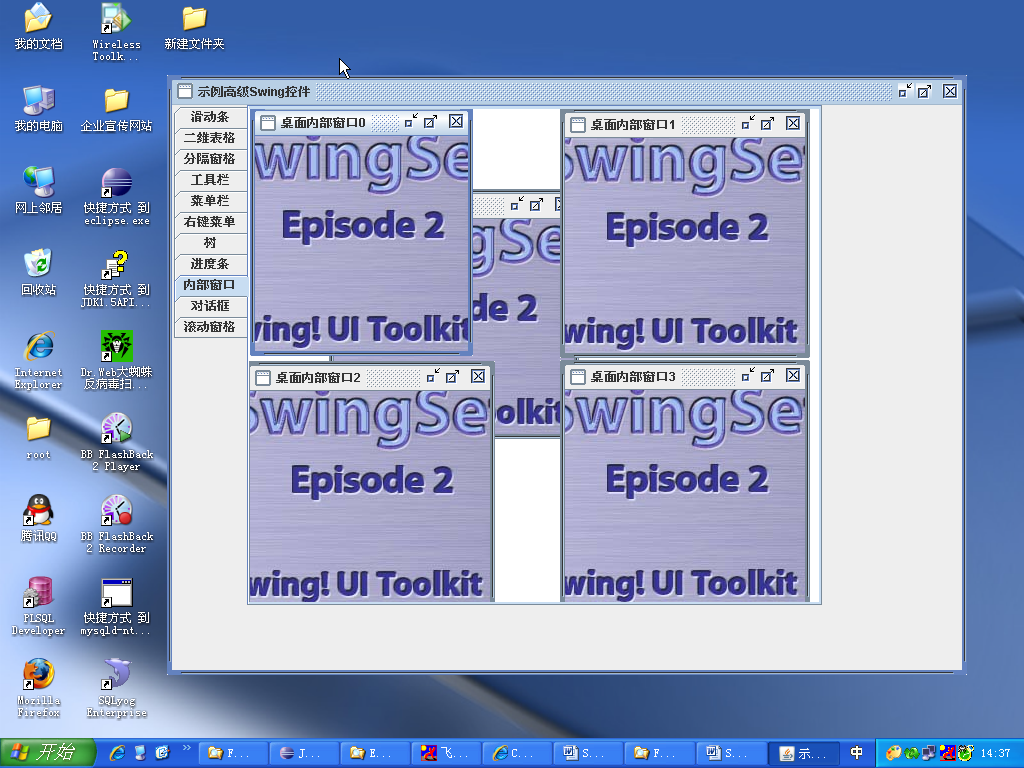
这篇文章给大家演示一下Swing中的高级控件的用法，为了方面大家入门，我们的示例将会尽可能的简化。Swing中的高级控件包括JTabbedPane（页签）,JPanel（面板）,JDesktopPane（桌面窗格）,JInternalFrame（内部窗口）,JProgressBar（进度条）,JTree（树）,JMenuBar（菜单栏）,JPopupMenu（右键菜单）,JToolBar（工具栏）,JSplitPane（分隔窗格）,JTable（表格）,JSlider（滑动栏）等。运行效果如下：



代码如下：

**package** org.leno.swing.demo1;

**import** [**Java**](http://lib.csdn.net/base/java).awt.\*;

**import** java.awt.event.\*;

**import** java[**.NET**](http://lib.csdn.net/base/dotnet).URL;

**import** javax.swing.\*;

**import** javax.swing.border.\*;

**import** javax.swing.tree.\*;

/\*\*

 \* **@author** leno

 \* **@version** 1.0

 \* 一个例子程序，让大家系统地了解Swing中一些高级控件(JTabbedPane,JPanel,JDesktopPane

 \* ,JInternalFrame,JProgressBar,JTree,JMenuBar,JPopupMenu,JToolBar,JSplitPane

 \* ,JTable,JSlider等)的基本用法，这里只是来个大杂烩，在实际应用中要注意取舍，灵活机变。

 \* （http://blog.csdn[**.net**](http://lib.csdn.net/base/dotnet)/lenotang 转载请保留相关出处信息!）

 \*/

**public** **class** DemoSuperComponent {

**public** **static** Dimension *HGAP5* = **new** Dimension(5, 1);

**public** **static** Dimension *VGAP5* = **new** Dimension(1, 5);

**public** **static** Dimension *HGAP25* = **new** Dimension(25, 1);

**public** **static** Dimension *VGAP25* = **new** Dimension(1, 25);

       @SuppressWarnings("serial")

