Payeezy

Minimum technical requirements

The iOS mobile SDK requires iOS SDK 6 and XCode 5.1 and above

For more details: https://www.apple.com/apple-pay

Payeezy iOS initialization

Build app to make purchases directly in your app. An easier way to pay within apps

Convenient checkout. On iPhone 6, iPad Air 2, and iPad mini 3, you can also use Apple Pay to pay with a single touch within apps

Steps to integrate code with Payeezy SDK

Step1: Collect card information

Step2: Using secure token to Authorize and Purchase

Step3: Sending token to complete Authorize and Purchase

Getting started

Using GitHub

Clone Payeezy SDK using with HTTPS or Subversion

HTTPS URL: https://github.com/payeezy/payeezy_ios.git

Using clone command: git clone https://github.com/payeezy/payeezy_ios.git

Subversion URL: https://github.com/payeezy/payeezy_ios

Or simply download zip file: https://github.com/payeezy/payeezy_ios/archive/master.zip

Then add the Payeezy library to your Xcode project:

1. In the menubar, click on 'File' then 'Add files to "Project"...'.
2. Select the 'Payeezy' directory in the downloaded repository.
3. Make sure 'Copy items into destination group's folder (if needed)' is checked.
4. Click 'Add'.
5. In your project settings, go to the "Build Phases" tab, and make sure MobileCoreServices.framework , Foundation.framework, Security.framework, SystemConfiguration.framework and PassKit.framework are included in the "Link Binary With Libraries" section. If you're targeting versions of iOS before iOS8, make sure PassKit.framework is listed as "Optional."

Cocoapods Cocoadocs - TO DO

Define merchant Token, Apikey and ApiSecret

```
#define KApiKey    @"test_apikey_bA8hIqpzuAVW6itHqXsXwSl6JtFWPCA0"
#define KApiSecret @"test_apitsecret_YmI4YzA1NmRkZmQzMzA1ZmIZjYzwMWIzZThkMWU2NGRjZmI4OWE5NGRiMzM4NA=="
#define KToken     @"test_merchant_token_fdoa-a480ce8951daa73262734cf102641994c1e55e7cdf4c02b6"
#define KURL       @"https://api-cert.payeezy.com/v1/transactions"
```

Initialize PayeezySDK object with KApiKey, KApiSecret, KToken and KURL

```
// initialize Payeezy object with key,token and secret value
PayeezySDK* myClient = [[PayeezySDK alloc]initWithApiKey:KApiKey apiSecret:KApiSecret merchantToken:KToken url:KURL];
```

Initialize NSDictionary to capture input parameter : this is optional

```
/* */
// credit card info
 NSDictionary* credit_card = @{
                                        @"type":@"visa",
                                       @"cardholder_name":@"Jacob Test",
                                       @"card_number":@"4012000033330026",
                                       @"exp_date":@"0416",
                                       @"cvv":@"123"
                                    };
 // Transaction info
 NSDictionary* transaction_info = @{
                                       @"currencyCode":@"USD",
                                       @"amount":amount,
                                        @"merchantRefForProcessing":@"abc1412096293369"
                                       };
```

Call transaction method and pass the required parameters. For more information on request and response parameters refer to apple doc located @

```
// initialize Payeezy object with key,token and secret value
   PayeezySDK* myClient = [[PayeezySDK alloc]initWithApiKey:KApiKey apiSecret:KApiSecret merchantToken:KToken url:KURL];


// call authorize method with payment/creditcard data
   [myClient submitAuthorizeTransactionWithCreditCardDetails:credit_card[@"type"]
         cardHolderName:credit_card[@"cardholder_name"] cardNumber:credit_card[@"card_number"]
         cardExpirymMonthAndYear:credit_card[@"exp_date"] cardCVV:credit_card[@"cvv"]
         currencyCode:transaction_info[@"currencyCode"] totalAmount:amount
         merchantRefForProcessing:transaction_info[@"merchantRefForProcessing"]
         completion:^(NSDictionary *dict, NSError *error) {

                        NSString *authStatusMessage = nil;

                        // handle error and response
                        if (error == nil)
                        {
                           authStatusMessage = [NSString stringWithFormat:@"Transaction
                           Successful\rType:%@\rTransaction ID:%@\rTransaction Tag:%@\rCorrelation Id:%@\rBank
                           Response Code:%@",
                                    [dict objectForKey:@"transaction_type"],
                                    [dict objectForKey:@"transaction_id"],
                                    [dict objectForKey:@"transaction_tag"],
                                    [dict objectForKey:@"correlation_id"],
                                    [dict objectForKey:@"bank_resp_code"]];
                        }
                        else
                        {
                           authStatusMessage = [NSString stringWithFormat:@"Error was encountered processing
                           transaction: %@", error.debugDescription];
                        }
```

Transaction Data requirements explanation –

https://developer.payeezy.com/payeezy-api-reference/apis

Submitting/Generating transaction (request properties) with example

https://developer.payeezy.com/payeezy-api-reference/apis

Security related (like: HMAC, Token generation) with example

GENERATE HMAC

Construct the data param by appending the parameters below in the same order as shown. a. apikey - API key of the developer. b. nonce - secure random number. c. timestamp - epoch timestamp in milliseconds. d. token - Merchant Token. e. payload - Actual body content passed as post request. ***Compute HMAC SHA256 hash on the above data param using the key below*** f. apiSecret - Consumer Secret token for the given api key ***Calculate the base64 of the hash which would be our required Authorization header value.***

```
// import required header files
#import <CommonCrypto/CommonDigest.h>;
#import <CommonCrypto/CommonHMAC.h>;

NSString* apiKey = @"<your api key>";
NSString* apiSecret = @"<your consumer secret>";
NSString* token = @"<Merchant Token>";
NSString* payload = @"<For POST - Request body / For GET - empty string>";
NSString* nonce = @"<Crypographically strong random number>";
NSString* timestamp = @"<Epoch timestamp in milli seconds>";
NSString *hmacData =
[NSString stringWithFormat:@"%@%@%@%@%@",apiKey,nonce,timestamp,token,payload];

// Make sure the HMAC hash is in hex
unsigned char outputHMAC[CC_SHA256_DIGEST_LENGTH];
const char* keyChar = [apiSecret cStringUsingEncoding:NSUTF8StringEncoding];
const char* dataChar = [hmacData cStringUsingEncoding:NSUTF8StringEncoding];
CCHmac(kCCHmacAlgSHA256, keyChar, strlen(keyChar), dataChar, strlen(dataChar), outputHMAC);
NSData* hmacHash =
[[NSData alloc] initWithBytes:outputHMAC length:sizeof(outputHMAC)];

// The conversion (NSData to HexString) shown below is just an example.Please use other appropriate methods to do the
conversion.
NSString* hmacHashHexString =
[[hmacHash description] stringByReplacingOccurrencesOfString:@" " withString:@""];


// Authorization : base64 of hmac hash -->
NSString* authorization =
[[hmacHashHexString dataUsingEncoding:NSUTF8StringEncoding] base64EncodedStringWithOptions:0];
NSLog(@"Authorization Generated: %@",authorization);
```

First Data

Sample codes in various languages (with example)

For Java and link github

b›yond the transaction℠