Sample Code

Determining service entitlement on the server

Identify a customer's entitlement to your service, offers, and messaging by analyzing a validated receipt and the state of their subscription.

Download

Overview

> **Note**
>
> This sample code project is associated with WWDC22 session 110404: Implement proactive in-app purchase restore.
>
> It's also associated with WWDC 2020 session 10671: Architecting for Subscriptions.

When your app provides a subscription service, you receive data in the form of in-app receipts and App Store server notifications that provide a subscription's state. After reading this data, your app determines whether to allow or restrict the user's access to service, products, offers, and messages.

This server-side sample code project analyzes a receipt to help the app determine entitlement by systematically examining all potential subscription states that affect service entitlement, such as offers periods and billing states. The output from this sample's REST API endpoints is a response that indicates whether the app can enable service or needs to perform other steps to retain or communicate with customers. You can extend this code and customize the response based on your own business logic. The sample code contains two endpoints:

- URL: `localhost:3000/simulate` — Use `/simulate` for testing using real or artificial receipts. This endpoint requires no special configuration.

- URL: `localhost:3000/entitle` — Use `/entitle` with real Base64-encoded receipts. This endpoint requires additional configuration.

Configure the sample code project

Follow these steps to run the sample code project:

1. Install the latest stable version of Node.js.
2. Open the Terminal app (`/Applications/Utilities`).
3. Navigate to the sample code `/Sample` directory.
4. In Terminal, enter `npm install` and press Return. Make sure it completes successfully.
5. In Terminal, copy the example environment file: `cp env.example`.
6. In Terminal, enter `npm start` and press Return to start the server. The server runs in your Terminal window. To stop the server, press Control-C while in Terminal.
7. In Terminal, choose Shell > New Windows > New Window with Profile to open a second Teminal window to use to send the requests.

Send a request using sample receipt data

The requests to the service entitlement engine take JSON data from a receipt. The data must be in the same JSON format that you receive by calling the `/verifyReceipt` endpoint when you pass it a valid receipt. The example file `flatJSONExample` in the `Source/Example` directory contains sample receipt data in the correct format for the requests.

To run the request, switch to the second Terminal window and type the following `curl` command. Replace the `<FLAT JSON DATA>` with the `flatJSONExample`.

```
curl -XPOST -H "Content-type: application/json" -d '<FLAT JSON DATA>' 'localhost:300
```

If the mocked JSON receipt data is in the correct format, the response contains the analyzed data the entitlement engine produces based on the receipt data.

If you have real receipt data, replace the `<FLAT JSON DATA>` in the command with your receipt data.

Receive the sample response

The response contains the result of the engine's analysis of the receipt data. Use it to determine whether a customer has the necessary entitlements to service, offers, or other messaging. The response object contains an array of subscription objects organized by product ID, each containing fields that provide insights for that subscription.

The following example response shows a customer who was previously subscribed and then resubscribed. The customer is in state 5, which indicates an active subscription with autorenew still in an enabled state.


```
{
    "products": [
        {
            "product_id": "com.example.monthly",
            "entitlementCode": 5,
            "expiration": 1599391591000,
            "totalRenewals": 7,
            "groupID": "13472270",
            "originalTransactions": [
                {
                    "originalTX": "190000625698817",
                    "start": 1564455960000,
                    "expiration": 1599391591000,
                    "renewals": 7
                }
            ],
            "trialConsumed": true
        }
    ],
    "trialConsumedForGroup": [
        "13472270"
    ]
}
```


In this case, the sample app issues an offer for a yearly subscription to the customer because they're in state 5, and they have a history of seven total renewals.