Activity通常展现为一个可视化的用户界面，是Android程序与用户交互的窗口，一个Activity占据当前的窗口，响应所有窗口事件，具备有控件，菜单等界面元素。从内部逻辑来看，Activity需要为了保持各个界面状态，需要做很多持久化的事情，还需要妥善管理生命周期，和一些转跳逻辑。

注意点：

1、首先一个应用程序可以有一个或者多个Activity的活动页面。

2、每一个应用程序都需要有一个activity入口，也就相当于程序的入口。唯一不同的他的入口地址需要在androidmianfest.xml中配置

3、启动一个activity的时候要跳转到另一个activity中就需要intent的组件。

我们开始讲解

1、第一个当我们创建好一个项目之后开始讲解。

问题一，mainactivity继承与activity这个父类，

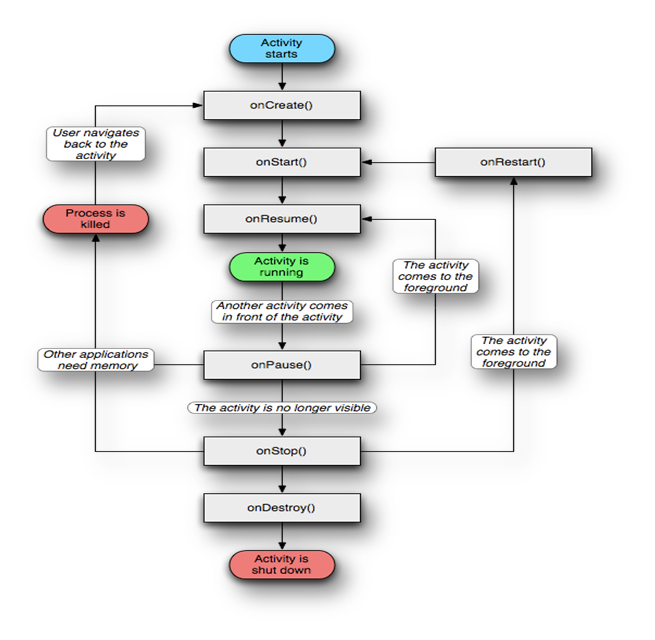
2、自动帮我重写了oncreate（）这个方法

3、setContentView(R.layout.activity\_main);

      这个是设置这个activity页面的视图与布局，讲界面与视图绑定，相当于相框与相片的关系。

4、除了oncreate方法之外还有其他的父类重写的方法。在Androidmianfest.xml进行配置

        除了oncreate之外还有onstart onResume onpause onstop ondestory onrestart



5、关于各个声明周期的执行过程

程序首先执行的是：oncreate，onstart，onresume三个方法。

 \* 点击home键：执行onpause onstop方法 处于暂停状态。

 \* 再次通过启动home键启动时：执行onrestart，onstart，onresume方法

 \* 点击退出键：onpause onstop ondestory方法

================================================

log的作用以及如何添加过滤标签来打印信息

//log 用来做标记的，常用来调试代码时使用打印信息判断代码的问题和错误。

==================================================

==================================================

**二，对于两个页面的跳转的功能**

**我们来看他们的生命周期的变化。**

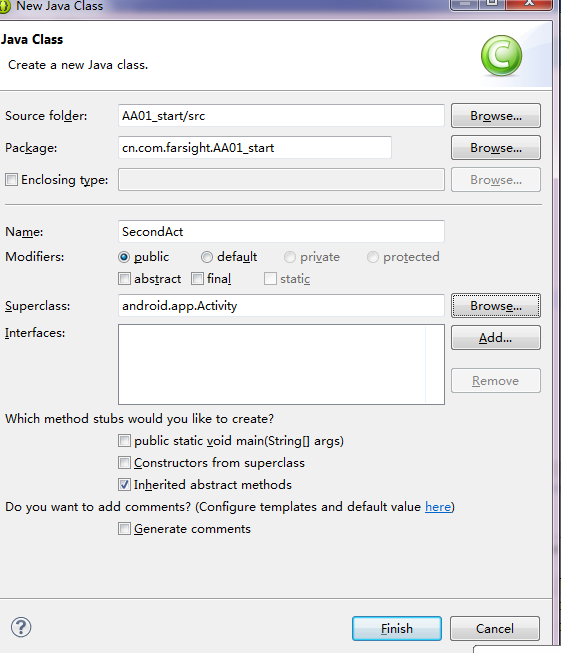
**首先我们**

**1、创建activity.java文件**

**在src上点击右键->new->class然后填出弹出这个框起名字**

**然后让他继承activity的类点击finish**

**、**



创建完成之后缺少他的入口方法即oncreate的方法

那么我们现在将他补全。

简单说一下这几个方法一般会做什么用法。

oncreate()方法用来寻找布局中控件的id 进行绑定，设定一些界面初始化的功能。

onresume()该方法用来处理与用户的交互，当onstart（）被调用后，onresume()也会自动被调用。较好

的习惯是将动画的开始、照相机等独占设备的开启等操作放在这个方法中实现。

onpause() 当activity 被置于后台切还没有被终止时调用，在系统内存不足时会被终止。具体来说，就是当activity被一个透明

或类似于对话框的activity覆盖时调用，比如突然接到了来电。调用完成后该activity任然被窗口管理器所维护，所以它仍然可见

，只是失去了焦点，所以不能在于用户交互。

onstop():当activity不在被用户可见时调用，也就是一个新的activity覆盖了这个activity。在系统需要内存时终止该activity

ondestory():当Acrtivity被终止前调用，完成最后的清理工作。finish方法可以直接调用该方法。

onrestart(): 当Activity调用onstop()被停止后又重新被用户可见时调用，比如用户手动单击“返回键”返回到这个activity。

Activity的四中基本状态

1、running 运行态    调用完onresume()后处于该状态

2、paused 暂停态      调用完onpause()后处于该状态

3、 stopped 停止态    调用完onstoped()后处于该状态

4、Destroyed 终止态  调用完ondestory()后处于该状态

@see Activity的加载模式：节约应用程序的内存,优化内存

  \* @see Standard,singletop,singletask,singleinstance

  \* @see Standard:有多少启动多少 （标准模式） 默认不加是是

  \* @see Singletop：谁来上面启动谁（顶单例模式）

  \* @see SingleTask:谁在你上面干掉谁（内单例模式）

  \* @see Singleinstance：管你是谁（全局单例模式）

  \*      当第一个页面为singleinstance时候后面跳转时，会出现两个栈顶之间跳转。

  \*      当第一个页面不是singleinstance时候后面跳转，其他页面再次去创建。

**代码地址**

<http://yunpan.cn/cmACB9ncx9qTG>  访问密码 fc2f

[Service（服务）.note](note://9DCBF5A75CD243AE8C33BB5DE7CFE776)