Media Processing Center

SDK Reference

Issue 01

Date 2024-05-09

HUAWEI

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property 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 Cloud 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.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 SDK Overview....1
2 Java SDK....2
2.1 Prerequisites....2
2.2 Video Transcoding....7
2.2.1 Creating a Transcoding Task....7
2.2.2 Canceling a Transcoding Task....9
2.2.3 Querying Transcoding Tasks....10
2.3 Snapshot Capturing....12
2.3.1 Creating a Snapshot Task....12
2.3.2 Canceling a Snapshot Task....14
2.3.3 Querying Snapshot Tasks....15
2.4 Encryption....17
2.4.1 Creating an Encryption Task....17
2.4.2 Canceling an Encryption Task....19
2.4.3 Querying Encryption Tasks....20
2.5 Animated GIFs....22
2.5.1 Creating an Animated GIF Task....22
2.5.2 Querying Animated GIF Tasks....24
2.5.3 Canceling an Animated GIF Task....25
2.6 Video Parsing....26
2.6.1 Creating a Video Parsing Task....27
2.6.2 Querying Video Parsing Tasks....28
2.6.3 Canceling a Video Parsing Task....30
2.7 Packaging....31
2.7.1 Creating a Packaging Task....31
2.7.2 Querying Packaging Tasks....33
2.7.3 Canceling a Packaging Task....35
2.8 Transcoding Templates....36
2.8.1 Creating a Transcoding Template....36
2.8.2 Deleting a Transcoding Template....38
2.8.3 Updating a Transcoding Template....39
2.8.4 Querying Transcoding Templates....41
2.9 Watermarking....43

2.9.1 Creating a Watermark Template....43
2.9.2 Updating a Watermark Template....44
2.9.3 Querying Watermark Templates....46
2.9.4 Deleting a Watermark Template....47
2.10 Mappings Between MPC SDK and APIs....49
3 Python SDK....52
4 Go SDK....58
5 Appendix....63
5.1 JDK Installation....63
5.2 Error Codes....64
5.3 Status Codes....80
5.4 Obtaining Key Parameters....81

1 SDK Overview

Media Processing Center (MPC) software development kits (SDKs) allow you to create, cancel, and query transcoding tasks, as well as create, delete, modify, and query transcoding templates.

The Java, Python, and Go SDKs are available. If you need SDKs in other programming languages, call **MPC APIs**.

NOTE

The MPC SDK code does not support escape.

Table 1-1 Server SDK

Language	GitHub Address	Reference
JAVA	**huaweicloud-sdk-java-v3**	**Java SDK User Guide**
Python	**huaweicloud-sdk-python-v3**	**Python SDK User Guide**
Go	**huaweicloud-sdk-go-v3**	**Go SDK User Guide**

2 Java SDK

2.1 Prerequisites

This section describes how to quickly integrate Java SDKs for development.

Prerequisites

- You have **registered** with Huawei Cloud and completed **real-name authentication**.

  NOTE

  If you are a **Huawei Cloud (International)** user, you need to complete real-name authentication when you:

  - Purchase and use cloud services on Huawei Cloud nodes in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Select the Chinese mainland region for MPC.
- The development environment (Java JDK 1.8 or later) is available.
- You have obtained the access key ID (AK) and secret access key (SK) of the account. You can create and view your AK/SK on the **My Credentials** > **Access Keys** page of the console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of MPC. You can view the project ID on the **My Credentials** > **API Credentials** page of the console. For details, see **API Credentials**.
- You have uploaded the media asset files to an OBS bucket in the region of MPC, and authorized MPC to access the OBS bucket. For details, see **Uploading Media Files** and **Authorizing Access to Cloud Resources**.

Procedure

Step 1 Import the dependent module.

```
// User authentication
import com.huaweicloud.sdk.core.auth.BasicCredentials;
// Request exceptions
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ServerResponseException;
```



```
// Configure HTTP.
import com.huaweicloud.sdk.core.http.HttpConfig;
// Import an MPC client.
import com.huaweicloud.sdk.mpc.v1.MpcClient;
// Import the request and response classes of an API.
import com.huaweicloud.sdk.mpc.v1.model.ListTranscodingTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListTranscodingTaskResponse;
// Print logs.
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
```

Step 2 Configure client attributes.

1. Use the default configuration.
   ```
   // Use the default configuration.
   HttpConfig config = HttpConfig.getDefaultHttpConfig();
   ```
2. (Optional) Configure the proxy.
   ```
   // (Optional) Use a proxy server.
   config.withProxyHost("http://proxy.myhuaweicloud.com")
       .withProxyPort(8080)
       .withProxyUsername("test")
       .withProxyPassword("test");
   ```
3. (Optional) Configure the connection.
   ```
   // (Optional) Configure the connection timeout interval.
   config.withTimeout(3);
   ```
4. (Optional) Configure SSL.
   ```
   // (Optional) Configure whether to skip server certificate verification.
   config.withIgnoreSSLVerification(true);
   ```

Step 3 Initialize authentication information.

You can use one of the following two authentication modes:

- Permanent AK/SK

  Obtain the permanent AK, SK, and project ID. For details, see **Prerequisites**.

  ```
  BasicCredentials credentials = new BasicCredentials()
      .withAk(ak)
      .withSk(sk)
      .withProjectId(projectId)
  ```

- Temporary AK/SK

  Obtain a temporary AK, SK, and SecurityToken. For details, see **Obtaining a Temporary Access Key and SecurityToken Through a Token** or **Obtaining a Temporary Access Key and SecurityToken Through an Agency**.

  ```
  BasicCredentials credentials = new BasicCredentials()
          .withAk(ak)
          .withSk(sk)
          .withSecurityToken(securityToken)
          .withProjectId(projectId)
  ```

The related parameters are as follows:

- **ak**: access key of an account.
- **sk**: secret access key of an account.
- **projectId**: project ID of the region where MPC is provided. Select a project ID based on the region of the project.
- **securityToken**: security token used for temporary AK/SK authentication

Step 4 Initialize the client.

```
// Initialize the MPC client.
MpcClient MpcClient = MpcClient.newBuilder()
    .withHttpConfig(config)
```



```
    .withCredential(credentials)
    .withEndpoint(endpoint)
    .build();
```

endpoint: regions where MPC is used and endpoints of each service. For details, see **Regions and Endpoints**.

Step 5 Send a request and view the response.

```
// Initialize the request. The following uses the API for querying transcoding templates as an example.
ListTranscodingTaskResponse response = mpcClient.
listTranscodingTask(new ListTranscodingTaskRequest().withTaskId(Collections.singletonList(1900293L))
));
logger.info(response.toString());
```

Step 6 Perform troubleshooting.

Table 2-1 Troubleshooting

Level 1	Description	Level 2	Description
ConnectionException	Connection exception	HostUnreachableException	The network is unreachable or access is rejected.
		SslHandShakeException	SSL authentication is abnormal.
RequestTimeoutException	Response timeout exception	CallTimeoutException	The server fails to respond to a single request before timeout.
		RetryOutageException	No valid response is returned after the maximum number of retries specified in the retry policy is reached.
ServiceResponseException	Server response exception	ServerResponseException	Internal server error. HTTP response code: [500,].
		ClientRequestException	Invalid request parameter. HTTP response code: [400, 500).

```
// Troubleshooting
try {
ListTranscodingTaskResponse  response= mpcClient.listTranscodingTask(new
ListTranscodingTaskRequest().withTaskId(Collections.singletonList(1900293L)));
} catch(ServiceResponseException e) {
    logger.error("HttpStatusCode: " + e.getHttpStatusCode());
    logger.error("RequestId: " + e.getRequestId());
    logger.error("ErrorCode: " + e.getErrorCode());
    logger.error("ErrorMsg: " + e.getErrorMsg());
}
```

Step 7 Use an asynchronous client.


```
// Initialize an asynchronous client.
MpcAsyncClient mpcAsyncClient =
MpcAsyncClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(credentials)
    .withEndpoint(endpoint)
    .build();

// Send an asynchronous request.
CompletableFuture<ListTranscodingTaskResponse> future = mpcAsyncClient.listTranscodingTaskAsync(new
ListTranscodingTaskRequest().withTaskId(Collections.singletonList(1900293L)));

// Obtain the asynchronous response.
ListTranscodingTaskResponse response = future.get();
logger.info(response.toString());
```

Step 8 Print access logs.

The running SDK uses Simple Logging Facade for Java (SLF4J) to print logs. If the logging library is not configured when the code instance is run, the following information is displayed:

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
```

You can introduce the logging library dependencies to the **pom.xml** file of the target project, as shown in the following examples.

- SLF4J

```
 <dependency>
     <groupId>org.slf4j</groupId>
     <artifactId>slf4j-simple</artifactId>
     <version>1.7.21</version>
 </dependency>
logback

<dependency>
    <groupId>ch.qos.logback</groupId>
    <artifactId>logback-classic</artifactId>
    <version>1.2.3</version>
</dependency>
<dependency>
    <groupId>ch.qos.logback</groupId>
    <artifactId>logback-core</artifactId>
    <version>1.2.3</version>
</dependency>
```

- Log4j

```
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.17</version>
</dependency>
```

By default, the SDK prints access logs. Each request has a record named **huaweiCloud-SDK-Access**. The log format is as follows:

```
"{httpMethod} {uri}" {httpStatusCode} {responseContentLength} {requestId}
```

requestId indicates the request ID returned by API Gateway, which can be used for issue tracking.

You can disable access logs in the corresponding log configuration file or record access logs in an independent file. For example, you can disable access logs in the Logback framework by adding the following configuration:

```
<logger name="huaweiCloud-SDK-Access" level="OFF"> </logger>
```


Step 9 Use the listener to obtain original HTTP requests and responses.

The original HTTP requests and responses are required for debugging HTTP requests sent by the service side. The SDK provides the listener to obtain the original and encrypted HTTP requests and responses.

CAUTION

Original information is printed only during debugging. Do not print the header and body of an original HTTP request in the production system because this information contains sensitive data but is not encrypted. If the request body is binary, that is, **Content-Type** is set to **binary**, the body will be displayed as *** without the detailed content.

```
HttpConfig config = new HttpConfig().addHttpListener(HttpListener.forRequestListener(requestListener ->
    // After the listener is registered, the original information about the HTTP requests is printed. Do not
print the original information in the production system.
    logger.debug("REQUEST: {} {} {} {}",
        requestListener.httpMethod(),
        requestListener.uri(),
        requestListener.headers().entrySet().stream().flatMap(entry ->
            entry.getValue().stream().map(value -> entry.getKey() + " : " + value))
            .collect(Collectors.joining(";")),
        requestListener.body().orElse(""))))
    .addHttpListener(HttpListener.forResponseListener(responseListener ->
    // After the listener is registered, the original information about the HTTP requests is printed. Do not
print the original information in the production system.
    logger.debug("RESPONSE: {} {} {} {} {}",
        responseListener.httpMethod(),
        responseListener.uri(),
        responseListener.statusCode(),
        responseListener.headers().entrySet().stream().flatMap(entry ->
            entry.getValue().stream().map(value -> entry.getKey() + " : " + value))
            .collect(Collectors.joining(";")),
        responseListener.body().orElse(""))));

MpcClient mpcClient = MpcClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(auth)
    .withEndpoint(endpoint)
    .build();
```

----End

Sample Code: Initializing the MPC Client

Before calling this API, replace *{your ak string}*, *{your sk string}*, *{your endpoint string}*, and *{your project id}* with the actual values.

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;

public class InitMpc {
    private static HttpConfig httpConfig;
    private static BasicCredentials auth;
    private static String endpoint;
    private static MpcClient mpcClient;

    public static MpcClient getMpcClient() {
        httpConfig = HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
```



```
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxx").withProxyPort(xxxxx).withProxyUsername("xxxxx").
        //        withProxyPassword("xxxxx");

        String ak = "xxxxx";
        String sk = "xxxxx";
        String projectId = "xxxxx";
        endpoint = "https://mpc.region01.myhuaweicloud.com";
        auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        mpcClient = MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
        return mpcClient;
    }
}
```

2.2 Video Transcoding

2.2.1 Creating a Transcoding Task

You can create a transcoding task by creating an MPC client instance and configuring related parameters.

