



秒支付 **S-Pay** 开发手册

支持 iOS, Android

v2.2

BeeCloud 文档: <http://beecloud.cn/doc>

联系我们: support@beecloud.cn

更新日期: 2/14/2015

目录

前期准备	4
第一波 微信支付	4
注册微信开放平台	4
申请开发者资质认证	4
创建微信应用	5
申请微信支付功能	6
在控制台填写微信支付要素	7
第二波 支付宝支付	8
签约移动支付产品	8
查看支付要素并添加商户密钥	10
在控制台上传支付宝支付要素	11
第三波 银联在线支付	11
在控制台上传银联支付要素。	13
开始集成	13
秒支付	13
注册	13
iOS SDK	14
下载安装 SDK	14
全局初始化	17
微信支付	17
微信支付功能	18
微信退款功能	19
支付宝支付	21
支付宝支付功能	21
支付宝退款功能	23
银联支付	23
银联支付功能	23

银联退款功能.....	24
内置购买.....	25
初始化产品列表.....	26
购买产品	27
恢复购买	28
查询订单.....	28
Android SDK	31
下载安装 SDK.....	31
配置工程.....	31
全局初始化.....	33
微信支付.....	33
微信支付功能.....	33
微信退款功能.....	34
支付宝支付.....	35
支付宝支付功能.....	35
支付宝退款功能.....	36
查询订单.....	37

前期准备

第一波 微信支付

申请应用的微信支付功能需要申请并通过微信开放平台的开发者资质认证，申请开发者资质认证的主体必须属于下图中某一类，且为保证快速通过认证，请按照要求如实填写开发者信息。



The screenshot shows the 'WeChat Open Platform' (微信·开放平台) registration interface. At the top, there's a progress bar with four steps: 1. 同意协议 (Agree to terms), 2. 填写资料 (Fill in information) - which is the current step and highlighted in green, 3. 填写发票 (Fill in invoice), and 4. 支付费用 (Pay fee). Below the progress bar, there's a section titled '类型 (请从下列类型中选择一项)' (Type (Please select one from the following types)). It lists several categories with radio buttons: 企业 (Company), 网店商家 (Online store merchant), 媒体 (Media), 政府及事业单位 (Government and public institutions), and 其他组织 (Other organizations). The '其他组织' category is selected. Under '其他组织', there are four sub-options: 其他组织-免费类型 (基金会、国外政府机构驻华代表处) (Other organizations - free type (Foundation, foreign government institutions in China)), 社会团体 (组织机构代码证上机构类型为社团法人) (Social groups (Institutional code certificate type is social group legal person)), 民办非企业 (组织机构代码证上机构类型为民办非企业) (Private non-enterprise (Institutional code certificate type is private non-enterprise)), and 其他组织 (Other organizations). The first sub-option is selected.

[微信 APP 支付接入商户服务中心](#)

[微信支付 App 支付 iOS 开发文档](#)

[微信支付 App 支付 Android 开发文档](#)

注册微信开放平台

注册地址: <https://open.weixin.qq.com>

申请开发者资质认证



基本资料	验证状态 未认证 ! 查看订单 现在申请
开发者资质认证	介绍 <ul style="list-style-type: none">• 微信开放平台帐号的开发者资质认证提供更安全、更严格的真实性认证、也能够更好的保护企业及用户的合法权益• 开发者资质认证通过后，微信开放平台帐号下的应用，将获得微信登录、智能接口、公众号第三方平台开发等高级能力• 认证有效期：一年，有效期最后两个月可申请年审即可续期• 审核费用：300元
修改密码	

认证通过的状态如下：

基本资料	验证状态 已认证 ✓ 查看订单
开发者资质认证	介绍 <ul style="list-style-type: none">• 微信开放平台帐号的开发者资质认证提供更安全、更严格的真实性认证、也能够更好的保护企业及用户的合法权益• 开发者资质认证通过后，微信开放平台帐号下的应用，将获得微信登录、智能接口、公众号第三方平台开发等高级能力• 认证有效期：一年，有效期最后两个月可申请年审即可续期• 审核费用：300元
修改密码	

创建微信应用

- 填写应用的基本信息，上传应用图标(28x28，108x108)。
- 填写平台信息。选择应用平台，填写各平台对应的应用信息。
- 提交审核，等待审核结果，预计 7 个工作日。

审核通过会获得 AppID 和 AppSecret。

创建移动应用 还可创建10个移动应用


应用名称	状态	操作
------	----	----

1 填写基本信息 2 填写平台信息 3 提交成功

✓ 您的应用已成功提交审核，预计在7天内完成审核
审核通过后，你可以获得AppID和AppSecret，来进行开发

查看管理中心

创建移动应用 还可创建9个移动应用

应用名称	状态	操作
 比克	审核中	查看

申请微信支付功能

点击“查看”，进入应用详情页，申请开通微信支付、需要填写应用基本资料、企业审核资料、财务审核资料，三者分别审核，全部审核通过后，微信官方

会通过邮件的方式,发送微信支付需要的要素。注:按照要求如实填写资料,有助于开速通过审核。



申请通过后获得 partnerID、appkey、支付专用签名串 paySignKey、初始密钥 partnerKey、安全证书(.pfx)。

在[控制台](#)填写微信支付要素



字段说明:

微信管理 ID	财付通商户后台设置的操作员 ID。退款必须
微信管理员密码	财付通商户后台设置的操作员密码。退款必须
微信 PartnerID	商家 ID
微信 PartnerKEY	商家私钥
微信 APPKEY	支付签名密钥

微信 APPID	微信开放平台申请的 APP ID
微信 Secret	微信开放平台申请的 APP Secret
微信证书密码	安全证书密码，退款必须。微信支付审核通过后，通过短信的方式发送到联系人的手机上，务必妥善保管，不能丢失。
微信证书	安全证书。微信支付审核通过后，微信发送的邮件附件就是安全证书。退款必须

第二波 支付宝支付

签约移动支付产品

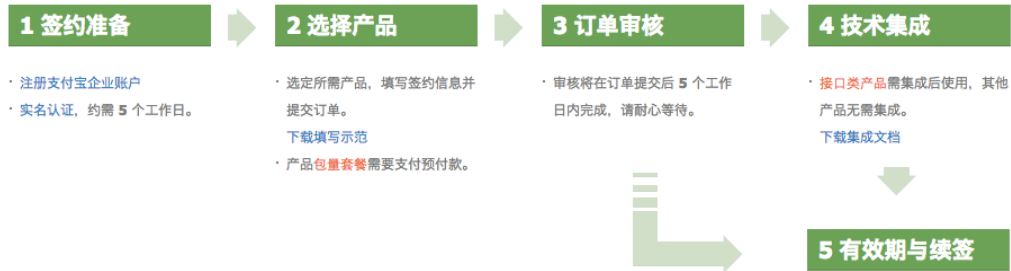
应用集成支付宝快捷支付(无线)，需要在

<https://app.alipay.com/container/web/enterpriseIndex.htm> 签约支付宝的移动支付产品。



<https://b.alipay.com/order/appInfo.htm?salesPlanCode=2014110308141993&channel=ent> 可以查看具体的签约流程

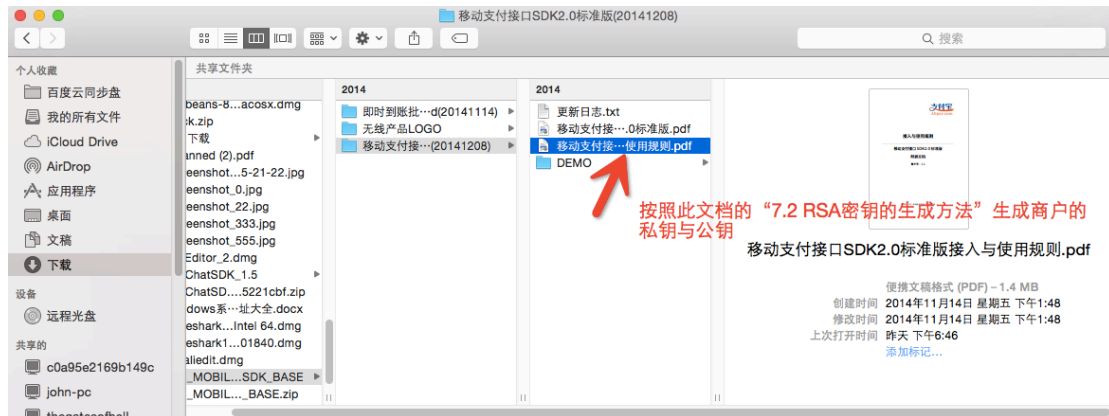
签约流程



按照支付宝官方签约流程自助签约, 审核通过后登录支付宝商户版

<https://b.alipay.com/newIndex.htm>->签约管理标签查看会获得支付要素。

点击[支付宝移动支付开发文档](#), 下载技术集成文档。



查看上图中箭头导向的文档的相关章节可以按步骤生成商户密钥与公钥, 需要上传商户公钥给支付宝, 用于支付宝验证签名。上传商户公钥会获得支付宝的公钥, 用于验证支付宝返回数据的合法性。

