**Android插件化和热修复**

参考：<https://blog.csdn.net/suyimin2010/article/details/80958742>

<https://www.jianshu.com/p/704cac3eb13d>

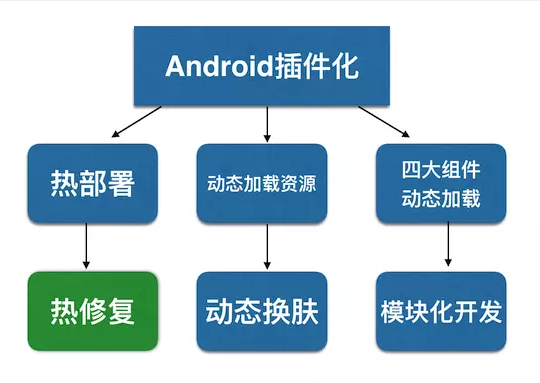
<https://www.jianshu.com/p/d0a6ffd87e09>

<https://www.infoq.cn/article/android-plug-ins-from-entry-to-give-up/>

<https://www.infoq.cn/article/ctrip-android-dynamic-loading/>

<https://www.cnblogs.com/Free-Thinker/p/6755524.html>

<https://blog.csdn.net/suyimin2010/article/details/80958742>



1. 插件化：把需要实现的模块或功能当做一个独立的提取出来，减少**宿主**的规模，当需要使用到相应的功能时再去加载相应的模块。
2. app的build.gradle文件中用apply plugin: 'com.android.library'申明工程为插件；（用apply plugin: 'com.android.application'申明，表示为应用程序）
3. 宿主APP访问插件
4. 类加载
5. BaseDexClassLoader中创建一个DexPathList（其内部将所有的插件和宿主程序封装成Element对象，并将这些对象添加到Element的数组集合dexElements中）

参考：<https://blog.csdn.net/xingzhong128/article/details/80470796>

1. ClassLoaer 使用**双亲委托机制**，使一个Class只会被加载一次。
2. 通过BaseDexClassLoader.findClass()顺序遍历dexElements，找到指定的类则返回，否则返回null
3. 资源加载
4. Android系统加载资源都是通过Resource资源对象来进行加载的，Resource又通过AssetManager来加载资源，因此只需要添加资源（即apk文件）所在路径到AssetManager中，即可实现对插件资源的访问。

参考：<https://www.jianshu.com/p/9132aba3e298>

1. AssetManager的构造方法是@hide隐藏的api，通过反射修改其assetPath实现加载插件资源。

注：AssetMananger是Resource的成员变量，Resource获取资源最终都是通过AssetMananger类loadResourceValue()的native方法查找的。

1. 四大组件加载
2. 通过预占“坑位”的思想，在startActivity之前替换为占坑的名字，Instrumentation#newActivity()之前替换回插件中activity。
3. 插件访问宿主app
4. 资源加载：增加宿主app的引用
5. 代码加载：将宿主app加入依赖
6. 热修复：从修复bug的角度出发，强调的是在不需要二次安装应用的前提下修复已知的bug；