**Android软键盘监听事件**

参考：<http://blog.csdn.net/breeze666/article/details/27082419>

<http://blog.csdn.net/xcookies/article/details/43024911>

1. **软件盘监听的几种方法**
2. 重写Activity的dispatchKeyEvent(KeyEvent event)方法,，监听相应按钮按下事件：

@Override

public boolean dispatchKeyEvent(KeyEvent event) {

if(event.getKeyCode() == KeyEvent.KEYCODE\_ENTER){

/\*隐藏软键盘\*/

InputMethodManager imm= (InputMethodManager)

getSystemService(Context.INPUT\_METHOD\_SERVICE);

if(imm.isActive()){

imm.hideSoftInputFromWindow(

MainActivity.this.getCurrentFocus().getWindowToken(), 0);

}

edittext.setText("success");

webview.loadUrl(URL);

return true;

}

return super.dispatchKeyEvent(event);

}

1. 使用OnKeyListener的方法来监听软键盘按键

private OnKeyListener onKeyListener = new OnKeyListener() {

@Override

public boolean onKey(View v, int keyCode, KeyEvent event) {

if(keyCode == KeyEvent.KEYCODE\_ENTER

&& event.getAction() == KeyEvent.ACTION\_DOWN){

/\*隐藏软键盘\*/

InputMethodManager imm = (InputMethodManager)

getSystemService(Context.INPUT\_METHOD\_SERVICE);

if(imm.isActive()){

imm.hideSoftInputFromWindow(

v.getApplicationWindowToken(), 0);

}

edittext.setText("success");

webview.loadUrl(URL);

return true;

}

return false;

}

};

edittext.setOnKeyListener(onKeyListener);

1. 使用onTouchEvent的方法来监听软键盘按键

@Override

public boolean onTouchEvent(MotionEvent event) {

if (indexOfChild(editText) > -1) {

// 可添加抛出收起事件代码

}

return super.onTouchEvent(event);

}

1. 使用ViewTreeObserver.OnGlobalLayoutListener从布局高度的变化来判断软键盘的打开和收起事件

RelativeLayout detailMainRL = (RelativeLayout)

findViewById(R.id.home\_news\_detail\_main\_rl);

detailMainRL.getViewTreeObserver().addOnGlobalLayoutListener(

new OnGlobalLayoutListener(){

@Override

public void onGlobalLayout() {

int heightDiff = detailMainRL.getRootView().getHeight()

- detailMainRL.getHeight();

Log.v(TAG, "detailMainRL.getRootView().getHeight() = "

+ detailMainRL.getRootView().getHeight());

Log.v(TAG, "detailMainRL.getHeight() = "

+ detailMainRL.getHeight());

if (heightDiff > 100) { // 说明键盘是弹出状态

Log.v(TAG, "键盘弹出状态");

commentBoxRL.setVisibility(View.VISIBLE);

} else{

Log.v(TAG, "键盘收起状态");

commentBoxRL.setVisibility(View.GONE);

}

}

});

1. 自定义布局管理器，监听布局变化。（如第二部分所示）
2. **自定义布局管理器，监听布局变化。**
3. 在AndroidManifet.xml中对相应的Activity进行设置。如：android:windowSoftInputMode="stateHidden|adjustResize"
4. 自定义一个布局管理器，一般继承LinearLayout或RelativeLayout。

\* 定义布局类，继承RelativeLayout类

\* 用于监听布局大小改变事件

\*/

public class ResizeLayout extends RelativeLayout {

private OnResizeListener mListener;

public interface OnResizeListener {

void OnResize(int w, int h, int oldw, int oldh);

}

/\*\*

\* 定义一个监听器，用于监听布局大小的改变

\*/

public void setOnResizeListener(OnResizeListener listener) {

mListener = listener;

}

public ResizeLayout(Context context, AttributeSet attrs) {

super(context, attrs);

}

/\*\*

\* @param width 当前宽度

\* @param height 当前高度

\* @param oldWidth 原始宽度

\* @param oldHeight 原始高度

\*/

@Override

protected void onSizeChanged(int width, int height,

int oldWidth, int oldHeight) {

super.onSizeChanged(width, height, oldWidth, oldHeight);

if (mListener != null) {

mListener.OnResize(width, height, oldWidth, oldHeight);

}

}

}

1. 布局文件，将自定义布局管理器作为布局文件的根节点

<?xml version="1.0" encoding="utf-8"?>

<com.example.zxg.myprogram.widge.scalescrollview.ResizeLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

android:id="@+id/rl\_scroll"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" >

</com.example.zxg.myprogram.widge.scalescrollview.ResizeLayout>

1. 调用，实现布局监听

ResizeLayout rl\_layout = (ResizeLayout) findViewById(R.id.rl\_scroll);

rl\_layout.setOnResizeListener(new ResizeLayout.OnResizeListener() {

public void OnResize(int w, int h, int oldw, int oldh) {

if(h != oldh)

...

}

});

1. **AndroidManifet.xml中Activity的android:windowSoftInputMode属性**
2. stateUnspecified：软键盘的状态并没有指定，系统将选择一个合适的状态或依赖于主题的设置
3. stateUnchanged：当这个activity出现时，软键盘将一直保持在上一个activity里的状态，无论是隐藏还是显示
4. stateHidden：用户选择activity时，软键盘总是被隐藏
5. stateAlwaysHidden：当该Activity主窗口获取焦点时，软键盘也总是被隐藏的
6. stateVisible：软键盘通常是可见的
7. stateAlwaysVisible：用户选择activity时，软键盘总是显示的状态
8. adjustUnspecified：默认设置，通常由系统自行决定是隐藏还是显示
9. adjustResize：该Activity总是调整屏幕的大小以便留出软键盘的空间
10. adjustPan：当前窗口的内容将自动移动以便当前焦点从不被键盘覆盖和用户能总是看到输入内容的部分

**注**：windowSoftInputMode设置必须是下面列表中的一个值，或一个”state…”值加一个”adjust…”值的组合。