HenCoder Plus 第 27 课 讲义

手写热更新

热更新 / 热修复

不安装新版本的软件，直接从网络下载新功能模块来对软件进行局部更新

热更新和插件化的区别

区别有两点

1. 插件化的内容在原 App 中没有，而热更新是原 App 中的内容做了改动
2. 插件化在代码中有固定的入口，而热更新则可能改变任何一个位置的代码

热更新的原理

- ClassLoader 的 dex 文件替换
- 直接修改字节码

前置知识：loadClass() 的类加载过程

- 宏观上：是一个带缓存的、从上到下的加载过程（即网上所说的「双亲委托机制」）
- 对于具体的一个 ClassLoader：
    - 先从自己的缓存中取
    - 自己没有缓存，就找父 ClassLoader 要（parent.loadClass()）
    - 父 View 也没有，就自己加载（findClass()）
- BaseDexClassLoader 或者它的子类（DexClassLoader、PathClassLoader 等）的 findClass()：
    - 通过它的 pathList.findClass()
    - 它的 pathList.loadClass() 通过 DexPathList 的 dexElements 的 findClass()
    - 所以热更新的关键在于，把补丁 dex 文件加载放进一个 Element，并且插入到 dexElements 这个数组的前面（插入到后面的话会被忽略掉）

手写热更新

- 因为无法在更新之前就指定要更新谁；所以不能定义新的 ClassLoader，而只能选择对 ClassLoader 进行修改，让它能够加载补丁里面的类
- 因为补丁的类在原先的 App 中已经存在，所以应该把补丁的 Element 对象插入到 dexElements 的前面才行，插入到后面会被忽略掉。
- 具体的做法：反射

1. 自己用补丁创建一个 PathClassLoader
2. 把补丁 PathClassLoader 里面的 elements 替换到旧的里面去
3. 注意：
    1. 尽早加载热更新（通用手段是把加载过程放在 Application.attachBaseContext()）
    2. 热更新下载完成后在需要时先杀死程序才能让补丁生效
4. 优化：热更新没必要把所有内容都打过来，只要把改变的类拿过来就行了
    - 用 d8 把指定的 class 打包进 dex
5. 完整化：从网上加载
6. 再优化：把打包过程写一个 task