工作台视图(透视图)相关节点说明

zjl

2018-10-08T00:00:00.000Z

## 工作台视图(透视图)相关节点说明

工作台是用户使用最频繁的部件，如代码编辑视图、调试视图都在工作台上。通过此部分说明，可实现定制简单工作台视图。主要步骤有：视图扩展和注册、工作台透视图位置布局，透视图功能实现。详见个子结点说明。

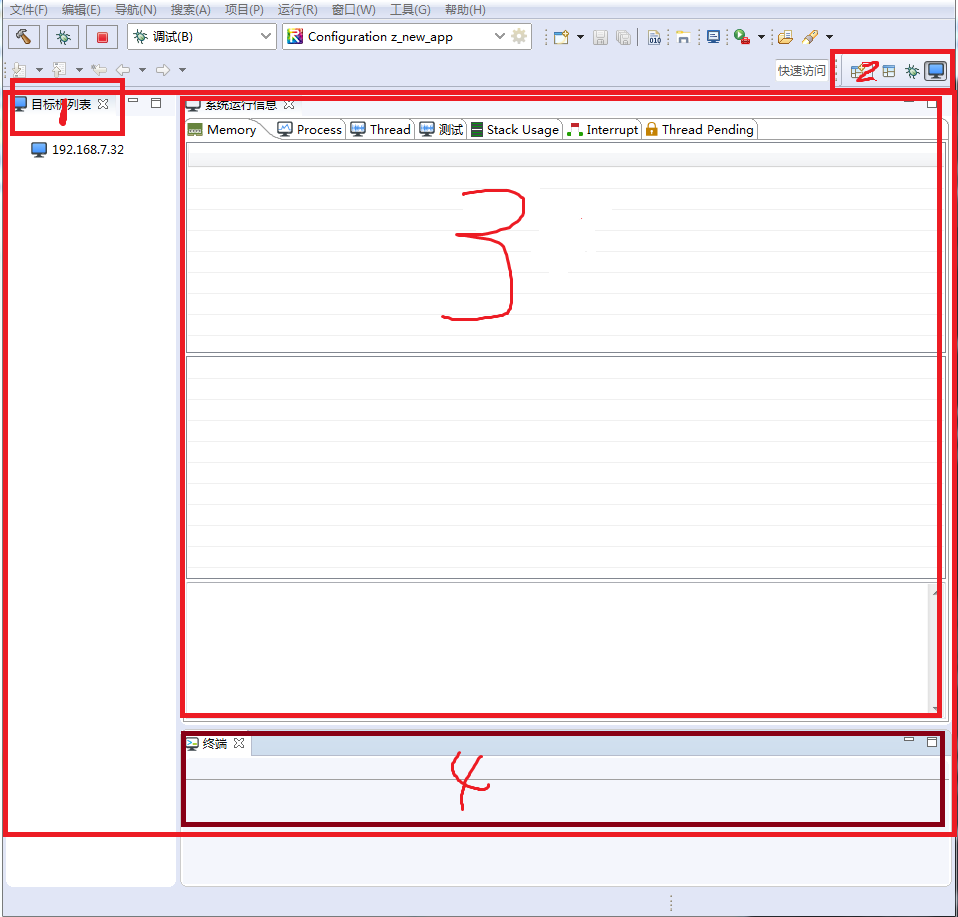
* 相关节点



* 节点概要

|  |  |  |
| --- | --- | --- |
| 节点 | 关键子节点 | 说明 |
| perspectiveExtensions | perspectiveShortcut | 视图扩展和注册 |
| org.eclipse.ui.perspective | perspective | 工作台透视图位置布局 |
| org.eclipse.ui.views | category、view | 透视图具体实现 |

* 视图说明



### (透视图扩展) org.eclipse.ui.perspectiveExtensions

perspectiveExtensions节点可以将透视图注册到 [窗口->透视图->xx] 中，如果不添加，则需要到 [窗口->透视图->其他->xx] 中打开你的透视图。主要节点见下图。



