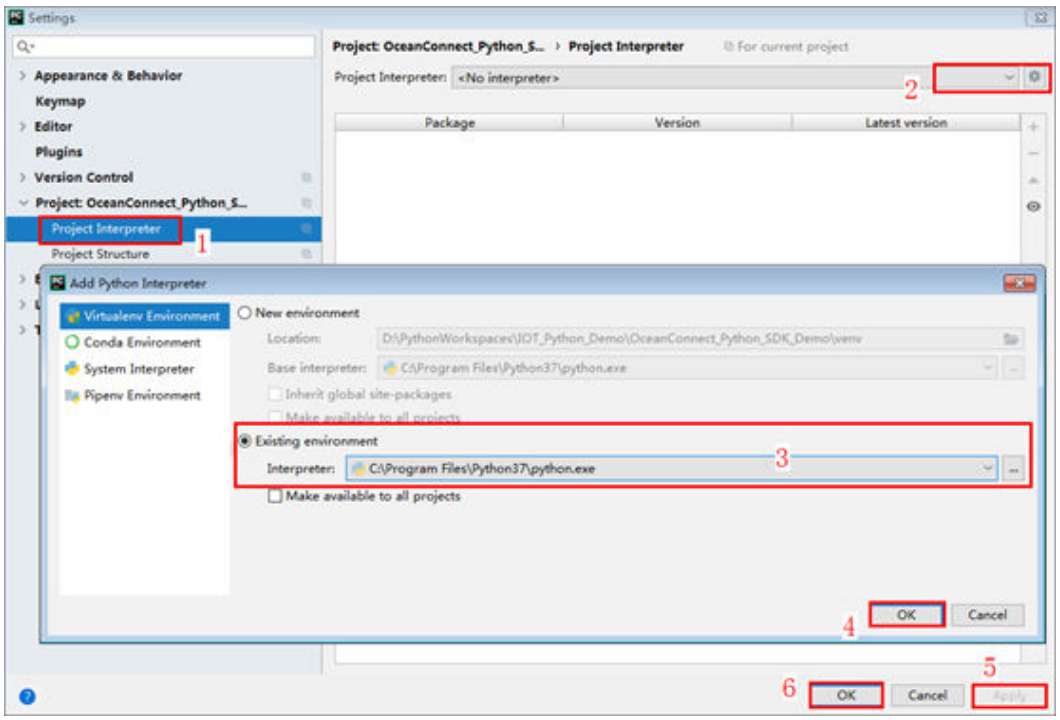
Step 2 Open PyCharm, select **Open**, select the path where the SDK is decompressed, and click **OK**.

Step 3 Configure the project interpreter.

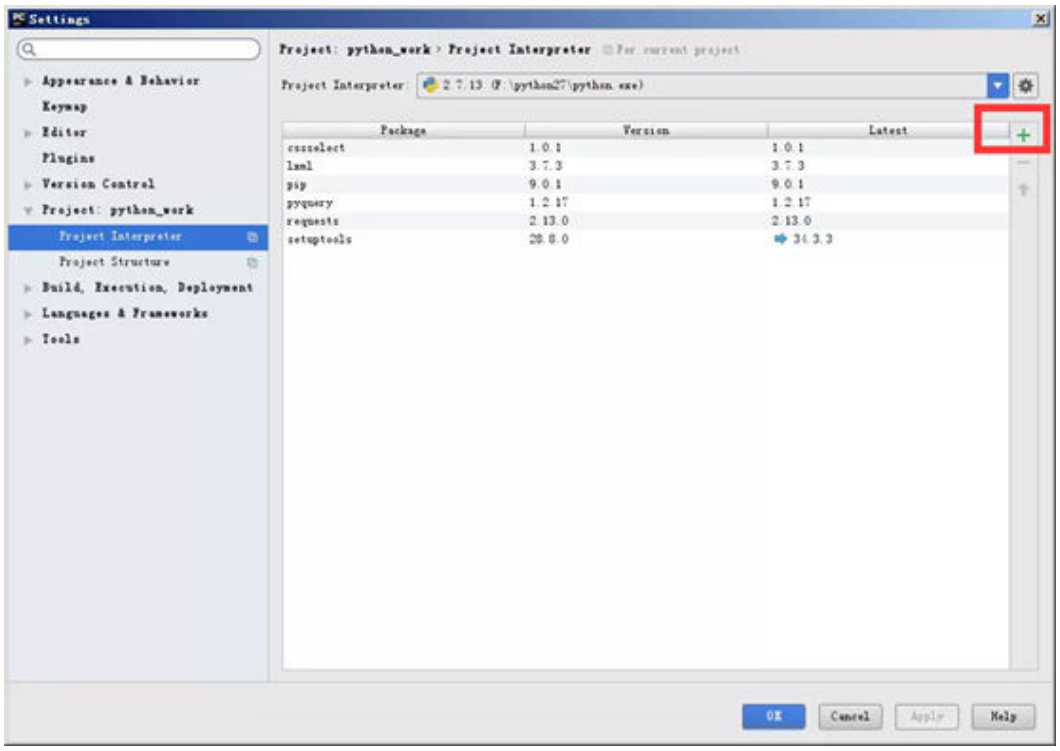
1. Choose **File > Settings > Project: OceanConnect _ Python _SDK_Demo > Project Interpreter**.



