# Android ContentProvider共享数据更新通知机制

参考：<https://blog.csdn.net/Luoshengyang/article/details/6985171>

1. Content Provider的数据更新机制划分为三部分：一是ContentService的启动过程；二是监控数据变化的ContentObserver的注册过程；二是数据更新通知的发送过程；
2. ContentService的启动过程分析
3. ContentService扮演者ContentObserver的注册中心的角色
4. Android系统进程Zygote在启动时，启动System进程来加载系统的一些关键服务，包括ContentService；
5. ServerThread.run()中调用ContentService.main()创建一个ContentService实例，并添加到ServiceManager中；
6. ContentObserver的注册过程分析
7. 通过Context.getContentResovler()获取到ContentResolver接口，通过ContentResolver.registerContentObserver()注册一个ContentObserver（需从外面传一个handler到ContentObserver）；ContentObserver负责监控数据的变化，数据变化时ContentService就会调用ContentObserver.onChange()来处理；
8. ContentResolver.registerContentObserver()主要做了以下三件事：一是调用getContentService函数来获得前面已经启动起来了的ContentService远程接口；二是调用从参数传进来的ContentObserver的getContentObserver()来获得一个Binder对象，ContentService服务可以通过这个Binder对象通知相应的ContentObserver所监控的数据发生变化了；三是通过调用这个ContentService远程接口的registerContentObserver函数来把这个Binder对象注册到ContentService中去。
9. 数据更新通知的发送过程
10. Content Provider的数据发送变化时，调用ContentResolver.notifyChange()，再调用ContentService.notifyChange()。ContentService首先收集注册了监控对应URI的ContentObserver（Binder对象）；然后分别调用了这些ContentObserver.onChange()来通知它们监控的数据发生变化了。
11. Binder对象ContentObserver.onChange()中调用ContentObserver.dispatchChange()，将更新通知封装成消息，并通过注册ContentObserver时传进来的handler将该消息发送到主线程；
12. NotificationRunnable.run()调用主线程的ContentObserver.onChange()来处理数据更新；