**消息推送SDK**

BFC小组 制定

2014-11-20

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **发布日期** | **主要内容** | **负责人** |
| V1.0 | 2014.11.20 | 定义统一推送系统的手机接口 | 邓俊辉 |
|  | 整合 | 核对SDK文档与实际接口 | 邓俊辉 |
|  | 2014.12.04 | 废弃百度推送功能 | 邓俊辉 |

目录

目录 3

版本信息 3

适用机型 3

依赖环境 3

原理简介 3

具体方法 4

申请应用 4

AndroidManifest.xml 需要注册的消息 4

接收推送内容 7

消息推送 8

## 版本信息

版本号：V1.0

## 适用机型

Android通用机型，包括第三方android手机平台

## 依赖环境

* 需要包含BFC\_Component\_PushIO\_VX.XX.jar，提供推送服务及相关功能；
* 需要包含 encrypt.jar，提供编解码服务；
* 需要包含jpush-sdk-releaseX.X.X.jar，提供极光推送服务；
* 需要包含极光相关的库libjpushXXX.so；
* 需要包含gson-X.X.X.jar;

## 原理简介

* BFC\_Component\_PushIO\_VX.XX.jar中维护了推送系统服务，主要负责对应用推送信息注册的扫描与绑定、分发第三方推送服务推送过来的信息；
* 应用通过在AndroidManifest.xml文件中按照约定注册相关信息；实现推送处理相关服务；向服务器传递机器号等信息之后，即可实现所有推送功能；

## 具体方法

#### 申请应用

向**服务管理人员**申请应用。服务管理人员将提供给你两个API KEY（JAPI和BAPI）和应用包名（包名可自己指定提供给服务管理人员）；

#### AndroidManifest.xml 需要注册的消息

1、包含用户授权信息一下

<!-- JPush service 运行需要的权限 -->

<permission

android:name="你的包名.permission.JPUSH\_MESSAGE"<!-加入自己的包名-->

android:protectionLevel="signature" />

<!-- Required 一些系统要求的权限，如访问网络等-->

<uses-permission android:name="android.permission.RECEIVE\_USER\_PRESENT" />

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

<uses-permission android:name="android.permission.WAKE\_LOCK" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.WRITE\_SETTINGS" />

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

<uses-permission android:name="android.permission.MOUNT\_UNMOUNT\_FILESYSTEMS" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW"/>

<!-- Optional for location -->

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_LOCATION\_EXTRA\_COMMANDS" />

<uses-permission android:name="android.permission.CHANGE\_NETWORK\_STATE" />

2、在 Application 的 onCreate中加入 如下：

if (PushIOContrllerService.ENABLED\_JG\_PUSH) {

JPushInterface.setDebugMode(true);

JpushInterface.init(this);

PushIOManager.init(getApplicationContext());

｝

<application

android:name="XXXXXXXXXX"

。。。。

</application>

5、声明极光的各种接收器以及服务

<!-- Jpush begin -->

<!-- Required SDK核心功能1-->

<activity

android:name="cn.jpush.android.ui.PushActivity"

android:theme="@android:style/Theme.Translucent.NoTitleBar"

android:configChanges="orientation|keyboardHidden">

<intent-filter>

<action android:name="cn.jpush.android.ui.PushActivity" />

<category android:name="android.intent.category.DEFAULT" />

<category android:name="com.eebbk.adpusher" />

</intent-filter>

</activity>

<!-- Required SDK核心功能2-->

<service

android:name="cn.jpush.android.service.DownloadService"

android:enabled="true"

android:exported="false">

</service>

<!-- Required SDK 核心功能3-->

<service

android:name="cn.jpush.android.service.PushService"

android:enabled="true"

android:exported="false">

<intent-filter>

<action android:name="cn.jpush.android.intent.REGISTER" />

<action android:name="cn.jpush.android.intent.REPORT" />

<action android:name="cn.jpush.android.intent.PushService" />

<action android:name="cn.jpush.android.intent.PUSH\_TIME" />

</intent-filter>

</service>

<!-- Required SDK核心功能4-->

<receiver

android:name="cn.jpush.android.service.PushReceiver"

android:enabled="true">

<intent-filter android:priority="1000">

<action android:name="cn.jpush.android.intent.NOTIFICATION\_RECEIVED\_PROXY" /><!--Required 显示通知栏 -->

