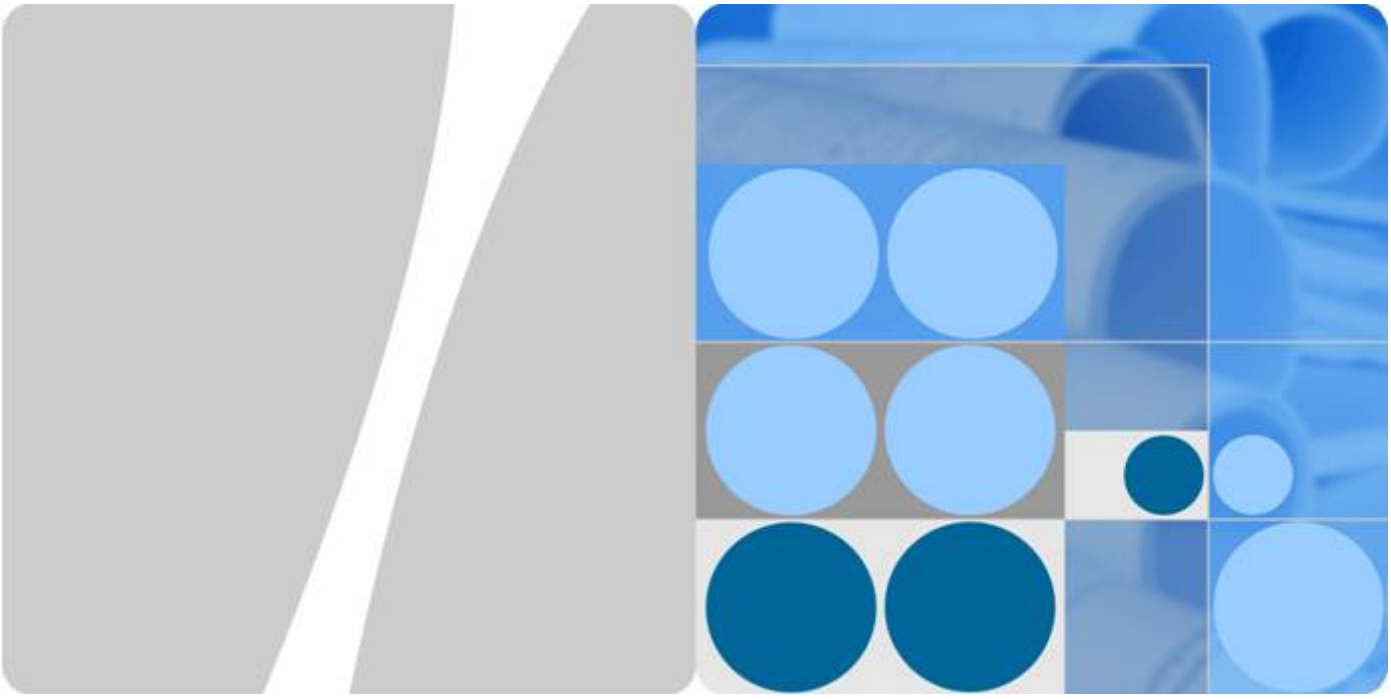


Data Lake Governance Center

SDK Reference

Issue 01

Date 2020-11-05

HUAWEI TECHNOLOGIES CO., LTD.

HUAWEI

Copyright © Huawei Technologies Co., Ltd. 2020. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions


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

Notice

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

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

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: https://www.huawei.com

Email: support@huawei.com

Contents

1 DIS SDK Reference....1
1.1 Overview....1
1.2 SDK Download....1
1.3 Compatibility....1
1.4 How Do I Check Software Package Integrity?....2
1.5 Obtaining Authentication Information....2
1.6 Using the Java SDK....3
1.6.1 Preparing the Environment....3
1.6.2 Connecting to kafka consumer....3
1.6.3 Configuring a Sample Project....3
1.6.4 Importing Third-party Dependency Packages....5
1.6.5 Initializing a DIS SDK Client Instance....5
1.6.6 Creating a Stream....6
1.6.7 Creating a Dump Task....7
1.6.8 Updating a Dump Task....12
1.6.9 Deleting a Dump Task....12
1.6.10 Querying a Dump Task List....13
1.6.11 Querying Dump Details....13
1.6.12 Deleting a Stream....14
1.6.13 Querying a Stream List....14
1.6.14 Querying Stream Details....15
1.6.15 Downloading Streaming Data....16
1.6.16 Uploading Streaming Data....25
1.6.17 Obtaining the Data Cursor....26
1.6.18 Creating an Application....26
1.6.19 Deleting an Application....26
1.6.20 Adding a Checkpoint....27
1.6.21 Querying a Checkpoint....27
1.6.22 Changing Partition Quantity....28
1.7 Using the C# SDK....28
1.7.1 Preparing the Installation Environment....28
1.7.2 Configuring a Sample Project....28
1.7.3 Initializing a DIS SDK Client Instance....32

 Copyright © Huawei Technologies Co., Ltd.

1.7.4 Creating a Stream 32
1.7.5 Deleting a Stream 35
1.7.6 Querying a Stream List 35
1.7.7 Querying Stream Details 36
1.7.8 Uploading Streaming Data 38
1.7.9 Downloading Streaming Data 40
1.7.10 Uploading a Real-Time File 42
1.7.11 Obtaining a Data Cursor 44
1.7.12 Creating a Consumer Application 45
1.7.13 Deleting a Consumer Application 45
1.7.14 Adding a Checkpoint 46
1.7.15 Querying a Checkpoint 47
1.7.16 Changing Partition Quantity 47
1.8 Using the Python SDK 48
1.8.1 Preparing the Installation Environment 48
1.8.2 Configuring a Sample Project 48
1.8.3 Initializing a DIS SDK Client Instance 49
1.8.4 Creating a Stream 50
1.8.5 Creating a Dump Task 50
1.8.6 Deleting a Stream 51
1.8.7 Deleting a Dump Task 51
1.8.8 Querying a Stream List 51
1.8.9 Querying a Dump List 52
1.8.10 Querying Stream Details 52
1.8.11 Querying Dump Details 52
1.8.12 Uploading Streaming Data in JSON Format 53
1.8.13 Uploading Streaming Data in Protobuf Format 53
1.8.14 Downloading Streaming Data 54
1.8.15 Creating an Application 55
1.8.16 Deleting an Application 55
1.8.17 Viewing Application Details 55
1.8.18 Querying an Application List 55
1.8.19 Adding a Checkpoint 56
1.8.20 Querying a Checkpoint 56
1.8.21 Changing Partition Quantity 56
1.8.22 Obtaining a Data Cursor 57
1.9 Using the C SDK 57
1.9.1 Preparing the Installation Environment 57
1.9.2 Configuring a Sample Project 57
1.9.3 Configuring an AK and SK 58
1.9.4 Initializing DIS 58
1.9.5 Creating a Stream Whose Source Data Type Is BLOB, JSON, or CSV 59

 Copyright © Huawei Technologies Co., Ltd.

1.9.6 Deleting a Stream 60
1.9.7 Querying a Stream List 61
1.9.8 Querying Stream Details 62
1.9.9 Uploading Streaming Data 65
1.9.10 Downloading Streaming Data 68
1.9.11 Obtaining a Data Cursor 71
1.10 DIS Server-Side Error Codes 74
2 Data Service SDK Reference 89
2.1 Overview 89
2.2 Calling APIs Through App Authentication 89
2.2.1 Preparation 89
2.2.2 Java 90
2.2.3 Go 102
2.2.4 Python 106
2.2.5 C# 110
2.2.6 JavaScript 111
2.2.7 PHP 116
2.2.8 C++ 120
2.2.9 C 122
2.2.10 Android 125
2.2.11 curl 127
2.2.12 Other Programming Languages 129
A Change History 135

 Copyright © Huawei Technologies Co., Ltd.

1 DIS SDK Reference

1.1 Overview

Introduction to DIS

DGCData Ingestion Service (DIS) addresses the challenge of transmitting data outside cloud services to cloud services. DIS builds data intake streams for custom applications capable of processing or analyzing streaming data. DIS continuously captures, transmits, and stores terabytes of data from hundreds of thousands of sources every hour, such as logs, Internet of Things (IoT) data, social media feeds, website clickstreams, and location-tracking events.

DIS features flexible expansion and allows infrastructure to be deployed in multiple regions, while delivering high reliability. Users can deploy DIS in specific regions based on site requirements to obtain rapid access speed at an affordable price.

Introduction to SDK

Data Ingestion Service Software Development Kit (DIS SDK) is the encapsulation of RESTful APIs provided by DIS to simplify user development. Users can directly use API functions provided by DIS SDK to obtain the DIS service capabilities.

1.2 SDK Download

Download **huaweicloud-sdk-dis-XXX-XXX.zip** from **https://dis-publish.obs-website.cn-north-1.myhuaweicloud.com/**.

1.3 Compatibility

Supported JDK versions: 1.8.0 and later versions

Supported Python versions: 2.7 and later versions

 Copyright © Huawei Technologies Co., Ltd.

1.4 How Do I Check Software Package Integrity?

This section describes how to verify integrity of the DIS SDK software package on a Linux system by using a verification file.

Prerequisites

- The PuTTY tool is available.
- The WinSCP tool is available.

Procedure

Step 1 Upload the DIS SDK software package **huaweicloud-sdk-dis-x.x.x1.2.3.zip** to any directory on the Linux system by using WinSCP.

NOTE

In **huaweicloud-sdk-dis-x.x.x.zip**, **x.x.x** indicates the version number of the DIS SDK software package.

Step 2 Log in to the Linux system by using PuTTY. In the directory in which **huaweicloud-sdk-dis-x.x.x1.2.3.zip** is stored, run the following command to obtain the verification code of the DIS SDK software package

sha256sum huaweicloud-sdk-dis-x.x.x.zip

Example verification code:

```
# sha256sum dis-sdk-x.x.x.zip
8be2c937e8d78b1a9b99777cee4e7131f8bf231de3f839cf214e7c5b5ba3c088 huaweicloud-sdk-dis-x.x.x.zip
```

Step 3 Open the DIS SDK verification file **huaweicloud-sdk-dis-x.x.x1.2.3.zip.sha256sum** and compare it with the verification code obtained in **Step 2**.

- If they are consistent, the DIS-SDK compression package has not been tampered with.
- If they are inconsistent, the DIS SDK software package is tampered with and you need to obtain it again.

----End

1.5 Obtaining Authentication Information

Obtaining AK/SK

To use DIS, you need a valid set of Access Key ID/Secret Access Key (AK/SK) for signature authentication. The AK/SK is used to call APIs. You can download the AK/SK from **My Credential** > **Access Keys**.

Obtaining the Project ID

A project ID identifies a tenant's resources. You can choose **My Credential** > **Projects** to view the IDs of different projects.

 Copyright © Huawei Technologies Co., Ltd.

Obtaining Regions and Endpoints

For details about regions and endpoints, see **Regions and Endpoints**.

1.6 Using the Java SDK

1.6.1 Preparing the Environment

- Download JDK1.8 or a later version from the **Oracle official website** and install it, and configure Java environment variables.
- Download Eclipse IDE for Java Developers of the latest version from the **Eclipse's official website**, and install it.
- Configure the JDK in Eclipse.

1.6.2 Connecting to kafka consumer

The DIS SDK provides compatibility with native kafka client APIs.

1.6.3 Configuring a Sample Project

The **huaweicloud-sdk-dis-java-*X.X.X*.zip** package downloaded from **SDK Download** provides a sample project. You can use a development tool (such as Eclipse) to compile and run the sample project on a local device. You can also develop applications based on the sample project. The sample project code is available in the **\dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo** directory.

Sample Code	Describes
ConsumerDemo.java	How to download data.
ProducerDemo.java	How to upload data.

Procedure

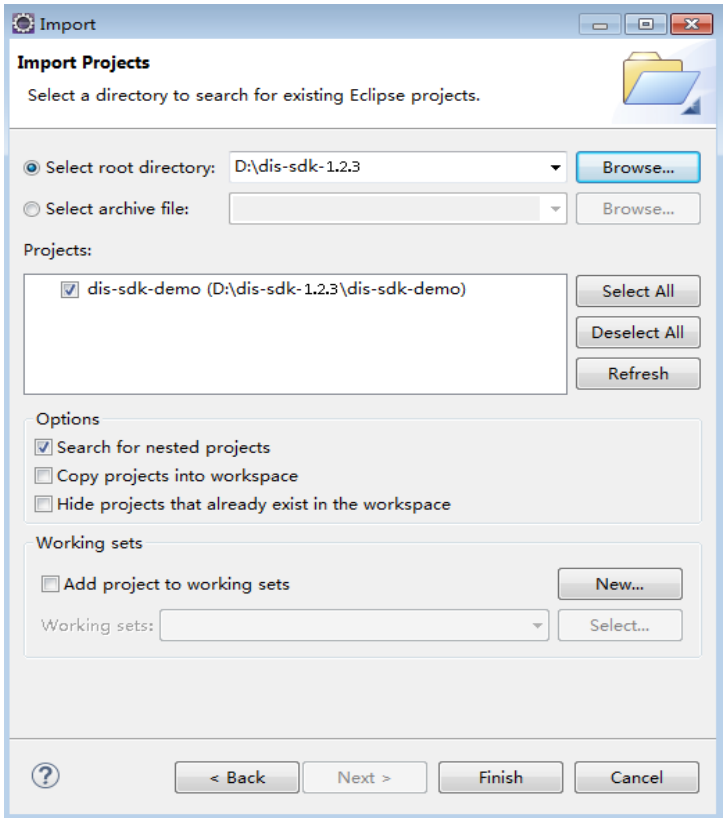
Step 1 Decompress the **huaweicloud-sdk-dis-java-*X.X.X*.zip** package downloaded from **SDK Download** to obtain the **dis-sdk-demo** package and sample project.

Step 2 Import the Eclipse project.

1. Start Eclipse. Choose **File** > **Import**. The **Import** dialog box is displayed.
2. Choose **Maven** > **Existing Maven Projects**, and click **Next**. The **Import** dialog box is displayed.
3. Click **Browse** and select a root directory for the **dis-sdk-demo** sample project. In the **Projects** area, select a sample project.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-1 Import Maven Projects



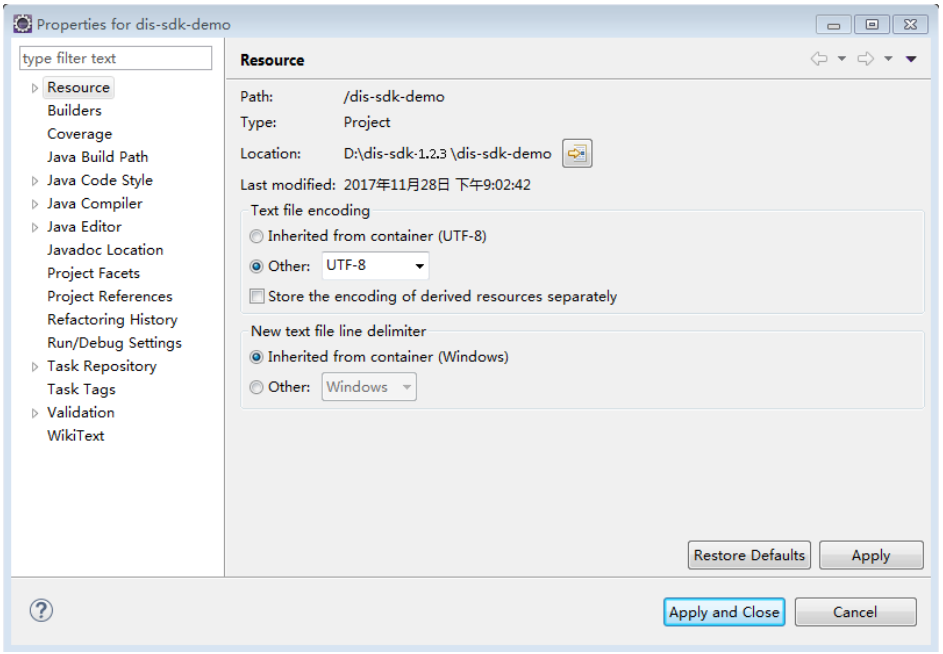
4. Click **Finish**.

Step 3 Configure the demo project.

1. Set the project code to UTF-8.
   a. In the navigation tree on the left, right-click the required project under **Project Explorer** and choose **Properties** from the shortcut menu. The **Properties for dis-sdk-demo** page is displayed.
   b. In the left list, select **Resource**. The **Resource** pane is displayed.
   c. Set **Text file encoding** to **Other**. In the **Others** drop-down list, select **UTF-8**.
   d. Click **Apply and Close** to complete the encoding configuration.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-2 Selecting Resource



2. Add a JDK.
   a. In the navigation tree on the left, right-click the required project under **Project Explorer** and choose **Properties** from the shortcut menu. The **Properties for dis-sdk-demo** page is displayed.
   b. In the left list, select **Java Build Path**. The **Java Build Path** pane is displayed.
   c. Click the **Libraries** tab, and then click **Add Library**. The **Add Library** dialog box is displayed.
   d. Select **JRE System Library** and click **Next**. Ensure that the version of **Workspace default JRE** is jdk1.8 or later.
   e. Click **Finish** to exit the **Add Library** dialog box.
   f. Click **Apply and Close** to add the JDK.

----End

1.6.4 Importing Third-party Dependency Packages

Step 1 In the navigation tree, click **Project Explorer**, right-click a project and choose **Properties > Java Build Path > Add External JARs**.

Step 2 Locate the directory containing the decompressed **dis-sdk** folder, and import all the files with the extension **JAR** in the current directory and the **third_party** folder to the project.

----End

1.6.5 Initializing a DIS SDK Client Instance

You can initialize the DIS SDK client instance using either of the following methods. The first method is recommended. For details about **endpoint**, **ak**, **sk**, **region**, and **projectId**, see **Obtaining Authentication Information**.

 Copyright © Huawei Technologies Co., Ltd.

- Use codes to initialize the DIS SDK client instance. The **ProducerDemo.java** file provides the sample code.

```
//Create a DIS client instance.
        DIS dic = DISClientBuilder.standard()
            .withEndpoint("https://ip:port")
            .withAk("xxxx")
            .withSk("xxxx")
            .withProjectId("xxxxxxx")
            .withRegion("******")
            .build();
```

- Perform the following steps to initialize the DIS client instance if a proxy needs to be used:

```
//Create a DIS client instance.
        DIS dic = DISClientBuilder.standard()
            .withEndpoint("https://ip:port")
            .withAk("xxxx")
            .withSk("xxxx")
            .withProjectId("xxxxxxx")
            .withRegion("******")
.withProxyHost("YourProxyHost") //Proxy IP address
.withProxyPort("YourProxyPort") //Proxy port
.withProxyProtocol(Protocol.HTTPS) //Proxy protocol. The default value is HTTP.
.withProxyUsername("YourProxyUserName") //Proxy username (optional)
.withProxyPassword("YourProxyPassword") // Proxy password (optional)
            .build();
```

- If data needs to be encrypted before being uploaded to DIS on the client, use the encryption method provided by the DIS SDK. That is, add the **DataEncryptEnabled** and **data.password** parameters when building the disclient.

```
//Create a DIS client instance.
        DIS dic = DISClientBuilder.standard()
            .withEndpoint("https://ip:port")
            .withAk("xxxx")
            .withSk("xxxx")
            .withProjectId("xxxxxxx")
            .withRegion("******")
            .withDataEncryptEnabled(true)
.withProperty("data.password", "xxx")//xxx indicates the data encryption key configured by the user.
            .build();
```

  NOTE

  If JAVA SDK is used to encrypt the uploaded data, you also need to use JAVA SDK to configure the same key for data reading.

- Use the configuration file to initialize the DIS SDK client instance.

  Add the following configuration items to the **dis.properties** file in the **dis-sdk-demo\src\main\resources** directory:

  – ak/sk: AK/SK created on the IAM
  – region: area where the stream is used
  – endpoint: access address of the DIS
  – projectId: project ID of the stream

```
//Create a DIS SDK client instance.
        DIS dic = DISClientBuilder.standard().build();
```

1.6.6 Creating a Stream

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

 Copyright © Huawei Technologies Co., Ltd.

When you use the DIS SDK to create a DIS stream, specify the stream name, number of partitions in the stream, and stream type.

STREAM_TYPE_COMMON indicates a common stream, and **STREAM_TYPE_ADVANCED** indicates an advanced stream.

```
CreateStreamRequest createStreamRequest = new CreateStreamRequest();
//Configure a stream name.
String streamName = "myStream";
createStreamRequest.setStreamName(streamName);
//Configure a stream type. The value can be Common or Advanced.
createStreamRequest.setStreamType(StreamType.COMMON.name());
//Configure the number of partitions in the stream.
createStreamRequest.setPartitionCount(3);
//Configure the retention period of the stream in units of hours. The value is N x 24, where N ranges from 1 to 7.
createStreamRequest.setDataDuration(24);
//Configure the source data type of the stream. Default value: BLOB
createStreamRequest.setDataType(DataTypeEnum.BLOB.name());
```

After configuring **CreateStreamRequest**, you can create a stream by calling createStream.

```
dic.createStream(createStreamRequest);
```

1.6.7 Creating a Dump Task

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

When using the DIS SDK to create a dump task, you need to specify the stream name, dump task name, dump interval, and dump destination. The sample code is available in the **TransferTaskDemo.java** file under the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo\example** directory.

Creating an OBS Dump Task

```
CreateTransferTaskRequest request = new CreateTransferTaskRequest();

//Configure the stream name. You can create streams on the DIS console.
request.setStreamName(streamName);

//Configure the dump task name.
OBSDestinationDescriptorRequest descriptor = new OBSDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure the OBS bucket name and folder name. You can create OBS buckets and files on the OBS console or client.
descriptor.setObsBucketPath("obs-dis");
descriptor.setFilePrefix("transfertask");

//Configure the dump interval that is expressed in units of seconds.
descriptor.setDeliverTimeInterval(900);

//(Optional) Create an IAM agency named dis_admin_agency on the DIS management page and use it to access specific cloud services. For the first time to create an IAM agency, you have to authorize it.
descriptor.setAgencyName("dis_admin_agency");

//(Optional) Configure the dump file format. By default, the value is Text. Other available options are Parquet and CarbonData.
descriptor.setDestinationFileType(DestinationFileTypeEnum.TEXT.getType());

//Configure the initial offset when data is pulled from the DIS stream. The value can be LATEST or TRIM_HORIZON. LATEST is the default value, indicating that data is read from the latest uploaded records in the stream. TRIM_HORIZON indicates that data is read from the earliest unexpired records in the stream.
descriptor.setConsumerStrategy(PartitionCursorTypeEnum.LATEST.name());
```

 Copyright © Huawei Technologies Co., Ltd.

```
request.setObsDestinationDescriptor(descriptor);
```

After configuring **CreateTransferTaskRequest**, you can call the **createTransferTask** method to create the dump task.

```
dic.createTransferTask(request);
```

Create an MRS Dump Task

```
CreateTransferTaskRequest request = new CreateTransferTaskRequest();

//Configure the stream name. You can create streams on the DIS console.
request.setStreamName(streamName);

//Configure the dump task name.
MRSDestinationDescriptorRequest descriptor = new MRSDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure the MRS cluster information about the cluster name and ID. You can create and query cluster information on the MRS console. The cluster must be in non-security mode.
descriptor.setMrsClusterName("mrs_dis");
descriptor.setMrsClusterId("fe69a732-c7d3-4b0f-8cda-ec9eca0cf141");

//Configure the OBS bucket and folder that is used to temporarily store the data to be dumped to MRS and the dump failure data. You can create OBS buckets and folders on the OBS console.
descriptor.setObsBucketPath("obs-dis");
descriptor.setFilePrefix("transfertask");

//Configure the dump interval that is expressed in units of seconds.
descriptor.setDeliverTimeInterval(900);

//(Optional) Create an IAM agency named dis_admin_agency on the DIS management page and use it to access specific cloud services. For the first time to create an IAM agency, you have to authorize it.
descriptor.setAgencyName("dis_admin_agency");

//Configure the dump file format. By default, the value is Text. Other available options are Parquet and CarbonData.
descriptor.setDestinationFileType(DestinationFileTypeEnum.TEXT.getType());

//Configure the initial offset when data is pulled from the DIS stream. The value can be LATEST or TRIM_HORIZON. LATEST is the default value, indicating that data is read from the latest uploaded records in the stream. TRIM_HORIZON indicates that data is read from the earliest unexpired records in the stream.
descriptor.setConsumerStrategy(PartitionCursorTypeEnum.LATEST.name());

request.setMrsDestinationDescriptor(descriptor);
```

After configuring **CreateTransferTaskRequest**, you can call the **createTransferTask** method to create the dump task.

```
dic.createTransferTask(request);
```

Creating a DLI Dump Task

```
CreateTransferTaskRequest request = new CreateTransferTaskRequest();

//Configure the stream name. You can create streams on the DIS console.
request.setStreamName(streamName);

//Configure the dump task name.
UqueryDestinationDescriptorRequest descriptor = new UqueryDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure DLI information about the database and internal table names. You can create and query DLI on the DLI console The DLI table must be the internal table.
descriptor.setDliDatabaseName("dis_dli");
descriptor.setDliTableName("dis_test");
```

 Copyright © Huawei Technologies Co., Ltd.

```
//Configure the OBS bucket and folder that is used to temporarily store the data to be dumped to DLI and the dump failure data. You can create OBS buckets and folders on the OBS console or client.
descriptor.setObsBucketPath("obs-dis");
descriptor.setFilePrefix("transfertask");

//Configure the dump interval that is expressed in units of seconds.
descriptor.setDeliverTimeInterval(900);

//(Optional) Create an IAM agency named dis_admin_agency on the DIS management page and use it to access specific cloud services. For the first time to create an IAM agency, you have to authorize it.
descriptor.setAgencyName("dis_admin_agency");

//Configure the initial offset when data is pulled from the DIS stream. The value can be LATEST or TRIM_HORIZON. LATEST is the default value, indicating that data is read from the latest uploaded records in the stream. TRIM_HORIZON indicates that data is read from the earliest unexpired records in the stream.
descriptor.setConsumerStrategy(PartitionCursorTypeEnum.LATEST.name());

request.setDliDestinationDescriptor(descriptor);
```

After configuring **CreateTransferTaskRequest**, you can call the **createTransferTask** method to create the dump task.

```
dic.createTransferTask(request);
```

Creating a DWS Dump Task

```
CreateTransferTaskRequest request = new CreateTransferTaskRequest();

//Configure the stream name. You can create streams on the DIS console.
request.setStreamName(streamName);

//Configure the dump task name.
DwsDestinationDescriptorRequest descriptor = new DwsDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure the DWS cluster information about the cluster name, ID, and database. You can create and query clusters on the DWS console, and create tables using its client or other methods.
descriptor.setDwsClusterName("dis_test");
descriptor.setDwsClusterId("92f90f6a-de4d-4689-82f6-320c328b0062");
descriptor.setDwsDatabaseName("postgres");
descriptor.setDwsSchema("dbadmin");
descriptor.setDwsTableName("distable01");
descriptor.setDwsDelimiter("|");
descriptor.setUserName("dbadmin");
descriptor.setUserPassword("xxxx");

//Call KMS to encrypt the DWS password to keep user data secure. You can create and query KMS on the KMS console.
descriptor.setKmsUserKeyName("qiyinshan");
descriptor.setKmsUserKeyId("9521c600-64a8-4971-ad36-7bbfa6d00c41");

//Configure the OBS bucket and folder that is used to temporarily store the data to be dumped to DWS and the dump failure data. You can create OBS buckets and folders on the OBS console or client.
descriptor.setObsBucketPath("obs-dis");
descriptor.setFilePrefix("transfertask");

//Configure the dump interval that is expressed in units of seconds.
descriptor.setDeliverTimeInterval(900);

//(Optional) Create an IAM agency named dis_admin_agency on the DIS management page and use it to access specific cloud services. For the first time to create an IAM agency, you have to authorize it.
descriptor.setAgencyName("dis_admin_agency");

//Configure the initial offset when data is pulled from the DIS stream. The value can be LATEST or TRIM_HORIZON. LATEST is the default value, indicating that data is read from the latest uploaded records in the stream. TRIM_HORIZON indicates that data is read from the earliest unexpired records in the stream.
descriptor.setConsumerStrategy(PartitionCursorTypeEnum.LATEST.name());

request.setDwsDestinationDescriptor(descriptor);
```

 Copyright © Huawei Technologies Co., Ltd.

After configuring **CreateTransferTaskRequest**, you can call the **createTransferTask** method to create the dump task.

```
dic.createTransferTask(request);
```

Creating a CloudTable HBase Dump Task

```
CreateTransferTaskRequest request = new CreateTransferTaskRequest();

//Configure the stream name. You can create streams on the DIS console.
request.setStreamName(streamName);

//Configure the dump task name.
CloudtableDestinationDescriptorRequest descriptor = new CloudtableDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure the CloudTable cluster information about the cluster name, ID, and database. You can create and query clusters on the CloudTable console, and create tables using its client or other methods.
descriptor.setCloudtableClusterName("dis_test");
descriptor.setCloudtableClusterId("92f90f6a-de4d-4689-82f6-320c328b0062");
descriptor.setCloudtableTableName("dis");

//To configure CloudtableSchema, see CloudTable User Guide.
CloudtableSchema cloudtableSchema = new CloudtableSchema();
List<SchemaField> rowKeySchema = new ArrayList<>();
SchemaField field1 = new SchemaField();
field1.setValue("id");
field1.setType("String");
rowKeySchema.add(field1);
SchemaField rField1 = new SchemaField();
rField1.setValue("group.users.id");
rField1.setType("String");
rowKeySchema.add(rField1);
List<SchemaField> columnsSchema = new ArrayList<>();
SchemaField field2 = new SchemaField();
field2.setColumnFamilyName("user");
field2.setQualifierName("id");
field2.setValue("group.users.id");
field2.setType("String");
SchemaField field3 = new SchemaField();
field3.setColumnFamilyName("user");
field3.setQualifierName("age");
field3.setValue("group.users.age");
field3.setType("Int");
columnsSchema.add(field2);
columnsSchema.add(field3);
cloudtableSchema.setRowKeySchema(rowKeySchema);
cloudtableSchema.setColumnsSchema(columnsSchema);
descriptor.setCloudtableSchema(cloudtableSchema);

//Configure the OBS bucket and folder that is used to store dump failure data. You can create OBS buckets and folders on the OBS console or client.
descriptor.setObsBackupBucketPath("obs-dis");
descriptor.setBackupfilePrefix("transfertask");

//(Optional) Create an IAM agency named dis_admin_agency on the DIS management page and use it to access specific cloud services. For the first time to create an IAM agency, you have to authorize it.
descriptor.setAgencyName("dis_admin_agency");

//Configure the initial offset when data is pulled from the DIS stream. The value can be LATEST or TRIM_HORIZON. LATEST is the default value, indicating that data is read from the latest uploaded records in the stream. TRIM_HORIZON indicates that data is read from the earliest unexpired records in the stream.
descriptor.setConsumerStrategy(PartitionCursorTypeEnum.LATEST.name());

request.setCloudtableDestinationDescriptor(descriptor);
```

After configuring **CreateTransferTaskRequest**, you can call the **createTransferTask** method to create the dump task.

 Copyright © Huawei Technologies Co., Ltd.

```
dic.createTransferTask(request);
```

Creating a CloudTable OpenTSDB Dump Task

```
CreateTransferTaskRequest request = new CreateTransferTaskRequest();

//Configure the stream name. You can create streams on the DIS console.
request.setStreamName(streamName);

//Configure the dump task name.
CloudtableDestinationDescriptorRequest descriptor = new CloudtableDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure the CloudTable OpenTSDB cluster information about the cluster name, ID, and database. You can create and query clusters on the CloudTable console, and create tables using its client or other methods.
descriptor.setCloudtableClusterName("dlf_test");
descriptor.setCloudtableClusterId("92f90f6a-de4d-4689-82f6-320c328b0062");

//To configure OpenTSDBSchema, see CloudTable User Guide.
List<SchemaField> metricSchema = new ArrayList<>();
SchemaField field1 = new SchemaField();
field1.setValue("group.users.id");
field1.setType("String");
metricSchema.add(field1);
SchemaField timestampSchema = new SchemaField();
timestampSchema.setColumnFamilyName("user");
timestampSchema.setFormat("yyyy/MM/dd HH:mm:ss");
timestampSchema.setValue("group.users.birthday");
timestampSchema.setType("String");
SchemaField valueSchema = new SchemaField();
valueSchema.setValue("group.users.age");
valueSchema.setType("Int");
List<SchemaField> tagsSchema = new ArrayList<>();
SchemaField field2 = new SchemaField();
field2.setName("group.users.id");
field2.setValue("group.users.id");
field2.setType("String");
SchemaField field3 = new SchemaField();
field3.setName("age");
field3.setValue("group.users.age");
field3.setType("Int");
tagsSchema.add(field2);
tagsSchema.add(field3);
OpenTSDBSchema openTSDBSchema = new OpenTSDBSchema();
openTSDBSchema.setMetricSchema(metricSchema);
openTSDBSchema.setTimestampSchema(timestampSchema);
openTSDBSchema.setValueSchema(valueSchema);
openTSDBSchema.setTagsSchema(tagsSchema);
List<OpenTSDBSchema> openTSDBSchemaList = new ArrayList<>();
openTSDBSchemaList.add(openTSDBSchema);
descriptor.setOpentsdbSchema(openTSDBSchemaList);

//Configure the OBS bucket and folder that is used to store dump failure data. You can create OBS buckets and folders on the OBS console or client.
descriptor.setObsBackupBucketPath("obs-dis");
descriptor.setBackupfilePrefix("transfertask");

//(Optional) Create an IAM agency named dis_admin_agency on the DIS management page and use it to access specific cloud services. For the first time to create an IAM agency, you have to authorize it.
descriptor.setAgencyName("dis_admin_agency");

//Configure the initial offset when data is pulled from the DIS stream. The value can be LATEST or TRIM_HORIZON. LATEST is the default value, indicating that data is read from the latest uploaded records in the stream. TRIM_HORIZON indicates that data is read from the earliest unexpired records in the stream.
descriptor.setConsumerStrategy(PartitionCursorTypeEnum.LATEST.name());

request.setCloudtableDestinationDescriptor(descriptor);
```

 Copyright © Huawei Technologies Co., Ltd.

