Live

Cloud Live Server SDK Reference

Issue 01

Date 2024-09-26

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

You can integrate the Live server SDK provided by Huawei Cloud to call Live APIs for quick operations. Currently, the Live SDK is available in Java, Python, Go, and PHP, as shown in **Table 1-1**.

NOTE

For details about known SDK security vulnerabilities, fixed versions, and workarounds, click the GitHub address of the desired SDK language in **Table 1-1**.

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**
PHP	**huaweicloud-sdk-php-v3**	**PHP SDK User Guide**

Table 1-2 lists the Live APIs supported by the SDK. The SDK will be upgraded to support all Live APIs.

Table 1-2 Mapping between the SDK and APIs

Java SDK	Python SDK	Go SDK	API Reference
createDomain	create_domain	CreateDomain	**Creating a Domain Name**
deleteDomain	delete_domain	DeleteDomain	**Deleting a Domain Name**
updateDomain	update_domain	UpdateDomain	**Modifying a Domain Name**

Java SDK	Python SDK	Go SDK	API Reference
showDomain	show_domain	ShowDomain	Querying a Domain Name
createDomainMapping	create_domain_mapping	CreateDomainMapping	Mapping Domain Names
deleteDomainMapping	delete_domain_mapping	DeleteDomainMapping	Deleting a Domain Name Mapping
createTranscodingsTemplate	create_transcodings_template	CreateTranscodingsTemplate	Creating a Transcoding Template
updateTranscodingsTemplate	update_transcodings_template	UpdateTranscodingsTemplate	Modifying a Transcoding Template
deleteTranscodingsTemplate	delete_transcodings_template	DeleteTranscodingsTemplate	Deleting a Transcoding Template
showTranscodingsTemplate	show_transcodings_template	ShowTranscodingsTemplate	Querying Transcoding Templates
createStreamForbidden	create_stream_forbidden	CreateStreamForbidden	Disabling a Push Stream
updateStreamForbidden	update_stream_forbidden	UpdateStreamForbidden	Modifying the Attribute of a Disabled Stream
deleteStreamForbidden	delete_stream_forbidden	DeleteStreamForbidden	Resuming a Push Stream
listStreamForbidden	list_stream_forbidden	ListStreamForbidden	Querying Disabled Streams
listLiveSampleLogs	list_live_sample_logs	ListLiveSampleLogs	Obtaining Live Streaming Logs
createRecordRule	create_record_rule	CreateRecordRule	Creating a Recording Template
listRecordRules	list_record_rules	ListRecordRules	Querying Recording Templates
updateRecordRule	update_record_rule	UpdateRecordRule	Modifying a Recording Template

Java SDK	Python SDK	Go SDK	API Reference
deleteRecordRule	delete_record_rule	DeleteRecordRule	Deleting a Recording Template
showRecordRule	show_record_rule	ShowRecordRule	Querying a Recording Template
runRecord	run_record	RunRecord	Submitting a Recording Command
createRecordCallbackConfig	create_record_callback_config	CreateRecordCallback-Config	Creating a Recording Callback
listRecordCallbackConfigs	list_record_callback_configs	ListRecordCallback-Configs	Querying the List of Recording Callbacks
updateRecordCallbackConfig	update_record_callback_config	UpdateRecordCall-backConfig	Modifying a Recording Callback
showRecordCallbackConfig	show_record_callback_config	ShowRecordCallback-Config	Querying a Recording Callback
deleteRecordCallbackConfig	delete_record_callback_config	DeleteRecordCall-backConfig	Deleting a Recording Callback
listDomainBandwidthPeak	list_domain_bandwidth_peak	ListDomainBandwidthPeak	Querying Peak Bandwidth
listDomainTrafficSummary	list_domain_traffic_summary	ListDomainTraffic-Summary	Querying Total Traffic
listQueryHttpCode	list_query_http_code	ListQueryHttpCode	Querying HTTP Status Codes for Pulling Live Streams
listTranscodeData	list_transcode_data	ListTranscodeData	Querying the Duration of Transcoded Outputs
listRecordData	list_record_data	ListRecordData	Querying Recording Channels
listSnapshotData	list_snapshot_data	ListSnapshotData	Querying the Number of Snapshots