<category android:name="你的包名" /><!—填入申请的应用名-->

</intent-filter>

<intent-filter>

<action android:name="android.intent.action.USER\_PRESENT" />

<action android:name="android.net.conn.CONNECTIVITY\_CHANGE" />

</intent-filter>

<!-- Optional -->

<intent-filter>

<action android:name="android.intent.action.PACKAGE\_ADDED" />

<action android:name="android.intent.action.PACKAGE\_REMOVED" />

<data android:scheme="package" />

</intent-filter>

</receiver>

<!-- Required SDK核心功能-->

<receiver android:name="cn.jpush.android.service.AlarmReceiver" />

<!—pushio.jar Jush接收器-->

<receiver

android:name="com.eebbk.bfc.mobile.sdk.pushio.PushIOJGPushReceiver "

android:enabled="true">

<intent-filter>

<action android:name="cn.jpush.android.intent.REGISTRATION" />

<action android:name="cn.jpush.android.intent.UNREGISTRATION" />

<action android:name="cn.jpush.android.intent.MESSAGE\_RECEIVED" />

<action android:name="cn.jpush.android.intent.NOTIFICATION\_RECEIVED" />

<action android:name="cn.jpush.android.intent.NOTIFICATION\_OPENED" />

<action android:name="cn.jpush.android.intent.ACTION\_RICHPUSH\_CALLBACK" />

<action android:name="cn.jpush.android.intent.CONNECTION" />

<category android:name="你的包名 " /><!—填入申请的应用包名-->

</intent-filter>

</receiver>

<meta-data

android:name="JPUSH\_CHANNEL"

android:value="developer-default"/>

<meta-data

android:name="JPUSH\_APPKEY"

android:value="XXXXXXXXXXXXXXX" /><!—填入申请的 JAPI Key-->

<!-- Jpush end -->

#### 接收推送内容

使用自己写的接继承 ReceiverPotocal在onMessage中可以收到消息，如：

PushIOManager.*setMessageListener*(**new** MessageListener() {

@Override

**public void** onMessage(String content, String tag, **int** type) {

Log.v("传过来的消息是：", context);

Log.v("传过来的标签是：", tag);

Log.v("传过来的类型是：", type);

});

也可以（定义自己的接收器进行广播接收）

<!-- 自己的接收器 -->

<receiver android:name=".MessagerReciver"

<intent-filter>

<action android:name="pushio.bfc.android.MESSAGE\_RECEIVED\_ACTION" /><!-- 这里的category 为监听push消息的tag值,一般应用只监听一个为包命 -->

<category android:name="com.eebbk.studyos.parentmanager" />

</intent-filter>

</receiver>

public class MessagerReciver extends JpushMsgReceiver{

@Override

public void onMessage(String context, String tag, int type) {

// TODO Auto-generated method stub

Log.v("MessagerReciver","context:"+context);

}

}

#### 消息推送

要对应用推送消息主要有两种方式

1. 如果需要在应用中直接对某个应用推送，则使用：

//context 推送调用的上下文

//msg 推送消息

//tag 需要推送的应用包名

// machinID 需要推送的家教机序列号

MessageSender.getInstance().sendMessage(Context context, String msg,String tag, String machinID);

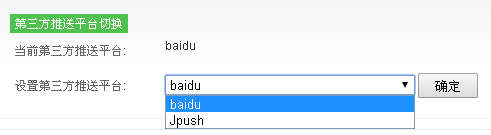
1. 使用bbk网络组分组平台

初次使用，

* 请先在网页最后面点击添加你要推送过的标签



* 选择推送平台



* 单推

