HMS-Wallet-Server Demo Development Tutorial

1. Introduction

HMS-Wallet-Server Demo is example code showing how to use the HMS-Wallet-Server interface. The HMS-Wallet-Server interface contains REST APIs for six types of passes (Loyalty Card, Offer, Gift Card, Boarding Pass, Transit Pass, and Event Ticket). You can use these REST APIs to implement operations such as adding, querying or updating passes.

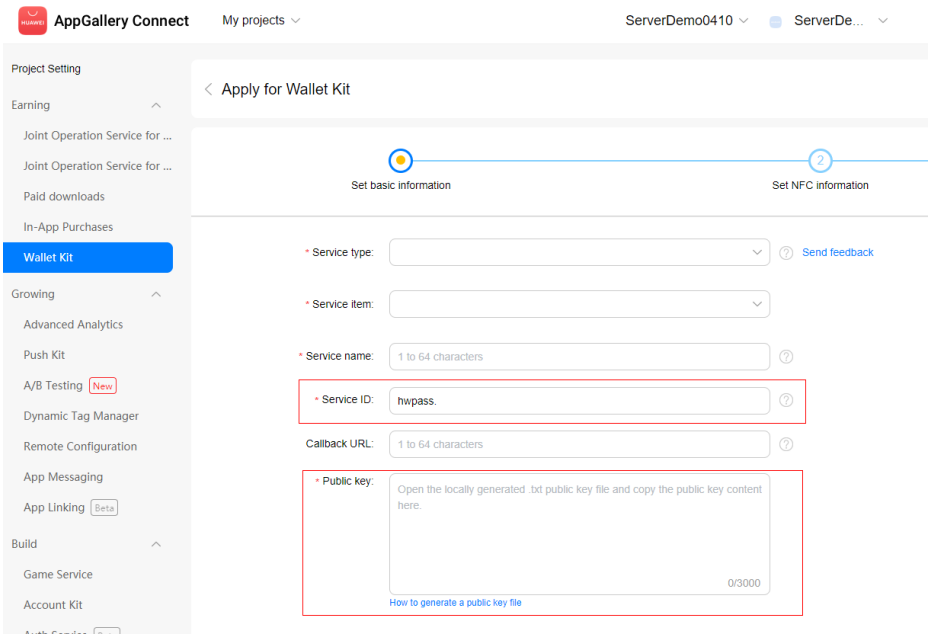
Before you use this Demo, you should have a HUAWEI developer account, and have already created an app to implement the HMS-Wallet-Kit API. This “app” doesn’t have to be a software that will be actually installed on cell phones. It means an application of using the HMS service. So please create an app on the HUAWEI AppGallery Connect (AGC) website even if you are not going to develop mobile apps. If you haven't, please refer to Register a HUAWEI ID and Create an App.

Maven and Oracle Java 1.8.0.211 or higher are required to run the Demo project.

2. Apply for Wallet Services

Follow Enabling Services to apply for services.

Please notice that you will set a “Service ID” in your application. The “Service ID” here is “passTypeIdentifier” you will use in the Demo code. You will set the “passTypeIdentifier” later.



You will generate a pair of RSA keys during this process. Store them properly, and you will use the private key to sign JWEs (see 8.8). The wallet server will use the public key you upload to verify signatures.