Java SDK	Python SDK	Go SDK	API Reference
showUpBandwidth	show_up_bandwidth	ShowUpBandwidth	**Querying Upstream Bandwidth**
showStreamCount	show_stream_count	ShowStreamCount	**Querying the Number of Pushed Streams by Domain Name**
listHistoryStreams	list_history_streams	ListHistoryStreams	**Querying the Historical Stream List**
showStreamPortrait	show_stream_portrait	ShowStreamPortrait	**Querying the Playback Profile**
listSingleStreamFramerate	list_single_stream_framerate	ListSingleStream-Framerate	**Querying the Stream Push Frame Rate**
listSingleStreamBitrate	list_single_stream_bitrate	ListSingleStreamBi-trate	**Querying the Stream Push Bitrate**
listLiveStreamsOnline	list_live_streams_online	ListLiveStreamsOnline	**Querying an Ongoing Stream**

2 Java SDK User Guide

This section describes how to quickly integrate the Java SDK for development.

Prerequisites

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

  NOTE

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

  - Purchase and use cloud services in Huawei Cloud regions in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Plan to use Live in Huawei Cloud regions in the Chinese mainland.
- You have obtained and **added an ingest domain name and a streaming domain name** (both licensed) on the Live console, and **associated the domain names**.
- The development environment (Java JDK 1.8 or later) is available.
- You have obtained the access key (AK) and secret access key (SK) of the Huawei Cloud account. You can create and view your AK and SK on the **My Credentials** > **Access Keys** page of the Huawei Cloud console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of Live. You can view the project ID on the **My Credentials** > **API Credentials** page of the Huawei Cloud console. For details, see **API Credentials**.

Notes

When the version of the introduced third-party library conflicts with the version of the third-party library on which the Live service depends, for example, the version conflict of Jackson or OkHttp3, the following bundle package (later than 3.0.40-rc) can be introduced to redirect the version of the third-party library on which the SDK depends to resolve the version conflict. For details, see **SDK Center** to check the SDK bundle package of the Java SDK of Live.

Note: The bundle package contains the core package and cloud service collection package. Therefore, you do not need to introduce the core package and service

package separately. Otherwise, the bundle package may not take effect based on the parsing sequence of Maven dependencies.

```
<dependency>
    <groupId>com.huaweicloud.sdk</groupId>
    <artifactId>huaweicloud-sdk-bundle</artifactId>
    <version>${version}</version>
</dependency>
```

Installing an SDK

You can obtain and install an SDK in Maven mode. You need to **download** and **install Maven** in your operating system (OS). After the installation is complete, you only need to add the corresponding dependencies to the **pom.xml** file of the Java project.

Before using the server SDK, you need to install **huaweicloud-sdk-core** and **huaweicloud-sdk-live**. For details about SDK versions, see **SDK Center**.

CAUTION

Replace the value of **version** in the following sample code with the actual SDK version.

```
<dependency>
    <groupId>com.huaweicloud.sdk</groupId>
    <artifactId>huaweicloud-sdk-core</artifactId>
    <version>3.1.11</version>
</dependency>
<dependency>
    <groupId>com.huaweicloud.sdk</groupId>
    <artifactId>huaweicloud-sdk-live</artifactId>
    <version>3.1.11</version>
</dependency>
```

Procedure

Step 1 Import the dependent module.

```
// Perform user authentication.
import com.huaweicloud.sdk.core.auth.BasicCredentials;
// Import a request exception class.
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ServerResponseException;
// Configure HTTP.
import com.huaweicloud.sdk.core.http.HttpConfig;
// Import a Live client.
import com.huaweicloud.sdk.live.v1.LiveClient;
// Import the request and response classes of an API.
import com.huaweicloud.sdk.live.v1.model.ShowTranscodingsTemplateRequest;
import com.huaweicloud.sdk.live.v1.model.ShowTranscodingsTemplateResponse;
// 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 a proxy.


```
// (Optional) Use a proxy server.
// There will be huge security risks if the password of the proxy server is directly written into the code. You are advised to store the password in ciphertext in the configuration file or environment variables and decrypt the password when using it.
// Before configuring a proxy, specify the environment variable PROXY_PASSWORD in the local environment.
config.withProxyHost("http://proxy.huaweicloud.com")
    .withProxyPort(8080)
    .withProxyUsername("test")
    .withProxyPassword(System.getenv("PROXY_PASSWORD"));
```

3. (Optional) Configure a connection.