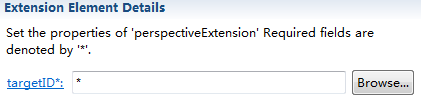
* 节点概要

|  |  |
| --- | --- |
| 关键子节点 | 说明 |
| perspectiveExtension | 透视图注册 |
| perspectiveShortcut | 透视图注册快捷方式 |

#### 透视图注册 perspectiveExtensions

透视图注册实现节点，主要进行透视图id注册，详见 **参数说明**

* 参数截图



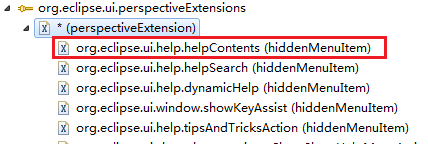
* 参数说明

|  |  |
| --- | --- |
| 参数 | 说明 |
| targetID\* | 目标Id，需要注册的透视图ID，即*org.eclipse.ui.perspectives*的id，注册后会在窗口->透视图->打开透视图 中添加你的视图，否则只会需要在其他中寻找 |

##### 隐藏菜单栏（菜单） hiddenMenuItem

此节点可对IDE的菜单栏进行简单隐藏处理，隐藏不需要的其他菜单，实现视图的简介和功能化区分。详见**参数说明**

* 节点关系图



* 参数说明

|  |  |
| --- | --- |
| 参数 | 说明 |
| id\* | 需要隐藏的菜单id |

##### 隐藏菜工具条 hiddenToolBarItem

此节点可对IDE的工具条进行简单隐藏处理，隐藏不需要的其他工具，实现视图的简介和功能化区分。详见**参数说明**

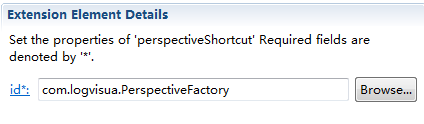
* 参数说明

|  |  |
| --- | --- |
| 参数 | 说明 |
| id\* | 需要隐藏的工具条id |

#### 快捷菜单注册 perspectiveShortcut

透视图快捷菜单注册，添加注册的透视图到菜单项中，详见 **参数说明**

* 参数截图



* 参数说明

|  |  |
| --- | --- |
| 参数 | 说明 |
| id\* | **org.eclipse.ui.perspectives**的子节点perspective的id，通过此id将透视图添加到[*窗口->透视图*]中，不然就只能到下级菜单：其他中寻找你的视图 |

### 透视图(工作台位置布局) org.eclipse.ui.perspectives\*

perspectives节点可是实现透视图子组件的简单预布局等配配置，详见 **参数说明**

* 打开方式:

窗口->透视图->透视图名称（或在其他中寻找创建的透视图名称）只有打开此透视图，透视图按钮才会出现在 图 2 标处，view中定义的视图才会按设计的思路布局，否则通过 菜单栏：窗口>显示视图> 视图名 打开的视图，只会出现在默认的底部区域（在控制台等视图的默认区域）。

* 节点概要

|  |  |
| --- | --- |
| 关键子节点 | 说明 |
| perspective | 负责管理需要实现的具体视图的对应Layout，初始化透视图 |

#### 子节点:perspective

perspective是一个透视图管理类节点，负责管理需要实现的具体视图的对应Layout，初始化透视图。如例图所示。1、3、4位置是独立的子view，并在节点[**org.eclipse.ui.views**]\*中实现。所有子View都通过2实现处按钮实现快速切换功能。

* 关键参数：

|  |  |
| --- | --- |
| 参数 | 说明 |
| id\* | 建议和class名一样 |
| name\* | 透视图名称，即图 标2 处的控件名称 |
| class\* | 实现类 |
| icon | 建议在工程目录下建立一个 [icons] 文件夹，存储所有图标文件。建议像素大小为16x16，如：icons/device.png |

* Demo - class的实现:

package com.xxx;  
   
import org.eclipse.ui.IFolderLayout;  
import org.eclipse.ui.IPageLayout;  
import org.eclipse.ui.IPerspectiveFactory;  
import org.eclipse.ui.wizards.newresource.BasicNewFileResourceWizard;  
import org.eclipse.ui.wizards.newresource.BasicNewFolderResourceWizard;  
   
public class DemoPerspectives implements IPerspectiveFactory {  
   
 public Perspectives() {  
 super();  
 }  
   
 @Override  
 public void createInitialLayout(IPageLayout layout) {  
 defineLayout(layout); // 增加透视图的布局和视图  
 // defineActions(layout); // 增加透视图的动作和快捷  
 }  
   