After configuring **CreateTransferTaskRequest**, you can call the **createTransferTask** method to create the dump task.

```
dic.createTransferTask(request);
```

1.6.8 Updating a Dump Task

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

When using the DIS SDK to update a dump task, you need to specify the stream name, dump task name, dump interval, and dump destination. The sample code is available in the **TransferTaskDemo.java** file under the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo\example** directory.

```
//Configure global parameters of the dump task. A single-parameter update is not supported.
UpdateTransferTaskRequest request = new UpdateTransferTaskRequest();

//Configure the name of the stream to which the dump task to be updated belongs.
request.setStreamName(streamName);

//Configure the name of the dump task to be updated.
OBSDestinationDescriptorRequest descriptor = new OBSDestinationDescriptorRequest();
descriptor.setTransferTaskName(taskName);

//Configure the OBS bucket name and folder name. You can create OBS buckets and files on the OBS console or client.
descriptor.setObsBucketPath("obs-dis1");
descriptor.setFilePrefix("transfertask");

//Configure the dump interval that is expressed in units of seconds.
descriptor.setDeliverTimeInterval(300);

//(Optional) Configure the dump file format. By default, the value is Text. Other available options are Parquet and CarbonData.
descriptor.setDestinationFileType(DestinationFileTypeEnum.TEXT.getType());

request.setObsDestinationDescriptor(descriptor);
```

After configuring **UpdateTransferTaskRequest**, you can call the **updateTransferTask** method to update the dump task.

```
dic.updateTransferTask(request);
```

1.6.9 Deleting a Dump Task

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

You can use the DIS SDK to delete a specified dump task. The sample code is available in the **TransferTaskDemo.java** file under the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo\example** directory.

```
DeleteTransferTaskRequest request = new DeleteTransferTaskRequest();

//Configure the name of the stream to which the dump task belongs.
request.setStreamName(streamName);

//Configure the name of the dump task to be deleted.
request.setTransferTaskName(taskName);
```

After configuring **DeleteTransferTaskRequest**, you can call the **deleteTransferTask** method to create the dump task.

```
dic.deleteTransferTask(request);
```

 Copyright © Huawei Technologies Co., Ltd.

1.6.10 Querying a Dump Task List

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

You can use the DIS SDK to query the dump task list of a specified stream. The sample code is available in the **TransferTaskDemo.java** file under the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo\example** directory.

```
ListTransferTasksRquest request = new ListTransferTasksRquest();

//Configure the name of the stream to be queried.
request.setStreamName(streamName);
```

After configuring **ListTransferTaskRequest**, you can call the **listTransferTask** method to query the dump task list of a specified stream.

```
ListTransferTasksResult result = dic.listTransferTasks(request);
```

Information similar to the following is displayed when you query the dump task list:

```
{
   "tasks":[
      {
         "destination_type":"DLI",
         "task_name":"task_Ztab",
         "create_time":1552457808502,
         "state":"RUNNING",
         "last_transfer_timestamp":1552458085454
      },
      {
         "destination_type":"OBS",
         "task_name":"task_qTd9",
         "create_time":1552355757885,
         "state":"RUNNING",
         "last_transfer_timestamp":1552458158527
      }
   ],
   "total_number":2
}
```

1.6.11 Querying Dump Details

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

You can use the DIS SDK to query the details of a specified dump task. The sample code is available in the **TransferTaskDemo.java** file under the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo\example** directory.

```
DescribeTransferTaskRequest request = new DescribeTransferTaskRequest();

//Configure the name of the stream to be queried.
request.setStreamName(streamName);

//Configure the name of the dump task to be queried.
request.setTransferTaskName(taskName);
```

After configuring **DescribeTransferTaskRequest**, you can call the **describeTransferTask** method to query the details of a specified dump task.

```
DescribeTransferTaskResult result = dic.describeTransferTask(request);
```

The returned dump details are as follows:

 Copyright © Huawei Technologies Co., Ltd.

```
{
   "partitions":[
      {
         "partitionId":"shardId-0000000000",
         "discard":0,
         "state":"RUNNING",
         "last_transfer_timestamp":1552458085454,
         "last_transfer_offset":56
      }
   ],
   "stream_name":"dis_test1",
   "task_name":"task_Ztab",
   "task_id":"gGGu2WN88XbmRTm64nJ",
   "destination_type":"DLI",
   "state":"RUNNING",
   "create_time":1552457808502,
   "last_transfer_timestamp":1552458085454,
   "dli_destination_description":{
      "agency_name":"dis_admin_agency",
      "file_prefix":"dli",
      "obs_bucket_path":"dis.test.not.delete",
      "deliver_time_interval":300,
      "consumer_strategy":"LATEST"
   }
}
```

1.6.12 Deleting a Stream

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to delete a specified DIS stream.

```
//Specify the name of the stream to be deleted.
String streamName = "myStream";
DeleteStreamRequest deleteStreamRequest = new DeleteStreamRequest();
deleteStreamRequest.setStreamName(streamName);
```

After configuring **DeleteStreamRequest**, you can delete the stream by calling deleteStream.

```
dic.deleteStream(deleteStreamRequest);
```

1.6.13 Querying a Stream List

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

You can use the DIS SDK to query active streams.

Use the **setLimit** method to set the number of streams returned each time. If **setLimit** is not specified, a maximum of 10 streams are returned by default.

```
ListStreamsRequest listStreamsRequest = new ListStreamsRequest();
listStreamsRequest.setLimit(5);
System.out.println("listStream: " + JsonUtils.objToJson(dic.listStreams(listStreamsRequest)));
```

 Copyright © Huawei Technologies Co., Ltd.

Table 1-1 Parameters

Parameter	Type	Description
limit	long	The maximum number of DIS streams to list in a single API call. Value range: 1 to 100 Default value: 10
exclusiveStartStream-Name	string	Name of the DIS stream to start the stream list with. The returned stream list does not contain this DIS stream name. If pagination query is required, this parameter is not transferred when you query data on the first page. If the value of **has_more_streams** is **true**, the query is performed on the next page. The value of **exclusiveStartStreamName** is the name of the last stream in the query result of the first page.

NOTE

In this demo, **start_Stream_Name** is defined as a stream name before **stream0**, and **limit** is set to **5**. The following information is returned:

```
listStream: {"total_number":20,"stream_names":
["Stream0","Stream1","Stream2","Stream3","Stream4"],"has_more_streams":true}
```

Table 1-2 Response parameter description

Parameter	Type	Description
total_number	Int	Total number of all the DIS streams created by the current tenant.
stream_names	List<String>	List of the streams meeting the current requests.
has_more_streams	Boolean	Specify whether there are more matching DIS streams to list. Possible values: • **true**: There are more streams. • **false**: There are no more partitions.

1.6.14 Querying Stream Details

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

 Copyright © Huawei Technologies Co., Ltd.

Use the DIS SDK to query the details about a specified stream.

```
String streamName = "myStream";
DescribeStreamRequest describeStreamRequest = new DescribeStreamRequest();
describeStreamRequest.setStreamName(streamName);
System.out.println("descStream: " + JsonUtils.objToJson(dic.describeStream(describeStreamRequest)));
```

The returned stream details are as follows:

```
descStream: DescribeStreamResult [streamId=JnYbsMfWNn81e8n2mOC, streamName=myStream,
createTime=1540977519187, lastModifiedTime=1540977519983, retentionPeriod=24, status=RUNNING,
streamType=ADVANCED, dataType=BLOB, writablePartitionCount=2, readablePartitionCount=2,
partitions=[PartitionResult{partitionId='shardId-0000000000', hashRange='[0 : 4611686018427387902]',
status='ACTIVE', parentPartitionIds='null', sequenceNumberRange='[0 : 13286444]'},
PartitionResult{partitionId='shardId-0000000001', hashRange='[4611686018427387903 :
9223372036854775807]', status='ACTIVE', parentPartitionIds='null', sequenceNumberRange='[0 :
13288589]'}], hasMorePartitions=false, updatePartitionCounts=null]
```

1.6.15 Downloading Streaming Data

Context

To download streaming data, you need to determine the position where the data is obtained from the partition, that is, to obtain the cursor. After the start position is determined, the data is obtained cyclically.

There are five cursor types available:

- AT_SEQUENCE_NUMBER
- AFTER_SEQUENCE_NUMBER
- TRIM_HORIZON
- LATEST
- AT_TIMESTAMP

To better understand the cursor type, you need to understand the following basic concepts:

- A sequence number is the unique identifier of each record. DIS automatically allocates a sequence number when the data producer calls the PutRecords operation to add data to the DIS stream. The sequence number of the same partition key usually changes with time. A longer interval between PutRecords requests results in a larger sequence number.
- Sequence number of each partition increases from 0. Each data record corresponds to a unique sequence number. As a lifecycle ends, the sequence number expires. For example, after a data record is uploaded to a new partition and its sequence number starts from **0**. After 100 data records are uploaded, the sequence number of the last data record is **99**. When the lifecycle ends, sequence numbers 0 to 99 become unavailable.
- The sequence number range of a partition can be obtained by calling the **describeStream** API for querying stream details. **sequenceNumberRange** indicates the data validity range. The first value is the sequence number of the earliest data, the last value is the sequence number of the next uploaded data, and the sequence number of the latest data is one less than the last value.

  For example, [100, 200] indicates that a total of 200 data records have been uploaded to the partition, data records 0 to 99 have expired, the earliest valid

 Copyright © Huawei Technologies Co., Ltd.

data record is 100, the latest data record is 199, and the sequence number of the next data record to be uploaded is 200.

Scenario Description

The following table describes the application scenarios of the five cursor types:

Table 1-3 Scenario Description

Cursor Type	Description	Application Scenario	Remarks
AT_SEQUENCE_NUMBER	Data is read from the position denoted by a specific sequence number. The sequence number is defined by **starting-sequence-number** in the demo. This is the default cursor type.	This type of cursor is applicable to the scenario where the start sequence number has been specified.	It is closely related to sequenceNumber and sequenceNumber Range[A and B] of the partition data. The specified sequence number must meet the following condition: A<=sequenceNumber<=B

 Copyright © Huawei Technologies Co., Ltd.

Cursor Type	Description	Application Scenario	Remarks
AFTER_SEQUENCE_NUMBER	Data is read from the position right after the position denoted by a specific sequence number. The sequence number is defined by **starting-sequence-number**.	This type of cursor is applicable to the scenario where the last consumption position is saved. For example, each consumption record is saved to a file or checkpoint. If the program is restarted, data will be restored from the position right after the save position. Comparatively, **AT_SEQUENCE_NUMBER** will restore from the save position, leading to a data duplicate.	It is closely related to sequenceNumber and sequenceNumber Range[A and B] of the partition data. The specified sequence number must meet the following condition: (A-1)<=sequence Number<=(B-1)
TRIM_HORIZON	Data is read from the earliest data record in the partition. For example, if sequenceNumber Range [100, 200] is applied, data consumption starts from record 100.	This type of cursor is applicable to scenarios where the consumption position is unknown and all valid data in the partition will be consumed.	None.

 Copyright © Huawei Technologies Co., Ltd.

Cursor Type	Description	Application Scenario	Remarks
LATEST	Data is read from the position just after the most recent record in the partition. That is, the system does not read the existing data in the partition but starts from the data to be uploaded. For example, if sequenceNumber Range [100, 200] is applied, data consumption starts from record 200. If no data is uploaded, the obtained data is empty. If the data is uploaded, record 200, 201, 202... will be obtained.	This type of cursor is applicable to the scenario where the consumption position is unknown, so the existing data in the partition is discarded and consumption position starts from the newly uploaded data.	None.
AT_TIMESTAMP	Data is read from the position denoted by a specific timestamp. A 13-bit timestamp is required when the cursor is obtained. For example, if 1541742263206 is specified, data uploaded from 2018-11-09 13:44:23 is read.	This type of cursor is applicable to the scenario where the consumption position is unknown, but the user wants to consume data from a specific time or from the end time of the last consumption.	• If the upload time of the earliest data record is C, timestamp ≥ C. • If the timestamp is greater than the timestamp of the latest data or the future time, the system reads from the data right after the latest data.

 Copyright © Huawei Technologies Co., Ltd.

Sample Code

Use the initialized client instance to obtain data through the DIS stream. The sample code is available in the **ConsumerDemo.java** file in the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo** directory.

- When the cursor type is set to **AT_SEQUENCE_NUMBER** or **AFTER_SEQUENCE_NUMBER**, the sample code is as follows:

```
//Initialize the DIS SDK client instance.
DIS dic = DISClientBuilder.standard()
          .withEndpoint("xxxx")
          .withAk("xxxx")
          .withSk("xxxx")
          .withProjectId("xxxx")
          .withRegion("xxxx")
          .build();
//Configure the stream name.
String streamName = "streamName";
// Configure the ID of the partition for data download.
String partitionId = "shardId-0000000000";
//Configure the sequence number for data download.
String startingSequenceNumber = "0";
//Configure the data download mode.
//AT_SEQUENCE_NUMBER: Data is read from the position denoted by a specific sequence number. The sequence number is defined by starting-sequence-number in the demo.
// AFTER_SEQUENCE_NUMBER: Data is read from the position right after the position denoted by a specific sequence number. The sequence number is defined by starting-sequence-number.
String cursorType = PartitionCursorTypeEnum.AT_SEQUENCE_NUMBER.name();

try
{
//Obtain the data cursor.
    GetPartitionCursorRequest request = new GetPartitionCursorRequest();
    request.setStreamName(streamName);
    request.setPartitionId(partitionId);
    request.setCursorType(cursorType);
    request.setStartingSequenceNumber(startingSequenceNumber);
    GetPartitionCursorResult response = dic.getPartitionCursor(request);
    String cursor = response.getPartitionCursor();
    LOGGER.info("Get stream {}[partitionId={}] cursor success : {}", streamName, partitionId, cursor);
    GetRecordsRequest recordsRequest = new GetRecordsRequest();
    GetRecordsResult recordResponse = null;
    while (true)
        {
            recordsRequest.setPartitionCursor(cursor);
            recordResponse = dic.getRecords(recordsRequest);
//Obtain the next-batch of data cursor.
            cursor = recordResponse.getNextPartitionCursor();

            for (Record record : recordResponse.getRecords())
            {
                LOGGER.info("Get record [{}], partitionKey [{}], sequenceNumber [{}].",
                    new String(record.getData().array()),
                    record.getPartitionKey(),
                    record.getSequenceNumber());
            }


        }
    }
  catch (DISClientException e)
    {
        LOGGER.error("Failed to get a normal response, please check params and retry. Error message [{}]",
            e.getMessage(),
            e);
    }
    catch (Exception e)
```

 Copyright © Huawei Technologies Co., Ltd.

```
        {
            LOGGER.error(e.getMessage(), e);
        }
    }
}
```

- When the cursor type is set to **TRIM_HORIZON** or **LATEST**, the sample code example is as follows:

```
//Initialize the DIS SDK client instance.
DIS dic = DISClientBuilder.standard()
          .withEndpoint("xxxx")
          .withAk("xxxx")
          .withSk("xxxx")
          .withProjectId("xxxx")
          .withRegion("xxxx")
          .build();
//Configure the stream name.
String streamName = "streamName";
//Configure the partition ID.
String partitionId = "shardId-0000000000";
//Configure the sequence number.
//String startingSequenceNumber = "0";
//Configure the cursor type.
//TRIM_HORIZON: Data is read from the earliest data record in the partition.
//LATEST: Data is read just after the most recent record in the partition. This setting ensures that you always read the most recent data in the partition.
String cursorType = PartitionCursorTypeEnum.TRIM_HORIZON.name();

try
{
//Obtain the data cursor.
    GetPartitionCursorRequest request = new GetPartitionCursorRequest();
    request.setStreamName(streamName);
    request.setPartitionId(partitionId);
    request.setCursorType(cursorType);
    //request.setStartingSequenceNumber(startingSequenceNumber);
    GetPartitionCursorResult response = dic.getPartitionCursor(request);
    String cursor = response.getPartitionCursor();
    LOGGER.info("Get stream {}[partitionId={}] cursor success : {}", streamName, partitionId, cursor);
    GetRecordsRequest recordsRequest = new GetRecordsRequest();
    GetRecordsResult recordResponse = null;
    while (true)
    {
            recordsRequest.setPartitionCursor(cursor);
            recordsRequest.setLimit(limit);
            recordResponse = dic.getRecords(recordsRequest);
//Obtain the next-batch of data cursor.
            cursor = recordResponse.getNextPartitionCursor();

            for (Record record : recordResponse.getRecords())
            {
                LOGGER.info("Get record [{}], partitionKey [{}], sequenceNumber [{}].",
                    new String(record.getData().array()),
                    record.getPartitionKey(),
                    record.getSequenceNumber());
            }

            if (recordResponse.getRecords().size() == 0)
            {
                Thread.sleep(1000);
            }
        }
    }
    catch (DISClientException e)
    {
        LOGGER.error("Failed to get a normal response, please check params and retry. Error message [{}]",
            e.getMessage(),
            e);
    }
```

 Copyright © Huawei Technologies Co., Ltd.

```
        catch (Exception e)
        {
            LOGGER.error(e.getMessage(), e);
        }
    }
}
```

- When the cursor type is set to **AT_TIMESTAMP**, the sample code example is as follows:

```
//Initialize the DIS SDK client instance.
DIS dic = DISClientBuilder.standard()
          .withEndpoint("xxxx")
          .withAk("xxxx")
          .withSk("xxxx")
          .withProjectId("xxxx")
          .withRegion("xxxx")
          .build();
//Configure the stream name.
String streamName = "streamName";
//Configure the partition ID.
String partitionId = "shardId-0000000000";
//Configure the sequence number.
//String startingSequenceNumber = "0";
//Configure the timestamp.
long timestamp = 1542960693804L;
//Configure the cursor type.
//AT_TIMESTAMP: Data is read from the position denoted by a specific timestamp.
String cursorType = PartitionCursorTypeEnum.AT_TIMESTAMP.name();

try
{
//Obtain the data cursor.
    GetPartitionCursorRequest request = new GetPartitionCursorRequest();
    request.setStreamName(streamName);
    request.setPartitionId(partitionId);
    request.setCursorType(cursorType);
    //request.setStartingSequenceNumber(startingSequenceNumber);
    request.setTimestamp(timestamp);
    GetPartitionCursorResult response = dic.getPartitionCursor(request);
    String cursor = response.getPartitionCursor();
    LOGGER.info("Get stream {}[partitionId={}] cursor success : {}", streamName, partitionId, cursor);

        GetRecordsRequest recordsRequest = new GetRecordsRequest();
        GetRecordsResult recordResponse = null;
      while (true)
        {
            recordsRequest.setPartitionCursor(cursor);
            recordsRequest.setLimit(limit);
            recordResponse = dic.getRecords(recordsRequest);
//Obtain the next-batch of data cursor.
            cursor = recordResponse.getNextPartitionCursor();

            for (Record record : recordResponse.getRecords())
            {
                LOGGER.info("Get record [{}], partitionKey [{}], sequenceNumber [{}].",
                    new String(record.getData().array()),
                    record.getPartitionKey(),
                    record.getSequenceNumber());
            }

            if (recordResponse.getRecords().size() == 0)
            {
                Thread.sleep(1000);
            }
        }
    }
    catch (DISClientException e)
    {
        LOGGER.error("Failed to get a normal response, please check params and retry. Error message [{}]",
```

 Copyright © Huawei Technologies Co., Ltd.

```
                e.getMessage(),
                e);
        }
        catch (Exception e)
        {
            LOGGER.error(e.getMessage(), e);
        }
    }
}
```

Parameter Description

Table 1-4 Parameter description

Parameter	Type	Description
partitionId	String	Partition ID. **NOTE** Define the return information fields on the console, such as **partitionId [shardId-0000000000]**, based on the execution results obtained from **Uploading Streaming Data**.
startingSequenceNumber	String	Sequence number of an individual data record. Each data record has a sequence number that is unique within its partition. The sequence number is assigned by DIS when a data producer calls PutRecords to add data to a DIS stream. Sequence numbers for the same partition key generally increase over time; the longer the time period between write requests (PutRecords requests), the larger the sequence numbers become. **NOTE** Define the return information fields on the console, such as **sequenceNumber [1]**, based on the execution results obtained from **Uploading Streaming Data**.

 Copyright © Huawei Technologies Co., Ltd.

Parameter	Type	Description
cursorType	String	Cursor type. • **AT_SEQUENCE_NUMBER**: Data is read from the position denoted by a specific sequence number. The sequence number is defined by **starting-sequence-number** in the demo. This is the default cursor type. • **AFTER_SEQUENCE_NUMBER**: Data is read from the position right after the position denoted by a specific sequence number. The sequence number is defined by **starting-sequence-number** in the demo. • **TRIM_HORIZON**: Data is read from the earliest data record in the partition. For example, a tenant uses a DIS stream to upload three pieces of data A1, A2, and A3. N days later, A1 has expired and A2 and A3 are still in the validity period. In this case, if the tenant sets the cursor type to **TRIM_HORIZON**, the system downloads data from A2. • **LATEST**: Data is read just after the most recent record in the partition. This setting ensures that you always read the most recent data in the partition. • **AT_TIMESTAMP**: Data is read from the position denoted by a specific timestamp.

Running the Program

Right-click the program and choose **Run As > 1 Java Application** from the shortcut menu. If the program is run successfully, you can view the information similar to the following on the console:

```
14:55:42.954 [main] INFOcom.bigdata.dis.sdk.DISConfig - get from classLoader
14:55:44.103 [main] INFOcom.bigdata.dis.sdk.util.config.ConfigurationUtils - get from classLoader
14:55:44.105 [main] INFOcom.bigdata.dis.sdk.util.config.ConfigurationUtils - propertyMapFromFile size : 2
14:55:45.235 [main] INFOcom.bigdata.dis.sdk.demo.ConsumerDemo - Get stream
streamName[partitionId=0] cursor success :
eyJnZXRJdGVyYXRvclBhcmFtIjp7InN0cmVhbS1uYW1lIjoiZGlzLTEzbW9uZXkiLCJwYXJ0aXRpb24taWQiOiIwIiwiY
3Vyc29yLXR5cGUiOiJBVF9TRVFVRU5DRV9OVU1CRVIiLCJzdGFydGluZy1zZXF1ZW5jZS1udW1iZXIiOiIxMDY4O
TcyIn0sImdlbmVyYXRlVGltZXN0YW1wIjoxNTEzNjY2NjMxMTYxfQ
14:55:45.305 [main] INFOcom.bigdata.dis.sdk.demo.ConsumerDemo - Get Record [hello world.],
partitionKey [964885], sequenceNumber [0].
14:55:45.305 [main] INFOcom.bigdata.dis.sdk.demo.ConsumerDemo - Get Record [hello world.],
partitionKey [910960], sequenceNumber [1].
14:55:46.359 [main] INFOcom.bigdata.dis.sdk.demo.ConsumerDemo - Get Record [hello world.],
partitionKey [528377], sequenceNumber [2].
```

 Copyright © Huawei Technologies Co., Ltd.

1.6.16 Uploading Streaming Data

Sample Code

Use the initialized client instance to upload your streaming data to DIS through a DIS stream. The sample code is available in the **ProducerDemo.java** file in the **dis-sdk-demo\src\main\java\com\bigdata\dis\sdk\demo** directory.

The value of **streamName** must be the same as the value of **Stream Name** in **Creating a DIS Stream**. For details about endpoints, AKs, SKs, regions, and project IDs, see **Obtaining Authentication Information**.

The code for uploading streaming data is as follows:

```
//Initialize the DIS SDK client instance.


DIS dic = DISClientBuilder.standard()
          .withEndpoint("xxxx")
          .withAk("xxxx")
          .withSk("xxxx")
          .withProjectId("xxxx")
          .withRegion("xxxx")
          .build();

//Configure the stream name.
String streamName = "xxxx";
//Configure the data to be uploaded.
String message = "hello world.";
 PutRecordsRequest putRecordsRequest = new PutRecordsRequest();
       putRecordsRequest.setStreamName(streamName);
       List<PutRecordsRequestEntry> putRecordsRequestEntryList = new
ArrayList<PutRecordsRequestEntry>();
       ByteBuffer buffer = ByteBuffer.wrap(message.getBytes());
       for (int i = 0; i < 3; i++)
       {
           PutRecordsRequestEntry putRecordsRequestEntry = new PutRecordsRequestEntry();
           putRecordsRequestEntry.setData(buffer);

putRecordsRequestEntry.setPartitionKey(String.valueOf(ThreadLocalRandom.current().nextInt(1000000)));
           putRecordsRequestEntryList.add(putRecordsRequestEntry);
       }
       putRecordsRequest.setRecords(putRecordsRequestEntryList);

       LOGGER.info("========== BEGIN PUT ============");

       PutRecordsResult putRecordsResult = null;
       try
       {
           putRecordsResult = dic.putRecords(putRecordsRequest);
       }
       catch (DISClientException e)
       {
           LOGGER.error("Failed to get a normal response, please check params and retry. Error message [{}]",
               e.getMessage(),
               e);
       }
       catch (Exception e)
       {
           LOGGER.error(e.getMessage(), e);
       }
```

 Copyright © Huawei Technologies Co., Ltd.

Running the Program

Right-click the program and choose **Run As > 1 Java Application** from the shortcut menu. If the program is run successfully, you can view the information similar to the following on the console:

```
15:19:29.298 [main] INFO  com.bigdata.dis.sdk.demo.ProducerDemo - ========== BEGIN PUT
============
15:19:30.992 [main] INFO  com.bigdata.dis.sdk.demo.ProducerDemo - Put 3 records[3 successful / 0 failed].
15:19:30.992 [main] INFO  com.bigdata.dis.sdk.demo.ProducerDemo - [hello world.] put success, partitionId
[shardId-0000000000], partitionKey [261045], sequenceNumber [1]
15:19:30.992 [main] INFO  com.bigdata.dis.sdk.demo.ProducerDemo - [hello world.] put success, partitionId
[shardId-0000000000], partitionKey [958815], sequenceNumber [2]
15:19:30.992 [main] INFO  com.bigdata.dis.sdk.demo.ProducerDemo - [hello world.] put success, partitionId
[shardId-0000000000], partitionKey [416421], sequenceNumber [3]
15:19:30.992 [main] INFO  com.bigdata.dis.sdk.demo.ProducerDemo - ========== END PUT ============
```

1.6.17 Obtaining the Data Cursor

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to obtain the information about the data cursor.

```
//Configure the stream name.
String streamName = "myStream";
// Configure the ID of the partition for data download.
String partitionId = "0";
//Configure the sequence number for data download.
String startingSequenceNumber = "0";
//Configure the data download mode.
String cursorType = PartitionCursorTypeEnum.AT_SEQUENCE_NUMBER.name();

GetPartitionCursorRequest request = new GetPartitionCursorRequest();
request.setStreamName(streamName);
request.setPartitionId(partitionId);
request.setStartingSequenceNumber(startingSequenceNumber);
request.setCursorType(cursorType);
GetPartitionCursorResult response = dic.getPartitionCursor(request);
String cursor = response.getPartitionCursor();
```

1.6.18 Creating an Application

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

When using DIS SDK to create an application, you need to specify an application name.

```
//Specify the application name.
String appName = "myApp";
```

After the application name is specified, create the application by calling createApp.

```
dic.createApp(myApp);
```

1.6.19 Deleting an Application

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

When using DIS SDK to delete an application, you need to specify an application name.

 Copyright © Huawei Technologies Co., Ltd.

```
//Specify the application to be deleted.
String appName = "myApp";
```

After the application name is specified, delete the application by calling deleteApp.

```
dic.deleteApp(myApp);
```

1.6.20 Adding a Checkpoint

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

When using the DIS SDK to create a checkpoint, you need to specify the stream name, application name, partition ID, sequence number, and checkpoint type.

The value of **streamName** must be the same as the value of **Stream Name** in **Creating a DIS Stream**.

```
//Specify the stream name.
String streamName = "myStream";
//Specify the application name.
String appName = "myApp";
CommitCheckpointRequest commitCheckpointRequest = new CommitCheckpointRequest();
commitCheckpointRequest.setStreamName(streamName);
commitCheckpointRequest.setAppName(appName);
//Specify the sequence number to be submitted.
commitCheckpointRequest.setSequenceNumber("100");
//Specify the partition No.
commitCheckpointRequest.setPartitionId("0");
//Specify the checkpoint type.
commitCheckpointRequest.setCheckpointType(CheckpointTypeEnum.LAST_READ.name());
```

After configuring **CommitCheckpointRequest**, add a checkpoint by calling commitCheckpoint.

```
dic.commitCheckpoint(commitCheckpointRequest);
```

1.6.21 Querying a Checkpoint

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

The value of **streamName** must be the same as the value of **Stream Name** in **Creating a DIS Stream**. For details about endpoints, AKs, SKs, regions, and project IDs, see **Obtaining Authentication Information**.

```
GetCheckpointRequest getCheckpointRequest = new GetCheckpointRequest();
//Specify the application name. (Note: setAppId is used in 1.3.0 and earlier versions, and setAppName is used in 1.3.1 and later versions.)
getCheckpointRequest.setAppName(appName);
getCheckpointRequest.setStreamName(streamName);
//Specify the partition No.
getCheckpointRequest.setPartitionId("0");
//Specify the checkpoint type.
getCheckpointRequest.setCheckpointType(CheckpointTypeEnum.LAST_READ.name());
System.out.println("getCheckpoint: " + JsonUtils.objToJson(dic.getCheckpoint(getCheckpointRequest)));
```

The returned checkpoint details are as follows:

```
getCheckpoint:
{"sequence_number": "10", "metadata": "metadata"}
```

 Copyright © Huawei Technologies Co., Ltd.

1.6.22 Changing Partition Quantity

Initialize a DIS SDK client instance named **dic**. For details, see **Initializing a DIS SDK Client Instance**.

The value of **streamName** must be the same as the value of **Stream Name** in **Creating a DIS Stream**. For details about endpoints, AKs, SKs, regions, and project IDs, see **Obtaining Authentication Information**.

```
//Specify the number of target partitions.
int targetPartitionCount = 2;

UpdatePartitionCountRequest update = new UpdatePartitionCountRequest();
update.setStreamName(streamName);
update.setTargetPartitionCount(targetPartitionCount);
try
{
    UpdatePartitionCountResult updatePartitionCountResult = dic.updatePartitionCount(update);
    LOGGER.info("Success to update partition count, {}", updatePartitionCountResult);
}
catch (Exception e)
{
    LOGGER.error("Failed to update partition count", e);
}
```

If the number of partitions is changed successfully, information similar to the following is returned:

```
Success to update partition count, UpdatePartitionCountResult [currentPartitionCount=2,
streamName=mystream, targetPartitionCount=2]
```

1.7 Using the C# SDK

1.7.1 Preparing the Installation Environment

Download the latest version of Microsoft Visual Studio from **Microsoft's official website** and install it.

1.7.2 Configuring a Sample Project

The **huaweicloud-sdk-dis-net-*X.X.X*.zip** package downloaded from **SDK Download** provides a sample project. You can use a development tool (such as Microsoft Visual Studio) to compile and run the sample project on a local device. You can also develop applications based on the sample project. The sample project code is available in **[$path]\huaweicloud-sdk-dis-net-*X.X.X*\DIS .NET SDK \DIS .NET SDK**.

Sample Project File	Description
demo	Sample code file, which contains sample codes of net35 and net45.
net35	Contains: • Third-party log library file **log4net.dll** • DIS library file • Related dependency file ***.dll**

Copyright © Huawei Technologies Co., Ltd.

Sample Project File	Description
net45	Contains: • Third-party log library file **log4net.dll** • DIS library file • Related dependency file ***.dll**
config	Contains the **dis.properties** file that records the DIS parameter configurations and the **version.properties** file that records the version information.

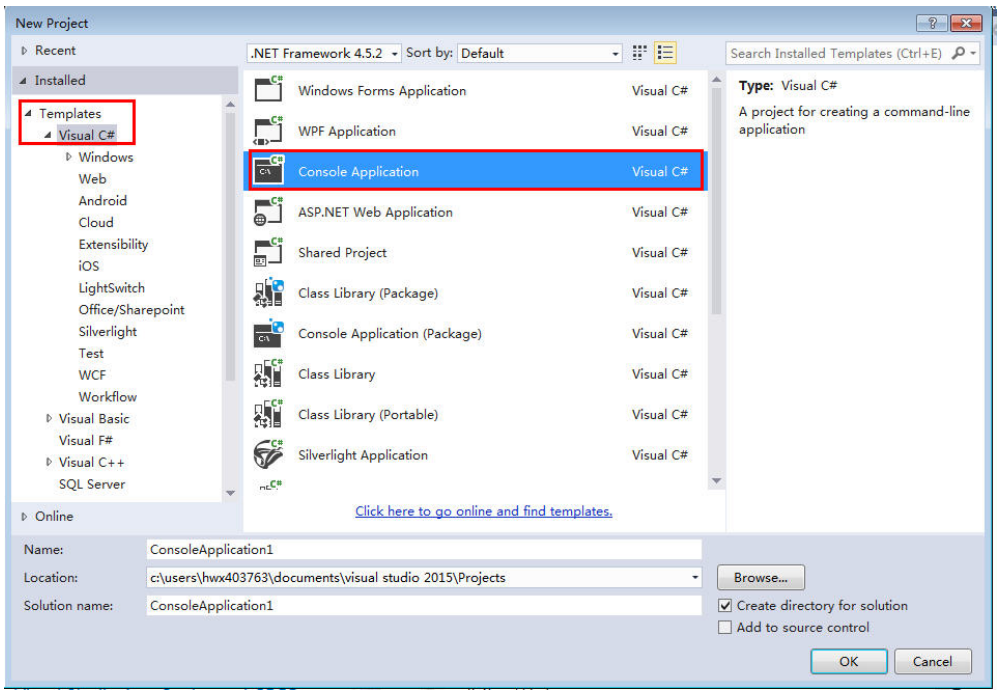
Procedure

Step 1 Decompress the **huaweicloud-sdk-dis-net-*X.X.X*1.2.3.zip** package downloaded from **SDK Download** to obtain the **dis-sdk-.NET** package and sample project.

Step 2 Start Visual Studio and choose **File** > **New** > **Project**. The **New Project** dialog box is displayed.

Step 3 Choose **Templates** > **Visual C#** > **Console Application**.

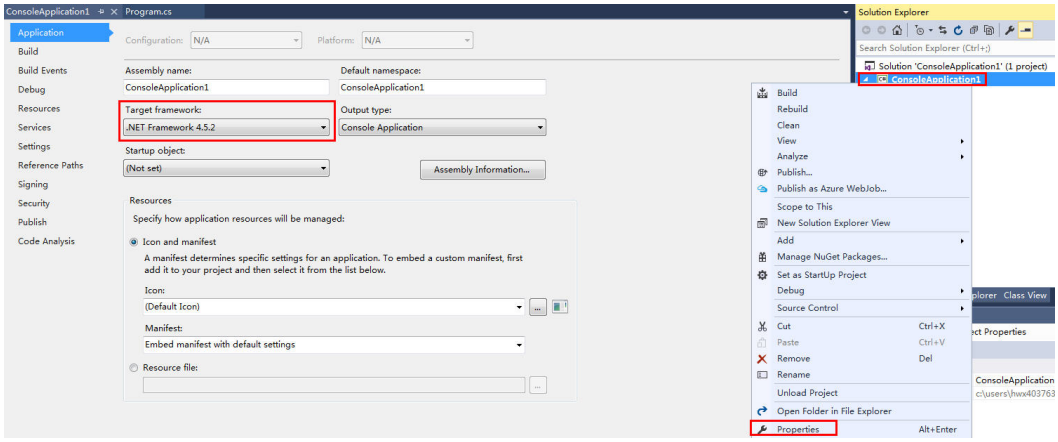
Figure 1-3 Creating a console application



Step 4 Right-click the new project and choose **Properties** from the shortcut menu. Change the version of **Target framework** to **.Net Framework 3.5**. In the displayed dialog box, click **OK**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-4 Changing the framework version.