```
// (Optional) Configure connection timeout.
config.withTimeout(3);
```

4. (Optional) Configure SSL.

```
// (Optional) Skip server certificate verification.
config.withIgnoreSSLVerification(true);
```

Step 3 Initialize authentication information.

You can use one of the following two authentication modes:

- Permanent AK and SK

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

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

- Temporary AK and SK

  Obtain a temporary AK, SK, and security token **through a token** or **through an agency**.

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

The related parameters are as follows:

- **ak**: AK of the Huawei Cloud account. You are advised to store the AK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **sk**: SK of the Huawei Cloud account. You are advised to store the SK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **projectId**: ID of the project where Live is provided. Select a project ID based on the region of the project.
- **securityToken**: security token used for temporary AK and SK authentication

Step 4 Initialize the client.

```
// Initialize the Live client.
LiveClient liveClient = LiveClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(credentials)
    .withRegion(region)
    .build();
```

region: regions where Live is used and endpoints of each service.

Step 5 Send a request and view the response.

```
// Initialize the request. The following uses the API for querying transcoding templates as an example.
ShowTranscodingsTemplateResponse showTranscodingsTemplateResponse =
```



```
liveClient.showTranscodingsTemplate(
    new ShowTranscodingsTemplateRequest().withDomain(domain) // domain indicates the domain name
to be queried.
);
logger.info(showTranscodingsTemplateResponse.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).

```
// Perform troubleshooting.
try {
    ShowTranscodingsTemplateResponse showTranscodingsTemplateResponse =
liveClient.showTranscodingsTemplate(
        new ShowTranscodingsTemplateRequest().withDomain("play.example.huaweicloud.com")
    );
} 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.
LiveAPIAsyncClient liveAsyncClient = LiveAPIAsyncClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(credentials)
    .withRegion(region)
    .build();
```



```
// Send an asynchronous request.
CompletableFuture<ShowTranscodingsTemplateResponse> future =
LiveAPIAsyncClient.showTranscodingsTemplate(
    new ShowTranscodingsTemplateRequest().withDomain("play.example.huaweicloud.com")
);

// Obtain the asynchronous response.
showTranscodingsTemplateResponse 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 Huawei Cloud 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(""))));

LiveClient liveClient = LiveClient.newBuilder()
    .withHttpConfig(config)
    .withCredential(auth)
    .withRegion(region)
    .build();
```

----End

Sample Code

Before the calling, replace the variables *{your endpoint string}* and *{your project id}* as needed.

There will be huge security risks if the AK and SK used for authentication are directly written into the code. You are advised to store the AK and SK in ciphertext in the configuration file or environment variables and decrypt them when using them.

In this example, the AK and SK are stored in environment variables. Before running this example, specify the environment variables *HUAWEICLOUD_SDK_AK* and *HUAWEICLOUD_SDK_SK* in the local environment.

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

// Perform user authentication.
import com.huaweicloud.sdk.core.auth.BasicCredentials;
// Import a request exception class.
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ServerResponseException;
// Configure HTTP.
import com.huaweicloud.sdk.core.http.HttpConfig;
// Import the Live client.
import com.huaweicloud.sdk.live.v1.LiveClient;
```



```
// Import the request and response classes of an API.
import com.huaweicloud.sdk.live.v1.model.ShowTranscodingsTemplateRequest;
import com.huaweicloud.sdk.live.v1.model.ShowTranscodingsTemplateResponse;
// Print logs.
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class Application {
    private static final Logger logger = LoggerFactory.getLogger(Application.class);

    public static void showTranscodingsTemplate(LiveClient client) {
        try {
            // Initialize the request. The following uses the API for querying transcoding templates as an
example.
            ShowTranscodingsTemplateResponse showTranscodingsTemplateResponse =
client.showTranscodingsTemplate(
                new ShowTranscodingsTemplateRequest().withDomain("play.example.huaweicloud.com")
            );
            // Output the response in JSON.
            logger.info(showTranscodingsTemplateResponse.toString());
        } catch (ClientRequestException e) {
            logger.error("HttpStatusCode: " + e.getHttpStatusCode());
            logger.error("RequestId: " + e.getRequestId());
            logger.error("ErrorCode: " + e.getErrorCode());
            logger.error("ErrorMsg: " + e.getErrorMsg());
        }
    }
    public static void main(String[] args) {
        String ak = System.getenv("HUAWEICLOUD_SDK_AK");
        String sk = System.getenv("HUAWEICLOUD_SDK_SK");
        String endpoint = "{your endpoint string}";
        String projectId = "{your project id}";