 // 增加透视图的布局和视图  
 public void defineLayout(IPageLayout paramIPageLayout) {  
 /\* paramIPageLayout 是方位布局，即上中下左右\*/  
   
 // 设置设置编辑器区域可见不可见  
 paramIPageLayout.setEditorAreaVisible(false);  
 // 获取编辑器区域  
 String strEditorArea = paramIPageLayout.getEditorArea();  
   
 /\* 子view 布局 \*/  
 // 左边 区域 布局  
 IFolderLayout leftFolderLayout = paramIPageLayout.createFolder("leftFolder", 1, 0.18F,  
 strEditorArea); // left  
 // 添加子view的节点 id (这里是此插件节点的布局，而不是class的类名)  
 paramIPageLayout.addView("com.acce.hyide.product.gadget.navigator.view", 3, 0.65F,  
 "leftFolder"); // left-bottom  
   
 // 底部 区域布局  
 IFolderLayout bottomIFolderLayout = paramIPageLayout.createFolder("bottomFolder", 4, 0.75F,  
 strEditorArea); // bottom  
 // 添加子view的节点 id  
 bottomIFolderLayout.addView("com.acce.hyide.product.gadget.terminal.view");  
 bottomIFolderLayout.addView("com.acce.hyide.product.gadget.test.view");  
   
 // 中间区域 布局  
 IFolderLayout middleIFolderLayout = paramIPageLayout.createFolder("MiddleFolder", 2, 0.75F,  
 strEditorArea); // Middle  
 // 添加子view的节点 id  
 middleIFolderLayout.addView("com.acce.hyide.product.gadget.information.view");  
 middleIFolderLayout.addView("com.acce.hyide.product.gadget.ftp.view");  
   
 }  
   
 // 增加透视图的动作和快捷  
 public void defineActions(IPageLayout layout) {  
   
 }  
   
}

### 视图实现 org.eclipse.ui.views

views是具体视图实现配置节点，详见 **参数说明**

* 节点关系



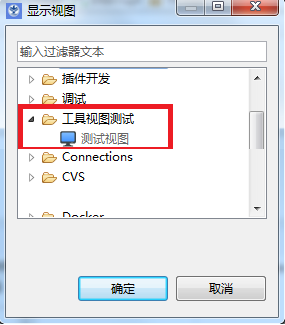
* 参数说明

#### 节点概要

|  |  |
| --- | --- |
| 关键子节点 | 说明 |
| category | 分类文件夹 |
| view | 实现具体视图的节点，被category分类 |

* 视图说明

打开方式： 菜单栏：窗口>显示视图>其他>category名



#### 节点关系

category 是分类文件夹 ，view 是视图实现节点，被category分类



#### 节点解析

##### 子节点 category：

|  |  |
| --- | --- |
| 参数 | 说明 |
| id\* | 分类id |
| name\* | 分类文件夹名称 |

##### 子节点 view：

|  |  |
| --- | --- |
| 参数 | 说明 |
| id\* | 建议为class名 |
| name\* | 子视图名称 |
| class\* | 实现类名，如com.xxxx.DemoRemoteSystemInfoView |
| category | 分类category的id |
| icon: | 图标 |

###### Demo - class实现类

DemoRemoteSystemInfoView.java

package com.xxx;  
   
import java.io.File;  
import java.text.DecimalFormat;  
import java.util.ArrayList;  
import java.util.Collections;  
import java.util.Comparator;  
import java.util.List;  
   