**public** **static** **void** main(String[] args) {

              // 创建一个主窗体

              JFrame jFrame = *getMainFrame*();

              // 获得窗体的内容窗格

[**Container**](http://lib.csdn.net/base/docker) contentPane = jFrame.getContentPane();

              // 创建一个选项面板(也叫页签）

              JTabbedPane jTabbedPane = *getMainTabbedPane*();

              // 创建一个放置滑动条的面板

              JPanel jPanel1 = *getSliderPanel*();

              // 创建一个放置二维表格的面板

              JPanel jPanel2 = *getTablePanel*();

              // 创建一个放置分隔窗格的面板

              JPanel jPanel3 = *getSplitPanePanel*();

              // 创建一个放置工具栏的面板

              JPanel jPanel4 = *getToolBarPanel*();

              // 创建一个放置菜单栏的面板

              JPanel jPanel5 = *getMenuBarPanel*();

              // 创建一个放置右键菜单的面板

              JPanel jPanel6 = *getPopupMenuPanel*();

              // 创建一个放置树结构的面板

              JPanel jPanel7 = *getTreePanel*();

              // 创建一个放置进度条的面板

              JPanel jPanel8 = *getProgressBarPanel*();

              // 创建一个放置内部窗体的面板(带桌面）

              JPanel jPanel9 = *getInternalFramesPanel*(jTabbedPane);

              // 创建一个放置对话框的面板

              JPanel jPanel10 = *getDialogPanel*();

              // 创建一个放置滚动窗格的面板

              JPanel jPanel11 = *getScrollPanel*();

              // 将创建的这些面板都放入到一个选项面板中

              jTabbedPane.add("滑动条", jPanel1);

              jTabbedPane.add("二维表格", jPanel2);

              jTabbedPane.add("分隔窗格", jPanel3);

              jTabbedPane.add("工具栏", jPanel4);

              jTabbedPane.add("菜单栏", jPanel5);

              jTabbedPane.add("右键菜单", jPanel6);

              jTabbedPane.add("树", jPanel7);

              jTabbedPane.add("进度条", jPanel8);

              jTabbedPane.add("内部窗口", jPanel9);

              jTabbedPane.add("对话框", jPanel10);

              jTabbedPane.add("滚动窗格", jPanel11);

              // 将选项面板组件放置在容器中(充分体现组件放在容器中,容器又是组件的思想)

              contentPane.add(jTabbedPane);

              // 设置窗体可见

              jFrame.setVisible(**true**);

       }

**private** **static** JPanel getScrollPanel() {

              JPanel jPanel11 = **new** JPanel();

              jPanel11.setLayout(**null**);

              Object[] privs = **new** Object[]{"增加用户","删除用户","修改用户","预览用户"};

              JList jList = **new** JList(privs);

              JScrollPane jScrollPane = **new** JScrollPane();

              jScrollPane.setBounds(200, 50, 120, 60);

              jScrollPane.getViewport().add(jList);

              jPanel11.add(jScrollPane);

**return** jPanel11;

       }

**private** **static** JFrame getMainFrame() {

              // 设置美化外观

              JFrame.*setDefaultLookAndFeelDecorated*(**true**);

              // 创建一个窗体

              JFrame jFrame = **new** JFrame("示例高级Swing控件");

              // 设置窗体的位置和大小

              jFrame.setSize(800, 600);

              // 取得屏幕大小和窗口大小

              Dimension screenSize = Toolkit.*getDefaultToolkit*().getScreenSize();

              Dimension frameSize = jFrame.getSize();

              // 设置窗口能够在当前屏幕居中显示

**if** (frameSize.width > screenSize.width) {

                     frameSize.width = screenSize.width;

              }

**if** (frameSize.height > screenSize.height) {

                     frameSize.height = screenSize.height;

              }

              jFrame.setLocation((screenSize.width - frameSize.width) / 2,

                            (screenSize.height - frameSize.height) / 2);

              // 设置手动布局，这样意味着放在jFrame内容窗格上的组件都要指定自身的位置和大小，否则将不会显示。JFrame默认布局方式为BorderLayout

              jFrame.setLayout(**null**);

              // 设置窗体的关闭行为

              jFrame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);

**return** jFrame;

       }

**private** **static** JTabbedPane getMainTabbedPane() {

              JTabbedPane jTabbedPane = **new** JTabbedPane();

              jTabbedPane.setTabPlacement(JTabbedPane.*LEFT*);

              jTabbedPane.setBounds(0, 0, 650, 500);