Step 5 In Visual Studio, right-click the project and choose **Open Folder in File Explorer** from the shortcut menu to obtain the project path, for example, **C:\Users\XXX\Documents\visual studio 2015\Projects\ConsoleApplication1\ConsoleApplication1**.

Step 6 Copy the **net35** file in the decompressed SDK package to the project path.

Step 7 Copy the **dis.properties** and **version.properties** files under the **config** directory from the decompressed SDK package to the *$project path***bin\Debug** directory.

Step 8 Modify the **dis.properties** file. For details about how to configure the values of **endpoint**, **ak**, **sk**, **region**, and **projectId** in this file, see **Obtaining Authentication Information**.

```
IS_DEFAULT_TRUSTED_JKS_ENABLED=false
data.encrypt.enabled=false

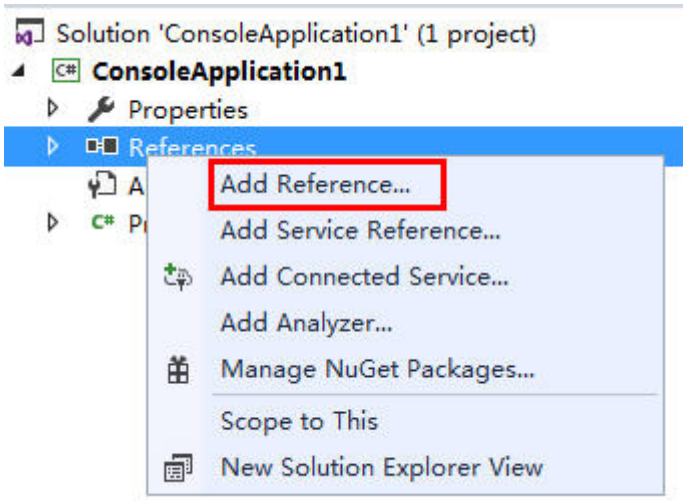
#todo
endpoint=https://yourdomainname
region=Provide your region
ak=Provide your Access Key
sk=Provide your Secret Key
projectId=Provide your project Id
```

Step 9 Use the **Program.cs** file under the **demo** folder in the decompressed SDK package to replace the corresponding file in the *$project path* directory.

Step 10 In Visual Studio, right-click **References** under the new project and choose **Add Reference** from the shortcut menu.

 Copyright © Huawei Technologies Co., Ltd.

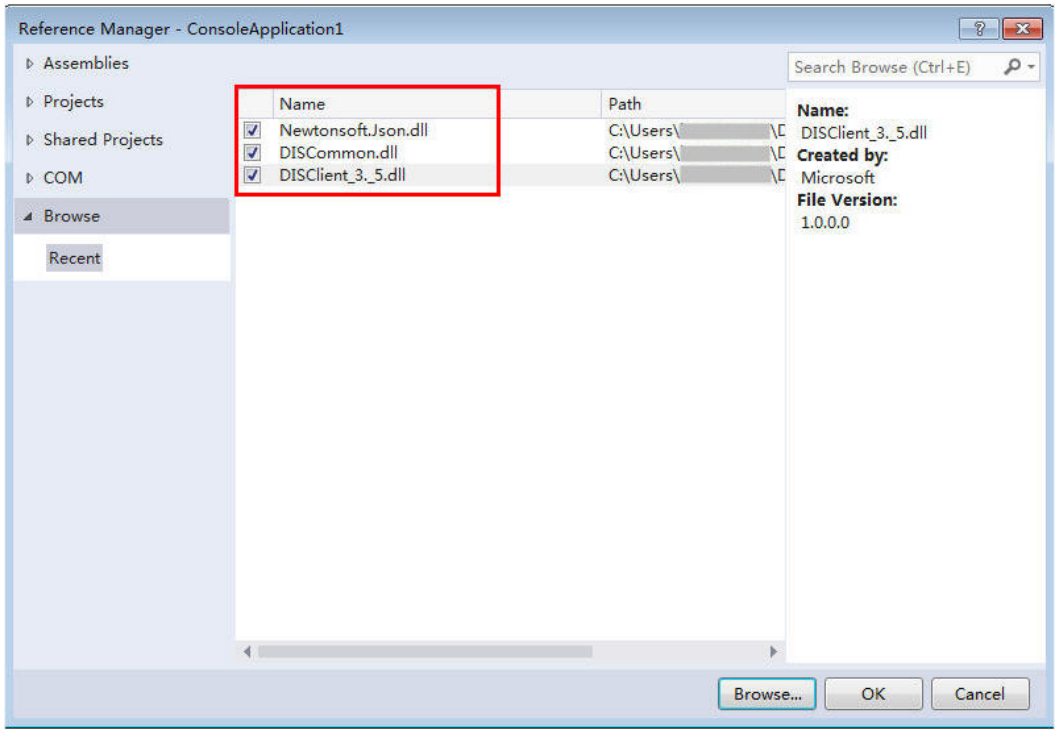
Figure 1-5 Adding references



Step 11 In the **Reference Manager** dialog box, click **Browse** and select the following files in *$project path***net35**:

DISClient_3._5.dll, **DISCommon.dll**, and **Newtonsoft.Json.dll**

Figure 1-6 Adding dependency file *.dll



Step 12 Click **OK**.

----End

 Copyright © Huawei Technologies Co., Ltd.

1.7.3 Initializing a DIS SDK Client Instance

You can use either of the following methods to initialize the DIS SDK client instance: For details about **endpoint**, **ak**, **sk**, **region**, and **projectId**, see **Obtaining Authentication Information**.

- Use codes to initialize the DIS SDK client instance. The code example is as follows:

```
//Create a DIS SDK client instance.
DISConfig disConfig = new DISConfig();
disConfig.SetEndpoint("https://ip:port");
disConfig.SetAK("xxxx");
disConfig.SetSK("xxxx");
disConfig.SetProjectId("xxxxxxx");
disConfig.SetRegion("XXX");
DISIngestionClient dic = new DISIngestionClient(disConfig);
```

- Use the configuration file to initialize the DIS SDK client instance.

  Add the following configuration items to the **dis.propertites** file in the **$project path\bin\Debug** directory:

  - ak/sk: AK/SK created on the IAM.
  - region: area where the stream is used.
  - endpoint: access address of the DIS.
  - projectId: project ID of the stream.

```
//Create a DIS SDK client instance.
        DISIngestionClient dic = new DISIngestionClient();
```

1.7.4 Creating a Stream

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

When you use the DIS SDK to create a DIS stream, specify the stream name, number of partitions in the stream, and stream type.

COMMON indicates a common stream, and **ADVANCED** indicates an advanced stream.

Creating a Stream That Has No Dump Tasks

```
//Stream name.
string streamName = "XXX";
//Partition quantity.
int partitionCount = XXX;
CreateStreamRequest request = new CreateStreamRequest
{
   //Stream name.
     StreamName =streamName,
   //Stream type (COMMON and ADVANCED).
    StreamType = "XXX",
//Partition quantity.
    PartitionCount =partitionCount,
//Source data type: BLOB, JSON, CSV, and file.
    DataType = "XXX",
//Data retention. Value: N x 24. N is an integer ranging from 1 to 7.
    DataDuration = 7 * 24,
};
```

After configuring **CreateStreamRequest**, you can create a stream by calling createStream.

 Copyright © Huawei Technologies Co., Ltd.

```
ResponseResult response = dic.CreateStream(request);
Console.WriteLine(response);
```

Viewing Execution Results

Press **Ctrl+F5**. If information similar to the following is displayed on the console, the stream is created successfully:

```
ResponseResult{status_code='201 Created', content='', errorCode='', message=''}
```

Table 1-5 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

Creating a Stream That Has Periodic OBS Dump Tasks

```
//Stream name.
string streamName = "XXX";
//Partition quantity.
int partitionCount = XXX;
 var dic = new DISIngestionClient();
          var request = new CreateStreamRequest
          {
              StreamName = streamName,
//Stream type (COMMON and ADVANCED).
              StreamType = "XXX",
              PartitionCount = partitionCount,
//Source data type: BLOB, JSON, CSV, and file.
               DataType = "XXX",
//Data retention. Value: N x 24. N is an integer ranging from 1 to 7.
                DataDuration = 7 * 24,

          };

          List<ObsDestinationDescriptorEntity> obsDestinationDescriptorEntitys = new
List<ObsDestinationDescriptorEntity>();
              var obsDestinationDescriptorEntity = new ObsDestinationDescriptorEntity
              {
//IAM agency name.
                  AgencyName = "XXX",
//Bucket name.
                  ObsBucketPath = "XXX",
//Custom file
                  FilePrefix = "XXX",
//Time when data is imported to OBS
                  DeliverTimeInterval =XXX,
//Directory level in the OBS bucket.
                  PartitionFormat = "XXX",
   //Delimiter.
                  RecordDelimiter = "XXX"
              };
              obsDestinationDescriptorEntitys.Add(obsDestinationDescriptorEntity);
          request.ObsDestinationDescriptor = obsDestinationDescriptorEntitys;
```

After configuring **CreateStreamRequest**, you can create a stream by calling createStream.

 Copyright © Huawei Technologies Co., Ltd.

```
ResponseResult response = dic.CreateStream(request);
Console.WriteLine(response);
```

Viewing Execution Results

Press **Ctrl+F5**. If information similar to the following is displayed on the console, the stream is created successfully:

```
ResponseResult{status_code='201 Created', content='', errorCode='', message=''}
```

Table 1-6 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

Creating a DIS Stream That Has Custom OBS Dump Tasks

```
//Stream name.
string streamName = "XXX";
//Partition quantity.
int partitionCount = XXX;
var dic = new DISIngestionClient();
        var request = new CreateStreamRequest
        {
            StreamName = streamName,
            PartitionCount = partitionCount,
//Data retention. Value: N x 24. N is an integer ranging from 1 to 7.
              DataDuration = 7 * 24,
//Stream type (COMMON and ADVANCED).
StreamType = "XXX",
        };

        List<ObsDestinationDescriptorEntity> obsDestinationDescriptorEntitys = new
List<ObsDestinationDescriptorEntity>();

            var obsDestinationDescriptorEntity = new ObsDestinationDescriptorEntity
            {
//IAM agency name.
                AgencyName = "XXX",
//Bucket name.
                ObsBucketPath = "XXX",
//Specify this parameter when Data Dump Type is Custom Dump. Default value: file_stream.
                DeliverDataType = "XXX",

            };
            obsDestinationDescriptorEntitys.Add(obsDestinationDescriptorEntity);
        request.ObsDestinationDescriptor = obsDestinationDescriptorEntitys;
```

After configuring **CreateStreamRequest**, you can create a stream by calling **createStream**.

```
ResponseResult response = dic.CreateStream(request);
Console.WriteLine(response);
```

Viewing Execution Results

 Copyright © Huawei Technologies Co., Ltd.

Press **Ctrl+F5**. If information similar to the following is displayed on the console, the stream is created successfully:

```
ResponseResult{status_code='201 Created', content='', errorCode='', message=''}
```

Table 1-7 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

1.7.5 Deleting a Stream

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to delete a specified DIS stream.

```
//Stream to be deleted.
String streamName = "XXX";
var request = new DescribeStreamRequest { StreamName = streamName };
```

After configuring **DescribeStreamRequest**, you can delete the stream by calling DeleteStream.

```
ResponseResult response = dic.DeleteStream(request);
        Console.WriteLine(response);
```

Viewing Execution Results

Press **Ctrl+F5**. If information similar to the following is displayed on the console, the stream is deleted successfully:

```
{ResponseResult{status_code='204 NO CONTENT', content='', errorCode='', message=''}}
```

Table 1-8 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

1.7.6 Querying a Stream List

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to query active streams.

 Copyright © Huawei Technologies Co., Ltd.

Use the setLimit method to set the number of streams returned each time. If setLimit is not specified, a maximum of 10 streams are returned by default.

```
String startStreamName = "XXX";
        int? limit=10;
            var dic = new DISIngestionClient();
            var request = new DescribeStreamRequest();
            if (!string.IsNullOrWhiteSpace(startStreamName))
            {
//Name of the DIS stream to start the stream list with. The returned stream list does not contain this DIS stream name.
                request.StartStreamName = startStreamName;
            }

            if (limit != null)
            {
//Max. number of DIS streams to list in a single API call.
                request.Limit = limit.Value;
            }
```

After configuring **DescribeStreamRequest**, you can query the stream list by calling DescribeStreamList.

```
            response = dic.DescribeStreamList(request);
            var reqJson = JsonConvert.SerializeObject(response);
            Console.WriteLine(reqJson);
            return response;
```

Viewing Execution Results

Press **Ctrl+F5**. Information similar to the following is displayed:

```
{"total_number":12,"stream_names":["dis-2TbN","dis-RVGG","dis-VKGL","dis-c-test-partition-1","dis-m8lK","dis-shawobs2","dis-shawobsfile","dis-test-stream","dis_CSV","dis_JSON","dis_bl","dis_file"],"has_more_streams":false}
```

Table 1-9 Description of response parameter **DescribeStreamListResult**

Parameter	Type	Description
total_number	int	Total number of all the DIS streams created by the current tenant.
stream_names	List<string>	Name of the list of the streams meeting the current requests.
has_more_streams	bool	Specify whether there are more matching DIS streams to list. Possible values:

1.7.7 Querying Stream Details

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to query the details about a specified stream.

```
//Stream to be queried.
string streamName = "XXX";
```

 Copyright © Huawei Technologies Co., Ltd.

```
//Start partition ID.
 string startPartitionId = "XXX";
//Max. number of partitions to list in a single API call.
int? limitPartitions =XXX;
  var request = new DescribeStreamRequest
{
//Stream to be queried.
    StreamName = streamName
};

          if (!string.IsNullOrWhiteSpace(startPartitionId))
          {
//Name of the partition to start the partition list with. The returned partition list does not contain this
partition.
              request.StartPartitionId = startPartitionId;
          }

          if (limitPartitions != null)
          {
//Max. number of partitions to list in a single API call.
              request.LimitPartitions = limitPartitions.Value;
          }
```

After configuring **DescribeStreamRequest**, you can query the stream details by calling DescribeStream.

```
         response = dic.DescribeStream(request);
          var responseJson = JsonConvert.SerializeObject(response);
          Console.WriteLine(responseJson);
          return response;
```

Viewing Execution Results

Press **Ctrl+F5**. Information similar to the following is displayed:

```
{"stream_name":"dis-shawobs2","stream_id":"IY6gsAE3HEsBI7hvdBp","create_time":
1531107213118,"last_modified_time":1531107213118,"retention_period":
24,"status":"RUNNING","stream_type":"COMMON","partitions":
[{"status":"ACTIVE","partition_id":"shardId-0000000000","hash_range":"[0 :
461168601842738790 2]","sequence_number_range":"[0 : 0]"},
{"status":"ACTIVE","partition_id":"shardId-0000000001","hash_range":"[461168601842738790 3 :
922337203685477580 7]","sequence_number_range":"[0 : 0]"}],"has_more_partitions":false}
```

Table 1-10 Description of response parameter **DescribeStreamResult**

Parameter	Type	Description
stream_name	String	Name of the stream.
stream_id	String	Unique identifier of the stream.
create_time	Long	Timestamp at which the stream was created.
last_modified_time	Long	Timestamp at which the stream was most recently modified.
retention_period	Int	Period of time for which data is retained in the stream.

 Copyright © Huawei Technologies Co., Ltd.

Parameter	Type	Description
status	String	Current status of the stream. Possible values: • CREATING • RUNNING • TERMINATING • FROZEN
stream_type	String	Stream type.
partitions	List<PartitionResult>	Partition list.
has_more_partitions	Boolean	Specify whether there are more matching partitions of the DIS stream to list. • **true**: There are such streams. • **false**: There are no such partitions.
Parameters in **PartitionResult**		
status	String	Current status of the partition. Possible values: • CREATING • ACTIVE
partition_id	String	Unique identifier of the partition.
hash_range	String	Possible value range of the hash key used by the partition.
sequence_number_range	String	Sequence number range of the partition.

1.7.8 Uploading Streaming Data

Sample Code

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**. Use the initialized client instance to upload your streaming data to DIS through a DIS stream.

```
//Initialize a DIS client.
var dic = new DISIngestionClient();
//Specify the stream name.
 String streamName = "streamName";
PutRecordsRequest putRecordsRequest = new PutRecordsRequest();
```

 Copyright © Huawei Technologies Co., Ltd.

```
putRecordsRequest.StreamName = streamName;
//Configure the data to be uploaded.
 var putRecordsRequestEntries = new List<PutRecordsRequestEntry>();
string shardId = "shardId-0000000000";
for (int i = 0; i < 3; i++)
{
string a = shardId + i;
var putRecordsRequestEntry = new PutRecordsRequestEntry
{
//Data to be uploaded. The uploaded data is the serialized binary data in the Base64 format.
Data = Encoding.UTF8.GetBytes(a),
//Specify the hash value of the data to be written into the partition.
ExplicitHashKey = "123",
//Partition into which data will be written.
PartitionKey = 1
};
putRecordsRequestEntries.Add(putRecordsRequestEntry);
}
putRecordsRequest.Records = putRecordsRequestEntries;
PutRecordsResult response = dic.PutRecords(putRecordsRequest);

foreach (var item in response.Records)
{
Console.WriteLine(item);
}
```

Viewing Execution Results

Run **Ctrl+F5**. Information similar to the following is displayed on the console:

```
PutRecordsResultEntry [shardId=shardId-0000000000, sequenceNumber=0, errorCode=,
 errorMessage=]
PutRecordsResultEntry [shardId=shardId-0000000000, sequenceNumber=1, errorCode=,
 errorMessage=]
PutRecordsResultEntry [shardId=shardId-0000000000, sequenceNumber=2, errorCode=,
 errorMessage=]
```

Table 1-11 Description of response parameter **PutRecordsResult**

Parameter	Type	Description
failed_record_count	Int	Number of data records that fail to be uploaded.
records	List<PutRecordsResultEntry>	Information of data records.
Description of parameter **PutRecordsResultEntry**		
errorCode	String	Error code.
errorMessage	String	Error message.
shardId	String	Partition ID.

 Copyright © Huawei Technologies Co., Ltd.

Parameter	Type	Description
sequenceNumber	String	Sequence number. A sequence number is the unique identifier of each record. DIS automatically allocates a sequence number when the data producer calls the PutRecords operation to add data to the DIS stream. The sequence number of the same partition key usually changes with time. A longer interval between PutRecords requests results in a larger sequence number.

1.7.9 Downloading Streaming Data

Sample Code

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**. Use the client instance to obtain data through the DIS stream.

```
var dic = new DISIngestionClient();
//Specify the stream name.
String streamName = "streamName";
//Specify the partition ID.
String shardId = "shardId-0000000000";
//Specify the sequence number.
const string startingSequenceNumber = "0";
//Specify the data download mode.
const string shardIteratorType = "TRIM_HORIZON";
//Obtain the data cursor.
        var request = new GetShardIteratorRequest
        {
//Specify the stream name.
            StreamName = streamName,
//Specify the partition ID.
            ShardId = shardId,
//Specify the sequence number.
            StartingSequenceNumber = startingSequenceNumber,
//Specify the cursor type.
            ShardIteratorType = shardIteratorType
        };

        var recordsRequest = new GetRecordsRequest();
        var response = dic.GetShardIterator(request);
        Console.Out.WriteLine(response);

        var iterator = response.ShardIterator;
//Download data.
        while (true)
        {
```

 Copyright © Huawei Technologies Co., Ltd.

```
//Obtain the data cursor.
            recordsRequest.ShardIterator = iterator;
            var recordResponse = dic.GetRecords(recordsRequest);
            if (recordResponse.Records.Count > 0)
            {
                foreach (var record in recordResponse.Records)
                {
                    Console.WriteLine("Record[{0}] = {1}", record.SequenceNumber, DecodeData(record.Data));
                }
            }
            else
            {
                break;
            }
            iterator = recordResponse.NextShardIterator;

}
```

Viewing Execution Results

Run **Ctrl+F5**. Information similar to the following is displayed on the console:

```
GetShardIteratorResult[shardIterator=eyJnZXRJdGVyYXRvclBhcmFtIjp7InN0cmVhbS1uYW1lIjoiZGlzLXNoYXc1IiwicGFydGl0aW9uLWlkIjoic2hhcmRJZC0wMDAwMDAwMDAwIiwiY3Vyc29yLXR5cGUiOiJUUklNX0hPUklaT04iLCJzdGFydGluZy1zZXF1ZW5jZS1udW1iZXIiOiIwIn0sImdlbmVyYXRlVGltZXN0YW1wIjoxNTMxMTI1ODEwNDE1fQ]
Record[0] = shardId-00000000000
Record[1] = shardId-00000000001
Record[2] = shardId-00000000002
```

Table 1-12 Response parameter description

Parameter	Type	Description
records	List<Record>	Information of data records.
next_partition_cursor	String	Next iterator. **NOTE** The validity period of a cursor is 5 minutes.
Description of parameter **Record**		
partition_key	String	Partition into which data will be written.

 Copyright © Huawei Technologies Co., Ltd.

Parameter	Type	Description
sequence_number	String	Sequence number. A sequence number is the unique identifier of each record. DIS automatically allocates a sequence number when the data producer calls the PutRecords operation to add data to the DIS stream. The sequence number of the same partition key usually changes with time. A longer interval between PutRecords requests results in a larger sequence number.
timestamp	Long	Timestamp when the record is written to DIS.
timestamp_type	String	Type of the timestamp. The value is **CreateTime**, specifying the creation time.

1.7.10 Uploading a Real-Time File

The DIS SDK supports synchronous file upload in Net3.5 and supports both synchronous and asynchronous file uploads in Net4.5.

Synchronous File Upload

```
UploadFileDemo("fileUploadStream", @"C:\Users\Public\Pictures\Sample Pictures\Penguins.jpg");
        public static bool UploadFileDemo(string streamName, string file)
        {
            int failedRecordCount = 0;
            var dic = new DISIngestionClient();
//Max. file upload size. A maximum of 128 MB files can be uploaded. Set this parameter to 500 KB in this example.
            int splitFileSize = 500 * 1024;
            string deliverDataId = Guid.NewGuid().ToString("N");
            FileInfo fileInfo = new FileInfo(file);
//Number of uploads.
            int steps = (int)Math.Ceiling((decimal)fileInfo.Length / splitFileSize);
            using (FileStream fs = new FileStream(file, FileMode.Open, FileAccess.Read))
            {
                using (BinaryReader br = new BinaryReader(fs))
                {
                    for (int i = 0; i < steps; i++)
                    {
                        byte[] input = br.ReadBytes(splitFileSize);
                        {
                            PutRecordsRequest putRecordsRequest = new PutRecordsRequest();
//Stream name.
```

 Copyright © Huawei Technologies Co., Ltd.

```
                    putRecordsRequest.StreamName = streamName;
                    var putRecordsRequestEntries = new List<PutRecordsRequestEntry>();
                    var putRecordsRequestEntry = new PutRecordsRequestEntry
                    {
//Data to be uploaded.
                        Data = input,
//Hash value of the data to be written into the partition.
                        ExplicitHashKey = null,
//Unique identifier of the partition.
                        //PartitionId = "shardId-0000000000",
//Partition into which data will be written.
                        //PartitionKey = "0",//i.ToString(),
//Configuration information.
                        ExtenedInfo = new PutRecordsRequestEntryExtendedInfo()
                        {
//File for storing the real-time file data.
                            FileName = @"thisisshawnfolder/2018/07/04/14/56/Penguins2.jpg",
//ID of the real-time file data, which is specified by the user.
                            DeliverDataId = deliverDataId,
//Block sequence number of the file to be uploaded. The start sequence number is 0. When a request with sequent number 0 is received, the system uploads a new file or re-uploads a file by default.
                            SequenceNumber = i,
//Specify whether the current file block is the last one. The value can be true or false.
                            EndFlag = i == steps - 1 ? true : false,
                        }
                    };
                    putRecordsRequestEntries.Add(putRecordsRequestEntry);
                    putRecordsRequest.Records = putRecordsRequestEntries;
                    PutRecordsResult response = dic.PutRecords(putRecordsRequest);
                    failedRecordCount += response.FailedRecordCount;
                    //Thread.Sleep(5000);
                    foreach (var item in response.Records)
                    {
                        Console.WriteLine(item);
                    }
                }
            }
        }
        return failedRecordCount > 0;
    }
```

Asynchronous File Upload

```
 var dic = new DISAsync();
        PutRecordsRequest putRecordsRequest = new PutRecordsRequest();
//Stream name.
        putRecordsRequest.StreamName = "fileUploadStream";
      //File for storing the real-time file data.
        putRecordsRequest.file_name = @"thisismyfolder/2018/07/04/14/56/Penguins1110.jpg";
   //File to be uploaded.
        putRecordsRequest.file_path = @"C:\Users\Public\Pictures\Sample Pictures\Penguins.jpg";
        var bflag = dic.PutFilesAsync(putRecordsRequest);
        Console.WriteLine(bflag.Result);
```

- Configure a stream for uploading real-time files.

  Use an existing stream that is used to transmit real-time files.

  ```
  //Stream name.
  String streamName = "fileUploadStream";
  ```

- Specify the path of the real-time file to be uploaded.

  Enter the absolute path of the file to be uploaded.

  ```
  //Path of the file to be uploaded.
  putRecordsRequest.file_path = @"C:\Users\Public\Pictures\Sample Pictures\Penguins.jpg";
  ```

- Specify the file for storing the uploaded real-time data.

  Configure the name for the file used to store the uploaded real-time data.

 Copyright © Huawei Technologies Co., Ltd.

```
//Name of the file for storing the uploaded real-time data.
putRecordsRequest.file_name = @"thisismyfolder/2018/07/04/14/56/
Penguins1110.jpg";
```

Viewing Execution Results

True: indicates that the data is uploaded successfully. False: indicates that the data is uploaded unsuccessfully.

1.7.11 Obtaining a Data Cursor

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to obtain the information about the data cursor.

```
//Stream name.
String streamName = "myStream";
//Partition ID.
String partitionId = "0";
//Sequence number.
String startingSequenceNumber = "0";
//Data download mode.
String shardIteratorType= "TRIM_HORIZON";
var dic = new DISIngestionClient();
var request = new GetShardIteratorRequest
            {
//Stream name.
                StreamName = streamName,
//Partition ID.
                ShardId = shardId,
//Cursor type.
                ShardIteratorType = shardIteratorType,
//Sequence number.
                StartingSequenceNumber = startingSequenceNumber,
 };
var response = dic.GetShardIterator(request);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
GetShardIteratorResult [shardIterator=eyJnZXRJdGVyYXRvclBhcmFtIjp7InN0cmVhbS1uYW
1lIjoiZGlzLXNoYXdvYnNmaWxlIiwicGFydGl0aW9uLWlkIjoic2hhcmRJZC0wMDAwMDAwMDAwIiwiY3
Vyc29yLXR5cGUiOiJUUklNX0hPUklaT04iLCJzdGFydGluZy1zZXF1ZW5jZS1udW1iZXIiOiIwIn0sIm
dlbmVyYXRlVGltZXN0YW1wIjoxNTMxMTI3NDA5NDcyfQ]
```

Table 1-13 Response parameter description

Parameter	Type	Description
shardIterator	String	Data cursor. Value: 1 to 512 characters. **NOTE** The validity period of a cursor is 5 minutes.

 Copyright © Huawei Technologies Co., Ltd.

1.7.12 Creating a Consumer Application

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to create a consumer application.

```
String appName = "appName";
        var dic = new DISIngestionClient();
        var request = new CreateAppRequest
        {
//Name of the consumer application to be created.
            AppName = appName,
        };
        ResponseResult response = dic.CreateApp(request);
Console.Out.WriteLine(response);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
{ResponseResult{status_code='201 Created', content='', errorCode='', message=''}}
```

Table 1-14 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

1.7.13 Deleting a Consumer Application

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Run the DIS SDK command to delete the consumer application.

```
String appName = "appName";
        var dic = new DISIngestionClient();
        var request = new CreateAppRequest
        {
//Name of the consumer application to be deleted.
            AppName = appName,
        };

        var response = dic.DeleteApp(request);
        Console.Out.WriteLine(response);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
{ResponseResult{status_code='204 NO CONTENT', content='', errorCode='', message=''}}
```

 Copyright © Huawei Technologies Co., Ltd.

Table 1-15 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

1.7.14 Adding a Checkpoint

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK command to add a checkpoint.

```
string appName= "appName";
var dic = new DISIngestionClient();
        var request = new CommitCheckpointRequest
          {
//Stream name.
              stream_name = streamName,
//Application name.
              AppName = appName,
//Unique identifier of the partition.
              partition_id = "shardId-0000000000",
//Sequence number.
              sequence_number = "10",
//Metadata information of the consumer application.
              metadata = "metadata",
//Type of the checkpoint.
              checkpoint_type = "LAST_READ",
          };

          var response = dic.CommitCheckpoint(request);
          Console.Out.WriteLine(response);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
{ResponseResult{status_code='201 Created', content='', errorCode='', message=''}}
```

Table 1-16 Description of response parameter **ResponseResult**

Parameter	Type	Description
content	String	Response body.
errorCode	String	Error code.
message	String	Error response body.
status_code	String	Status code.

 Copyright © Huawei Technologies Co., Ltd.

1.7.15 Querying a Checkpoint

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to query the checkpoint.

```
string streamName = "streamName";
string appName= "appName";
        var dic = new DISIngestionClient();
        var request = new GetCheckpointRequest
        {
//Stream name.
            StreamName = streamName,
//Application ID.
            AppId = appName,
//Unique identifier of the partition.
            ShardId = "shardId-0000000000",
//Type of the checkpoint.
            CheckpointType = "LAST_READ",
        };

        var response = dic.GetCheckpoint(request);
        Console.Out.WriteLine(response);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
GetCheckpointResult [sequence_number=1,metadata=]
```

Table 1-17 Description of response parameter **GetCheckpointResult**

Parameter	Type	Description
sequence_number	String	Sequence number. A sequence number is the unique identifier of each record.
metadata	String	Metadata information of the consumer application.

1.7.16 Changing Partition Quantity

Initialize a DIS client as instructed in **Initializing a DIS SDK Client Instance**.

Use the DIS SDK to change the number of partitions.

```
string streamName = "streamName";
int count= 3;

var dic = new DISIngestionClient();
        var request = new UpdateShardsRequest
        {
//Stream name.
            StreamName = streamName,
//Number of the target partitions.
            TargetPartitionCount = count,
        };
```

 Copyright © Huawei Technologies Co., Ltd.

```
var response = dic.UpdatePartition(request);
Console.Out.WriteLine(response);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
{UpdateShardsResult{stream_name='dis-shaw', current_partition_count='1', target_partition_count='3'}}
```

Table 1-18 Description of response parameter **UpdateShardsResult**

Parameter	Type	Description
stream_name	String	Name of the stream whose partition quantity needs to be changed.
current_partition_count	int	Number of the current partitions
target_partition_count	int	Number of the target partitions.

1.8 Using the Python SDK

1.8.1 Preparing the Installation Environment

- Python 2.7 or a later version has been installed and its environment variables have been configured.
- PyCharm has been installed.

1.8.2 Configuring a Sample Project

The **huaweicloud-sdk-dis-python-*X.X.X*.zip** package downloaded from **SDK Download** provides a sample project. You can use a development tool (such as PyCharm) to compile and run the sample project on a local device. You can also develop applications based on the sample project. The code path of the example project is as follows: [$path]\huaweicloud-sdk-dis-python-X.X.X\dis_sdk_python-X.X\dis_sdk_python\examples

Procedure

Step 1 Obtain the **huaweicloud-sdk-dis-python-*X.X.X*.zip** package (you do not need to decompress it), launch the Command Prompt program in the directory of the current package, and enter **pip install huaweicloud-sdk-dis-python-*X.X.X*.zip**.

Step 2 Import the PyCharm project.

1. Choose **File** > **Open**. The **Open File or Project** window is displayed.

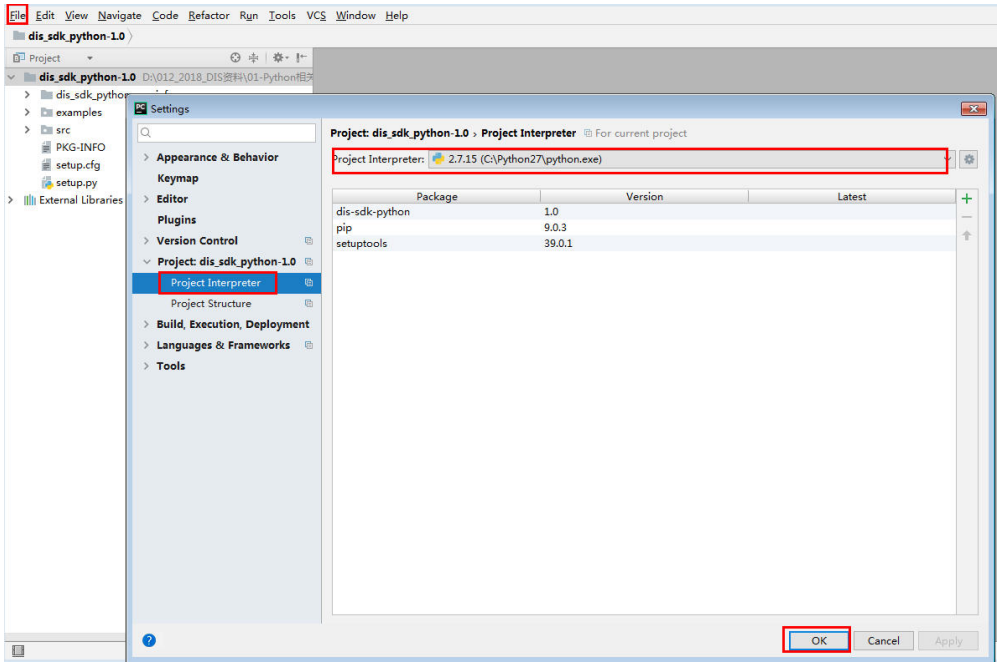
 Copyright © Huawei Technologies Co., Ltd.

2. Select **dis_sdk_python** from the **\Lib\site-packages** Python installation directory. (If **dis_sdk_python** cannot be found after the installation, upgrade the pip and reinstall the package.)
3. Click **OK**.

Step 3 Configure the **sdk_python** project.

1. In the navigation tree on the left, choose **File** > **Settings** > **Editor** > **Font** and set the font background color.
2. In the navigation tree on the left, choose **File** > **Settings** > **Project Interpreter** and add Python.
3. Select a proper project interpreter and click **OK**.