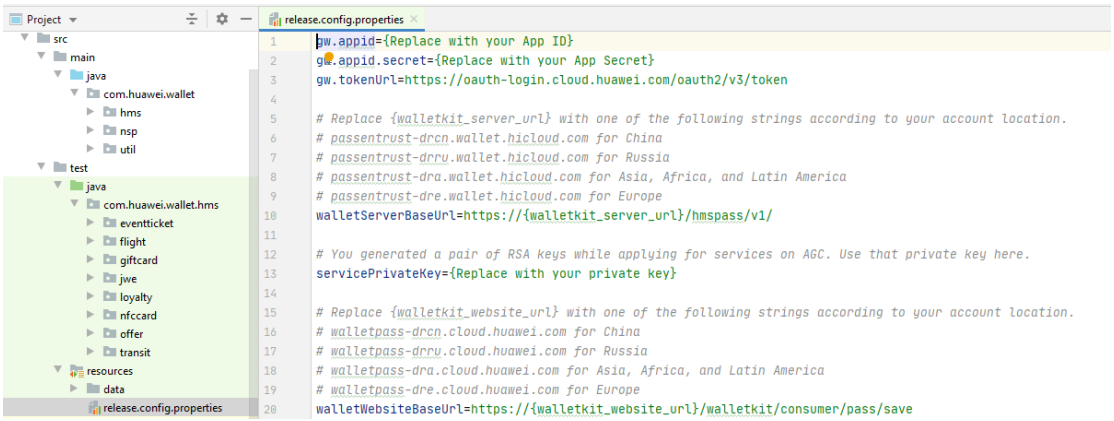
After you finished applying for a service, you can begin to test the corresponding pass. Apply for other services if you want to test other passes.

3. Download the Demo

Download “huawei-wallet-server-windows-jwe-demo.zip” from HMS Wallet Kit Server Sample and extract the zip file.

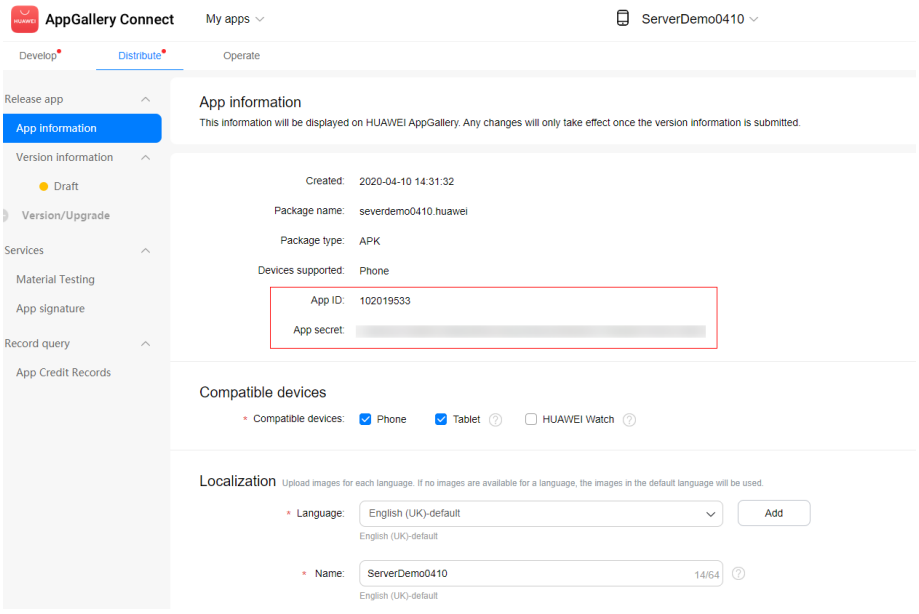
4. Set Configuration Values

Before running the Demo project，you need to set the following configuration values in the “src\test\resources\release.config.properties” file: “gw.appid”, “gw.appid.secret”, “gw.tokenUrl”, “walletServerBaseUrl”, “servicePrivateKey”, and “walletWebsiteBaseUrl”.



4.1 Set “gw.appid” and “gw.appid.secret”

To implement the HMS Wallet Kit to an app, "gw.appid" and "gw.appid.secret" are this app's "App ID" and "App secret". Go to the AGC website, login to your account, click “My apps” and then click the app you want to operate. Then you can find its App ID and App secret as shown below:



4.2 Set “gw.tokenUrl”

Set gw.tokenUrl = https://oauth-login.cloud.huawei.com/oauth2/v3/token. This is the address to obtain a REST API authentication token.

