**Android EditText光标定位**

|  |
| --- |
| * Selection 这个完全是一个工具类, 是关于区间选取的工具类, 自有的方法都是静态的. 来了兴致就把所有方法都测试了一下, 又要熬夜了. 测试是在一个继承EditText 的子类  onTouchEvent 方法里面进行的. 算是一个总结了. * text 实参是 getEditableText(), layout 实参是 getLayout() * **选中区域的方法:** * **Selection.extendDown(Spannable text ,Layout layout); //当前光标的坐标, 选中到下一行的(X轴)位置**              1 2 3 4光标 5 6 7 8 9             1 2 3 4       5 6 7 8 9   * **Selection.extendUp(Spannable text ,Layout layout); //当前光标的坐标, 选中到上一行的(X轴)位置**              1 2 3 4       5 6 7 8 9             1 2 3 4光标 5 6 7 8 9   * **Selection.extendLeft(Spannable text ,Layout layout); //当前光标的坐标, 选中到x-1的位置**              1 2 3   4   5 6 7 8 9             1 2 3 4光标 5 6 7 8 9   * **Selection.extendRight(Spannable text ,Layout layout); //当前光标的坐标, 选中到x+1的位置**              1 2 3 4       5 6 7 8 9             1 2 3 4光标 5 6 7 8 9   * **Selection.extendToLeftEdge(Spannable text ,Layout layout); //当前光标的坐标, 选中到同行的最左边的位置**              1 2 3 4       5 6 7 8 9             1 2 3 4光标 5 6 7 8 9   * **Selection.extendToRightEdge(Spannable text ,Layout layout);//当前光标的坐标, 选中到同行的最右边的位置**              1 2 3 4       5 6 7 8 9             1 2 3 4光标 5 6 7 8 9   * **跳转光标的方法:** * **Selection.moveDown(Spannable text ,Layout layout);////当前光标的坐标, 跳转到下一行的(X轴)的位置**              1 2 3 4 原光标 5 6 7 8 9             1 2 3 4 新光标 5 6 7 8 9   * **还有moveUp, moveLeft,  moveToLeftEdge等, 结合上面其他方法的说明, 大家应该都懂得! 这里省略了.**        * **指定位置的方法, 全选和取消:** * **Selection.setSelection(Spannable text, int index); //当前光标的坐标, 跳转到索引为index 的位置(索引从0开始), 注意不是moveSelection.**             如, Selection.setSelection(getEditableText(), 5);             1 2 3 4 原光标 5新光标 6 7 8 9             1 2 3 4           5          6 7 8 9   * **Selection.extendSelection(Spannable text,int index);  //当前光标的坐标, 选中到索引为index的位置(索引从0开始)**             如, Selection.extendSelection(getEditableText(), 5);             1 2 3 4 原光标 5 6 7 8 9             1 2 3 4           5 6 7 8 9   * **Selection.setSelection(Spannable text, int start, int stop)     //选中从索引start 到索引stop 的区域**             如, Selection.setSelection(getEditableText(), 5, 9);             1 2 3 4  5 6 7 8 9             1 2 3 4  5 6 7 8 9   * **Selection.selectAll(Spannable text);        //全选**              1 2 3 4 5 6 7 8 9             1 2 3 4 5 6 7 8 9   * **Selection.removeSelection(Spannable text);      //取消选中状态, 光标坐标重置到最前面**              1 2 3 4       5 6 7 8 9             ====>       光标1 2 3 4 5 6 7 8 9             1 2 3 4光标 5 6 7 8 9                                     1 2 3 4 5 6 7 8 9       * **int Selection.getSelectionEnd(CharSequence text)**    //text 实参是 getEditableText()  1. 如果没有选中区域,有光标, 返回光标所在的索引; 2. 如果有选中区域, 返回区域的结束索引; 3. 如果没有选中区域和光标, 返回-1.  * **int Selection.getSelectionStart(CharSequence text)**   //text 实参是 getEditableText()  1. 没有选中区域,有光标, 返回光标所在的索引; 2. 如果有选中区域, 返回区域的开始索引; 3. 如果没有选中区域和光标, 返回-1.        这两个方法EditText 也有的.  如何获取选中的文字呢? 靠的就是EditText 的这两个方法. 取一个开始索引start 和 结束索引 end, 判断它们是否相同, 不相同证明有文字被选中,**选中文字 = mEditText.getText().subSequence(start, end);** |