Core Code

1. **Create an MPC client instance.**

```
public static MpcClient initMpcClient() {
    // Set httpConfig.
    HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
    // Set the HTTP proxy as required.
    //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
          //withProxyPassword("xxxxxx");
    // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
    String ak = "xxxxxx";
    String sk = "xxxxxx";
       // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under
your account on the console.
    String projectId = "xxxxxx";
    // Enter the endpoint. The following uses region01 as an example.
    String endpoint = "https://mpc.region01.myhuaweicloud.com";
    BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
    return MpcClient.newBuilder()
            .withHttpConfig(httpConfig)
            .withCredential(auth)
            .withEndpoint(endpoint)
            .build();
}
```

2. **Create a transcoding request and set the request body.**

   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**.

```
// Set the input file path.
ObsObjInfo input = new ObsObjInfo()
    // Set the bucket name.
    .withBucket("mpc-east-2")
    // Set the region where the OBS bucket is located.
    .withLocation("region01")
    // Set the input file object.
```



```
    .withObject("input/ok.mp4");
// Set the output file path.
ObsObjInfo output = new ObsObjInfo().withBucket("mpc-east-2").withLocation("region01")
        // Set the output file path.
        .withObject("output");
// Create a transcoding request.
CreateTranscodingTaskRequest request
        = new CreateTranscodingTaskRequest().withBody(new CreateTranscodingReq()
        .withInput(input)
        .withOutput(output)
        // Configure a transcoding template. To view IDs of system templates, choose Global Settings > System Templates on the MPC console.
        .withTransTemplateId(Collections.singletonList(7000530))
        // Set output file names. Each template has a name.
        .withOutputFilenames(Collections.singletonList("output_"))
        // Configure snapshot parameters and fill in the thumbnail structure as required.
        //.withThumbnail(new Thumbnail())
        // Configure encryption parameters and fill in the encryption structure as required.
        //.withEncryption(new Encryption())
);
```

3. **Send the transcoding request.**

```
// Send the transcoding request.
CreateTranscodingTaskResponse response = initMpcClient().createTranscodingTask(request);
// Return a message.
System.out.println("CreateTranscodingTaskResponse=" + response);
```

Sample Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CreateTranscodingReq;
import com.huaweicloud.sdk.mpc.v1.model.CreateTranscodingTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateTranscodingTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.ListTranscodingTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListTranscodingTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.ObsObjInfo;
import com.obs.services.internal.ServiceException;
import org.junit.Test;

import java.util.Arrays;
import java.util.Collections;

public class TestTranscode {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
```



```
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create a transcoding task.
     * @param args
     */
    public static void main(String[] args) {
        // Set the input file path.
        ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.mp4");
        // Set the output file path.
        ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
        // Create a transcoding request.
        CreateTranscodingTaskRequest request
                = new CreateTranscodingTaskRequest().withBody(new CreateTranscodingReq()
                .withInput(input)
                .withOutput(output)
                // Configure a transcoding template. To view IDs of system templates, choose Global Settings >
System Templates on the MPC console.
                .withTransTemplateId(Collections.singletonList(7000530))
                // Set output file names. Each template has a name.
                .withOutputFilenames(Collections.singletonList("output_"))
                // Configure snapshot parameters.
                //.withThumbnail(new Thumbnail())
                // Configure encryption parameters.
                //.withEncryption(new Encryption())
        );
        try {
            CreateTranscodingTaskResponse response = initMpcClient().createTranscodingTask(request);
            System.out.println("CreateTranscodingTaskResponse=" + response);
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.2.2 Canceling a Transcoding Task

Notes

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

Configuring Parameters

```
// Create a request for canceling a task. Use the task ID returned in the transcoding response.
DeleteTranscodingTaskRequest req = new DeleteTranscodingTaskRequest().withTaskId(3273178);
// Send the request.
DeleteTranscodingTaskResponse deleteTranscodingTaskResponse =
        initMpcClient().deleteTranscodingTask(req);
// Return a handle message.
System.out.println(JsonUtils.toJSON(deleteTranscodingTaskResponse));
```

Sample Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
```



```
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteTranscodingTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteTranscodingTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteTranscode {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Cancel a task in the queue.
     */
    public static void main(String[] args) {
        try {
            // Create a request for canceling a task. Use the task ID returned in the transcoding response.
            DeleteTranscodingTaskRequest req = new DeleteTranscodingTaskRequest().withTaskId(3273178);
            // Send the request.
            DeleteTranscodingTaskResponse deleteTranscodingTaskResponse =
                    initMpcClient().deleteTranscodingTask(req);
            // Return a handle message.
            System.out.println(JsonUtils.toJSON(deleteTranscodingTaskResponse));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.2.3 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.

If there are more than 10 records and the page number and maximum number of records on each page are not specified, 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

```
// Create a request for querying a task by task ID. Use the task ID returned in the transcoding response.
ListTranscodingTaskRequest req = new
ListTranscodingTaskRequest().withTaskId(Collections.singletonList(3273178L));
// Send the request.
ListTranscodingTaskResponse listTranscodingTaskResponse = initMpcClient().listTranscodingTask(req);
System.out.println(JsonUtils.toJSON(listTranscodingTaskResponse));
```

Querying Transcoding Tasks

```
// Create a request for querying tasks by task IDs. Use the task IDs returned in the transcoding responses.
ListTranscodingTaskRequest req = new ListTranscodingTaskRequest().withTaskId(Arrays.asList(3273178L,
3273179L));
// Send the request.
ListTranscodingTaskResponse listTranscodingTaskResponse = initMpcClient().listTranscodingTask(req);
System.out.println(JsonUtils.toJSON(listTranscodingTaskResponse));
```

Querying a Task by Status

```
// Create a request for querying tasks by status.
ListTranscodingTaskRequest req = new ListTranscodingTaskRequest().withStatus("FAILED");
// Send the request.
ListTranscodingTaskResponse listTranscodingTaskResponse = initMpcClient().listTranscodingTask(req);
System.out.println(JsonUtils.toJSON(listTranscodingTaskResponse));
```

Querying a Task by Start Time and End Time

```
// Create a request for querying a task by start time and end time.
ListTranscodingTaskRequest req = new
ListTranscodingTaskRequest().withStartTime("20210401001517").withEndTime("20210402081517");
// Send the request.
ListTranscodingTaskResponse listTranscodingTaskResponse = initMpcClient().listTranscodingTask(req);
System.out.println(JsonUtils.toJSON(listTranscodingTaskResponse));
```

Querying a Task by Page Number

```
// Create a request for querying a task by the page number and number of records on each page.
ListTranscodingTaskRequest req = new ListTranscodingTaskRequest().withPage(0).withSize(4);
// Send the request.
ListTranscodingTaskResponse listTranscodingTaskResponse = initMpcClient().listTranscodingTask(req);
System.out.println(JsonUtils.toJSON(listTranscodingTaskResponse));
```

Sample Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListTranscodingTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListTranscodingTaskResponse;

public class TestListTranscode {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
```



```
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Query a transcoding task.
     */
    public static void main(String[] args) {
        try {
            // Create a request for querying a task by task ID. Use the task ID returned in the transcoding response.
            ListTranscodingTaskRequest req = new ListTranscodingTaskRequest().withTaskId(Collections.singletonList(3273178L));
            // Send the request.
            ListTranscodingTaskResponse listTranscodingTaskResponse = initMpcClient().listTranscodingTask(req);
            System.out.println(JsonUtils.toJSON(listTranscodingTaskResponse));
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

2.3 Snapshot Capturing

2.3.1 Creating a Snapshot Task

Prerequisites

- You have purchased OBS and uploaded source videos for media processing in the region of MPC (for example, CN North-Beijing4) by referring to **Uploading Media Files**.
- You have obtained the authorization for MPC to access cloud resources by following the instructions provided in **Authorizing Access to Cloud Resources**.

Core Code

1. **Create a snapshot request.**

   The request includes paths of the input and output files. For details about the parameters, see **Creating a Snapshot Task**.

```
// Set the input file path.
ObsObjInfo input = new ObsObjInfo().withBucket("mpc-east-2").withLocation("region01").withObject("ok.mp4");
// Set the output file path.
ObsObjInfo output = new ObsObjInfo().withBucket("mpc-east-2").withLocation("region01").withObject("output");
// Create a snapshot request.
CreateThumbnailsTaskRequest req = new CreateThumbnailsTaskRequest()
        .withBody(new CreateThumbReq().withInput(input).withOutput(output)
                // Set the snapshot capturing type. Snapshots are captured by interval.
                .withThumbnailPara(new ThumbnailPara().withType(ThumbnailPara.TypeEnum.DOTS)
```



```
                // Set the snapshot file name.
                .withOutputFilename("photo")
                // Set the interval for capturing snapshots.
                .withDots(Collections.singletonList(2))
                // Set the width of snapshots.
                .withWidth(480)
                // Set the height of snapshots.
                .withHeight(360)));
```

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

2. **Send the snapshot request and return a message.**

```
CreateThumbnailsTaskResponse rsp = initMpcClient().createThumbnailsTask(req);
System.out.println("CreateThumbnailsTaskResponse=" + JsonUtils.toJSON(rsp));
```

Sample Code

```
package SdkTestCase.thumbnail;

import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CreateThumbReq;
import com.huaweicloud.sdk.mpc.v1.model.CreateThumbnailsTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateThumbnailsTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.ObsObjInfo;
import com.huaweicloud.sdk.mpc.v1.model.ThumbnailPara;
import com.obs.services.internal.ServiceException;

import java.util.Collections;

public class TestThumbnail {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create a snapshot task.
     * @param args
```



```
    */
    public static void main(String[] args) {
        // Set the input file path.
        ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.mp4");
        // Set the output file path.
        ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
        // Create a snapshot request.
        CreateThumbnailsTaskRequest req = new CreateThumbnailsTaskRequest()
                .withBody(new CreateThumbReq().withInput(input).withOutput(output)
                        // Set the snapshot capturing type. Snapshots are captured by interval.
                        .withThumbnailPara(new ThumbnailPara().withType(ThumbnailPara.TypeEnum.DOTS)
                                // Set the snapshot file name.
                                .withOutputFilename("photo")
                                // Set the interval for capturing snapshots.
                                .withDots(Collections.singletonList(2))
                                // Set the width of snapshots.
                                .withWidth(480)
                                // Set the height of snapshots.
                                .withHeight(360)));

        // Send the request.
        try {
            CreateThumbnailsTaskResponse rsp = initMpcClient().createThumbnailsTask(req);
            System.out.println("CreateThumbnailsTaskResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.3.2 Canceling a Snapshot Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be in the task queue. Ongoing or completed snapshot tasks cannot be canceled.

Core Code

```
// Send a request to MPC.
DeleteThumbnailsTaskRequest req = new DeleteThumbnailsTaskRequest().withTaskId("2210744");
DeleteThumbnailsTaskResponse rsp = initMpcClient().deleteThumbnailsTask(req);
// Return a message.
System.out.println("DeleteThumbnailsTaskResponse=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteThumbnailsTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteThumbnailsTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteThumbnail {
    /**
     * Initialize the MPC client.
     * @return
```



```
    */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Cancel a snapshot task.
     * @param args
     */
    public static void main(String[] args) {
        DeleteThumbnailsTaskRequest req = new DeleteThumbnailsTaskRequest().withTaskId("2210744");
        try {
            DeleteThumbnailsTaskResponse rsp = initMpcClient().deleteThumbnailsTask(req);
            System.out.println("DeleteThumbnailsTaskResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.3.3 Querying Snapshot Tasks

Notes

- You can query snapshot tasks by task ID, task status, time range, or page number, or perform compound query.
- If there are more than 10 records and the page number and maximum number of records on each page are not specified, 10 records are displayed by default on each page.

By Task ID

```
// You can query up to 10 tasks.
ListThumbnailsTaskRequest req = new
ListThumbnailsTaskRequest().withTaskId(Collections.singletonList("2210744"));
// Send the request to MPC.
ListThumbnailsTaskResponse rsp = initMpcClient().listThumbnailsTask(req);
// Return a message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

By Page Number

```
// Set the page number and number of records on each page.
ListThumbnailsTaskRequest req = new ListThumbnailsTaskRequest().withPage(1).withSize(4);
// Send the request to MPC.
```



```
ListThumbnailsTaskResponse rsp = initMpcClient().listThumbnailsTask(req);
// Return a message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

By Time Range

```
ListThumbnailsTaskRequest req = new
ListThumbnailsTaskRequest().withStartTime("20201220131400").withEndTime("20201220131400");
// Send the request to MPC.
ListThumbnailsTaskResponse rsp = initMpcClient().listThumbnailsTask(req);
// Return a message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

By Task Status

```
// Create a request for querying a task by task status.
ListThumbnailsTaskRequest req = new
ListThumbnailsTaskRequest().withStatus(ListThumbnailsTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListThumbnailsTaskResponse rsp = initMpcClient().listThumbnailsTask(req);
// Return a message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

Compound Query

```
// Configure the following parameters:
ListThumbnailsTaskRequest req = new ListThumbnailsTaskRequest().withPage(1).withSize(4)
        .withStartTime("20201220131400")
        .withEndTime("20201220131400")
        .withStatus(ListThumbnailsTaskRequest.StatusEnum.FAILED);
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListThumbnailsTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListThumbnailsTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestListThumbnail {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
```



```
            .withEndpoint(endpoint)
            .build();
    }

    /**
     * Query snapshot tasks.
     * @param args
     */
    public static void main(String[] args) {
        ListThumbnailsTaskRequest req = new ListThumbnailsTaskRequest().withPage(1).withSize(4)
                .withStartTime("20201220131400")
                .withEndTime("20201220131400")
                .withStatus(ListThumbnailsTaskRequest.StatusEnum.FAILED);
        try {
            ListThumbnailsTaskResponse rsp = initMpcClient().listThumbnailsTask(req);
            System.out.println("rsp=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.4 Encryption

2.4.1 Creating an Encryption Task

You can create an encryption task by creating an MPC client instance and configuring related parameters.

Prerequisites

- You have purchased OBS and uploaded source videos for media processing in the region of MPC (for example, CN North-Beijing4) by referring to **Uploading Media Files**.
- You have obtained the authorization for MPC to access cloud resources by following the instructions provided in **Authorizing Access to Cloud Resources**.

Core Code

1. **Create an encryption request.**

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

```
// Set the paths of the input and output files.
ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("input/hls/index.m3u8");
ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
// Create a request.
CreateEncryptTaskRequest req = new CreateEncryptTaskRequest()
        .withBody(new CreateEncryptReq().withInput(input).withOutput(output)
                .withEncryption(new Encryption().withHlsEncrypt(new HlsEncrypt()
                        // Set the encryption algorithm.
                        .withAlgorithm("AES-128-CBC")
                        // Set the URL to obtain the key.
                        .withUrl("www.xxxxx.com")
                        // Set the initialization vector.
                        .withIv("xxxxxxxxxxxxxxxxxx")
                        // Set the key.
```



```
                .withKey("xxxxxxxxxxxxxxxxxxxxxx"))));
        // Send the request to MPC.
        CreateEncryptTaskResponse rsp = initMpcClient().createEncryptTask(req);
        // Print the response message.
        System.out.println("CreateEncryptTaskResponse=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CreateEncryptReq;
import com.huaweicloud.sdk.mpc.v1.model.CreateEncryptTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateEncryptTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.Encryption;
import com.huaweicloud.sdk.mpc.v1.model.HlsEncrypt;
import com.huaweicloud.sdk.mpc.v1.model.ObsObjInfo;
import com.obs.services.internal.ServiceException;

public class TestEncrypt {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create an encryption task.
     * @param args
     */
    public static void main(String[] args) {
        // Set the paths of the input and output files.
        ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("input/hls/index.m3u8");
        ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
        // Create a request.
        CreateEncryptTaskRequest req = new CreateEncryptTaskRequest()
                .withBody(new CreateEncryptReq().withInput(input).withOutput(output)
                        .withEncryption(new Encryption().withHlsEncrypt(new HlsEncrypt()
                                // Set the encryption algorithm.
                                .withAlgorithm("AES-128-CBC")
                                // Set the URL to obtain the key.
                                .withUrl("www.xxxxx.com")
```



```
                        // Set the initialization vector.
                        .withIv("xxxxxxxxxxxxxxxxxx")
                        // Set the key.
                        .withKey("xxxxxxxxxxxxxxxxxxxxxx"))));
        // Send the encryption request.
        try {
            CreateEncryptTaskResponse rsp = initMpcClient().createEncryptTask(req);
            System.out.println("CreateEncryptTaskResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.4.2 Canceling an Encryption Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be in the task queue. Ongoing or completed encryption tasks cannot be canceled.