**return** jTabbedPane;

       }

**private** **static** JPanel getDialogPanel() {

**final** JPanel jPanel10 = **new** JPanel();

              JButton jButton1 = **new** JButton("输入对话框");

              jButton1.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

                            JOptionPane.*showInputDialog*(jPanel10, "请输入您的姓名？");

                     }

              });

              JButton jButton2 = **new** JButton("警告对话框");

              jButton2.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

                            JOptionPane

                                          .*showMessageDialog*(

                                                        jPanel10,

                                                        "<html><h1><span style=/"color:red/">HTML格式的警告内容！</span></h1></html>",

                                                        "紧急通知", JOptionPane.*WARNING\_MESSAGE*);

                     }

              });

              JButton jButton3 = **new** JButton("确认对话框");

              jButton3.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

**int** result = JOptionPane.*showConfirmDialog*(jPanel10,

                                          "你认为自己足够优秀吗？");

**if** (result == JOptionPane.*YES\_OPTION*) {

                                   JOptionPane.*showMessageDialog*(jPanel10, "不错！");

                            } **else** **if** (result == JOptionPane.*NO\_OPTION*) {

                                   JOptionPane.*showMessageDialog*(jPanel10, "要自信！");

                            }

                     }

              });

              JButton jButton4 = **new** JButton("消息对话框");

              jButton4.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

                            URL img = getClass().getResource(

                                          "/resources/images/optionpane/bottle.gif");

                            String imagesrc = "<img src=/"" + img

                                          + "/" width=/"284/" height=/"100/">";

                            String message = "你快迟到了！";

                            JOptionPane.*showMessageDialog*(jPanel10, "<html>" + imagesrc

                                          + "<br><center>" + message + "</center><br></html>");

                     }

              });

              JButton jButton5 = **new** JButton("组件对话框");

              jButton5.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

                            JTextField t[] = **new** JTextField[4];

                            t[0] = **new** JTextField("输入姓名:");

                            t[0].setEditable(**false**);

                            t[1] = **new** JTextField();

                            t[2] = **new** JTextField("输入密码:");

                            t[2].setEditable(**false**);

                            t[3] = **new** JTextField();

                            String but[] = **new** String[] { "确定" };

**int** result = JOptionPane.*showOptionDialog*(**null**, t, "登陆信息",

                                          JOptionPane.*YES\_OPTION*,

                                          JOptionPane.*INFORMATION\_MESSAGE*, **null**, but, but[0]);

**if** (result == 0) {

                                   JOptionPane.*showMessageDialog*(jPanel10, "登陆成功");

                            }

                     }

              });

              jPanel10.add(jButton1);

              jPanel10.add(jButton2);

              jPanel10.add(jButton3);

              jPanel10.add(jButton4);

              jPanel10.add(jButton5);

**return** jPanel10;

       }

**private** **static** JPanel getInternalFramesPanel(JTabbedPane jTabbedPane) {

              JPanel jPanel9 = **new** JPanel();

              jPanel9.setLayout(**null**);

              // 创建桌面

              JDesktopPane desktop = **new** JDesktopPane();

              desktop

                            .setBounds(0, 0, jTabbedPane.getWidth(), jTabbedPane

                                          .getHeight());

              // 为桌面添加5个内部窗口

**for** (**int** i = 0; i < 5; i++) {

                     JInternalFrame internal = **new** JInternalFrame("桌面内部窗口" + i, **true**,

**true**, **true**, **true**);

                     internal.setSize(250, 250);

                     internal.setIconifiable(**true**);

                     // 将组件移到新位置。通过此组件父级坐标空间中的 x 和 y 参数来指定新位置的左上角。

                     internal.setLocation(i \* 20, i \* 20);

                     internal.setVisible(**true**);

                     Icon icon = **new** ImageIcon(DemoSuperComponent.**class**

                                   .getResource("/resources/images/plain/Splash.jpg"));

                     internal.getContentPane().add(**new** JLabel(icon));

                     desktop.add(internal);

              }

              jPanel9.add(desktop);

**return** jPanel9;

       }

       @SuppressWarnings("serial")

**private** **static** JPanel getProgressBarPanel() {

              JPanel jPanel8 = **new** JPanel();

**final** JProgressBar jProgressBar = **new** JProgressBar(

                            JProgressBar.*HORIZONTAL*, 0, 100) {

**public** Dimension getPreferredSize() {

**return** **new** Dimension(300, **super**.getPreferredSize().height);

                     }

              };

