Historial de revisiones

Fecha	Versión	Descripción	Autor
06/09/2018	1.0	Generación del documento	Miguel Tomairo

Objetivo del documento

El objetivo de este documento es describir el proceso de integración de un comercio a la pasarela de pagos Payme Mobile de una manera dinámica y eficiente.

Este documento recoge los pasos requeridos por el comercio para realizar la integración a la pasarela de pago para poder procesar pagos con tarjetas de débito o crédito.

El lenguaje de programación de esta librería fue Swift, el cual es independiente al lenguaje de la APP con el que se integre.

Contenidos

1.0) Integración con Cocoapods

1.1) Software

IDE

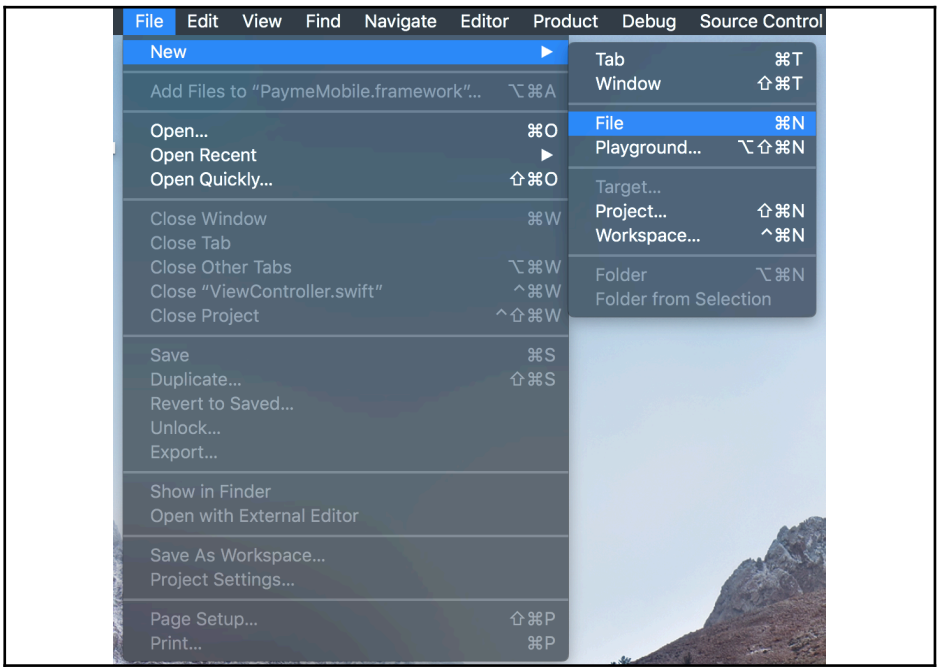
- Xcode 9.4.1

Plugin

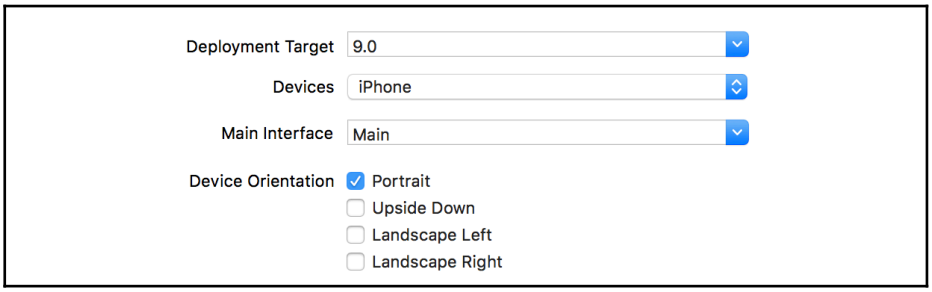
- Librería Payme Mobile

1.2) Creación de proyecto que usará la API

- Para Crear un proyecto en el Xcode, dirigirse a File > New > Project.



- La versión mínima del SDK debe ser iOS 9, Luego seguir el proceso normal de creación de una aplicación nueva.



1.3) Agregar cocoapods al proyecto

1.3.1 Antes de empezar

- Revisar las especificaciones del repositorio o cocoapods.org para estar seguro que la librería se encuentre habilitada.