2. Select the interpreter where the SDK is located, for example, **C:\Program Files \Python37\python.exe**, click **Apply**, and then click **OK**.



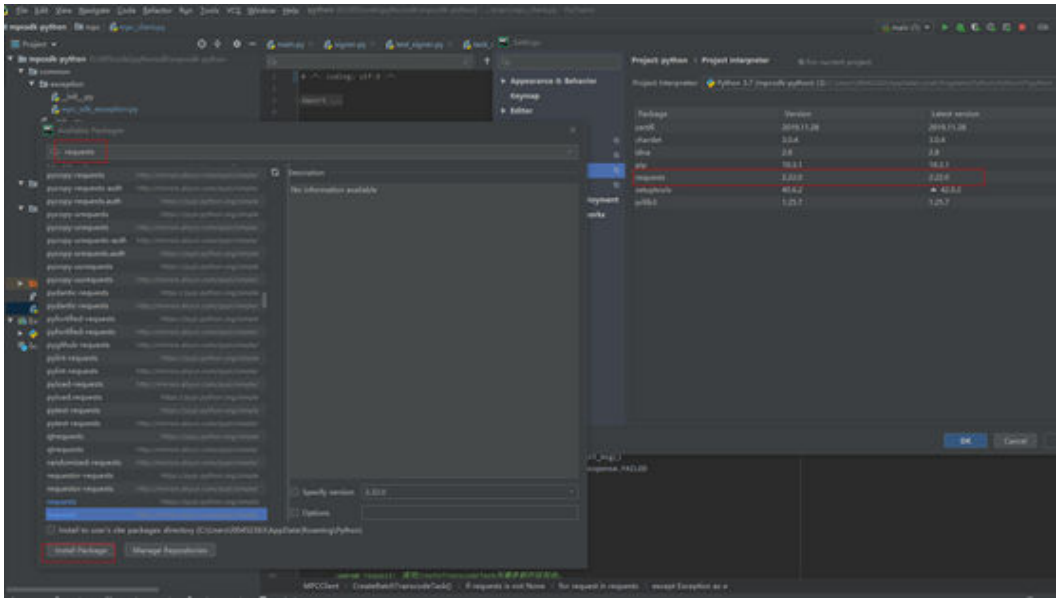
3. Add the requests module to PyCharm, choose **pycharm > File > Settings > Project: Python > Project Interpreter> file**, and click the green plus sign (+) on the right.



4. On the page displayed, search for **requests** and click **install Package**.

   NOTE

   If the installation fails, ensure that PIP of the latest version is installed.



----End

Setting MPC Configuration Items

MPC configuration items are used to obtain authorization. You can set the configuration items when creating an MpcClient instance. **Table 4-1** describes the parameters.