**final** Timer timer = **new** javax.swing.Timer(100, **new** AbstractAction() {

**public** **void** actionPerformed(ActionEvent e) {

**if** (jProgressBar.getValue() < jProgressBar.getMaximum()) {

                                   jProgressBar.setValue(jProgressBar.getValue() + 1);

                            }

                     }

              });

              Action loadAction = **new** AbstractAction("启动") {

**public** **void** actionPerformed(ActionEvent e) {

                            timer.start();

                     }

              };

              JButton loadButton = **new** JButton(loadAction);

              Action stopAction = **new** AbstractAction("停止") {

**public** **void** actionPerformed(ActionEvent e) {

                            timer.stop();

                     }

              };

              JButton stopButton = **new** JButton(stopAction);

              jPanel8.add(jProgressBar);

              jPanel8.add(loadButton);

              jPanel8.add(stopButton);

**return** jPanel8;

       }

**private** **static** JPanel getTreePanel() {

              JPanel jPanel7 = **new** JPanel();

              DefaultMutableTreeNode root = **new** DefaultMutableTreeNode("根节点");

              DefaultMutableTreeNode parent = **new** DefaultMutableTreeNode("枝节点");

              DefaultMutableTreeNode leaf = **new** DefaultMutableTreeNode("叶子节点");

              parent.add(leaf);

              root.add(parent);

              JTree tree = **new** JTree(root);

              jPanel7.add(tree);

**return** jPanel7;

       }

**private** **static** JPanel getPopupMenuPanel() {

**final** JPanel jPanel6 = **new** JPanel();

**final** JPopupMenu jPopupMenu = **new** JPopupMenu();

              JMenuItem menuItem1 = **new** JMenuItem("保存");

              JMenuItem menuItem2 = **new** JMenuItem("编辑");

              JMenuItem menuItem3 = **new** JMenuItem("删除");

              jPopupMenu.add(menuItem1);

              jPopupMenu.add(menuItem2);

              jPopupMenu.add(menuItem3);

              jPanel6.addMouseListener(**new** MouseAdapter() {

                     @Override

**public** **void** mousePressed(MouseEvent e) {

**int** mask = e.getModifiers();

**if** (mask == 4) {

                                   jPopupMenu.show(jPanel6, e.getX(), e.getY());

                            }

                     }

              });

              jPanel6.add(jPopupMenu);

**return** jPanel6;

       }

**private** **static** JPanel getMenuBarPanel() {

              JPanel jPanel5 = **new** JPanel();

              jPanel5.setLayout(**null**);

              // 菜单栏

              JMenuBar menuBar = **new** JMenuBar();

              menuBar.setBounds(0, 0, 200, 30);

              // 菜单

              JMenu menu = **new** JMenu("菜单一");

              JMenu menu2 = **new** JMenu("菜单二");

              JMenu menu3 = **new** JMenu("菜单三");

              // 菜单项

              JMenuItem jMenuItem = **new** JMenuItem("普通菜单项");

              JMenuItem jMenuItem2 = **new** JRadioButtonMenuItem("单选菜单项");

              JMenuItem jMenuItem3 = **new** JCheckBoxMenuItem("复选菜单项");

              menu.add(jMenuItem);

              menu.addSeparator();

              menu.add(jMenuItem2);

              menu.add(jMenuItem3);

              menuBar.add(menu);

              menuBar.add(menu2);

              menuBar.add(menu3);

              jPanel5.add(menuBar);

**return** jPanel5;

       }

**private** **static** JPanel getToolBarPanel() {

              JPanel jPanel4 = **new** JPanel();

              // 产生一个工具栏对象

              JToolBar jToolBar = **new** JToolBar();

**for** (**int** i = 0; i < 20; i++) {

**final** String text = String.*valueOf*(i + 1);

                     Icon icon = **new** ImageIcon(DemoSuperComponent.**class**

                                   .getResource("/resources/images/buttons/" + i + ".gif"));

                     JButton jButton = **new** JButton(text, icon);

                     jButton.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

                                   JOptionPane.*showMessageDialog*(**null**, text);

                            }

                     });

                     jToolBar.add(jButton);

              }

              jPanel4.add(jToolBar);

**return** jPanel4;

       }

**private** **static** JPanel getSplitPanePanel() {

              JPanel jPanel3 = **new** JPanel();

              jPanel3.setLayout(**new** BorderLayout());

              JLabel lab1 = **new** JLabel("左边", JLabel.*CENTER*);