import org.eclipse.jface.dialogs.MessageDialog;  
import org.eclipse.jface.viewers.TableViewer;  
import org.eclipse.swt.SWT;  
import org.eclipse.swt.custom.CTabFolder;  
import org.eclipse.swt.custom.CTabItem;  
import org.eclipse.swt.custom.SashForm;  
import org.eclipse.swt.custom.StyleRange;  
import org.eclipse.swt.custom.StyledText;  
import org.eclipse.swt.events.MenuEvent;  
import org.eclipse.swt.events.MenuListener;  
import org.eclipse.swt.events.SelectionAdapter;  
import org.eclipse.swt.events.SelectionEvent;  
import org.eclipse.swt.events.SelectionListener;  
import org.eclipse.swt.layout.GridData;  
import org.eclipse.swt.layout.GridLayout;  
import org.eclipse.swt.widgets.Button;  
import org.eclipse.swt.widgets.Combo;  
import org.eclipse.swt.widgets.Composite;  
import org.eclipse.swt.widgets.Control;  
import org.eclipse.swt.widgets.DirectoryDialog;  
import org.eclipse.swt.widgets.Display;  
import org.eclipse.swt.widgets.Event;  
import org.eclipse.swt.widgets.Group;  
import org.eclipse.swt.widgets.Listener;  
import org.eclipse.swt.widgets.Menu;  
import org.eclipse.swt.widgets.MenuItem;  
import org.eclipse.swt.widgets.MessageBox;  
import org.eclipse.swt.widgets.Shell;  
import org.eclipse.swt.widgets.Table;  
import org.eclipse.swt.widgets.TableColumn;  
import org.eclipse.swt.widgets.TableItem;  
import org.eclipse.swt.widgets.Text;  
import org.eclipse.swt.widgets.Tree;  
import org.eclipse.swt.widgets.TreeItem;  
import org.eclipse.ui.part.ViewPart;  
   
public class RemoteSystemInfoView2 extends ViewPart {  
 // 分页标签的主页面  
 public CTabFolder tabFolder;  
 // 页面子标签项  
 public CTabItem tbtmNewItem;  
 public CTabItem tbtmNewItem\_1;  
 // 子标签视图  
 public TableViewer tableViewer;  
 public TableViewer tableViewer\_1;  
 public TableViewer tableViewer\_2;  
 // 标签组件  
 // 组件 ： 列表  
 public Table table;  
 public Table table\_1;  
 public Table table\_2;  
 // 组件 ： 文本框  
 public Text text;  
   
 // 布局 ：Group，可作为一个容器或页面，添加其他组件、布局等  
 public Group group;  
 // 组件 ： 下拉框  
 public Combo combo;  
 // 组件 ： 按键  
 public Button button;  
 // 组件,和 Text 差不多，和设置字体和颜色  
 public StyledText styledText;  
   
 // 树状图  
 public Tree tree;  
 private TreeItem treeItem;  
 private TreeItem treeItem2;  
 private MenuItem menuItem1;  
 private MenuItem menuItem2;  
   
 // 简单消息对话框  
 MessageBox messagebox;  
   
   
 @Override  
 public void createPartControl(Composite parent) {  
 // 容器初始化 (所有Tab页面都在此容器下)  
 tabFolder = new CTabFolder(parent, SWT.BORDER);  
 tabFolder.setSimple(false);  
   
 // 添加 UI  
 addTableView\_1();  
 addTableView\_2();  
 addGroupView\_3(parent);  
 addTreeView\_4(parent);  
   
 // 添加右键菜单  
 addMenu(table);  
   
 }  
 // 添加标签页1 方式1  
 // 布局1  
 void addTableView\_1()  
 {  
 // 标签1  
 tbtmNewItem = new CTabItem(tabFolder, SWT.NONE);  
 //tbtmNewItem.setImage(DeviceUtil.getImage("icons/Memory.png")); //$NON-NLS-1$  
 tbtmNewItem.setText("标签1");  
   
 // 标签的容器，在此容器内添加组件  
 SashForm sashForm = new SashForm(tabFolder, SWT.VERTICAL);  
 tbtmNewItem.setControl(sashForm);  
   