Figure 1-7 Adding Python



4. In the navigation tree on the left, choose **File** > **Settings** > **Editor** > **File Encodings** and set the PyCharm code.

   Set **Global Encoding**, **Project Encoding**, and **Default encoding for properties files** to **UTF-8**.

----End

1.8.3 Initializing a DIS SDK Client Instance

You can use either of the following methods to initialize the DIS SDK client instance: For details about **endpoint**, **ak**, **sk**, **region**, and **projectId**, see **Obtaining Authentication Information**.

- Use codes to initialize the DIS SDK client instance. For a code example, refer to all the files, except the **__init__.py** file, in the **dis_sdk_python-X.X\dis_sdk_python\examples** directory. The configuration parameters are as follows:

```
//Create a DIS SDK client instance.
projectid =" "       ID of the project for the DIS stream is used.
```

 Copyright © Huawei Technologies Co., Ltd.

```
endpoint=" "       Access address of DIS.
ak = " "           AK created by the user on IAM.
sk = " "           SK created by the user on IAM.
region = " "       Area where the stream used by a user is located.
```

- Use the configuration file to initialize the DIS SDK client instance.

  Set the configuration file information in the **initialization_dis_sdk.py** file under the **dis_sdk_python-*X.X*\dis_sdk_python** directory.

  ```
  conf.set('Section1', 'projectid', '** your projectid **')
  conf.set('Section1', 'ak', '*** Provide your Access Key ***')
  conf.set('Section1', 'sk', '*** Provide your Secret Key ***')
  conf.set('Section1', 'region', '** region **')
  conf.set('Section1', 'endpoint', '** {ip}:{port} ***')
  ```

  After configuring the preceding parameters, run **initialization_dis_sdk.py** to generate the **conf.ini** file and obtain the **dis_sdk_python\conf.ini** configuration file.

1.8.4 Creating a Stream

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **createstream_sample.py** file provides a code sample. Open the **createstream_sample.py** file and configure parameters according to the setting of **stream_type**. For details about how to configure the parameters, go to **Creating a Stream**.

- stream_type= " " #Not specified.

If **stream_type** is not specified, configure the parameters of **Dump_switch** in the **createstream_sample.py** file.

- stream_type= "FILE" #Set to file.

If **stream_type** is set to **File**, configure the parameters of **Dump_switch_FILE** in the **createstream_sample.py** file.

After configuring the parameters, run the **createstream_sample.py** file to call **createStream_test** by default. If response code 201 is returned, the stream is successfully created.

1.8.5 Creating a Dump Task

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **add_dump_task_sample.py** file provides a code sample. Open the **add_dump_task_sample.py** file and configure parameters. For details about how to configure the parameters, go to **Adding a Dump Task**.

```
Configure the following parameters:
streamname ="dis-test1"  #Name of an existing stream
task_name='113'
```

Adding an OBS dump task is used as an example. The **value** parameter corresponds to the **key** parameter.

```
basic_Schema=DumpTask.setSchema(key=['consumer_strategy','deliver_time_int
erval','agency_name','retry_duration'],
    value=['LATEST', 30, 'dis_admin_agency',1800])
```

 Copyright © Huawei Technologies Co., Ltd.

```
obs_dump_task =['destination_file_type','obs_bucket_path','file_prefix', 'partition_format','record_delimiter']
obs_Schema = DumpTask.setSchema(basic_Schema=basic_Schema,
                key=obs_dump_task,value=['text','obs-1253', '','yyyy', '|'])
```

#Add an OBS dump task and configure **osb_Schema**.

```
cli.add_dump_task(streamname, task_name,'OBS',obs_Schema)
```

1.8.6 Deleting a Stream

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **deleteStream_sample.py** file provides a code sample. Configure the following parameter in this file:

```
streamname = ""  #Name of an existing stream.
```

After configuring the parameter, run the **deleteStream_sample.py** file to call **deleteStream_test** by default. If response code 204 is returned, the stream is successfully deleted.

1.8.7 Deleting a Dump Task

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **delete_dump_task_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname = ""  #Name of an existing stream.
task_name="xx"
```

NOTE

After configuring **task_name**, the dump task will be deleted from the stream.

After configuring the parameters, run the **delete_dump_task_sample.py** file to call **delete_dump_task_test** by default. If response code 200 is returned, the dump task is successfully deleted.

1.8.8 Querying a Stream List

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **listStream_sample.py** file provides a code sample. Configure the following parameter in this file:

```
start_stream_name = ""  #Leave unspecified or set to an existing stream name.
```

Run the **listStream_sample.py** file to call **listStream_test**. If response code 200 is returned, the stream list is displayed.

The following is an example response:

```
200
{"stream_names": ["dis-test1", "dis-xx"], "has_more_streams": false, "stream_info_list": [{"status":
"RUNNING", "stream_name": "dis-xx", "data_type": "BLOB", "partition_count": 1, "stream_type":
"COMMON", "retention_period": 168, "create_time": 1532423804480}, {"status": "RUNNING",
"stream_name": "dis-test1", "data_type": "CSV", "partition_count": 1, "stream_type": "ADVANCED",
"retention_period": 24, "create_time": 1532328674686}], "total_number": 8}
```

 Copyright © Huawei Technologies Co., Ltd.

1.8.9 Querying a Dump List

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **list_dump_task_sample.py** file provides a code sample. Configure the following parameter in this file:

```
streamname = "XXX"  #Name of an existing stream.
```

Run the **list_dump_task_sample.py** file to call **list_dump_task_test**. If response code 200 is returned, the dump list is displayed.

The following is an example response:

```
200
{"has_more_tasks": false, "details": [{"state": "RUNNING", "create_time": 1535595851017,
"last_transfer_timestamp": 1535608269805, "task_name": "task_rsjV", "destination_type": "OBS"}],
"total_number": 1}
```

1.8.10 Querying Stream Details

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **describeStream_sample.py** file provides a code sample. Configure the following parameter in this file:

```
streamname = "dis-test1"  #Name of an existing stream.
```

After configuring the parameter, run the **describeStream_sample.py** file to call **describeStream_test** by default.

The following is an example response:

```
200
{"status": "RUNNING", "stream_name": "dis-test1", "data_type": "BLOB", "has_more_partitions": false,
"stream_type": "COMMON", "stream_id": "L84hxfES223eVrFyxiE", "retention_period": 168, "create_time":
1532423353637, "last_modified_time": 1532423354625, "partitions": [{"status": "ACTIVE", "hash_range":
"[0 : 9223372036854775807]", "sequence_number_range": "[0 : 10]", "partition_id":
"shardId-0000000000"}]}
```

1.8.11 Querying Dump Details

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **describe_dump_task_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname ="dis-test1"  #Name of an existing stream.
task_name="test_1"  #Query the xx dump task of the stream.
```

After configuring the parameters, run the **describe_dump_task_sample.py** file to call **describe_dump_task_test** by default.

The following is an example response:

```
200
{'state': 'RUNNING', 'create_time': 1536562174110, 'task_name': 'test_1', 'destination_type': 'OBS',
'last_transfer_timestamp': 1536562183057}
```

 Copyright © Huawei Technologies Co., Ltd.

1.8.12 Uploading Streaming Data in JSON Format

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **putRecords_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname ="dis-test1"  #Name of an existing stream.
```

records of the **putRecords_test** method in the **putRecords_sample.py** file is the data to be uploaded. The data is uploaded in the following format:

```
records=[{"data": "abcdefd", "partition_key": "1"}]
#data: data to be uploaded. The value is user-definable; partition_key: partition to which data is written. The value is user-definable.
 record1 = {"data": "xxx","partition_key": partition_key}
#You can write multiple pieces of data. The data format is shown in record1. Each time a piece of data is written, the append method is used to transfer the data to the records.
```

After configuring the parameters, run the **putRecords_sample.py** file to call **putRecords_test**. The following is an example response:

```
200
{'failed_record_count': 0, 'records': [{'partition_id': 'shardId-0000000001', 'sequence_number': '15'}]}
```

1.8.13 Uploading Streaming Data in Protobuf Format

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **protobuf_putRecords_sample.py** file provides a code sample. Configure the following parameters in this file:

NOTE

Run the **protoc --version** command to check whether the environment variables are set successfully. If not, install Protobuf.

```
streamname ="dis-test1"  #Name of an existing stream.
```

Parameter **bodySerializeType** in the **test** method must be set to **protobuf** so that streaming data is uploaded in protobuf format.

records of the **protobuf_putRecords_test** method in the **protobuf_putRecords_sample.py** file is the data to be uploaded. The data is uploaded in the following format:

```
records=[{"data": "abcdefd", "partition_key": "1"}]
#data: data to be uploaded. The value is user-definable; partition_key: partition to which data is written. The value is user-definable.
 record1 = {"data": "xxx","partition_key": partition_key}
#You can write multiple pieces of data. The data format is shown in record1. Each time a piece of data is written, the append method is used to transfer the data to the records.
```

After configuring the parameters, run the **protobuf_putRecords_sample.py** file to call **protobuf_putRecords_test**. The following is an example response:

```
200
{'failed_record_count': 0, 'records': [{'partition_id': 'shardId-0000000001', 'sequence_number': '15'}]}
```

 Copyright © Huawei Technologies Co., Ltd.

1.8.14 Downloading Streaming Data

Downloading Stream Data in JSON Format

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **getRecords_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname=" "   #Name of the stream.
startSeq=' 0' #Sequence number.
partitionId="shardId-0000000000"
```

Run **getCursor_test** to change the value of **cursorType** to **AT_SEQUENCE_NUMBER**.

After configuring the parameters, run the **getRecords_sample.py** file to call **getRecords_test**. The following is an example response:

```
200
{"records": [{"sequence_number": "21", "data": "abcdefd"}],"next_partition_cursor":
"eyJpdGVyR2VuVGltZSI6MTQ5MDk1MDE1Nzc0NywiU3RyZWFtTmFtZSI6IjY2MCIsIlNoYXJkSWQiOiIwIiwiU2hh
cmRJdGVyYXRvclR5cGUiOiJBVF9TRVFVRU5DRV9OVU1CRVIiLCJTdGFydGluZ1NlcXVlbmNlTnVtYmVyIjoiMjIiLCJ
UaW1lU3RhbXAiOjB9"}
```

Downloading Streaming Data in Protobuf Format

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **protobuf_getrecords_sample.py** file provides a code sample.

NOTE

Run the **protoc --version** command to check whether the environment variables are set successfully. If not, install Protobuf.

Configure the following parameters in this file:

```
streamname=" "   #Name of the stream.
startSeq=' 0'  #Sequence number.
partitionId="shardId-0000000000"
```

Run **getCursor_test** to change the value of **cursorType** to **AT_SEQUENCE_NUMBER**.

NOTE

The **test_0** method is used to obtain a cursor and the **test** method is used to download data. Compared to **test_0**, **test** has the **bodySerializeType="protobuf"** parameter, which is not provided in **test_0**.

After configuring the parameters, run the **protobuf_getrecords_sample.py** file to call **getRecords_test**. The following is an example response:

```
200
{"records": [{"sequence_number": "21", "data": "abcdefd"}],"next_partition_cursor":
"eyJpdGVyR2VuVGltZSI6MTQ5MDk1MDE1Nzc0NywiU3RyZWFtTmFtZSI6IjY2MCIsIlNoYXJkSWQiOiIwIiwiU2hh
cmRJdGVyYXRvclR5cGUiOiJBVF9TRVFVRU5DRV9OVU1CRVIiLCJTdGFydGluZ1NlcXVlbmNlTnVtYmVyIjoiMjIiLCJ
UaW1lU3RhbXAiOjB9"}
```

 Copyright © Huawei Technologies Co., Ltd.

1.8.15 Creating an Application

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **createApp_sample.py** file provides a code sample. Configure the following parameter in this file:

```
appName = ""  #Application name.
```

After configuring the parameter, run the **createApp_sample.py** file to call **createApp_test**. If response code 201 is returned, the application is successfully created.

1.8.16 Deleting an Application

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **deleteApp_sample.py** file provides a code sample. Configure the following parameter in this file:

```
appName = "" #Name of the application to be deleted.
```

After configuring the parameter, run the **deleteApp_sample.py** file to call **deleteApp_test**. If response code 204 is returned, the application is successfully deleted.

1.8.17 Viewing Application Details

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **describeApp_sample.py** file provides a code sample. Configure the following parameter in this file:

```
appname= "app1"  #Name of the application to be queried.
```

After configuring the parameter, run the **describeApp_sample.py** file to call **describeApp_test**.

The following is an example response:

```
200
{'app_name': 'app1', 'app_id': 'OPKQuggQVtfqhyvK0cs', 'create_time': 1532425956631}
```

1.8.18 Querying an Application List

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **Applist_sample.py** file provides a code sample. Configure the following parameter in this file:

```
appname="app1"  #Application name. The application list is returned from the current stream and does not contain the application.)
```

After configuring the parameter, run the **Applist_sample.py** file to call **Applist_test**.

 Copyright © Huawei Technologies Co., Ltd.

The following is an example response:

```
200
{'total_number': 2, 'apps_list': ['app1'], 'appinfo_list': [{'create_time': 1532425956631, 'app_id':
'OPKQuggQVtfqhyvK0cs', 'app_name': 'app1'}], 'has_more_app': False}
```

1.8.19 Adding a Checkpoint

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **commitCheckpoint_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname=" "   #Name of the stream.
appName="xx"  #Name of an existing application.
partitionId="shardId-0000000000"   #Unique identifier of the partition.
seqNumber="0"   #Sequence number.
metadata=""  #Metadata information of the consumer application. The maximum length of the metadata
information is 1,000 characters.
```

NOTE

Before obtaining the partition ID, import the stream name and then refer to the instructions in **Querying Stream Details**.

After configuring the parameters, run the **commitCheckpoint_sample.py** file to call **commitCheckpoint_test**. If response code 201 is returned, the checkpoint is successfully added.

1.8.20 Querying a Checkpoint

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **getCheckpoint_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname=" "   #Name of the stream.
appName="xx"  #Name of an existing application.
partitionId="shardId-0000000000"   #Unique identifier of the partition.
```

NOTE

Before obtaining the partition ID, import the stream name and then refer to the instructions in **Querying Stream Details**.

After configuring the parameters, run the **getCheckpoint_sample.py** file to call **getCheckpoint_test**. The following is an example response:

```
{
  "sequence_number": "10",
  "metadata": "metadata"
}
```

1.8.21 Changing Partition Quantity

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **changepartitionQuantity_sample.py** file provides a code sample. Configure the following parameters in this file:

 Copyright © Huawei Technologies Co., Ltd.

```
streamname = "" #Name of the running stream.
target_partition_count = "3"  #Number of the target partitions.
```

After configuring the parameters, run the **changepartitionQuantity_sample.py** file to call **changepartitionQuantity_test**. The following is an example response:

```
{
 "stream_name":"stream_name_test",
 "current_partition_count":2
 "target_partition_count":5
 }
```

1.8.22 Obtaining a Data Cursor

Initialize a DIS SDK client instance as instructed in **Initializing a DIS SDK Client Instance**.

The **getCursor_sample.py** file provides a code sample. Configure the following parameters in this file:

```
streamname=" "    #Name of the stream.
startSeq=' 0'  #Sequence number.
partitionId="shardId-0000000000"
cursorType    #cursor type (AT_SEQUENCE_NUMBER, AFTER_SEQUENCE_NUMBER, TRIM_HORIZON, or
LATEST)
```

After configuring the parameters, run the **getCursor_sample.py** file to call **getCursor_test**. The following is an example response:

```
200
{"partition_cursor":
"eyJnZXRJdGVyYXRvclBhcmFtIjp7InN0cmVhbS1uYW1lIjoiSCIsInBhcnRpdGlvbi1pZCI6InNoYXJkSWQtMDAwMD
AwMDAwMCIsImN1cnNvci10eXBlIjoiQVRfU0VRVUVOQ0VfTlVNQkVSIiwic3RhcnRpbmctc2VxdWVuY2UtbnVt
YmVyIjoiMCJ9LCJnZW5lcmF0ZVRpbWVzdGFtcCI6MTUzMjQyNDg4NzE1NH0"}
```

1.9 Using the C SDK

1.9.1 Preparing the Installation Environment

Install GNU Compiler Collection (GCC) for Linux.

1.9.2 Configuring a Sample Project

Step 1 Obtain the **huaweicloud-sdk-dis-c-*XXX*.zip** development package from **SDK Download** and upload it to any directory in Linux. Go to the directory and run the **unzip huaweicloud-sdk-dis-c-*XXX*.zip** command to decompress the package. The package contains the **dis-source** folder (source code), **sample** folder (sample code), and **third_party** folder (third-party class) are displayed.

Step 2 Go to the **dis-source/src** directory and run the **protoc-c --c_out=. httpbody.proto** command. The **httpbody.pb-c.c** and **httpbody.pb-c.h** files are generated in the current directory.

Step 3 Return to the **dis-source** folder and run the **make** command to compile source codes.

 Copyright © Huawei Technologies Co., Ltd.

NOTE

The warning information generated during command execution will not adversely affect the program. After the command is executed, a **build** folder is generated to store the dynamic link library.

Step 4 Run the **./build.sh** command.

The following information is displayed in the command output:

```
-bash: ./build.sh: /bin/bash^M: bad interpreter: No such file or directory
```

Run the following commands:

chmod 700 ./build.sh

vim build.sh

:set ff=unix

:wq!

NOTE

The warning information generated during command execution will not adversely affect the program. After the command is executed, a **dis_api** folder is generated. Files in this folder are required for executing the sample code.

Step 5 Go to the **sample** folder and modify the DIS parameters in **test.c**, such as **ak**, **sk**, and **projectId**.

Step 6 Run the **make** command to compile the program and generate the executable file **dis_bin**.

Step 7 Run the **./dis_bin** command to check the running result of the program.

----End

1.9.3 Configuring an AK and SK

In **test.c**, use the **GetUserAuthInfo** function to set an AK and SK.

```
DISStatus GetUserAuthInfo(char *ProjectId, char *akArray, char *skArray, char *xSecurityToken)
{
char *ak = "Provide your Access Key";
char *sk = "Provide your Secret Key";
printfdbgfunc;
strncpy(akArray, ak, strlen(ak));
strncpy(skArray, sk, strlen(sk));
printfdbgoutfunc;
return 0;
}
```

1.9.4 Initializing DIS

The following describes how to use the **main** function to configure DIS parameters, initialize DIS, set a transmission mode, call upload and download samples, and perform recycling.

```
int main()
{
int ret = 0;
FILE *logFile = NULL;
```

 Copyright © Huawei Technologies Co., Ltd.

```
//Specify the project ID of the stream.
char *projectId = "c159a24641da49b2a729ea6f57647888";
//Specify the region where the stream is used.
char *region = "******";
//Specify the access address of DIS.
char *host = "dis.cn-north-1.myhuaweicloud.com:20004";
//Specify the stream name.
char *streamName = "lifei_test";

srand((unsigned) time(NULL ));
logFile = fopen("log.txt", "a+");
if(NULL == logFile)
{
printf("open file error\r\n");
return 1;
}
printf("step 1\r\n");
ret = DisInit(logFile, GetUserAuthInfo);
if(0 != ret)
{
printf("init error: %d\r\n", ret);
getchar();
return ret;
}
printf("start test\r\n");

logPrint(1, "this is a test: %d", ret);
.
testSendRecord(streamName);
testGetRecord(streamName);

getchar();
DisDeinit();

return 0;
}
```

- To initialize DIS, use DisInit(logFile, GetUserAuthInfo).
- To set the serialization mode of data transmission to protobuf, use DISSetSerializedMode("protobuf").
- To set the serialization mode of data transmission to base64, use DISSetSerializedMode("base64").
- To send data, use testSendRecord(streamName).
- To download data, use testGetRecord(streamName).

1.9.5 Creating a Stream Whose Source Data Type Is BLOB, JSON, or CSV

```
//Configure the stream name.
char *pucStreamName = "myStream";
char *projectId = "d575b0b740e54221aeb9a165653b103d";
char *region = "southchina";
char *host = "XXX.XXX.XXX.XXX:XXX";
int ret = 0;
DISResponseInfo RspInfo = { 0 };
DISCreateStream *pstCreateStream = disMemAlloc(sizeof(DISCreateStream));
DISCreateStreamExpend *pstCreateStreamExpend = disMemAlloc(sizeof(DISCreateStreamExpend));
pstCreateStreamExpend->dataType = DISDataTypeBlob;
pstCreateStreamExpend->dataDuration = 2 * 24;

pstCreateStream->partitionCount = 10;
pstCreateStream->streamName = pucStreamName;
pstCreateStream->streamType = DISStreamTypeAdvanced;
pstCreateStream->pucReserved = pstCreateStreamExpend;
```

 Copyright © Huawei Technologies Co., Ltd.

```
printf("==================%s Begin======================\n", __FUNCTION__);
ret = CreateStream(host, projectId, region, pstCreateStream, &RspInfo);
if (ret != 0)
{
printf("Create Error: %d\r\n", ret);
printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
printf("ErrorCode: %s\r\n", RspInfo.ErrorCode);
printf("ErrorDetail: %s\r\n", RspInfo.ErrorDetail);
}
else
{
printf("Create stream %s success\n", pstCreateStream->streamName);
printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
}

disMemFree(pstCreateStream);
printf("==================%s End======================\n", __FUNCTION__);
```

After configuring **DISCreateStream**, you can create a stream by calling createStream.

```
ret = CreateStream(host, projectId, region, pstCreateStream, &RspInfo);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

```
Create stream myStream success
HttpResponseCode: 201
```

1.9.6 Deleting a Stream

Use the DIS SDK to delete a specified DIS stream.

```
//Specify the stream to be deleted.
char *streamName = "myStream";
char *projectId = "d575b0b740e54221aeb9a165653b103d";
char *region = "southchina";
char *host = "XXX.XXX.XXX.XXX:XXX";
int ret = 0;
DISResponseInfo RspInfo = {0};
printf("==================%s Begin======================\n",__FUNCTION__);
ret = DeleteStream(host, projectId, region, streamName, &RspInfo);
    if(ret != 0)
    {
        printf("Delete Error: %d\r\n", ret);
        printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
        printf("ErrorCode: %s\r\n", RspInfo.ErrorCode);
        printf("ErrorDetail: %s\r\n", RspInfo.ErrorDetail);
    }
    else
    {
        printf("Delete stream %s success\r\n ",streamName);
        printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
}
printf("==================%s End======================\n",__FUNCTION__);
```

Delete a stream by calling **DeleteStream**.

```
ret = DeleteStream(host, projectId, region, streamName, &RspInfo);
```

Viewing Execution Results

Information similar to the following is displayed on the console:

 Copyright © Huawei Technologies Co., Ltd.

```
Delete stream myStream success
HttpResponseCode: 204
```

1.9.7 Querying a Stream List

Use the DIS SDK to query active streams.

Use the **Limit** method to set the number of streams returned each time. If **Limit** is not specified, a maximum of 10 streams are returned by default.

```
   char *projectId = "d575b0b740e54221aeb9a165653b103d";
   char *region = "southchina";
   char *service = "dis";
   char *host = "XXX.XXX.XXX.XXX:XXX";
   int ret = 0;
  int i = 0;
   DISResponseInfo RspInfo = {0};
DISListStream listStream = {0};
//Set the value to 2.
listStream.limit = 2;
//Query from the dis-shawn stream. The result does not contain this stream.
listStream.startStreamName = "dis-shawn";
   printf("==================%s Begin=====================\n",__FUNCTION__);
   ret = ListStream(host, projectId, region, &listStream, &RspInfo);
   if(ret != 0)
   {
       printf("List Error: %d\r\n", ret);
       printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
       printf("ErrorCode: %s\r\n", RspInfo.ErrorCode);
       printf("ErrorDetail: %s\r\n", RspInfo.ErrorDetail);
       return;
   }
   printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
printf("the totalcount is %d, the current count is %d\r\n", listStream.streamListResult.totalNumber,
listStream.streamListResult.currentNumber);
   printf("hasMoreStream: %ld\r\n", listStream.streamListResult.hasMoreStream);
   printf("streamList include: ");
   for (i = 0; i < listStream.streamListResult.currentNumber; i++)
   {
 printf("%s, ", listStream.streamListResult.streamList[i].streamName);
 }
 printf("\b\b \r\n");
    printf("==================%s End=====================\n",__FUNCTION__);
```

Table 1-19 DISListStream parameter description

Parameter	Type	Description
limit	long	The maximum number of DIS streams to list in a single API call. Value range: 1-100 Default value: 10
startStreamName	char *	Name of the DIS stream to start the stream list with. The returned stream list does not contain this DIS stream name.

Copyright © Huawei Technologies Co., Ltd.

Viewing Execution Results

Information similar to the following is displayed on the console:

```
HttpResponseCode: 200
the totalcount is 4, the current count is 2
hasMoreStream: 0
streamList include: dis-shawn-1, testmonit
```

Table 1-20 DISListStream parameter description

Parameter	Type	Description
streamListResult	DISListStreamResult	Stream list.

Table 1-21 DISListStreamResult parameter description

Parameter	Type	Description
totalNumber	long	Total number of all the DIS streams created by the current tenant.
streamList	DISStream [100]	Name of the list of the streams meeting the current requests.
hasMoreStream	long	Specify whether there are more matching DIS streams to list. Possible values:
currentNumber	long	Number of streams that meet the current condition.

Table 1-22 DISStream parameter description

Parameter	Type	Description
streamName	char [64]	Stream name array.

1.9.8 Querying Stream Details

Use the DIS SDK to query the details about a specified stream.

```
char *streamName = "myStream";
char *projectId = "d575b0b740e54221aeb9a165653b103d";
char *region = "southchina";
int ret = 0;
int i = 0;
DISResponseInfo RspInfo = { 0 };
DISDescribeStream streamDetail = { 0 };
streamDetail.streamName = streamName;
//streamDetail.startPartitionId="";
//streamDetail.limitPartitionNum=10;
printf("==================%s Begin======================\n", __FUNCTION__);
```

 Copyright © Huawei Technologies Co., Ltd.

```
ret = DescribeStream(host, projectId, region, &streamDetail, &RspInfo);
if (ret != 0)
{
printf("List Error: %d\r\n", ret);
printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
printf("ErrorCode: %s\r\n", RspInfo.ErrorCode);
printf("ErrorDetail: %s\r\n", RspInfo.ErrorDetail);
return;
}
printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
printf("streamId:%s\n", streamDetail.streamInfo.streamId);
printf("streamName:%s\n", streamDetail.streamInfo.streamName);
printf("createTime:%ld\n", streamDetail.streamInfo.createTime);
printf("lastModifyTime:%ld\n", streamDetail.streamInfo.lastModifyTime);
printf("retentionPeriod:%ld\n", streamDetail.streamInfo.retentionPeriod);
printf("dataType:%s\n", streamDetail.streamInfo.dataType == DISDataTypeBlob ? "BLOB"
: (streamDetail.streamInfo.dataType == DISDataTypeJson) ? "JSON"
: (streamDetail.streamInfo.dataType == DISDataTypeCsv) ? "CSV"
: "FILE");
printf("streamStatus:%s\n", streamDetail.streamInfo.streamStatus == DISStreamStatusCreating ?
"CREATING"
: (streamDetail.streamInfo.streamStatus == DISStreamStatusRunning) ? "RUNNING"
: (streamDetail.streamInfo.streamStatus == DISStreamStatusTerminating) ? "TERMINATING"
: "FROZEN");
printf("streamType:%s\n", streamDetail.streamInfo.streamType == DISStreamTypeCommon ? "Common" :
"Advanced");
printf("hasMorepartition:%s\n", streamDetail.streamInfo.hasMorepartition == 0 ? "false" : "true");
printf("currentNumber:%ld\n", streamDetail.streamInfo.currentNumber);
for (i = 0; i < streamDetail.streamInfo.currentNumber; i++)
{
printf("loop %d partitionStatus:%s\n", i, streamDetail.streamInfo.partitionList[i].partitionStatus ==
DISPartitionStatusCreating ? "CREATING"
: (streamDetail.streamInfo.partitionList[i].partitionStatus == DISPartitionStatusActive) ? "ACTIVE"
: (streamDetail.streamInfo.partitionList[i].partitionStatus == DISPartitionStatusDeleted) ? "DELETED"
: "EXPIRED");
printf("loop %d partitionId:%s\n", i, streamDetail.streamInfo.partitionList[i].partitionId);
printf("loop %d hashRange:%s\n", i, streamDetail.streamInfo.partitionList[i].hashRange);
printf("loop %d sequenceNumberRange:%s\n", i,
streamDetail.streamInfo.partitionList[i].sequenceNumberRange);
}
printf("==================%s End======================\n", __FUNCTION__);
```

Table 1-23 DISDescribeStream parameter description

Parameter	Mandatory	Type	Description
streamName	Yes	Char *	Stream to be queried.
startPartitionId	No	Char *	Name of the partition to start the partition list with. The returned partition list does not contain this partition.
limitPartitionNum	No	long	Max. number of partitions to list in a single API call. Value range: 1-1000 Default value: 100

The returned stream details are as follows:

 Copyright © Huawei Technologies Co., Ltd.

```
HttpResponseCode: 200
streamId:fXdsj6XzvxKJVLQoplg
streamName: myStream
createTime:1536546556151
lastModifyTime:1536546557001
retentionPeriod:72
dataType:FILE
streamStatus:RUNNING
streamType:Advanced
hasMorepartition:false
currentNumber:2
loop 0 partitionStatus:ACTIVE
loop 0 partitionId:shardId-0000000000
loop 0 hashRange:[0 : 4611686018427387902]
loop 0 sequenceNumberRange:[0 : 0]
loop 1 partitionStatus:ACTIVE
loop 1 partitionId:shardId-0000000001
loop 1 hashRange:[4611686018427387903 : 9223372036854775807]
loop 1 sequenceNumberRange:[0 : 0]
```

Table 1-24 DISDescribeStream parameter description

Parameter	Type	Description
streamInfo	DISStreamDetailInfo	Stream details obtained through response.

Table 1-25 DISStreamDetailInfo parameter description

Parameter	Type	Description
streamName	Char *	Name of the DIS stream to be created.
streamId	Char *	Unique identifier of the stream.
createTime	Long	Timestamp at which the stream was created.
lastModifyTime	Long	Timestamp at which the stream was most recently modified.
retentionPeriod	Long	Period of time for which data is retained in the stream.
streamStatus	DISStreamStatus	Current status of the stream. Possible values: • DISStreamStatusCreating • DISStreamStatusRunning • DISStreamStatusTerminating • DISStreamStatusFrozen
streamType	DISStreamType	Stream type. • DISStreamTypeCommon • DISStreamTypeAdvanced

Copyright © Huawei Technologies Co., Ltd.

Parameter	Type	Description
hasMorepartition	long	Specify whether there are more matching partitions of the DIS stream to list. • **1**: yes • **0**: no
dataType	DISDataType	Source Data Type • DISDataTypeBlob • DISDataTypeJson • DISDataTypeCsv • DISDataTypeFile
currentNumber	long	Number of partitions.
partitionList	DISPartitionDetai-lInfo[100]	Partition list of the stream

Table 1-26 DISPartitionDetailInfo parameter description

Parameter	Type	Description
partitionStatus	DISPartitionStatus	Current status of the partition. Possible values: • DISPartitionStatusCreating • DISPartitionStatusActive • DISPartitionStatusDeleted • DISPartitionStatusExpired
partitionId	Char [128]	Unique identifier of the partition.
hashRange	Char [128]	Possible value range of the hash key used by the partition.
sequenceNumberRange	Char [128]	Sequence number range of the partition.

1.9.9 Uploading Streaming Data

Sample Code

Data can be uploaded in base64 or protobuf mode.

- To set the upload mode to base64, use **DISSetSerializedMode("base64")**.
- To set the upload mode to protobuf, use **DISSetSerializedMode("protobuf")**.

To obtain the upload mode set by the user, use **DISGetSerializedMode()**.

The code for uploading streaming data is as follows:

 Copyright © Huawei Technologies Co., Ltd.

```
#define MAX_RECORD_COUNT   500
#define MAX_RECORD_LEN   (1024*10)
```

The upload mode is synchronous upload. A long delay occurs when the data volume is large. To avoid a long delay, a maximum of 500 records can be uploaded at a time, and the data volume in a record cannot exceed 10,000. If the limit is exceeded, the program displays the following message:

```
Send Record Faiure, the httprspcode is 0. ret: 5
ErrorCode:
ErrorDetail:
```

ret: 5 indicates an invalid parameter.

The following is a specific implementation method:

```
char*streamName = "myStream";
char *host = "XXX.XXX.XXX.XXX:XXX";
char *region="southchina";
char *projectId="43eec61d4b514f359c97008a4f8bfb02";
 int ret = 0;
DISResponseInfo RspInfo = {0};
int serrializedMode = DISGetSerializedMode();
 DISPutRecord Record[MAX_RECORD_COUNT] = {0};
int i = 0;
int count = MAX_RECORD_COUNT;
int len = MAX_RECORD_LEN;
char *pmsg[MAX_RECORD_COUNT];
printf("==================%s Begin=======================\n", __FUNCTION__);
if (GetProtobufMode() == serrializedMode)
{
for (i = 0; i < count; i++)
{
Record[i].recordData.stringLen = rand() % len;
pmsg[i] = malloc(Record[i].recordData.stringLen);
Record[i].recordData.data = genRandomStringWithoutNull(Record[i].recordData.stringLen, pmsg[i]);
Record[i].partitionKey = "2";
}
}
else if (GetBase64Mode() == serrializedMode)
{
for (i = 0; i < count; i++)
{
Record[i].recordData.stringLen = rand() % len;
pmsg[i] = malloc(Record[i].recordData.stringLen);
Record[i].recordData.data = genRandomString(Record[i].recordData.stringLen, pmsg[i]);
Record[i].partitionKey = "2";
}
}
    printf("inputrecord:%s\n",Record[0].recordData.data);
    printf("inputrecord:%s\n",Record[1].recordData.data);
    printf("inputrecord:%s\n",Record[2].recordData.data);
    ret = PutRecords(host, projectId, region, streamName, 3, Record, PutRecordCallBack, &RspInfo);
    if(ret != 0 || RspInfo.HttpResponseCode >= 300)
    {
        printf("Send Record Faiure, the httprspcode is %d. ret: %d\r\n", RspInfo.HttpResponseCode, ret);
        printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
        printf("ErrorCode: %s\r\n", RspInfo.ErrorCode);
        printf("ErrorDetail: %s\r\n", RspInfo.ErrorDetail);
    }
    else
    {
        printf("send Record Success,ret:%d\n",RspInfo.HttpResponseCode);
        printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
}
for (i = 0; i < count; i++)
{
free(pmsg[i]);
```

 Copyright © Huawei Technologies Co., Ltd.