Core Code

```
// Send a request to MPC.
DeleteEncryptTaskRequest req = new DeleteEncryptTaskRequest().withTaskId("3223179");
DeleteEncryptTaskResponse rsp = initMpcClient().deleteEncryptTask(req);
// Print the response message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteEncryptTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteEncryptTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteEncrypt {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
```



```
            .withHttpConfig(httpConfig)
            .withCredential(auth)
            .withEndpoint(endpoint)
            .build();
    }

    /**
     * Cancel a task in the queue.
     * @param args
     */
    public static void main(String[] args) {
        DeleteEncryptTaskRequest req = new DeleteEncryptTaskRequest().withTaskId("3223179");
        try {
            DeleteEncryptTaskResponse rsp = initMpcClient().deleteEncryptTask(req);
            System.out.println("rsp=" + JsonUtils.toJSON(rsp));
            System.out.println(rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.4.3 Querying Encryption Tasks

Notes

- You can query encryption tasks by task ID, task status, time range, or page number, or perform compound query.
- If there are more than 10 records and the page number and maximum number of records on each page are not specified, 10 records are displayed by default on each page.

By Task ID

```
// You can query up to 10 tasks.
ListEncryptTaskRequest req = new
ListEncryptTaskRequest().withTaskId(Collections.singletonList("3223179"));
// Send the request to MPC.
ListEncryptTaskResponse rsp = initMpcClient().listEncryptTask(req);
// Print the response message.
System.out.println(rsp.toString());
```

By Page Number

```
// Set the page number and number of records on each page.
ListEncryptTaskRequest req = new ListEncryptTaskRequest().withPage(1).withSize(4);
// Send the request to MPC.
ListEncryptTaskResponse rsp = initMpcClient().listEncryptTask(req);
// Print the response message.
System.out.println(rsp.toString());
```

By Time Range

```
// Set the start time and end time.
ListEncryptTaskRequest req = new
ListEncryptTaskRequest().withStartTime("20201220131400").withEndTime("20201221131400");
// Send the request to MPC.
ListEncryptTaskResponse rsp = initMpcClient().listEncryptTask(req);
// Print the response message.
System.out.println(rsp.toString());
```


By Task Status

```
// Create a request for querying a task by task status.
ListEncryptTaskRequest req = new
ListEncryptTaskRequest().withStatus(ListEncryptTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListEncryptTaskResponse rsp = initMpcClient().listEncryptTask(req);
// Print the response message.
System.out.println(rsp.toString());
```

Compound Query

```
// Configure the following parameters:
ListEncryptTaskRequest req = new ListEncryptTaskRequest().withPage(1).withSize(4)
        .withStartTime("20201220131400").withEndTime("20201221131400")
        .withStatus(ListEncryptTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListEncryptTaskResponse rsp = initMpcClient().listEncryptTask(req);
// Print the response message.
System.out.println(rsp.toString());
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListEncryptTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListEncryptTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestListEncrypt {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Query encryption tasks.
     * @param args
     */
    public static void main(String[] args) {
        ListEncryptTaskRequest req = new ListEncryptTaskRequest().withPage(1).withSize(4)
                .withStartTime("20201220131400").withEndTime("20201221131400")
                .withStatus(ListEncryptTaskRequest.StatusEnum.FAILED);
```



```
        try {
            ListEncryptTaskResponse rsp = initMpcClient().listEncryptTask(req);
            System.out.println(rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.5 Animated GIFs

2.5.1 Creating an Animated GIF Task

You can create an MPC client instance and configure related parameters to create an animated GIF task, which is used to convert a video to an animated GIF.

Prerequisites

- You have purchased OBS and uploaded source videos for media processing in the region of MPC (for example, CN North-Beijing4) by referring to **Uploading Media Files**.
- You have obtained the authorization for MPC to access cloud resources by following the instructions provided in **Authorizing Access to Cloud Resources**.

Core Code

1. **Create an animated GIF task.**

   Configure parameters such as the paths of the input and output files, as well as frame rate and width and height of the output file.

```
// Set the paths of the input and output files.
ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.mp4");
ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
// Create an animated GIF request.
CreateAnimatedGraphicsTaskRequest req = new CreateAnimatedGraphicsTaskRequest()
        .withBody(new CreateAnimatedGraphicsTaskReq().withInput(input).withOutput(output)
                .withOutputParam(new AnimatedGraphicsOutputParam()
                        // Set the output image format.
                        .withFormat(AnimatedGraphicsOutputParam.FormatEnum.GIF)
                        // Set the output image frame rate.
                        .withFrameRate(15)
                        // Set the start time, in milliseconds.
                        .withStart(0)
                        // Set the end time, in milliseconds. The maximum interval is 60 seconds.
                        .withEnd(3_000)));

// Send the request.
CreateAnimatedGraphicsTaskResponse rsp = initMpcClient().createAnimatedGraphicsTask(req);
// Print the result.
System.out.println("CreateAnimatedGraphicsTaskResponse=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
```



```
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.AnimatedGraphicsOutputParam;
import com.huaweicloud.sdk.mpc.v1.model.CreateAnimatedGraphicsTaskReq;
import com.huaweicloud.sdk.mpc.v1.model.CreateAnimatedGraphicsTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateAnimatedGraphicsTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.ObsObjInfo;
import com.obs.services.internal.ServiceException;

public class TestAnimation {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create an animated GIF task.
     * @param args
     */
    public static void main(String[] args) {
        // Set the paths of the input and output files.
        ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.mp4");
        ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
        // Create an animated GIF request.
        CreateAnimatedGraphicsTaskRequest req = new CreateAnimatedGraphicsTaskRequest()
                .withBody(new CreateAnimatedGraphicsTaskReq().withInput(input).withOutput(output)
                        .withOutputParam(new AnimatedGraphicsOutputParam()
                                // Set the output image format.
                                .withFormat(AnimatedGraphicsOutputParam.FormatEnum.GIF)
                                // Set the output image frame rate.
                                .withFrameRate(15)
                                // Set the start time, in milliseconds.
                                .withStart(0)
                                // Set the end time, in milliseconds. The maximum interval is 60 seconds.
                                .withEnd(3_000)));
        try {
            CreateAnimatedGraphicsTaskResponse rsp = initMpcClient().createAnimatedGraphicsTask(req);
            System.out.println("CreateAnimatedGraphicsTaskResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```


2.5.2 Querying Animated GIF Tasks

Notes

- You can query animated GIF tasks by task ID, task status, time range, or page number, or perform compound query.
- If there are more than 10 records and the page number and maximum number of records on each page are not specified, 10 records are displayed by default on each page.

By Task ID

```
// You can query up to 10 tasks.
ListAnimatedGraphicsTaskRequest req = new
ListAnimatedGraphicsTaskRequest().withTaskId(Collections.singletonList("3198527"));
// Send the request to MPC.
ListAnimatedGraphicsTaskResponse rsp = initMpcClient().listAnimatedGraphicsTask(req);
// Print the response message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

By Page Number

```
// Set the page number and number of records on each page.
ListAnimatedGraphicsTaskRequest req = new ListAnimatedGraphicsTaskRequest().withPage(1).withSize(10);
// Send the request to MPC.
ListAnimatedGraphicsTaskResponse rsp = initMpcClient().listAnimatedGraphicsTask(req);
// Print the response message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

By Time Range

```
// Set the start time and end time.
ListAnimatedGraphicsTaskRequest req = new
ListAnimatedGraphicsTaskRequest().withStartTime("20201220131400").withEndTime("20201221131400");
// Send the request to MPC.
ListAnimatedGraphicsTaskResponse rsp = initMpcClient().listAnimatedGraphicsTask(req);
// Print the response message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

By Task Status

```
// Create a request for querying a task by task status.
ListAnimatedGraphicsTaskRequest req = new
ListAnimatedGraphicsTaskRequest().withStatus(ListAnimatedGraphicsTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListAnimatedGraphicsTaskResponse rsp = initMpcClient().listAnimatedGraphicsTask(req);
// Print the response message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

Compound Query

```
// Configure the following parameters:
ListAnimatedGraphicsTaskRequest req = new ListAnimatedGraphicsTaskRequest().withPage(0).withSize(10)
        .withStartTime("20201220131400").withEndTime("20201221131400")
        .withStatus(ListAnimatedGraphicsTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListAnimatedGraphicsTaskResponse rsp = initMpcClient().listAnimatedGraphicsTask(req);
// Print the response message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

Full Code

```
package com.huawei.mpc;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
```



```
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListAnimatedGraphicsTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListAnimatedGraphicsTaskResponse;
import com.obs.services.internal.ServiceException;

import java.util.Collections;

public class TestListAnimation {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Query animated GIF tasks.
     * @param args
     */
    public static void main(String[] args) {
        // Query tasks by task ID, which is returned in animated GIF task responses.
        ListAnimatedGraphicsTaskRequest req = new
ListAnimatedGraphicsTaskRequest().withTaskId(Collections.singletonList("3198527"));
        try {
            ListAnimatedGraphicsTaskResponse rsp = initMpcClient().listAnimatedGraphicsTask(req);
            System.out.println("rsp=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.5.3 Canceling an Animated GIF Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be in the task queue. Ongoing or completed animated GIF tasks cannot be canceled.

Core Code

```
DeleteAnimatedGraphicsTaskRequest req = new
DeleteAnimatedGraphicsTaskRequest().withTaskId("3198527");
DeleteAnimatedGraphicsTaskResponse rsp = initMpcClient().deleteAnimatedGraphicsTask(req);
System.out.println("rsp=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteAnimatedGraphicsTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteAnimatedGraphicsTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteAnimation {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Cancel a task in the queue.
     * @param args
     */
    public static void main(String[] args) {
        DeleteAnimatedGraphicsTaskRequest req = new
DeleteAnimatedGraphicsTaskRequest().withTaskId("3198527");
        try {
            DeleteAnimatedGraphicsTaskResponse rsp = initMpcClient().deleteAnimatedGraphicsTask(req);
            System.out.println("rsp=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.6 Video Parsing

2.6.1 Creating a Video Parsing Task

You can create a task to parse video metadata by creating an MPC client instance and configuring related parameters.

Prerequisites

- You have purchased OBS and uploaded source videos for media processing in the region of MPC (for example, CN North-Beijing4) by referring to **Uploading Media Files**.
- You have obtained the authorization for MPC to access cloud resources by following the instructions provided in **Authorizing Access to Cloud Resources**.

Core Code

1. **Create a video parsing task.**

   Configure input video file parameters. If necessary, you can save the metadata file in a specified path.

```
// Set the paths of the input and output files.
ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.mp4");
ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
// Create a video parsing request.
CreateExtractTaskRequest req = new CreateExtractTaskRequest()
        .withBody(new CreateExtractTaskReq().withInput(input));
// Send the request.
CreateExtractTaskResponse rsp = initMpcClient().createExtractTask(req);
// Print the result.
System.out.println("CreateExtractTaskResponse=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CreateExtractTaskReq;
import com.huaweicloud.sdk.mpc.v1.model.CreateExtractTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateExtractTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.ObsObjInfo;
import com.obs.services.internal.ServiceException;

public class TestParse {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
```



```
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create a video parsing task.
     * @param args
     */
    public static void main(String[] args) {
        // Set the paths of the input and output files.
        ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.mp4");
        ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
        // Create a video parsing request.
        CreateExtractTaskRequest req = new CreateExtractTaskRequest()
                .withBody(new CreateExtractTaskReq().withInput(input));

        // Send the request.
        try {
            CreateExtractTaskResponse rsp = initMpcClient().createExtractTask(req);
            System.out.println("CreateExtractTaskResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.6.2 Querying Video Parsing Tasks

Notes

- You can query video parsing tasks by task ID, task status, time range, or page number, or perform compound query.
- If there are more than 10 records and the page number and maximum number of records on each page are not specified, 10 records are displayed by default on each page.

By Task ID

```
// You can query up to 10 tasks.
ListExtractTaskRequest req = new ListExtractTaskRequest().withTaskId(Collections.singletonList("3223182"));
// Send the request to MPC.
ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

By Page Number

```
// Set the page number and number of records on each page.
ListExtractTaskRequest req = new ListExtractTaskRequest().withPage(0).withSize(10);
// Send the request to MPC.
ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```


By Time Range

```
// Set the start time and end time.
ListExtractTaskRequest req = new
ListExtractTaskRequest().withStartTime("20201220131400").withEndTime("20201221131400");
// Send the request to MPC.
ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

By Task Status

```
// Create a request for querying a task by task status.
ListExtractTaskRequest req = new
ListExtractTaskRequest().withStatus(ListExtractTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

Compound Query

```
// Configure the following parameters:
ListExtractTaskRequest req = new ListExtractTaskRequest().withPage(0).withSize(10)
        .withStartTime("20201220131400").withEndTime("20201221131400")
        .withStatus(ListExtractTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListExtractTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListExtractTaskResponse;
import com.obs.services.internal.ServiceException;

import java.util.Collections;

public class TestListParse {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
```



```
            .withCredential(auth)
            .withEndpoint(endpoint)
            .build();
    }

    /**
     * Query video parsing tasks.
     * @param args
     */
    public static void main(String[] args) {
        ListExtractTaskRequest req = new
ListExtractTaskRequest().withTaskId(Collections.singletonList("3223182"));
        try {
            ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
            System.out.println("rsp=" + rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.6.3 Canceling a Video Parsing Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be in the task queue. Ongoing or completed video parsing tasks cannot be canceled.