        // Configure client attributes.
        HttpConfig config = HttpConfig.getDefaultHttpConfig();
        config.withIgnoreSSLVerification(true);

        // Create a credential.
        BasicCredentials auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk)
            .withProjectId(projectId);

        // Create and initialize a Live client instance.
        LiveClient liveClient = LiveClient.newBuilder()
            .withHttpConfig(config)
            .withCredential(auth)
            .withRegion(region)
            .build();

        showTranscodingsTemplate(liveClient);
    }
}
```


3 Python SDK User Guide

This section describes how to quickly integrate the Python SDK for development.

Prerequisites

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

  NOTE

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

  - Purchase and use cloud services in Huawei Cloud regions in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Plan to use Live in Huawei Cloud regions in the Chinese mainland.
- You have obtained and **added an ingest domain name and a streaming domain name** (both licensed) on the Live console, and **associated the domain names**.
- The development environment (Python 3 or later) is available.
- You have obtained the access key (AK) and secret access key (SK) of the Huawei Cloud account. You can create and view your AK and SK on the **My Credentials** > **Access Keys** page of the Huawei Cloud console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of Live. You can view the project ID on the **My Credentials** > **API Credentials** page of the Huawei Cloud console. For details, see **API Credentials**.

Installing an SDK

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

Before using the server SDK, you need to install **huaweicloudsdkcore** and **huaweicloudsdklive**. For details about SDK versions, 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 Live service library.
pip install huaweicloudsdklive
```

- 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 Live service library.
cd huaweicloudsdklive-${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 Live library.
from huaweicloudsdklive.v1 import *
```

Step 2 Configure client attributes.

1. Use the default configuration.

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

2. (Optional) Configure a proxy.

```
# (Optional) Use a proxy server.
# There will be huge security risks if the password of the proxy server is directly written into the code. You are advised to store the password in ciphertext in the configuration file or environment variables and decrypt the password when using it.
# Before configuring a proxy, specify the environment variable PROXY_PASSWORD in the local environment.
config.proxy_protocol = 'http'
config.proxy_host = 'proxy.huaweicloud.com'
config.proxy_port = 80
config.proxy_user = 'username'
config.proxy_password = os.environ['PROXY_PASSWORD']
```

3. (Optional) Configure a 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) 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 and SK

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

```
credentials = BasicCredentials(ak, sk, project_id)
```

- Temporary AK and SK

  Obtain a temporary AK, SK, and security token **through a token** or **through an agency**.


```
credentials = BasicCredentials(ak, sk, project_id).with_security_token(security_token)
```

The related parameters are as follows:

- **ak**: AK of the Huawei Cloud account. You are advised to store the AK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **sk**: SK of the Huawei Cloud account. You are advised to store the SK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **project_id**: ID of the project where Live is provided. Select a project ID based on the region of the project.
- **security_token**: security token used for temporary AK and SK authentication

Step 4 Initialize the client.

```
# Initialize the Live client.
client = LiveAPIClient.new_builder(LiveAPIClient) \
    .with_http_config(config) \
    .with_credentials(credentials) \
    .with_endpoint(endpoint) \
    .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 Live is used and endpoints of each service.
- **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 templates as an example.
request = huaweicloudsdklive.ShowTranscodingsTemplateRequest("play.example.huaweicloud.com")
response = client.show_transcodings_template(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.

Level 1	Description	Level 2	Description
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 = huaweicloudsdklive.ShowTranscodingsTemplateRequest("play.example.huaweicloud.com")
    response = client.show_transcodings_template(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 an asynchronous client.

```
# Initialize the asynchronous client, as shown in the following example:
live_client = LiveAsyncClient.new_builder(LiveAsyncClient) \
    .with_http_config(config) \
    .with_credentials(credentials) \
    .with_endpoint(endpoint) \
    .build()

# Send an asynchronous request.
request = huaweicloudsdklive.ShowTranscodingsTemplateRequest("play.example.huaweicloud.com")
response = live_client.show_transcodings_template_async(request)

# Obtain the asynchronous response.
print(response.result())
```

Step 8 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)

client = LiveAPIClient.new_builder(LiveAPIClient)\
    .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 the calling, replace the variables *{your endpoint string}* and *{your project id}* as needed.