```
}
printf("==================%s End======================\n",__FUNCTION__);
```

Table 1-27 DISPutRecord parameter description

Parameter	Mandatory	Type	Description
recordData	Yes	DISString	Data to be uploaded.
explicitHashKey	No	Char *	Specify the hash value of the data to be written into the partition. The hash value overwrites the hash value of **partitionKey**.
partitionId	No	Char *	Unique identifier of the partition.
partitionKey	No	Char *	Partition into which data will be written. **NOTE** If **partitionId** is specified, it is preferentially used. If **partitionId** is not specified, use **partitionKey**.
pucReserved	No	void *	Empty pointer, used to extend the request body.

Table 1-28 DISString parameter description

Parameter	Mandatory	Type	Description
stringLen	Yes	long	Length of the data to be uploaded.
data	Yes	Char *	Data to be uploaded.

The **PutRecords** method has a callback function **PutRecordCallBack**, which is used to process responses. This function can be customized. An example is provided for reference only.

The following shows how to use the callback function:

```
DISStatus PutRecordCallBack(char *errorCode, char *errorDetails, char *streamName, DISPutRecord
*putRecord, char *SeqNumber, char *partitiodId)
{
    if(NULL != SeqNumber)
    {
        printf("Send Record:%s. key: %s success, the seqnum:%s, pid:%s\r\n", putRecord->recordData.data,
putRecord->partitionKey, SeqNumber, partitiodId);
    }
    else
    {
        printf("Send Record:%s. key: %s fail, the errCode:%s, message:%s\r\n", putRecord->recordData.data,
putRecord->partitionKey, errorCode, errorDetails);
```

 Copyright © Huawei Technologies Co., Ltd.

```
    }
    return 0;
}
```

The following table describes the response parameters:

Parameter	Type	Description
errorCode	char *	Error code.
errorDetails	char *	Error message.
streamName	char *	Stream name.
putRecord	DISPutRecord *	Data to be uploaded.
partitiodId	char *	Partition ID.
SeqNumber	char *	Sequence number. A sequence number is the unique identifier of each record. DIS automatically allocates a sequence number when the data producer calls the PutRecords operation to add data to the DIS stream. The sequence number of the same partition key usually changes with time. A longer interval between PutRecords requests results in a larger sequence number.

Viewing Execution Results

Information similar to the following is displayed on the console:

```
Send Record:aaaaaaaaaa. key: 1 success, the seqnum:190, pid:shardId-0000000000
Send Record:
0EDk3AwJuBQOdtTZbTUnw21ruxbRpNE4WMY71qIH3pujwWo93d9Puzojp0jYahVD06w2FZbL1dgaxz238So07
RMYHr02D2CQW4T0DE5MF39KYILr9uR81mm2h6T5TU781kmT8xjBKWN5g9h33KN2QX0332xbnJ5ItJ3tu48Y
WOTzf3hKk7yk9aNp1fUXrbteIxyEOS48m4e17X. key: 1 success, the seqnum:191, pid:shardId-0000000000
Send
Record:dHsTTR40lmym0l0Oxh580IIiuesLcnzYa27yyQ7e5nD40Td2z59OEFhU88HK3BxUfiNZdVpSaswNixL0lv7
Tye4fl0Cp16g7PT5WflWZR9D83pCWUXfLTB9BXG1dO14266FOJs9N6Th3SqFRNDe5l6Jz5Vt07uyeeiDYRwwerF
9y0RG41Vga. key: 2 success, the seqnum:192, pid:shardId-0000000000
send Record Success,ret:200
HttpResponseCode: 200
```

1.9.10 Downloading Streaming Data

Sample Code

Data can be uploaded in base64 or protobuf mode.

- To set the upload mode to base64, use **DISSetSerializedMode("base64")**.
- To set the upload mode to protobuf, use **DISSetSerializedMode("protobuf")**.
- To obtain the upload mode set by the user, use **DISGetSerializedMode()**.

 Copyright © Huawei Technologies Co., Ltd.

The main codes for downloading streaming data are as follows: The value of **streamName** must be the same as the value of **Stream Name** in section "Creating a Stream."

```
char *streamName = "myStream";
char *projectId = "d575b0b740e54221aeb9a165653b103d";
char *region = "southchina";
char *host = "XXX.XXX.XXX.XXX:XXX";
int ret = 0;
DISResponseInfo RspInfo = { 0 };
DISCursor  NestCursor = { 0 };
printf("====================%s Begin=======================\n", __FUNCTION__);
DISGetCursor tempCursor = { 0 };
tempCursor.streamName = streamName;
tempCursor.partitionId = "shardId-0000000002";
tempCursor.cursorType = DISCursorTypeAtSeqNumber;
tempCursor.startingSequenceNumber = "1";
ret = DisGetCursor(host, projectId, region, &tempCursor, &RspInfo);
if (ret != 0 || RspInfo.HttpResponseCode >= 300)
{
printf("GetCursor Error: %d, the http code id %d, Error Code :%s, message: %s\r\n", ret,
RspInfo.HttpResponseCode, RspInfo.ErrorCode, RspInfo.ErrorDetail);
printf("====================%s End=======================\n", __FUNCTION__);
return;
}
printf("====================Got Cursor=======================\n");
RspInfo.HttpResponseCode = 0;
ret = GetRecords(host, projectId, region, tempCursor.streamName, tempCursor.cursorResult.partitionCursor,
0, &NestCursor, GetRecordCallBack, &RspInfo);
if (ret != 0 || RspInfo.HttpResponseCode >= 300)
{
printf("GetRecords Error: %d, the http code id %d, Error Code :%s, message: %s\r\n", ret,
RspInfo.HttpResponseCode, RspInfo.ErrorCode, RspInfo.ErrorDetail);
printf("====================%s Begin=======================\n", __FUNCTION__);
return;
}
```

Before downloading streaming data, you need to obtain cursor data. **Obtaining a Data Cursor** describes the cursor parameters.

Table 1-29 Cursor parameters

Parameter	Mandatory	Type	Description
partitionCursor	Yes	Char *	Data cursor. Value: 1 to 512 characters. **NOTE** The validity period of a cursor is 5 minutes.

The **GetRecords** method has a callback function **GetRecordCallBack**, which is used to process responses. This function can be customized. An example is provided for reference only.

```
DISStatus GetRecordCallBack(char *streamName, const DISGetRecords *record)
{
if (NULL == streamName || NULL == record || NULL == record->seqNumber || NULL == record->partitionId)
{
printf("%s the input param invalid\r\n", __FUNCTION__);
return 0;
}
```

 Copyright © Huawei Technologies Co., Ltd.

```
printf("the partition key is %s\r\n", record->partitionId);
printf("the seqNumber is %s\r\n", record->seqNumber);
//record->data.data indicates the start address of the data, and record->data.stringLen indicates the data length.
printf("the data is %d: %s\r\n", record->data.stringLen, record->data.data);
return 0;
}
```

The following table describes the response parameters:

Parameter	Type	Description
streamName	char *	Stream name.
record	DISGetRecords *	Record details.

Table 1-30 DISGetRecords parameter description

Parameter	Type	Description
partition_key	String	Partition key set when data is being uploaded. **NOTE** If **partition_key** is specified when data is uploaded, **partition_key** is returned when data is downloaded. If **partition_id** instead of **partition_key** is specified when data is uploaded, no **partition_key** is returned.
seqNumber	Char *	Sequence number. A sequence number is the unique identifier of each record. DIS automatically allocates a sequence number when the data producer calls the PutRecords operation to add data to the DIS stream. The sequence number of the same partition key usually changes with time. A longer interval between PutRecords requests results in a larger sequence number.
timestamp	long long	Timestamp when the record is written to DIS.
timestampType	Char *	Type of the timestamp. The value is **CreateTime**, specifying the creation time.
pucReserved	void *	Empty pointer, used to extend the response body.

 Copyright © Huawei Technologies Co., Ltd.

Table 1-31 DISString parameter description

Parameter	Mandatory	Type	Description
stringLen	Yes	long	Length of the data to be uploaded.
data	Yes	Char *	Data to be uploaded.

Viewing Execution Results

Information similar to the following is displayed on the console:

```
the partition key is 1
the seqNumber is 0
the data is 10: aaaaaaaaaa
the partition key is 1
the seqNumber is 1
the data is 55: 6o96l4h0fG05Zm4sXUKEI1fyFHz68pYISa7AAdYsi373CHi2W5gl6S6
the partition key is 1
the seqNumber is 2
the data is 62: A72p5Cw04c6dy4M756Rmehm2DyKzx4Cas1OJSUliv52Q6x3lxxCi6z0Zs1b0la
```

1.9.11 Obtaining a Data Cursor

Use the DIS SDK to obtain the information about the data cursor.

```
char *streamName = "myStream";
char *projectId = "c159a24641da49b2a729ea6f57647888";
char *region = "******";
char *host = "dis.cn-north-1.myhuaweicloud.com:20004";
int ret = 0;
DISResponseInfo RspInfo = { 0 };
DISGetCursor     tempCursor = { 0 };
tempCursor.streamName = streamName;
tempCursor.partitionId = "0";
tempCursor.cursorType = DISCursorTypeAtSeqNumber;
tempCursor.startingSequenceNumber = "0";
//tempCursor.cursorType = DISCursorTypeAtSeqNumber;
//tempCursor.timestamp = "1536026725191";
printf("==================%s Begin=====================\n", __FUNCTION__);
ret = DisGetCursor(host, projectId, region, &tempCursor, &RspInfo);
if (ret != 0)
{
printf("GetCursor Error: %d\r\n", ret);
}
printf("the rsp code is %d\r\n", RspInfo.HttpResponseCode);
if (RspInfo.HttpResponseCode >= 300 || RspInfo.HttpResponseCode < 200)
{
printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
printf("the errorCode is %s, message is %s\r\n", RspInfo.ErrorCode, RspInfo.ErrorDetail);
printf("==================%s End=====================\n", __FUNCTION__);
}
else
{
printf("HttpResponseCode: %ld\r\n", RspInfo.HttpResponseCode);
printf("the partition cursor is %s\r\n", tempCursor.cursorResult.partitionCursor);
printf("==================%s End=====================\n", __FUNCTION__);
}
```

 Copyright © Huawei Technologies Co., Ltd.

Table 1-32 DISGetCursor parameter description

Parameter	Mandatory	Type	Description
streamName	Yes	char *	Name of the stream created on the management console. The name can contain letters, digits, hyphens (-), and underscores (_). The value must be 1 to 64 characters long.
partitionId	Yes	char *	ID of the partition to get the cursor for. Value range: 0 to 2147483647
cursorType	No	DISCursorType	Cursor type. • **DISCursorTypeAtSeqNumber**: Data is read from the position denoted by a specific sequence number (that is defined by **starting-sequence-number**). This is the default cursor type. • **DISCursorTypeAfterSeqNumber**: Data is read right after the position denoted by a specific sequence number (that is defined by **starting-sequence-number**). • **DISCursorTypeTrimHorizon**: Data is read at the last untrimmed record in the partition in the system, which is the oldest data record in the partition. • **DISCursorTypeLatest**: Data is read just after the most recent record in the partition. This setting ensures that you always read the most recent data in the partition. • **DISCursorTypeAtTimestamp**: Data is read from the position denoted by a specific timestamp (that is defined by **timestamp**).

 Copyright © Huawei Technologies Co., Ltd.

Parameter	Mandatory	Type	Description
startingSequenceNumber	No	char *	Sequence number. A sequence number is the unique identifier of each record. DIS automatically allocates a sequence number when the data producer calls the PutRecords operation to add data to the DIS stream. The sequence number of the same partition key usually changes with time. A longer interval between PutRecords requests results in a larger sequence number. The sequence number is closely related to cursor types **AT_SEQUENCE_NUMBER** and **AFTER_SEQUENCE_NUMBER**. The two parameters determine the location of the data to be accessed. Value range: 0 to 9223372036854775807
timestamp	No	char *	Timestamp when the data record starts to be read, which is closely related to the **DISCursorTypeAtTimestamp** cursor type. The two parameters determine the position of the data to be read. **NOTE** This timestamp is accurate to milliseconds.
pucReserved	No	void *	Empty pointer, used to extend the request body.

Viewing Execution Results

Information similar to the following is displayed on the console:

```
the rsp code is 200
HttpResponseCode: 200
the partition cursor is
eyJnZXRJdGVyYXRvclBhcmFtIjp7InN0cmVhbS1uYW1lIjoiZGlzLXNoYXduIiwicGFydGl0aW9uLWlkIjoiMCIsImN1c
nNvci10eXBlIjoiQVRfU0VRVUVOQ0VfTlVNQkVSIiwic3RhcnRpbmctc2VxdWVuY2UtbnVtYmVyIjoiMCJ9LCJnZW
5lcmF0ZVRpbWVzdGFtcCI6MTUzNTk3MjU4NzIxOX0
```

Table 1-33 DISGetCursor parameter description

Parameter	Type	Description
cursorResult	DISCursor	Cursor that is obtained in response.

 Copyright © Huawei Technologies Co., Ltd.

Table 1-34 DISCursor parameter description

Parameter	Type	Description
partitionCursor	Char [520]	Data cursor. Value: 1 to 512 characters. **NOTE** The validity period of a cursor is 5 minutes.

1.10 DIS Server-Side Error Codes

If an error occurs when you use the SDK, an error code is displayed on the DIS console.

HTTP Status Code	Error Code	Error Message	Description	Measure
441	DIS.4100	Authorization error.	The signature information generated using AK and SK is incorrect.	Ensure that the signature information in the request header is correct.
441	DIS.4101	Authorization header cannot be empty.	The signature information generated using AK and SK is blank.	Ensure that the signature information is generated.
441	DIS.4102	Incorrectly parsed authorization header.	The signature cannot be parsed.	Ensure that the signature information in the request header is correct.
441	DIS.4103	Empty X-Sdk-Date header.	The **X-Sdk-Date** field in the request header is blank.	Ensure that the **X-Sdk-Date** field in the request header is not blank.

Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
441	DIS.4104	Error parsing X-Sdk-Date header.	The **X-Sdk-Date** field in the request header cannot be parsed.	Ensure that the **X-Sdk-Date** field in the request header is correct.
441	DIS.4105	Invalid X-Sdk-Date header.	The **X-Sdk-Date** field in the request header is invalid.	Ensure that the **X-Sdk-Date** field in the request header is correct.
441	DIS.4106	Empty AcessKey header.	The **Authorization** field of the request header does not contain AK.	Ensure that AK is contained in the **Authorization** field.
441	DIS.4107	Invalid AcessKey header.	AK in the **Authorization** field of the request header is invalid.	Ensure that AK is valid and correct.
441	DIS.4108	Empty ServiceName header.	The **Authorization** field of the request header does not contain the service name.	Ensure that the **Authorization** field of the request header contain service name **dis**.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
441	DIS.4109	The Authorization header must contain the following field: {Credential,SignedHeaders,Signature;}	The **Authorization** field of the request header is incorrect.	Ensure that the **Authorization** field of the request header contains **Credential**, **SignedHeaders**, and **Signature**.
441	DIS.4110	Empty Signature header.	The **Authorization** field of the request header does not contain **SignedHeaders**.	Ensure that the signature generation mode is correct.
441	DIS.4111	Invalid Region header.	The region in the **Authorization** field of the request header is invalid.	Ensure that the region is valid.
441	DIS.4112	Invalid authorization request.	The signature information generated using AK and SK is incorrect.	Ensure that the signature generation mode and the information about AK, SK, and region are correct.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
441	DIS.4113	Empty Token header.	When token authentication is used, the **X-Auth-Token** field of the request header is blank.	Ensure that the **X-Auth-Token** field of the request header is not blank.
441	DIS.4114	Invalid Token header.	When token authentication is used, the **X-Auth-Token** field of the request header is invalid.	Ensure that the **X-Auth-Token** field of the request header is valid.
403	DIS.4116	Invalid RBAC.	User operations are restricted.	Ensure that real-name authentication have been performed, all bills have been paid, and DIS operating permissions have been obtained.
400	DIS.4117	Invalid Project Id.	The project ID input by the subscriber is invalid.	Ensure that the project ID is valid and correct.
400	DIS.4200	Invalid request.	The user request is invalid.	Check the request by referring to the API document.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4201	Invalid partition_id.	The partition ID input by the subscriber is invalid.	Ensure that the partition ID is valid.
400	DIS.4202	Empty request.	The user request is empty.	Input a valid request.
400	DIS.4203	Invalid monitoring period.	The start time for query the monitoring information is invalid.	Input a valid timestamp.
400	DIS.4204	The monitoring period cannot be longer than 7 days.	Only the monitoring information generated in the recent seven days can be queried.	Query the monitoring information generated in the recent seven days.
400	DIS.4205	Stream is not running.	The stream is not in the running state.	Check the stream status.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4208	Invalid MRS cluster.	The MRS cluster input during MRS dump task creation is invalid.	Ensure the MRS cluster name and ID are correct and the cluster is running in security mode.
400	DIS.4209	Invalid metrics label.	The monitoring metric input during monitoring information query is invalid.	Check and correct the monitoring metric by referring to the API document.
400	DIS.4215	Invalid cursor type.	The cursor type input during the data cursor acquisition is invalid.	Check and correct the cursor type by referring to the API document.
400	DIS.4216	Invalid sequence_number.	The sequence number input during data cursor acquisition is invalid.	Input a valid sequence number.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4217	Invalid partition cursor.	The partition cursor input during data download from DIS is invalid.	Obtain the partition cursor again and download the data.
400	DIS.4219	The file is constantly resent.	The file has been received.	Do not upload the file again.
400	DIS.4220	The block whose sequence number is %s needs to be resent.	The file block needs to be uploaded again.	Upload the corresponding file block as instructed.
400	DIS.4221	Block seq %s is expected.	Duplicate file blocks are input.	Upload the file block from the one expected by the system.
400	DIS.4222	Block seq %s is expected.	The input file blocks are not consecutive.	Upload the file block from the one expected by the system.
400	DIS.4223	The file size exceeds the limit.	The file capacity exceeds the upper limit.	Split the file and upload it again.
400	DIS.4224	The sequence number is out of range.	The sequence number input during data cursor acquisition is invalid.	Input a valid sequence number.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4225	Expired partition cursor.	The partition cursor input during data download from DIS expires.	Obtain the partition cursor again and download the data.
400	DIS.4226	A partition iterator error occurred or a record to which the SN corresponds has expired. Try to obtain the partition iterator again.	The sequence number of the partition cursor input during data acquisition expires.	Obtain the data cursor and use the new cursor to obtain data.
400	DIS.4300	Request error.	The request body error occurs.	Correct the request body by referring to the API document.
400	DIS.4301	The stream does not exist.	The stream does not exist.	Ensure that the stream exists.
400	DIS.4302	The partition does not exist.	The stream partition does not exist.	Ensure that the partition ID exists.
400	DIS.4303	The traffic control limit is exceeded.	The flow control limit is exceeded.	Add partitions or reduce the upload rate.

Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4305	Too many stream requests.	There is an excessive number of user requests at the same time.	Decrease the requesting frequency and try again.
400	DIS.4306	The bucket does not exist.	The OBS bucket does not exist.	Ensure that the OBS bucket exists.
400	DIS.4307	The stream already exists.	The stream already exists.	Change the name of the new stream.
400	DIS.4308	Insufficient quota.	The quotas of the streams or partitions are insufficient.	Release the quota or **submit a service ticket** to change the quota of the account.
400	DIS.4309	Too many request failures. Please try again later.	The IP address is added to the blacklist.	Ensure that the authentication information and request are valid and try again later.
400	DIS.4310	OBS access error.	OBS failed to be accessed.	Ensure that the user has permission to access OBS.
400	DIS.4319	Partition is expired.	The partition has expired.	The partition has expired. Check whether the partition has expired in the scaling-in scenario and use a correct and valid partition.

Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4329	app quota exceeded.	The application quota exceeds the limit.	Release the applications that are not used.
400	DIS.4330	app already exist.	An application with the same name already exists.	Change the name of the new application.
400	DIS.4331	app is using.	The application failed to be deleted.	Ensure that the application that you want to delete is not being used.
400	DIS.4332	app not found.	The specified application does not exist.	Ensure the application name is correct.
400	DIS.4335	Invalid IAM agency.	The IAM agency used during dump task creation is invalid.	Ensure that **dis_admin_agency** created by DIS or the user-defined IAM agency exists and permission is complete.
400	DIS.4336	Invalid HDFS path.	The MRS HDFS path input during MRS dump task creation is invalid.	Ensure that the MRS HDFS path exists.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4337	The DLI database does not exist.	The DLI database input during DLI dump task creation does not exist.	Ensure that the DLI database exists.
400	DIS.4338	The DLI table does not exist.	The DLI table input during DLI dump task creation does not exist.	Ensure that the DLI table exists and is an internal table.
400	DIS.4339	Consumer quota exceeded.	The consumer quota of the consumer group is insufficient.	The number of consumers in the consumer group has exceeded the maximum quota. Allocate consumers properly or create a consumer group to meet the requirement.
400	DIS.4341	The CloudTable cluster does not exist.	The CloudTable cluster input during CloudTable dump task creation does not exist.	Ensure that the CloudTable cluster exists and is running properly.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4342	The CloudTable table does not exist	The CloudTable table input during CloudTable dump task creation does not exist.	Ensure that the CloudTable table exists.
400	DIS.4343	The CloudTable table family does not exist.	The CloudTable column family input during CloudTable dump task creation does not exist.	Ensure that the CloudTable column family exists.
400	DIS.4345	Invalid CloudTable schema.	The schema input during CloudTable dump task creation is invalid.	Check the schema based on the returned details to ensure that the configured JSON attribute name exists and the parameters are valid.
400	DIS.4348	Invalid CloudTable openTSDB schema.	The schema input during CloudTable OpenTSDB dump task creation is invalid.	Check the schema based on the returned details to ensure that the JSON attribute name exists and the parameters are valid.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4350	Invalid DWS cluster.	The DWS cluster input during DWS dump task creation does not exist.	Ensure that the DWS cluster exists and is running properly.
400	DIS.4351	Invalid KMS userKey.	The KMS key input during DWS dump task creation is invalid.	Ensure that the KMS key exists.
400	DIS.4354	The transfer task does not exist.	The dump task to be deleted or updated does not exist.	Ensure that the dump task exists.
400	DIS.4355	The transfer task already exists.	When you create a dump task in a stream, another dump task with the same name already exists in this stream.	Change the name of the new dump task.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4357	Exceeded transfer task quota.	A maximum of five dump tasks can be created for one stream at the same time. Creating six dump tasks will exceed the quota limit.	Delete the discarded dump tasks and then add dump tasks again.
400	DIS.4358	The stream supports specific transfer tasks. Check the data type of the stream.	Common dump tasks cannot be created in the stream for small file dump.	Create a new stream and then dump tasks in the stream.
400	DIS.4360	Invalid data schema.	The data schema input during stream creation or update is invalid.	Ensure that the data schema format is correct and try again.
400	DIS.4601	The number of resource tags has reached the maximum.	A resource has a maximum of 10 tags. Adding 11 tags will exceed the quota limit.	Delete the discarded tags and then add tags again.

 Copyright © Huawei Technologies Co., Ltd.

HTTP Status Code	Error Code	Error Message	Description	Measure
400	DIS.4602	Invalid resource type.	The resource type is invalid.	Ensure that the resource type is valid.
400	DIS.4603	The resource does not exist.	The resource does not exist.	Ensure that the resource is not deleted.
400	DIS.4604	The key does not exist.	The tag key does not exist.	Ensure that the tag key exists.
400	DIS.4605	The action is not supported.	The current tag operation is not supported.	Ensure that the current tag operation is valid. Currently, only the create and delete operations are supported.
400	DIS.4375	The app does not commit checkpoint	The application does not submit the checkpoint operation in the stream.	Check whether the application has submitted the checkpoint operation in the consumption stream.
500	DIS.5000 to DIS.5999	System error. **NOTE** Contact technical support to handle system errors.	-	-

 Copyright © Huawei Technologies Co., Ltd.

2 Data Service SDK Reference

2.1 Overview

This document provides guidance for you to call APIs through App authentication.

When the App authentication is used, the SDK is required for access.

2.2 Calling APIs Through App Authentication

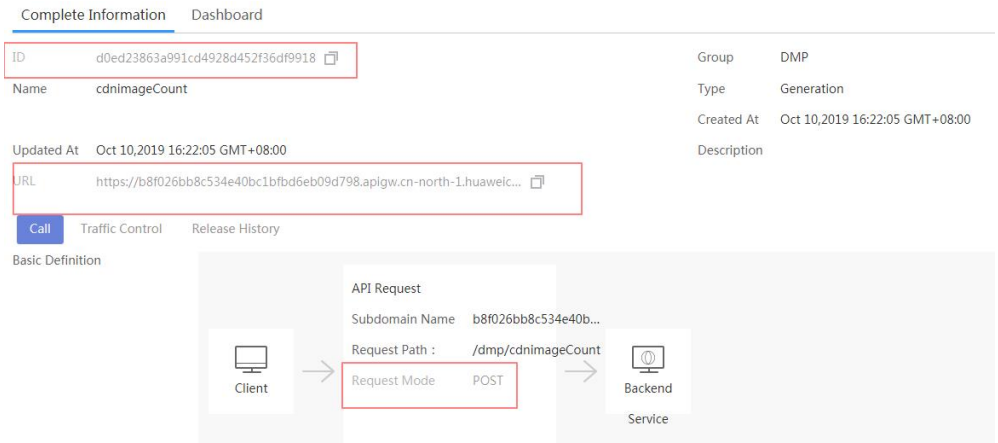
2.2.1 Preparation

Before accessing an API through an SDK in App authentication mode, you must collect the required information.

- Obtain the ID, domain name, request URL, and request method of the API.

  On the **Service Market** page of DLM, click an API name. On the **Complete Information** page, view the API ID, URL, and request method.

  Figure 2-1 Basic Information

- Provide a valid AppKey and AppSecret to generate an authentication signature.

 Copyright © Huawei Technologies Co., Ltd.

Create an app on the **Apps** page and bind it to the API. Then you can use the AppKey and AppSecert of the app to access the API. View the AppKey and AppSecret on the app details page.

Figure 2-2 Viewing AppKey and AppSecret



NOTE

- AppKey: access key ID of the app. It is the unique ID associated with a secret access key. The AppKey and AppSecret are together used to obtain an encrypted signature for a request.
- AppSecret: secret access key used together with an AppKey to sign requests. The AppKey and AppSecret can be together used to identify a request sender to prevent the request from being modified.

- When sending an API request, add the current time to the X-Sdk-Date header, and add the signature information to the Authorization header. The signature is valid only within a limited period of time.

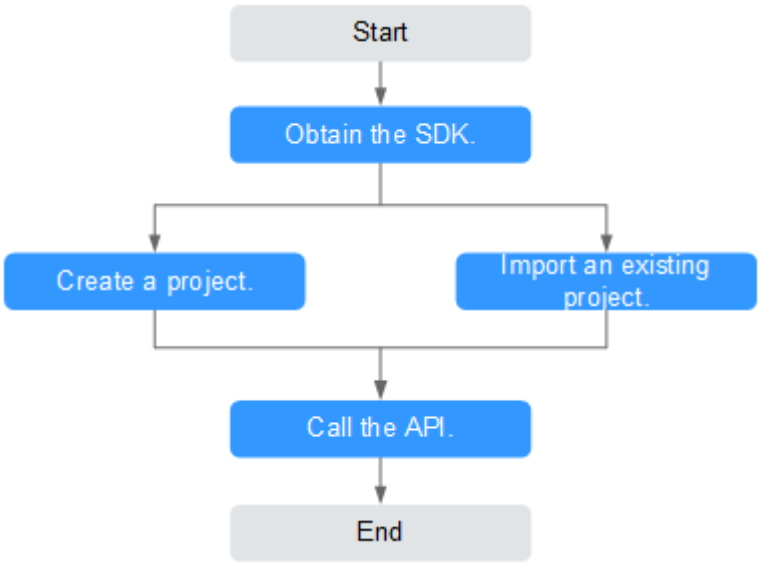
2.2.2 Java

Scenarios

To use Java to call an API through App authentication, obtain the Java SDK, create a project or import an existing project, and then call the API by referring to the API calling example.

This section uses Eclipse 4.5.2 as an example.

Figure 2-3 API calling process



Copyright © Huawei Technologies Co., Ltd.

Prerequisites

- You have obtained the domain name, ID, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed Eclipse 3.6.0 or a later version. If not, download Eclipse from the **official Eclipse website** and install it.
- You have installed Java Development Kit (JDK) 1.8.111 or a later version. If not, download JDK from the **official Oracle website** and install it.

Obtaining the SDK

Download SDKs. Obtain the ApiGateway-java-sdk.zip package. The following table shows the files decompressed from the package.

Name	Description
libs\	SDK dependencies
libs\java-sdk-core-*x.x.x*.jar	SDK package
src\com\apig\sdk\demo\Main.java	Sample code for signing requests
src\com\apig\sdk\demo\OkHttpDemo.java	
src\com\apig\sdk\demo\LargeFileUpload-Demo.java	
.classpath	Java project configuration files
.project	

To build a project with Maven, download the **java-sdk-core-*x.x.x*.jar** file in the SDK from **https://mirrors.huaweicloud.com/repository/maven/huaweicloudsdk/**.

The following are the Maven configuration items for adding the **java-sdk-core** dependency:

```
<dependency>
    <groupId>com.huawei.apigateway</groupId>
    <artifactId>java-sdk-core</artifactId>
    <version>3.0.10</version>
</dependency>
```

Importing a Project

Step 1 Open Eclipse and choose **File** > **Import**.

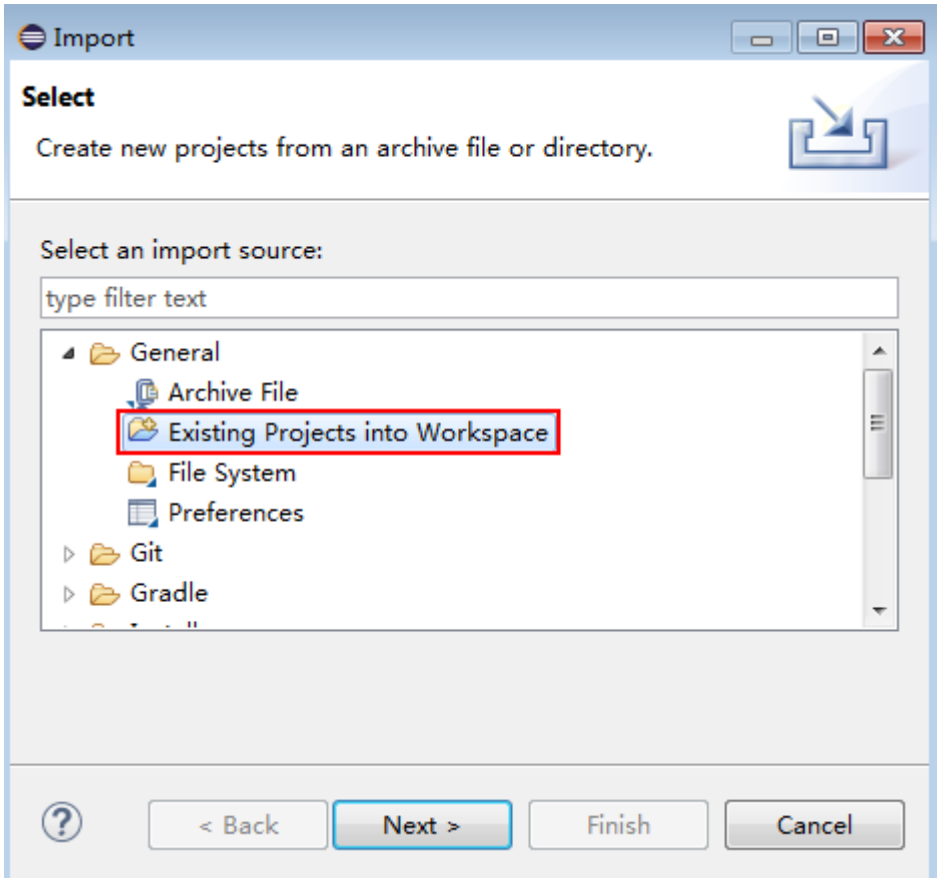
The **Import** dialog box is displayed.

Step 2 Choose **General** > **Existing Projects into Workspace** and click **Next**.

The **Import Projects** dialog box is displayed.

 Copyright © Huawei Technologies Co., Ltd.

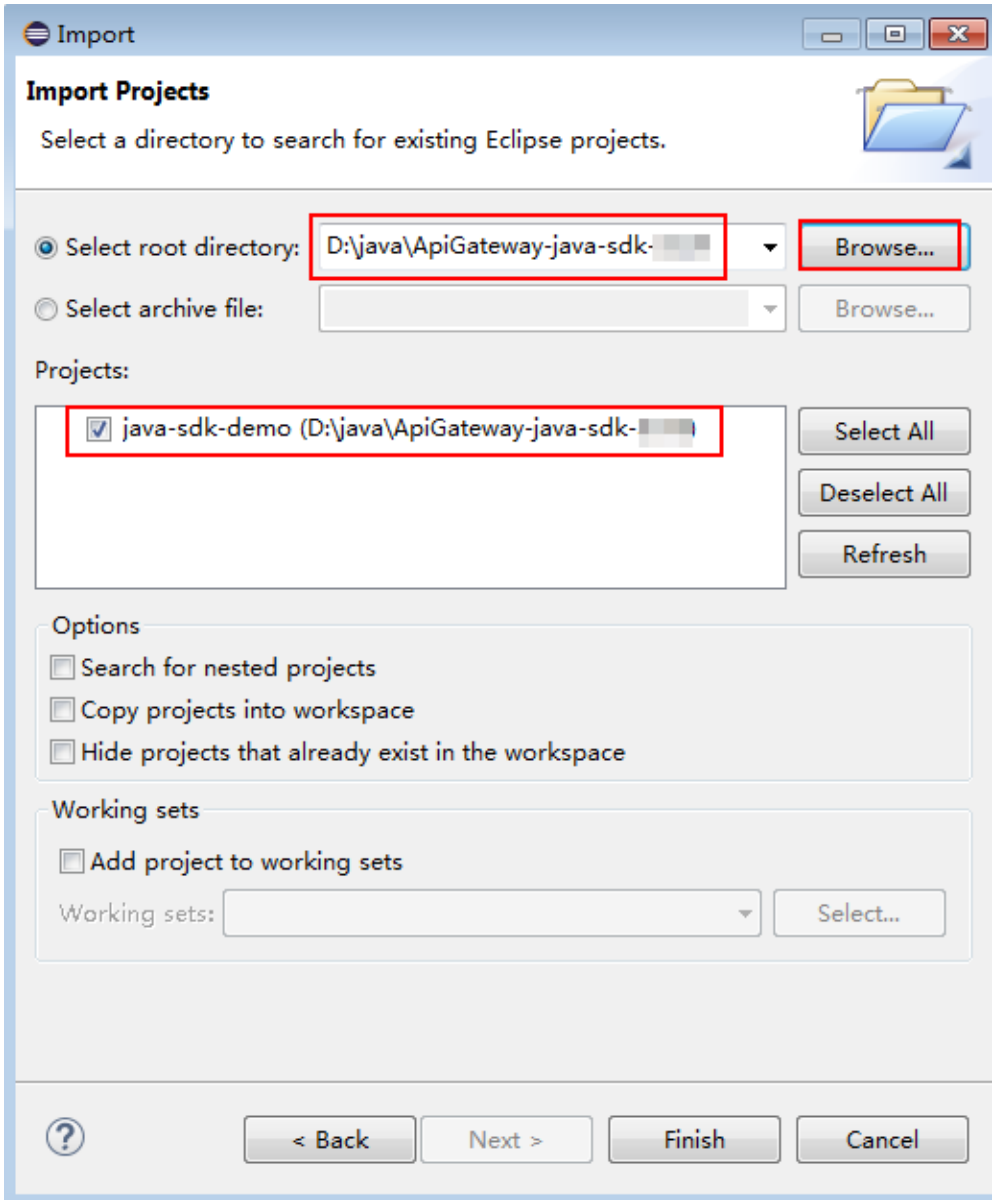
Figure 2-4 Importing a project



Step 3 Click **Browse** and select the directory where the SDK is decompressed.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-5 Selecting the demo project

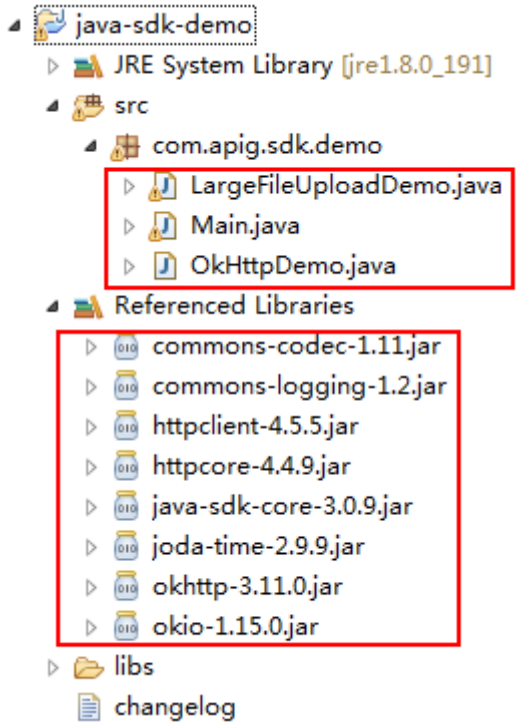


Step 4 Click **Finish**.

The following figure shows the directory structure of the project.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-6 Directory structure of the imported project



Modify the parameters in sample code **Main.java** as required. For details about the sample code, see **API Calling Example**.

----End

Creating a Project

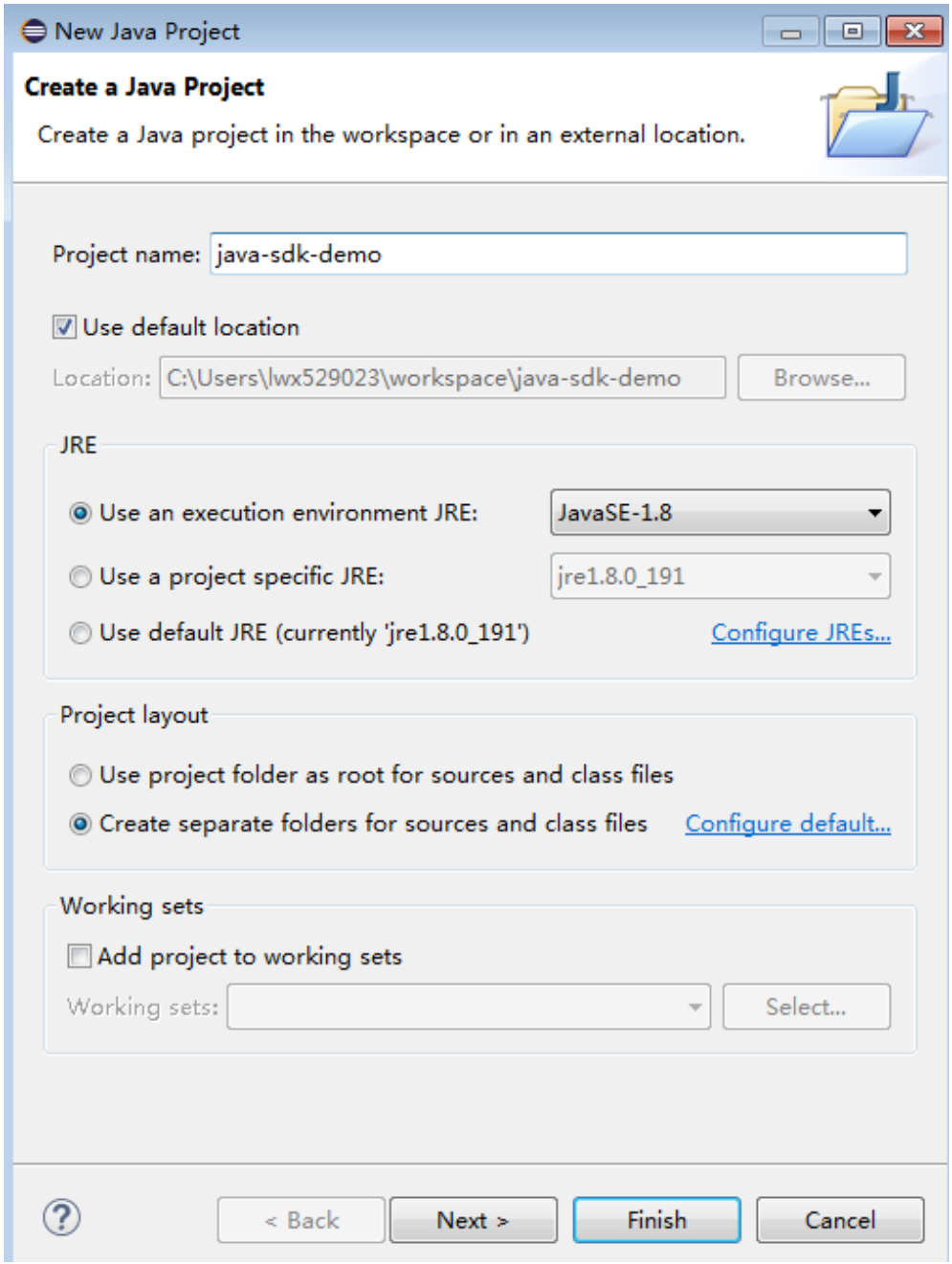
Step 1 Open Eclipse and choose **File** > **New** > **Java Project**.

The **New Java Project** dialog box is displayed.

Step 2 Enter a project name, for example, **java-sdk-demo**, retain the default settings for other parameters, and click **Finish**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-7 Creating a project

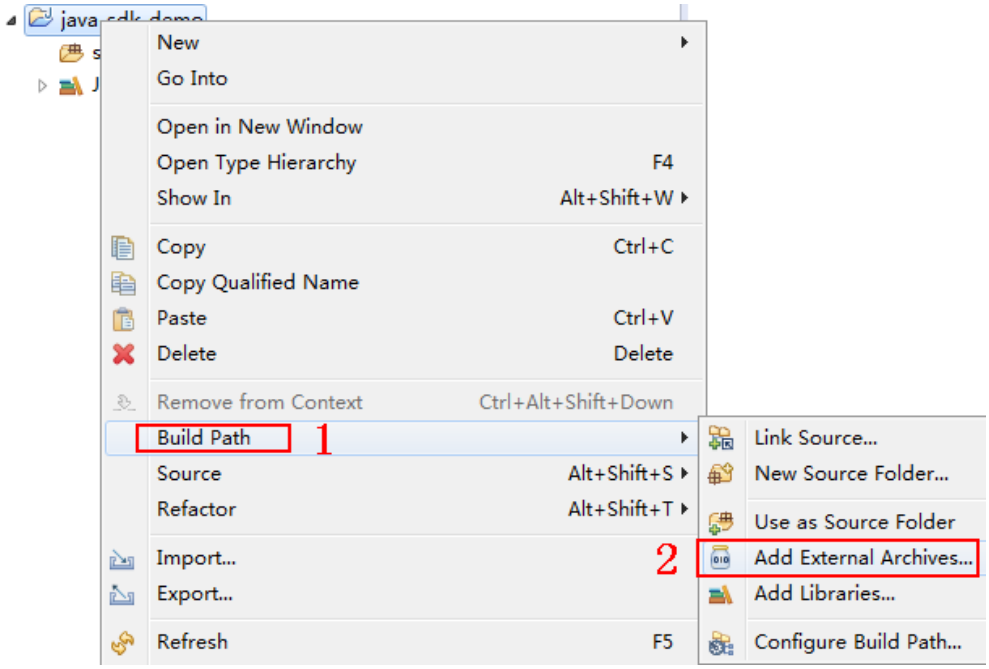


Step 3 Import the .jar files in the API Gateway Java SDK.

1. Choose **java-sdk-demo**, right-click, and choose **Build Path** > **Add External Archives** from the shortcut menu.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-8 Importing the .jar files



2. Select all .jar files in the **\libs** directory.

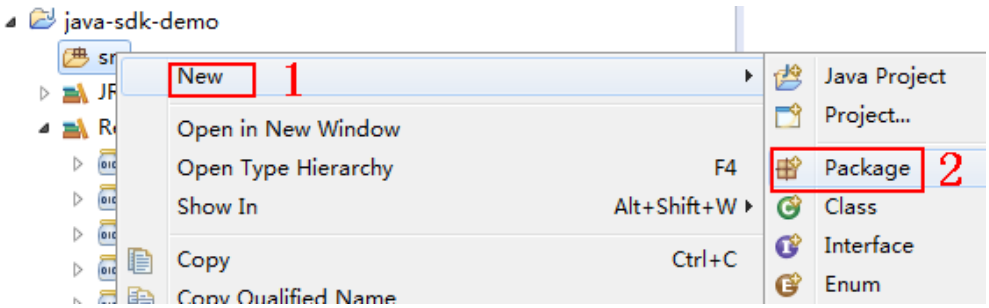
Figure 2-9 Selecting all .jar files



Step 4 Create a package and **Main** file.

1. Choose **src**, right-click, and choose **New** > **Package** from the shortcut menu.

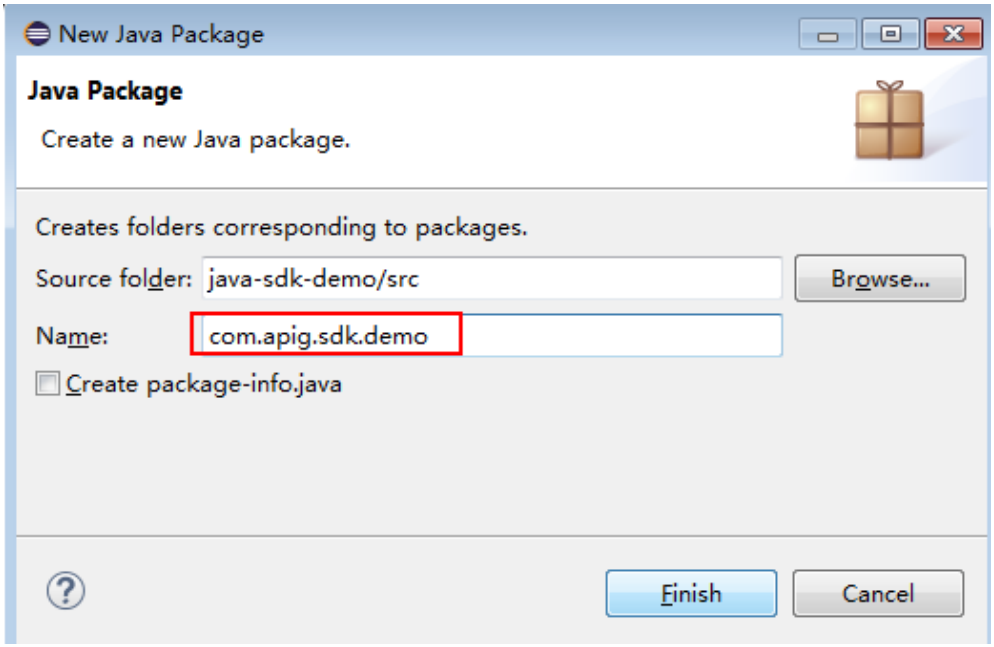
Figure 2-10 Creating a package



2. Enter **com.apig.sdk.demo** for **Name**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-11 Setting a package name

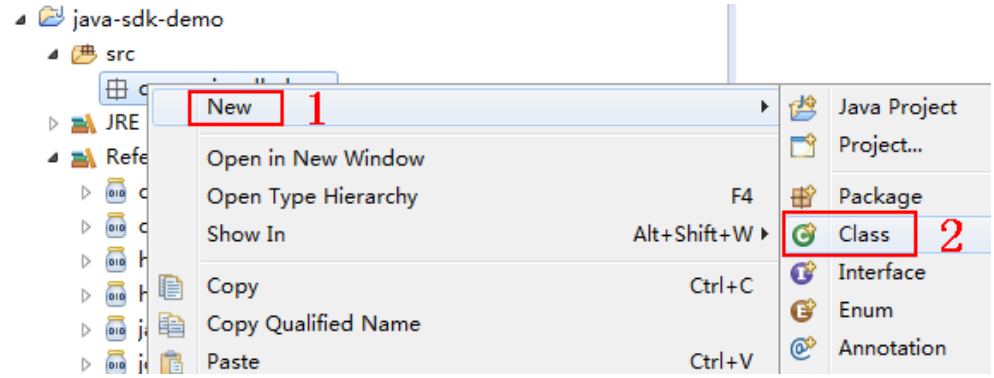


3. Click **Finish**.

   The package is created.

4. Choose **com.apig.sdk.demo**, right-click, and choose **New** > **Class** from the shortcut menu.

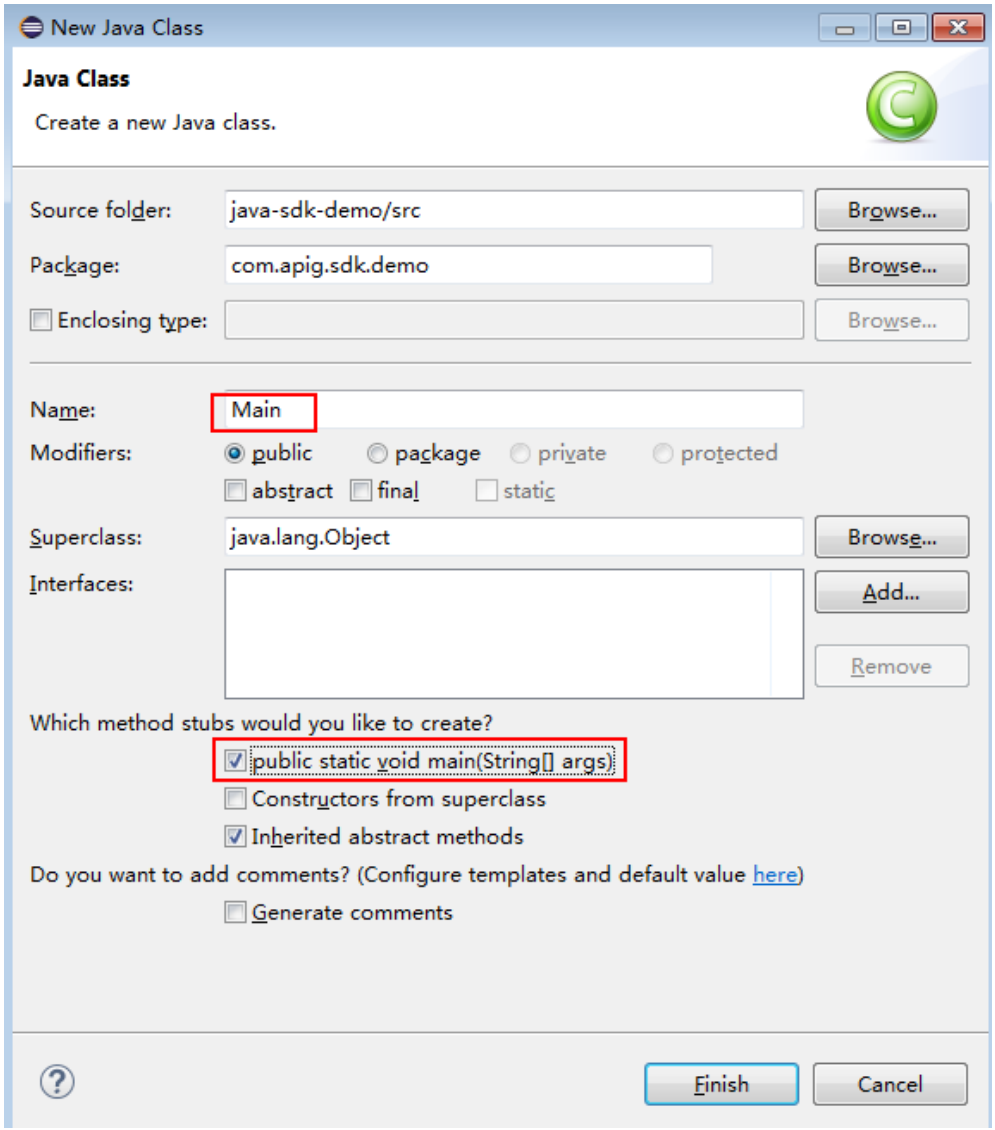
Figure 2-12 Creating a class



5. Enter **Main** for **Name** and select **public static void main(String[] args)**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-13 Configuring the class

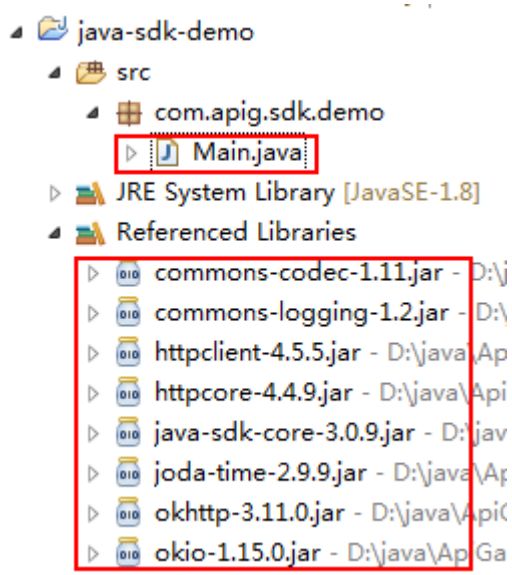


6. Click **Finish**.
   The **Main** file is created.

Step 5 View the directory structure of the project.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-14 Directory structure of the new project



Before using **Main.java**, enter the required code according to **API Calling Example**.

----End

API Calling Example

NOTE

- This section demonstrates how to access a published API.
- You need to create and release an API on the DLM management console. For details about how to create and release an API, see *DGC User Guide*.
- The backend of this API is a fake HTTP service, which returns response code **200** and message body **Congratulations, sdk demo is running**.

Step 1 Add the following references to **Main.java**:

```
import java.io.IOException;
import javax.net.ssl.SSLContext;

import org.apache.http.Header;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpRequestBase;
import org.apache.http.conn.ssl.AllowAllHostnameVerifier;
import org.apache.http.conn.ssl.SSLConnectionSocketFactory;
import org.apache.http.conn.ssl.SSLContexts;
import org.apache.http.conn.ssl.TrustSelfSignedStrategy;
import org.apache.http.impl.client.CloseableHttpClient;
import org.apache.http.impl.client.HttpClients;
import org.apache.http.util.EntityUtils;

import com.cloud.apigateway.sdk.utils.Client;
import com.cloud.apigateway.sdk.utils.Request;
```

Step 2 Construct a request by configuring the following parameters:

- **AppKey**: Obtain it by referring to **Preparation**. The sample code uses **4f5f626b-073f-402f-a1e0-e52171c6100c**.

 Copyright © Huawei Technologies Co., Ltd.

- **AppSecret**: Obtain it by referring to **Preparation**. Set this parameter based on the site requirements. The example code uses ****** as an example.
- **Method**: Specify a request method. The sample code uses **POST**.
- **url**: Request URL of the API, excluding the QueryString and fragment parts. For the domain name, use your own independent domain name bound to the group to which the API belongs. The sample code uses http://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-north-1.huaweicloud.com/java-sdkhttp://serviceEndpoint/java-sdk.
- **queryString**: Specify query parameters to be carried in the URL. Characters (0-9a-zA-Z./;[]\-=~#%^&_+:'') are allowed. The sample code uses **name=value**.
- Header: Request header. Set a request header as required. It cannot contain underscores (_). The sample code uses **Content-Type:text/plain**. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.
- **body**: Specify the request body. The sample code uses **demo**.

The sample code is as follows:

```
    Request request = new Request();
    try
    {
        request.setKey("4f5f626b-073f-402f-a1e0-e52171c6100c"); //Obtain the value when creating an app.
        request.setSecret("*****"); //Obtained when an app is created.
        request.setMethod("POST");
        request.setUrl("http://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-north-1.huaweicloud.comserviceEndpoint/java-sdk");
         //Obtain the URL when creating an API group.
        request.addQueryStringParam("name", "value");
        request.addHeader("x-api-id","**********");
        request.addHeader("Content-Type", "text/plain");
        //request.addHeader("x-stage", "publish_env_name"); //Specify an environment name before publishing the API in a non-RELEASE environment.
        request.setBody("demo");
    } catch (Exception e)
    {
        e.printStackTrace();
        return;
    }
```

Step 3 Sign the request, add header **x-Authorization**, access the API, and print the result.

The sample code is as follows:

```
    CloseableHttpClient client = null;
    try
    {
        HttpRequestBase signedRequest = Client.sign(request);
        Header[] authorization = signedRequest.getHeaders("Authorization");
        signedRequest.addHeader("x-Authorization",authorization[0].getValue());

        client = HttpClients.custom().build();
        HttpResponse response = client.execute(signedRequest);
        System.out.println(response.getStatusLine().toString());
        Header[] resHeaders = response.getAllHeaders();
        for (Header h : resHeaders)
        {
            System.out.println(h.getName() + ":" + h.getValue());
        }
        HttpEntity resEntity = response.getEntity();
        if (resEntity != null)
        {
            System.out.println(System.getProperty("line.separator") + EntityUtils.toString(resEntity, "UTF-8"));
```

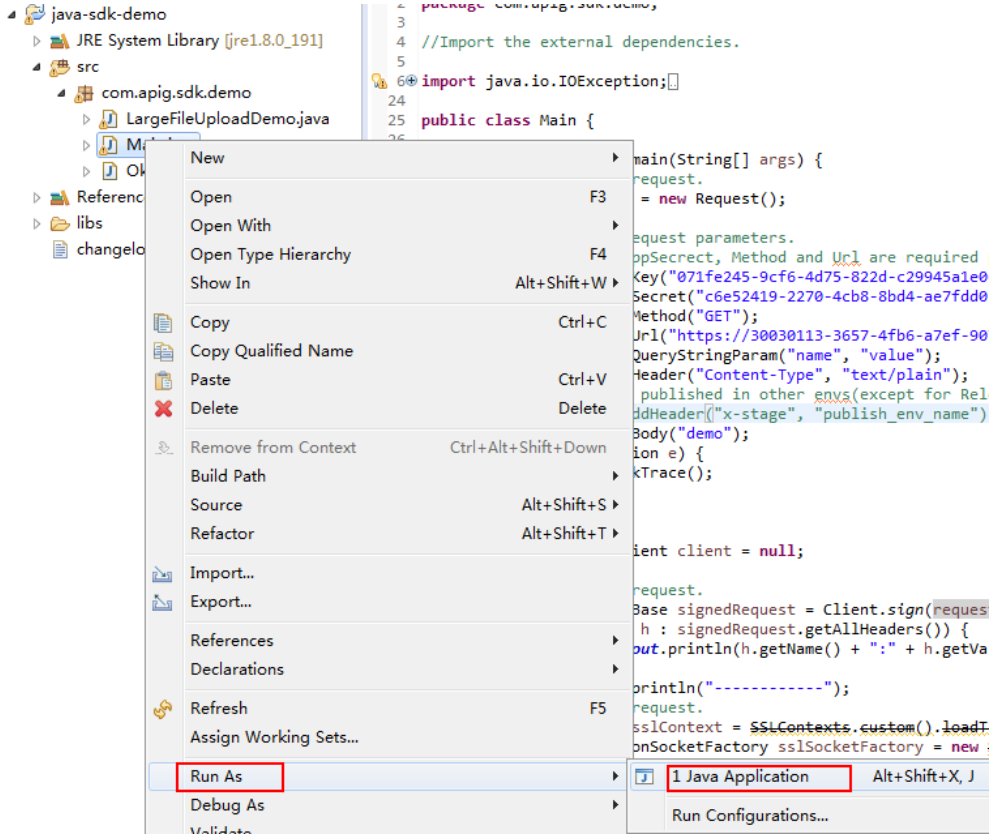
 Copyright © Huawei Technologies Co., Ltd.

```
        }

    } catch (Exception e)
    {
        e.printStackTrace();
    } finally
    {
        try
        {
            if (client != null)
            {
                client.close();
            }
        } catch (IOException e)
        {
            e.printStackTrace();
        }
    }
```

Step 4 Choose **Main.java**, right-click, and choose **Run As** > **Java Application** to run the project test code.

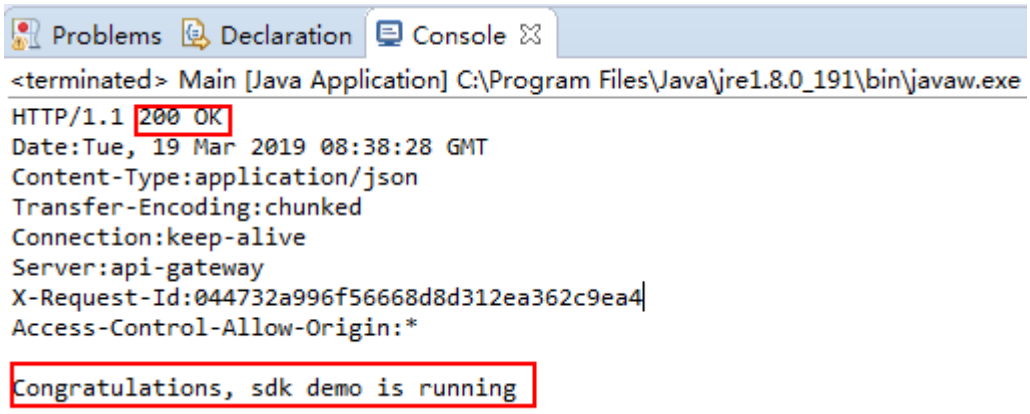
Figure 2-15 Running the project test code



Step 5 On the **Console** tab page, view the running result.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-16 Response displayed if the calling is successful



----End

2.2.3 Go

Scenarios

To use Go to call an API through App authentication, obtain the Go SDK, create a new project, and then call the API by referring to the API calling example.

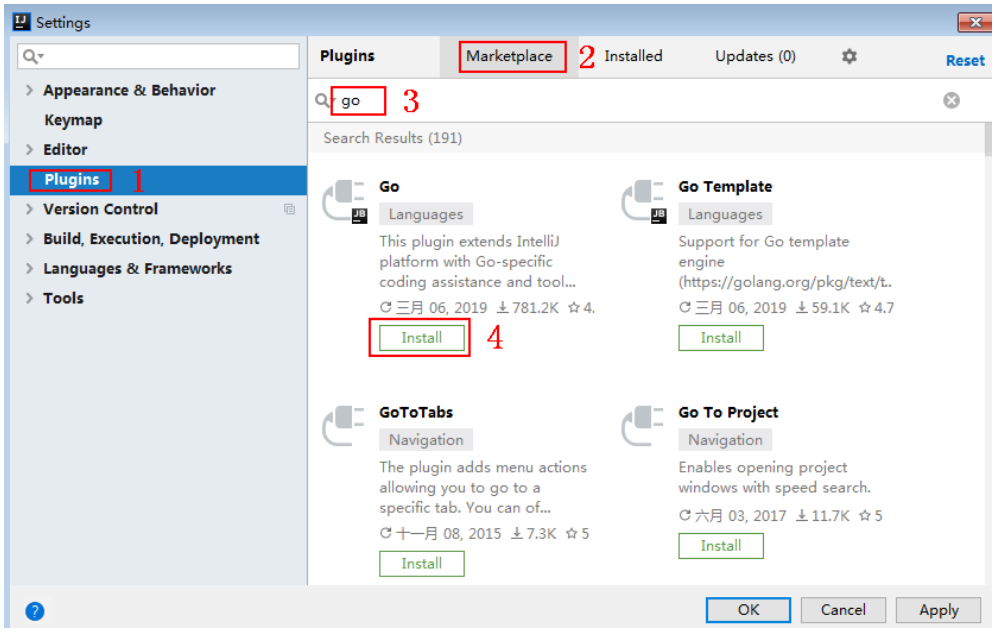
This section uses IntelliJ IDEA 2018.3.5 as an example.

Prerequisites

- You have obtained the domain name, ID, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed the Go programming language. If not, download the Go installation package from the **official Go website** and install it.
- You have installed IntelliJ IDEA. If not, download IntelliJ IDEA from the **official IntelliJ IDEA website** and install it.
- You have installed the Go plug-in on IntelliJ IDEA. If not, install the Go plug-in according to **Figure 2-17**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-17 Installing the Go plug-in



Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-go-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-go-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
core\escape.go	SDK code
core\signer.go	
demo.go	Sample code

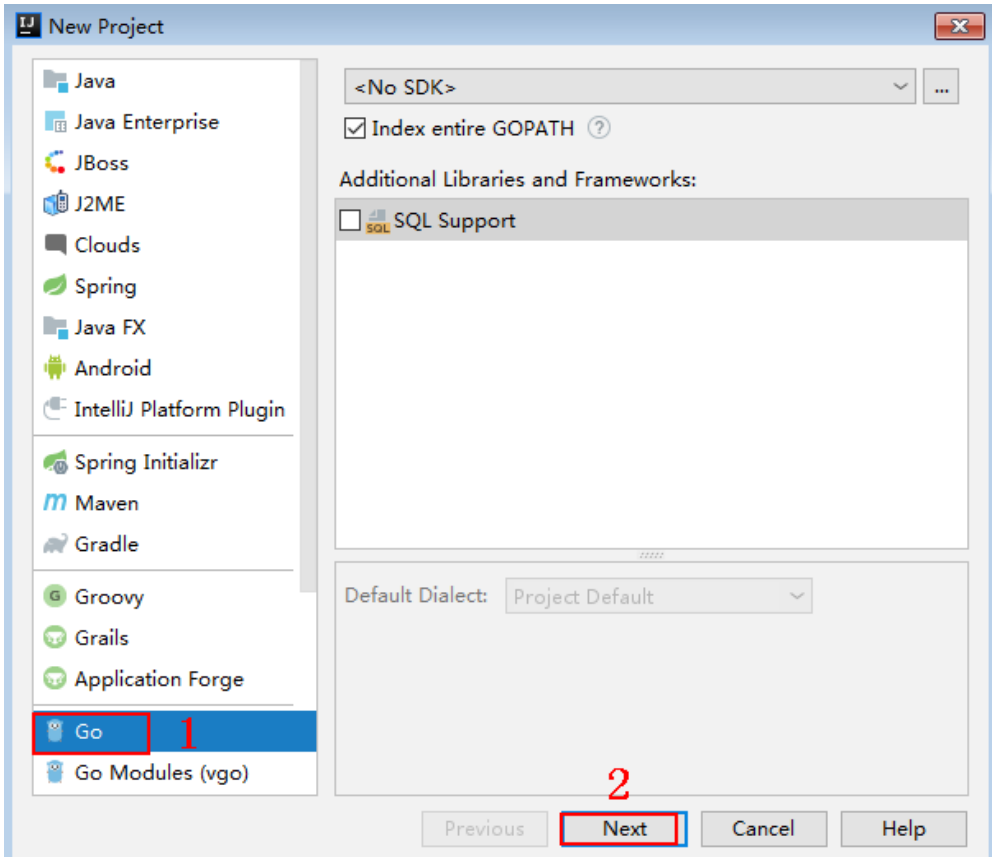
Creating a Project

Step 1 Start IntelliJ IDEA and choose **File** > **New** > **Project**.

On the displayed **New Project** page, choose **Go** and click **Next**.

 Copyright © Huawei Technologies Co., Ltd.

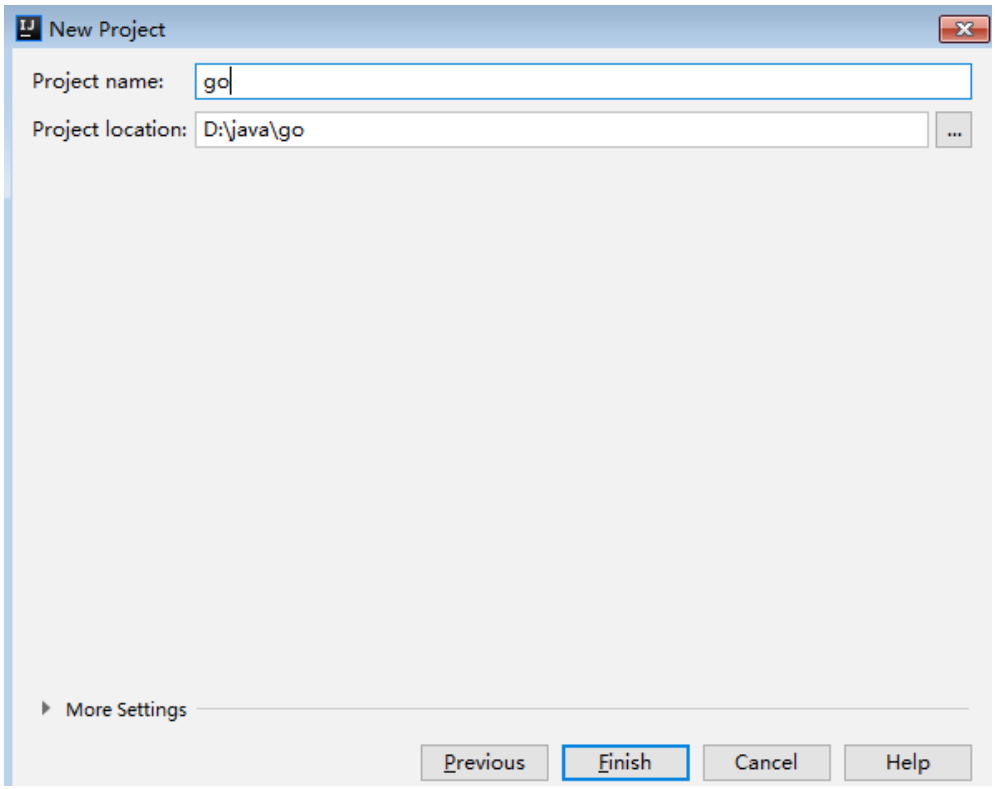
Figure 2-18 New Project



Step 2 Click **...**, select the directory where the SDK is decompressed, and click **Finish**.

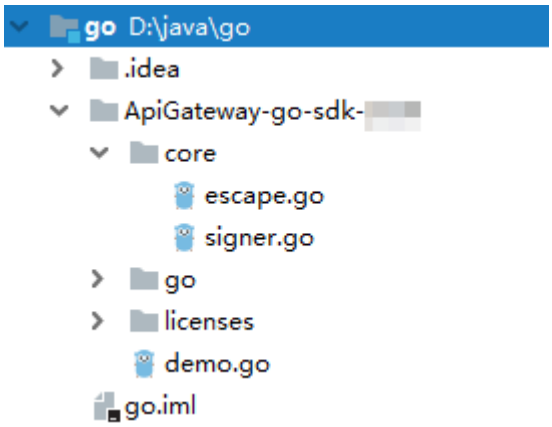
 Copyright © Huawei Technologies Co., Ltd.

Figure 2-19 Selecting the SDK directory after decompression



Step 3 View the directory structure shown in the following figure.

Figure 2-20 Directory structure of the new project



Modify the parameters in sample code **demo.go** as required. For details about the sample code, see **API Calling Example**.

----End

API Calling Example

Step 1 Import the Go SDK (signer.go) to the project.