4.3 Set “walletServerBaseUrl”

“walletServerBaseUrl” is a common section of the REST APIs’ http requests. Its format is: walletServerBaseUrl = https://{walletkit_server_url}/hmspass/v1/. Set {walletkit_server_url} with one of the values in the following table according to your account’s location.

location	walletkit_server_url
China	passentrust-drcn.wallet.hicloud.com
Russia	passentrust-drru.wallet.hicloud.com
Asia, Africa, and Latin America	passentrust-dra.wallet.hicloud.com
Europe	passentrust-dre.wallet.hicloud.com

4.4 Set “servicePrivateKety”

You generated a pair of RSA private key and public key while you applied HMS Wallet service on the AGC website. Set the private key here and you will use it to sign JWEs.

4.5 Set “walletWebsiteBaseUrl”

“walletWebsiteBaseUrl” is the address of HMW-Wallet-H5 server. Its format is: walletWebsiteBaseUrl=https://{walletkit_website_url}/walletkit/consumer/pass/save. Set {walletkit_server_url} with one of the values in the following table according to your account’s location.

location	walletkit_website_url
China	walletpass-drcn.cloud.huawei.com
Russia	walletpass-drru.cloud.huawei.com
Asia, Africa, and Latin America	walletpass-dra.cloud.huawei.com
Europe	walletpass-dre.cloud.huawei.com

5. Pass Models and Pass Instances

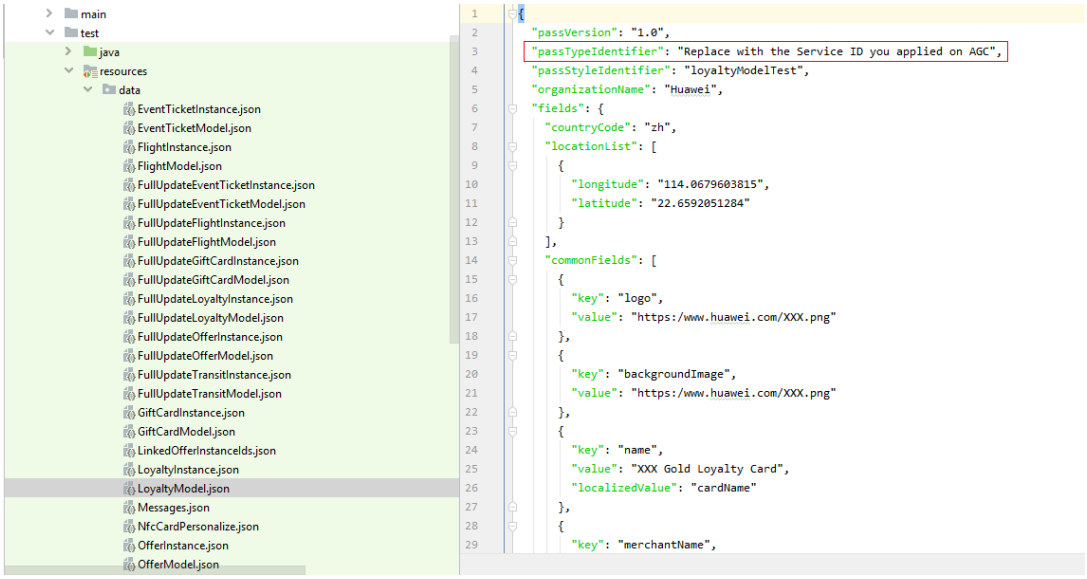
A pass model is a style of pass instances. Instances belonging to the same model share some common parameters. For example, a boarding pass model contains information about departure time and arrival time, while a boarding pass instance contains a passenger’s name, his seat, his boarding sequence, etc. Each pass instance belongs to a specific model. Hence, you should first create a model before creating instances and performing other operations.

All pass models and instances have the same data format, which is HwWalletObject. Refer to HwWalletObject Definition for more details.

