## Android插件化开发

#### 到底什么是插件化开发

插件化开发是将整个app拆分成很多模块，这些模块包括一个宿主和多个插件，每个模块都是一个apk，最终打包的时候将宿主apk和插件apk分开或者联合打包。

**为什么要将一定要使用插件化开发呢？**

1. 宿主和插件分开编译。
 2. 并发开发。
 3. 动态更新插件。
 4. 按需下载模块。
 5. 方法数或变量数爆棚。

**那应该怎么去拆分插件呢？如图：**



淘宝App模块图（部分）

#### 插件化开发的实现原理

**插件化开发中要解决的问题：**

1. 如何在主APP模块中去加载插件APK？
 2. 如何在主模块中跳转到插件APK里面的Activity中去？
 3. 插件APK中的Actiivity怎样能像一个正常的Activity一样？
 4. 插件APK中的Activity的上下文以及资源这块如何处理？
 ...

问题一的解决方式：

Android ClassLoader，用类加载机制去加载插件APK，这样就可以加载到它里面的类。如图:



问题二，三，四的解决方式：

首先，我们Android的跳转都必须要遵循AndroidManifest机制，所有要跳转到ctivity都必须要注册在AndroidManifest.xml清单文件中，但是插件APK中的Activity显然是不可能注册到主APP的AndroidManifest.xml中，所以这里我们要采取插桩的方式来创建一个空的Activity类。

这个代理Activity类不显示任何内容，这个类的意义在于，用它来装载插件APK里面的Activity，把Activity的特性赋予插件APK中的Activity。如图：



在主APP某个Activity进行跳转到时候，要先去加载这个插件（这个插件肯定是已经下载在了本地存储的某个位置），加载的过程中我们已经获取到了它的类加载器以及包信息类。通过包信息类获取到它里面ActivityInfo的集合，然后获取到这个插件的第一个Activity，然后获取到它的名字，传入到代理Ativity中。



在代理Activity的onCreate()方法中通过插件APK的类加载器加载真正目的地的类对象，然后调用真正目的地（插件APK里面的Activity）的onCreate()方法。

注意：在这里我们采取的是接口的引用指向父类的实例的方法来调用的，为什么呢？因为不采取这种方式，我们就必须用到反射，所以通过接口的方式去做更合适。这个代理Activity里面的每个生命周期的方法，我们要相对应的调用插件APK里面的Activity相对应的生命周期的方法，没错，我们就是要把代理Activity的生命周期给插件APK的Activity。实际上就是吧插件APK里面的Activity依附于代理Activity。接口类图：



同时大家应该注意到了一句代码，在代理Activity里面的OnCreate()方法中我们还调用了插件APK中Activity的attch()方法，在这个方法中，我们传递了一个参数this，这个this到底有什么用呢？还记不记得我们的第四个问题：插件APK中的Activity的上下文以及资源这块如何处理？没错，就是把代理Activity的上下文传递到插件APK的Activity中去，插件APK里面所有的Activity用到了上下文的API都要用传进去的代理Activity的上下文来替代，但是这里有个问题，我们很多系统的API都是自动用的上下文，我们该怎么解决呢？看图：



我相信看到这张图大家想到了一点，没错，这个BaseActivity就是插件APK里面Activity的基类，这个基类定义在我们的插件化框架中，当然包括上面的接口以及代理Activity都是定义在我们的插件化框架中。在这个BaseActivity中我们把插件APK中Activity的所有用到上下文的地方都用that来代替了，that是谁？就是代理Activity本身。其实就是当我们在插件APK里面的Activity的时候实际上调用的是代理Activity里面的API。



上图为插件APK里面的Activity代码

所以，当我们去DEBUG的时候，会发现插件APK的Activity实际上就是代理Activity。

举一反三，Activity可以这样玩，那Service呢，其他的组件呢，一样的。同样的配方，同样的味道。

#### 插件化开发的必要性

1. 满足岗位需求。
2. 架构师的必备技能。
3. 一线大厂的必备技能。
4. 团队开发的利器。



当然以上只是一部分代码，大家如果想要了解更多知识点的同学可以加我们助教老师的QQ或者我的QQ。

叮当老师QQ：1979846055

我的QQ：321163715