Core Code

```
// Set the ID of the task to be canceled.
DeleteExtractTaskRequest req = new DeleteExtractTaskRequest().withTaskId("3223182");
// Send a message to MPC.
DeleteExtractTaskResponse rsp = initMpcClient().deleteExtractTask(req);
System.out.println("rsp=" + rsp.toString());
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteExtractTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteExtractTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteParse {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
```

```
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Cancel a task in the queue.
     * @param args
     */
    public static void main(String[] args) {
        DeleteExtractTaskRequest req = new DeleteExtractTaskRequest().withTaskId("3223182");
        try {
            DeleteExtractTaskResponse rsp = initMpcClient().deleteExtractTask(req);
            System.out.println("rsp=" + rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.7 Packaging

2.7.1 Creating a Packaging Task

You can create a video packaging task by creating an MPC client instance and configuring related parameters.

Prerequisites

- You have purchased OBS and uploaded source videos for media processing in the region of MPC (for example, CN North-Beijing4) by referring to **Uploading Media Files**.
- You have obtained the authorization for MPC to access cloud resources by following the instructions provided in **Authorizing Access to Cloud Resources**.

Core Code

1. **Create a packaging task.**

```
// Set the paths of the input and output files.
ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.flv");
ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
// Create a packaging request.
CreateRemuxTaskRequest req = new CreateRemuxTaskRequest()
        .withBody(new CreateRemuxTaskReq().withInput(input).withOutput(output)
                // Configure packaging parameters.
                .withOutputParam(new RemuxOutputParam()
                        // Set the packaging format.
                        .withFormat("HLS")
                        // Set the HLS segment interval.
                        .withSegmentDuration(5)));
```



```
        // Send the packaging request.
        CreateRemuxTaskResponse rsp = initMpcClient().createRemuxTask(req);
        System.out.println(rsp.toString())
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CreateRemuxTaskReq;
import com.huaweicloud.sdk.mpc.v1.model.CreateRemuxTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateRemuxTaskResponse;
import com.huaweicloud.sdk.mpc.v1.model.ObsObjInfo;
import com.huaweicloud.sdk.mpc.v1.model.RemuxOutputParam;
import com.obs.services.internal.ServiceException;

public class TestRemux {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create a packaging task.
     * @param args
     */
    public static void main(String[] args) {
        // Set the paths of the input and output files.
        ObsObjInfo input = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("ok.flv");
        ObsObjInfo output = new ObsObjInfo().withBucket("mpc-
east-2").withLocation("region01").withObject("output");
        // Create a packaging request.
        CreateRemuxTaskRequest req = new CreateRemuxTaskRequest()
                .withBody(new CreateRemuxTaskReq().withInput(input).withOutput(output)
                        // Configure packaging parameters.
                        .withOutputParam(new RemuxOutputParam()
                                // Set the packaging format.
                                .withFormat("HLS")
                                // Set the HLS segment interval.
                                .withSegmentDuration(5)));
        // Send the packaging request.
        try {
            CreateRemuxTaskResponse rsp = initMpcClient().createRemuxTask(req);
```



```
            System.out.println(rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.7.2 Querying Packaging Tasks

Notes

- You can query packaging tasks by task ID, task status, time range, or page number, or perform compound query.
- If there are more than 10 records and the page number and maximum number of records on each page are not specified, 10 records are displayed by default on each page.

By Task ID

```
// Create a request for querying packaging tasks.
ListRemuxTaskRequest req = new ListRemuxTaskRequest().withTaskId(Collections.singletonList("8191203"));
// Send the request to MPC.
ListRemuxTaskResponse rsp = initMpcClient().listRemuxTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

By Page Number

```
// Set the page number and number of records on each page.
ListRemuxTaskRequest req = new ListRemuxTaskRequest().withPage(0).withSize(10);
// Send the request to MPC.
ListRemuxTaskResponse rsp = initMpcClient().listRemuxTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

By Time Range

```
// Set the start time and end time.
ListRemuxTaskRequest req = new
ListRemuxTaskRequest().withStartTime("20201220131400").withEndTime("20201221131400");
// Send the request to MPC.
ListRemuxTaskResponse rsp = initMpcClient().listRemuxTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

By Task Status

```
// Create a request for querying a task by task status.
ListRemuxTaskRequest req = new
ListRemuxTaskRequest().withStatus(ListRemuxTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
ListRemuxTaskResponse rsp = initMpcClient().listRemuxTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

Compound Query

```
// Configure the following parameters:
ListRemuxTaskRequest req = new ListRemuxTaskRequest().withPage(0).withSize(10)
        .withStartTime("20201220131400").withEndTime("20201221131400")
        .withStatus(ListRemuxTaskRequest.StatusEnum.FAILED);
// Send the request to MPC.
```



```
ListExtractTaskResponse rsp = initMpcClient().listExtractTask(req);
// Print the response message.
System.out.println("rsp=" + rsp.toString());
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListRemuxTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListRemuxTaskResponse;
import com.obs.services.internal.ServiceException;

import java.util.Collections;

public class TestListRemux {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Query packaging tasks.
     * @param args
     */
    public static void main(String[] args) {
        ListRemuxTaskRequest req = new
ListRemuxTaskRequest().withTaskId(Collections.singletonList("8191203"));
        try {
            ListRemuxTaskResponse rsp = initMpcClient().listRemuxTask(req);
            System.out.println("rsp=" + rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```


2.7.3 Canceling a Packaging Task

Notes

- To cancel a task, you need to provide the task ID.
- The task to be canceled must be in the task queue. Ongoing or completed packaging tasks cannot be canceled.

Core Code

```
// Set the ID of the task to be canceled.
CancelRemuxTaskRequest req = new CancelRemuxTaskRequest().withTaskId("8191203");
// Send a message to MPC.
CancelRemuxTaskResponse rsp = initMpcClient().cancelRemuxTask(req);
System.out.println("rsp=" + rsp.toString());
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CancelRemuxTaskRequest;
import com.huaweicloud.sdk.mpc.v1.model.CancelRemuxTaskResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteRemux {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Cancel a task in the queue.
     * @param args
     */
    public static void main(String[] args) {
        CancelRemuxTaskRequest req = new CancelRemuxTaskRequest().withTaskId("8191203");
        try {
            CancelRemuxTaskResponse rsp = initMpcClient().cancelRemuxTask(req);
            System.out.println("rsp=" + rsp.toString());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
```



```
e) {
            System.out.println(e);
        }
    }
}
```

2.8 Transcoding Templates

2.8.1 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. Set template parameters.

```
// Set a request for creating a transcoding template.
CreateTransTemplateRequest req = new CreateTransTemplateRequest()
        .withBody(new TransTemplate().withTemplateName("test_123")
// Set video parameters.
                .withVideo(new Video()
                        // Video encoding format. The value 1 indicates H.264, and the value 2 indicates
H.265.
                        .withCodec(1)
                        // Video bitrate (kbit/s)
                        .withBitrate(6000)
                        // Encoding level. The recommended value is 3.
                        .withProfile(3)
                        .withLevel(15)
                        // Encoding quality. A larger value indicates higher quality and longer duration.
                        .withPreset(3)
                        .withRefFramesCount(4)
                        .withMaxIframesInterval(5)
                        .withBframesCount(4)
                        .withHeight(1080)
                        .withWidth(1920))
                // Set audio parameters.
                .withAudio(new Audio()
                        // Audio encoding format. The value can be 1 (AAC), 2 (HEAAC1), 3 (HEAAC2), or 4
(MP3).
                        .withCodec(1)
                        // Sampling rate. The value can be 1 (AUDIO_SAMPLE_AUTO), 2 (22,050 Hz), 3
(32,000 Hz), 4 (44,100 Hz), 5 (48,000 Hz), or 6 (96,000 Hz).
                        .withSampleRate(4)
                        // Audio bitrate (kbit/s)
                        .withBitrate(128)
                        // Number of audio channels.
                        .withChannels(2))
                // Set common parameters.
                .withCommon(new Common()
                        .withDashInterval(5)
                        .withHlsInterval(5)
                        // Whether to enable low bitrate HD.
                        .withPvc(false)
                        // Pack type. The value can be 1 (HLS), 2 (DASH), 3 (HLS+DASH), 4 (MP4), 5 (MP3),
or 6 (ADTS).
                        .withPackType(1)));
```

2. Send the request for creating a transcoding template and return a message.

```
// Send the request.
CreateTransTemplateResponse rsp = initMpcClient().createTransTemplate(req);
// Print the response message.
System.out.println("CreateTransTemplateResponse=" + JsonUtils.toJSON(rsp));
```


Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.Audio;
import com.huaweicloud.sdk.mpc.v1.model.Common;
import com.huaweicloud.sdk.mpc.v1.model.CreateTransTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateTransTemplateResponse;
import com.huaweicloud.sdk.mpc.v1.model.TransTemplate;
import com.huaweicloud.sdk.mpc.v1.model.Video;
import com.obs.services.internal.ServiceException;

public class TestTranscodeTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Create a transcoding template.
     * @param args
     */
    public static void main(String[] args) {
        // Create a transcoding template request.
        CreateTransTemplateRequest req = new CreateTransTemplateRequest()
                .withBody(new TransTemplate().withTemplateName("test_123")
                        // Set video parameters.
                        .withVideo(new Video()
                                // Video encoding format. The value 1 indicates H.264, and the value 2 indicates
H.265.
                                .withCodec(1)
                                // Video bitrate (kbit/s)
                                .withBitrate(6000)
                                // Encoding level. The recommended value is 3.
                                .withProfile(3)
                                .withLevel(15)
                                // Encoding quality. A larger value indicates higher quality and longer duration.
                                .withPreset(3)
                                .withRefFramesCount(4)
                                .withMaxIframesInterval(5)
                                .withBframesCount(4)
                                .withHeight(1080)
                                .withWidth(1920))
```



```
                    // Set audio parameters.
                    .withAudio(new Audio()
                            // Audio encoding format. The value can be 1 (AAC), 2 (HEAAC1), 3 (HEAAC2), or 4
(MP3).
                            .withCodec(1)
                            // Sampling rate. The value can be 1 (AUDIO_SAMPLE_AUTO), 2 (22,050 Hz), 3
(32,000 Hz), 4 (44,100 Hz), 5 (48,000 Hz), or 6 (96,000 Hz).
                            .withSampleRate(4)
                            // Audio bitrate (kbit/s)
                            .withBitrate(128)
                            // Number of audio channels.
                            .withChannels(2))
                    // Set common parameters.
                    .withCommon(new Common()
                            .withDashInterval(5)
                            .withHlsInterval(5)
                            // Whether to enable low bitrate HD.
                            .withPvc(false)
                            // Pack type. The value can be 1 (HLS), 2 (DASH), 3 (HLS+DASH), 4 (MP4), 5 (MP3),
or 6 (ADTS).
                            .withPackType(1)));
        // Send a transcoding template request.
        try {
            CreateTransTemplateResponse rsp = initMpcClient().createTransTemplate(req);
            System.out.println("CreateTransTemplateResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.8.2 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.
DeleteTemplateRequest req = new DeleteTemplateRequest().withTemplateId(346090L);
DeleteTemplateResponse rsp = initMpcClient().deleteTemplate(req);
// Return a message.
System.out.println("rsp=" + JsonUtils.toJSON(rsp) + " httpCode=" + rsp.getHttpStatusCode());
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteTemplateResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteTranscodeTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
```



```
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Delete a transcoding template.
     * @param args
     */
    public static void main(String[] args) {
        // Set the ID of the transcoding template to be deleted.
        DeleteTemplateRequest req = new DeleteTemplateRequest().withTemplateId(346090L);
        try {
            DeleteTemplateResponse rsp = initMpcClient().deleteTemplate(req);
            System.out.println("rsp=" + JsonUtils.toJSON(rsp) + " httpCode=" + rsp.getHttpStatusCode());
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.8.3 Updating a Transcoding Template

You can reset template parameters to update a transcoding template.

Core Code

1. Set template parameters.

```
// Set a request for updating a transcoding template.
UpdateTransTemplateRequest req = new UpdateTransTemplateRequest()
        .withBody(new
ModifyTransTemplateReq().withTemplateName("test_123").withTemplateId(346090L)
                // Set video parameters.
                .withVideo(new Video()
                        // Video encoding format. The value 1 indicates H.264, and the value 2 indicates
H.265.
                        .withCodec(1)
                        // Video bitrate (kbit/s)
                        .withBitrate(3200)
                        // Encoding level. The recommended value is 3.
                        .withProfile(3)
                        .withLevel(15)
                        // Encoding quality. A larger value indicates higher quality and longer duration.
                        .withPreset(3)
                        .withRefFramesCount(4)
                        .withMaxIframesInterval(5)
                        .withBframesCount(4)
                        .withHeight(480)
                        .withWidth(720))
                // Set audio parameters.
                .withAudio(new Audio()
                        // Audio encoding format. The value can be 1 (AAC), 2 (HEAAC1), 3 (HEAAC2), or 4
(MP3).
                        .withCodec(1)
                        // Sampling rate. The value can be 1 (AUDIO_SAMPLE_AUTO), 2 (22,050 Hz), 3
```



```
(32,000 Hz), 4 (44,100 Hz), 5 (48,000 Hz), or 6 (96,000 Hz).
                        .withSampleRate(4)
                        // Audio bitrate (kbit/s)
                        .withBitrate(128)
                        // Number of audio channels.
                        .withChannels(2))
                // Set common parameters.
                .withCommon(new Common()
                        .withDashInterval(5)
                        .withHlsInterval(5)
                        // Whether to enable low bitrate HD.
                        .withPvc(false)
                        // Pack type. The value can be 1 (HLS), 2 (DASH), 3 (HLS+DASH), 4 (MP4), 5 (MP3),
or 6 (ADTS).
                        .withPackType(1)));
```

2. Send the request for updating a transcoding template and return a message.