查看支付要素并添加商户密钥



点击上图中“查看PID|Key”，输入支付密码，可以查看PID等要素。



添加公钥成功，点击“查看支付宝公钥”可以获取支付宝公钥。

在[控制台](#)上传支付宝支付要素

beecloud_der 切换

应用数据

应用分析

导入数据

支付管理

设置

点击隐藏应用菜单

应用信息

BC File

BC Config

微信支付

支付宝

银联支付

支付宝设置

提示:光标焦点移出输入框,内容将自动保存

支付宝Partner ID

PID

支付宝商家ID

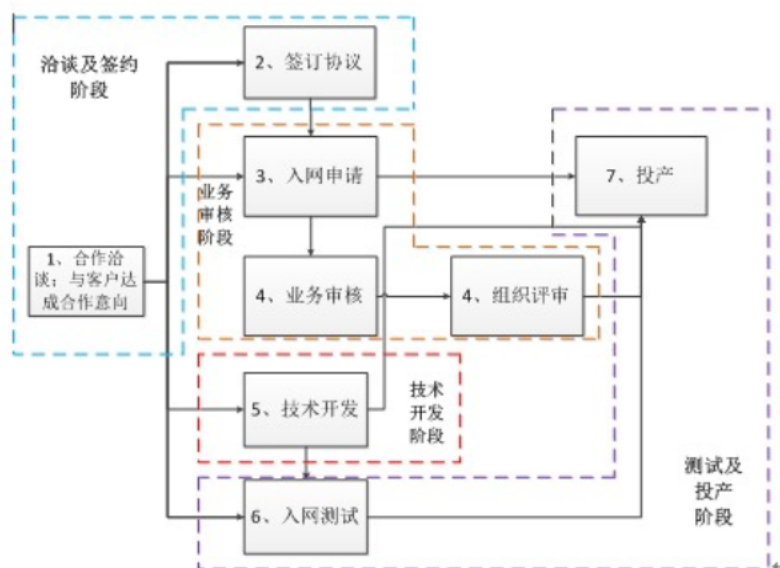
支付宝私钥

MIICdgIBADANBgkqhkiG9w0BAQE
nl+ovCGzafYlhb17hV3m1rxq4U4v
guFhXGcDFeDITa0euhEDM/aukPw

通过openssl工具生成的RSA商户密钥，不能有空格，制表符，换行

第三波 银联在线支付

银联在线支付的接入流程大致如下：



环节说明：

1、合作洽谈，与客户达成入网意向：包括客户选择、确定接入银联业务或产品、形成业务方案。

2、签订协议：客户与银联签订合作协议、入网协议、清算协议，提供代理清算协议等。

3、入网申请：客户以正式方式确认开通的业务信息和要素，以及提供证明客户具备开通该业务相应资质的材料。

4、业务审核：针对某些特定业务，对客户是否具备开展相应业务资质、业务风险是否可接受以及分配客户开展银联业务的特定身份标识、权限的工作。

5、技术开发：包括银联为了支持客户相应业务而进行的开发和开发阶段测试，以及客户为实现与银联系统对接而进行的开发和测试工作。

6、入网测试：在正式投产前，为验证客户和银联系统应用及技术、业务参数配置的正确性、可用性，降低生产系统运营风险而进行的测试。

7、投产：包括支持客户业务的生产系统对接成功并技术上线、在系统中业务参数配置生效。

以上流程环节根据实际需要不同，可进行选择配置。一般情况下，2、3、4、5、6工作可以并行。

注：入网流程问题发送邮件至：operation@unionpay.com进行业务申请和咨询，银联有专门的人员进行处理。联系电话：021-50362408。在提交入网申请，进入业务审核阶段的时候，银联的技术客服会将您加入到客服全中，方便您及时地解决遇到的问题。审核通过后会收到“商户入网通知参数信息（请注意保密）”

--（公司名称）”标题的邮件，邮件中包含了一些商户信息，具体内容在邮件的附件中。

在[控制台](#)上传银联支付要素。

开始集成

秒支付

秒支付主要实现有关支付方式的相关方法的实现，当前主要支持内置购买、微信支付、支付宝支付等.

注册

两个步骤，2分钟轻松搞定：1. 注册开发者：猛击 [这里](#) 注册成为BeeCloud开发者。

2. 注册应用：使用注册的账号登陆 [控制台](#) 后，点击"新应用+"创建新应用。

iOS SDK

下载安装 SDK

方法一：

1. 下载SDK：猛击 [这里](#) 下载最新版的BeeCloud SDK（内含BeeCloud.framework和AlipaySDK.framework）。
2. 安装SDK：将BeeCloud.framework和AlipaySDK.framework增加到XCode的项目中（参考[这里](#)）
3. 添加系统framework依赖：
 - CoreGraphics.framework
 - CoreLocation.framework
 - MobileCoreServices.framework
 - Security.framework
 - SystemConfiguration.framework
 - UIKit.framework
 - CoreTelephony.framework
4. 添加系统lib依赖：
 - libc++.dylib
 - libz.dylib
 - libsqlite3.dylib

方法二（一键配置BeeCloud API）：

对于熟悉CocoaPods的同学，使用如下命令：

```
pod 'BeeCloud' , :git =>
```

```
'https://github.com/beecloud/ios-sdk-cocoapods-release.git'
```

对初次使用CocoaPods的同学：

1.安装CocoaPods（已安装请跳过）

在终端输入

```
sudo gem install cocoapods
```

如果安装成功，会有一个提示

Successfully installed cocoaPods

若很久没有反应，则是因为安装被墙阻拦

解决办法1：打开vpn下载

解决办法2：请查看详细指南

<http://code4app.com/article/cocoapods-install-usage>

2.搜索BeeCloud库

```
pod search BeeCloud
```

若无返回结果，则先运行

```
pod repo update
```

然后再进行搜索

3.新建一个Xcode工程

4.使用CocoaPods

在当前工程（.xcodeproj）所在文件夹下，打开terminal

创建Podfile：

```
touch Podfile
```

编辑Podfile内容如下：

```
platform :ios, '7.0'
```

```
pod 'BeeCloud' , :git =>
```

```
'https://github.com/beecloud/ios-sdk-cocoapods-release.git'
```

