# 在极光开发者服务创建应用

应用管理：

<https://www.jiguang.cn/dev2/#/overview/appCardList>

推送测试：

<https://www.jiguang.cn/jpush2/#/app/d6ff5ca64d8911604c30557e/push_form/notification>

# 配置app的build.gradle

defaultConfig {  
 applicationId "com.android.java"  
   
 /\* 极光start \*/  
 ndk {  
 //选择要添加的对应 cpu 类型的 .so 库。  
 abiFilters 'armeabi', 'armeabi-v7a', 'arm64-v8a'  
 // 还可以添加 'x86', 'x86\_64', 'mips', 'mips64'  
 }  
  
 manifestPlaceholders = [  
 JPUSH\_PKGNAME: "com.android.java", //应用包名  
 JPUSH\_APPKEY : "d6ff5ca64d8911604c30557e", //JPush上注册的包名对应的Appkey  
 JPUSH\_CHANNEL: "developer-default", //暂时填写默认值即可.  
 ]  
 /\* 极光end \*/  
  
}

dependencies {  
 //极光推送  
 api 'cn.jiguang.sdk:jpush:3.4.0' // 此处以JPush 3.4.0 版本为例  
 api 'cn.jiguang.sdk:jcore:2.2.2' // 此处以JCore 2.2.2 版本为例  
}

# 配置Androidmanifest

<!-- 极光start -->  
<permission  
 android:name="com.android.java.permission.JPUSH\_MESSAGE"  
 android:protectionLevel="signature" />  
  
<uses-permission android:name="com.android.java.permission.JPUSH\_MESSAGE" />  
<uses-permission android:name="android.permission.RECEIVE\_USER\_PRESENT" />  
<!--<uses-permission android:name="android.permission.INTERNET" />-->  
<!--<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.MOUNT\_UNMOUNT\_FILESYSTEMS" />  
<!--<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />-->  
<!--<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />-->  
  
<uses-permission android:name="android.permission.ACCESS\_BACKGROUND\_LOCATION" />  
<uses-permission android:name="android.permission.SYSTEM\_ALERT\_WINDOW" /> <!-- 用于开启 debug 版本的应用在 6.0 系统上的层叠窗口权限 -->  
<!--<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />-->  
<!--<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" />-->  
<uses-permission android:name="android.permission.GET\_TASKS" />  
<uses-permission android:name="android.permission.VIBRATE" />  
  
<!-- 极光end -->

# 创建Service

public class JPushService extends JCommonService {  
}

# 创建Receiver

public class JPushReceiver extends JPushMessageReceiver {

}

# 在Androidmanifest中配置Service和Receiver

<!-- 极光start -->  
<service  
 android:name="com.android.frame.push.jpush.JPushService"  
 android:enabled="true"  
 android:exported="false"  
 android:process=":pushcore">  
 <intent-filter>  
 <action android:name="cn.jiguang.user.service.action" />  
 </intent-filter>  
</service>  
<receiver  
 android:name="com.android.frame.push.jpush.JPushReceiver"  
 android:enabled="true"  
 android:exported="false">  
 <intent-filter>  
 <action android:name="cn.jpush.android.intent.RECEIVE\_MESSAGE" />  
 <category android:name="com.android.java" /> <!--应用的包名-->  
 </intent-filter>  
</receiver>  
<!-- 极光end -->

# 初始化SDK

//在Application的onCreate()初始化SDK  
public void init(Context context) {  
 try {  
 JPushInterface.*setDebugMode*(BuildConfig.*DEBUG*);  
 JPushInterface.*init*(context);  
 LogUtil.*i*(*TAG*, "极光初始化完成");  
 } catch (Exception e) {  
 LogUtil.*e*(*TAG*, "极光初始化异常，异常信息：" + e.getMessage());  
 e.printStackTrace();  
 }  
}

# 获取token和利用别名推送

//获取极光token，利用别名推送  
public void registerToken(Context context) {  
 try {  
 LogUtil.*i*(*TAG*, "获取极光token开始");  
 String token = JPushInterface.*getRegistrationID*(context);  
 if (!TextUtils.*isEmpty*(token)) {  
 LogUtil.*w*(*TAG*, "获取极光token成功，token为：" + token);  
 String alias = MessageDigestUtil.*md5*(token);  
 JPushInterface.*setAlias*(context, 0, alias); //利用别名精准推送  
 LogUtil.*w*(*TAG*, "推送别名：" + alias);  
 } else {  
 LogUtil.*e*(*TAG*, "获取极光token失败，token为空！");  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
}

# 在自定义Receiver中对推送进行处理

public class JPushReceiver extends JPushMessageReceiver {  
  
 private static final String *TAG* = PushConstant.*JPUSH\_TAG*;  
  
 //连接极光服务器  
 @Override  
 public void onConnected(Context context, boolean b) {  
 super.onConnected(context, b);  
 LogUtil.*w*(*TAG*, "[onConnected] isConnected:" + b + ",token:" + JPushInterface.*getRegistrationID*(context));  
 }  
  
 //注册极光时的回调  
 @Override  
 public void onRegister(Context context, String s) {  
 super.onRegister(context, s);  
 LogUtil.*w*(*TAG*, "[onRegister] " + s);  
 }  
  
 //注册以及解除注册别名时回调  
 @Override  
 public void onAliasOperatorResult(Context context, JPushMessage jPushMessage) {  
 super.onAliasOperatorResult(context, jPushMessage);  
 LogUtil.*w*(*TAG*, "[onAliasOperatorResult] " + jPushMessage);  
 }  
  
 */\*\*  
 \* 接收到推送下来的通知  
 \* 可以利用附加字段（notificationMessage.notificationExtras）来区别Notication,  
 \* 指定不同的动作,附加字段是个json字符串  
 \* 通知（Notification），指在手机的通知栏（状态栏）上会显示的一条通知信息  
 \*/* @Override  
 public void onNotifyMessageArrived(Context context, NotificationMessage notificationMessage) {  
 super.onNotifyMessageArrived(context, notificationMessage);  
 LogUtil.*w*(*TAG*, "[onNotifyMessageArrived] " + notificationMessage);  
 }  
  
 */\*\*  
 \* 打开了通知  
 \* notificationMessage.notificationExtras(附加字段)的内容处理代码  
 \* 比如打开新的Activity， 打开一个网页等..  
 \*/* @Override  
 public void onNotifyMessageOpened(Context context, NotificationMessage notificationMessage) {  
 super.onNotifyMessageOpened(context, notificationMessage);  
 LogUtil.*w*(*TAG*, "[onNotifyMessageOpened] " + notificationMessage);  
 }  
  
 */\*\*  
 \* 接收到推送下来的自定义消息  
 \* 自定义消息不是通知，默认不会被SDK展示到通知栏上，极光推送仅负责透传给SDK。  
 \* 其内容和展示形式完全由开发者自己定义。  
 \* 自定义消息主要用于应用的内部业务逻辑和特殊展示需求  
 \*/* @Override  
 public void onMessage(Context context, CustomMessage customMessage) {  
 super.onMessage(context, customMessage);  
 LogUtil.*w*(*TAG*, "[onMessage] 自定义消息：" + customMessage.message);  
 }  
}

# 代码结构