1.3.2 Instalación

Abrimos el terminal y nos ubicamos en la carpeta del proyecto iOS y escribimos lo siguiente y presionamos enter.

pod init

Luego de ello, abrimos la carpeta del proyecto y ubicamos el archivo Podfile. Lo abrimos y escribimos lo siguiente.

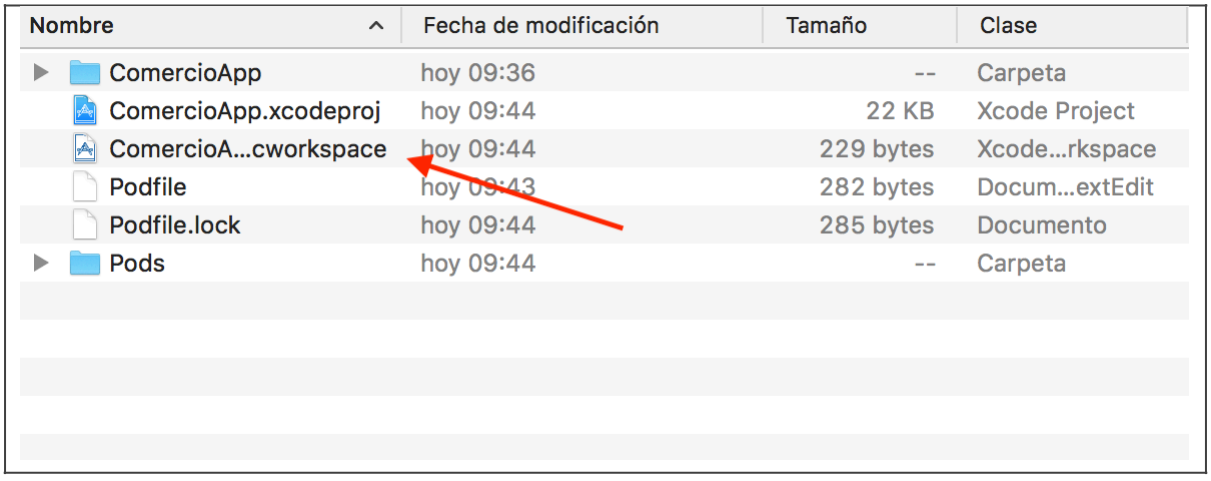
pod 'PaymeMobile'

```
target 'App' do
  # Comment the next line if you're not using Swift and don't
  use_frameworks!

  pod 'PaymeMobile'

end
```

Grabamos el archivo y lo cerramos. Ahora nos ubicamos de nuevo en la carpeta del proyecto iOS y escribimos lo siguiente:

pod install

Finalmente, abrimos el archivo de extension xcworkspace. Esto abrirá el Xcode y estaremos listos para utilizar la librería.



1.3.3 Permisos

Para empezar a utilizar la librería en tu proyecto iOS, debes agregar el permiso para utilizar la cámara en el archivo plist de tu proyecto.

Copiar la siguiente etiqueta en su archivo plist

<key>NSCameraUsageDescription</key>
<string>Se necesita usar la camara para escanear las tarjetas.</string>

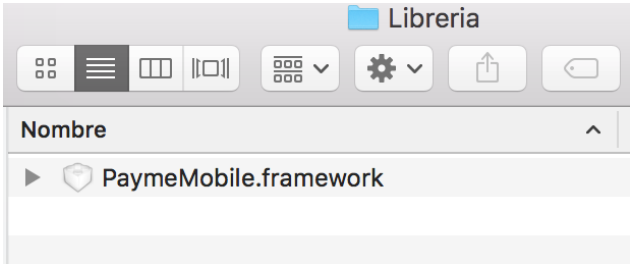
Information Property List	Dictionary	(15 items)
Localization native development re...	String	$(DEVELOPMENT_LANGUAGE)
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	APPL
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
Required device capabilities	Array	(1 item)
Supported interface orientations	Array	(3 items)
Supported interface orientations (i...	Array	(4 items)
Privacy - Camera Usage Description	String	Se necesita usar la camara para escanear las tarjetas.

Recordar que sin el permiso, su aplicativo no podrá usar la cámara y presentara un crash al momento del despliegue.

