PassMate Integration Guide

This document provides a guide for integrating PassMate with your Android application to enable push notifications and updates for digital passes (Passbook).

PassMate follows Apple's (Passbook) Wallet Web Service API standards, so if you have already integrated with Apple's Wallet Web Service, you will find the integration with PassMate to be very similar. The only difference is the replacement of APNs (Apple Push Notification Service) with PassMate's push notification service, which is used to send updates to the passes installed on Android devices.

This guide assumes you already have issued passes in the Passbook format (.pkpass) and are familiar with the structure of the pass.json file.
If you do not have a pass yet, you can refer to the Apple's Passbook documentation for creating passes: https://developer.apple.com/documentation/walletpasses

Authorization

PassMate provides a platform for all companies developing Digital Card (Passbook) solutions aiming at Android Users. In order to use PassMate's push notification service and update passes, you need to register your application and obtain an API key.

To Register your company to get your API Key Please just send an inquiry to
support@getpassmate.com
with following details:
Company Name, Contact Name (PIC), E-mail, Phone, Website

Pass Configuration

Before you can integrate PassMate with your application, you need to ensure that your pass is correctly configured. This is crucial for the pass to function properly and to allow updates after installation.

Specifically you need to have *webServiceURL* and *authenticationToken* keys set in the top-level of your *pass.json* file.

webServiceURL is the base URL of your server that will handle the registration requests and updates for the pass.

authenticationToken is a unique token that your server will use to authenticate requests from the PassMate service. This token should be securely generated and stored.

Registering users for push notifications

When a user installs your pass, their device will send a registration request to your server with the pass details. Your server should handle this request and store the registration information for future updates.

The registration request will be in following format:

POST
{webserviceUrl}/v1/devices/{deviceLibraryIdentifier}/registrations/{passTypeIdentifier}/{serialNumber}

The `deviceLibraryIdentifier` is a unique identifier for the device, `passTypeIdentifier` is the identifier for your pass type, and `serialNumber` is the unique serial number of the pass.

The Authorization header of the request will contain the authenticationToken you included in your pass.json type, which is used to verify the request. The header should look like this :

[Authorization] => AndroidPass authToken

The request body will contain the following JSON data:

```
{
  "pushToken": "406a05c6-86c3-469c-bda3-34571869f6b7"
}
```

This token is a unique identifier for the device that will be used to send push notifications to the pass. You should store this token as it will be used later to send updates to the pass.

After receiving the registration request, your server should then respond with a 200 OK status code to acknowledge the registration. If there is an error, you should return an appropriate HTTP status code (e.g., 400 Bad Request, 500 Internal Server Error).

Sending push notifications for pass updates

To send push notifications to registered devices, you will need to make a POST request to the PassMate API with the necessary details.
The request should be made to the following endpoint:

POST api.getpassmate.com/api/partners/push

Request Headers
"Content-Type: application/json"
"X-Api-key: YOUR_API_KEY"

Request Body
```
{
   "passTypeIdentifier": "pass.com.yourcompany.sample",
   "pushTokens": [
      "abc1234",
      "bcd5678",
      ...
   ]
}
```

The `passTypeIdentifier` is the identifier for your pass type, and `pushTokens` is an array of push tokens that the devices have sent during registration for the devices you want to notify.

After receiving the request, PassMate will send push notifications to all devices with the specified push tokens. Each device will receive a notification that the pass has been updated, and the device will automatically refresh their pass to reflect the changes.

User-Agent Information to Distinguish PassMate Requests

If you have already integrated with Apple's (Passbook) Wallet Web Service and receive registrations from both platforms, you can easily detect the PassMate users by the User-Agent header. All requests from PassMate always contain the following User-Agent:

[User-Agent] => PassMate/{versionCode} (Linux; U; {versionName}; {deviceBrand} {deviceModel} Build/{buildId})

Linking Passes to PassMate App

PassMate supports **intent://** links that let you open passes directly in the app if it’s installed. If PassMate is not installed, the user is redirected to Google Play. After installation, the pass is automatically added to PassMate.

Format

intent://add?url=PASS_URL#Intent;scheme=passmate;package=com.getpassmate.wallet;S.browser_fallback_url=https%3A%2F%2Fplay.google.com%2Fstore%2Fapps%2Fdetails%3Fid%3Dcom.getpassmate.wallet%26referrer%3DPASS_URL;end

Replace PASS_URL with your pkpass download link

Always URL-encode your pass URL to avoid breaking links