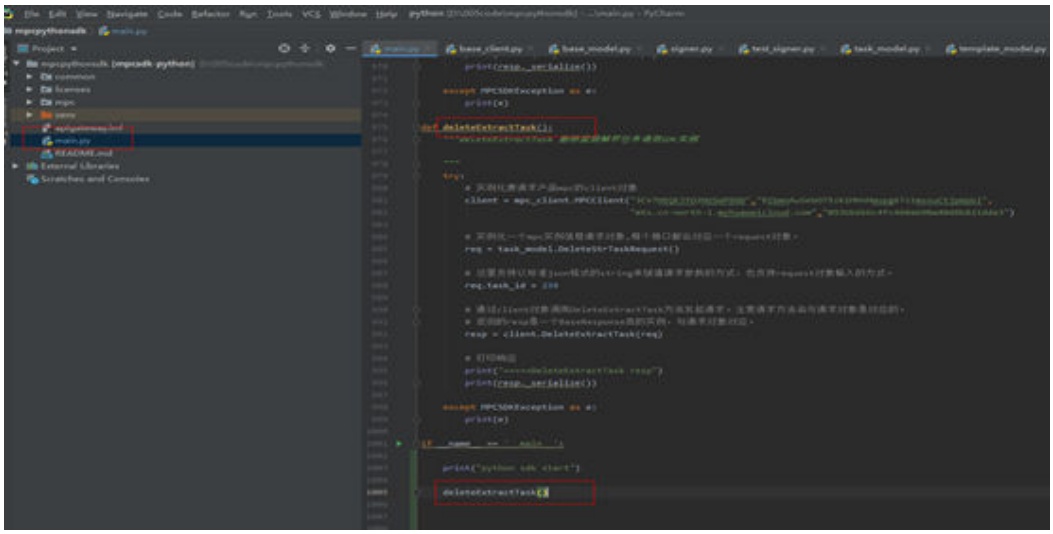

```
# Initialize the client object.
client = mpc_client.MPCClient("ak","sk", "endpoint","ProjectId")
```

Table 4-1 MPC parameters

Parameter	Type	Description
ak	String	Access key ID (AK). For details, see **Obtaining the AK/SK Pair**.
sk	String	Secret Access Key (SK) used together with the AK. For details, see **Obtaining the AK/SK Pair**.
endPoint	String	Endpoint. For details, see **Obtaining an Endpoint**.
ProjectId	String	Project ID. For details, see **Obtaining a Project ID and Account Name**.

API Calling

All service APIs are in **main.py**. You can add the target API to **if __name__ == '_main__'**. **Table 4-2** shows the APIs in the Python SDK.



Mappings Between MPC SDK & APIs

Table 4-2 Mapping between MPC Python SDK and APIs

API	Request URI	Description
createTrancodeTemplate	POST /v1/{project_id}/template/transcodings	Creating a Transcoding Template
deleteTrancodeTemplate	DELETE /v1/{project_id}/template/transcodings	Deleting a Transcoding Template
createTrancodeTask	POST /v1/{project_id}/transcodings	Creating a Transcoding Task
deleteTrancodeTask	DELETE /v1/{project_id}/transcodings	Deleting a Transcoding Task
createThumbnailTask	POST /v1/{project_id}/thumbnails	Creating a Snapshot Task
deleteThumbnailTask	DELETE /v1/{project_id}/thumbnails	Deleting a Snapshot Task
createExtractTask	POST /v1/{project_id}/extract-metadata	Creating a Video Parsing Task
deleteExtractTask	DELETE /v1/{project_id}/extract-metadata	Deleting a Video Parsing Task

