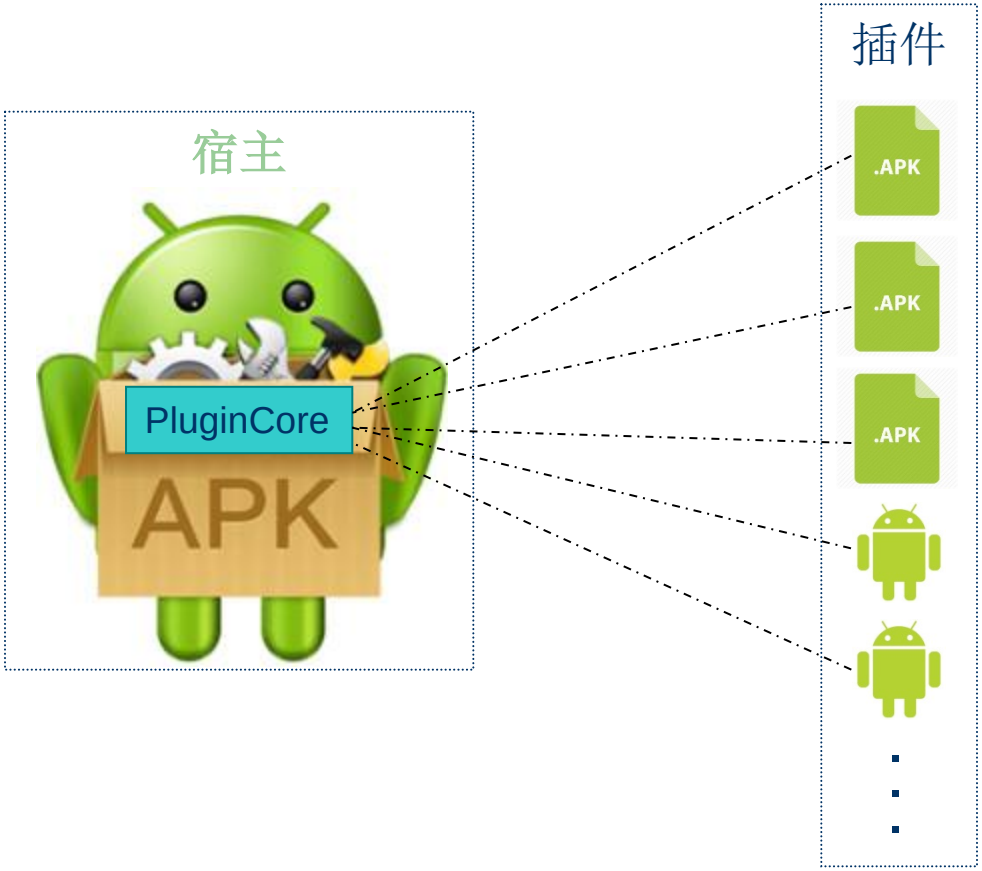
内容

- 插件技术的优势
- 插件技术历史
- 插件技术中的三成员
- 不一样的知识领域
- 不一样的流程
- 插件技术的核心思想
- 插件化道路上的问题
- 组件完整生命周期
- TwsPluginFramework