There will be huge security risks if the AK and SK used for authentication are directly written into the code. You are advised to store the AK and SK in ciphertext in the configuration file or environment variables and decrypt them when using them.

In this example, the AK and SK are stored in environment variables. Before running this example, specify the environment variables *HUAWEICLOUD_SDK_AK* and *HUAWEICLOUD_SDK_SK* in the local environment.

```
# coding: utf-8

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

def show_transcodings_template(client):
    try:
        request = huaweicloudsdklive.ShowTranscodingsTemplateRequest("play.example.huaweicloud.com")
        response = client.show_transcodings_template(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["HUAWEICLOUD_SDK_AK"]
sk = os.environ["HUAWEICLOUD_SDK_SK"]
endpoint = "{your endpoint}"
project_id = "{your project id}"

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

live_client = LiveAPIClient.new_builder(LiveAPIClient) \
    .with_http_config(config) \
    .with_credentials(credentials) \
    .with_endpoint(endpoint) \
    .build()

show_transcodings_template(live_client)
```


4 Go SDK User Guide

This section describes how to quickly integrate the Go SDK for development.

Prerequisites

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

  NOTE

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

  - Purchase and use cloud services in Huawei Cloud regions in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Plan to use Live in Huawei Cloud regions in the Chinese mainland.
- You have obtained and **added an ingest domain name and a streaming domain name** (both licensed) on the Live console, and **associated the domain names**.
- The development environment (Go 1.14 or later) is available.
- You have obtained the access key (AK) and secret access key (SK) of the Huawei Cloud account. You can create and view your AK and SK on the **My Credentials** > **Access Keys** page of the Huawei Cloud console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of Live. You can view the project ID on the **My Credentials** > **API Credentials** page of the Huawei Cloud console. For details, see **API Credentials**.

Installing an SDK

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

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

```
# Install the Huawei Cloud Go library.
go get github.com/huaweicloud/huaweicloud-sdk-go-v3
```



```
# Install dependencies.
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"
    live "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/live/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/live/v1/model"
    "net/http"
    "os"
)
```

Step 2 Configure client attributes.

1. Use the default configuration.

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

2. (Optional) Configure a proxy.

   ```
   // Configure the network proxy as required.
   // There will be huge security risks if the password of the proxy server is directly written into the code. You are advised to store the password in ciphertext in the configuration file or environment variables and decrypt the password when using it.
   // Before configuring a proxy, specify the environment variable PROXY_PASSWORD in the local environment.
   httpConfig.WithProxy(config.NewProxy().
       WithSchema("http").
       WithHost("proxy.huaweicloud.com").
       WithPort(80).
       WithUsername("testuser").
       WithPassword(os.Getenv("PROXY_PASSWORD")))
   ```

3. (Optional) Configure a connection.

   ```
   httpConfig.WithTimeout(30);
   ```

4. (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 and SK

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

  ```
  auth := basic.NewCredentialsBuilder().
              WithAk(ak).
              WithSk(sk).
              WithProjectId(projectId).
              Build()
  ```

- Temporary AK and SK

  Obtain a temporary AK, SK, and security token **through a token** or **through an agency**.

  ```
  auth := basic.NewCredentialsBuilder().
              WithAk(ak).
              WithSk(sk).
              WithProjectId(projectId).
              WithSecurityToken(securityToken).
              Build()
  ```

The related parameters are as follows:

- **ak**: AK of the Huawei Cloud account. You are advised to store the AK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **sk**: SK of the Huawei Cloud account. You are advised to store the SK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **projectId**: ID of the project where Live is provided. Select a project ID based on the region of the project.
- **securityToken**: security token used for temporary AK and SK authentication

Step 4 Initialize the client.

```
# Initialize the Live client.
client := live.NewLiveClient(
    live.LiveClientBuilder().
        WithEndpoints(endpoints).
        WithCredential(auth).
        WithHttpConfig(config.DefaultHttpConfig()).
        Build())
```

endpoint: regions where Live is used and endpoints of each service.

