**Android应用双开实现**

原创 2017年10月23日 16:44:20

* 标签：
* [双开](http://so.csdn.net/so/search/s.do?q=%E5%8F%8C%E5%BC%80&t=blog)
* 509

Android应用双开是某些特殊人群的需要，目前已知的双开方案个人终结为3种：

1.反编译apk，然后修改uid等相关信息让系统弄认为apk有不同，然后重新安装。这个方法是简单粗暴的，不过目前主流的app估计都会预防这种修改方法的。对于手机厂商来说，拥有整个系统的代码权限，不会用这么低级的手段。此种方法我个人也无深入研究。

2.插件化，在启动代理apk后动态加载需要双开的apk，此种方法是双开实现中最复杂最有技术含量的实现方式。插件化开源的有360 发布的DroidPlugin，未开源最有名气的就是lbe的MultiDroid，据我所知MultiDroid也是诸多手机厂商使用的较多的方案。此种方案的好处是无需修改系统代码，问题隔离的完善，对已有系统没有任何影响。关于插件化的原理见http://www.jianshu.com/u/e347b97e2f0c，weishu的多篇文章讲解的很透彻了。

3.多用户，小米使用的是这种实现方案，也是我个人想到双开第一个冒出的想法就是多用户。用户系统是Android系统支持的完整的一套东西，基于此开发就相当于站在巨人的肩膀上。基于账户系统的双开本来以为有一定的难度，要改不少地方，最近看了不少UserManagerService的文章和相关源码，最后发现一行代码不改就能看到双开的效果。

见原生代码的demo

development/samples/browseable/BasicManagedProfile

此demo的本意是展示android5.0的新特性，设备管理managed profiles 。源码相关就是DevicePolicyManager，它可以限制app的行为，例如无法使用摄像头，浏览器只能打开固定几个网页等等。基本原理是新建了一个拥有FLAG\_MANAGED\_PROFILE flag的user，并且创建想要控制的app到该user，然后控制app行为。该user的app和原有的app都会显示在launcher上面，这个是市面主流的双开app需要的所要的效果是一样一样的.......

不仅桌面同时显示，按history键显示的历史堆栈会显示所有打开的app，两个user的同app也都会显示。

原生的demo只显示了system app的使用，如果是三方app需要修改几行代码。

development/samples/browseable/BasicManagedProfile/src/com/example/android/basicmanagedprofile/BasicManagedProfileFragment.java

**[java]** [view plain](http://blog.csdn.net/firedancer0089/article/details/78320436) [copy](http://blog.csdn.net/firedancer0089/article/details/78320436)

1. **private** **void** setAppEnabled(String packageName, **boolean** enabled) {
3. //                    devicePolicyManager.enableSystemApp(
4. //                            BasicDeviceAdminReceiver.getComponentName(activity), packageName);
6. **int** userId = UserHandle.myUserId();
7. packageManager.installExistingPackageAsUser(packageName, userId);
9. }

注释的代码是原有代码，后面的是我针对三方app写的代码。注意AndroidManifest中要加相应的权限

**[java]** [view plain](http://blog.csdn.net/firedancer0089/article/details/78320436) [copy](http://blog.csdn.net/firedancer0089/article/details/78320436)

1. <uses-permission android:name="android.permission.INSTALL\_PACKAGES"/>
2. <uses-permission android:name="android.permission.DELETE\_PACKAGES"/>
3. <uses-permission android:name="android.permission.INTERACT\_ACROSS\_USERS" />
4. <uses-permission android:name="android.permission.INTERACT\_ACROSS\_USERS\_FULL" />

基本功能ok了，其余的双开需求就是苦力活了。

再次证明了对系统代码熟悉的话，实现功能可能异常的简单。