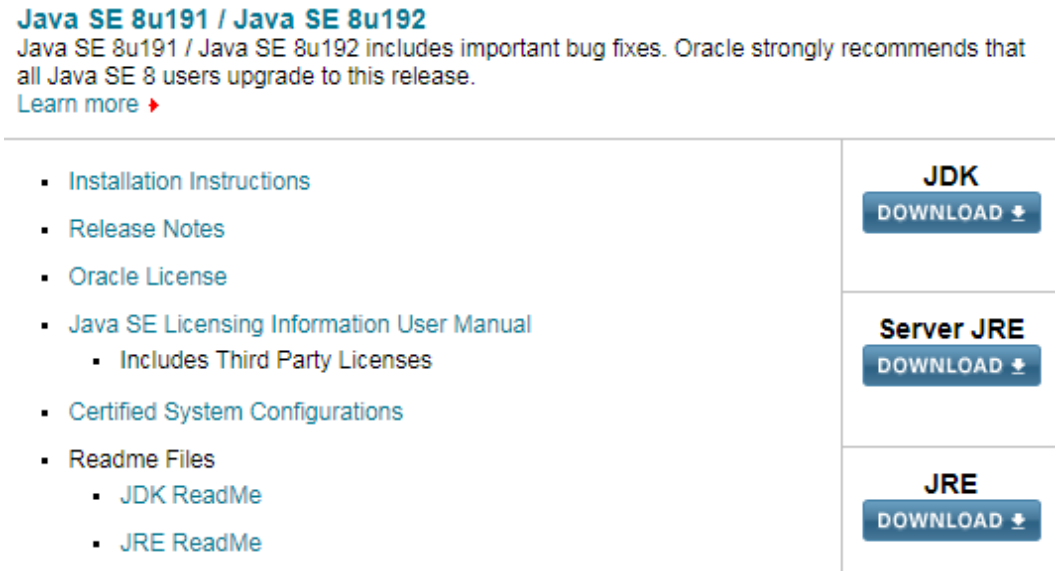
A Appendix

A.1 JDK Installation

The MPC SDK uses JDK 8 or later. The following uses JDK 8 (Windows x64) running on Windows 7 as an example. If you have downloaded the JDK and configured the environment, skip this section.

Procedure

Step 1 Download the JDK file from the **official website**. JDK 8 is used as an example. Click **DOWNLOAD** below JDK.



Step 2 After the JDK file is downloaded, install the JDK as prompted. For example, install the JDK to the **C:\Program Files\Java\jdk1.8.0_131** directory on the local PC.

Step 3 Configure Java environment variables.

1. Right-click **Computer**, choose **Properties** > **Advanced system settings**.
2. Click the **Advanced** tab, and then click **Environment Variables**.

3. Set JAVA_HOME, PATH, and CLASSPATH (case-insensitive) in the **System variables** area. See **Table A-1**.

   If the three variables already exist, click **Edit**. If not, click **New**.

   Table A-1 JAVA environment variables

Variable	Value	Description
JAVA_HOME	JDK installation path	Example: **C:\Program Files (x86)\Java\jdk1.8.0_1311**
PATH	%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin	Add it to the end of the original **PATH** value.
CLASSPATH	.;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar;	There is a dot (.) in front of the value.

Step 4 Open the CLI and run **java -version**. The Java environment variables have been configured if the Java version is displayed.

The following is a successful response example:

```
C:\>java -version
java version "1.8.0_131"
Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)
```

----End

A.2 Error Codes

The format of an error code is "MPC.Internal error code."

Table A-2 Error codes

Error Code	Description
MPC.10050	System error. Contact technical support.
MPC.10051	The selected template is a super-resolution template and is not supported now.
MPC.10052	Failed to obtain the input file. Check the path.
MPC.10053	The input file does not exist.
MPC.10054	Failed to obtain the subtitle file.
MPC.10055	The audio sampling rate 7,350 will be supported soon.
MPC.10056	This type of output frame rate will be supported soon.
MPC.10057	This type of output bitrate will be supported soon.