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.ShowTranscodingsTemplateRequest{
    Domain: "play.example.huaweicloud.com",
}
response, err := client.ShowTranscodingsTemplate(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.ShowTranscodingsTemplate(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.

There will be huge security risks if the AK and SK used for authentication are directly written into the code. You are advised to store the AK and SK in ciphertext in the configuration file or environment variables and decrypt them when using them.

In this example, the AK and SK stored in the environment variables are used. Specify the environment variables *HUAWEICLOUD_SDK_AK* and *HUAWEICLOUD_SDK_SK* in the local environment first.

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

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

client := live.NewLiveAPIClient(
    live.LiveAPIClientBuilder().
        WithEndpoints([]string{"{your endpoint}"}).
        WithCredential(
            basic.NewCredentialsBuilder().
                WithAk(os.Getenv("HUAWEICLOUD_SDK_AK")).
                WithSk(os.Getenv("HUAWEICLOUD_SDK_SK")).
                WithProjectId("{your project id}").
                  Build()).
        WithHttpConfig(config.DefaultHttpConfig().
            WithIgnoreSSLVerification(true).
            WithHttpHandler(httphandler.
                NewHttpHandler().
                    AddRequestHandler(RequestHandler).
                    AddResponseHandler(ResponseHandler))).
        Build())
```

----End

Sample Code

Before the calling, replace the variables *{your endpoint}* and *{your project id}* as needed.

There will be huge security risks if the AK and SK used for authentication are directly written into the code. You are advised to store the AK and SK in ciphertext in the configuration file or environment variables and decrypt them when using them.

In this example, the AK and SK are stored in environment variables. Before running this example, specify the environment variables *HUAWEICLOUD_SDK_AK* and *HUAWEICLOUD_SDK_SK* in the local environment.

```
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"
    live "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/live/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/live/v1/model"
    "net/http"
    "os"
)

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

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

func main() {
    client := live.NewLiveAPIClient(
        live.LiveAPIClientBuilder().
            WithEndpoints([]string{"{your endpoint}"}).
            WithCredential(
                basic.NewCredentialsBuilder().
                    WithAk(os.Getenv("HUAWEICLOUD_SDK_AK")).
                    WithSk(os.Getenv("HUAWEICLOUD_SDK_SK")).
                    WithProjectId("{your project id}").
                    Build()).
            WithHttpConfig(config.DefaultHttpConfig().
                WithIgnoreSSLVerification(true).
                WithHttpHandler(httphandler.
                    NewHttpHandler().
                        AddRequestHandler(RequestHandler).
                        AddResponseHandler(ResponseHandler))).
            Build())

    request := &model.ShowTranscodingsTemplateRequest{
        Domain: "play.example.huaweicloud.com",
    }
    response, err := client.ShowTranscodingsTemplate(request)
    if err == nil {
        fmt.Println("%+v\n",response)
    } else {
        fmt.Println(err)
    }
}
```


5 PHP SDK User Guide

This section describes how to quickly integrate the PHP SDK for development.

Prerequisites

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

  NOTE

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

  - Purchase and use cloud services in Huawei Cloud regions in the Chinese mainland. In this case, real-name authentication is required by the laws and regulations of the Chinese mainland.
  - Plan to use Live in Huawei Cloud regions in the Chinese mainland.
- You have obtained and **added an ingest domain name and a streaming domain name** (both licensed) on the Live console, and **associated the domain names**.
- The development environment (PHP 5.6 or later) is available.
- You have obtained the access key (AK) and secret access key (SK) of the Huawei Cloud account. You can create and view your AK and SK on the **My Credentials** > **Access Keys** page of the Huawei Cloud console. For details, see **Access Keys**.
- You have obtained the project ID of the corresponding region of Live. You can view the project ID on the **My Credentials** > **API Credentials** page of the Huawei Cloud console. For details, see **API Credentials**.

Installing an SDK

The Live server SDK supports PHP 5.6 or later. Run the **php --version** command to check the PHP version.

You are advised to install the SDK using Composer.

Composer is a tool for dependency management in PHP. It allows you to declare the libraries your project depends on and it will install them for you.

```
# Install Composer.
curl -sS https://getcomposer.org/installer | php
```



```
# Install the PHP SDK.
composer require huaweicloud/huaweicloud-sdk-php
```

After the installation is complete, you need to generate the necessary files that Composer will use for autoloading.

