Citcon Pay iOS SDK Documentation

Version 2.4.1

I.   Introduction

Citcon's iOS development SDK was designed for online merchants to integrate Citcon payment solutions effortlessly into their own iOS apps. By using the SDK, merchant developers can focus on business logics without having to understand the plumbing of payment transactions. The payment experience will be totally transparent and seamless to end consumers. This version of the SDK supports payments through Alipay, AlipayHK, WeChat Pay, Union Pay, KakaoPay, Dana.

This version of the SDK targets iOS 8.0 and above. Because it is distributed as a standard framework and therefore language-independent, merchant developers can program in either Objective-C or Swift to integrate with it.

II.   The Citcon Pay Framework

This section details the main components of the Citcon Pay Framework for iOS development. The commonly used header files and their purposes will be listed here, and A step-by-step example of integrating the framework in a demo merchant app will be shown in the next section.

1. Framework identity

    a. Name: CPay

    b. Bundle identifier: com.cpay.cpay

    c. Current version: 2.4.1

2. Required frameworks or libraries
   NA

3. Class references

    a. **CPayManager :NSObject**

    The CPayManager class performs the most common payment related tasks: set up merchant token, send order to Citcon Pay and query the status of a specific transaction.

    Class methods:

Name	Description

+ setupTokenKey	Set up merchant token (provided by Citcon)
+ setupMode	Set up SDK running environment, default CPayModePROD
+ requestOrder	Sends order information to Citcon and initiates the payment transaction. It is recommended to check the order result (CPayOrderResult) in the callback function (completion).
+ inquireResult	Inquire about the status of a payment transaction
inqureResultByRef	Query by reference id , currency and vendor
+ processOpenUrl	Handle the callback function of payment vendor
+ processUserActivity	Handle the callback function of payment vendor
+ isWechatInstalled	Check if WeChat is installed
+ isAlipayInstalled	Check if Alipay is installed
+ isAlipayHKInstalled	Check if AlipayHK is installed
+ isKakaoPayInstalled	Check if Kakao pay is installed
+ isUnionPayInstalled	Check if Union pay is installed
+ getVersion	Return SDK version

b. CPayOrder: NSObject

Represents the order and payment information merchant wants to send to Citcon Pay for processing.

Note: The currency parameter cannot be empty. If you want to use overseas RMB to pay, please set currency to USD and set trans_currency to CNY.

Property summary

Name	Type	Description	Example
referenceId	NSString	A reference merchant creates and assigns to the transaction.	123xyz
amount	NSString	Total charge amount of the transaction in cents	245, e.g. $2.45
currency	NSString	Settlement currency type defined by three-letter code	USD or CAD or CNY
transCurrency	NSString	Transaction currency type defined by three-letter code	USD or CAD or CNY
vendor	NSString	The name of vendor that will process the payment	alipayor ,wechatpay, alipay_hk, kakaopay, dana, upop
subject	NSString	A customer-defined description of the transaction	Gift for Mom
body	NSString	A more detailed customer-defined description of the transaction	A Blu-ray player and a few great movies
scheme	NSString	Your app scheme for alipay payment, set in the Info.-plist.	cpaydemo.citconpay.com

univer-salLink	NSString	The URL you registered on the WeChat Pay open plat-form.	https://www.xyz.com/apps/
ipnUrl	NSString	The URL of a page Citcon Pay should post transaction status to. Normally this should be a page on the merchant's website.	http://www.xyz.com/noti-fy.php
callback-Url	NSString	The URL of a page Citcon Pay should redirect cus-tomer to after the payment transaction has completed. Normally this should be a page on the merchant's website.	http://xyz.com/confirm.php
allow-Dupli-cate	BOOL	Flag to control duplicate orders.	YES/NO
transac-tionId	NSString	An unique identifier of the transaction. This ID is gen-erated by Citcon Pay.	123456789
ext	hash-Map	Customized key, value <String, String>	"reference2", "123456789"

c. CPayOrderResult :NSObject

Holds the status and message for a transaction returned by Citcon Pay service. An instance of the CPayOrderResult class can be inspected in the callback handler of the requestOrder method of CPayManager.

Property summary

Name	Type	Description	Example
message	NSString	Detailed description of the status of a transaction	Transaction succeeded
result	NSString	The result of the transaction	success
resultStatus	NSInteger	The status code for the result	0 (refer to 4.Status Code)
order	CPayOrder	Order object containg order details	

d. CPayCheckResult :NSObject

An instance of the CPayCheckResult class holds the details of a payment transaction's state at any time the inquireResult method of CPayManager is called. A common practice is to inspect the properties in the callback handler of inquireResult.