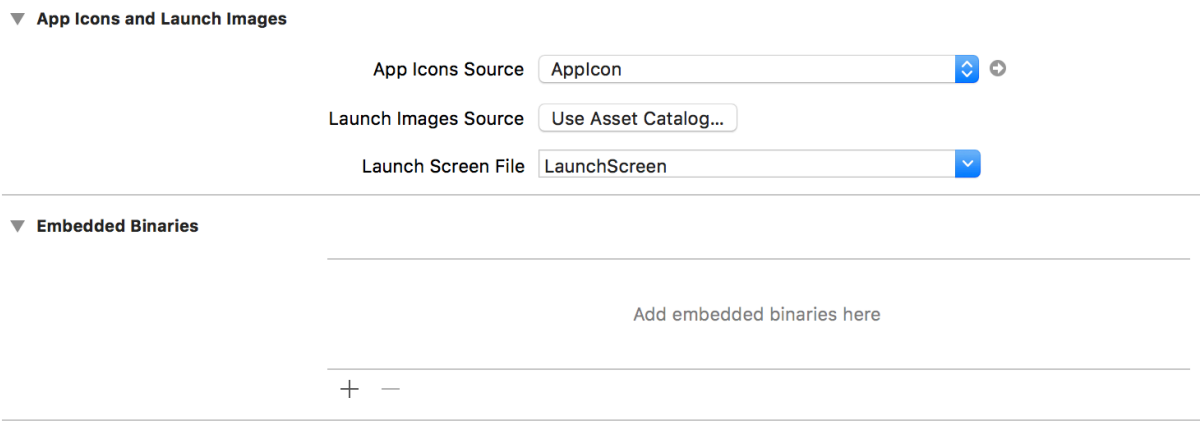
2.0) Integración manual

2.1) Integración con archivo Framework

Descargamos el archivo PaymeMobile.framework que esta publicado en la siguiente ruta (https://github.com/alignetdev/paymemobile-ios) en la carpeta Librería. Luego de ello nos dirigimos al proyecto iOS que deseamos instalarle la librería y le soltamos el archivo para que se integre al proyecto.



En el tag del proyecto. Por la parte inferior arrastramos y soltamos el archivo en la sección de Embedded Binaries.



3) Modelo de datos de la librería

3.1) Módulo de pagos

- A modo de prueba se creó campos para poder enviar dinámicamente los siguientes campos
- Locale
- Merchant
- Brands
- Operation
- number
- amount
- Currency
- productDes
- WalletUser.C
- PlanQuota
- Signature Key

3.2) Atributos de entrada y Salida

3.2.1) Entrada

MerchantOperationData
Objeto principal

Propiedad	Tipo	Descripción
features	MerchantData	Características adicionales durante la realización de la transacción.
data	SettingsData	Información de la compra con la cual se realizará la transacción. **Obligatorio**
settings	FeaturesData	Contiene los valores de ajustes del comportamiento del formulario de pago. **Obligatorio**

MerchantData
Objeto que contiene los valores de la compra con la cual se realizará la transacción.

Propiedad	Tipo	Descripción
operation	OperationData	Contiene los datos de la operación con la cual se procesará la transacción. **Obligatorio**
shipping	PersonaData	Información de datos de envió de la orden de compra.
billing	PersonaData	Información de datos de facturación de la orden de compra.
customer	PersonaData	Información del comprador que realiza la operación **Obligatorio**
signatureKey	String	HASH Generado con el algoritmo SHA-512 utilizando la cadena resultado de concatenar los campos ***{número de operación} + {monto} + {código de moneda} + {llave para firmar}***

SettingsData
Ajustes sobre el comportamiento del formulario de pago.

Propiedad	Tipo	Descripción
locale	String	Localización del idioma que utilizará el formulario para mostrarse. **Valor Descripción** es_PE Español Perú en_US Inglés EEUU
identifier	String	Identificador del comercio

brands	Array<String>	Contiene los valores de las marcas que serán habilitadas para el formulario de pago. **Valor** **Descripción** VISA Visa MSCD MasterCard AMEX American Express DINC Diners Club Nota: Las marcas se visualizarán en el formulario de pago de acuerdo al orden enviado en el Array. **Valor por defecto**: ['VISA']

FeaturesData
Características adicionales durante la realización de la transacción.

Propiedad	Tipo	Descripción
wallet	FeatureWalletData	Contiene los valores de ajustes del comportamiento para activar Wallet
reserved	Array<FeaturedReservedData>	Contiene los valores de ajustes del comportamiento para los datos reservados que venga desde el comercio
planQuota	Boolean	campo para poder activar planes y cuota . Nota: 1) N = no está activado. 2) Y = está activado.

