iOS 秒支付 S-Pay 开发手册

v2.2

BeeCloud 文档：http://beecloud.cn/doc

联系我们：support@beecloud.cn

更新日期：2/13/2015

目录

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。

[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

认证通过的状态如下：

创建微信应用

A. 填写应用的基本信息，上传应用图标(28x28，108x108)。

B. 填写平台信息。选择应用平台，填写各平台对应的应用信息。

C. 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。

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

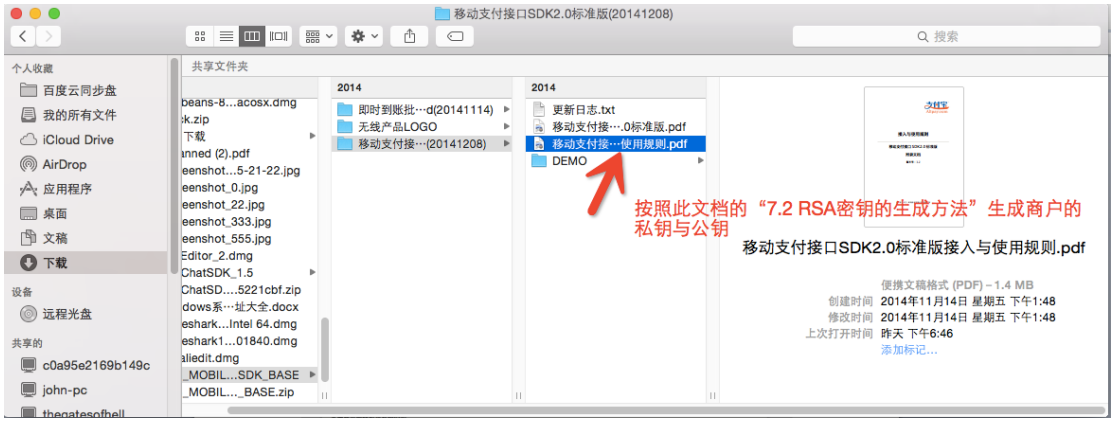
签约移动支付产品

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=20141103081419 93&channel=ent 可以查看具体的签约流程

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

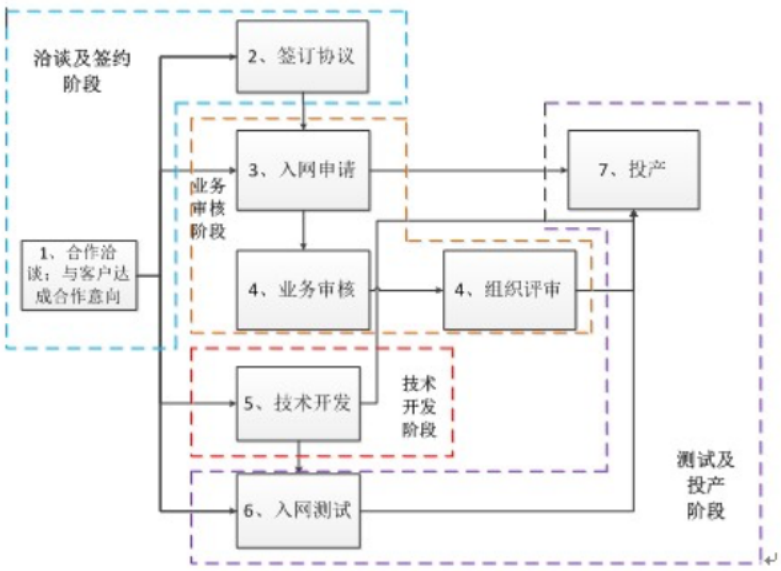
点击上图中“查看 PID|Key”，输入支付密码，可以查看 PID 等要素。

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

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

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。

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 ID 和 App Secret， 全局只需调用一次。

每个应用对应一个 App Key 和 Secret，请务必保管好您的 App Key 和 Secret，可在控制台中相应 App 的 基本信息 里获得:

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

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];
```

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.

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 ApplicationDelegate.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 ApplicationDelegate.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

当用户在使用 BeeCloudSDK 进行银联支付时，需要以下几步:

第一步：在网页控制台填写 银联商户代码，证书密码, 上传银联证书

第二步：在 ApplicationDelegate.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);
    }];
}
```

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 itune 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
}];
```

(从 iOS5.1 开始 deprecated)

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、支付宝三种渠道的订单查询，同时可支持查询支付订单和查询退款订单两项操作，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 emai

“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