Error Code	Description
MPC.10058	This type of output video width will be supported soon.
MPC.10059	This type of output video height will be supported soon.
MPC.10060	This type of I-frame interval will be supported soon.
MPC.10061	Capturing snapshots at non-fixed intervals will be supported. Change the interval to a fixed interval.
MPC.10062	The video codec is incorrect in the snapshot scenario.
MPC.10063	The video format is incorrect in the snapshot scenario.
MPC.10064	Multiple watermark inputs will be supported soon. Currently, only two inputs are supported.
MPC.10065	This type of output file format will be supported soon.
MPC.10066	The format of the input file is incorrect.
MPC.10067	Failed to obtain the ID of the video codec.
MPC.10068	Failed to obtain the ID of the audio codec.
MPC.10069	Failed to obtain the ID of the subtitle codec.
MPC.10070	Failed to obtain the encoding/decoding format.
MPC.10071	Failed to obtain the parameters of the input video stream.
MPC.10072	The frame rate of the video stream is incorrect.
MPC.10080	Failed to open the input file.
MPC.10081	The file does not contain audio streams.
MPC.10082	Failed to obtain the input audio or video stream.
MPC.10083	This type of codec will be supported soon.
MPC.10084	This chroma subsampling format will be supported soon.
MPC.10085	The file format is not supported.
MPC.10086	Failed to obtain the input file. Check the path.
MPC.10087	Incorrect task parameters.
MPC.10088	The image file does not exist.
MPC.10089	The template file does not exist.
MPC.10090	The template file does not exist.
MPC.10091	The template name already exists.

Error Code	Description
MPC.10092	The image file does not exist.
MPC.10093	The file name is too long.
MPC.10094	File format error.
MPC.10095	The watermark is placed in a wrong position. The watermark size cannot exceed the video range.
MPC.10096	The watermark size is incorrect. The resolution of a watermark cannot be lower than 8x8 and greater than 4,096x4,096. The size of the watermark cannot exceed 10 MB.
MPC.10097	The watermark scaling ratio is incorrect. The watermark can only be scaled down. The watermark scaling-down ratio cannot exceed 256.
MPC.10098	Incorrect watermark duration.
MPC.10099	The media stream type is not supported.
MPC.10100	An error occurred when parsing the video frame rate information.
MPC.10101	Incorrect input parameter.
MPC.10102	Failed to open the input file.
MPC.10103	Open GOP will be supported soon.
MPC.10104	System error. Contact technical support.
MPC.10105	The transcoding process is abnormal.
MPC.10106	The audio sampling rate is lower than 12,000. The audio will be discarded.
MPC.10107	The input video resolution is incorrect.
MPC.10108	The audio sampling rate of the input video is incorrect.
MPC.10109	Incorrect resolution in the template.
MPC.10110	The watermark is placed in a wrong position.
MPC.10111	Failed to obtain the OBS file.
MPC.10112	The video or audio format of the input file is not supported.
MPC.10113	The Decode Time Stamp (DTS) of the input file is abnormal.
MPC.10114	The input OBS file does not exist.

Error Code	Description
MPC.10115	The watermark cannot be scaled down by more than 256 times.
MPC.10116	The audio encoding format of the input file is not supported.
MPC.10117	The audio and video in the input file are not synchronized.
MPC.10118	Failed to upload files to the OBS path.
MPC.10119	Data in the input file is invalid.
MPC.10120	The task does not exist.
MPC.10121	The duration of a WMV file exceeds three hours.
MPC.10122	The resolution in the template is greater than the input video resolution.
MPC.10123	The header information of the input file is incorrect.
MPC.10124	Some data in the input file are missing.
MPC.10125	Input data error. Check whether the input file can be played.
MPC.10126	Input data error. Check whether the input file can be played.
MPC.10135	Failed to package the digital watermark due to the incorrect xformat configuration.
MPC.10136	Failed to package the digital watermark because xformat fails to be started.
MPC.10137	xformat failed to create a digital watermark packaging task.
MPC.10138	xformat failed to query the digital watermark packaging task.
MPC.10139	Failed to package the digital watermark. The xformat task timed out.
MPC.10200	System error. Contact technical support.
MPC.10201	Internal communication error. Contact technical support.
MPC.10202	Invalid request parameter, {0}.
MPC.10203	Identity authentication fails.
MPC.10204	Incorrect request method.
MPC.10205	Incorrect request content type.