              JLabel lab2 = **new** JLabel("右边", JLabel.*CENTER*);

              JSplitPane jSplitPane = **new** JSplitPane(JSplitPane.*HORIZONTAL\_SPLIT*,

                            lab1, lab2);

              jSplitPane.setDividerSize(10); // 设置分割条的宽度

              // 设置continuousLayout属性的值，在用户干预要使子组件连续地重新显示和布局子组件

              // jSplitPane.setContinuousLayout(true);

              // 分隔面板的分隔条显示出箭头

              jSplitPane.setOneTouchExpandable(**true**);

              jSplitPane.setDividerLocation(200);

              jPanel3.add(jSplitPane);

**return** jPanel3;

       }

**private** **static** JPanel getTablePanel() {

              JPanel jPanel2 = **new** JPanel();

              // 标题列

              Object[] booksColumn = { "图书名称", "图书作者", "图书价格" };

              // 二维表格数据

              Object[][] dataset = { { "java", "brucl", "85" },

                            { "jsp", "lenotang", "67" }, { "[**JavaScript**](http://lib.csdn.net/base/javascript)", "lunce", "54" } };

              JTable jTable = **new** JTable(dataset, booksColumn);

              jTable.setBackground(Color.*orange*);

              // jTable.setBounds(50, 50, 400, 150);

              // JTableHeader header = jTable.getTableHeader();

              // header.setBackground(Color.blue);

              // header.setBounds(50,30,400,20);

              JScrollPane scrollPane = **new** JScrollPane(jTable);

              jPanel2.add(scrollPane);

**return** jPanel2;

       }

**private** **static** JPanel getSliderPanel() {

              JPanel jPanel1 = **new** JPanel();

              GridLayout g = **new** GridLayout(1, 2);

              g.setHgap(5);

              g.setVgap(5);

              jPanel1.setLayout(g);

              BevelBorder border = **new** BevelBorder(BevelBorder.*LOWERED*);

              // 创建一个水平面板，容纳水平滚动条

              JPanel hp = **new** JPanel();

              hp.setLayout(**new** BoxLayout(hp, BoxLayout.*Y\_AXIS*));

              hp.setBorder(**new** TitledBorder(border, "SliderDemo.horizontal",

                            TitledBorder.*LEFT*, TitledBorder.*ABOVE\_TOP*));

              jPanel1.add(hp);

              // 创建一个垂直面板，容纳垂直滚动条

              JPanel vp = **new** JPanel();

              vp.setLayout(**new** BoxLayout(vp, BoxLayout.*X\_AXIS*));

              vp.setBorder(**new** TitledBorder(border, "SliderDemo.vertical",

                            TitledBorder.*LEFT*, TitledBorder.*ABOVE\_TOP*));

              jPanel1.add(vp);

              JSlider jSlider = **new** JSlider();

              jSlider.setOrientation(JSlider.*HORIZONTAL*);// 设置方向

              jSlider.setMinimum(0);// 设置最小值

              jSlider.setMaximum(100);// 设置最大值

              jSlider.setMajorTickSpacing(20);// 设置主标号间隔

              jSlider.setMinorTickSpacing(5);// 设置辅标号间隔

              jSlider.setPaintLabels(**true**);// Default:false显示标签

              jSlider.setPaintTicks(**true**);// Default:false显示标号

              jSlider.setPaintTrack(**true**);// Determines whether the track is painted

              // on the slider

              jSlider.setValue(0);// 设置初始值

              JPanel p = **new** JPanel();

              p.setLayout(**new** BoxLayout(p, BoxLayout.*Y\_AXIS*));

              p.setBorder(**new** TitledBorder("SliderDemo.plain"));

              p.add(Box.*createRigidArea*(*VGAP5*));

              p.add(jSlider);

              p.add(Box.*createRigidArea*(*VGAP5*));

              hp.add(p);

              hp.add(Box.*createRigidArea*(*VGAP25*));

              p = **new** JPanel();

              p.setLayout(**new** BoxLayout(p, BoxLayout.*X\_AXIS*));

              p.setBorder(**new** TitledBorder("SliderDemo.plain"));

              jSlider = **new** JSlider(JSlider.*VERTICAL*, -10, 100, 20);

              p.add(Box.*createRigidArea*(*HGAP25*));

              p.add(jSlider);

              p.add(Box.*createRigidArea*(*HGAP25*));

              vp.add(p);

              vp.add(Box.*createRigidArea*(*HGAP25*));

**return** jPanel1;

       }

}