IOS Push SDK开发手册

IOS Push服务包括推送和统计两个功能，请参考使用步骤和demo源码将其成功集成到您的工程中。

使用步骤:

1. 将libIXPushSdk.a和IXPushSdk.h加入到您的工程中

2.在工程中添加以下库:

```
SystemConfiguration.framework
CoreTelephony.framework
CoreLocation.framework
Security.framework
libz.tbd
libsqlite3.tbd
```

3.在工程的info.Plist中添加NSLocationWhenInUseUsageDescription key，允许APP获取GPS信息（iOS8以上）

```
<key>NSLocationWhenInUseUsageDescription</key>
<string></string>
```

在工程的info.Plist中添加NSAppTransportSecurity key，允许http网络请求（iOS9以上）

```
<key>NSAppTransportSecurity</key>
<dict>
      <key>NSAllowsArbitraryLoads</key>
      <true/>
</dict>
```

4. 增加代码

在AppDelegate中的- (BOOL)application:(UIApplication *)application didFinishLaunching-WithOptions:(NSDictionary *)launchOptions 调用api，初始化push:

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

    // Override point for customization after application
launch.

   #ifdef __IPHONE_8_0

    if ([[[UIDevice currentDevice] systemVersion] floatValue] >=
8.0){

        UIMutableUserNotificationAction *action =
[[UIMutableUserNotificationAction alloc] init];

        action.identifier = @"acceptAction";

        action.title=@"Accept";

        action.activationMode =
UIUserNotificationActivationModeForeground;

        UIMutableUserNotificationAction *action2 =
[[UIMutableUserNotificationAction alloc] init];

        action2.identifier = @"rejectAction";

action2.title=@"Reject";

        action2.activationMode =
UIUserNotificationActivationModeBackground;

        action2.authenticationRequired = YES;

        action2.destructive = YES;



        UIMutableUserNotificationCategory *categorys =
[[UIMutableUserNotificationCategory alloc] init];

        categorys.identifier = @"Category";

        [categorys setActions:@[action,action2] forContext:
(UIUserNotificationActionContextMinimal)];

        UIUserNotificationSettings *settings =
[UIUserNotificationSettings settingsForTypes:
(UIUserNotificationTypeAlert|UIUserNotificationTypeBadge|
UIUserNotificationTypeSound) categories:[NSSet
setWithObjects:categorys, nil]];

        [IXPushSdkApi register:launchOptions settings:settings];

    } else {

        [IXPushSdkApi register:launchOptions types:
(UIRemoteNotificationTypeAlert|UIRemoteNotificationTypeSound|
UIRemoteNotificationTypeBadge)];

    }

#else

    [IXPushSdkApi register:launchOptions types:
(UIRemoteNotificationTypeAlert|UIRemoteNotificationTypeSound|
UIRemoteNotificationTypeBadge)];
```

```
#endif


    return YES;
}
```

在- (void)application:(UIApplication *) application didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken 中注册deviceToken:

```
- (void)application:(UIApplication *) application
didRegisterForRemoteNotificationsWithDeviceToken:(NSData
*)deviceToken {

    [IXPushSdkApi registerDeviceToken:deviceToken
channel:@"test" version:@"1.0" appId:123456789];

}
```

如果deviceToken获取失败，可在下面的方法中查看原因：

```
- (void) application:(UIApplication *)application
didFailToRegisterForRemoteNotificationsWithError:(NSError
*)error {

    NSLog(@"register fail,%@",error);

}
```

在- (void) application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary *)userInfo fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler 中对收到的消息进行处理：

```
- (void) application:(UIApplication *)application
didReceiveRemoteNotification:(NSDictionary *)userInfo
fetchCompletionHandler:(void (^)
(UIBackgroundFetchResult))completionHandler {

    [IXPushSdkApi handleNotification:userInfo];

}
```

5. 接口的使用

类：public class IXPushSdkApi，所有Api均封装为此类的静态成员

5.1 注册通知服务

```
+(void)register: (NSDictionary *)launchOptions settings:(UIUserNotificationSettings *) settings;(iOS8.0及以上版本)
```

```
+(void)register:(NSDictionary *)launchOptions types:(UIRemoteNotificationType) types;（iOS8.0以下版本）
```

该接口在启动时调用，注册苹果通知服务，另外可以对通知点击行为进行统计

```
// 调用示例
[IXPushSdkApi register:launchOptions settings:settings];
```

5.2注册token和应用id

```
+(void)registerDeviceToken:(NSData *)token channel:(NSString *)channel version:(NSString *)version
                    appId:(uint32_t)appId;
```

该接口在获取到通知服务的token时调用

```
// 调用示例
[IXPushSdkApi registerDeviceToken:deviceToken channel:@"test" version:@"1.0" appId:123456789];
```

5.3注销通知服务