Error Code	Description
MPC.10206	Your real name has not been authenticated.
MPC.10207	Your account is in an abnormal state.
MPC.10208	Failed to verify the tenant ID.
MPC.10209	Invalid input or output OBS path.
MPC.10210	Failed to obtain the input file from OBS.
MPC.10211	The task does not exist.
MPC.10212	Operation failed. The task is in progress or has been completed.
MPC.10213	Operation failed. The task is not in the final state.
MPC.10214	The topic does not exist.
MPC.10215	The topic already exists.
MPC.10216	Failed to set event notifications. You do not have the permission to publish messages to the topic.
MPC.10217	The OBT quota exceeds the threshold.
MPC.10218	The task has completed.
MPC.10219	Invalid request parameter.
MPC.10220	The task has expired.
MPC.10221	Internal service error. Check the template and try again.
MPC.10222	Key parameters in the template are inconsistent.
MPC.10223	An agency has been created.
MPC.10224	The agency has been deleted.
MPC.10225	Failed to obtain the key.
MPC.10226	The resource does not exist.
MPC.10227	You do not have the permission to access the requested resource.
MPC.10228	Your account is in arrears. Top up your account.
MPC.10229	You do not have the permission to perform this operation.

A.3 Status Codes

The following table lists the status codes.

Status Code	Description
200 OK - [GET]	The request has succeeded.
201 CREATED - [POST/PUT/PATCH]	Data has been deleted or modified.
202 Accepted - [*]	The request has been put into the queue (asynchronous tasks).
204 NO CONTENT - [DELETE]	Data has been deleted.
400 INVALID REQUEST - [POST/PUT/PATCH]	Data is not created or deleted due to a client error.
401 Unauthorized - [*]	You do not have permission to perform this operation. The token, username, or password is incorrect.
403 Forbidden - [*]	You are authorized (opposite to error code 401), but access is forbidden.
404 NOT FOUND - [*]	No operation has been performed because the requested data does not exist.
406 Not Acceptable - [GET]	The format requested by a user is not supported. For example, the user requests the JSON format, but only the XML format is available.
410 Gone -[GET]	The target resource is no longer available at the origin server and that this condition is likely to be permanent.
422 Unprocesable entity - [POST/PUT/PATCH]	Validation failed during object creation.
500 INTERNAL SERVER ERROR - [*]	The server encountered an unexpected condition that prevented it from fulfilling the request.

A.4 Obtaining Key Parameters

Before using the SDK, you need to obtain the following key parameters for signature authentication:

- AK: access key ID, which is a unique identifier used in conjunction with a secret access key to sign requests cryptographically.
- SK: secret access key used in conjunction with an AK to sign requests cryptographically. It identifies a request sender and prevents the request from being modified.
- Project_ID: project ID. Some request URLs must contain this field.

- Account name: This field must be contained in some request URLs.
- Endpoint: regions covered by HUAWEI CLOUD and endpoints for available services

Prerequisites

You have **registered** with HUAWEI CLOUD and completed **real-name authentication**.

Obtaining the AK/SK Pair

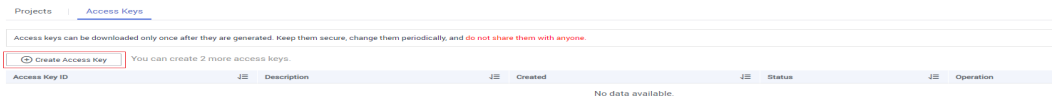
Note: An access key has full access permissions for your account. If the access key is disclosed, data leakage may occur. For account security, you are advised to periodically change and keep the access key secure. Each account can create a maximum of two access keys.

Step 1 Log in to the **HUAWEI CLOUD website**, hover the mouse cursor over the username in the upper right corner, and choose **My Account**.

Step 2 On the **Basic Information** page, click **Manage** beside **Security Credentials**.

Step 3 In the navigation pane, choose **Access Keys**. Click **Create Access Key**. On the displayed page, enter the HUAWEI CLOUD account and SMS verification code.

Figure A-1 Access key



Step 4 Click **OK** to download the **credentials.csv** file that contains the AK and SK pair.

