**Droid Plugin**

**DroidPlugin 是\*\*\*360手机助手\*\*\*在Android系统上实现了一种新的\*\*\*插件机制\*\*\*:它可以在无需安装、修改的情况下运行APK文件,此机制对改进大型APP的架构，实现多团队协作开发具有一定的好处。**

##定义：

**HOST程序**：插件的宿主。

**插件**：免安装运行的APK

##限制和缺陷:

1. 无法在插件中发送具有自定义资源的Notification，例如： a. 带自定义RemoteLayout的Notification b. 图标通过R.drawable.XXX指定的通知（插件系统会自动将其转化为Bitmap）
2. 无法在插件中注册一些具有特殊Intent Filter的Service、Activity、BroadcastReceiver、ContentProvider等组件以供Android系统、已经安装的其他APP调用。
3. 缺乏对Native层的Hook，对某些带native代码的apk支持不好，可能无法运行。比如一部分游戏无法当作插件运行。

##特点：

1. 支持Androd 2.3以上系统
2. 插件APK完全不需做任何修改，可以独立安装运行、也可以做插件运行。要以插件模式运行某个APK，你**无需**重新编译、无需知道其源码。
3. 插件的四大组件完全不需要在Host程序中注册，支持Service、Activity、BroadcastReceiver、ContentProvider四大组件
4. 插件之间、Host程序与插件之间会互相认为对方已经"安装"在系统上了。
5. API低侵入性：极少的API。HOST程序只是需要一行代码即可集成Droid Plugin
6. 超强隔离：插件之间、插件与Host之间完全的代码级别的隔离：不能互相调用对方的代码。通讯只能使用Android系统级别的通讯方法。
7. 支持所有系统API
8. 资源完全隔离：插件之间、与Host之间实现了资源完全隔离，不会出现资源窜用的情况。
9. 实现了进程管理，插件的空进程会被及时回收，占用内存低。
10. 插件的静态广播会被当作动态处理，如果插件没有运行（即没有插件进程运行），其静态广播也永远不会被触发。

##使用方法：

####集成

在host中集成Droid Plugin项目非常简单：

1. 我们只需要将Droid Plugin当作一个lib工程应用到主项目中，然后：
2. 在AndroidManifest.xml中使用插件的com.morgoo.droidplugin.PluginApplication：
3. <application android:name="com.morgoo.droidplugin.PluginApplication"
4. android:label="@string/app\_name"
5. android:icon="@drawable/ic\_launcher"
6. 如果你使用自定义的Application，那么你需要在自定义的Application class onCreate和attachBaseContext方法中添加如下代码：
7. @Override
8. public void onCreate() {
9. super.onCreate();
10. //这里必须在super.onCreate方法之后，顺序不能变
11. PluginHelper.getInstance().applicationOnCreate(getBaseContext());
12. }
14. @Override
15. protected void attachBaseContext(Context base) {
16. PluginHelper.getInstance().applicationAttachBaseContext(base);
17. super.attachBaseContext(base);
18. }
19. 将插件中Libraries\DroidPlugin\AndroidManifest.xml中**所有**的provider对应的authorities修改成自己的，默认为com.morgoo.droidplugin\_stub\_P00，如下：
20. <provider
21. android:name="com.morgoo.droidplugin.stub.ContentProviderStub$StubP00"
22. android:authorities="com.morgoo.droidplugin\_stub\_P00"
23. android:exported="false"
24. android:label="@string/stub\_name\_povider" />

可以修改为自己的包名，如: com.example.droidplugin\_stub\_P00 防止跟其它本插件使用者冲突：

<provider

android:name="com.morgoo.droidplugin.stub.ContentProviderStub$StubP00"

android:authorities="com.example.droidplugin\_stub\_P00"

android:exported="false"

android:label="@string/stub\_name\_povider" />

并且修改PluginManager.STUB\_AUTHORITY\_NAME 为你的值:

PluginManager.STUB\_AUTHORITY\_NAME="com.example.droidplugin\_stub"

1. 集成完成。

####安装、卸载插件：

1. **安装、更新插件**,使用如下方法：
2. int PluginManager.getInstance().installPackage(String filepath, int flags)

说明：安装插件到插件系统中，filepath为插件apk路径，flags可以设置为0，如果要更新插件，则设置为PackageManagerCompat.INSTALL\_REPLACE\_EXISTING返回值及其含义请参见PackageManagerCompat类中的相关字段。

1. **卸载插件**，使用如下方法：
2. int PluginManager.getInstance().deletePackage(String packageName,int flags);

说明：从插件系统中卸载某个插件，packageName传插件包名即可，flags传0。

1. **启动插件**：启动插件的Activity、Service等都和你启动一个以及安装在系统中的app一样，使用系统提供的相关API即可。组件间通讯也是如此。

##实现原理：

请参见源码

**FAQ**

[FAQ](https://github.com/Qihoo360/DroidPlugin/wiki/FAQ)

##谁在使用：

[360手机助手](http://sj.360.cn/)

如果想要您的项目展示在这里，请发送邮件到zhangyong232#gmail.com

**支持：**

任何问题可以在项目中提交bug报告，也可以发送邮件以寻求帮助。

QQ群：318901026