```
// Send the request.
UpdateTransTemplateResponse rsp = initMpcClient().updateTransTemplate(req);
// Print the response parameters.
System.out.println("UpdateTransTemplateResponse=" + JsonUtils.toJSON(rsp));
```

Sample Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.Audio;
import com.huaweicloud.sdk.mpc.v1.model.Common;
import com.huaweicloud.sdk.mpc.v1.model.ModifyTransTemplateReq;
import com.huaweicloud.sdk.mpc.v1.model.UpdateTransTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.UpdateTransTemplateResponse;
import com.huaweicloud.sdk.mpc.v1.model.Video;
import com.obs.services.internal.ServiceException;

public class TestUpdateTranscodeTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Update a transcoding template.
     * @param args
```



```
    */
    public static void main(String[] args) {
        // Set the request for updating a transcoding template.
        UpdateTransTemplateRequest req = new UpdateTransTemplateRequest()
                .withBody(new
ModifyTransTemplateReq().withTemplateName("test_123").withTemplateId(346090L)
                        // Set video parameters.
                        .withVideo(new Video()
                                // Video encoding format. The value 1 indicates H.264, and the value 2 indicates
H.265.
                                .withCodec(1)
                                // Video bitrate (kbit/s)
                                .withBitrate(3200)
                                // Encoding level. The recommended value is 3.
                                .withProfile(3)
                                .withLevel(15)
                                // Encoding quality. A larger value indicates higher quality and longer duration.
                                .withPreset(3)
                                .withRefFramesCount(4)
                                .withMaxIframesInterval(5)
                                .withBframesCount(4)
                                .withHeight(480)
                                .withWidth(720))
                        // Set audio parameters.
                        .withAudio(new Audio()
                                // Audio encoding format. The value can be 1 (AAC), 2 (HEAAC1), 3 (HEAAC2), or 4
(MP3).
                                .withCodec(1)
                                // Sampling rate. The value can be 1 (AUDIO_SAMPLE_AUTO), 2 (22,050 Hz), 3
(32,000 Hz), 4 (44,100 Hz), 5 (48,000 Hz), or 6 (96,000 Hz).
                                .withSampleRate(4)
                                // Audio bitrate (kbit/s)
                                .withBitrate(128)
                                // Number of audio channels.
                                .withChannels(2))
                        // Set common parameters.
                        .withCommon(new Common()
                                .withDashInterval(5)
                                .withHlsInterval(5)
                                // Whether to enable low bitrate HD.
                                .withPvc(false)
                                // Pack type. The value can be 1 (HLS), 2 (DASH), 3 (HLS+DASH), 4 (MP4), 5 (MP3),
or 6 (ADTS).
                                .withPackType(1)));
        // Send the request for updating a transcoding template.
        try {
            UpdateTransTemplateResponse rsp = initMpcClient().updateTransTemplate(req);
            System.out.println("UpdateTransTemplateResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.8.4 Querying Transcoding Templates

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

Notes

- You can query one or more 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 the parameters for querying transcoding templates. A maximum of 10 templates can be queried.
ListTemplateRequest req = new ListTemplateRequest().withTemplateId(Collections.singletonList(346090));
// Send a request for querying transcoding templates.
ListTemplateResponse rsp = initMpcClient().listTemplate(req);
// Return the query results.
System.out.println("httpCode=" + rsp.getHttpStatusCode() + " rsp=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListTemplateResponse;
import com.obs.services.internal.ServiceException;

import java.util.Collections;

public class TestListTranscodeTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Query transcoding templates.
     * @param args
     */
    public static void main(String[] args) {
        ListTemplateRequest req = new
ListTemplateRequest().withTemplateId(Collections.singletonList(346090));
        try {
            ListTemplateResponse rsp = initMpcClient().listTemplate(req);
            System.out.println("httpCode=" + rsp.getHttpStatusCode() + " rsp=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
```



```
    }
  }
}
```

2.9 Watermarking

2.9.1 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. Set template parameters.

```
// Set a request for creating a watermark template.
CreateWatermarkTemplateRequest req = new CreateWatermarkTemplateRequest()
        .withBody(new WatermarkTemplate()
                // Set the template name.
                .withTemplateName("watermark_name")
                // Set the template type.
                .withType("Image")
                // Set the image watermark processing mode.
                .withImageProcess("Grayed")
                // Set the watermark width.
                .withWidth("1920")
                // Set the watermark height.
                .withHeight("1080")
                // Set the horizontal offset of the watermark relative to the video vertex.
                .withDx("10")
                // Set the vertical offset of the watermark relative to the video vertex.
                .withDy("10")
                // Set the watermark position.
                //.withReferpos("BottomLeft")
                // Set the start time of the watermark, which is used together with timeline_duration.
                .withTimelineStart("6")
                // Set the watermark duration. The default value is ToEND, indicating that the watermark persists until the video ends.
                .withTimelineDuration("8"));
```

2. Send the request for creating a watermark template and return a message.

```
// Send the request for creating a watermark template to MPC.
CreateWatermarkTemplateResponse rsp = initMpcClient().createWatermarkTemplate(req);
// Print the response message.
System.out.println("CreateWatermarkTemplateResponse=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.CreateWatermarkTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.CreateWatermarkTemplateResponse;
import com.huaweicloud.sdk.mpc.v1.model.WatermarkTemplate;

public class TestWatermarkTemplate {
    /**
     * Initialize the MPC client.
     *
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
```



```
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder().withHttpConfig(httpConfig).withCredential(auth).withEndpoint(endpoint).build();
    }

    /**
     * Create a watermark template.
     *
     * @param args
     */
    public static void main(String[] args) {
        // Set a request for creating a watermark template.
        CreateWatermarkTemplateRequest req = new CreateWatermarkTemplateRequest().withBody(new WatermarkTemplate()
                        // Set the template name.
                        .withTemplateName("watermark_name")
                        // Set the template type.
                        .withType("Image")
                        // Set the image watermark processing mode.
                        .withImageProcess("Grayed")
                        // Set the watermark width.
                        .withWidth("1920")
                        // Set the watermark height.
                        .withHeight("1080")
                        // Set the horizontal offset of the watermark relative to the video vertex.
                        .withDx("10")
                        // Set the vertical offset of the watermark relative to the video vertex.
                        .withDy("10")
                        // Set the watermark position.
                        //.withReferpos("BottomLeft")
                        // Set the start time of the watermark, which is used together with timeline_duration.
                        .withTimelineStart("6")
                        // Set the watermark duration. The default value is ToEND, indicating that the watermark persists until the video ends.
                        .withTimelineDuration("ToEND"));
        // Send a watermark template request.
        try {
            CreateWatermarkTemplateResponse rsp = initMpcClient().createWatermarkTemplate(req);
            System.out.println("CreateWatermarkTemplateResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException e) {
            System.out.println(e);
        }
    }
}
```

2.9.2 Updating a Watermark Template

Core Code

1. Set template parameters.

```
// Create a request for updating a watermark template.
UpdateWatermarkTemplateRequest req = new UpdateWatermarkTemplateRequest()
    .withBody(new WatermarkTemplate()
            // Set the template name.
            .withTemplateName("watermark_name")
```



```
            // Set the template type.
             .withType("Image")
            // Set the image watermark processing mode.
             .withImageProcess("Grayed")
            // Set the watermark width.
             .withWidth("1920")
            // Set the watermark height.
             .withHeight("1080")
            // Set the horizontal offset of the watermark relative to the video vertex.
             .withDx("10")
             // Set the vertical offset of the watermark relative to the video vertex.
             .withDy("10")
             // Set the watermark position.
             //.withReferpos("BottomLeft")
             // Set the start time of the watermark, which is used together with timeline_duration.
             .withTimelineStart("0")
             // Set the watermark duration. The default value is ToEND, indicating that the watermark persists until the video ends.
             .withTimelineDuration("ToEND"));
```

2. Send the request for updating a watermark template and return a message.

```
// Send the request for updating the watermark configuration to MPC.
UpdateWatermarkTemplateResponse rsp = initMpcClient().updateWatermarkTemplate(req);
// Print the response message.
System.out.println("UpdateWatermarkTemplateResponse=" + JsonUtils.toJSON(rsp));
```

Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.UpdateWatermarkTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.UpdateWatermarkTemplateResponse;
import com.huaweicloud.sdk.mpc.v1.model.WatermarkTemplate;

public class TestUpdateWatermarkTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
      * Update a watermark template.
     * @param args
     */
    public static void main(String[] args) {
        // Create a request for updating a watermark template.
```



```
        UpdateWatermarkTemplateRequest req = new UpdateWatermarkTemplateRequest()
                .withBody(new WatermarkTemplate()
                        // Set the template name.
                        .withTemplateName("watermark_name")
                        // Set the template type.
                        .withType("Image")
                        // Set the image watermark processing mode.
                        .withImageProcess("Grayed")
                        // Set the watermark width.
                        .withWidth("1920")
                        // Set the watermark height.
                        .withHeight("1080")
                        // Set the horizontal offset of the watermark relative to the video vertex.
                        .withDx("10")
                        // Set the vertical offset of the watermark relative to the video vertex.
                        .withDy("10")
                        // Set the watermark position.
                        //.withReferpos("BottomLeft")
                        // Set the start time of the watermark, which is used together with timeline_duration.
                        .withTimelineStart("0")
                        // Set the watermark duration. The default value is ToEND, indicating that the watermark
persists until the video ends.
                        .withTimelineDuration("ToEND"));
        // Send a watermark template request.
        try {
            UpdateWatermarkTemplateResponse rsp = initMpcClient().updateWatermarkTemplate(req);
            System.out.println("UpdateWatermarkTemplateResponse=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.9.3 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. Set query parameters.
   a. Query watermark templates by template ID.
      ```
      ListWatermarkTemplateRequest req = new
      ListWatermarkTemplateRequest().withTemplateId(Collections.singletonList(215728));
      ```
   b. Query watermark templates by page number.
      ```
      Pagination query based on page and size
      ListWatermarkTemplateRequest req = new
      ListWatermarkTemplateRequest().withPage(1).withSize(10);
      ```
2. Send a query request and return a message.
   ```
   // Send a request to MPC.
   ListWatermarkTemplateResponse rsp = initMpcClient().listWatermarkTemplate(req);
   // Print the response message.
   System.out.println("rsp=" + JsonUtils.toJSON(rsp));
   ```


Full Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.ListWatermarkTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.ListWatermarkTemplateResponse;
import com.obs.services.internal.ServiceException;

import java.util.Collections;

public class TestListWatermarkTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Query watermark templates.
     * @param args
     */
    public static void main(String[] args) {
        ListWatermarkTemplateRequest req = new
ListWatermarkTemplateRequest().withTemplateId(Collections.singletonList(215728));
        try {
            ListWatermarkTemplateResponse rsp = initMpcClient().listWatermarkTemplate(req);
            System.out.println("rsp=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```

2.9.4 Deleting a Watermark Template

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

Core Code

```
// Send a request to MPC.
DeleteWatermarkTemplateRequest req = new DeleteWatermarkTemplateRequest().withTemplateId(215728);
```



```
DeleteWatermarkTemplateResponse rsp = initMpcClient().deleteWatermarkTemplate(req);
// Print the response message.
System.out.println("httpCode=" + rsp.getHttpStatusCode() + ", rsp=" + JsonUtils.toJSON(rsp));
```

Sample Code

```
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.core.utils.JsonUtils;
import com.huaweicloud.sdk.mpc.v1.MpcClient;
import com.huaweicloud.sdk.mpc.v1.model.DeleteWatermarkTemplateRequest;
import com.huaweicloud.sdk.mpc.v1.model.DeleteWatermarkTemplateResponse;
import com.obs.services.internal.ServiceException;

public class TestDeleteWatermarkTemplate {
    /**
     * Initialize the MPC client.
     * @return
     */
    public static MpcClient initMpcClient() {
        HttpConfig httpConfig =
HttpConfig.getDefaultHttpConfig().withIgnoreSSLVerification(true).withTimeout(3);
        // Configure the HTTP proxy.
        //httpConfig.withProxyHost("xxxxxx").withProxyPort(xxxxxx).withProxyUsername("xxxxxx").
        //        withProxyPassword("xxxxxx");
        // Enter the AK and SK. To view your AK and SK, choose My Credentials > Access Keys under your
account on the console.
        String ak = "xxxxxx";
        String sk = "xxxxxx";
        // Enter the project ID. To view your project ID, choose My Credentials > API Credentials under your
account on the console.
        String projectId = "xxxxxx";
        // Enter the endpoint. The following uses region01 as an example.
        String endpoint = "https://mpc.region01.myhuaweicloud.com";
        BasicCredentials auth = new BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);
        return MpcClient.newBuilder()
                .withHttpConfig(httpConfig)
                .withCredential(auth)
                .withEndpoint(endpoint)
                .build();
    }

    /**
     * Delete a watermark template.
     * @param args
     */
    public static void main(String[] args) {
        DeleteWatermarkTemplateRequest req = new
DeleteWatermarkTemplateRequest().withTemplateId(215728);
        try {
            DeleteWatermarkTemplateResponse rsp = initMpcClient().deleteWatermarkTemplate(req);
            System.out.println("httpCode=" + rsp.getHttpStatusCode() + ", rsp=" + JsonUtils.toJSON(rsp));
        } catch (ClientRequestException | ConnectionException | RequestTimeoutException | ServiceException
e) {
            System.out.println(e);
        }
    }
}
```


2.10 Mappings Between MPC SDK and APIs

Table 2-2 Mapping between MPC SDK and APIs

SDK Interface	API	Description
createTranscoding-Task	POST /v1/{project_id}/transcodings	Creates a transcoding task.
deleteTranscoding-Task	DELETE /v1/{project_id}/transcodings{?task_id}	Cancels a transcoding task.
listTranscodingTask	GET /v1/{project_id}/transcodings{?task_id}	Queries transcoding tasks.
createTransTemplate	POST /v1/{project_id}/template/transcodings	Creates a transcoding template.
deleteTemplate	DELETE /v1/{project_id}/template/transcodings{?temp_id}	Deletes a transcoding template.
updateTransTemplate	PUT /v1/{project_id}/template/transcodings	Updates a transcoding template.
listTemplate	GET /v1/{project_id}/template/transcodings{?temp_id}	Queries transcoding templates.
createThumbnail-sTask	POST /v1/{project_id}/thumbnails	Creates a snapshot task.
deleteThumbnail-Task	DELETE /v1/{project_id}/thumbnails{?task_id}	Deletes a snapshot task.
listThumbnailsTask	GET /v1/{project_id}/thumbnails{?task_id,start_time,end_time,status,page,size}	Queries snapshot tasks.
createWatermark-Template	POST /v1/{project_id}/template/watermark	Creates a watermark template.
updateWatermark-Template	PUT /v1/{project_id}/template/watermark	Updates a watermark template.

