跟踪Activity的源码就会发现：

Activity.attch() -> PolicyManager -> Policy -> PhoneWindow -> mLayoutInflater.inflate()&mContentParent.addView()

这只是一个简单的跟踪过程描述。通过跟踪源代码，就可以很清晰的看出他们三者的关系。

Activity像一个工匠（控制单元），Window像窗户（承载模型），View像窗花（显示视图）

LayoutInflater像剪刀，Xml配置像窗花图纸。

要点：

1. 一个Activity构造的时候一定会构造一个Window（准确的说是一个PhoneWindow）

此Window有且只有一个。

1. 这个Window有一个ViewRoot【DecorView】，它是一个逻辑性的根视图，实际上就是一个View类或者是一个ViewGruop类
2. 那么，在这个ViewRoot的根视图上面就会添加一个一个的View，也就是执行addView()方法，来达到我们最终显示的效果。
3. WindowManagerService接收消息，并且回调Activity函数，比如典型的onKeyDown()

总的来说，Activity实际上就是一个控制单元，它就像一个工匠一样，在它实例化的时候会创造一个Window（Only One）。

而对于Window，它是我们视图的承载模型。

Window本身并不是显式的视图，就如同我们的窗户上的玻璃，它只是我们贴窗花的地方，不是真正的视图。我们接着比喻。我们要有一个漂亮的窗户就要贴窗花，而我们的每一个窗花都是一个View。下面的问题是，我们怎么去修剪这个窗花使它变得漂亮呢？这里就 用到了我们的Layout，也就是布局。要把窗花贴到窗户上，就要通过LayoutInflater和addView()来执行这个贴的过程。

对于View，它才是我们真正的显式视图。