Input parameters of models and instances are passed by JSON files in the Demo project. You

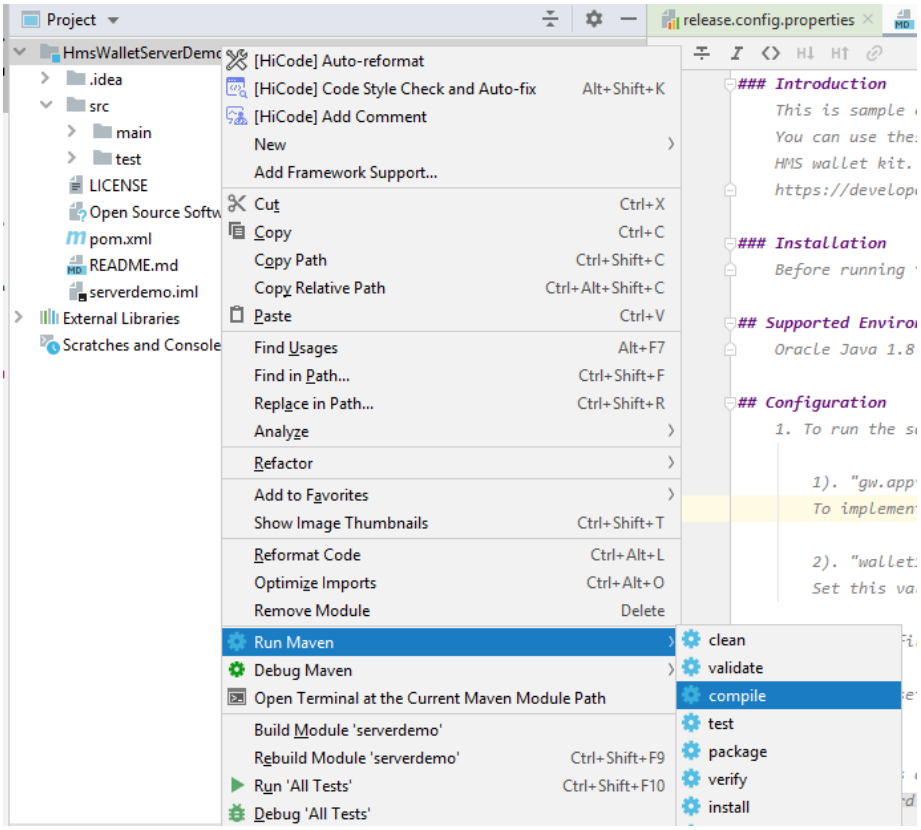
can generate your own data by modifying the JSON files in the “src/test/resources/data” folder.

Remember to set the correct “passTypeIdentifier” in these input JSON files, which should be identical to your “Service ID” on the AGC website.



6. Compile the Demo as a Maven Project

After you set all the configurations values, compile the Demo as a Maven project. It may take around 20 minutes to deploy all dependencies. It depends on your settings files and your network environment.



7. Example Methods for Pass Models

7.1 Register a Pass Model

The HMS wallet server provides REST APIs to create pass models. See Creating a Model. You can add a loyalty model to the server's database by calling the "createLoyaltyModel" method, and create other types of models likewise. You should create a pass model first before calling any other methods.

7.2 Query a Pass Model

The HMS wallet server provides REST APIs to query a pass model by its model ID. See Query a Model. You can query a loyalty model by calling the "getLoyaltyModel" method, and query other types of models likewise.

7.3 Query a List of Pass Models

If your app created multiple models of a pass type (e.g. a gold loyalty card model and a diamond loyalty card model), you can use these APIs to query a list of these models. See Query Models. You can query a list of loyalty models by calling the "getLoyaltyModelList" method, and query other types of models likewise.