SDK Interface	API	Description
listWatermarkTemplate	GET /v1/{project_id}/template/watermark{?template_id,page,size}	Queries watermark templates.
deleteWatermarkTemplate	DELETE /v1/{project_id}/template/watermark{?template_id}	Deletes a watermark template.
createEncryptTask	POST /v1/{project_id}/encryptions	Creates an encryption task.
deleteEncryptTask	DELETE /v1/{project_id}/encryptions/?task_id={task_id}	Deletes an encryption task.
listEncryptTask	GET /v1/{project_id}/encryptions{?task_id,start_time,end_time,status,page,size}	Queries encryption tasks.
createAnimatedGraphicsTask	POST /v1/{project_id}/animated-graphics	Creates an animated GIF task.
listAnimatedGraphicsTask	GET /v1/{project_id}/animated-graphics{?task_id,start_time,end_time,status,page,size}	Queries animated GIF tasks.
deleteAnimatedGraphicsTask	DELETE /v1/{project_id}/animated-graphics?task_id={task_id}	Deletes an animated GIF task.
createExtractTask	POST /v1/{project_id}/extract-metadata	Creates a video parsing task.
listExtractTask	GET /v1/{project_id}/extract-metadata{?task_id,start_time,end_time,status,page,size}	Queries video parsing tasks.
deleteExtractTask	DELETE /v1/{project_id}/extract-metadata{?task_id}	Deletes a video parsing task.
creatRemuxTask	POST /v1/{project_id}/remux	Creates a packaging task.
listRemuxTask	GET /v1/{project_id}/remux{?task_id,start_time,end_time,status,page,size}	Queries packaging tasks.

SDK Interface	API	Description
cancelRemuxTask	DELETE /v1/{project_id}/remux{?task_id}	Cancels a packaging task.

3 Python SDK

This section describes how to quickly integrate Python SDKs for development.

Prerequisites

- You have **registered** with Huawei Cloud and completed **real-name authentication**.

  NOTE

  If you are a **Huawei Cloud (International)** user, you need to complete real-name authentication when you:

  - Purchase and use cloud services on Huawei Cloud nodes in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Select the Chinese mainland region for MPC.
- The development environment (Python 3 or later) is available.
- You have obtained the access key (AK) and secret access key (SK) of the account. You can create and view your AK/SK on the **My Credentials** > **Access Keys** page of the console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of MPC. You can view the project ID on the **My Credentials** > **API Credentials** page of the console. For details, see **API Credentials**.
- You have uploaded the media asset files to an OBS bucket in the region of MPC, and authorized MPC to access the OBS bucket. For details, see **Uploading Media Files** and **Authorizing Access to Cloud Resources**.

Installing SDK

The MPC SDK supports Python 3 or later. Run the **python --version** command to check the Python version.

Before using the SDK, you must install **huaweicloudsdkcore** and **huaweicloudsdkmpc**. For details about the SDK version, see **SDK Center**.

- Using pip

  Run the following commands to install the Python SDK core library and related service libraries:


```
# Install the core library.
pip install huaweicloudsdkcore
# Install the MPC service library.
pip install huaweicloudsdkmpc
```

- Using source code

  Run the following commands to install the Python SDK core library and related service libraries:

```
# Install the core library.
cd huaweicloudsdkcore-${version}
python setup.py install

# Install the MPC service library.
cd huaweicloudsdkmpc-${version}
python setup.py install
```

Procedure

Step 1 Import the dependent module.

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials, GlobalCredentials
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcore.http.http_config import HttpConfig
# Import the specified MPC library.
from huaweicloudsdkmpc.v1 import *
```

Step 2 Configure client attributes.

1. Use the default configuration.

```
# Use the default configuration.
config = HttpConfig.get_default_config()
```

2. (Optional) Configure the proxy.

```
user_name = os.environ["USER_NAME"]
user_password = os.environ["USER_PASSWARD"]
# (Optional) Use a proxy server.
config.proxy_protocol = 'http'
config.proxy_host = 'proxy.huaweicloud.com'
config.proxy_port = 80
config.proxy_user = user_name
config.proxy_password = user_password
```

3. (Optional) Configure the connection.

```
# (Optional) Configure the connection timeout. The timeout can be set to timeout in a unified manner, or set to connect timeout or read timeout as required.
config.timeout = 3
```

4. (Optional) Configure SSL.

```
# (Optional) Configure whether to skip server certificate verification.
config.ignore_ssl_verification = True
# Configure the server CA certificate so that the SDK can verify the server certificate.
config.ssl_ca_cert = ssl_ca_cert
```

Step 3 Initialize authentication information.

You can use one of the following two authentication modes:

- Permanent AK/SK

  Obtain the permanent AK, SK, and project ID. For details, see **Prerequisites**.

```
ak = os.environ["SDK_AK"]
sk = os.environ["SDK_SK"]
project_id = os.environ["PROJECT_ID"]
credentials = BasicCredentials(ak, sk, project_id)
```

- Temporary AK/SK

Obtain a temporary AK, SK, and security token. For details, see **Obtaining a Temporary Access Key and Security Token Through a Token** or **Obtaining a Temporary Access Key and Security Token Through an Agency**.

```
ak = os.environ["SDK_AK"]
sk = os.environ["SDK_SK"]
project_id = os.environ["PROJECT_ID"]
security_token = os.environ["SECURITY_TOKEN"]
credentials = BasicCredentials(ak, sk, project_id).with_security_token(security_token)
```

The related parameters are as follows:

- **ak**: access key of an account
- **sk**: secret access key of an account
- **project_id**: ID of the project where MPC is provided. Select a project ID based on the region of the project.
- **security_token**: security token used for temporary AK/SK authentication

Step 4 Initialize the client.

```
# Initialize the MPC client.
client = MpcClient.new_builder(MpcClient) \
    .with_http_config(config) \
    .with_credentials(credentials) \
    .with_endpoint(endpoint) \    # endpoint value, for example, "https://mpc.region01.myhuaweicloud.com"
    .with_file_log(path="test.log", log_level=logging.INFO) \  # Print logs to a file.
    .with_stream_log(log_level=logging.INFO) \                 # Print logs to the console.
    .build()
```

- **endpoint**: regions where MPC is used and endpoints of each service. For details, see **Regions and Endpoints**.
- **with_file_log** supports the following configurations:
  - **path**: log file path
  - **log_level**: log level. The default value is **INFO**.
  - **max_bytes**: size of a log file. The default value is **10485760** bytes.
  - **backup_count**: number of log files. The default value is **5**.
- **with_stream_log** supports the following configurations:
  - **stream**: stream object. The default value is **sys.stdout**.
  - **log_level**: log level. The default value is **INFO**.

After logging is enabled, access logs are printed for each request in the following format:

```
'%(asctime)s %(thread)d %(name)s %(filename)s %(lineno)d %(levelname)s %(message)s'
```

Step 5 Send a request and view the response.

```
// Initialize the request. The following uses the API for querying transcoding tasks as an example.
request = ListTranscodingTaskRequest(task_id)
response = client.list_transcoding_task(request)
print(response)
```

Step 6 Perform troubleshooting.

Table 3-1 Troubleshooting

Level 1	Description	Level 2	Description
ConnectionException	Connection exception	HostUnreachableException	The network is unreachable or access is rejected.
		SslHandShakeException	SSL authentication is abnormal.
RequestTimeoutException	Response timeout exception	CallTimeoutException	The server fails to respond to a single request before timeout.
		RetryOutageException	No valid response is returned after the maximum number of retries specified in the retry policy is reached.
ServiceResponseException	Server response exception	ServerResponseException	Internal server error. HTTP response code: [500,].
		ClientRequestException	Invalid request parameter. HTTP response code: [400, 500).

```
# Troubleshooting
try:
    request = request = ListTranscodingTaskRequest(task_id = [1900293])
    response = client.list_transcoding_task(request)
except exception.ServiceResponseException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Step 7 Use the listener to obtain original HTTP requests and responses.

The original HTTP requests and responses are required for debugging HTTP requests sent by the service side. The SDK provides the listener to obtain the original and encrypted HTTP requests and responses.

CAUTION

Original information is printed only during debugging. Do not print the header and body of an original HTTP request in the production system because this information contains sensitive data but is not encrypted. If the request body is binary, that is, **Content-Type** is set to **binary**, the body will be displayed as ******* without the detailed content.

```
def response_handler(**kwargs):
    logger = kwargs.get("logger")
    response = kwargs.get("response")
    request = response.request
```



```
    base = "> Request %s %s HTTP/1.1" % (request.method, request.path_url) + "\n"
    if len(request.headers) != 0:
        base = base + "> Headers:" + "\n"
        for each in request.headers:
            base = base + "    %s : %s" % (each, request.headers[each]) + "\n"
    base = base + "> Body: %s" % request.body + "\n\n"

    base = base + "< Response HTTP/1.1 %s " % response.status_code + "\n"
    if len(response.headers) != 0:
        base = base + "< Headers:" + "\n"
        for each in response.headers:
            base = base + "    %s : %s" % (each, response.headers[each],) + "\n"
    base = base + "< Body: %s" % response.content
    logger.debug(base)

MpcClient client = MpcClient.new_builder(MpcClient)\
    .with_http_config(config) \
    .with_credentials(credentials) \
    .with_endpoint(endpoint) \
    .with_file_log(path="test.log", log_level=logging.INFO) \
    .with_stream_log(log_level=logging.INFO) \
    .with_http_handler(HttpHandler().add_response_handler(response_handler)) \
```

The HttpHandler supports the **add_request_handler** and **add_response_handler** methods.

----End

Sample Code

Before calling this API, replace *SDK_AK*, *SDK_SK*, *{your endpoint string}*, and *{your project id}* with the actual ones.

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcore.http.http_config import HttpConfig
from huaweicloudsdkmpc.v1 import *

def list_transcoding_task(client):
    try:
        request = ListTranscodingTaskRequest(task_id = [1900293])
        response = client.list_transcoding_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)

if __name__ == "__main__":

    ak = os.environ["SDK_AK"]
    sk = os.environ["SDK_SK"]
    project_id = os.environ["{your project id}"]
    endpoint = "{your endpoint}"

    config = HttpConfig.get_default_config()
    config.ignore_ssl_verification = True
    credentials = BasicCredentials(ak, sk, project_id)

    mpc_client = MpcClient.new_builder(MpcClient) \
        .with_http_config(config) \
        .with_credentials(credentials) \
        .with_endpoint(endpoint) \
        .build()
```



```
list_transcoding_task(mpc_client)
```


4 Go SDK

This section describes how to quickly integrate Go SDKs for development.

Prerequisites

- You have **registered** with Huawei Cloud and completed **real-name authentication**.

  NOTE

  If you are a **Huawei Cloud (International)** user, you need to complete real-name authentication when you:

  - Purchase and use cloud services on Huawei Cloud nodes in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Select the Chinese mainland region for MPC.
- The development environment (Go 1.14 or later) is available.
- You have obtained the access key (AK) and secret access key (SK) of the account. You can create and view your AK/SK on the **My Credentials** > **Access Keys** page of the console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of MPC. You can view the project ID on the **My Credentials** > **API Credentials** page of the console. For details, see **API Credentials**.
- You have uploaded the media asset files to an OBS bucket in the region of MPC, and authorized MPC to access the OBS bucket. For details, see **Uploading Media Files** and **Authorizing Access to Cloud Resources**.

Installing SDK

The Go SDK for media transcoding supports Go 1.14 or later. Run the **go version** command to check the Go version.

Run the **go get** command to install the Go SDK. Then run the following commands to install the Go SDK library and dependencies. For details about the SDK version, see **SDK Center**.

```
# Install the Go library.
go get github.com/huaweicloud/huaweicloud-sdk-go-v3
 # Install dependent libraries.
go get github.com/json-iterator/go
```


Procedure

Step 1 Import the dependent module.

```
import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/config"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/httphandler"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/mpc/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/mpc/v1/model"
    "net/http"
)
```

Step 2 Configure client attributes.

1. Use the default configuration.

   ```
   # Use default configuration
   httpConfig := config.DefaultHttpConfig()
   ```

2. (Optional) Configure the proxy.

   ```
   username := os.Getenv("USER_NAME")
   password := os.Getenv("USER_PASSWARD")
   // Configure the network proxy as required.
   httpConfig.WithProxy(config.NewProxy().
       WithSchema("http").
       WithHost("proxy.huaweicloud.com").
       WithPort(80).
       WithUsername(username).
       WithPassword(password))
   ```

3. (Optional) Configure SSL.

   ```
   // Configure whether to skip SSL certificate verification as required.
   httpConfig.WithIgnoreSSLVerification(true);
   ```

Step 3 Initialize authentication information.

You can use one of the following two authentication modes:

- Permanent AK/SK

  Obtain the permanent AK, SK, and project ID. For details, see **Prerequisites**.

  ```
  ak := os.Getenv("SDK_AK")
  sk := os.Getenv("SDK_SK")
  projectId := os.Getenv("PROJECT_ID")
  auth := basic.NewCredentialsBuilder().
          WithAk(ak).
          WithSk(sk).
          WithProjectId(projectId).
          Build()
  ```

- Temporary AK/SK

  Obtain a temporary AK, SK, and security token. For details, see **Obtaining a Temporary Access Key and Security Token Through a Token** or **Obtaining a Temporary Access Key and Security Token Through an Agency**.

  ```
  ak := os.Getenv("SDK_AK")
  sk := os.Getenv("SDK_SK")
  projectId := os.Getenv("PROJECT_ID")
  securityToken := os.Getenv("SECURITY_TOKEN")
  auth := basic.NewCredentialsBuilder().
          WithAk(ak).
          WithSk(sk).
          WithProjectId(projectId).
          WithSecurityToken(securityToken).
          Build()
  ```

The related parameters are as follows:

- **ak**: access key of an account
- **sk**: secret access key of an account
- **projectId**: ID of the project where MPC is provided. Select a project ID based on the region of the project.
- **securityToken**: security token used for temporary AK/SK authentication

Step 4 Initialize the client.

```
# Initialize the MPC client.
client := mpc.NewMpcClient
(
    mpcMpcClientBuilder().
        WithEndpoint(endpoint).     // endpoint value, for example, "https://
mpc.region01.myhuaweicloud.com"
        WithCredential(auth).
        WithHttpConfig(config.DefaultHttpConfig()).
        Build())
```

endpoint: regions where MPC is used and endpoints of each service. For details, see **Regions and Endpoints**.

Step 5 Send a request and view the response.

