**Context区别**

* Activity和Service以及Application的Context是不一样的,Activity继承自ContextThemeWraper.其他的继承自ContextWrapper
* 每一个Activity和Service以及Application的Context都是一个新的ContextImpl对象
* getApplication()用来获取Application实例的，但是这个方法只有在Activity和Service中才能调用的到。那么也许在绝大多数情况下我们都是在Activity或者Service中使用Application的，但是如果在一些其它的场景，比如BroadcastReceiver中也想获得Application的实例，这时就可以借助getApplicationContext()方法，getApplicationContext()比getApplication()方法的作用域会更广一些，任何一个Context的实例，只要调用getApplicationContext()方法都可以拿到我们的Application对象。
* Activity在创建的时候会new一个ContextImpl对象并在attach方法中关联它，Application和Service也差不多。ContextWrapper的方法内部都是转调ContextImpl的方法
* 创建对话框传入Application的Context是不可以的
* 尽管Application、Activity、Service都有自己的ContextImpl，并且每个ContextImpl都有自己的mResources成员，但是由于它们的mResources成员都来自于唯一的ResourcesManager实例，所以它们看似不同的mResources其实都指向的是同一块内存
* Context的数量等于Activity的个数 + Service的个数 + 1，这个1为Application

**Java中反射的作用是什么?什么时候会用到**

JAVA反射机制是在#运行时#，对于任意一个类，都能够知道这个类的所有属性和方法；对于任意一个对象，都能够调用它的任意一个方法；这种动态获取的信息以及动态调用对象的方法的功能称为java语言的反射机制。 Java反射机制主要提供了以下功能： a)在运行时判断任意一个对象所属的类； b)在运行时构造任意一个类的对象； c)在运行时判断任意一个类所具有的成员变量和方法； d)在运行时调用任意一个对象的方法；生成动态代理。