7.4 Overwrite a Pass Model

The HMS wallet server provides REST APIs to overwrite an entire pass model by its model ID. See Overwrite a Model. You can overwrite a loyalty model by calling the "fullUpdateLoyaltyModel" method, and overwrite other types of models likewise.

7.5 Update a Pass Model

The HMS wallet server provides REST APIs to update part of a pass model by its model ID. See Update a Model. You can overwrite a loyalty model by calling the "partialUpdateLoyaltyModel" method, and update other types of models likewise.

7.6 Add Messages to a Pass Model

"messageList" is one of the attributes in a pass model, which is a list of messages. You can add messages to a loyalty model by calling the "addMessageToLoyaltyModel" method, and add to other types of models likewise. The "messageList" in a pass model has at most 10 messages. You cannot add more than 10 messages at a time. If the list's size is already 10 and you keep adding messages, the oldest messages will be removed. See Add Messages to a Model.

8. Example Methods for Pass Instances

8.1 Add a Pass Instance

The HMS wallet server provides REST APIs to add pass instances. See Add an Instance. You can add a loyalty instance to the server's database by calling the "createLoyaltyInstance" method, and create other types of instances likewise. You should create a pass model first before creating

instances belonging to it.

8.2 Query a Pass Instance

The HMS wallet server provides REST APIs to query a pass instance by its instance ID. See Query an Instance. You can query a loyalty instance by calling the “getLoyaltyInstance” method, and query other types of instances likewise.

8.3 Query a List of Pass Instances

The HMS wallet server provides REST APIs to query all instances for a certain model. See Query Instances. You can query a list of instances belonging to a specific loyalty model by calling the “getLoyaltyInstanceList” method, and query other types of instances likewise.

8.4 Overwrite a Pass Instance

The HMS wallet server provides REST APIs to overwrite an entire pass instance by its instance ID. See Overwrite an Instance. You can overwrite a loyalty instance by calling the “fullUpdateLoyaltyInstance” method, and overwrite other types of instances likewise.

8.5 Update a Pass Instance

The HMS wallet server provides REST APIs to update part of a pass instance by its instance ID. See Update an Instance. You can overwrite a loyalty instance by calling the “partialUpdateLoyaltyInstance” method, and update other types of instances likewise.

8.6 Add Messages to a Pass Instance

“messageList” can also be an attribute of a pass instance. You can add messages to a loyalty instance by calling the “addMessageToLoyaltyInstance” method, and add to other types of instances likewise. The “messageList” in a pass instance has at most 10 messages. You cannot add more than 10 messages at a time. If the list’s size is already10 and you keep adding messages, the oldest messages will be removed. See Add Messages to an Instance.

8.7 Link/unlink Offer Instances to/from a Loyalty Instance

This API is only provided for loyalty instances. See Link/Unlink an Offer. You can link/unlink offer instances to/from a loyalty instance by calling the “updateLinkedOffersToLoyaltyInstance” method. You should make sure the offers you want to add already exist before you use this API. Otherwise, the client cannot show an offer that is not in the server’s database. These offer instances can belong to other apps or other developers.

9. Generate JSON Web Encryption (JWE)

Developers need to generate JWE strings and send them to HMS-Wallet-H5 server (See 4.4) to bind a pass instance to a Huawei Wallet user.

There are two ways to generate JWEs. The first way: you can generate a JWE string with complete information of a pass instance and send it to the HMS-Wallet-H5 server. By this way, you don’t need to call methods in 8.1. The second way: you can call methods in 8.1 to add a pass instance to wallet server first, and then generate a thin JWE (with only instance ID information) and send it

to the HMS-Wallet-H5 server to bind the pass instance to a user.

The demo has example methods for generating JWEs and thin JWEs. Please refer to the demo and implement the code in your own system. Please refer to Integrate Add to HUAWEI Wallet Button (section “Generating a JWE and Sending It to the Huawei Server” and section “Generating a Thin JWE and Sending It to the Huawei Server”) for more details.