在Podfile所在文件夹输入命令：

```
pod install
```

若已经install过，使用命令：

```
pod update
```

来更新版本

成功以后，会出现如下记录：

```
localhost:yourWorkDir yourUserName$ pod install
```

```
Analyzing dependencies
```

```
Downloading dependencies
```

```
Installing BeeCloud(2.2)
```

```
Generating Pods project
```

```
Integrating client project!
```

```
[!] Form now on user 'yourProj.xcworkspace'.
```

打开workspace， 就可以工作了

配置完成，更多CocoaPods相关设置，请见官方文档

<http://guides.cocoapods.org/>

全局初始化

首先，在需要使用BeeCloud API的地方，import头文件：

```
#import <BeeCloud/BeeCloud.h>
```

其次，在APP启动的时候（在applicationDidFinishLaunching函数里），初始化BeeCloud需要的App Key， 全局只需调用一次。

每个应用对应一个App Key，可在控制台中相应App的 基本信息 里获得：

```
[BeeCloud initWithAppID:@"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"
andAppSecret:@"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"];
```

BeeCloud所有的网络请求默认超时为5秒，如果需要更改（如改为10秒），请调用以下函数，全局只需调用一次：

```
[BeeCloud setNetworkTimeout:10.0];
```

微信支付

BeeCloud提供了快速集成微信支付的服务，其方法主要在 - BCPay。

开发者只需调用简单的API就可以在应用中快速集成微信支付功能。

在使用接口之前需要做以下准备工作：1、在微信开发平台注册应用。2、开发者资质认证。3、申请开通应用的支付功能。

注：由于支付功能的相关申请需要15个工作日,建议在应用开发准备阶段完成微信支付功能的相关申请。

具体参考[微信开放平台](#)

微信支付功能申请完成后，按以下步骤“秒”集成微信支付：

第一步:在网页控制台界面填写微信支付相关信息，包括微信开放平台创建应用获得的appid、appSecret，以及申请微信支付功能获得的partnerID、partnerKey、appKey。

第二步:点击项目名称,点击“Info”选项卡,在“URL Types”选项中,点击“+”，在“URL Schemes”中输入微信appid,Role = “Editor”,Identifier自定义。

第三步:在AppDelegate.m 文件下添加如下代码：

```
- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url sourceApplication:(NSString *)sourceApplication annotation:(id)annotation {
    return [BCPay handleOpenUrl:url withBlock:^(BOOL success, NSString *strMsg, NSError *error) {
        NSLog(@"strMsg = %@", strMsg);
    }];
}
```

微信支付功能

添加下列代码发起微信支付

```
// @"自制白开水"为:商品描述
// totalFee为:支付金额*以分为单位*
```

```

// outTradeNo为需要您自行生成的订单号,*32个字符内、包含数字与字母,确保在
商户系统中唯一*可根据[[BCUtil generateRandomUUID]
stringByReplacingOccurrencesOfString:@"-" withString:@""]获取;
// traceid为:支付用户ID,必须保证在商户系统中的唯一性
// block支付结果回调
// 通过该方法实现对各种参数的初始化,然后发起微信支付,并跳转到微信.
- (void)wxPay {
    NSString *outTradeNo = [[BCUtil generateRandomUUID]
stringByReplacingOccurrencesOfString:@"-" withString:@""];
    [BCPay reqWXPayment:@"自制白开水" totalFee:@"1"
outTradeNo:outTradeNo traceID:@"BeeCloud" payBlock:^(BOOL
success, NSString *strMsg, NSError *error) {
        if (success) {
            // 表明微信支付成功
        } else {
            // 表明支付过程中出现错误, strMsg为错误原因
        }
        NSLog(@"%s strMsg = %@", __func__, strMsg);
    }];
}

```

在支付成功后会在云端存放购买记录。

微信退款功能

需要申请退款时, 可以调用退款接口来实现.在调用之前需要在云端先配置三个

参数:

1.operatorID

2.安全证书（在商户审核通过后，收到的财付通邮件中有附件以商户号

_XXXX.pfx文件即为安全证书）

3.operatorPassword

在客户端用户通过以下代码发起退款请求：

```
// outTradeNo:用户要退订的订单号
// outRefundNo:您自行生成的退订订单号,*32个字符内、包含数字与字母,确保在
商户系统中唯一*可根据[[BCUtil generateRandomUUID]
 stringByReplacingOccurrencesOfString:@"-" withString:@""]获取;
// refundReason:退款原因
// refundFee:退款金额
// payBlock:发起退款结果回调, 用户在app发起退款请求成功后, 商户会收到退
款请求, 商户可在后台确认, 完成退款操作

- (void)wxRefund:(NSString*)outTradeNo {
    NSString *outRefundNo = [[BCUtil generateRandomUUID]
 stringByReplacingOccurrencesOfString:@"-" withString:@""];
    NSLog(@"refund no = %@", outRefundNo);
    [BCPay reqWXRefund:outTradeNo outRefundNo:outRefundNo
 refundReason:@"不好喝" refundFee:@"1" payBlock:^(BOOL success,
 NSString *strMsg, NSError *error) {
        if (success) {
            // 退款成功
        } else {
            // 退款失败, strMsg为错误原因
        }
        NSLog(@"%s,%s,%d,%@", __FILE__, __func__, __LINE__,
error);
    }];
}
```

在退款成功后，会在云端生成退款记录.