FeatureWalletData
Objeto contenedor de la configuración de wallet.

Propiedad	Tipo	Descripción
userCommerce	String	dato poder activar wallet, donde el comercio definirá el valor.

FeaturedReservedData

Nombre del campo reservado y valor asignado para el comercio.

Propiedad	Tipo	Descripción
name	String	Nombre del campo reservado. **Obligatorio**
value	String	Valor del campo reservado. **Obligatorio**

OperationData
Contiene los datos de la operación con la cual se procesará la transacción.

Propiedad	Tipo	Descripción
operationNumber	String	Número de operación único en la pasarela, el cual será utilizado para realizar la transacción.
amount	String	Monto a cobrar por la pasarela de pago, se debe considerar punto como separador de decimales y máximo de dos dígitos decimales (#0.00) **Obligatorio**
productDescription	String	Descripción del producto por el cual se realiza la transacción. **Obligatorio**
currency	CurrencyData	Información de la moneda con la que se realizará la transacción. **Obligatorio**

CurrencyData
Información de la moneda con la cual se realizará a transacción.

Propiedad	Tipo	Descripción
code	String	Código de moneda ISO-4217
symbol	String	Símbolo de moneda con el cual se visualizará el monto en el formulario, no tiene ningún efecto en la transacción. Se recomienda el uso de los símbolos determinados en el ISO-4217

PersonaData
Campos que almacenan datos personales para algún propósito dentro de la pasarela.

Propiedad	Tipo	Descripción

firstName	String	Nombres
lastName	String	Apellidos
email	String	Correo electrónico
address	String	Dirección
zip	String	Código postal
city	String	Ciudad/Provincia
state	String	Estado/Departamento
country	String	País
phone	String	Teléfono

3.2.2) Salida

PaymentData

Respuesta del servidor luego de la ejecución del pago desde el comercio.
Esta respuesta es referencial para el comprador, el comercio recibirá la notificación desde los servidores de Alignet en línea.

Propiedad	Tipo	Descripción
success	Boolean	Tiene valor **true** si se realizó el proceso correctamente, **false** si ocurrió algún error durante el procesamiento del pago.
messageCode	String	Código del mensaje de respuesta
message	String	Mensaje de respuesta
payment	PaymentResponse Data	Resultado del proceso de pago siempre y cuando no ocurra ningún error en el procesamiento de transacción (Propiedad success: true).

PaymentResponseData

Resultado del procesamiento del pago de acuerdo a la marca que corresponda.