```
+(void)unregisterPush;
```

该接口在需要停止通知服务时调用

```
// 调用示例
[IXPushSdkApi unregisterPush];
```

5.4 恢复通知服务

该接口在停止通知服务后需要恢复时使用

+(void)registerNotificationSettings:(UIUserNotification-Settings *) settings; （iOS8.0及以上版本）

+(void)registerNotificationTypes:(UIRemoteNotification-Type) types; （iOS8.0以下版本）

```
// 调用示例
[IXPushSdkApi registerNotificationSettings:settings];
```

5.5 检查通知服务是否已注册

+(BOOL)isRegistered;

```
// 调用示例
BOOL isRegistered = [IXPushSdkApi isRegistered];
```

5.6 增加标签接口

+(void)addTags:(NSArray *)tags delegate:(id)tagOpDele-gate;

@protocol IXTagOpDelegate<NSObject>

-(void)onResult:(BOOL)result tags:(NSArray *)tags;

@end

应用可以通过标签来标记特定的用户群体， 如“北京”用户，“男”用户， “商务”用户等。Push系统根据伴随消息下发的标签来对特定人群进行消息发送。在push系统中，标签是一个16位的整数，应用将标签值和具体含义相对应。

可同时添加多个标签，用英文逗号（”,”）分隔。

请注意，添加标签必须在成功初始化（注册token和应用id）之后进行。 添加成功的结果会在回调中返回。

方法如下:

```
// tags: NSArray类型,其中的元素是NSNumber，且其值为int16
// delegate: 用于接收添加结果的回调
[IXPushSdkApi addTags:tags delegate:delegate];
```

添加结果通过IXTagOpDelegate的onResult返回：

```
- (void)onResult:(BOOL)result tags:(NSArray *)tags{
    if (tags != nil) {
        if (result == TRUE) {
            NSLog(@"add tag succeed,%@",tags);
        }
    }
}
```

注意，由于系统限制一个用户最多设置16个tag，超出部分会被丢弃（超过16个则最先添加的tag会自动被剔除，只保留最后新添加的16个）。返回结果中只有成功添加的tag。

5.7删除标签接口

```
+(void)deleteTags:(NSArray *)tags delegate:(id)tagOpDelegate;
```

和增加标签接口类似，本接口必须在成功注册token和应用id之后才能调用

```
// tags: NSArray类型,其中的元素是NSNumber，且其值为int16
// delegate: 用于接收删除结果的回调
[IXPushSdkApi deleteTags:tags delegate:delegate];
```

结果会异步通过delegate返回。删除标签时可同时删除多个tag，以英文逗号相隔，若其中有不存在的tag，只会成功删除存在的tag。若删除失败，原因可能是数组中的tag都不存在或网络问题。

5.8 列出标签接口

```
+(NSArray *)listTags;
```

```
NSArray *tags = [IXPushSdkApi listTags];
```

5.9绑定别名接口

```
+(BOOL)bindAlias:(NSString *)alias delegate:(id)aliasOpDelegate;
```

```
@protocol IXAliasOpDelegate<NSObject>
-(void)onResult:(BOOL)result alias:(NSString *)alias;
@end
```

alias为String，请注意限制长度为40个字节，这里长度并不是字符数，而是字符转成字节表示后的字节数，比如一个英文字母的UTF-8只占用一个字节，但是一个中文字符要占用二到三个字节。

本接口必须在成功注册token和应用id之后才能调用

```
[IXPushSdkApi bindAlias:“name” delegate:delegate];
```

绑定结果会在delegate中回调

```
-(void)onResult:(BOOL)result alias:(NSString *)alias{
    if (result == TRUE) {
        NSLog(@“bind alias succeed,%@”,alias);
    }
}
```

5.10解绑别名接口

```
+(BOOL)unbindAlias:(NSString *)alias delegate:(id)alias-OpDelegate;
```

本接口必须在成功注册token和应用id之后才能调用

```
[IXPushSdkApi unbindAlias:“name” delegate:delegate];
```

解绑结果会在delegate中回调

5.12 处理推送消息

```
+(void)handleNotification:(NSDictionary *)userInfo;
```

在接收到远程通知时进行调用

```
[IXPushSdkApi handleNotification:userInfo];
```

5.13 设置Badge接口

```
+ (BOOL)setBadge:(int)value;
```

本接口必须在成功注册token和应用id之后才能调用

```
[IXPushSdkApi setBadge:0];
```

value取值小于0时返回FALSE， 否则返回**TRUE**。

注意事项:

1． 对于增加标签，绑定别名等异步API，依赖于token和appId，请确保初始化token和appId成功返回异步结果后再进行调用;

2．测试过程中，请确认手机或者模拟器已成功连入网络。