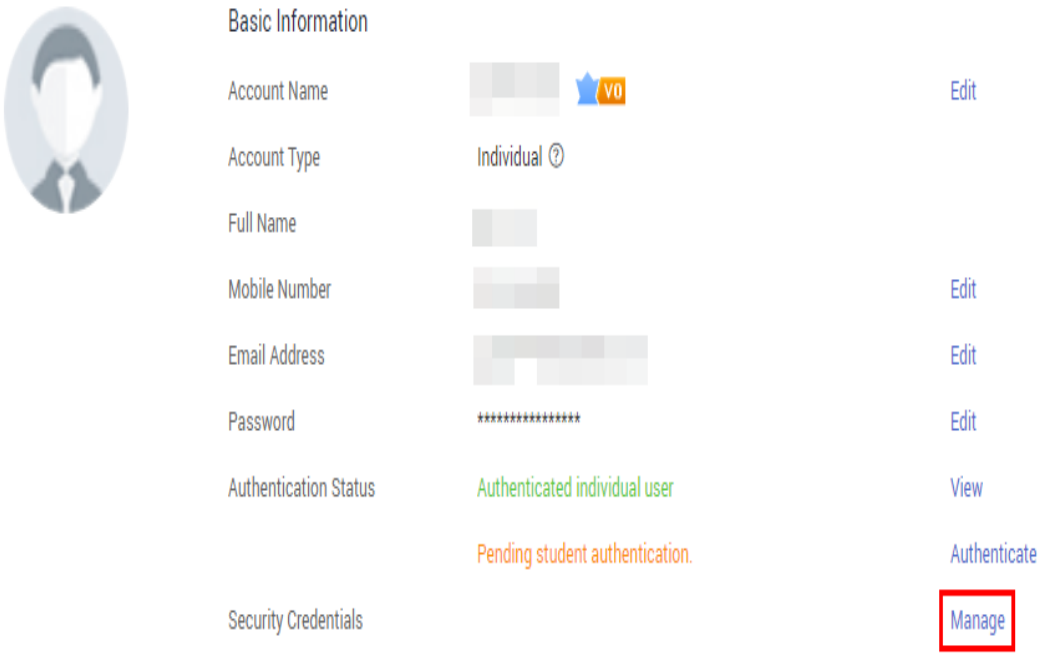
----End

Obtaining a Project ID and Account Name

Step 1 Log in to the **HUAWEI CLOUD website**, hover the mouse cursor over the username in the upper right corner, and choose **My Account**.

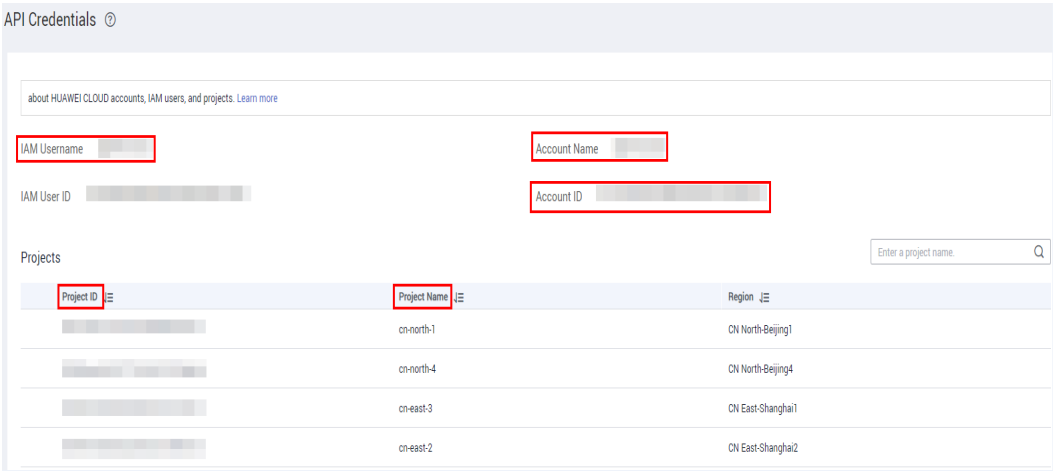
Step 2 On the **Basic Information** page, click **Manage** beside **Security Credentials**.

Figure A-2 Basic information



Step 3 On the displayed **API Credentials** page, view the **Project ID**.

Figure A-3 Obtaining a Project ID



----End

Obtaining an Endpoint

The endpoint is required during SDK initialization. You can obtain the endpoints of MPC from **Regions and Endpoints**.