Property summary

Name	Type	Description	Example
transactionId	NSString	An unique identifier of the transaction. This ID is generated by Citcon Pay.	123456679
referenceId	NSString	A reference identifying the transaction. This ID is generated by merchant	Abc123
type	NSString	The type of the transaction: "charge" or "refund"	charge
amount	NSString	The total amount of the transaction in cents	245, e.g. $2.45
currency	NSString	Settlement currency type defined by three-letter code	USD
transCurrency	NSString	Transaction currency type defined by three-letter code	USD

time	NSString	The timestamp for the transaction	2017/06/22 1:23:12 PM
status	NSString	The status of the transaction	success
refundedAmount	NSString	The actual amount in centsrefunded to the customer, which is may or may not be the same as the charge amount	179, e.g. $1.79
refundStatus	NSString	The status of a "re-fund" transaction	success
note	NSString	Additional informa-tion for the transac-tion	

4. Status code

Status Code	Description

0	Success
-1	Bad signature, unregistered AppId, wrong AppId, mismatched AppId or some other error.
-2	The customer cancels the payment, switches back to the app, the customer cannot pay, and so on.
8000	Order is being processed
4000	Payment failed
5000	Duplicate request
6001	The customer cancels the payment
6002	Network connection error
6004	Unknown payment result
others	Other payment error

5. Notification

After the payment is completed, the Citcon SDK will automatically query immediately to ensure that the correct results are obtained. And post a notification as soon as the automated query is complete. Therefore, you can also add observers to receive notifications. The notification is kOrderPaymentFinishedNotification.

III.   Example

In this section, a demo merchant app making Alipay payments through Citcon Pay is demonstrated step-by-step using Objective-C. The source code of this demo app is also provided as part of the SDK distribution package.

1. Create a new Xcode project → Choose the iOS template → Choose Single View Application --> Choose options for the project.

2. Modify the configurations of your project as the following steps:

   2.1 Add the following KeyValues to the info.plist of your project
   <key>LSApplicationQueriesSchemes</key>

   <array>

   <string>weixin</string>

   <string>wechat</string>

   <string>weixinULAPI</string>

   <string>alipay</string>

   <string>alipayhk</string>

   <string>safepay</string>

   <string>uppaywallet</string>

   <string>uppaysdk</string>

   <string>uppayx1</string>

   <string>uppayx2</string>

   <string>uppayx3</string>

   <string>kakaokompassauth</string>

   <string>kakaolink</string>

   <string>cash</string>
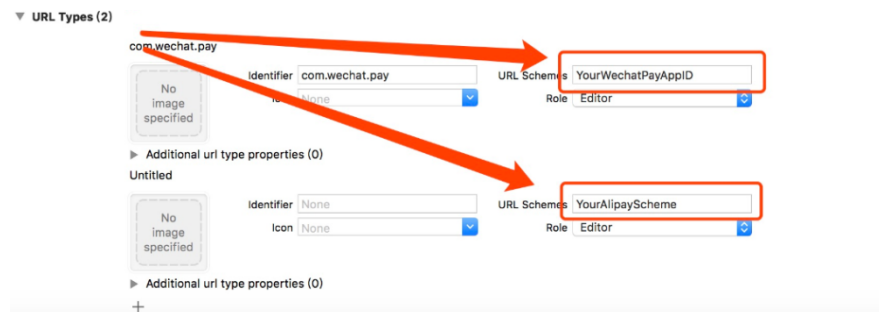
```
</array>
<key>NSAppTransportSecurity</key>
<dict>
        <key>NSAllowsArbitraryLoads</key>
        <true/>
</dict>
```

This allows WeChat or Alipay to return to your mobile app once payment is finished.

2.2 Add the URL Types to the configuration of your project

Make sure the value of URL Schemes of WechatPayAppID is the same as your Wechat pay AppID

and the value of URL Schemes of YourAlipayScheme is the same as the field ' scheme' of the CPayOder which you want to send to Citcon Pay for processing.

3. Set up universal link

3.1 Turn on the "Associated Domains" option of your AppID in Apple Developer Center.
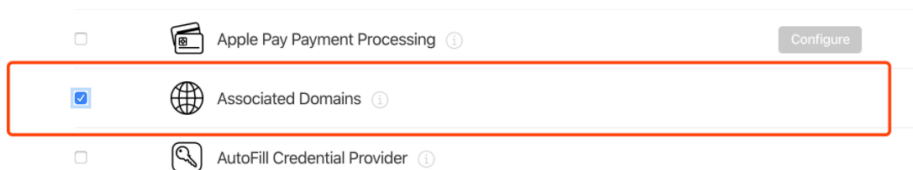
3.2 Edit and reinstall the profile which associated with the AppID that you turn on "Associated Domains" .

3.3 Create a json file named "apple-app-site-association" and upload the file to the webroot directory or to the ".well-known" directory inside the the webroot of your website that supports HTTPS. **NOTE**: Don't append .json to the apple-app-site-association filename.
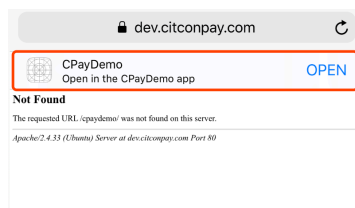
```
{
 "webcredentials": {
       "apps": [    "TeamId.BundleID" ]
  },
  "applinks":{
    "apps":[],
    "details":[
        {
          "appID": "TeamId.BundleId",
          "paths":[ "NOT /xxx/*",
                    "/apps/*" ]
        }
      ]
    }
  }
```

3.4 Add "applinks:yourdomain" to your Xcode project. Set the URL to the location where you uploaded the file "apple-app-site-association" without https.