支付宝支付

当用户在使用BeeCloudSDK进行支付宝支付时，请注意以下几点:

第一步:在网页控制台填写支付宝partnerID (PID) ,支付宝卖家ID (邮箱或者手机号) , 支付宝商家私钥(RSA私钥), 私钥的生成方法见[官方文档](#).

第二步:点击项目名称，点击“Info”选项卡，在“URL Types”选项中，点击“+”，在“URL Schemes”中输入应用标识，如“BCPayDemo”.

第三步:在AppDelegate.m 文件下添加如下代码:

```
- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url sourceApplication:(NSString *)sourceApplication annotation:(id)annotation {
    return [BCPay handleOpenUrl:url withBlock:^(BOOL success, NSString *strMsg, NSError *error) {
        NSLog(@"strMsg = %@", strMsg);
    }];
}
```

支付宝支付功能

当用户确定选择用支付宝进行支付时，可以调用以下接口进行集成支付宝支付.

```

// @param trace_id      支付用户ID，必须保证在商户系统中唯一。可通过
trace_id查询订单详情。

// @param out_trade_no  商户系统内部的支付订单号，包含数字与字母，确保在
商户系统中唯一

// @param subject      商品的标题/交易标题/订单标题/订单关键字等。该参
数最长为 128 个汉字

// @param body          对一笔交易的具体描述信息。如果是多种商品，请将商品
描述字符串累加传给body

// @param total_fee     该笔订单的资金总额，单位为RMB-Yuan。取值范围为
[0.01,100000000.00]，精确到小数点后两位

// @param scheme        调用支付的app注册在info.plist中的scheme

// @param block          支付结果回调

- (void)aliPay {
    NSString *outTradeNo = [[BCUtil generateRandomUUID]
stringByReplacingOccurrencesOfString:@"-" withString:@""];
    [BCPay reqAliPayment:@"BeeCloud" outTradeNo:outTradeNo
subject:@"自制白开水" body:@"BeeCloud 自制白开水" totalFee:@"0.01"
scheme:@"payTestDemo" payBlock:^(BOOL success, NSString *strMsg,
NSError *error) {
        if (success) {
            // 表明支付宝支付成功
        } else {
            // 表明支付过程中出现错误，strMsg为错误原因
        }
        NSLog(@"AliPay strMsg = %@", strMsg);
    }];
}

```

支付宝退款功能

当需要进行支付宝退款时，调用以下代码：

// 根据out_trade_no, refund_no, refund_reason, refund_fee查询订单是否可退款，允许退款情况下自动生成预退款订单，否则返回不可退款原因。预退款订单生成成功后，在BeeCloud商户后台对预退款订单进行处理。（订单状态trade_status=@”TRADE_SUCCESS“）时支持退款，其他状态下不支持退款。

// @param out_trade_no 商户系统内部的支付订单号,包含数字与字母,确保在商户系统中唯一

// @param refund_no 格式为:退款日期(8位)+流水号(3~8位)。不可重复,且退款日期必须是当天日期(年月日)。

// 流水号可以接受数字或英文字符,建议使用数字,但不可接受“000”。例如: 201101120001

// @param refund_fee 退款金额

// @param refund_reason 退款原因

// @param block 退款结果回调

```
[BCPay reqAliRefund:out_trade_no refundNo:dateString
refundFee:@"0.01" refundReason:@"不好喝" refundBlock:^(BOOL
success, NSString, *strMsg, NSError *error) {
    NSLog(@"AliRefund strMsg = %@", strMsg);
}];
```

银联支付

银联支付功能

当用户确定选择用银联进行支付时，可以调用以下接口进行集成银联支付。

```

// @param trace_id      支付用户ID, 必须保证在商户系统中唯一. 可通过
trace_id查询订单详情。

// @param body          商品的标题/交易标题/订单标题/订单关键字等。该参
数最长为128个汉字

// @param out_trade_no  商户系统内部的支付订单号, 包含数字与字母, 确保
在商户系统中唯一

// @param total_fee     支付金额, 以分为单位

// @param viewController 调起银联支付的页面, 一般为self

// @param block         接收支付结果回调

- (void)unionPay{
    NSString *outTradeNo = [[BCUtil generateRandomUUID]
stringByReplacingOccurrencesOfString:@"-" withString:@""];
    [BCPay reqUnionPayment:@"BeeCloud01" body:@"渣渣菜鸡"
outTradeNo:outTradeNo totalFee:@"1" viewController:self
payblock:^(BOOL success, NSString *strMsg, NSError *error) {
        if (success) {
            //支付成功
        } else {
            NSLog(@"UnionPay Faild:%@", error.description);
        }
        NSLog(@"Msg:%@", strMsg);
    }];
}