```
import "apig-sdk/go/core"
```

 Copyright © Huawei Technologies Co., Ltd.

Step 2 Generate a new signer and enter the AppKey and AppSecret.

```
s := core.Signer{
        Key: "4f5f626b-073f-402f-a1e0-e52171c6100c",
        Secret: "******",
}
```

Step 3 Generate a new request, and specify the domain name, method, request URL, query parameters, and body.

```
r, _ := http.NewRequest("POST", "http://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-north-1.huaweicloud.comserviceEndpoint/api?a=1&b=2",
                        ioutil.NopCloser(bytes.NewBuffer([]byte("foo=bar"))))
```

Step 4 Add a header to the request. The header contains specific parameters. Add other headers to be signed as necessary. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
r.Header.Add("x-stage", "RELEASE")
r.Header.Add("x-api-id","********")
```

Step 5 Execute the following function to add the X-Sdk-Date and Authorization headers for signing: Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
s.Sign(r)
authorization := r.Header.Get("Authorization")
r.Header.Add("x-Authorization", authorization)
```

Step 6 Access the API and view the access result.

```
resp, err := http.DefaultClient.Do(r)
body, err := ioutil.ReadAll(resp.Body)
```

----End

2.2.4 Python

Scenarios

To use Python to call an API through App authentication, obtain the Python SDK, create a new project, and then call the API by referring to the API calling example.

This section uses IntelliJ IDEA 2018.3.5 as an example.

Preparing the Environment

- You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed Python 2.7.9 or 3.X. If not, download the Python installation package from the **official Python website** and install it.

  After Python is installed, run the **pip** command to install the **requests** library.

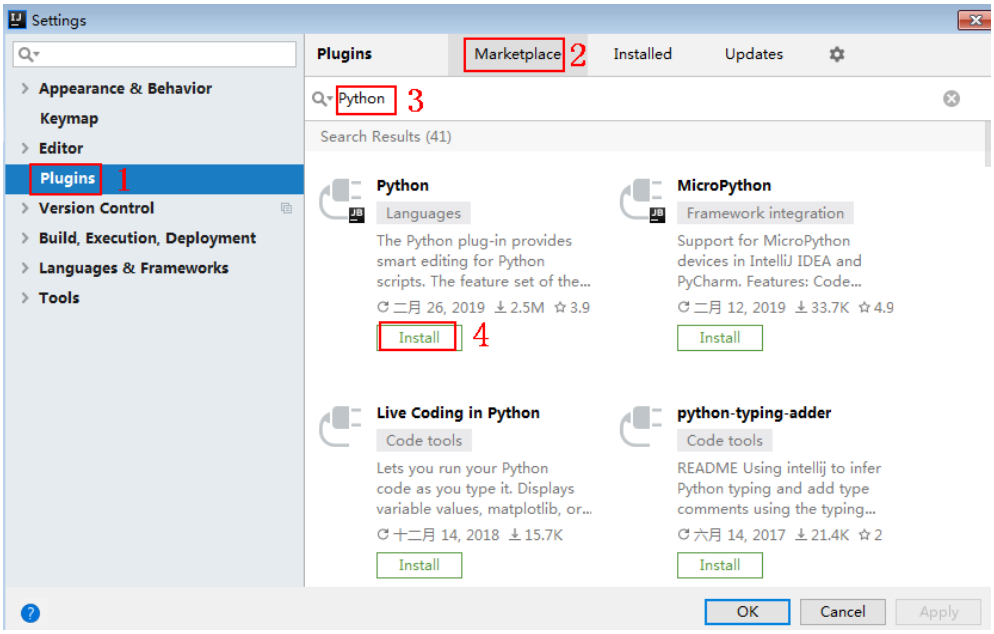
  ```
  pip install requests
  ```

  NOTE

  If a certificate error occurs during the installation, download the **get-pip.py** file to upgrade the pip environment, and try again.
- You have installed IntelliJ IDEA. If not, download IntelliJ IDEA from the **official IntelliJ IDEA website** and install it.
- You have installed the Python plug-in on IntelliJ IDEA. If not, install the Python plug-in according to **Figure 2-21**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 2-21 Installing the Python plug-in



Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-python-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-python-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
apig_sdk__init__.py	SDK code
apig_sdk\signer.py	
main.py	Sample code
backend_signature.py	Sample code for backend signing
licenses\license-requests	Third-party license

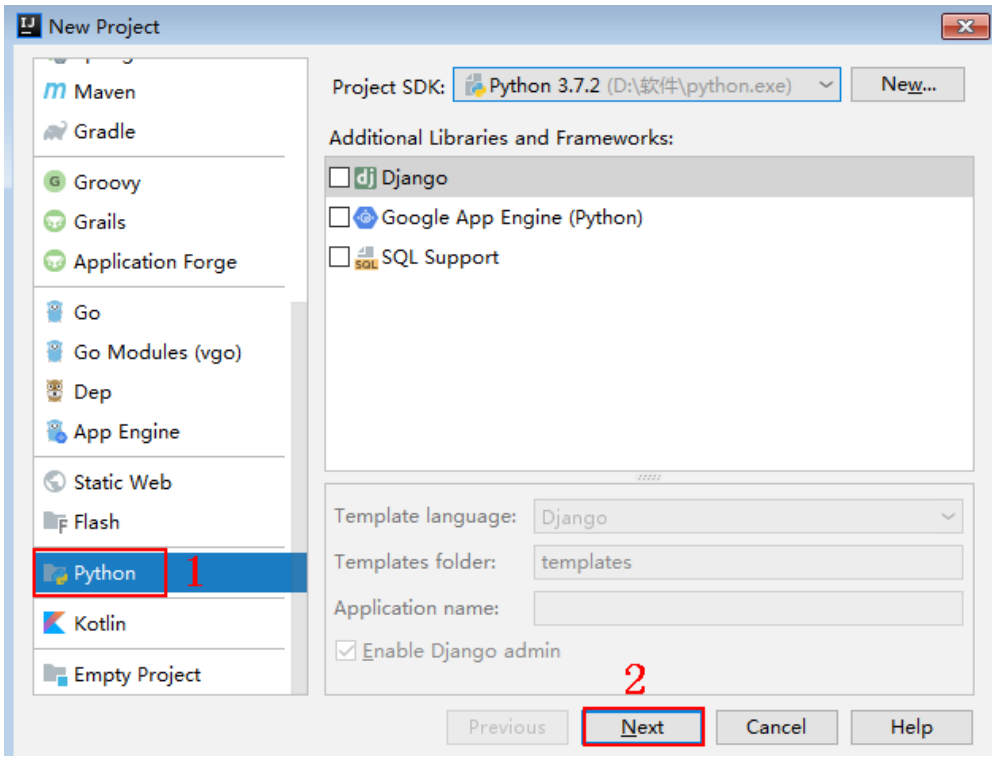
Creating a Project

Step 1 Start IDEA and choose **File** > **New** > **Project**.

On the displayed **New Project** page, choose **Python** and click **Next**.

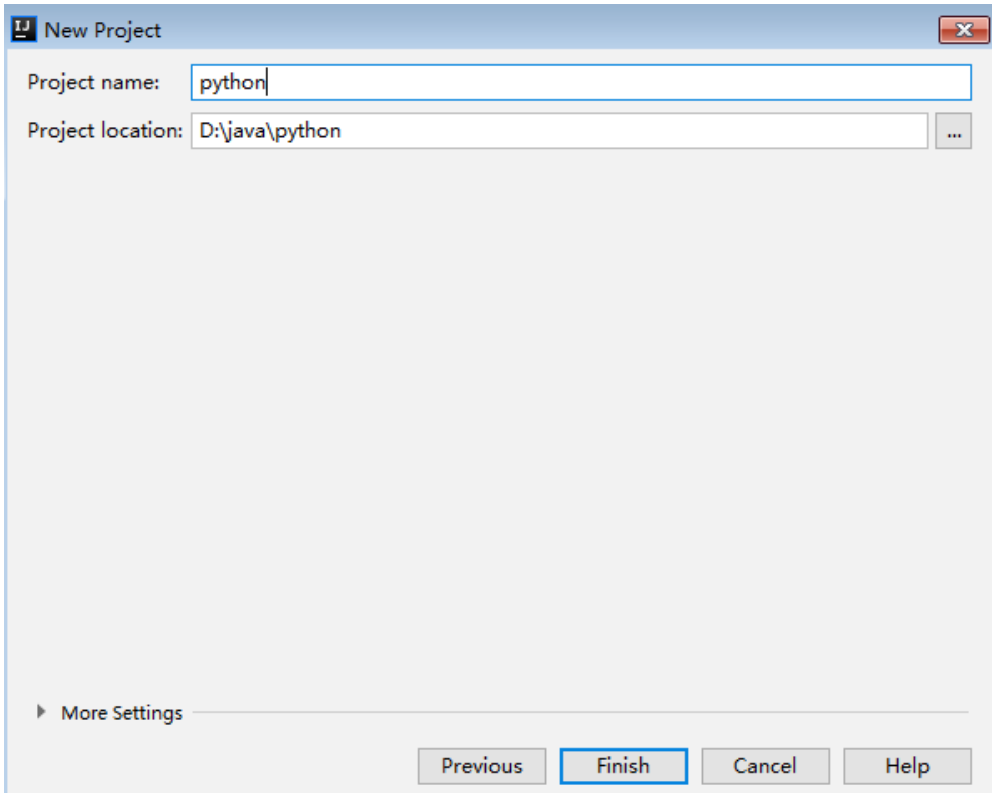
 Copyright © Huawei Technologies Co., Ltd.

Figure 2-22 New Project



Step 2 Click **Next**. Click **...**, select the directory where the SDK is decompressed, and click **Finish**.

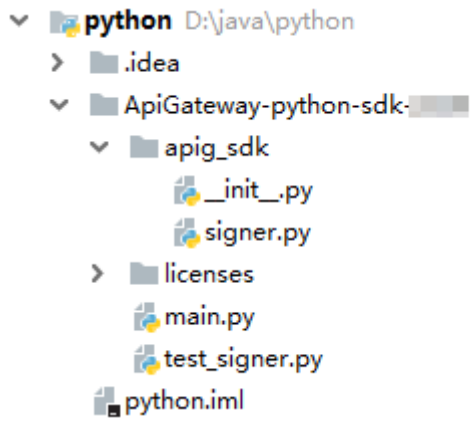
Figure 2-23 Selecting the SDK directory after decompression



 Copyright © Huawei Technologies Co., Ltd.

Step 3 View the directory structure shown in the following figure.

Figure 2-24 Directory structure of the new project



Modify the parameters in sample code **main.py** as required. For details about the sample code, see **API Calling Example**.

----End

API Calling Example

Step 1 Import **apig_sdk** to the project.

```
from apig_sdk import signer
import requests
```

Step 2 Generate a new signer and enter the AppKey and AppSecret.

```
sig = signer.Signer()
sig.Key = "4f5f626b-073f-402f-a1e0-e52171c6100c"
sig.Secret = "******"
```

Step 3 Generate a request, and specify the method, request URI, header, and request body. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
r = signer.HttpRequest("POST",
                       "https://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-
north-1.huaweicloud.comserviceEndpoint/app1?a=1",
                       {"x-stage": "RELEASE", "x-api-id": "**********"},
                       "body")
```

Step 4 Execute the following function to add the X-Sdk-Date and Authorization headers for signing: Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
sig.Sign(r)
r.headers["x-Authorization"] = r.headers["Authorization"]
```

Step 5 Access the API and view the access result.

```
resp = requests.request(r.method, r.scheme + "://" + r.host + r.uri, headers=r.headers, data=r.body)
print(resp.status_code, resp.reason)
print(resp.content)
```

----End

 Copyright © Huawei Technologies Co., Ltd.

2.2.5 C#

Scenarios

To use C# to call an API through App authentication, obtain the C# SDK, open the project file in the SDK, and then call the API by referring to the API calling example.

Preparing the Environment

- You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed Visual Studio. If not, download it from the **official Visual Studio website** and install it.

Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-csharp-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-csharp-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
apigateway-signature\Signer.cs	SDK code
apigateway-signature\HttpEncoder.cs	
sdk-request\Program.cs	Sample code for signing requests
backend-signature\	Sample project for backend signing
csharp.sln	Project file
licenses\license-referencesource	Third-party license

Opening a Project

Double-click **csharp.sln** in the SDK package to open the project. The project contains the following:

- **apigateway-signature**: Shared library that implements the signature algorithm. It can be used in the .Net Framework and .Net Core projects.
- **backend-signature**: Example of a backend service signature.
- **sdk-request**: Example of invoking the signature algorithm. Modify the parameters as required. For details about the sample code, see **API Calling Example**.

 Copyright © Huawei Technologies Co., Ltd.

API Calling Example

Step 1 Import the C# SDK to your project.

```
using APIGATEWAY_SDK;
```

Step 2 Generate a new signer and enter the AppKey and AppSecret.

```
Signer signer = new Signer();
signer.Key = "4f5f626b-073f-402f-a1e0-e52171c6100c";
signer.Secret = "******";
```

Step 3 Generate an HttpRequest, and specify the method, request URL, and body.

```
HttpRequest r = new HttpRequest("POST",
                    new Uri("https://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-
north-1.huaweicloud.comserviceEndpoint/app1?query=value"));
r.body = "{\"a\":1}";
```

Step 4 Add a header to the request. The header contains specific parameters. Add other headers to be signed as necessary. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
r.headers.Add("x-stage", "RELEASE");
r.headers.Add("x-api-id","*******");
```

Step 5 Execute the following function to generate **HttpWebRequest**, and add the X-Sdk-Date and Authorization headers for signing the request: Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
HttpWebRequest req = signer.Sign(r);
req.Headers.Add("x-Authorization", req.Headers.GetValues("authorization")[0]);
```

Step 6 Access the API and view the access result.

```
var writer = new StreamWriter(req.GetRequestStream());
writer.Write(r.body);
writer.Flush();
HttpWebResponse resp = (HttpWebResponse)req.GetResponse();
var reader = newStreamReader(resp.GetResponseStream());
Console.WriteLine(reader.ReadToEnd());
```

----End

2.2.6 JavaScript

Scenarios

To use JavaScript to call an API through App authentication, obtain the JavaScript SDK, create a new project, and then call the API by referring to the API calling example.

This section uses IntelliJ IDEA 2018.3.5 as an example to describe how to set up a Node.js development environment.

Preparing the Environment

- You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed the Node.js programming language. If not, download the Node.js installation package from the **official Node.js website** and install it.

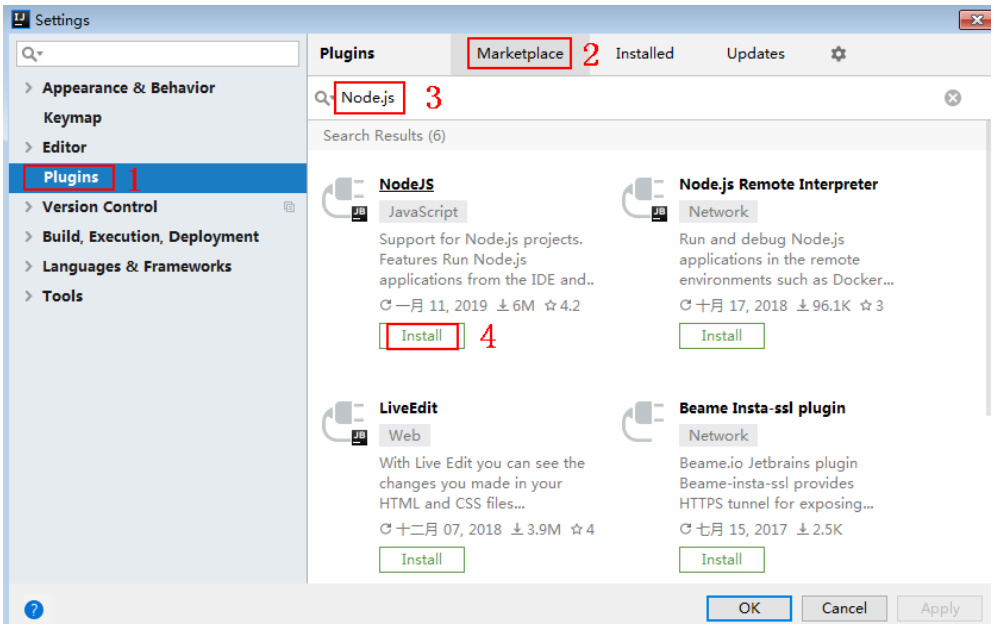
 Copyright © Huawei Technologies Co., Ltd.

After Node.js is installed, run the **npm** command to install the **moment** and **moment-timezone** modules.

```
npm install moment --save
npm install moment-timezone --save
```

- You have installed IntelliJ IDEA. If not, download IntelliJ IDEA from the **official IntelliJ IDEA website** and install it.
- You have installed the Node.js plug-in on IntelliJ IDEA. If not, install the Python plug-in according to **Figure 2-25**.

Figure 2-25 Installing the Node.js plug-in



Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-javascript-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-javascript-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
signer.js	SDK code
node_demo.js	Node.js sample code
demo.html	Browser sample code
demo_require.html	Browser sample code (loaded using **require**)
test.js	Test case

 Copyright © Huawei Technologies Co., Ltd.

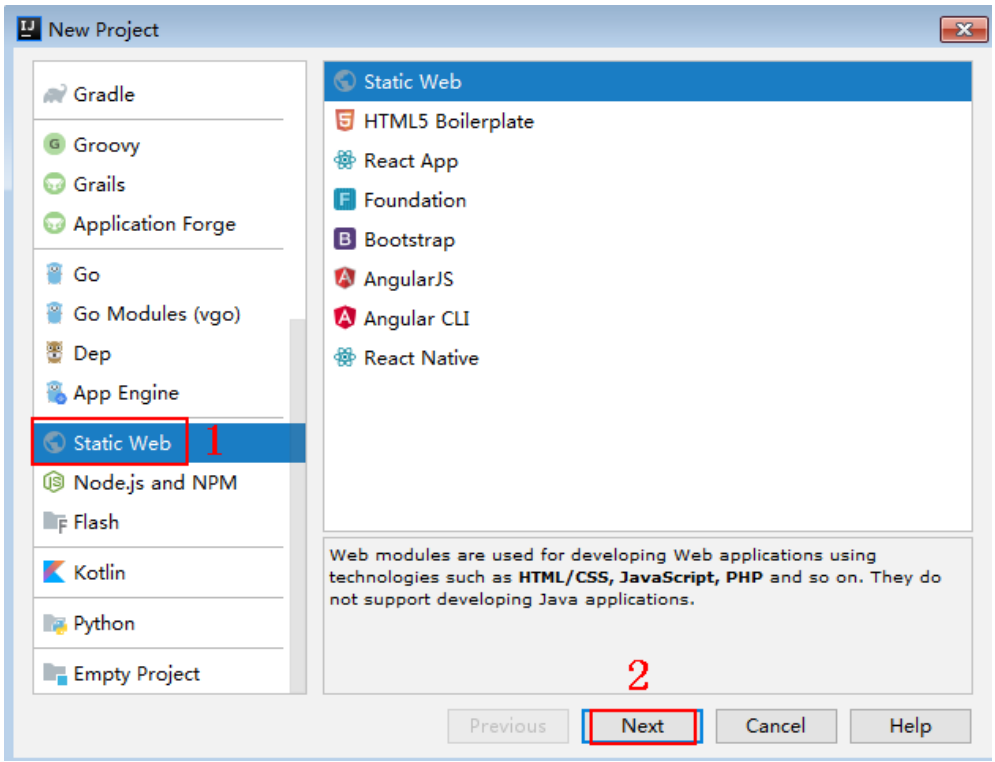
Name	Description
js\hmac-sha256.js	Dependencies
js\moment.min.js	
js\moment-timezone-with-data.min.js	
licenses\license-crypto-js	Third-party licenses
licenses\license-moment	
licenses\license-moment-timezone	
licenses\license-node	

Creating a Project

Step 1 Start IntelliJ IDEA and choose **File** > **New** > **Project**.

In the **New Project** dialog box, choose **Static Web** and click **Next**.

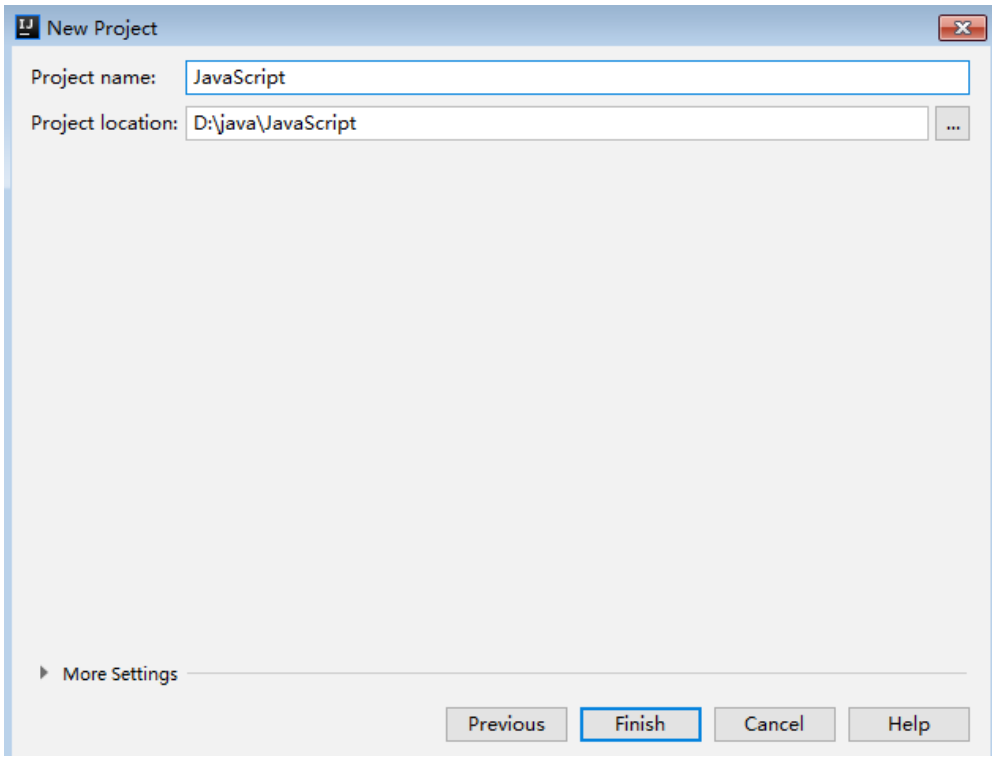
Figure 2-26 New Project



Step 2 Click **...**, select the directory where the SDK is decompressed, and click **Finish**.

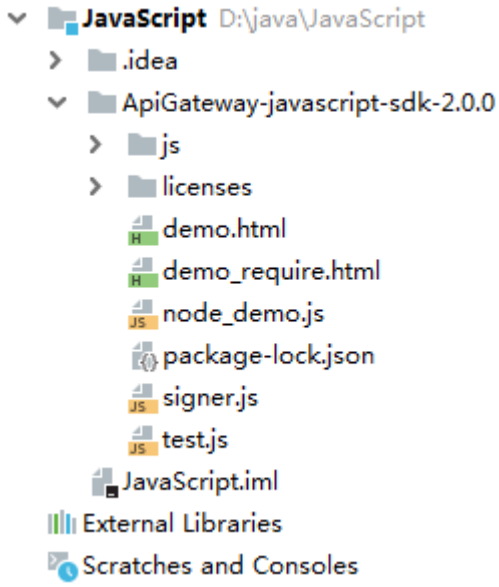
 Copyright © Huawei Technologies Co., Ltd.

Figure 2-27 Selecting the SDK directory after decompression



Step 3 View the directory structure shown in the following figure.

Figure 2-28 Directory structure of the new project

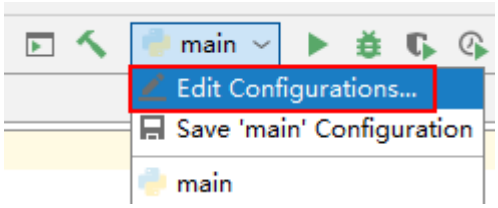


- **node_demo.js**: Sample code in Node.js. Modify the parameters in the sample code as required. For details about the sample code, see **API Calling Example (Node.js)**.

Step 4 Click **Edit Configurations**.

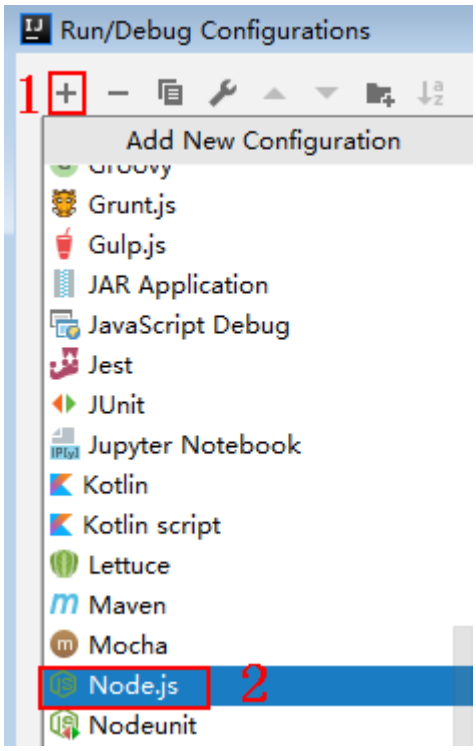
 Copyright © Huawei Technologies Co., Ltd.

Figure 2-29 Edit Configurations



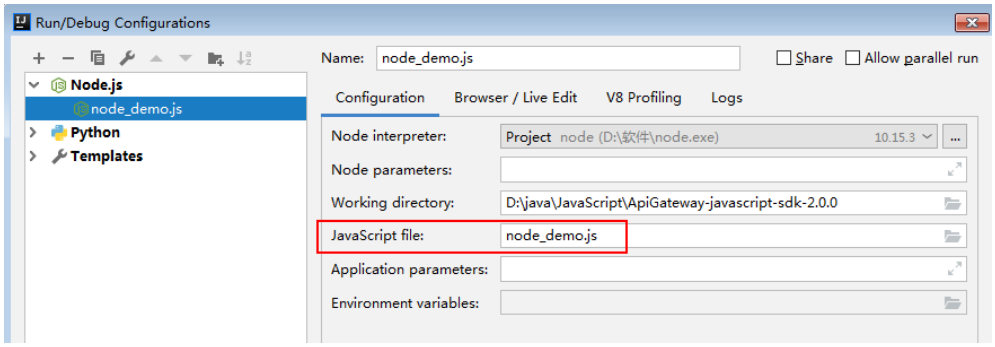
Step 5 Click + and select **Node.js**.

Figure 2-30 Selecting Node.js



Step 6 Set **JavaScript file** to **node_demo.js** and click **OK**.

Figure 2-31 Selecting node_demo.js



----End

 Copyright © Huawei Technologies Co., Ltd.

API Calling Example (Node.js)

Step 1 Import **signer.js** to your project.

```
var signer = require('./signer')
var http = require('http')
```

Step 2 Generate a new signer and enter the AppKey and AppSecret.

```
var sig = new signer.Signer()
sig.Key = "4f5f626b-073f-402f-a1e0-e52171c6100c"
sig.Secret = "******"
```

Step 3 Generate a request, and specify the method, request URI, and request body.

```
var r = new signer.HttpRequest("POST", "c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-north-1.huaweicloud.comserviceEndpoint/app1?a=1");
r.body = '{"a":1}'
```

Step 4 Add a header to the request. The header contains specific parameters. Add other headers to be signed as necessary. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
r.headers = { "x-stage":"RELEASE", "x-api-id": "********"}
```

Step 5 Execute the following function to generate HTTP(s) request parameters, and add the X-Sdk-Date and Authorization headers for signing the request: Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
var opt = sig.Sign(r)
opt.headers["x-Authorization"] = opt.headers["Authorization"]
```

Step 6 Access the API and view the access result. If you access the API using HTTPS, change **http.request** to **https.request**.

```
var req=http.request(opt, function(res){
        console.log(res.statusCode)
        res.on("data",    function(chunk){
        console.log(chunk.toString())
    })
})
req.on("error",function(err){
    console.log(err.message)
})
req.write(r.body)
req.end()
```

----End

2.2.7 PHP

Scenarios

To use PHP to call an API through App authentication, obtain the PHP SDK, create a new project, and then call the API by referring to **API Calling Example**.

This section uses IntelliJ IDEA 2018.3.5 as an example.

Preparing the Environment

- You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed IntelliJ IDEA. If not, download IntelliJ IDEA from the **official IntelliJ IDEA website** and install it.

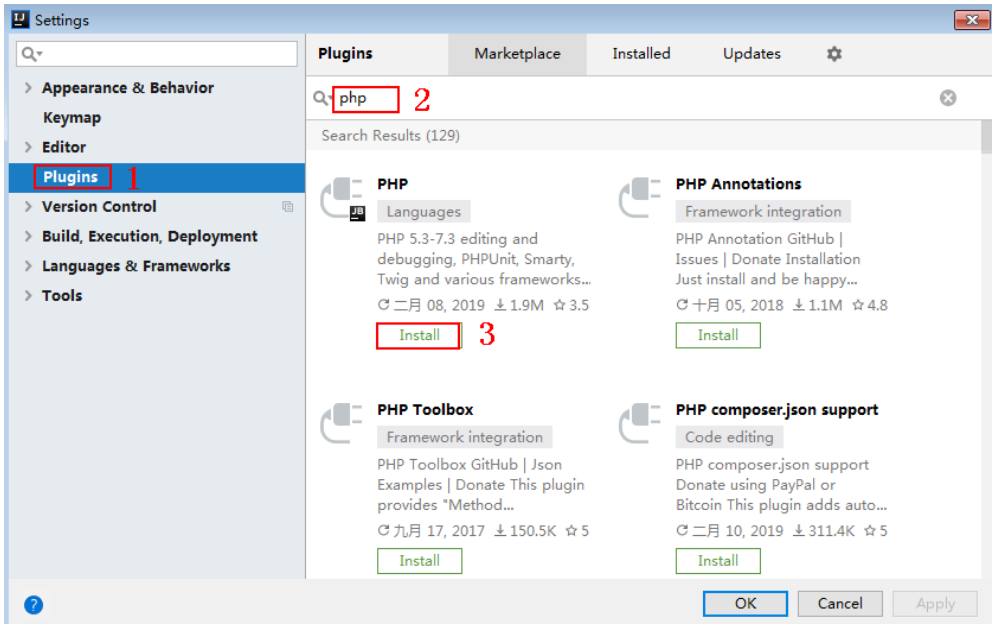
 Copyright © Huawei Technologies Co., Ltd.

- You have installed the PHP programming language. If not, download the PHP installation package from the **official PHP website** and install it.
- Copy the **php.ini-production** file from the PHP installation directory to the **C:\windows** directory, rename the file as **php.ini**, and then add the following lines to the file:

```
extension_dir = "PHP installation directory/ext"
extension=openssl
extension=curl
```

- You have installed the PHP plug-in on IntelliJ IDEA. If not, install the PHP plug-in according to **Figure 2-32**.

Figure 2-32 Installing the PHP plug-in



Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-php-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-php-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
signer.php	SDK code
index.php	Sample code

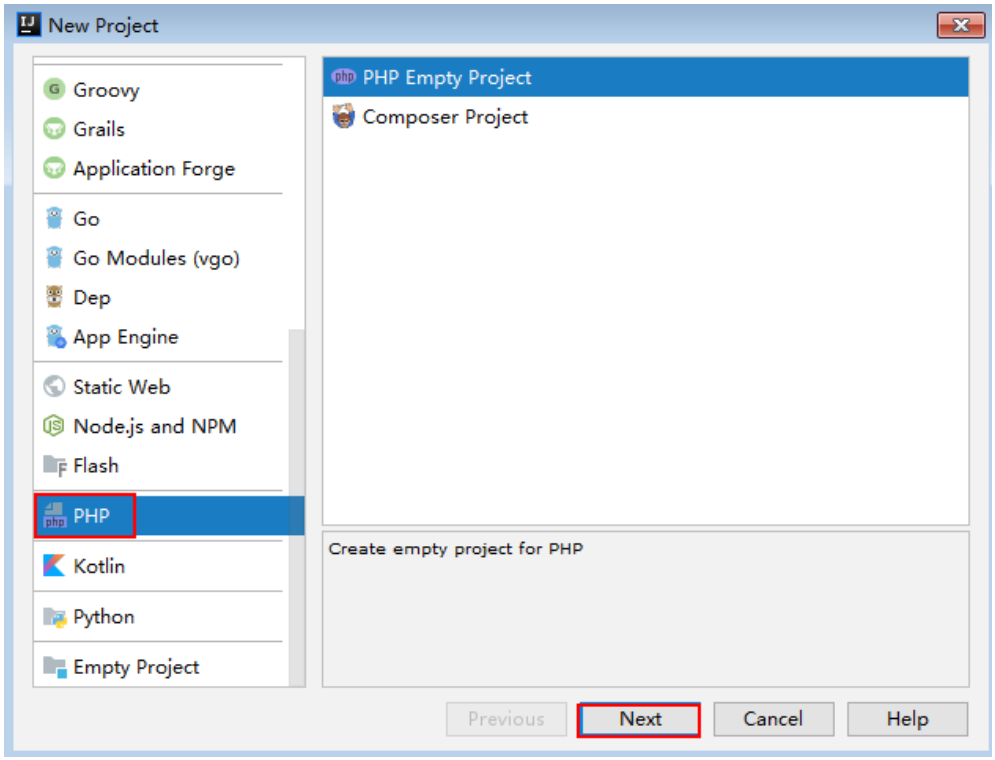
Creating a Project

Step 1 Start IDEA and choose **File** > **New** > **Project**.

 Copyright © Huawei Technologies Co., Ltd.

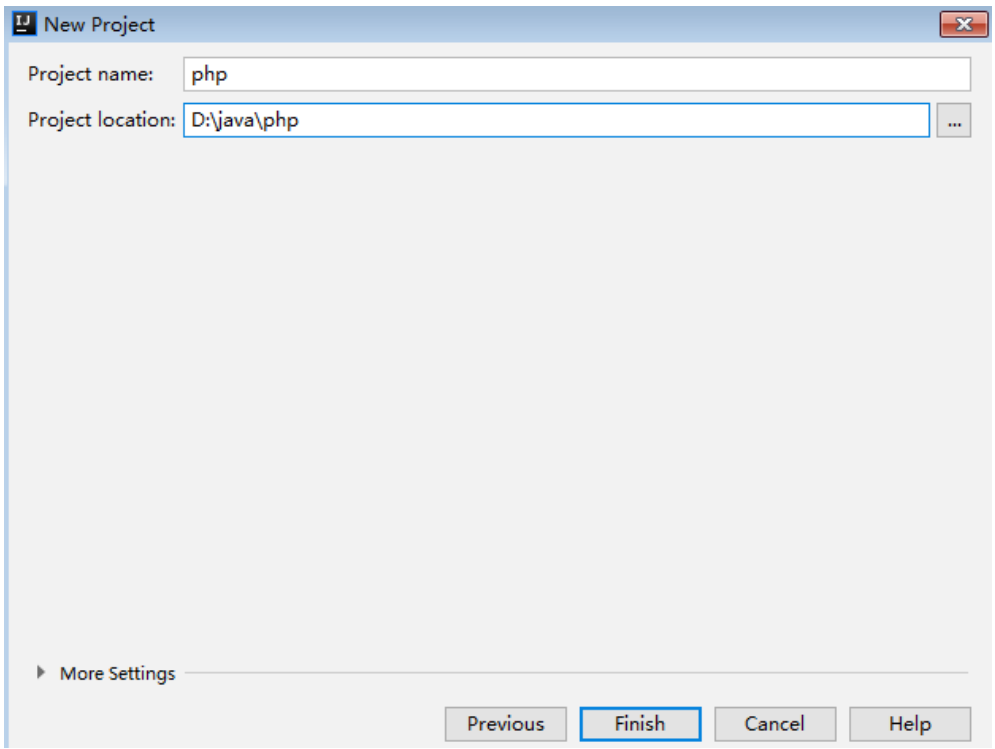
On the displayed **New Project** page, choose **PHP** and click **Next**.

Figure 2-33 New Project



Step 2 Click **...**, select the directory where the SDK is decompressed, and click **Finish**.

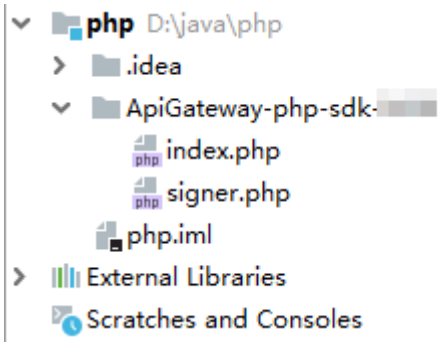
Figure 2-34 Selecting the SDK directory after decompression



 Copyright © Huawei Technologies Co., Ltd.

Step 3 View the directory structure shown in the following figure.

Figure 2-35 Directory structure of the new project



Modify the parameters in sample code **signer.php** as required. For details about the sample code, see **API Calling Example**.

----End

API Calling Example

Step 1 Import the PHP SDK to your code.

