

Application类

课程目标

了解Application类的相关知识，掌握通过扩展Application类实现维护应用全局状态的方法。

学习内容

- **Application类简介**
- **自定义Application类**
- **Application对象的生命周期和回调函数**
- **Application对象的作用**
- **Application对象 vs. 静态单例**

Application类简介

Application是维护应用全局状态的基类。Android系统会在启动应用进程时创建一个Application对象。

```
<application
    android:icon= "@mipmap/ic_launcher"
    android:label= "@string/app_name"
    android:theme= "@style/AppTheme" >
    <activity android:name= ".MainActivity">
        ...
    </activity>
</application>
```

Application类简介

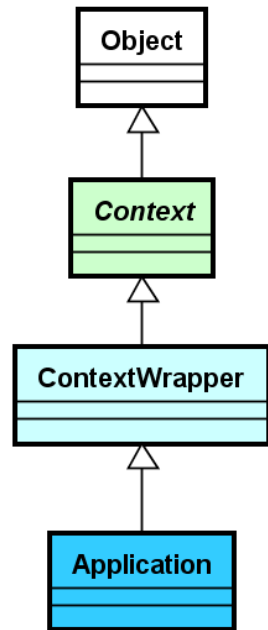
Application是维护应用全局状态的基类。Android系统会在启动应用进程时创建一个Application对象。

```
<application
    android:icon= "@mipmap/ic_launcher"
    android:label= "@string/app_name"
    android:theme= "@style/AppTheme" >
    <activity android:name= ".MainActivity">
        ...
    </activity>
</application>
```

Application类简介

Application是维护应用全局状态的基类。Android系统会在启动应用进程时创建一个Application对象。

```
<application
    android:icon= "@mipmap/ic_laun
    android:label= "@string/app_na
    android:theme= "@style/AppThem
    <activity android:name= ".Main
    ...
    </activity>
</application>
```



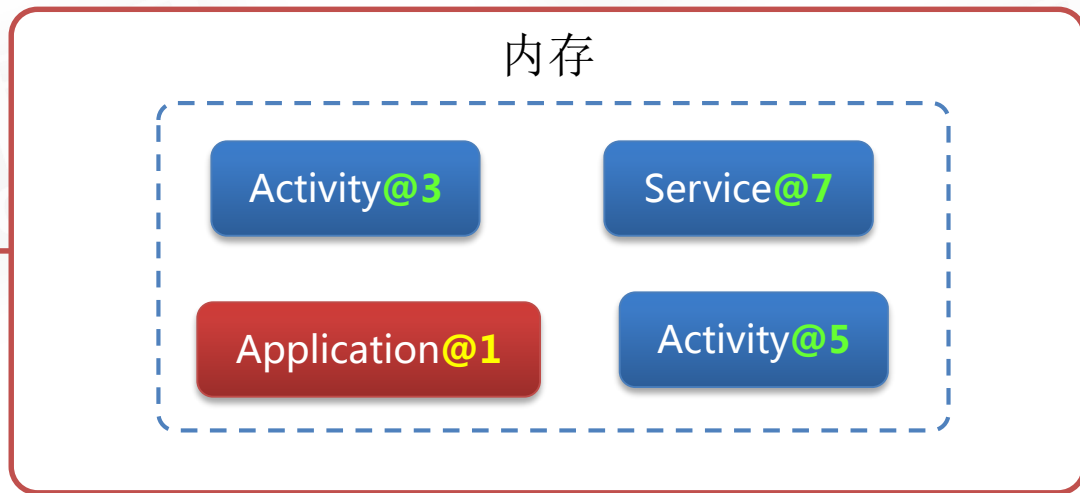
自定义Application类

我们可以扩展Application类，让Android系统使用我们自定义的Application类来创建Application对象。

- 创建Application子类
- 在清单文件中为 `application` 标签添加 `android:name` 属性

Application对象的生命周期

Application对象诞生于其它任何组件对象之前，并且一直存活，直到应用进程结束。



Application对象的回调函数

Application对象由Android系统管理，它的回调函数都运行于UI线程。

- onCreate
- onConfigurationChanged
- onLowMemory

Application对象的作用

Application对象全局可访问，且全程陪同应用进程。所以特别适合完成以下任务：

- **共享全局状态**
- **初始化全应用所需的服务**

静态单例

静态单例也可以实现应用全局状态的共享，以共享otto的Bus对象为例，我们分别

- 用Application对象实现共享
- 用静态单例实现共享

Application对象 vs. 静态单例

- 静态单例模块化程度更好
- Application就是一个context，所以有访问资源的能力
- 静态单例可以接受context参数
- Application对象能接收系统回调，自动知悉系统环境变化
- Application对象的生命周期由系统控制

小结

- 什么是Application类？
- 如何自定义Application类？
- Application对象的生命周期和回调函数