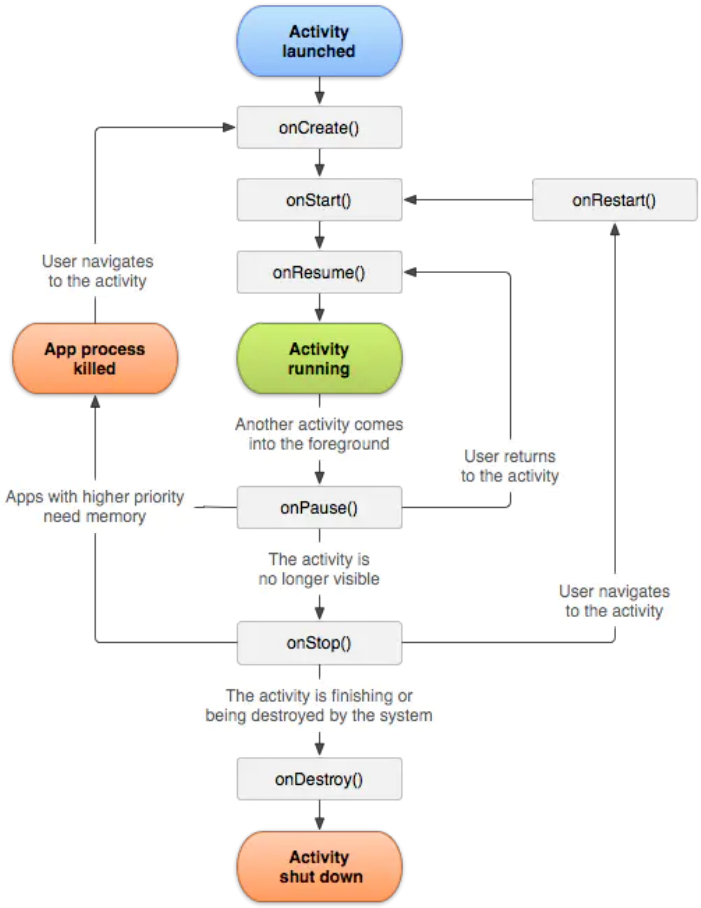
Activity生命周期之onStop 何种情况不被调用。

feifei_fly 关注
2018.05.13 18:31:23 字数 794 阅读 2,903



image

什么是生命周期

典型的生命周期就是在有用户参与的情况下，Activity经历从创建，运行，停止，销毁等正常的生命周期过程

- onCreate : 该方法是在Activity**被创建时回调**，它是生命周期第一个调用的方法，我们在创建Activity时一般都需要重写该方法，然后在该方法中做一些初始化的操作，如通过setContentView设置界面布局的资源，初始化所需要的组件信息等。
- onStart : Activity 为可见状态，但是尚未在前台显示（无法与用户交互）
- onResume : 当此方法回调时，则说明Activity已在前台可见，可与用户交互了（处于前面所说的Active/Running形态）。onResume方法与onStart的相同点是两者都表示Activity可见，只不过onStart回调时Activity还是后台无法与用户交互，而onResume则已显示在前台，可与用户交互。当然从流程图，我们也可以看出当Activity停止后（onPause方法和onStop方法被调用），重新回到前台时也会调用onResume方法，因此我们也可以在onResume方法中初始化一些资源，比如重新初始化在onPause或者onStop方法中释放的资源。
- onPause : 此方法被回调时则表示Activity正在停止（Paused形态），用户退出前台显示，可见但不能与用户进行交互。
- onStop : Activity被完全覆盖（不可见，仅在后台运行）。 表示Activity即将停止或者完全被覆盖（Stopped形态）

- onRestart :表示Activity正在重新启动，当Activity由不可见变为可见状态时，该方法被回调。这种情况一般是用户打开了一个新的Activity时，当前的Activity就会被暂停（onPause和onStop被执行了），接着又回到当前Activity页面时，onRestart方法就会被回调。
- onDestroy :此时Activity正在被销毁，也是生命周期最后一个执行的方法，一般我们可以在此方法中做一些回收工作和最终的资源释放。

在开发工程发现 onStop()有时不被回调。

什么时候不被回调呢?

Activity 不被完全覆盖的情况
ActivityA 跳转到Activity B,Activity A 再以下情况不被回调
(1)B 的theme 被设置为 android:theme="@android:style/Theme.Dialog"，以对话框的形式显示的时候。被覆盖的A 不会调用onStop()

(2)B 的theme被设置成 Android:theme="@android:style/Theme.Translucent.NoTitleBar" 或者 Android:theme="@android:style/Theme.Translucent.NoTitleBar"。
被覆盖的Activity 不会调用onStop

(3)B 的theme 被设置为 android:theme中设置了android:windowIsTranslucent 为true， (透明的,遮不住下面的Activity),被遮住的ActivityA 不会调用onStop()

```
<style name="AppTheme" parent="android:Theme.Black.NoTitleBar">
      <item name="android:windowBackground">@android:color/transparent</item>
      <item name="android:windowIsTranslucent">true</item>
      <item name="android:activityOpenEnterAnimation">@anim/in_from_left</item>
      <item name="android:activityOpenExitAnimation">@anim/out_from_right</item>
      <item name="android:activityCloseEnterAnimation">@anim/in_from_right</item>
      <item name="android:activityCloseExitAnimation">@anim/out_from_left</item>
  </style>
```

Acitvity A **跳转到**Activity B**的过程，遵循先创建 后销毁的原则。**

ActivityA.onCreate()
ActivityA.onStart()
ActivityA.onResume()
ActivityA.onPause()

ActivityB.onCreate() //--先创建
ActivityB.onStart()
ActivityB.onResume()

ActivityA.onStop() //-- 后销毁