# **[DroidPlugin](https://github.com/DroidPluginTeam/DroidPlugin)**

## **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：

我们只需要将Droid Plugin当作一个lib工程应用到主项目中，然后：

在AndroidManifest.xml中使用插件的com.morgoo.droidplugin.PluginApplication：

<application android:name="com.morgoo.droidplugin.PluginApplication"

android:label="@string/app\_name"

android:icon="@drawable/ic\_launcher"

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：

@Override

public void onCreate() {

super.onCreate();

//这里必须在super.onCreate方法之后，顺序不能变

PluginHelper.getInstance().applicationOnCreate(getBaseContext());

}

@Override

protected void attachBaseContext(Context base) {

PluginHelper.getInstance().applicationAttachBaseContext(base);

super.attachBaseContext(base);

}

修改 Libraries\DroidPlugin\build.gradle 的 defaultConfig 配置中 authorityName 的值（建议改为自己的包名+标识，防止跟其它本插件使用者冲突）

集成完成。

#### **安装、卸载插件：**

****安装、更新插件****,使用如下方法：

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。

****卸载插件****，使用如下方法：

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。

****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。