```
// Initialize the request. The following uses the API for querying transcoding templates as an example.
request := &model.ListTranscodingTaskRequest{
    TaskId:&[]int64{1900293},
}
response, err := client.ListTranscodingTask(request)
if err == nil {
    fmt.Printf("%+v\n",response)
} else {
    fmt.Println(err)
}
```

Step 6 Perform troubleshooting.

Table 4-1 Troubleshooting

Level 1	Description
ServiceResponseError	Service response error
url.Error	Endpoint connection error

```
# Troubleshooting
response, err := client.ListTranscodingTask(request)
if err == nil {
    fmt.Println(response)
} else {
    fmt.Println(err)
}
```

Step 7 Use the listener to obtain original HTTP requests and responses.

The original HTTP requests and responses are required for debugging HTTP requests sent by the service side. The SDK provides the listener to obtain the original and encrypted HTTP requests and responses.

CAUTION

Original information is printed only during debugging. Do not print the header and body of an original HTTP request in the production system because this information contains sensitive data but is not encrypted. If the request body is binary, that is, **Content-Type** is set to **binary**, the body will be displayed as *** without the detailed content.

```
func RequestHandler(request http.Request) {
    fmt.Println(request)
}

func ResponseHandler(response http.Response) {
    fmt.Println(response)
}

ak := os.Getenv("SDK_AK")
sk := os.Getenv("SDK_SK")
projectId := os.Getenv("{your project id}")
client := mpc.NewMpcClient(
    mpc.MpcClientBuilder().
        WithEndpoint("{your endpoint}").
        WithCredential(
            basic.NewCredentialsBuilder().
                WithAk(ak).
                WithSk(sk).
                WithProjectId(projectId).
                  Build()).
        WithHttpConfig(config.DefaultHttpConfig().
            WithIgnoreSSLVerification(true).
            WithHttpHandler(httphandler.
                NewHttpHandler().
                    AddRequestHandler(RequestHandler).
                    AddResponseHandler(ResponseHandler))).Build())
```

----End

Sample Code

Before calling this API, replace *SDK_AK*, *SDK_SK*, *{your endpoint string}*, and *{your project id}* with the actual ones.

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/config"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/httphandler"
    mpc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/mpc/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/mpc/v1/model"
    "net/http"
)

func RequestHandler(request http.Request) {
    fmt.Println(request)
}

func ResponseHandler(response http.Response) {
    fmt.Println(response)
}
ak := os.Getenv("SDK_AK")
sk := os.Getenv("SDK_SK")
projectId := os.Getenv("{your project id}")
func main() {
```



```
    client := mpc.NewMpcClient(
        mpc.MpcClientBuilder().
            WithEndpoint("{your endpoint}").
            WithCredential(
                basic.NewCredentialsBuilder().
                    WithAk(ak).
                    WithSk(sk).
                    WithProjectId(projectId).
                    Build()).
            WithHttpConfig(config.DefaultHttpConfig().
                WithIgnoreSSLVerification(true).
                WithHttpHandler(httphandler.
                    NewHttpHandler().
                        AddRequestHandler(RequestHandler).
                        AddResponseHandler(ResponseHandler))).
            Build())

    request := &model.ListTranscodingTaskRequest{
        TaskId:&[]int64{1900293},
    }
    response, err := client.ListTranscodingTask(request)
    if err == nil {
        fmt.Println("%+v\n",response)
    } else {
        fmt.Println(err)
    }
}
```

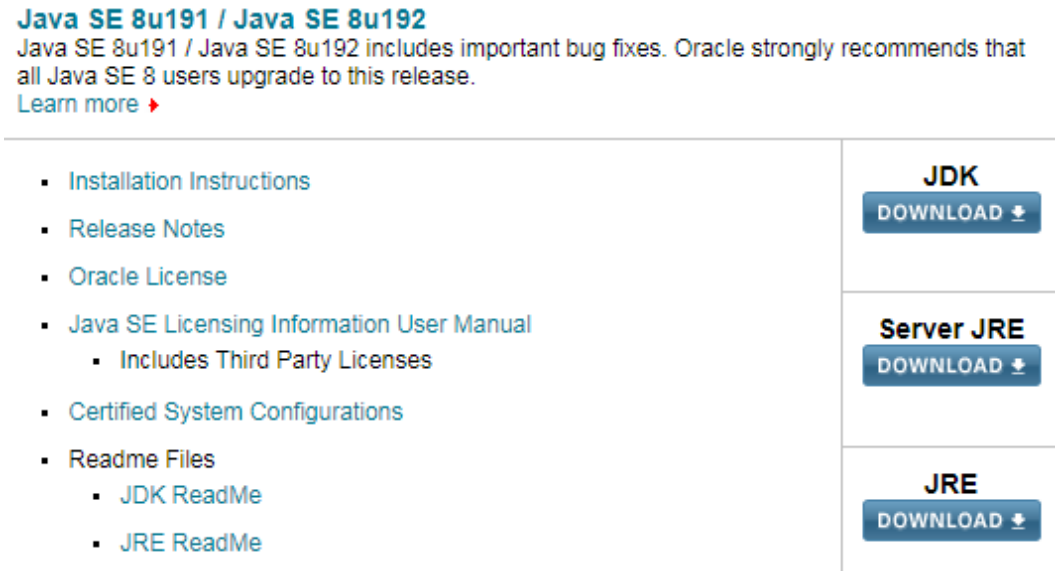
5 Appendix

5.1 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 5-1**.

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

   Table 5-1 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

5.2 Error Codes

If an error code starting with APIGW is returned after you call an API, rectify the fault by referring to the instructions provided in **API Gateway Error Codes**.

Status Code	Error Codes	Error Message	Description	Solution
400	MPC.10089	The template file does not exist.	The template file does not exist.	The template file does not exist. Please check.
400	MPC.10090	The template file does not exist.	The template file does not exist.	The template file does not exist. Please check.
400	MPC.10091	The template name already exists.	The template name already exists.	Enter another template name.

Status Code	Error Codes	Error Message	Description	Solution
400	MPC.10155	The subtitle file list and M3U8 file do not contain the specified default language.	The subtitle file list and M3U8 file do not contain the specified default language.	The subtitle file list and M3U8 file do not contain the specified default language.
400	MPC.10156	File deletion failed.	File deletion failed.	File deletion failed.
400	MPC.10202	Invalid request parameter.	Invalid request parameter.	Check whether the request parameter is correct.
400	MPC.10204	Incorrect request method.	Incorrect request method.	Check the request method.
400	MPC.10205	Incorrect request content type.	Incorrect request content type.	Check the request content type.
400	MPC.10223	An agency has been created.	An agency has been created.	An agency has been created. Please check.
400	MPC.10224	The agency has been deleted.	The agency has been deleted.	The agency has been deleted. Please check.
400	MPC.10230	The template group already exists.	The template group already exists.	The template group already exists. Please check.
400	MPC.10231	The template group does not exist.	The template group does not exist.	The template group does not exist. Please check.
401	MPC.10203	Identity authenticatio n failed.	Identity authenticatio n failed.	Check whether authentication parameters such as Token are correct.

Status Code	Error Codes	Error Message	Description	Solution
401	MPC.10206	You have not completed real-name authentication.	You have not completed real-name authentication.	Check whether you have completed real-name authentication.
401	MPC.10207	Your account is in an abnormal state.	Your account is in an abnormal state.	Check your account state.
401	MPC.10208	Tenant ID verification failed, please check.	Tenant ID verification failed, please check.	Check whether the tenant ID is correct.
403	MPC.10211	The task does not exist.	The task does not exist.	The task does not exist. Please check.
403	MPC.10212	Operation failed. The task is in progress or has been completed.	Operation failed. The task is in progress or has been completed.	The task is in progress or has been completed. Please check.
403	MPC.10214	The topic does not exist.	The topic does not exist.	The topic does not exist. Please check.
403	MPC.10215	The topic already exists.	The topic already exists.	The topic already exists. Please check.
403	MPC.10226	The resource does not exist.	The resource does not exist.	The resource does not exist.
403	MPC.10240	Failed to obtain the basic information about the media file.	Failed to obtain the basic information about the media file.	Check whether the OBS bucket has been authorized and whether KMS encryption has been configured for the OBS bucket.

