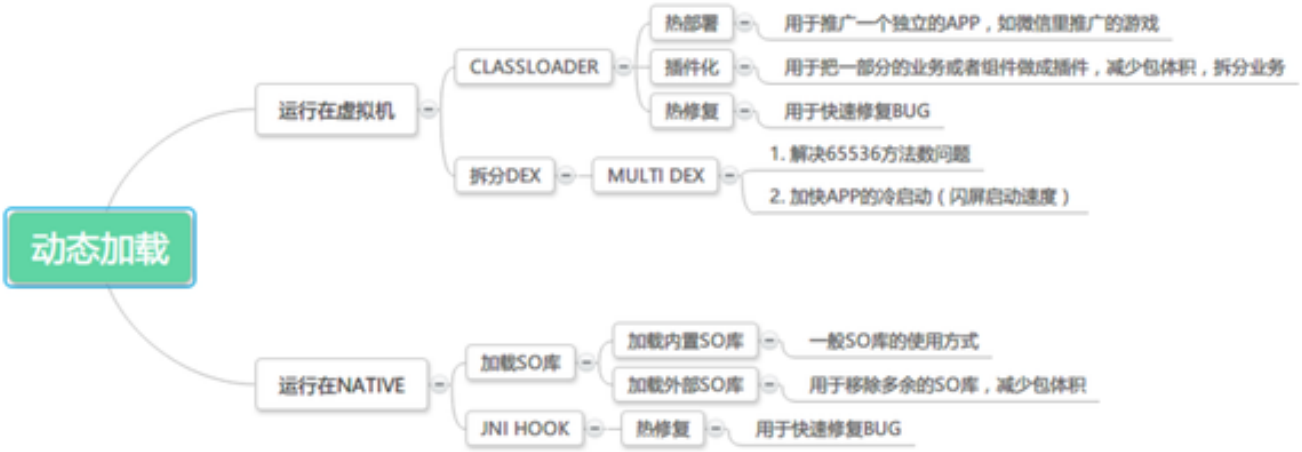
ANDROID动态加载的类型

魔法使Kaede · 1 年前

基本信息

- 作者：kaedea
- 项目：android-dynamical-loading



现在网络上有许多关于动态加载的介绍的文章，谈及的关键词汇有动态加载、插件化、热部署、热修复等，对于一些刚接触这方面开发技术的人来说，可能容易混淆。

虽然我在动态加载系列的文章中或多或少有谈到这些概念的区别，但是我觉得认识这些区别对于使用动态加载技术还是挺重要的，所以特别开这个新的文章进行分析。

动态加载的类型

无论是插件化、热部署还是热修复，这些技术的根源都可是说是动态加载，这也是我把“动态加载”作为这个系列文章主题的原因。

动态加载，就是在程序运行时，加载外部的可执行文件并运行。这里的运行时就是指应用冷启动并开始工作后；外部可以是可以是SD卡，可以是data目录，也可以是jniLib目录，这些可执行文件是没有随着应用一起编译的。

Android的动态加载按照工作机制的不同，可以分为虚拟机层动态加载和Native层动态加载两大类。

运行在虚拟机

简单来说就是只用JAVA代码搞定的类型。

基于虚拟机的动态加载技术的核心是类加载器ClassLoader，通过它我们能够加载一些新的类，这种方式也是目前大部分技术文章谈到的加载方式。其中，根据ClassLoader使用方式的不同，又演变出“热部署”、“插件化”、“热修复”等技术。

热部署

加载外部可执行文件的ClassLoader实例与原APP的ClassLoader实例是互相独立的（不在同一棵代理树上），加载进来的新的类与原APP（宿主）里存在的类互相独立，根据Java对类的定义，因为这些类的ClassLoader不同，所以他们即便包名和类名一致，或者有继承关系，他们也属于不懂的类。所以以这种方式加载进来的类与原有的类不能互通，不能污染宿主原有的类，适合用来动态加载一些独立的业务，比如一些推广的游戏，在宿主上提供一个入口，用户不需要安装游戏就能运行。因为这种方式起到不用安装就能部署游戏的作用，所以称为热部署。

插件化

加载外部可执行文件的ClassLoader实例与宿主的ClassLoader实例不是互相独立的，用宿主的ClassLoader加载过的类就无法从外部可执行文件中再次加载，它们可以共用一个公共库，习惯上把外部可执行文件称为插件。插件里可以存放公共库里一些接口的实现类，可以有一些新的Activity或者Service等组件，可以把一些宿主里的业务挪到插件中，插件可以自主升级，不用随着宿主APP发版。