```
require 'path/to/vendor/autoload.php';
```

Procedure

Step 1 Import the dependent module.

```
namespace HuaweiCloud\SDK\Live\V1\Model;
require_once "vendor/autoload.php";
use HuaweiCloud\SDK\Core\Auth\BasicCredentials;
use HuaweiCloud\SDK\Core\Http\HttpConfig;
use HuaweiCloud\SDK\Core\Exceptions\ConnectionException;
use HuaweiCloud\SDK\Core\Exceptions\RequestTimeoutException;
use HuaweiCloud\SDK\Core\Exceptions\ServiceResponseException;
// Import the specified Live library.
use HuaweiCloud\SDK\Live\V1\LiveClient;
```

Step 2 Configure client attributes.

1. Use the default configuration.

   ```
   // Use the default configuration.
   $config = HttpConfig::getDefaultConfig();
   ```

2. (Optional) Configure a proxy.

   ```
   // (Optional) Use a proxy server.
   // There will be huge security risks if the password of the proxy server is directly written into the code. You are advised to store the password in ciphertext in the configuration file or environment variables and decrypt the password when using it.
   // Before configuring a proxy, specify the environment variable PROXY_PASSWORD in the local environment.
   $config->setProxyProtocol('http');
   $config->setProxyHost('proxy.huawei.com');
   $config->setProxyPort(8080);
   $config->setProxyUser('username');
   $config->setProxyPassword(getenv('PROXY_PASSWORD'));
   ```

3. (Optional) Configure a 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->setConnectionTimeout(3);
   ```

4. (Optional) Configure SSL.

   ```
   # (Optional) Skip server certificate verification.
   $config->setIgnoreSslVerification(true);
   # Configure the server CA certificate so that the SDK can verify the server certificate.
   $config->setCertFile("{yourCertFile}");
   ```

Step 3 Initialize authentication information.

You can use one of the following two authentication modes:

- Permanent AK and SK

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

  ```
  $basicCredentials = new BasicCredentials($ak,$sk,$projectId);
  ```

- Temporary AK and SK

  Obtain a temporary AK, SK, and security token **through a token** or **through an agency**.

  ```
  $basicCredentials = BasicCredentials(ak, sk, projectId).withSecurityToken(securityToken);
  ```

The related parameters are as follows:

- **ak**: AK of the Huawei Cloud account. You are advised to store the AK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **sk**: SK of the Huawei Cloud account. You are advised to store the SK in ciphertext in the configuration file or environment variables and decrypt it when using it.
- **project_id**: ID of the project where Live is provided. Select a project ID based on the region of the project.
- **securitytoken**: security token used for temporary AK and SK authentication

Step 4 Initialize the client.

```
// Initialize the Live client.
$client = LiveClient::newBuilder(new LiveClient)
  ->withHttpConfig($config)
  ->withEndpoint($endpoint)
  ->withCredentials($credentials)
  ->build();
$request = new CreateDomainRequest();
```

endpoint: regions where Live is used and endpoints of each service.

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 = new ShowTranscodingsTemplateRequest("play.example.huaweicloud.com");
$response = $client->ShowTranscodingsTemplate($request);
echo $response;
```

Step 6 Perform troubleshooting.

Table 5-1 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).


```
// Perform troubleshooting.
try {
  $response = $client->ShowTranscodingsTemplate($request);
} catch (ConnectionException $e) {
  $msg = $e->getMessage();
  echo "\n". $msg ."\n";
} catch (RequestTimeoutException $e) {
  $msg = $e->getMessage();
  echo "\n". $msg ."\n";
} catch (ServiceResponseException $e) {
  echo "\n";
  echo $e->getHttpStatusCode(). "\n";
  echo $e->getErrorCode() . "\n";
  echo $e->getErrorMsg() . "\n";
}
```

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.