▼ 🌐 **Associated Domains**   ✕

Domains applinks:dev.citconpay.com
         webcredentials:dev.citconpay.com

3.5 Enter the URL set by your applinks + paths ("apple-app-site-association") to ensure that you can navigate to your app from safari.

🔒 dev.citconpay.com   ⟳

CPayDemo   OPEN
Open in the CPayDemo app

Not Found

The requested URL /cpaydemo/ was not found on this server.

Apache/2.4.33 (Ubuntu) Server at dev.citconpay.com Port 80

4. Import the CPay.framwork.

    a. Embed local framework file

              or

    b.   CocoaPods: Pod 'CPay', ''

5. Add Run Script Phase in Build Phases as below
   Shell: /bin/sh
   bash "./Frameworks/CPay.framework/strip-frameworks.sh"

   Important: This step will allow the merchant app to embed the framework for development on both simulator and actual device.

6. Set up merchant token and SDK running environment in appDelegate

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

[CPayManager setupTokenKey:@"abc1234"];

[CPayManager setupMode:CPayModePROD];

//... other initializations
}
```

7. Handle callback in appDelegate

```objc
- (BOOL)application:(UIApplication*)app openURL:(NSURL *)url options:(NSDictionary<UIApplicationOpenURLOptionsKey,id> *)options {
   BOOL processed = [CPayManager processOpenUrl:app url:url standbyCallback:^(CPayOrderResult *result) {
   //optional further processing
   }];

   if (processed) {
   return YES;
   }

   //optional further processing

   return YES;
   }
```

```objc
- (BOOL)application:(UIApplication*)application continueUserActivity:(nonnull NSUserActivity *)userActivity restorationHandler:(nonnull void (^)(NSArray<id<UIUserActivityRestoring>> * _Nullable))restorationHandler {
   NSLog(@"userActivity: %@", userActivity.webpageURL.description);
   return [CPayManager processUserActivity:userActivity];
   }
```

8. Initiate a payment transaction

```objc
CPayOrder * order = [CPayOrder new];
order.referenceId = @"1ZLLJULOCRW3LAU";
```

```objc
order.subject = _@"Test";
order.body = @"This is a test transaction";
order.amount = @"2.00";
order.currency = @"USD";
order.vendor = "alipay"  ;
order.ipnUrl = @"http://www.xyz.com/listen.php";
order.callbackUrl = http://www.xyz.com/confirmation.php;
order.scheme = @"cpaydemo.citconpay.com";
order.universalLink = @"https://dev.citconpay.com/cpaydemo/";

[CPayManager requestOrder:order completion:^(CPayOrderResult * payResult) {

// Interpret the result in callback handler


};
```

(Optional) Inquire status of the transaction and displays the status information in the merchant app

```objc
[CPayManager requestOrder:order completion:^(CPayOrderResult * payResult) {

[CPayManager inquireResult:order.transactionId method:@"real"completion:^(CPayCheckResult *checkResult) {

        NSMutableString * result = [NSMutableString new];

        [result appendFormat:@"- ORDER RESULT\n"];
        [result appendFormat:@"    STATUS: %d\n", (int)payResult.resultStatus];
        [result appendFormat:@"    MESSAGE: %@\n", payResult.message];
        [result appendFormat:@"\n"];
        [result appendFormat:@"- CHECK RESULT\n"];
        [result appendFormat:@"    TRANSACTION_ID: %@\n", checkResult.transactionId];
        [result appendFormat:@"    REFERENCE_ID  : %@\n", checkResult.referenceId];
        [result appendFormat:@"    TYPE: %@  TIME: %@\n", checkResult.type,
checkResult.time];
        [result appendFormat:@"    STATUS: %@  AMOUNT: %@ (%@)\n", checkResult.status,
checkResult.amount, checkResult.currency];
        [result appendFormat:@"    REFUND_STATUS: %@  AMOUNT: %@\n", checkResult.re-
fund_status, checkResult.refunded_amount];
        [result appendFormat:@"    NOTE: %@\n", checkResult.note];

    }];
};
```

9. Add observer to receive notification (optional)

```objc
- (void)completeAutoInquire:(NSNotification *)notification {
```

```objc
        CPayCheckResult *checkResult = (CPayCheckResult *) notification.object;
        // TODO: add your code
    }

    // Add observer before you request order
    [[NSNotificationCenter defaultCenter]addObserver:self selector:@selector(completeAutoInquire:)
name:kOrderPaymentFinishedNotification object:nil];
```