

iOS SDK 集成指南

- [使用提示](#)
- [产品功能说明](#)
- [主要特点](#)
- [SDK压缩包内容](#)
- [SDK集成步骤](#)
 - [1.配置推送证书](#)
 - [2.导入SDK到您的项目工程](#)
 - [3.配置MPushAppID](#)
 - [4.添加必须的framework](#)
 - [5.编写代码](#)
 - [6.测试确认](#)

使用提示

本文是 mPush iOS SDK 标准集成指南。如有问题，请咨询技术支持QQ群：[372650575](#)

- 如果您想要快速地测试、感受下mPush的效果，请参考 [3 分钟快速 Demo](#)。
- 如果您看到本文档，但还未下载iOS SDK，请访问[SDK下载](#)页面下载。
- 本SDK只在真机环境下有效！

产品功能说明

mPush 是一个端到端的推送服务，使得服务器端消息能够及时地推送到终端用户手机上，让开发者积极地保持与用户的连接，从而提高用户活跃度、提高应用的留存率。

本 iOS SDK 方便开发者基于 mPush 来快捷地为 iOS App 增加推送功能。

主要特点

- 客户端集成简单，无需对APNs进行相关操作。
- 可定制展示的界面。方便在接收到指定类型的通知后，打开展示界面。
- 可注册多个推送代理，分别处理回调信息。

SDK压缩包内容

- lib文件夹：包含头文件 PushManager.h，静态库文件 libmPushSDK.a。
- doc文件夹：开发指南文档
- demo文件夹：（一个完整的iOS示例项目，演示mPush基本用法，可参考。）



Demo



doc



lib

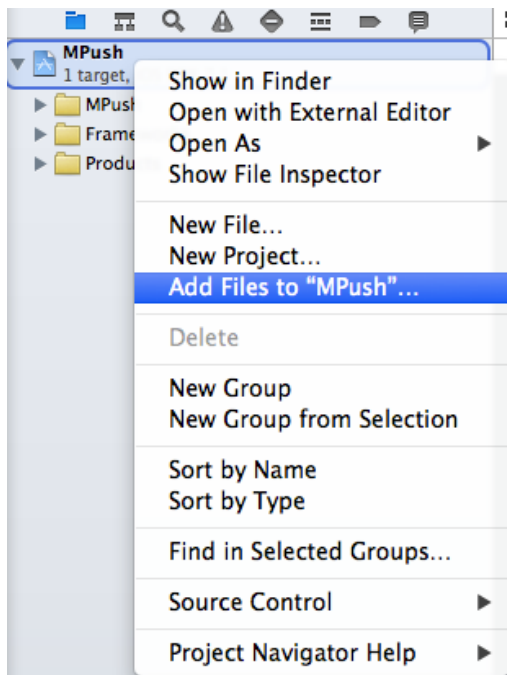
SDK集成步骤

- 1.配置推送证书

- 此部分内容请参考[iOS 推送证书配置指南](#)。

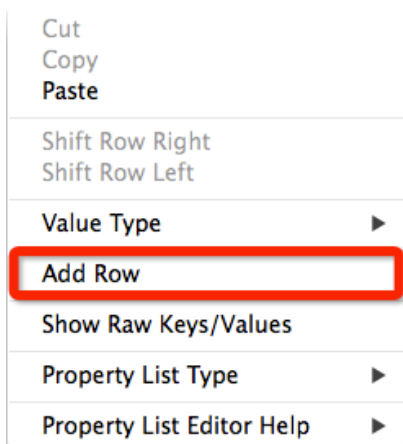
- 2. 导入SDK开发包到您的项目工程

- 解压缩mPush-sdk-v1.x.y.zip压缩包
- 复制lib文件夹到您的项目工程中，或者在项目工程名称处右键选择“Add files to 'Your project name'...”，将解压后的lib子文件夹添加到你的工程目录中。



- 3. 配置 MPushAppID

- 在您的项目文件‘Your project name’-Info.plist中右键选择“Add Row”,添加一行。
- 命名新建行的key值为MPushAppID,类型为string,value为在Protal上创建应用后所分配的APP ID。（此步骤为必选项，否则mPush将不会起作用！）



“

如果您的应用将支持 **iOS8.0** 以上的系统，请在您的项目文件 ‘Your project name’-Info.plist 中添

加 `CLLocationWhenInUseUsageDescription` 缺省字段，类型为 `string`，值为您自定义的提示文字。例如：

| | | |
|--|--------|-----------------|
| <code>NSLocationAlwaysUsageDescription</code> | String | 请允许使用以便更好的使用服务。 |
| <code>NSLocationWhenInUseUsageDescription</code> | String | 请允许使用以便更好的使用服务。 |

• 4. 添加必要的框架

```
CFNetwork.framework
CoreFoundation.framework
CoreTelephony.framework
SystemConfiguration.framework
Security.framework
StoreKit.framework (Optional)
AdSupport.framework (Optional)
CoreLocation.framework
libz.dylib
Foundation.framework
UIKit.framework
```

关于IDFA的声明

“

如果您的应用本身不获取IDFA，只内嵌本SDK时，建议您提交至AppStore时按如下方式配置：

Advertising Identifier

Does this app use the Advertising Identifier (IDFA)?