Android系统就是插件化思想产品

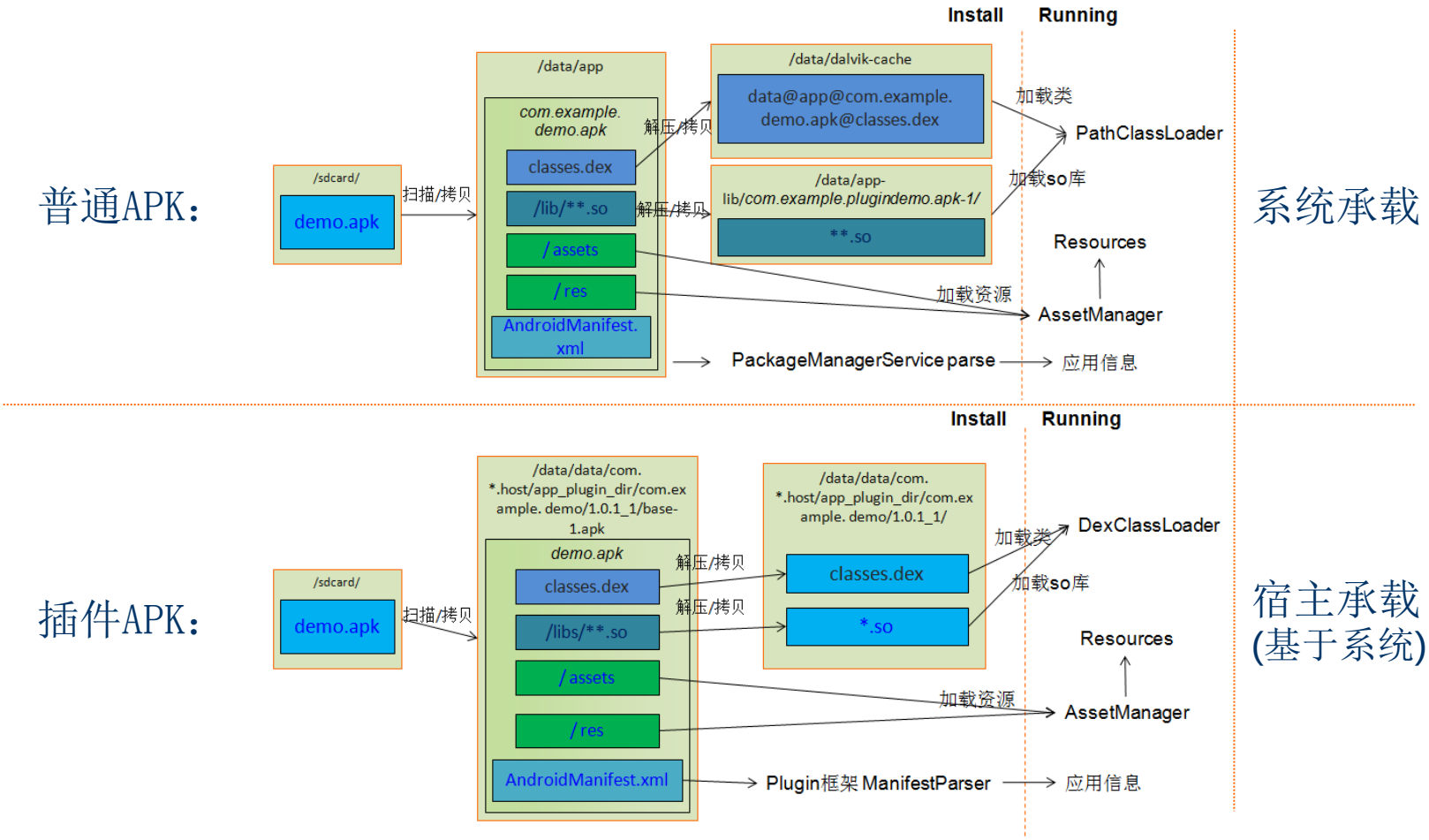
插件技术中的三成员



不一样的知识领域

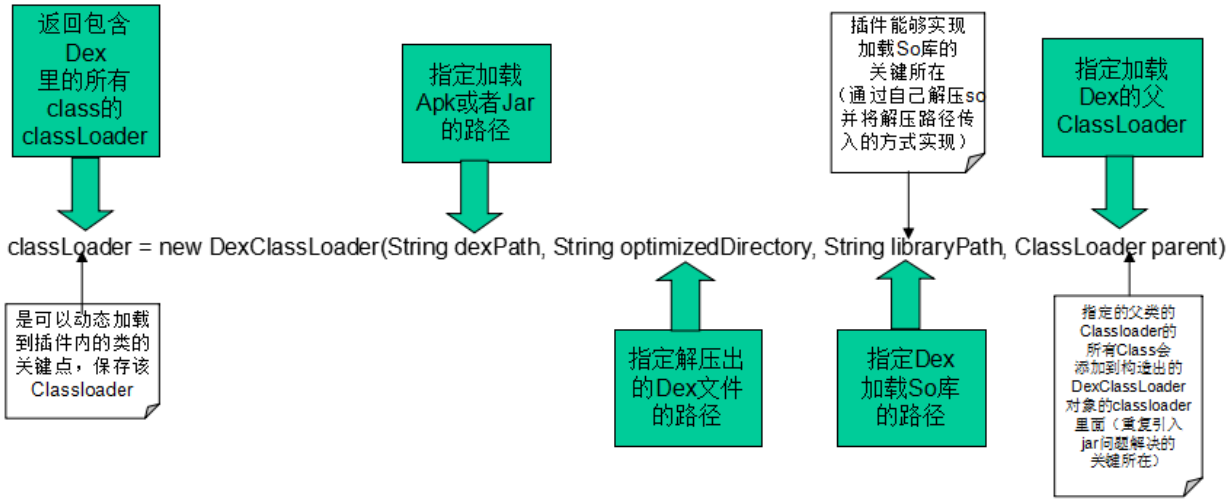

开发插件化工程
开发普通应用工程
Applications
Home
Contacts
Phone
Browser
...
HOOK
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource Manager
Location Manager
XMPP Service

不一样的流程


普通APK：
Install
Running
/sdcard/
demo.apk
扫描/拷贝
/data/app
com.example.demo.apk
classes.dex
/lib/**.so
/assets
/res
AndroidManifest.xml
解压/拷贝
解压/拷贝
/data/dalvik-cache
data@app@com.example.demo.apk@classes.dex
/data/app-lib/com.example.plugindemo.apk-1/
**.so
加载类
加载so库
PathClassLoader
Resources
加载资源
AssetManager
PackageManagerService parse
应用信息
系统承载
插件APK：
Install
Running
/sdcard/
demo.apk
扫描/拷贝
/data/data/com.*.host/app_plugin_dir/com.example. demo/1.0.1_1/base-1.apk
demo.apk
classes.dex
/libs/**.so
/assets
/res
AndroidManifest.xml
解压/拷贝
解压/拷贝
/data/data/com.*.host/app_plugin_dir/com.example. demo/1.0.1_1/
classes.dex
*.so
加载类
加载so库
DexClassLoader
Resources
加载资源
AssetManager
Plugin框架 ManifestParser
应用信息
宿主承载
(基于系统)