```
require 'signer.php';
```

Step 2 Generate a new signer and enter the AppKey and AppSecret.

```
$signer = new Signer();
$signer->Key = '4f5f626b-073f-402f-a1e0-e52171c6100c';
$signer->Secret = "******";
```

Step 3 Generate a new request, and specify the method, request URL, and body.

```
$req = new Request('GET', "https://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-north-1.huaweicloud.comserviceEndpoint/app1?a=1");
$req->body = '';
```

Step 4 Add a header to the request. The header contains specific parameters. Add other headers to be signed as necessary. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
$req->headers = array(
    'x-stage' => 'RELEASE',
    'x-api-id'=> 'f9d56b010e5c34d9ff92a08bfa3d7fb5',
);
```

Step 5 Execute the following function to generate a **$curl** context variable. Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
$curl = $signer->Sign($req);
$req->headers['x-Authorization'] = $req->headers['Authorization'];
$header = array();
foreach ($req->headers as $key => $value) {
    array_push($header, strtolower($key) . ':' . trim($value));
}
curl_setopt($curl, CURLOPT_HTTPHEADER, $header);
```

Step 6 Access the API and view the access result.

```
$response = curl_exec($curl);
echo curl_getinfo($curl, CURLINFO_HTTP_CODE);
```

 Copyright © Huawei Technologies Co., Ltd.

```
echo $response;
curl_close($curl);
```

----End

2.2.8 C++

Scenarios

To use C++ to call an API through App authentication, obtain the C++ SDK, and then call the API by referring to the API calling example.

Preparing the Environment

1. You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
2. Install the OpenSSL library.
   ```
   apt-get install libssl-dev
   ```
3. Install the curl library.
   ```
   apt-get install libcurl4-openssl-dev
   ```

Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-cpp-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-cpp-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
hasher.cpp	SDK code
hasher.h	
header.h	
RequestParams.cpp	
RequestParams.h	
signer.cpp	
signer.h	
Makefile	**Makefile** file
main.cpp	Sample code

 Copyright © Huawei Technologies Co., Ltd.

API Calling Example

Step 1 Add the following references to **main.cpp**:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <curl/curl.h>
#include "signer.h"
```

Step 2 Generate a new signer and enter the AppKey and AppSecret.

```
Signer signer("4f5f626b-073f-402f-a1e0-e52171c6100c", "******");
```

Step 3 Generate a new **RequestParams** request, and specify the method, domain name, request URI, query strings, and request body.

```
 RequestParams* request = new RequestParams("POST", "c967a237-cd6c-470e-906f-
a8655461897e.apigw.cn-north-1.huaweicloud.comserviceEndpoint", "/app1",
        "Action=ListUsers&Version=2010-05-08", "demo");
```

Step 4 Add a header to the request. The header contains specific parameters. Add other headers to be signed as necessary. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
request->addHeader("x-stage", "RELEASE");
request->addHeader("x-api-id","*******");
```

Step 5 Execute the following function to add the generated headers to the request variable. Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
signer.createSignature(request);
for (auto header : *request->getHeaders()) {
    if( strcmp(header.getKey().data(), "Authorization") == 0){
        request->addHeader("x-Authorization", header.getValue());
    }
}
```

Step 6 Use the curl library to access the API and view the access result.

```
static size_t
WriteMemoryCallback(void *contents, size_t size, size_t nmemb, void *userp)
{
    size_t realsize = size * nmemb;
    struct MemoryStruct *mem = (struct MemoryStruct *)userp;

    mem->memory = (char*)realloc(mem->memory, mem->size + realsize + 1);
    if (mem->memory == NULL) {
        /* out of memory! */
        printf("not enough memory (realloc returned NULL)\n");
        return 0;
    }

    memcpy(&(mem->memory[mem->size]), contents, realsize);
    mem->size += realsize;
    mem->memory[mem->size] = 0;

    return realsize;
}

//send http request using curl library
int perform_request(RequestParams* request)
{
    CURL *curl;
    CURLcode res;
    struct MemoryStruct resp_header;
    resp_header.memory = (char*)malloc(1);
    resp_header.size = 0;
    struct MemoryStruct resp_body;
    resp_body.memory = (char*)malloc(1);
```

 Copyright © Huawei Technologies Co., Ltd.

```
    resp_body.size = 0;

    curl_global_init(CURL_GLOBAL_ALL);
    curl = curl_easy_init();

    curl_easy_setopt(curl, CURLOPT_CUSTOMREQUEST, request->getMethod().c_str());
    std::string url = "http://" + request->getHost() + request->getUri() + "?" + request->getQueryParams();
    curl_easy_setopt(curl, CURLOPT_URL, url.c_str());
    struct curl_slist *chunk = NULL;
    std::set<Header>::iterator it;
    for (auto header : *request->getHeaders()) {
        std::string headerEntry = header.getKey() + ": " + header.getValue();
        printf("%s\n", headerEntry.c_str());
        chunk = curl_slist_append(chunk, headerEntry.c_str());
    }
    printf("--------------\n");
    curl_easy_setopt(curl, CURLOPT_HTTPHEADER, chunk);
    curl_easy_setopt(curl, CURLOPT_COPYPOSTFIELDS, request->getPayload().c_str());
    curl_easy_setopt(curl, CURLOPT_NOBODY, 0L);
    curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, WriteMemoryCallback);
    curl_easy_setopt(curl, CURLOPT_HEADERDATA, (void *)&resp_header);
    curl_easy_setopt(curl, CURLOPT_WRITEDATA, (void *)&resp_body);
    //curl_easy_setopt(curl, CURLOPT_VERBOSE, 1L);
    res = curl_easy_perform(curl);
    if (res != CURLE_OK) {
        fprintf(stderr, "curl_easy_perform() failed: %s\n", curl_easy_strerror(res));
    }
    else {
        long status;
        curl_easy_getinfo(curl, CURLINFO_HTTP_CODE, &status);
        printf("status %d\n", status);
        printf(resp_header.memory);
        printf(resp_body.memory);
    }
    free(resp_header.memory);
    free(resp_body.memory);
    curl_easy_cleanup(curl);

    curl_global_cleanup();

    return 0;
}
```

Step 7 Run the **make** command to obtain a **main** executable file, execute the file, and then view the execution result.

----End

2.2.9 C

Scenarios

To use C to call an API through App authentication, obtain the C SDK, and then call the API by referring to the API calling example.

Preparing the Environment

1. You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
2. Install the OpenSSL library.
   ```
   apt-get install libssl-dev
   ```
3. Install the curl library.

 Copyright © Huawei Technologies Co., Ltd.

```
apt-get install libcurl4-openssl-dev
```

Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-c-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-c-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
signer_common.c	SDK code
signer_common.h	
signer.c	
signer.h	
Makefile	**Makefile** file
main.c	Sample code

API Calling Example

Step 1 Add the following references to **main.c**:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <curl/curl.h>
#include "signer.h"
```

Step 2 Generate a sig_params_t variable, and enter the AppKey and AppSecret.

```
sig_params_t params;
sig_params_init(&params);
sig_str_t app_key = sig_str("4f5f626b-073f-402f-a1e0-e52171c6100c");
sig_str_t app_secret = sig_str("******");
params.key = app_key;
params.secret = app_secret;
```

Step 3 Specify the method, domain name, request URI, query strings, and request body.

```
sig_str_t host = sig_str("c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-
north-1.huaweicloud.comserviceEndpoint");
sig_str_t method = sig_str("GET");
sig_str_t uri = sig_str("/app1");
sig_str_t query_str = sig_str("a=1&b=2");
sig_str_t payload = sig_str("");
params.host = host;
params.method = method;
params.uri = uri;
params.query_str = query_str;
params.payload = payload;
```

Step 4 Add a header to the request. The header contains specific parameters. Add other headers to be signed as necessary. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

 Copyright © Huawei Technologies Co., Ltd.

```
sig_headers_add(&params.headers, "x-stage", "RELEASE");
sig_headers_add(&params.headers, "x-api-id","*******");
```

Step 5 Execute the following function to add the generated headers to the request variable. Then, add the **x-Authorization** header to the request. The value of the x-Authorization header is the same as that of the **Authorization** header.

```
sig_sign(&params);
char* authorization = sig_headers_get(&params.headers, "Authorization")->value.data;
sig_headers_add(&params.headers, "x-Authorization", authorization);
```

Step 6 Use the curl library to access the API and view the access result.

```
static size_t
WriteMemoryCallback(void *contents, size_t size, size_t nmemb, void *userp)
{
    size_t realsize = size * nmemb;
    struct MemoryStruct *mem = (struct MemoryStruct *)userp;

    mem->memory = (char*)realloc(mem->memory, mem->size + realsize + 1);
    if (mem->memory == NULL) {
        /* out of memory! */
        printf("not enough memory (realloc returned NULL)\n");
        return 0;
    }

    memcpy(&(mem->memory[mem->size]), contents, realsize);
    mem->size += realsize;
    mem->memory[mem->size] = 0;

    return realsize;
}

//send http request using curl library
int perform_request(RequestParams* request)
{
    CURL *curl;
    CURLcode res;
    struct MemoryStruct resp_header;
    resp_header.memory = malloc(1);
    resp_header.size = 0;
    struct MemoryStruct resp_body;
    resp_body.memory = malloc(1);
    resp_body.size = 0;

    curl_global_init(CURL_GLOBAL_ALL);
    curl = curl_easy_init();

    curl_easy_setopt(curl, CURLOPT_CUSTOMREQUEST, params.method.data);
    char url[1024];
    sig_snprintf(url, 1024, "http://%V%V?%V", &params.host, &params.uri, &params.query_str);
    curl_easy_setopt(curl, CURLOPT_URL, url);
    struct curl_slist *chunk = NULL;
    for (int i = 0; i < params.headers.len; i++) {
        char header[1024];
        sig_snprintf(header, 1024, "%V: %V", &params.headers.data[i].name, &params.headers.data[i].value);
        printf("%s\n", header);
        chunk = curl_slist_append(chunk, header);
    }
    printf("-------------\n");
    curl_easy_setopt(curl, CURLOPT_HTTPHEADER, chunk);
    curl_easy_setopt(curl, CURLOPT_POSTFIELDS, params.payload.data);
    curl_easy_setopt(curl, CURLOPT_NOBODY, 0L);
    curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, WriteMemoryCallback);
    curl_easy_setopt(curl, CURLOPT_HEADERDATA, (void *)&resp_header);
    curl_easy_setopt(curl, CURLOPT_WRITEDATA, (void *)&resp_body);
    //curl_easy_setopt(curl, CURLOPT_VERBOSE, 1L);
    res = curl_easy_perform(curl);
    if (res != CURLE_OK) {
        fprintf(stderr, "curl_easy_perform() failed: %s\n", curl_easy_strerror(res));
```

 Copyright © Huawei Technologies Co., Ltd.

```
    }
    else {
        long status;
        curl_easy_getinfo(curl, CURLINFO_HTTP_CODE, &status);
        printf("status %d\n", status);
        printf(resp_header.memory);
        printf(resp_body.memory);
    }
    free(resp_header.memory);
    free(resp_body.memory);
    curl_easy_cleanup(curl);

    curl_global_cleanup();

    //free signature params
    sig_params_free(&params);
    return 0;
}
```

Step 7 Run the **make** command to obtain a **main** executable file, execute the file, and then view the execution result.

----End

2.2.10 Android

Scenarios

To use Android to call an API through App authentication, obtain the Android SDK, create a new project, and then call the API by referring to the API calling example.

Preparing the Environment

- You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.
- You have installed Android Studio. If not, download Android Studio from the **official Android Studio website** and install it.

Obtaining the SDK

Download the SDK at "https://apig.*${Domain name of OBS}*/apig-sdk/ApiGateway-android-sdk.zip".

Replace *${Domain name of OBS}* with the actual domain name of OBS obtained from the administrator.

Download ApiGateway-android-sdk.zip. The following table shows the files decompressed from the package.

Name	Description
app\	Android project code
gradle\	Gradle files
build.gradle	Gradle configuration files
gradle.properties	

 Copyright © Huawei Technologies Co., Ltd.

Name	Description
settings.gradle	
gradlew	Gradle Wrapper scripts
gradlew.bat	

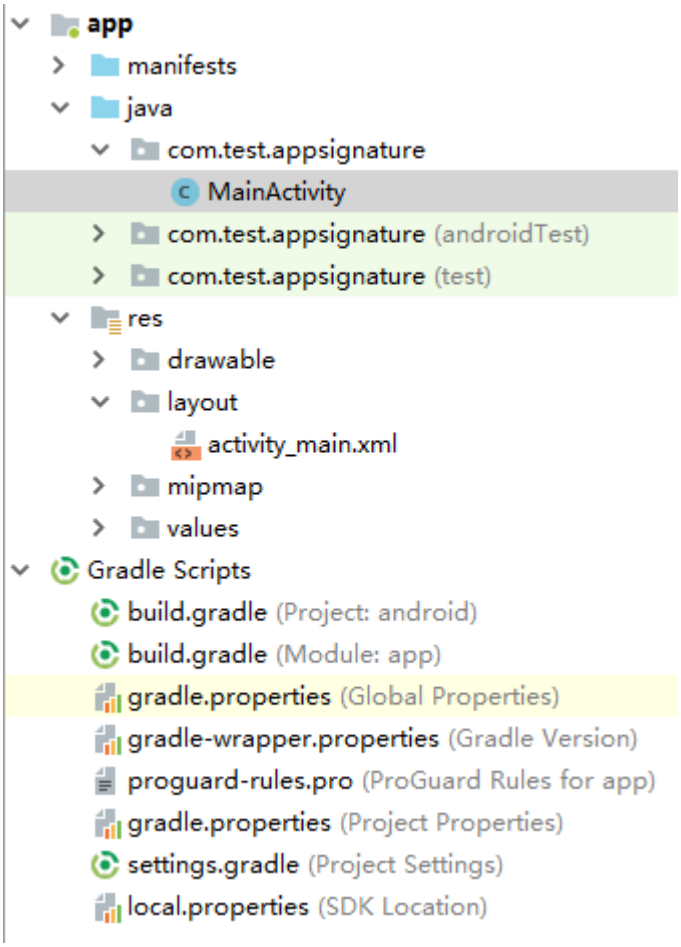
Opening a Project

Step 1 Start the Android Studio and choose **File** > **Open**.

Select the directory where the SDK is decompressed.

Step 2 View the directory structure of the project shown in the following figure.

Figure 2-36 Project directory structure



----End

API Calling Example

Step 1 Add required JAR files to the **app/libs** directory of the Android project. The following JAR files must be included:

 Copyright © Huawei Technologies Co., Ltd.

- java-sdk-core-x.x.x.jar
- commons-logging-1.2.jar
- joda-time-2.9.9.jar

Step 2 Add dependencies of the **okhttp** library to the **build.gradle** file.

Add **implementation 'com.squareup.okhttp3:okhttp:3.11.0'** in the **dependencies** field of the **build.gradle** file.

```
dependencies {
    ...
    ...
    implementation 'com.squareup.okhttp3:okhttp:3.11.0'
}
```

Step 3 Create a request, enter an AppKey and AppSecret, and specify the domain name, method, request URI, and body. The ID of the accessed API is mandatory. You need to enter the specific ID information. **x-api-id** is used in the sample code.

```
Request request = new Request();
try {
    request.setKey("4f5f626b-073f-402f-a1e0-e52171c6100c");
    request.setSecrect("******");
    request.setMethod("POST");
    request.setUrl("https://c967a237-cd6c-470e-906f-a8655461897e.apigw.cn-
north-1.huaweicloud.comserviceEndpoint/app1");
    request.addQueryStringParam("name", "value");
    request.addHeader("Content-Type", "text/plain");
        request.addHeader("x-api-id", "*********");
    request.setBody("demo");
} catch (Exception e) {
    e.printStackTrace();
    return;
}
```

Step 4 Sign the request and add the **x-Authorization** header to the request. The value of the **x-Authorization** header is the same as that of the **Authorization** header. The **okhttp3.Request** object is then generated to access the API.

```
okhttp3.Request signedRequest = Client.signOkhttp(request);
String authorization = signedRequest.header("Authorization");
signedRequest = signedRequest.newBuilder().addHeader("x-Authorization",authorization).build();
OkHttpClient client = new OkHttpClient.Builder().build();
Response response = client.newCall(signedRequest).execute();
```

----End

2.2.11 curl

Scenarios

To use the curl command to call an API through App authentication, download the JavaScript SDK to generate the curl command, and copy the command to the CLI to call the API.

Prerequisites

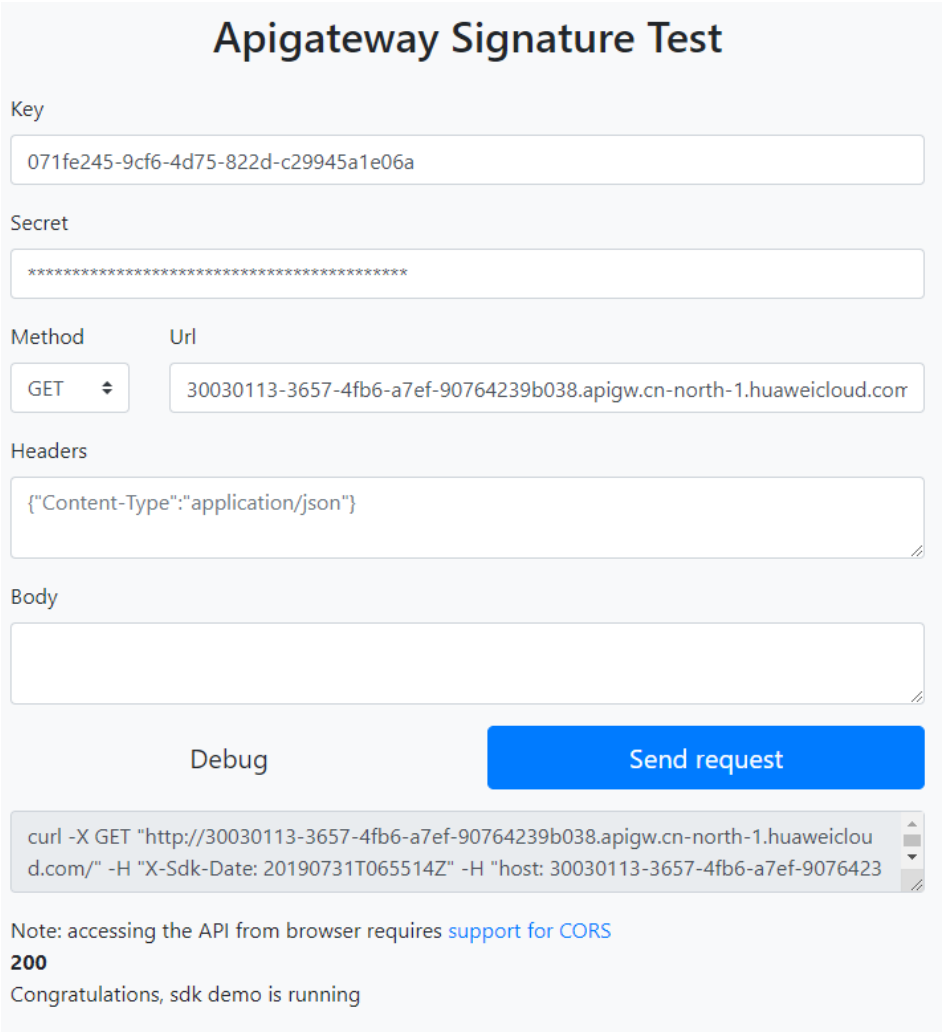
You have obtained the domain name, request URL, and request method of the API to be called, and the AppKey and AppSecret of the App for calling the API. For more information, see **Preparation**.

 Copyright © Huawei Technologies Co., Ltd.

API Calling Example

Step 1 Use the JavaScript SDK to generate the curl command.

Download the **JavaScript SDK** and decompress it. Open **demo.html** in a browser. The following figure shows the demo page.



Step 2 Specify the key, secret, method, protocol, domain name, and URL. For example:

```
Key=4f5f626b-073f-402f-a1e0-e52171c6100c
Secret=******
Method=POST
Url=https://{apig-endpoint}serviceEndpoint
```

Step 3 Specify query and header parameters in JSON format, and set the request body. The ID of the accessed API is required. You need to enter the specific ID information. The **x-api-id** parameter is used in **Headers**.

Step 4 Click **Send request** to generate a **curl** command.

```
$ curl -X POST "https://{apig-endpoint}serviceEndpoint/" -H "X-Sdk-Date: 20180530T115847Z" -H
"Authorization: SDK-HMAC-SHA256 Access=071fe245-9cf6-4d75-822d-c29945a1e06a,
SignedHeaders=host;x-sdk-date, Signature=9e5314bd156d517******dd3e5765fdde4" -d ""
```

Step 5 Then, add the **x-Authorization** header to the command. The value of **x-Authorization header** is the same as that of the **Authorization** header. Copy the **curl** command to the CLI to access the API.

 Copyright © Huawei Technologies Co., Ltd.

```
$ curl -X POST "https://{apig-endpoint}serviceEndpoint/" -H "X-Sdk-Date: 20180530T115847Z" -H
"Authorization: SDK-HMAC-SHA256 Access=071fe245-9cf6-4d75-822d-c29945a1e06a,
SignedHeaders=host;x-sdk-date, Signature=9e5314bd156d517******dd3e5765fdde4" -H "X-Authorization:
SDK-HMAC-SHA256 Access=071fe245-9cf6-4d75-822d-c29945a1e06a, SignedHeaders=host;x-sdk-date,
Signature=9e5314bd156d517******dd3e5765fdde4" -d ""
Congratulations, sdk demo is running
```

----End

2.2.12 Other Programming Languages

App Authentication Principle

1. Construct a standard request.

   Assemble the request content according to the rules of API Gateway, ensuring that the client signature is consistent with that in the backend request.
2. Create a to-be-signed string using the standard request and other related information.
3. Calculate a signature using the AK/SK and to-be-signed string.
4. Add the generated signature to an HTTP request as a header or query parameter.
5. After receiving the request, API Gateway performs **1** to **3** to calculate a signature.
6. The new signature generated in **3** is compared with the signature generated in **5**. If they are consistent, the request is processed; otherwise, the request is rejected.

NOTE

The body of a signing request in app authentication mode cannot exceed 12 MB.

Step 1: Constructing a Standard Request

To access an API through app authentication, standardize the request content, and then sign the request. The client must follow the same request specifications as API Gateway so that each HTTP request can obtain the same signing result from the frontend and backend to complete identity authentication.

The pseudocode of standard HTTP requests is as follows:

```
CanonicalRequest =
    HTTPRequestMethod + '\n' +
    CanonicalURI + '\n' +
    CanonicalQueryString + '\n' +
    CanonicalHeaders + '\n' +
    SignedHeaders + '\n' +
    HexEncode(Hash(RequestPayload))
```

The following example shows how to construct a standard request.

Original request:

```
GET https://30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint/
app1?b=2&a=1 HTTP/1.1
Host: 30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint
X-Sdk-Date: 20180330T123600Z
```

 Copyright © Huawei Technologies Co., Ltd.

1. Specify an HTTP request method (**HTTPRequestMethod**) and end with a carriage return line feed (CRLF).

   Specify GET, PUT, POST, or another request method. Example of a request method:

   ```
   GET
   ```

2. Add a standard URI (**CanonicalURI**) and end with a CRLF.

   Description

   Path of the requested resource, which is the URI code of the absolute path.

   Format

   According to RFC 3986, each part of a standard URI except the redundant and relative paths must be URI-encoded. If a URI does not end with a slash (/), add a slash at its end.

   Example

   For the URI **/app1**, the standard URI code is as follows:

   ```
   GET
   /app1/
   ```

   NOTE

   During signature calculation, the URI must end with a slash (/). When a request is sent, the URI does not need to end with a slash (/).

3. Add a standard query string (**CanonicalQueryString**) and end with a CRLF.

   Description

   Query parameters. If no query parameters are configured, the query string is an empty string.

   Format

   Standard query strings must meet the following requirements:

   - Perform URI encoding on each parameter and value according to the following rules:
     - Do not perform URI encoding on any non-reserved characters defined in RFC 3986, including A–Z, a–z, 0–9, hyphen (-), underscore (_), period (.), and tilde (~).
     - Use **%XY** to perform percent encoding on all non-reserved characters. **X** and **Y** indicate hexadecimal characters (0–9 and A–F). For example, the space character must be encoded as **%20**, and an extended UTF-8 character must be encoded in the "%XY%ZA%BC" format.
   - Add "*URI-encoded parameter name* = *URI-encoded parameter value*" to each parameter. If no value is specified, use a null string instead. The equal sign (=) is required.

     For example, in the following string that contains two parameters, the value of parameter **parm2** is null.

     ```
     parm1=value1&parm2=
     ```

   - Sort the parameters in alphabetically ascending order. For example, a parameter starting with uppercase letter **F** precedes another parameter starting with lowercase letter **b**.

 Copyright © Huawei Technologies Co., Ltd.

– Construct standard query strings from the first parameter after sorting.

Example

The following example contains two optional parameters **a** and **b**.

```
GET
/app1/
a=1&b=2
```

4. Add standard headers (**CanonicalHeaders**) and end with a CRLF.

Description

List of standard request headers, including all HTTP message headers in the to-be-signed request. The **X-Sdk-Date** header must be included to verify the signing time, which is in the UTC time format *YYYYMMDDTHHMMSSZ* as specified in ISO 8601. When publishing an API in a non-RELEASE environment, you need to specify an environment name.

Format

CanonicalHeaders consists of multiple message headers, for example, **CanonicalHeadersEntry0 + CanonicalHeadersEntry1 + ...**. Each message header (**CanonicalHeadersEntry**) is in the format of **Lowercase(HeaderName) + ':' + Trimall(HeaderValue) + '\n'**.

NOTE

- **Lowercase**: a function for converting all letters into lowercase letters.
- **Trimall**: a function for deleting the spaces before and after a value.
- The last message header carries a CALF. Therefore, an empty line appears because the **CanonicalHeaders** field also contains a CALF according to the specifications.

Example

```
GET
/app1/
a=1&b=2
host:30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint
x-sdk-date:20180330T123600Z
```

NOTICE

Standard message headers must meet the following requirements:

- All letters in a header are converted to lowercase letters, and all spaces before and after the header is deleted.
- All headers are sorted in alphabetically ascending order.

For example, the original headers are as follows:

```
Host: 30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint\n
Content-Type: application/json;charset=utf8\n
My-header1:    a   b   c  \n
X-Sdk-Date:20180330T123600Z\n
My-Header2:    "a   b   c"  \n
```

A standard header is as follows:

```
content-type:application/json;charset=utf8\n
host:30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint\n
my-header1:a   b   c\n
my-header2:"a   b   c"\n
x-sdk-date:20180330T123600Z\n
```

5. Add message headers (**SignedHeaders**) for request signing, and end with a CALF.

 Copyright © Huawei Technologies Co., Ltd.

Description

List of message headers used for request signing. This step is to determine which headers are used for signing the request and which headers can be ignored during request verification. **x-sdk-date** and **x-api-id** must be used as the signed message headers. **x-api-id** indicates the ID of the accessed API. You need to enter the specific ID information.

Format

SignedHeaders = Lowercase(HeaderName0) + ';' + Lowercase(HeaderName1) + ";" + ...

Letters in the message headers are converted to lowercase letters. All headers are sorted alphabetically and separated with commas.

Lowercase is a function for converting all letters into lowercase letters.

Example

In the following example, the message headers **host**, **x-sdk-date**, and **x-api-id** are used for signing the request.

```
GET
/app1/
a=1&b=2
host:30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint
x-sdk-date:20180330T123600Z
x-api-id:*******

host;x-sdk-date;x-api-id
```

6. Use a hash function, such as SHA-256, to create a hash value based on the body (**RequestPayload**) of the HTTP or HTTPS request.

   Description

   Request message body. The message body needs two layers of conversion (**HexEncode(Hash(*RequestPayload*))**). **Hash** is a function for generating message digest. Currently, SHA-256 is supported. **HexEncode**: the Base16 encoding function for returning a digest consisting of lowercase letters. For example, **HexEncode ("m")** returns **6d** instead of **6D**. Each byte you enter is expressed as two hexadecimal characters.

   NOTE

   If **RequestPayload** is null, the null value is used for calculating a hash value.

   Example

   For a request with the GET method and an empty body, the body (empty string) after hash processing is as follows:

   ```
   GET
   /app1/
   a=1&b=2
   host:30030113-3657-4fb6-a7ef-90764239b038.apigw.cn-north-1.huaweicloud.comserviceEndpoint
   x-sdk-date:20180330T123600Z

   host;x-sdk-date
   e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
   ```

7. Perform hash processing on the standard request in the same way as that on the **RequestPayload**. After hash processing, the standard request is expressed with lowercase hexadecimal strings.

   Algorithm pseudocode:
   Lowercase(HexEncode(Hash.SHA256(CanonicalRequest)))

   Example of standard request after hash processing:

 Copyright © Huawei Technologies Co., Ltd.

```
4bd8e1afe76738a332ecff075321623fb90ebb181fe79ec3e23dcb081ef15906
```

Step 2: Creating a To-Be-Signed String

After a standard HTTP request is constructed and the request hash value is obtained, create a to-be-signed string by combining them with the signature algorithm and signing time.

```
StringToSign =
  Algorithm + \n +
  RequestDateTime + \n +
  HashedCanonicalRequest
```

Parameters in the pseudocode are described as follows:

- **Algorithm**

  Signature algorithm. For SHA256, the value is **SDK-HMAC-SHA256**.

- **RequestDateTime**

  Request timestamp, which is the same as **X-Sdk-Date** in the request header. The format is *YYYYMMDDTHHMMSSZ*.

- **HashedCanonicalRequest**

  Standard request generated after hash processing.

In this example, the following to-be-signed string is obtained:

```
SDK-HMAC-SHA256
20180330T123600Z
4bd8e1afe76738a332ecff075321623fb90ebb181fe79ec3e23dcb081ef15906
```

Step 3: Calculating the Signature

Use the AppSecret and created character string as the input of the encryption hash function, and convert the calculated binary signature into a hexadecimal expression.

The pseudocode is as follows:

```
signature = HexEncode(HMAC(APP secret, string to sign))
```

HMAC indicates hash calculation, and **HexEncode** indicates hexadecimal conversion. **Table 2-1** describes the parameters in the pseudocode.

Table 2-1 Parameter description

Parameter	Description
AppSecret	Signature key.
To-be-signed string	Character string to be signed.

Assuming that the AppSecret is **12345678-1234-1234-1234-123456781234**, a signature similar to the following will be calculated:

```
cb978df7c06ac242bab1d1b39d697ef7df4806664a6e09d5f5308a6b25043ea2
```

 Copyright © Huawei Technologies Co., Ltd.

Step 4: Adding the Signature to the Request Header

Add the signature to the HTTP Authorization header. The Authorization header is used for identity authentication and not included in the signed headers.

The pseudocode is as follows:

```
Authorization header creation pseudocode:
Authorization: algorithm Access=APP key, SignedHeaders=SignedHeaders, Signature=signature
```

There is no comma before the algorithm and **Access**. **SignedHeaders** and **Signature** must be separated with commas.

The signed headers are as follows:

```
Authorization: SDK-HMAC-SHA256 Access=071fe245-9cf6-4d75-822d-c29945a1e06a, SignedHeaders=host;x-sdk-date, Signature=cb978df7c06ac242bab1d1b39d697ef7df4806664a6e09d5f5308a6b25043ea2
```

After obtaining the signed message headers, add them to the original HTTP request header with the **Authorization** and **x-Authorization** parameters. The request is sent to API Gateway for identity authentication. If the identity authentication is successful, the request is sent to the corresponding backend service for processing.

 Copyright © Huawei Technologies Co., Ltd.

A Change History

Released On	Description
2019-11-22	Adjusted the contents structure.
2019-09-03	This issue is the first official release.

 Copyright © Huawei Technologies Co., Ltd.