The **Advertising Identifier (IDFA)** is a unique ID for each iOS device and is the only way to offer targeted ads. Users can choose to limit ad targeting on their iOS device.

If your app is using the Advertising Identifier, check your code—including any third-party code—before you submit it to make sure that your app uses the Advertising Identifier only for the purposes listed below and respects the Limit Ad Tracking setting. If you include third-party code in your app, you are responsible for the behavior of such code, so be sure to check with your third-party provider to confirm compliance with the usage limitations of the Advertising Identifier and the Limit Ad Tracking setting.

This app uses the Advertising Identifier to (select all that apply):

☐ Serve advertisements within the app

☒ Attribute this app installation to a previously served advertisement

☒ Attribute an action taken within this app to a previously served advertisement

If you think you have another acceptable use for the Advertising Identifier, [contact us](#).

Limit Ad Tracking setting in iOS

☒ I, Weiqiang Li, confirm that this app, and any third party that interfaces with this app, uses the Advertising Identifier checks and honors a user's Limit Ad Tracking setting in iOS and, when it is enabled by a user, this app does not use Advertising Identifier, and any information obtained through the use of the Advertising Identifier, in any way other than for "Limited Advertising Purposes" as defined in the [iOS Developer Program License Agreement](#).

如果您应用本身获取IDFA，提交时请按应用本身情况配置。如果不幸被拒，您可尝试下述方式来解决：注意苹果拒绝上架邮件中，此原因以外的其他理由。

• 5. 添加代码

在 `application: didFinishLaunchingWithOptions:` 中调用 `startPushService: pushDelegate: deviceTokenDelegate: customViews:`，注册mPush服务：

```

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    // Override point for customization after application launch.
    .....
    /**
    开启推送服务并设置推送代理以及DeviceToken回调代理、自定义展示页面数据
    pushDelegate 推送代理，必须，参数对象必须实现onMessage:content:extention:方法
    deviceTokenDelegate DeviceToken获取代理，如果无需获取SDK中得DeviceToken，此项可为nil。
    customViews 为自定义展示页面数据字典，如果无需自定义，此项可为nil
    /

    [PushManager startPushService:launchOptions
                  pushDelegate:self
                  deviceTokenDelegate:nil
                  customViews:nil];

    return YES;
}

```

实现 `PushManagerDelegate` 协议，必须实现方法 `onMessage:content:extention:`：

```

/
代理回调方法

@param title    消息title内容
@param content  消息content内容
@param extention 参数字典,此字典包含用户自定义参数和应用状态参数。
                获取应用状态参数可通过 AppStateKey 来获取，状态类型参见 UIApplicationState

@return BOOL 当返回YES时，仅处理至当前事件处，后续事件将不再执行
            当返回NO时，按照事件链继续执行，直至返回YES或者所有事件执行完。

/

- (BOOL)onMessage:(NSString *)title content:(NSString *)content extention:(NSDictionary *)extention{
    //在这里您可以处理收到的消息内容
    NSLog(@"title : %@ \n content : %@ \n extention : %@ \n",title,content,[extention description]);
    return YES;
}

```

• 6. 测试确认

- 确认 MPushAppID（你在 mPush Portal 上创建应用后所分配给应用的APP ID）已经正确的写入 ‘Your project name’-Info.plist中，如未填写，请参考3. 配置 MPushAppID进行配置。
- 确认所使用的签名证书是具有推送权限的证书，如果不是，请参考iOS 推送证书配置指南进行配置。
- 如证书无误，确认项目的Bundle ID和推送证书配置时的Bundle ID是否一致。
- 确认程序启动时调用了startPushService: pushDelegate: deviceTokenDelegate: customViews:接口。并实现了PushManagerDelegate协议方法。
- 确认测试设备已成功接入网络，且网络正常。
- 确认以上步骤无误后，启动应用程序，在 Portal 上向应用程序发送推送消息（详情请参考管理Portal），如果SDK 工作正常，客户端应可收到下发的通知，应有消息到达的日志信息。
- 如以上步骤确认无误，仍无法接收到推送消息，请咨询技术支持QQ群：372650575。