插件框架的核心思想

一、代码加载



二、资源加载

反射调用AssetManager的隐藏方法addAssetPath添加指定路径资源

三、程序info解析

解析AndroidManifest.xml文件

插件化道路上的问题

- Android系统对应用有限制
- 插件和宿主的资源处理
- 对插件的能力支持到什么程度

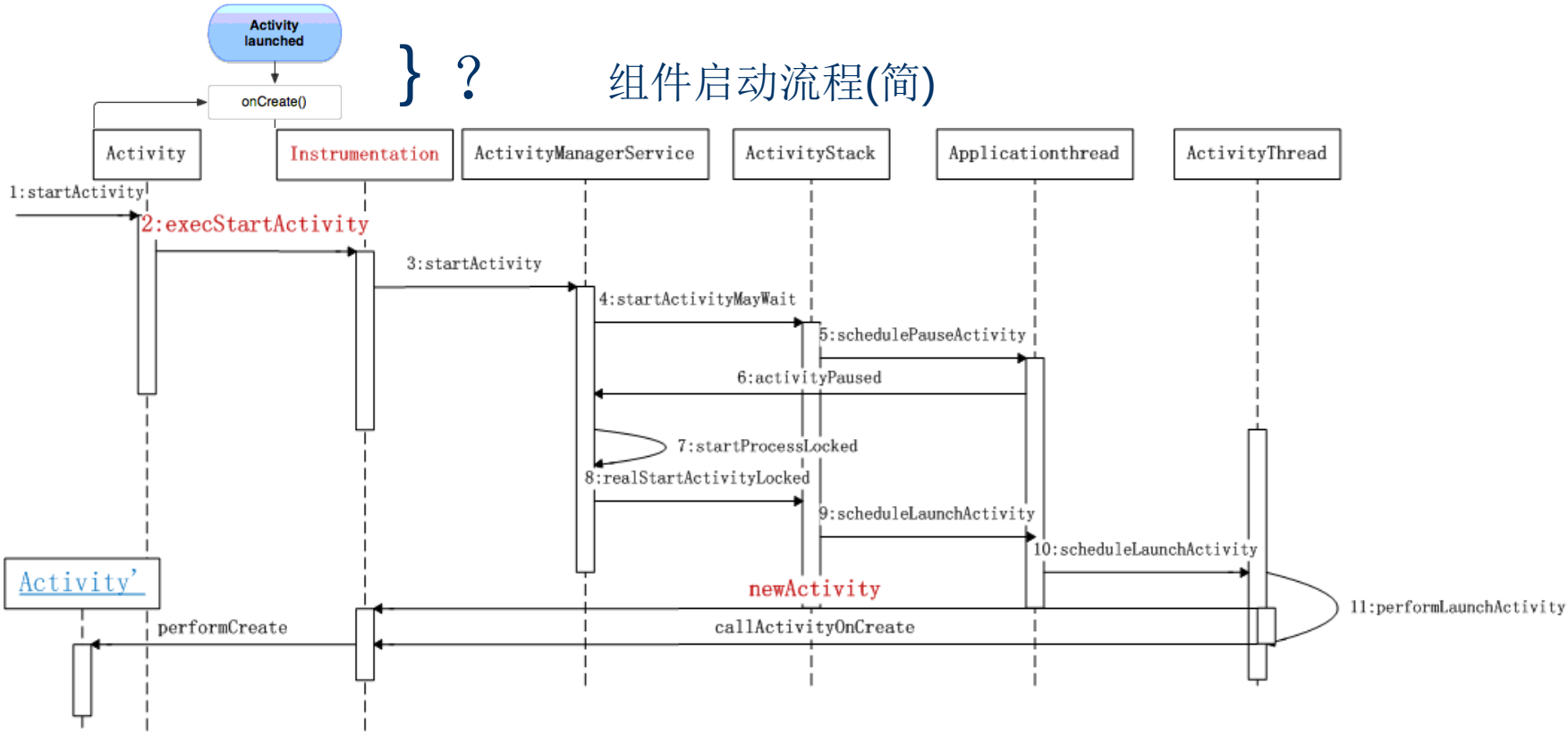
  [application、四大组件、fragment、组件独立生命周期、so、Toast、notification、本地页面、宿主资源&能力共享、白盒插件、. . .]支持的越多，对系统的掌握要求越广、适配成本也就越大
- **组件生命周期**
- 独立插件支持
- 皮肤
- 宿主能力如何共享
- 插件应用的包信息合法化
- **插件&宿主解耦**

  运行关系、显示关系、...
- **第三方应用 是否可以启动 插件的组件？**
- . . .

完整的生命周期（一）

android.content.ActivityNotFoundException: Unable to find explicit activity class

AndroidManifest.xml里面没注册



完整的生命周期（二）

step1:构建Instrumentation代理 - 主要处理启动Activity进入系统流程之前的替换和流程后面的Activity创建以及组件生命周期入口前的调整处理。

step2:构建自己的ContextWrapper，用于广播发送、服务启动或者绑定进入系统流程之前的替换

step3:构建Handler.Callback代理 - 主要处理receiver和service在流程后的还原

step4:替换宿主主进程ActivityThread的成员变量mInstrumentation;

step5:替换宿主主进程ActivityThread的成员变量mInstrumentation;

step6:替换宿主主进程Application[ContextWrapper]的成员变量mBase;

TwsPluginFramework

优势[在巨人的肩膀上进一步提升]

支持的功能：

1、插件包无需安装，由宿主程序动态加载运行。

2、支持fragment、activity(包括4个LaunchMode)、service、receiver、contentprovider、so、application、notification。

3、支持插件自定义控件。

4、开发插件apk对齐原生apk，相比原生无门槛。

5、插件中的组件拥有真正生命周期，完全交由系统管理、非反射代理。

6、支持插件共享宿主的代码和资源。

7、支持插件使用宿主主题、系统主题、插件自身主题以及style。

8、支持非独立插件和独立插件。

9、支持插件资源文件中直接通过@方式引用宿主共享资源

10、支持插件发送notification时在RemoteViews携带插件自定义的布局资源（只支持5.x及以上）

11、支持插件热更新。

12、支持第三方apk启动插件组件activity、service

13、支持插件的service另外配置进程

附件 - 参考资料

Android 插件技术实战总结:

https://mp.weixin.qq.com/s/1p5Y0f5XdVXN2EZYT0AM_A

github地址：

https://github.com/rickdynasty/TwsPluginFramework.git