 // (列表视图)标签页面的组件  
 tableViewer = new TableViewer(sashForm, SWT.BORDER | SWT.FULL\_SELECTION);  
 table = tableViewer.getTable();  
 table.setHeaderVisible(true);  
 table.setLinesVisible(true);  
 // 添加列表项的监听  
 table.addListener(SWT.MouseDoubleClick, new Listener() {  
 @Override  
 public void handleEvent(org.eclipse.swt.widgets.Event event) {  
 TableItem tableitm;  
 try {  
 tableitm = table.getSelection()[0];// 获取选中的行  
 putMsg("" + tableitm.getText());  
 } catch (Exception e) {  
 return;  
 }  
 File fitem = (File) tableitm.getData();  
 if (fitem.toString().equals("..")) { //$NON-NLS-1$  
 File file = new File(strlocpath);  
 file = file.getParentFile();  
 listLocalFiles(file, table);  
 return;  
 }  
 if (fitem.isDirectory()) {  
 try {  
 listLocalFiles(fitem, table);  
 } catch (Exception e) {  
 // 遇到无权限的文件夹则返回他上级目录  
 MessageDialog.openInformation(Display.getCurrent().getActiveShell(),  
 "title",  
 "无权限的文件夹");  
 File file = fitem.getParentFile();  
 listLocalFiles(file, table);  
 return;  
 }  
 } else {  
 //if (!rimplementation.isListener) {  
 // return;  
 //}  
 //logMessage(Messages.RemoteSystemFtpImplementation\_Startuploading);  
 //rimplementation.Upload(filelist);  
 }  
 }  
 });  
   
 tableViewer\_2 = new TableViewer(sashForm, SWT.BORDER | SWT.FULL\_SELECTION);  
 table\_2 = tableViewer\_2.getTable();  
 table\_2.setToolTipText("table\_2");  
 table\_2.setHeaderVisible(false);  
 table\_2.setLinesVisible(true);  
 // (文本框视图)标签页面的组件  
 text = new Text(sashForm, SWT.BORDER | SWT.WRAP | SWT.V\_SCROLL | SWT.READ\_ONLY);  
 //text.setFont(DeviceUtil.getFont());  
 text.setBackground(Display.getCurrent().getSystemColor(SWT.COLOR\_WHITE));  
 text.setEditable(true);  
   
 // sashForm 的权重  
 sashForm.setWeights(new int[] { 161, 170, 103 });  
 }  
 // 添加标签页2 方式2  
 // 布局2  
 void addTableView\_2()  
 {  
 /\* 方式2 \*/  
 Composite compsoite = new Composite(tabFolder, SWT.NONE);  
 tbtmNewItem\_1 = new CTabItem(tabFolder, SWT.NONE);  
 //tbtmNewItem\_1.setImage(DeviceUtil.getImage("icons/Process.png"));  
 tbtmNewItem\_1.setText("标签2");  
 // 组件1  
 tableViewer\_1 = new TableViewer(compsoite, SWT.BORDER | SWT.FULL\_SELECTION);  
 table\_1 = tableViewer\_1.getTable();  
 // 组件布局方式  
 GridData gd\_tablea = new GridData(SWT.FILL, SWT.FILL, true, true, 1, 1);  
 gd\_tablea.heightHint = 220;  
 // 添加布局到 table  
 table\_1.setLayoutData(gd\_tablea);  
 table\_1.setHeaderVisible(true);  
 table\_1.setLinesVisible(true);  
 // 添加UI内容到 CTabItem  
 tbtmNewItem\_1.setControl(compsoite);  
 GridLayout layout = new GridLayout();  
 layout.marginWidth = 0;  
 layout.marginHeight = 0;  
 // 添加布局到 Composite  
 compsoite.setLayout(layout);  
   
 }  
 // 布局3 另起一组  
 void addGroupView\_3(Composite parent)  
 {  
 group = new Group(parent, SWT.NULL);  
 group.setText("Group 布局");  
 GridLayout gridLayout = new GridLayout(2, true);  
 gridLayout.marginHeight = 0;  
 group.setLayout(gridLayout);  
 group.setLayoutData(new GridData(GridData.FILL\_HORIZONTAL));  
   