```

银联退款功能

// 银联预退款, 支持部分退款或者全额退款。如果提供的支付订单的交易状态不支持退款, 在block中返回具体的信息; 如果支持退款, 生成预退款订单, 商户在管理后台管理预退款订单。


```

// @param out_trade_no 商户系统内部的支付订单号,包含数字与字母,确保在
商户系统中唯一
// @param refund_fee 退款金额
// @param out_refund_no 商户系统内部的退款订单号,包含数字与字母,确保在
商户系统中唯一
// @param refund_reason 退款原因
// @param block 接收预退款订单生成结果
- (void)unionRefund:(NSString*)outTradeNo {
    NSString *outRefundNo = [[BCUtil generateRandomUUID]
stringByReplacingOccurrencesOfString:@"-" withString:@""];
    [BCPay reqUnionRefund:outTradeNo refundFee:@"0.01"
outRefundNo:outRefundNo refundReason:@"难吃" refundBlock:^(BOOL
success, NSString *strMsg, NSError *error) {
        if (success) {
            //申请退款成功
        } else {
            //申请退款失败, 返回失败原因
            NSLog(@"UnionRefund Faild:%@",error.description);
        }
        NSLog(@"Msg:%@", strMsg);
    }];
}

```

内置购买

BeeCloud提供一个便捷集成的内置购买服务 - BCPay。

开发者只需调用简单的API,就可以快速完成应用的内置购买。

按照苹果官方提供的思路，[官方文档地址](#) 开发者需要了解自己要卖的产品是什么，比如是可消费的还是不可消费的， 自动订阅的还是free订阅的等等，然后设计产品， 购买界面， 恢复购买等等。

但总结起来一般都只需要完成下面五步：

1. 去itunes connect添加产品， 获得产品的唯一id
2. 设计应用内的购买界面
3. 初始化产品， 通过第一步获得的id获取产品实例
4. 完成产品购买的请求以及回调
5. 完成恢复购买的请求以及回调

其中 1, 2步需要开发者自行完成； 3,4,5步可以使用BCPay完成

初始化产品列表

开发者可以自行选择初始化产品的时间，可以在程序启动时，也可以在load自己的购买界面前， 通过上文第1步生成的产品id或者多个id构造一个

NSArray:someProductIds, 然后调用以下代码：

```
// someProductIds 是productId的列表， 如果不传block， 则默认将结果存储在bcPay实例的products(NSArray)和idToProduct(NSMutableDictionary)
// 用户可以通过- (SKProduct *)getProductById:(NSString *)productId
// 获得某个productId对应的product
```

```
BCPay * bcPay = [BCPay sharedInstance]
initProducts:@"someProductIds" withBlock:^(NSArray *products,
NSArray *failedIds, NSError *error) {
    //your logic here
```

```
}
```

开发者可以传递一个nil的block，BCPay会将获得的产品列表存储在BCPay实例的 products里面，并且可以通过

- (SKProduct *)getProductById:(NSString *)productId 方法获得某个 productId对应的product

购买产品

当用户在购买界面触发购买事件（比如点击购买button）之后，开发者可以添加下面这段代码完成内置购买的请求，block是请求完成后的处理，其中state 0代表购买成功，1代表购买失败，2代表是恢复购买 购买成功后会在云端存储购买记录

```
// 方法一：通过 (SKProduct *) product实现购买
// product 可以通过- (SKProduct *)getProductById:(NSString
* )productId 获得
[[BCPay sharedInstance] purchase:product traceID:@"BeeCloud"
withBlock:^(NSString *productId, NSInteger state, NSError *error)
{
    //your code here
}];

// 方法二：直接通过产品ID (productId) 实现购买
[[BCPay sharedInstance] purchaseWithId:productId
traceID:@"BeeCloud" withBlock:^(NSString *productId, NSInteger
state, NSError *error) {
    //your code here
```

```
    }];
```

恢复购买

当用户在购买界面触发恢复购买事件（可以恢复一个或多个或所有产品的购买记录）时，开发者可以添加下面这段代码完成恢复购买的请求，其中productIds为需要恢复购买的产品id列表，state与上相同

```
// productIds即为要恢复的产品id列表 (NSArray)
[[BCPay sharedInstance] restore:[NSArray
 arrayWithObject:productId] traceID:@"BeeCloud"
 withBlock:^(NSString * productId, NSInteger state, NSError * error)
 {
    //恢复购买
 }];
```

查询订单

当用户需要进行订单查询时，可根据traceID或者out_trade_no或者refund_no分别来进行查询。

注：该接口可实现**微信、内置购买、支付宝、银联**四种渠道的订单查询，同时可支持**查询微信、支付宝、银联的退款订单**查询操作，提供同步和异步两种查询方式。

```
/**
 * 同步查询支付订单或退款订单。内置购买只支持查询支付订单表。
 *
 * @param key 根据key查询。trace_id,out_trade_no,our_refund_no
```

```

* @param value 要查询的值
* @param orderType 支付平台的支付订单或者退款订单。
*
* @return 符合条件的订单列表
// Refund status
REFUND_START = 0; // 退款开始
REFUND_REJECT = 1; // 退款被商家拒绝
REFUND_ACCEPT = 2; // 退款被商家同意
REFUND_SUCCESS = 3; // 退款成功
REFUND_FAIL = 4; // 退款被渠道拒绝
REFUND_RETRY = 5; // 退款被渠道拒绝，但原因不明， 需要用原退款单号重试
REFUND_NEED_OFFLINE = 6; // 用户银行卡已注销，现金回流到商户账户，需要
走线下人工操作
*/
+ (NSArray *)queryOrderByKey:(BCPayOrderKey)key value:(NSString
*)value orderType:(BCPayOrderType)type;

/**
* 异步查询支付订单或退款订单。内置购买只支持查询支付订单表。
*
* @param key 根据key查询。trace_id,out_trade_no,our_refund_no
* @param value 要查询的值
* @param ca 支付平台的支付订单或者退款订单
* @param block 接收查询结果
*/
+ (void)querOrderByKeyAsync:(BCPayOrderKey)key value:(NSString
*)value orderType:(BCPayOrderType)type
block:(BCArrayResultBlock)block;