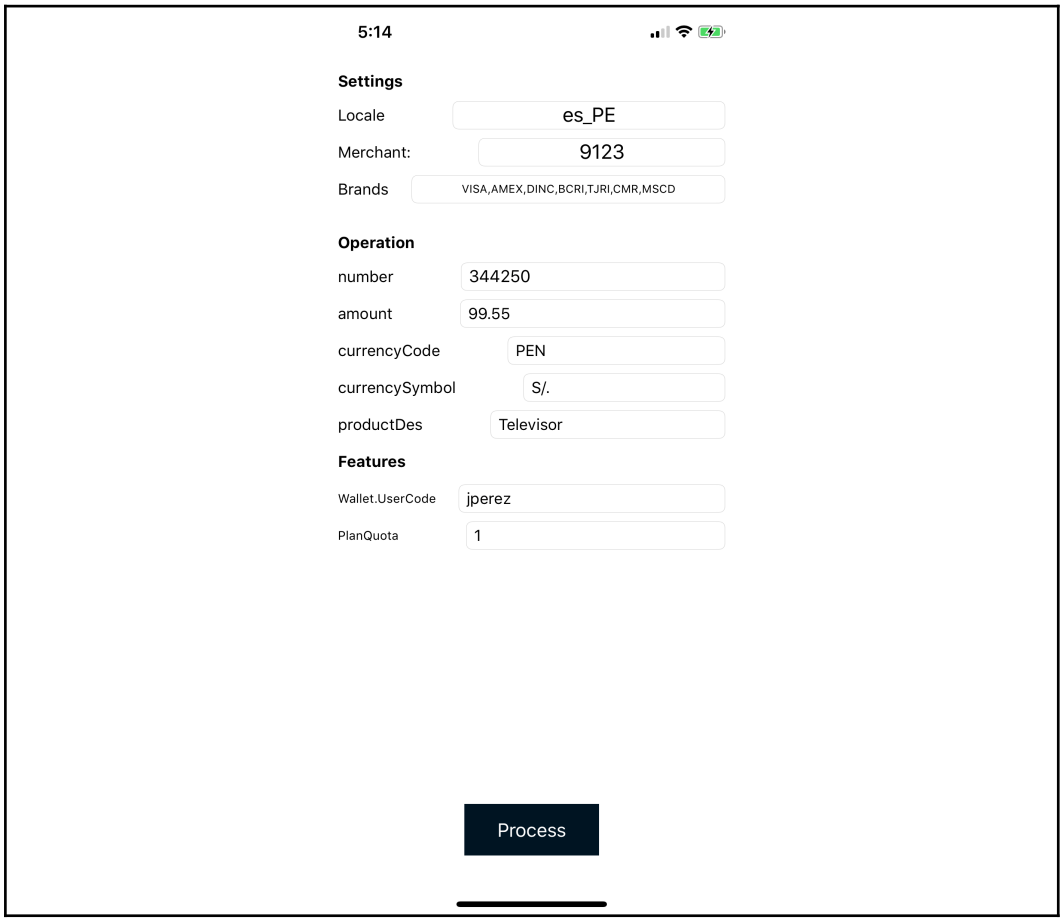
Propiedad	Tipo	Descripción
accepted	Boolean	**true**, cuando el pago ha sido procesado correctamente, **false** si se ha rechazado por alguna razón al momento de realizar la transacción. En caso sea **false** se puede visualizar la razón en los campos **resultCode** y **resultMessage**.
resultCode	String	Código de resultado de procesamiento de pago

resultMessage	String	Mensaje de resultado de procesamiento de pago

4) Consumir la libreria

Para realizar el consumo de la librería, tomaremos como base el proyecto de ejemplo. Lo pueden encontrar en la siguiente url. https://github.com/alignetdev/paymemobile-ios en la carpeta Ejemplo.

A continuación, se muestra la pantalla de un carrito que se construyo con el fin de ingresar datos para consumir la librería en iOS. Cuando revise el ViewController de la pantalla del carrito podrá identificar los métodos delegados que requiere la librería para llenar la entidad principal Merchant para iniciar el consumo de servicios que requiere la librería, así como del método delegado para recibir la respuesta al momento de efectuar el pago.



Debemos hacer el import de la librería y seguidamente implementar los métodos delegados que esta solicita.

```
//
6
7  import UIKit
8  import PaymeMobile
9
```

Luego de agregar los métodos delegados. Se debe proceder a llenar el objeto Merchant, este a su vez se alimenta de otras entidades hijas. Tomar como referencia el proyecto de ejemplo para realizar el llenado correcto por entidad. Tome en consideración que si no envía los parámetros de forma completa, la librería le responderá con un mensaje de error. Hasta que este requisito no este terminado no podrá interactuar con la librería.

```
// MARK: - PaymeProtocol delegate method
extension ViewController: PaymeProtocol {

    func setParamsMerchant() -> ModelMerchant { ••• }

    func getResponsePay(response: ModelPayment?) { ••• }

}
```

El método **setParamsMerchant** se envía como parámetro de entrada para los múltiples servicios que utiliza la librería, por lo que es necesario enviarlo tal como lo hace la aplicación de ejemplo. Para mayor detalle de los campos que se están enviando, consulte la sección 3 del presente documento.

Para empezar a invocar la librería debemos colocar este código en una función o IBAction.

```
let sb = UIStoryboard (name: "MainPaymeApi", bundle: Bundle(for: PayController.self))
let vcStartPayment = sb.instantiateViewController(withIdentifier: "PayController") as! PayController
vcStartPayment.paymeProtocol = self

// self.present(vcStartPayment, animated: true, completion: nil)
self.navigationController?.pushViewController(vcStartPayment, animated: true)
```

El método **getResponsePay** retorna la respuesta al momento de realizar el pago. El comercio puede especificar la lógica que desee mostrar con su app al momento de obtener la respuesta. Por ejemplo, invocar otro flujo, pantalla o retornar a la pantalla inicial.