```
$requestHandler = function ($argsMap) {
    if (isset($argsMap['request'])) {
        $sdkRequest = $argsMap['request'];
        $requestHeaders = $sdkRequest->headerParams;
        $requestBase = "> Request " . $sdkRequest->method . ' ' .
            $sdkRequest->url . "\n";
        if (count($requestHeaders) > 0) {
            $requestBase = $requestBase . '> Headers:' . "\n";
            foreach ($requestHeaders as $key => $value) {
                $requestBase = $requestBase . '    ' . $key . ' : ' .
                    $value . "\n";
            }
            $requestBase = $requestBase . '> Body: ' .
                $sdkRequest->body . "\n\n";
        }
        if (isset($argsMap['logger'])) {
            $logger = $argsMap['logger'];
            $logger->addDebug($requestBase);
        }
    }
};

$responseHandler = function ($argsMap) {
    if (isset($argsMap['response'])) {
        $response = $argsMap['response'];
        $responseBase = "> Response HTTP/1.1 " .
            $response->getStatusCode() . "\n";
        $responseHeaders = $response->getHeaders();
        if (count($responseHeaders) > 0) {
            $responseBase = $responseBase . '> Headers:' . "\n";
            foreach ($responseHeaders as $key => $value) {
                $valueToString = '';
                if (is_array($value)) {
                    $valueToString = ''.join($value);
```



```
            }
            $responseBase = $responseBase . '    ' . $key . ' : '
                . $valueToString . "\n";
        }
        $responseBody = $response->getBody();
        $responseBase = $responseBase . '> Body: ' . (string)
            $responseBody . "\n\n";
    }
    if (isset($argsMap['logger'])) {
        $logger = $argsMap['logger'];
        $logger->addDebug($responseBase);
    }
  }
};

$httpHandler = new HttpHandler();
$httpHandler->addRequestHandlers($requestHandler);
$httpHandler->addResponseHandlers($responseHandler);

$iamClient = LiveClient::newBuilder()
   ->withHttpConfig($config)
   ->withEndpoint($endpoint)
   ->withCredentials(null)
   ->withStreamLogger($stream = 'php://stdout',$logLevel =Logger::INFO) // Print logs to the console.
   ->withFileLogger($logPath='./test_log.txt', $logLevel = Logger::INFO) // Print logs to a file.
   ->withHttpHandler($httpHandler)
   ->build();
```

----End

Sample Code

Before the calling, replace the variables *{your endpoint string}* and *{your project id}* as needed.

There will be huge security risks if the AK and SK used for authentication are directly written into the code. You are advised to store the AK and SK in ciphertext in the configuration file or environment variables and decrypt them when using them.

In this example, the AK and SK are stored in environment variables. Before running this example, specify the environment variables *HUAWEICLOUD_SDK_AK* and *HUAWEICLOUD_SDK_SK* in the local environment.

```
<?php
namespace HuaweiCloud\SDK\Live\V1\Model;
require_once "vendor/autoload.php";
use HuaweiCloud\SDK\Core\Auth\BasicCredentials;
use HuaweiCloud\SDK\Core\Http\HttpConfig;
use HuaweiCloud\SDK\Core\Exceptions\ConnectionException;
use HuaweiCloud\SDK\Core\Exceptions\RequestTimeoutException;
use HuaweiCloud\SDK\Core\Exceptions\ServiceResponseException;
use HuaweiCloud\SDK\Live\V1\LiveClient;

$ak = getenv('HUAWEICLOUD_SDK_AK');
$sk = getenv('HUAWEICLOUD_SDK_SK');
$endpoint = "https://live.cn-north-4.myhuaweicloud.com";
$projectId = "";
$credentials = new BasicCredentials($ak,$sk,$projectId);
$config = HttpConfig::getDefaultConfig();
$config->setIgnoreSslVerification(true);

$client = LiveClient::newBuilder(new LiveClient)
  ->withHttpConfig($config)
  ->withEndpoint($endpoint)
  ->withCredentials($credentials)
```



```
  ->build();
$request = new ShowTranscodingsTemplateRequest();

try {
  $response = $client->ShowTranscodingsTemplate($request);
} catch (ConnectionException $e) {
  $msg = $e->getMessage();
  echo "\n". $msg ."\n";
} catch (RequestTimeoutException $e) {
  $msg = $e->getMessage();
  echo "\n". $msg ."\n";
} catch (ServiceResponseException $e) {
  echo "\n";
  echo $e->getHttpStatusCode(). "\n";
  echo $e->getErrorCode() . "\n";
  echo $e->getErrorMsg() . "\n";
}
echo "\n";
echo $response;
```


6 Change History

Released On	Description
2024-07-30	This issue is the first official release.