```

查询订单后返回内容如下：

"subject": 商品名称

"body": 商品描述

"buyer_id": 买家ID

"buyer_email": 买家email

"trace_time": 订单创建时间

"out_trade_no": 商家自定义订单号

"partner": 商家ID

"refund_status": 退款状态

"seller_email": 卖家email

"seller_id": 卖家ID

"total_fee": 支付金额

"trace_id": 商户平台用户名

"trade_state": 交易状态

"trace_no": 交易流水号

"refund_finish": 退款操作是否完成

"refund_fee": 退款金额

"refund_reason": 退款原因

"reject_reason": 拒绝退款原因

"refund_status": 退款实时状态

对于refund status:

REFUND_START = 0; 退款开始

REFUND_REJECT = 1; 退款被商家拒绝

REFUND_ACCEPT = 2; 退款被商家同意

REFUND_SUCCESS = 3; 退款成功

REFUND_FAIL = 4; 退款被渠道拒绝

REFUND_RETRY = 5; 退款被渠道拒绝，但原因不明，需要用原退款单号重试

REFUND_NEED_OFFLINE = 6; 用户银行卡已注销，现金回流到商户账户，
需要走线下人工操作

Android SDK

下载安装 SDK

1. 下载SDK: 猛击[这里](#) 下载最新版的BeeCloud SDK, 标准SDK以jar包方式提供给商户应用工程集成, 里面包括微信支付、支付宝支付相关的jar包, 支付宝支付相关的jar包为alipaysdk.jar、alipaysecsdk.jar、alipayutdid.jar这三个jar包。微信支付为libammsdk.jar, 客户无需直接调用这些jar包, 这些将统一通过BeeCloud.jar包进行调用。**目前支持微信支付、支付宝支付, 银联支付即将发布。**

2. 安装SDK: 将alipaysdk.jar、alipaysecsdk.jar、alipayutdid.jar、libammsdk.jar、gson-2.2.4增加到libs中, 右击将jar包作为库文件引入。

配置工程

在AndroidManifest文件中引入如下设置:

```
<uses-permission android:name="android.permission.INTERNET" />
```

```
<uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" />

<uses-permission
    android:name="android.permission.READ_PHONE_STATE" />

<uses-permission
    android:name="android.permission.ACCESS_NETWORK_STATE" />

<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>

<activity
    android:name="cn.beecloud.PaymentActivity"
    android:launchMode="singleTop"
    android:theme="@android:style/Theme.Translucent.NoTitleBar"
/>

<activity-alias
    android:name=".wxapi.WXPayEntryActivity"
    android:exported="true"
    android:targetActivity="cn.beecloud.PaymentActivity"></activity-alias>

s>
```


全局初始化

首先，在APP启动的时候，初始化BeeCloud需要的App Key， 全局只需调用一次。

每个应用对应一个App Key，可在控制台中相应App的 基本信息 里获得：

```
BeeCloud.setAppKey("xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx ", this);
```

BeeCloud所有的网络请求默认超时为5秒，如果需要更改（如改为10秒），请调用以下函数，全局只需调用一次：

```
BeeCloud.setNetworkTimeout(10);
```

微信支付

微信支付功能

添加下列代码发起微信支付

```
// @"自制白开水"为:商品描述
// totalFee为:支付金额*以分为单位*
// outTradeNo为需要您自行生成的订单号,*32个字符内、包含数字与字母,确保在
// 商户系统中唯一*可根据BCUtil.generateRandomUUID]
stringByReplacingOccurrencesOfString:@"-" withString:@""]获取;
// traceid为:支付用户ID,必须保证在商户系统中的唯一性
// 通过该方法实现对各种参数的初始化，然后发起微信支付，并跳转到微信。
BCPay.sharedInstance(MainActivity.this).reqWXPAYMENTAsync("test", "1",
BCUtil.generateRandomUUID().replace("-", ""), "BeeCloud-Android", new
BCPayCallback() {
    @Override
    public void done(boolean b, String s) {
```

```
        System.out.println("微信支付:" + b + "|" + s);
    }
});
```

在支付成功后会在云端存放购买记录。

微信退款功能

需要申请退款时，可以调用退款接口来实现.在调用之前需要在云端先配置三个参数:

1.operatorID

2.安全证书（在商户审核通过后，收到的财付通邮件中有附件以商户号_XXXX.pfx文件即为安全证书）

3.operatorPassword

在客户端用户通过以下代码发起退款请求：

```
// outTradeNo: 用户要退订的订单号
// outRefundNo: 您自行生成的退订订单号, *32个字符内、包含数字与字母, 确保在
// 商户系统中唯一
// refundReason: 退款原因
// refundFee: 退款金额
// payBlock: 发起退款结果回调, 用户在app发起退款请求成功后, 商户会收到退
// 款请求, 商户可在后台确认, 完成退款操作
BCPay.sharedInstance(MainActivity.this).refundWxPayStartRefundAsync(out_trade_no, "201101120001", "1", "退款原因!", new BCPayCallback() {
    @Override
    public void done(boolean b, String s) {
```

```

        System.out.println("微信退款:" + b + "|" + s);
    }
});

```

在退款成功后，会在云端生成退款记录.

