

最近公司要求把维护期的项目都集成极光推送，集成期间遇到一些小坑，特此在这总结！

极光推送能干嘛？

- 1.为 JPush Server 上报 Device Token，免除开发者管理 Device Token 的麻烦
- 2.支持iOS APNs推送
- 3.前台运行时，可接收由JPush下发的（透传的）自定义消息
- 4.灵活管理接收用户：Tag（标签分组）、Alias（用户别名）、RegistrationID（设备注册ID）

知道能干嘛了那就开始动手集成????

配push证书：

此步骤直接看极光的[文档](#)即可,写得很详细。

导入必要的框架

- CFNetwork.framework
- CoreFoundation.framework
- CoreTelephony.framework
- SystemConfiguration.framework
- CoreGraphics.framework
- Foundation.framework
- UIKit.framework
- Security.framework
- Xcode7需要的是libz.tbd；Xcode7以下版本是libz.dylib

创建并配置PushConfig.plist文件

在你的工程中创建一个新的Property List文件，并将其命名为PushConfig.plist，文件所含字段如下：

CHANNEL：指明应用程序包的下载渠道，为方便分渠道统计，具体值由你自行定义，如：App Store。

APP_KEY：填写管理Portal上创建应用后自动生成的AppKey值。请确保应用内配置的 AppKey 与第1步在 Portal 上创建应用后生成的 AppKey 一致。

APS_FOR_PRODUCTION

1.3.1版本新增，用于标识当前应用所使用的APNs证书环境。

0 (默认值)表示采用的是开发证书，1 表示采用生产证书发布应用。

注：此字段的值要与Build Settings的Code Signing配置的证书环境一致。

在1.2.2或之前版本的配置文件中，有 TEST_MODE 这个键，新版的SDK不再使用，可以将它删除。

Key	Type	Value
▼ Root	Dictionary	(3 items)
APS_FOR_PRODUCTION	String	0
APP_KEY	String	6ba480302411e162ce0efbae
CHANNEL	String	Publish channel

核心代码

首先在AppDelegate.m 导入#import "JPUSHService.h"

~didFinishLaunchingWithOptions方法贴上核心代码：

```
1 | //极光推送
2 | if ([[UIDevice currentDevice].systemVersion floatValue] >= 8.0) {
3 |     //可以添加自定义categories
4 |     [JPUSHService registerForRemoteNotificationTypes:(UIUserNotificationTypeBadg
5 |
6 |
7 |
8 | } else {
9 |     //categories 必须为nil
10 |    [JPUSHService registerForRemoteNotificationTypes:(UIRemoteNotificationTypeBa
11 |
12 |
13 |
14 | }
15 | //JAppKey : 是你在极光推送申请下来的appKey Jchannel : 可以直接设置默认值即可 Pu
16 | [JPUSHService setupWithOptions:launchOptions appKey:JAppKey
17 |                                channel:Jchannel apsForProduction:NO];
```

之后还需要加入以下方法：

```
1 | - (void)application:(UIApplication *)application didRegisterForRemoteNotificationsWithDevic
2 |     [JPUSHService registerForRemoteNotifications];
3 | }

1 | -(void)application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary
2 |     NSString *alert = [[userInfo objectForKey:@"aps"] objectForKey:@"alert"];
```

```

3 |         if (application.applicationState == UIApplicationStateActive) {
4 |             UIAlertView *alertView = [[UIAlertView alloc] initWithTitle:@"推送消息"
5 |
6 |
7 |
8 |
9 |             [alertView show];
10 |         }
11 |         [application setApplicationIconBadgeNumber:0];
12 |         [JPUSHService handleRemoteNotification:userInfo];
13 |     }

```

我们还能监听极光推送生命周期通知。API里面提供了下面 5 种类型的通知：

API里面提供了下面 5 种类型的通知：

```

1 | extern NSString * const kJPFNetworkDidSetupNotification; // 建立连接
2 | extern NSString * const kJPFNetworkDidCloseNotification; // 关闭连接
3 | extern NSString * const kJPFNetworkDidRegisterNotification; // 注册成功
4 | extern NSString * const kJPFNetworkDidLoginNotification; // 登录成功
5 | 温馨提示：
6 | Registration id 需要在执行到kJPFNetworkDidLoginNotification的方法里获取
7 | extern NSString * const kJPFNetworkDidReceiveMessageNotification; // 收到自定义消息(非AP
8 | 其中，kJPFNetworkDidReceiveMessageNotification传递的数据可以通过NSNotification中的userInfo方法

```

使用方法

Tag (标签分组) & Alias (用户别名)

```

1 | //用于绑定Tag的 根据自己想要的Tag加入，值得注意的是这里Tag需要用到NSSet
2 | [JPUSHService setTags:[NSSet set] callbackSelector:nil object:self];
3 | //用于绑定Alias的 使用NSString 即可
4 | [JPUSHService setAlias:@"" callbackSelector:nil object:self];

```

如果想要即要绑定Alias也要绑定Tag,必须使用以下方法，已被坑

```

1 | //用于同时绑定Tag与Alias的
2 | [JPUSHService setTags:[NSSet set] alias:@"" callbackSelector:nil target:self];

```

一般在项目哪里绑定呢？

我主要是在项目的登录成功或者自动登录后，使用用户的唯一标示进行绑定，或者根据需求添加一些前缀

去除绑定

用户进行退出登录的方法里添加去除绑定即可，值得注意的是用到即时通讯的话，被挤下线也要去除绑定，已被坑，贴代码：

```
1 | //没有值就代表去除
2 | [JPUSHService setTags:[NSSet set]callbackSelector:nil object:self];
3 | [JPUSHService setAlias:@"" callbackSelector:nil object:self];
4 | [JPUSHService setTags:[NSSet set] alias:@"" callbackSelector:nil target:self];
```

测试是否集成成功。到极光平台测试推送：

The screenshot shows the JPush console interface for testing a push notification. At the top, there is a text input field labeled "推送内容" (Push Content) with the text "测试推送" (Test Push) entered. Below this, a status bar indicates "您还剩下 54 个汉字可以输入" (You still have 54 Chinese characters left to enter). The main area displays a preview of an iOS Lock Screen with a notification banner that says "测试推送" (Test Push). Two red arrows point to the "推送对象" (Push Object) section: one labeled "debug模式使用" (Use in debug mode) pointing to the "iOS 开发环境" (iOS Development Environment) checkbox, and another labeled "release模式使用" (Use in release mode) pointing to the "iOS 生产环境" (iOS Production Environment) checkbox. The "推送对象" section also includes radio buttons for "广播(所有人)" (Broadcast to all), "设备标签(Tag)" (Device tags), "设备别名(Alias)" (Device alias), and "Registration ID". The "设备别名(Alias)" option is selected. Below this is a text input field for "输入设备别名" (Enter device alias). The "发送时间" (Send time) section has radio buttons for "立即" (Immediately) and "定时" (Scheduled), with "立即" selected. There is also a checkbox for "是否启用定速推送?" (Enable scheduled push?) which is unchecked.

只要推送成功，剩下的由后台根据Alias或者Tag来推送就可以了????

JPushSDK资源下载

来源：<http://www.cocoachina.com/ios/20160226/15366.html>