热修复

在使用插件化技术的同时，也可以使用插件中的新的类来替换宿主同名的类，这样就能修复宿主中原有的类存在的BUG。相比插件化，热修复因为不需要考虑组件和res资源的问题，所以相对简单得许多，要保证插件种新的类的加载要在加载宿主中原有类的之前。

拆分DEX

相信大家都知道打包DEX时65536方法数超标问题，也就是一个DEX只能有65536个方法，因此有了multi-dex的解决方案，把本来只有一个的DEX，拆分成复数以上的DEX，运行时挨个加载进来，这其实也算是一种动态加载，只不过实现过程对开发者是透明的。

除此之外，还有另一种拆分DEX是用于减少冷启动的时间的。冷启动是指应用第一次从用户点击到完成初始化工作的全部过程。随着现在APP的体积不断增长，一些APP的DEX文件十分庞大，APP在启动的时候，单单加载所有的DEX文件就需要非常多的耗时，所以用户点击APP的时候会有一个明显的卡顿过程。因此有一种拆分DEX的方案是“拆分一个启动闪屏用的DEX，里面只存放启动闪屏界面需要用到的类，因此非常小，其他类放到其他DEX里面”，启动的时候因为只需要加载闪屏的DEX，所以非常快，APP进入闪屏后，通过异步任务去完成其他DEX的加载，就能消除卡顿的过程。

第一种拆分DEX是官方支持的，开发者只需要打开multi-dex功能即可；第二种拆分DEX则需要开发者自己设计。

基于ClassLoader的动态加载都有个共同的特点，就是新的类一旦加载进内存了，就无法再次替换了，所以无法在运行时候升级功能，需要重启APP才能生效。

运行在Native

有另一种动态加载方式是工作在Native层的，相比于ClassLoader，在Native层的动态加载不需要重新启动APP就能生效，这类的加载有 加载SO库 和 基于JNI HOOK 的热修复。

加载SO库

加载SO库是最常见的Native动态加载，我们项目经常中使用SO库，编译APP的时候，SO并不会参与编译，会原封不动被拷贝到APK包里的lib目录下，安装APK的时候，系统会扫描lib文件夹下支持当前设备CPU类型（比如arm或x86）的SO库（APK包会带有多种CPU类型对应的SO库，安装的时候只需要对应类型的）并拷贝到系统安装目录，APP在运行时可以调用 System#loadLibrary 方法动态加载对应的SO库，此外还可以调用 System#load 加载指定路径上的SO库。

现在的APK里面往往带有非常多的SO库，而APP运行时只需要用到对应CPU类型的SO库，因此把SO库从APK包里剥离出来也是APK瘦身的有效手段。

JNI HOOK

基于JNI HOOK的热修复技术的代表框架有阿里巴巴的 AndFix。Android中，修复BUG的方式就是更新类的方法，和ClassLoader通过加载新的类来更换方法的实现的想法一样，AndFix 也是通过更换方法的做法来实现热修复，不过做法比较取巧。Android中执行Native方法的时候，会去SO库中查找对应的C/C++方法，而 AndFix 先把普通Java方法用Native方法代替，再通过更换不同SO库还更换Native方法的实现。

Android 开发

与下你的评论…

王维民

条理清晰，分类清楚！有几个错别字。。。

1 年前

纸鸢

感谢

1 年前

魔法使Kaede（作者） 回复 王维民 查看对话

更新的时候会改过来

1 年前

Jian Wang

学习了~

1 年前

wptdxii

学习了 感谢

1 年前

Desmond

666666

1 年前

陈风

Mark

1 年前

文章被以下专栏收录

中二病也要开发 ANDROID

中二ANDROID魔法使 (●'◡'●) kaedea.com

进入专栏

推荐阅读

使用SO库时需要注意的一些问题

基本信息作者：kaedea项目：android-dynamical-loadingAndroid项目里的SO库正好动态加载系列… 查看全文 >

魔法使Kaede · 1 年前 · 发表于 中二病也要开发 ANDROID

阅读ANDROID源码的一些姿势

前面吐槽了 有没有必要阅读Android源码，后面觉得只吐槽不太好，还是应该多少弄点干货。日常… 查看全文 >

魔法使Kaede · 2 年前 · 发表于 中二病也要开发 ANDROID

喵星&汪星人也会导致全球气候变暖+大量能源消耗

本文是在我读完一篇神奇的学术论文后写出来的，特与大家分享讨论。该论文题目为Environmen… 查看全文 >

弗雷刘 · 9 天前 · 编辑精选

坚信过程！76人到崛起的时候了？

我们在谈论如何守时且有效的完成一件事的时候，总会提到一个词，叫“计划”，就是写下一… 查看全文 >

o代号9527o · 16 天前 · 编辑精选 · 发表于 9527的篮球梦