支付宝支付

支付宝支付功能

当用户确定选择用支付宝进行支付时，可以调用以下接口进行集成支付宝支付.

```

// @param trace_id    支付用户ID，必须保证在商户系统中唯一.可通过
// trace_id查询订单详情。
// @param out_trade_no 商户系统内部的支付订单号,包含数字与字母,确保在
// 商户系统中唯一
// @param subject      商品的标题/交易标题/订单标题/订单关键字等。该参
// 数最长为 128 个汉字
// @param body          对一笔交易的具体描述信息。如果是多种商品,请将商品
// 描述字符串累加传给body
// @param total_fee     该笔订单的资金总额,单位为RMB-Yuan。取值范围为
// [0.01,100000000.00],精确到小数点后两位
// @param scheme        调用支付的app注册在info.plist中的scheme
// @param block         支付结果回调
BCPay.sharedInstance(MainActivity.this).reqAliPaymentAsync("test",
BCUtil.generateRandomUUID().replace("-", ""), "订单标题", "对一笔交易的具体
描述信息", "0.01", new BCPayCallback() {
    @Override
    public void done(boolean b, String s) {
        System.out.println("支付宝支付:" + b + "|" + s);
    }
}

```

```
    }  
});
```

支付宝退款功能

当需要进行支付宝退款时，调用以下代码：

```
// 根据out_trade_no, refund_no, refund_reason, refund_fee查询订单是  
// 否可退款，允许退款情况下自动生成预退款订单，否则返回不可退款原因。预退款订单  
// 生成成功后，在BeeCloud商户后台对预退款订单进行处理。（订单状态  
// trade_status="@TRADE_SUCCESS"）时支持退款，其他状态下不支持退款。  
// @param out_trade_no 商户系统内部的支付订单号,包含数字与字母,确保在  
// 商户系统中唯一  
// @param refund_no 格式为:退款日期(8位)+流水号(3~8位)。不可重复,  
// 且退款日期必须是当天日期(年月日)。  
// 流水号可以接受数字或英文字符,建议使用数字,但不可接受“000”。例如：  
// 201101120001  
// @param refund_fee 退款金额  
// @param refund_reason 退款原因  
// @param block 退款结果回调  
BCPay.sharedInstance(MainActivity.this).reqAliRefundAsync(out_trade_no,  
"201101120001", "0.01", "退款原因！ ", new BCPayCallback() {  
    @Override  
    public void done(boolean b, String s) {  
        System.out.println("支付宝退款:" + b + "|" + s);  
    }  
});
```

查询订单

当用户需要进行订单查询时，可根据traceID或者out_trade_no或者refund_no分别来进行查询。

注：该接口可实现**微信、支付宝**两种渠道的订单查询，同时可支持**查询支付订单**和**查询退款订单**两项操作查询方式。

```
/**
 * 异步查询支付订单或退款订单。内置购买只支持查询支付订单表。
 *
 * @param key 根据key查询。trace_id,out_trade_no,our_refund_no
 * @param value 要查询的值
 * @param ca 支付平台的支付订单或者退款订单
 * @param block 接收查询结果
 */
BCPay pay = BCPay.sharedInstance(MainActivity.this);
pay.queryOrderByKeyAsync(BCPay.BCPayOrderKey.OrderKeyTraceID,
"BeeCloud-Android", BCPay.BCPayOrderType.BCPayWxPay, new
BCArrayResultCallback() {
    @Override
    public void done(List<Map<String, Object>> lstMaps) {
        System.out.println("微信订单数: " + lstMaps.size());
    }
});
```

查询订单后返回内容如下：

"subject": 商品名称

"body": 商品描述

"buyer_id": 买家ID

"buyer_email": 买家email

"trace_time": 订单创建时间

"out_trade_no": 商家自定义订单号

"partner": 商家ID

"refund_status": 退款状态

"seller_email": 卖家email

"seller_id": 卖家ID

"total_fee": 支付金额

"trace_id": 商户平台用户名

"trade_state": 交易状态

"trace_no": 交易流水号

"refund_finish": 退款操作是否完成

"refund_fee": 退款金额

"refund_reason": 退款原因

"reject_reason": 拒绝退款原因

"refund_status": 退款实时状态

对于refund status

REFUND_START = 0; 退款开始

REFUND_REJECT = 1; 退款被商家拒绝

REFUND_ACCEPT = 2; 退款被商家同意

REFUND_SUCCESS = 3; 退款成功

REFUND_FAIL = 4; 退款被渠道拒绝

REFUND_RETRY = 5; 退款被渠道拒绝，但原因不明， 需要用原退款单号重试

REFUND_NEED_OFFLINE = 6; 用户银行卡已注销，现金回流到商户账户，需
要走线下人工操作

Happy Coding!