Implementing the S-money InAppProvisioning SDK

GERALD MEPHANE

2019

Change History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Revision # | Author | Revision Description | Approved |
| 07/11/2019 | 00.01 | Gérald Mephane | Document Creation |  |
| 28/11/2019 | 00.02 | Gérald Mephane | Change SMIAPButton to SMIAPButtonManager |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Change history

Confidentiality notice

All information contained in this documentation, as well as the software described in it, is proprietary to S-money. Except as permitted by license agreement, no part of this documentation may be reproduced, stored in a retrieval system, or transmitted in any form or by electronic, mechanical, recording, or any other means, without the prior written permission of S-money.

Distribution of this document is restricted as indicated by the Authorized distribution displayed below.

For information regarding permissions, write to: (address)

Phone: +33……..… Email: ……………….. Web: ………………..

Table of contents

1. Introduction 3

2. Before you begin 4

3. Request access from Apple 5

4. Preliminary steps 8

4.1 importing the sdk into your project 8

4.2 setting up the sdk 9

4.2.1 SDK Registration. 9

4.2.2 Set UserId 10

4.2.3 Enable log 10

4.2.4 Configure the SDK (OPTIONAL) 11

4.2.4.1 Debug: 11

4.2.4.2 Security: 12

5. Implementation of the principal SDK functions 13

5.1 SMIAPButtonManager 14

5.1.1 Initialization 14

5.1.2 Add payment pass 15

5.1.3 Callbacks 16

1. Introduction

This document is intended for app developers on the iOS platform (smartphones and tablets).

The SDK, which is explained in this document, allows app developers to quickly and easily add card-provisioning feature to their applications.

1. Before you begin

Check that you have the following before you start the implementation:

* Request access from Apple.
* The latest version of the SmoneyInAppProvisioningSDK.
* All functionalities of sdk requires a **valid S-money access token.**

This access token will only be valid for this application. For more information on how to obtain this access token please contact S-money.

1. Request access from Apple

You will need to submit a request to enable your developer Team ID for the appropriate Apple Pay In-App Provisioning entitlements. Enterprise Team IDs are not supported. Provide your app name, Team ID, and Adam ID via e-mail to apple-pay-provisioning@apple.com.

Once the entitlements have been granted, you’ll need to include the distribution entitlement into a provisioning profile and ensure you are leveraging the same profile to develop the app within Xcode.

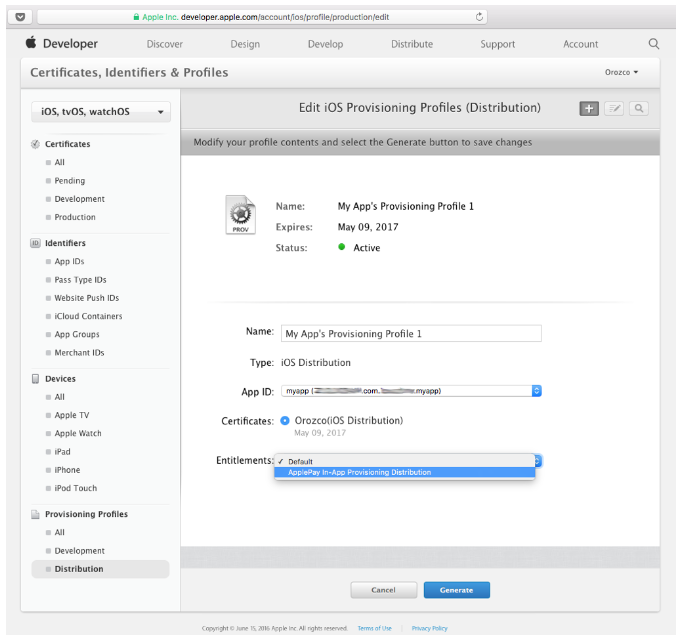
Please follow these steps:

1.Head to the Apple Developer Website and proceed to login

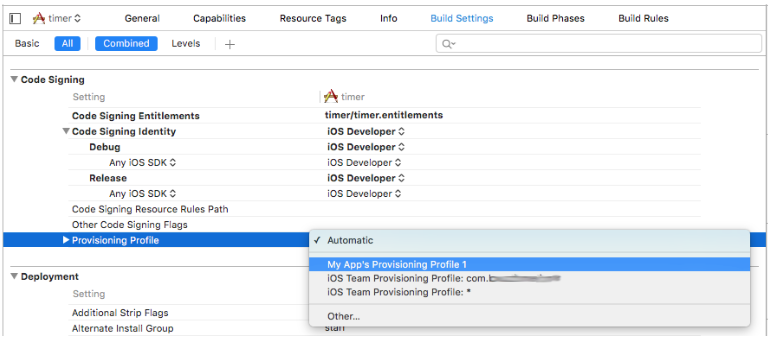
2.Select Certificates, Identifiers & Profiles

