 android:windowSoftInputMode属性一共有9个取值，分别是：

    stateUnspecified，stateUnchanged，stateHidden，stateAlwaysHidden，stateVisible，stateAlwaysVisible，adjustUnspecified，adjustResize，adjustPan。

**1.stateUnspecified**

    未指定状态，当我们没有设置android:windowSoftInputMode属性的时候，软件默认采用的就是这种交互方式，系统会根据界面采取相应的软键盘的显示模式，比如，当界面上只有文本和按钮的时候，软键盘就不会自动弹出，因为没有输入的必要。但是，如果我们在这个布局的外面，包裹上一个

ScrollView或者添加ListVIew，软键盘就会自动的弹出来。因此，当设置属性为stateUnspecified的时候，系统是默认不弹出软键盘的，但是当有获得焦点的输入框的界面有滚动的需求的时候，会自动弹出软键盘。

**2.stateUnchanged**

   状态不改变，大概意思就是当前界面的软键盘状态，取决于上一个界面的软键盘状态。

**3.stateHidden**

   键盘状态一定是隐藏，不管上个界面什么状态，也不管当前界面有没有输入的需求，反正就是不显示。因此，我们设置这个属性，可以来控制软键盘不自动的弹出。

**4.stateVisible**

    键盘显示，假如界面没有输入框的情况下也显示出来。

**5.stateAlwaysVisible**

    这个属性也可以将键盘显示出来，但是与stateVisible属性有不同之处。举个例子，当我们设置为stateVisible属性，如果当前的界面键盘是显示的，当我们点击按钮跳转到下个界面的时候，软键盘会因为输入框失去焦点而隐藏起来，当我们再次回到当前界面的时候，键盘这个时候是隐藏的。但是如果我们设置为stateAlwaysVisible，我们跳转到下个界面，软键盘还是隐藏的，但是当我们再次回来的时候，软键盘是会显示出来的。

**6.adjustUnspecified**

    从这个属性开始，就不是设置软键盘的显示与隐藏模式了，而是设置软键盘与软件的显示内容之间的显示关系。当你跟我们没有设置这个值的时候，这个选项也是默认的设置模式。在这中情况下，系统会根据界面选择不同的模式。如果界面里面有可以滚动的控件，比如ScrowView，系统会减小可以滚动的界面的大小，从而保证即使软键盘显示出来了，也能够看到所有的内容。如果布局里面没有滚动的控件，那么软键盘可能就会盖住一些内容，我们从下面的图中可以看出差别。

   没有滚动控件，软键盘下面的布局都被遮挡住了，若想修改，只能隐藏软键盘，然后选择。而且，重点注意一下上面的布局，当我们选择的输入框偏下的时候，上面的标题栏和布局被软键盘顶上去了。记住这个特征，因为后面有个属性和这个的效果不一样。

布局里面有滑动控件，系统会自动的缩小整个界面的大小，因此，我们可以软键盘上面的小区域中显示所有的输入框。

    这就是两中显示模式之间的差别。

**8.adjustResize**

    这个属性表示Activity的主窗口总是会被调整大小，从而保证软键盘显示空间。

    我们先看显示效果。

    注意观察这个上面的标题栏和按钮，设置为adjustResize属性之后，对于没有滑动控件的布局，虽然还是不能选择所有的输入框，但是，窗口的显示方式发生了变化，默认属性时，整个布局是被顶上去了，但是设置为adjustResize属性，布局的位置并没有发生什么变化，这就是最大的区别。

而对于有滑动控件的布局来说，显示效果和默认是一样的。

**9.adjustPan**

    如果设置为这个属性，那么Activity的屏幕大小并不会调整来保证软键盘的空间，而是采取了另外一种策略，系统会通过布局的移动，来保证用户要进行输入的输入框肯定在用户的失业范围里面，从而让用户可以看到自己输入的内容。对于没有滚动控件的布局来说，这个其实就是默认的设置，如果我们选择的位置偏下，上面的标题栏和部分控件会被顶上去。但是对于有滚动控件的布局来说，则不太一样，我们看下面的效果图。

    首先，这是软键盘没有弹出的时候，有滚动控件的显示范围，最下面显示的是9.

    当我们点击5这个输入框，我们会发现下面的现象。

    最上面只能够显示到按钮，标题栏已经不能看到了。

    而最下面也只能滑动到8，下面的内容也不能够滑动了。

   因此，我们就能够理解这个属性的作用了。

    通过以上的实验，我们可以得出结论，如果我们不设置"adjust..."的属性，对于没有滚动控件的布局来说，采用的是adjustPan方式，而对于有滚动控件的布局，则是采用的adjustResize方式。

以上内容参考：http://www.jb51.net/article/56043.htm

了解了windowSoftInputMode属性后，我们再说一下如何让EditText不自动获取焦点，其实方法很简单，在EditText父级空间中，设置如下属性即可

android:focusable="true"

 android:focusableInTouchMode="true"

这样就对EditText默认行为做了截断

一、打开输入法窗口:

InputMethodManager inputMethodManager = (InputMethodManager)              getSystemService(Context.INPUT\_METHOD\_SERVICE);

// 接受软键盘输入的编辑文本或其它视图

imm.showSoftInput(submitBt,InputMethodManager.SHOW\_FORCED);

二、关闭出入法窗口

InputMethodManager inputMethodManager = (InputMethodManager)              getSystemService(Context.INPUT\_METHOD\_SERVICE);

inputMethodManager.hideSoftInputFromWindow(OpeListActivity.this.getCurrentFocus().getWindowToken(),

InputMethodManager.HIDE\_NOT\_ALWAYS);

//接受软键盘输入的编辑文本或其它视图

inputMethodManager

.showSoftInput(submitBt,InputMethodManager.SHOW\_FORCED);

三、如果输入法打开则关闭，如果没打开则打开

InputMethodManager m=(InputMethodManager) getSystemService(Context.INPUT\_METHOD\_SERVICE);

m.toggleSoftInput(0, InputMethodManager.HIDE\_NOT\_ALWAYS);

四、获取输入法打开的状态

InputMethodManager imm = (InputMethodManager)getSystemService(Context.INPUT\_METHOD\_SERVICE);  
boolean isOpen=imm.isActive();  
isOpen若返回true，则表示输入法打开