**Activity中四大启动模式**

在AndroidManifest.xml中 ，有一个默认的activity 在它里面可以设置activity启动模式， android:launchMode="“ ,该属性用于配置Activity的加载模式，该属性支持4中属性 每不同的模式出现不同的效果，下面详解启动模式。

**standard:标准模式,默认加载模式**

**singleTop：Task顶单例模式  
 singleTask:Task内单例模式  
 singleInstance:全局单例模式**

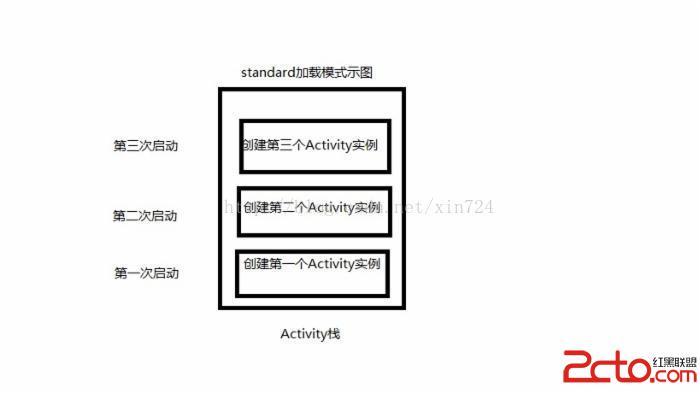
**1 Activity为什么要用指定模式？？**

首先介绍下Android对Activity的管理：Android采用Task来管理多个Activity，当我们启动一个Activity时，系统就会创建一个Task，然后启动这个Activity的入口。  
Android并未给Task提供API,只能通过调用Activity的getTaskId()方法获取它所在的Task的ID,我们可以把Task理解为Activity 栈，Task以栈来管理Activity。

**2.四大启动模式的理解**

**standard加载模式**：  
  
每次通过这种模式启动Activity时,Android总会为启动的Activity创建一个新的实例,并将该Activity添加到当前Task栈中，这种模式不会创建新的Task,只是将新  
Activity添加到原有的Task 中

例如：若我有一个Activity名为MainActivity, 上面有一个按钮可跳转到MainActivity。那么如果我点击按钮，便会新启一个MainActivity叠在刚才的MainActivity之上，再点击，又会再新启一个在它之上……点back键会依照栈顺序依次退出。



**singleTop模式**

如果在任务的栈顶正好存在该Activity的实例， 就重用该实例，否者就会创建新的实例并放入栈顶(即使栈中已经存在该Activity实例，只要不在栈顶，都会创建实例)。 为了防止出现一些奇怪的用户体验.推荐使用单一顶部模式.整个任务栈可以有多个实例存在.

例如：若我有两个Activity名为MainActivity,SecondActivity,两个Activity内容功能完全相同，都有两个按钮可以跳到MainActivity或者SecondActivity，唯一不同的是MainActivity为standard，SecondActivity为singleTop。  
若我意图打开的顺序为MainActivity->SecondActivity->SecondActivity，则实际打开的顺序为MainActivity->SecondActivity  
若我意图打开的顺序为MainActivity->SecondActivity->MainActivity->SecondActivity，则实际打开的顺序与意图的一致，为MainActivity->SecondActivity->MainActivity->SecondActivity。

**singleTask模式**  
  
  
被启动的Activity在同一个Task内只有一个Activity实例,具体分为如下三种情况：  
<1>.如果启动的目标Activity不存在Task栈中,系统将会创建一个目标Activity实例,并将它加入到Task栈顶  
<2>.如果启动的目标Activity已存在Task栈顶,重用该实例   
<3>.若果启动的目标Activity已存在但没有位于Task栈顶,系统将会把该目标Activity上面的所有Activity移除Task栈,使该Activity置于Task栈顶

例如：  
若我的应用程序中有三个Activity,MainActivity,SecondActivity,ThirdActivity，三个Activity可互相启动，其中SecondActivity为singleTask模式，那么，无论我在这个程序中如何点击启动，如：MainActivity->SecondActivity->ThirdActivity->SecondActivity->ThirdActivity->MainActivity-SecondActivity，MainActivity,ThirdActivity可能存在多个实例，但是SecondActivity只会存在一个，并且这三个Activity都在同一个task里面。  
但是MainActivity->SecondActivity->ThirdActivity->SecondActivity->ThirdActivity->MainActivity-SecondActivity，这样的操作过程实际应该是如下这样的，因为singleTask会把task中在其之上的其它Activity destory掉。  
操作：MainActivity->SecondActivity          MainActivity->SecondActivity->ThirdActivity          MainActivity->SecondActivity->ThirdActivity->SecondActivity            MainActivity->SecondActivity->ThirdActivity->SecondActivity->ThirdActivity->MainActivity             MainActivity->SecondActivity->ThirdActivity->SecondActivity->ThirdActivity->MainActivity-SecondActivity  
实际：MainActivity->SecondActivity          MainActivity->SecondActivity->ThirdActivity          MainActivity->SecondActivity                              MainActivity->SecondActivity->ThirdActivity->MainActivity                               MainActivity->SecondActivity

应用场景: 如果一个Activity非常消耗内存和cpu资源,建议把这个Activity做成singletask的模式.浏览器的browserActivity

**singleInstance模式**  
  
这种加载模式下,无论从哪个Task中启动目标Activity,只会创建一个目标Activity实例,并会使用一个全新的Task栈来装载该Activity实例。具体可分为两种情况：  
<1>.如果创建的目标Activity不存在,系统先会创建一个全新的Task,接着创建一个Activity实例,然后将该目标Activity加入到新的Task栈顶  
<2>.如果创建的目标Activity已经存在,无论在哪个Task栈中,系统将会把Activity所在的栈置于前台。

例如：  
程序有三个ActivityD1,D2,D3，三个Activity可互相启动，其中D2为singleInstance模式。那么程序从D1开始运行，假设D1的taskId为200，那么从D1启动D2时，D2会新启动一个task，即D2与D1不在一个task中运行。假设D2的taskId为201，再从D2启动D3时，D3的taskId为200，也就是说它被压到了D1启动的任务栈中。  
注意：采用singleInstance加载模式的Activity总是位于Task栈顶,并且Activity所在的Task栈只包含该Activity。

 整个手机操作系统只有一个实例存在,并且是运行在自己单独的任务栈里面.  
>通话界面的Activity