3.Select “Distribution” underneath the “Provisioning Profiles” heading on the sidebar 4.On the right, select the distribution iOS provisioning profile that you'll use to deploy your App to the App Store

5.Click “edit" and, from the ensuing entitlements drop down, select “ApplePay In-App Provisioning Distribution” to add the entitlement to the profile.



Once you’ve generated a profile which has been assigned the entitlement for In-App Provisioning, within Xcode, head to the Preferences > Accounts > (Your Account) > View Details pane where you can then find and download the profile you generated. Lastly, in build settings, you can then adjust the Provisioning Profile to the newly generated profile.



1. Preliminary steps

There are two preliminary steps that you need to perform to get started with the SDK:

1. Import the SDK into your project.
2. Set up the SDK

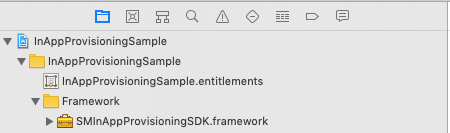
After you perform these steps, you will be ready to start implementing the core SDK functionality.

* 1. importing the sdk into your project

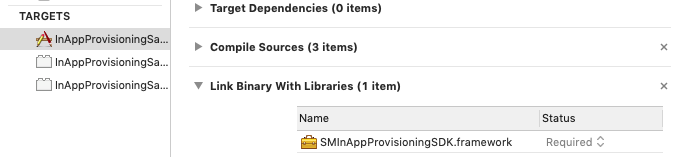
The SDK library must be imported into your Xcode project.

**Procedure**

1. Add the SMInAppProvisioningSDK.framework file to your project.



1. Check the build phases of your target.



* 1. setting up the sdk

The SDK must be set up and initialize before use all his features.

All SDK features requires a **valid S-money access token.**

SDK Registration.

The SDK must be initialized by calling the method initWithPartnerAccessToken of the SmoneyInAppProvisioning class.

The **access token** must be specified at this point. In the following example, it is configured in the application:didFinishLaunchingWithOptions: method of the AppDelegate class.

#import <SMInAppProvisioningSDK/SmoneyInAppProvisioning.h>

…

- (**BOOL**)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions {

[SmoneyInAppProvisioning setDebugMode:YES];

**[SmoneyInAppProvisioning initWithPartnerAccessToken:@"AccessToken"];**

return YES;

}

**N.B**: Replace the **AccessToken** value with the actual value provided by S-money.

Set UserId

The **user id** must be specified. In the following example, it is configured in the viewDidLoad method of the ViewController class.

#import <SMInAppProvisioningSDK/SmoneyInAppProvisioning.h>

…

- (**void**)viewDidLoad {

[**super** viewDidLoad];

*// Do any additional setup after loading the view, typically from a nib.*

**[SmoneyInAppProvisioning setUserID:@"UserId"];**

}

**N.B**: Replace the **UserId** value with the id used to subscribe your user.

Enable log

* Enable/disable **log**: No sensitive information such as credentials, tokens, etc. should be logged to the Console log facility. All logging code was **disable by default in release version** to ensure that no information is leaked via the logging mechanism or source files.

Method: setDebugMode:

*Default value*: debug mode (enabled) / **release mode (disabled)**

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions {

**[SmoneyInAppProvisioning setDebugMode:YES];**

[SmoneyInAppProvisioning initWithPartnerAccessToken:@"AccessToken"];

return YES;

}

Configure the SDK (OPTIONAL)

The SDK exposes configuration options to the developer. Additional configuration can be done with the initWithPartnerAccessToken:config: method.

Example:

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions {

[SmoneyInAppProvisioning setDebugMode:YES];

**SMIAPConfiguration \*configuration = [SMIAPConfiguration defaultConfiguration];**

**configuration.enableDebugDetection = YES;**

**[SmoneyInAppProvisioning initWithPartnerAccessToken:@"AccessToken" config:configuration];**

return YES;

}

There are some optional features that are enabled by default.

