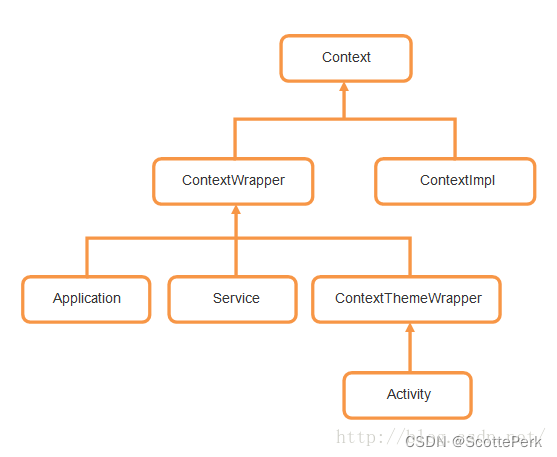
# Context



Context到底是什么

Context的中文翻译为：语境; 上下文; 背景; 环境，在开发中我们经常说称之为“上下文”，Android程序员把“场景”抽象为Context类，他们认为用户和操作系统的每一次交互都是一个场景，比如打电话、发短信，这些都是一个有界面的场景，还有一些没有界面的场景，比如后台运行的服务（Service）。

Android程序不像Java程序一样，随便创建一个类，写个main()方法就能运行，Android应用模型是基于组件的应用[设计模式](https://so.csdn.net/so/search?q=%E8%AE%BE%E8%AE%A1%E6%A8%A1%E5%BC%8F&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/xielongfeiabcd/article/details/_blank)，组件的运行要有一个完整的Android工程环境，在这个环境下，Activity、Service等系统组件才能够正常工作，而这些组件并不能采用普通的Java对象创建方式，new一下就能创建实例了，而是要有它们各自的上下文环境，也就是我们这里讨论的Context。可以这样讲，Context是维持Android程序中各组件能够正常工作的一个核心功能类。

源码中的注释是这么来解释Context的：Context提供了关于应用环境全局信息的接口。它是一个抽象类，它的执行被Android系统所提供。它允许获取以应用为特征的资源和类型，是一个统领一些资源（应用程序环境变量等）的上下文。

Activity，Application，Service虽都继承自ContextWrapper（Activity继承自ContextWrapper的子类ContextThemeWrapper），但它们初始化的过程中都会创建ContextImpl对象，由ContextImpl实现Context中的方法。

Broadcast Receiver，Content Provider并不是Context的子类，他们所持有的Context都是其他地方传过去的，所以并不计入Context总数。

绝大多数场景下，Activity、Service和Application这三种类型的Context都是可以通用的。不过有几种场景比较特殊，比如启动Activity，还有弹出Dialog。出于安全原因的考虑，Android是不允许Activity或Dialog凭空出现的，一个Activity的启动必须要建立在另一个Activity的基础之上，也就是以此形成的返回栈。而Dialog则必须在一个Activity上面弹出（除非是System Alert类型的Dialog）



Activity继承自ContextThemeWrapper，而Application和Service继承自ContextWrapper，很显然ContextThemeWrapper在ContextWrapper的基础上又做了一些操作使得Activity变得更强大

一句话总结：凡是跟UI相关的，都应该使用Activity做为Context来处理；其他的一些操作，Service,Activity,Application等实例都可以，当然了，注意Context引用的持有，防止内存泄漏。

getApplication()和getApplicationContext()的区别

Application本身就是一个Context，所以这里获取getApplicationContext()得到的结果就是Application本身的实例。实际上这两个方法在作用域上有比较大的区别。getApplication()方法的语义性非常强，一看就知道是用来获取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()方法了。

如何避免Context造成内存泄漏

当Context销毁的时候，却因为被引用导致销毁失败，而Application的Context对象可以理解为随着进程存在的，所以我们总结出使用Context的正确姿势：

1：当Application的Context能搞定的情况下，并且生命周期长的对象，优先使用Application的Context。

2：不要让生命周期长于Activity的对象持有到Activity的引用。

3：尽量不要在Activity中使用非静态内部类，因为非静态内部类会隐式持有外部类实例的引用，如果使用静态内部类，将外部实例引用作为弱引用持有。

getContext(),getBaseContxet(),getApplicationContext的区别

Activity,Service和Application都有getBaseContxet(),getApplicationContext()这两个方法。但没有getContext方法。在Fragment中，才有getContext方法。而Fragment的getContext方法获取的对象是他的寄主对象，也就是Activity。