Status Code	Error Codes	Error Message	Description	Solution
403	MPC.10243	Due to security reasons, your account has been restricted from purchasing certain pay-per-use cloud service resources according to the CLOUD Customer Agreement. If you have any questions, contact customer service.	Due to security reasons, your account has been restricted from purchasing certain pay-per-use cloud service resources according to the CLOUD Customer Agreement. If you have any questions, contact customer service.	Due to security reasons, your account has been restricted from purchasing certain pay-per-use cloud service resources according to the CLOUD Customer Agreement. If you have any questions, contact customer service.
403	MPC.10244	Insufficient account balance. Top up your account.	Insufficient account balance. Top up your account.	Insufficient account balance. Top up your account.
406	MPC.10051	The selected template is a super-resolution template and is not supported.	The selected template is a super-resolution template and is not supported.	Select a valid template.
406	MPC.10052	Failed to obtain the input file.	Failed to obtain the input file.	Check the file path.
406	MPC.10053	The input file does not exist.	The input file does not exist.	The input file does not exist. Please check.
406	MPC.10054	Failed to obtain the subtitle file.	Failed to obtain the subtitle file.	Check the file path.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10055	The audio sampling rate 7,350 is not supported.	The audio sampling rate 7,350 is not supported.	Set a valid audio sampling rate.
406	MPC.10056	This type of output frame rate is not supported.	This type of output frame rate is not supported.	Set a valid frame rate.
406	MPC.10057	This type of output bitrate is not supported.	This type of output bitrate is not supported.	Set a valid bitrate.
406	MPC.10058	This type of output video width is not supported.	This type of output video width is not supported.	Set a valid width.
406	MPC.10059	This type of output video height is not supported.	This type of output video height is not supported.	Set a valid height.
406	MPC.10060	This type of I-frame interval is not supported.	This type of I-frame interval is not supported.	Set a valid I-frame interval.
406	MPC.10061	Capturing snapshots at non-fixed intervals is not supported.	Capturing snapshots at non-fixed intervals is not supported.	Change the interval to a fixed interval.
406	MPC.10062	Invalid video codec.	Invalid video codec.	Set a valid video codec.
406	MPC.10063	Invalid video format.	Invalid video format.	Set a valid video format.
406	MPC.10064	Multiple watermarks are not supported.	Multiple watermarks are not supported.	Only two watermarks are supported.
406	MPC.10065	Invalid output file format.	Invalid output file format.	Set a valid output format.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10066	The input file format does not match the actual format.	The input file format does not match the actual format.	The input file format does not match the actual format. Please check.
406	MPC.10067	Failed to obtain the ID of the video codec.	Failed to obtain the ID of the video codec.	Failed to obtain the ID of the video codec. Please check.
406	MPC.10068	Failed to obtain the ID of the audio codec.	Failed to obtain the ID of the audio codec.	Failed to obtain the ID of the audio codec. Please check.
406	MPC.10069	Failed to obtain the ID of the subtitle codec.	Failed to obtain the ID of the subtitle codec.	Failed to obtain the ID of the subtitle codec. Please check.
406	MPC.10070	Failed to obtain the encoding/ decoding format.	Failed to obtain the encoding/ decoding format.	Failed to obtain the encoding/ decoding format. Please check.
406	MPC.10071	Failed to obtain the parameters of the input video stream.	Failed to obtain the parameters of the input video stream.	Failed to obtain the parameters of the input video stream. Please check.
406	MPC.10072	Invalid frame rate of the video stream.	Invalid frame rate of the video stream.	Invalid frame rate of the video stream. Please check.
406	MPC.10080	Invalid frame rate of the input file.	Invalid frame rate of the input file.	Invalid frame rate of the input file. Please check.
406	MPC.10081	The file does not contain audio streams.	The file does not contain audio streams.	Check the input file.
406	MPC.10082	Failed to obtain the input audio or video stream.	Failed to obtain the input audio or video stream.	Failed to obtain the input audio or video stream. Please check.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10083	This type of codec is not supported.	This type of codec is not supported.	This type of codec is not supported. Please check.
406	MPC.10084	This chroma subsampling format is not supported.	This chroma subsampling format is not supported.	This chroma subsampling format is not supported. Please check.
406	MPC.10085	The file format is not supported.	The file format is not supported.	The file format is not supported. Please check.
406	MPC.10086	Failed to obtain the input file.	Failed to obtain the input file.	Check the file path.
406	MPC.10087	Invalid task parameters.	Invalid task parameters.	Invalid task parameters. Please check.
406	MPC.10088	The image file does not exist.	The image file does not exist.	The image file does not exist. Please check.
406	MPC.10092	The image file does not exist.	The image file does not exist.	The image file does not exist. Please check.
406	MPC.10093	The file name exceeds the maximum length.	The file name exceeds the maximum length.	The file name exceeds the maximum length. Please check.
406	MPC.10094	Invalid file format.	Invalid file format.	Invalid file format. Please check.
406	MPC.10095	The watermark is placed in a wrong position.	The watermark is placed in a wrong position.	The watermark is placed in a wrong position. Please check.
406	MPC.10096	Invalid watermark size.	Invalid watermark size.	Invalid watermark size. Please check.
406	MPC.10097	Invalid watermark scaling ratio.	Invalid watermark scaling ratio.	Invalid watermark scaling ratio. Please check.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10098	Invalid watermark duration.	Invalid watermark duration.	Invalid watermark duration. Please check.
406	MPC.10099	The media stream type is not supported.	The media stream type is not supported.	The media stream type is not supported. Please check.
406	MPC.10100	An error occurred when parsing the video frame rate information.	An error occurred when parsing the video frame rate information.	An error occurred when parsing the video frame rate information. Please check.
406	MPC.10101	Invalid input parameters.	Invalid input parameters.	Invalid input parameters. Please check.
406	MPC.10102	Failed to open the input file.	Failed to open the input file.	Failed to open the input file. Please check.
406	MPC.10103	Open GOP is not supported.	Open GOP is not supported.	Open GOP is not supported. Please check.
406	MPC.10104	Internal error.	Internal error.	Try again or contact technical support.
406	MPC.10105	An error occurred during transcoding.	An error occurred during transcoding.	Try again or contact technical support.
406	MPC.10106	The audio sampling rate is lower than 12,000. The audio will be discarded.	The audio sampling rate is lower than 12,000. The audio will be discarded.	The audio sampling rate is lower than 12,000. The audio will be discarded.
406	MPC.10107	Invalid input video resolution.	Invalid input video resolution.	Invalid input video resolution. Please check.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10108	The audio sampling rate of the input video is incorrect.	The audio sampling rate of the input video is incorrect.	The audio sampling rate of the input video is incorrect. Please check.
406	MPC.10109	Invalid resolution in the template.	Invalid resolution in the template.	Invalid resolution in the template. Please check.
406	MPC.10110	The video encoding format of the input file is not supported.	The video encoding format of the input file is not supported.	The video encoding format of the input file is not supported.
406	MPC.10111	Failed to obtain the file from OBS.	Failed to obtain the file from OBS.	Failed to obtain the file from OBS.
406	MPC.10112	The video or audio format of the input file is not supported.	The video or audio format of the input file is not supported.	The video or audio format of the input file is not supported.
406	MPC.10113	The DTS of the input file is not supported.	The DTS of the input file is not supported.	The DTS of the input file is not supported.
406	MPC.10114	The header information of the input file is incorrect.	The header information of the input file is incorrect.	The header information of the input file is incorrect. Please check.
406	MPC.10115	The watermark cannot be scaled down by more than 256 times.	The watermark cannot be scaled down by more than 256 times.	The watermark cannot be scaled down by more than 256 times.
406	MPC.10116	The audio encoding format of the input file is not supported.	The audio encoding format of the input file is not supported.	The audio encoding format of the input file is not supported.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10117	The audio and video in the input file are not synchronized.	The audio and video in the input file are not synchronized.	The audio and video in the input file are not synchronized.
406	MPC.10118	Failed to upload files to the OBS path.	Failed to upload files to the OBS path.	Try again or contact technical support.
406	MPC.10119	Invalid input data.	Invalid input data.	Invalid input data. Please check.
406	MPC.10120	The task does not exist.	The task does not exist.	The task does not exist. Please check.
406	MPC.10121	The subtitle file does not exist.	The subtitle file does not exist.	The subtitle file does not exist. Please check.
406	MPC.10122	The resolution in the template is greater than the input video resolution.	The resolution in the template is greater than the input video resolution.	The resolution in the template is greater than the input video resolution. Please check.
406	MPC.10123	The header information of the input file is incorrect.	The header information of the input file is incorrect.	The header information of the input file is incorrect. Please check.
406	MPC.10124	Some data in the input file are missing.	Some data in the input file are missing.	Check the input file for its completeness.
406	MPC.10125	Input data error.	Input data error.	Check whether the input file can be played.
406	MPC.10126	Input data error.	Input data error.	Check whether the input file can be played.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10127	Failed to obtain the level-1 m3u8 when an HLS media file is encrypted with DRM.	Failed to obtain the level-1 m3u8 when an HLS media file is encrypted with DRM.	Failed to obtain the level-1 m3u8 when an HLS media file is encrypted with DRM. Please check.
406	MPC.10128	Failed to obtain the level-2 m3u8 when an HLS media file is encrypted with DRM.	Failed to obtain the level-2 m3u8 when an HLS media file is encrypted with DRM.	Failed to obtain the level-2 m3u8 when an HLS media file is encrypted with DRM. Please check.
406	MPC.10129	Failed to obtain the index file when a DASH media file is encrypted with DRM.	Failed to obtain the index file when a DASH media file is encrypted with DRM.	Failed to obtain the index file when a DASH media file is encrypted with DRM. Please check.
406	MPC.10130	The HLS content fails to be encrypted using DRM.	The HLS content fails to be encrypted using DRM.	The HLS content fails to be encrypted using DRM. Please check.
406	MPC.10131	Failed to modify the index file when an HLS media file is encrypted with DRM.	Failed to modify the index file when an HLS media file is encrypted with DRM.	Failed to modify the index file when an HLS media file is encrypted with DRM. Please check.
406	MPC.10132	Failed to obtain the IV during DRM encryption.	Failed to obtain the IV during DRM encryption.	Failed to obtain the IV during DRM encryption. Please check.
406	MPC.10133	The DASH content fails to be encrypted using DRM.	The DASH content fails to be encrypted using DRM.	The DASH content fails to be encrypted using DRM. Please check.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10134	Failed to modify the index file when a DASH media file is encrypted with DRM.	Failed to modify the index file when a DASH media file is encrypted with DRM.	Failed to modify the index file when a DASH media file is encrypted with DRM. Please check.
406	MPC.10135	Failed to package the digital watermark due to the incorrect xformat configuration.	Failed to package the digital watermark due to the incorrect xformat configuration.	Failed to package the digital watermark due to the incorrect xformat configuration. Please check.
406	MPC.10136	Failed to package the digital watermark because xformat fails to be started.	Failed to package the digital watermark because xformat fails to be started.	Failed to package the digital watermark because xformat fails to be started. Please check.
406	MPC.10137	Failed to package the digital watermark because xformat fails to create a task.	Failed to package the digital watermark because xformat fails to create a task.	Failed to package the digital watermark because xformat fails to create a task. Please check.
406	MPC.10138	Failed to package the digital watermark because xformat fails to query the task.	Failed to package the digital watermark because xformat fails to query the task.	Failed to package the digital watermark because xformat fails to query the task. Please check.
406	MPC.10139	Failed to package the digital watermark because the xformat task timed out.	Failed to package the digital watermark because the xformat task timed out.	Try again or contact technical support.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10140	The I-frame interval exceeds 500.	The I-frame interval exceeds 500.	Set the I-frame interval to 500 or less.
406	MPC.10141	The input file is an audio file. The selected template contains video parameters.	The input file is an audio file. The selected template contains video parameters.	Select a valid template.
406	MPC.10143	Invalid index file content.	Invalid index file content.	Invalid index file content.
406	MPC.10144	Black bars seem to be on the input video.	Black bars seem to be on the input video.	Review the input video.
406	MPC.10145	Data frames imported to the detection module seem to be not enough for identifying the specific position of the black bar.	Data frames imported to the detection module seem to be not enough for identifying the specific position of the black bar.	Review the input video.
406	MPC.10146	The black bar seems to overlap with subtitles.	The black bar seems to overlap with subtitles.	Review the input video.
406	MPC.10147	The black bar seems to overlap with the watermark.	The black bar seems to overlap with the watermark.	Review the input video.
406	MPC.10148	The black bars seem to be asymmetric.	The black bars seem to be asymmetric.	Review the input video.
406	MPC.10149	The specific position of the black bar cannot be identified.	The specific position of the black bar cannot be identified.	Review the input video.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10150	The cropped black bar size exceeds the input video size.	The cropped black bar size exceeds the input video size.	Review the input video.
406	MPC.10151	Failed to download the subtitle file in the slicing phase.	Failed to download the subtitle file in the slicing phase.	Failed to download the subtitle file in the slicing phase. Please check.
406	MPC.10152	The video encoding format of the input file is not supported.	The video encoding format of the input file is not supported.	The video encoding format of the input file is not supported. Please check.
406	MPC.10153	Input file error.	Input file error.	Check whether the input file can be played.
406	MPC.10154	Failed to open the input file.	Failed to open the input file.	Check whether the input file can be played.
406	MPC.10200	System error.	System error.	Contact technical support.
406	MPC.10201	Internal communication error.	Internal communication error.	Contact technical support.
406	MPC.10209	Invalid input or output OBS path.	Invalid input or output OBS path.	Check whether the input or output OBS path is correct.
406	MPC.10210	Failed to obtain the input file from OBS.	Failed to obtain the input file from OBS.	Failed to obtain the input file from OBS. Please check.
406	MPC.10213	Operation failed. The task is not in the final state.	Operation failed. The task is not in the final state.	The task is not in the final state. Please check.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10216	Failed to set event notifications. You do not have the permission to publish messages to the topic.	Failed to set event notifications. You do not have the permission to publish messages to the topic.	You do not have the permission to publish messages to the topic. Please check.
406	MPC.10217	The usage exceeds the OBT quota.	The usage exceeds the OBT quota.	The usage exceeds the threshold. Please check.
406	MPC.10218	The task has completed.	The task has completed.	The task has completed. Please check.
406	MPC.10219	Invalid request parameter.	Invalid request parameter.	Invalid request parameter. Please check.
406	MPC.10220	The task has expired.	The task has expired.	The task has expired. Please check.
406	MPC.10221	Internal service error.	Internal service error.	Check the template and try again.
406	MPC.10222	Key parameters in the template are inconsistent.	Key parameters in the template are inconsistent.	The obtained input file is not correct. Please check.
406	MPC.10225	KMS service error.	KMS service error.	Contact technical support.
406	MPC.10227	You do not have the permission to access the requested resource.	You do not have the permission to access the requested resource.	You do not have the access permission. Please check.
406	MPC.10228	Your account is in arrears. Top up your account.	Your account is in arrears. Top up your account.	Top up your account as soon as possible.

Status Code	Error Codes	Error Message	Description	Solution
406	MPC.10229	You do not have the permission to perform this operation.	You do not have the permission to perform this operation.	You do not have the permission to perform this operation. Please check.
406	MPC.10232	GIF task failed.	GIF task failed.	GIF task failed. Please check.
406	MPC.10233	Packaging task failed.	Packaging task failed.	Packaging task failed. Please check.
406	MPC.10234	The function is temporarily brought offline.	The function is temporarily brought offline.	The function is offline. Please check.
406	MPC.10235	Identity authenticatio n failed due to an invalid token.	Identity authenticatio n failed due to an invalid token.	Check whether the token is correct.
406	MPC.10236	You do not have permission to access the OBS bucket.	You do not have the permission to access the OBS bucket.	Contact the tenant administrator to perform bucket authorization or to grant member accounts the permission to access OBS.
406	MPC.10237	API Gateway rate limiting	API Gateway rate limiting	API Gateway is implementing rate limiting. Please check.
500	MPC.10001	IAM service exception.	IAM service exception.	Contact technical support.
500	MPC.10002	OBS service exception.	OBS service exception.	Contact technical support.
500	MPC.10003	SMN service exception.	SMN service exception.	Contact technical support.
500	MPC.10004	CBC service exception.	CBC service exception.	Contact technical support.
500	MPC.10005	SDR service exception.	SDR service exception.	Contact technical support.

Status Code	Error Codes	Error Message	Description	Solution
500	MPC.10006	ZK service exception.	ZK service exception.	Contact technical support.
500	MPC.10007	MONGO service exception.	MONGO service exception.	Contact technical support.
500	MPC.10008	MPE service exception.	MPE service exception.	Contact technical support.
500	MPC.10050	XCODE service exception.	XCODE service exception.	Contact technical support.

5.3 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.

Status Code	Description
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.

5.4 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**: Some request URLs must contain this field.
- Account name: This field must be contained in some request URLs.
- Endpoint: regions and endpoints for available services

Prerequisites

You have **registered** with Huawei Cloud and completed **real-name authentication**.

NOTE

If you are a **Huawei Cloud (International)** user, you need to complete real-name authentication when you:

- Purchase and use cloud services on Huawei Cloud nodes in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
- Select the Chinese mainland region for MPC.

Obtaining the AK/SK Pair

Note: Access keys have full access permissions for your account. If access keys are disclosed, data leakage may occur. For account security, you are advised to periodically change and keep access keys secure. You can create up to two access keys for each account.

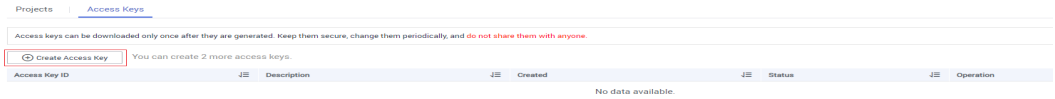
Step 1 Log in to the console.

Step 2 Point to the username and choose **My Credentials** from the drop-down list.

Step 3 In the navigation pane, choose **Access Keys**.

Step 4 Click **Create Access Key**. On the displayed page, enter the account and password and SMS verification code.

Figure 5-1 Access key



Step 5 Click **OK** to download the **credentials.csv** file that contains the AK and SK pair.

----End

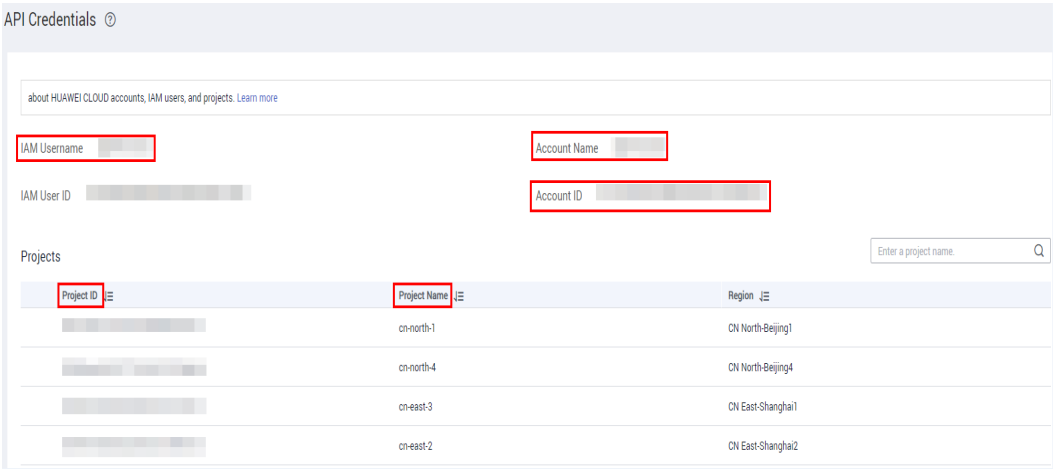
Obtaining a Project ID and Account Name

Step 1 Log in to the console.

Step 2 Point to the username and choose **My Credentials** from the drop-down list.

Step 3 On the **API Credentials** page, obtain the project ID and account name.

Figure 5-2 Obtaining a Project ID



----End

Obtaining an Endpoint

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