* + - 1. Debug:
* Enable/disable **debug detection**: **Outside of development**, releasing apps with **debuggable enabled** **is discouraged** as it allows connected computers to access and debug the app.

Parameter: enableDebugDetection

*Default value*: debug mode (disabled) / **release mode (disabled)**

* + - 1. Security:
* Enable/Disable **jailbreak detection**: Outside of development, it's unlikely that your app should be running on an emulator as it allows connected computers to access and debug the app.

Parameter: enableJailbreakDetection

*Default value*: debug mode (enabled) / **release mode (enabled)**

* Enable/Disable **emulator detection**: Outside of development, it's unlikely that your app should be running on an emulator.

Parameter: enableEmulatorDetection

*Default value*: debug mode (disabled) / **release mode (disabled)**

1. Implementation of the principal SDK functions

This section describes how to configure SMIAPButtonManager with your payment pass information and add your payment pass. Code examples are also provided.

* 1. SMIAPButtonManager

Initialization

The **SMIAPButtonManager** must be initialized by calling the method initWithCardHolderName:primaryAccountIdentifier: primaryAccountSuffix:delegate:sender.

**Request parameters**:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Description | |
| userholdername | String | Indicates The name of the person as shown on the card  **Required** | |
| primaryAccountIdentifier | String | Indicates the user's id.  **Required** | |
| primaryAccountSuffix | String | Indicates the last four or five digits of the card’s number.  **Required** | |
| delegate | SMIAPButtonManagerDelegate | | Indicates the protocol used to get add pass events.  **Required** |
| sender | UIViewController | | Indicates the view controller used to present PKAddPaymentPassViewController.  **Required** |

Table 1 - Request Parameters - Registration API

**Sample code:**

#import <SMInAppProvisioningSDK/SMIAPButtonManager.h>

…

- (void) onAddPass {

// Initialize manager with pass configuration

**SMIAPButtonManager\* manager = [[SMIAPButtonManager alloc] initWithCardHolderName:@"UserHolderName" primaryAccountIdentifier:@"CardPrimaryAccountIdentifier" primaryAccountSuffix:@"CardPrimaryAccountSuffix" delegate:self sender:self];**

}

Add payment pass

Once you’ve initialized the **SMIAPButtonManager** you can then add your payment pass by calling the method addPass.

**Sample code:**

#import <SMInAppProvisioningSDK/SMIAPButtonManager.h>

…

- (void) onAddPass {

// Initialize manager with pass configuration

…

// Start add payment pass

**[manager addPass];**

}

Callbacks

The SMIAPButtonManager uses **SMIAPButtonManagerDelegate** methods to convey important events that can occur during the life-cycle of the application.

This protocol let you implement the following methods:

* didFinishWithSuccess: Called when request has been made successfully.
* didFinishWithSuccess:(NSError \*)error: Called when an error occurs in the process.

**Sample code:**

#import <SMInAppProvisioningSDK/SMIAPButton.h>

**@interface** ViewController : UIViewController <**SMIAPButtonManagerDelegate**>

…

**@end**

- (**void**)didFinishWithSuccess {

}

- (**void**)didFinishWithError:(NSError \*) error {

}

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | | Description |
| SMIAPAddPaymentPassErrorUnsupported | 1 | | The app cannot add cards to Apple Pay. |
| SMIAPAddPaymentPassErrorUserCancelled | 2 | | The user canceled the request to add a card to Apple Pay. |
| SMIAPAddPaymentPassErrorSystemCancelled | 3 | | The system canceled the request to add a card to Apple Pay. |
| SMIAPErrorCodeUserCannotBeFound | 4 | | The user cannot be found. |
| SMIAPErrorCodeInvalidParameter | 5 | | The operation is invalid. |
| SMIAPErrorCodeCardNotFound | 6 | | The card cannot be found. |
| SMIAPErrorCodeOperationNotAuthorized | 7 | The operation is not authorized. | |
| SMIAPErrorCodeOperationInvalid | 8 | The operation is invalid. | |
| SMIAPErrorCodeOAuthTokenExpired | 9 | The token is expired. | |
| SMIAPErrorCodeOAuthTokenInvalid | 10 | The token is invalid. | |
| SMIAPErrorSecurityCheck | 11 | The app not passed security check. | |

*Table 2 - Error Code Description*