 /\* 下拉框 \*/  
 combo = new Combo(group, SWT.FLAT);  
 combo.setLayoutData(new GridData(SWT.FILL, SWT.CENTER, true, false, 1, 1));  
 combo.setText("默认数据");  
 combo.add("下拉数据1", 0);//添加到最前  
 combo.add("下拉数据2", 0);//不会覆盖  
 // String[] data = combo.getItems()获得所有项  
 // 为该下拉框选中事件  
 combo.addSelectionListener(new SelectionAdapter() {  
 public void widgetSelected(SelectionEvent e) {  
 String strcomnocontent = combo.getText().trim();// 选择的数据  
 MessageDialog  
 .openInformation(  
 Display.getCurrent().getActiveShell(),  
 "标题",  
 "内容:" + strcomnocontent);  
 }  
 });  
   
 /\* 按键 \*/  
 button = new Button(group, SWT.PUSH);  
 //button.setImage(DeviceUtil.getImage("icons/up.png"));  
 button.addSelectionListener(new SelectionAdapter() {  
 @Override  
 public void widgetSelected(SelectionEvent e) {  
 // 打开文件资源管理器  
 DirectoryDialog dialog = new DirectoryDialog(parent.getShell());  
 // 返回选中的文件路径  
 String strpath = dialog.open();  
 if (strpath == null) {  
 return;  
 }  
 File file = new File(strpath);  
 if (file.getParentFile() == null) {  
 //buttonUpLocalDir.setEnabled(false);  
 } else {  
 listLocalFiles(file, table);  
 putStyledText(file.toPath().toString());  
 }  
 }  
 });  
 //  
 styledText = new StyledText(group, SWT.BORDER | SWT.V\_SCROLL);  
 styledText.setSize(200, 400);  
 styledText.setEditable(false);  
 }  
   
 // 给tab页面添加右键菜单  
 // 添加 Tree 视图  
 void addTreeView\_4(Composite parent)  
 {  
 this.tree = new Tree(parent, SWT.NONE);  
 // 添加双点击事件  
 this.tree.addSelectionListener(new SelectionListener() {  
 // 双击时  
 @Override  
 public void widgetDefaultSelected(SelectionEvent e) {  
 putMsg("TreeView点击响应" + tree.getSelection()[0].getData().toString() );// 获取选中信息  
 }  
   
 // 获得(单击)焦点时  
 @Override  
 public void widgetSelected(SelectionEvent e) {  
 //putMsg("TreeView点击响应widgetSelected");  
 }  
 });  
   
 // 添加右键菜单  
 addMenu(tree);  
   
 treeItem = new TreeItem(tree, SWT.None);  
   
 treeItem.setText("subItem");  
 treeItem.setData("subItemData");  
 //treeItem.setImage(DeviceUtil.getImage("icons/connect.png")); //$NON-NLS-1$  
   
 treeItem2 = new TreeItem(treeItem, SWT.None);  
 treeItem2.setText("subItem2");  
 treeItem2.setData("subItemData2");  
 //treeItem2.setImage(DeviceUtil.getImage("icons/connect.png")); //$NON-NLS-1$  
   
 }  
   
 // 添加右键菜单 给Table、Tree等继承Control的类都行  
 private void addMenu(Control table) {  
 // 添加右键菜单  
 Menu memorymenu = new Menu(table);  
   
 // 添加操作项  
 MenuItem memoryItem = new MenuItem(memorymenu, SWT.PUSH);  
 // 操作项名称  
 memoryItem.setText("操作项1");  
 // 添加操作响应监听  
 memoryItem.addListener(SWT.Selection, new Listener() {  
 @Override  
 public void handleEvent(Event event) {  
 Thread memorymenu = new Thread() {  
 public void run() {  
 //操作响应  
 // 列表头  
 String[] strtitle = {"strtitle1","strtitle2"};  
 // 列表数据行  
 String[][] strcontent = {{"strcontent1", "strcontent2"},{"strcontent3", "strcontent4"}};  
 setTableData\_table(strtitle, strcontent);  
 }  
 };  
 memorymenu.setName("memorymenu");  
 memorymenu.start();  
 }  
 });  
   
 // 创建一个分割线  
 MenuItem separator1 = new MenuItem(memorymenu, SWT.SEPARATOR);  
   
 // 添加其他操作项  
 MenuItem connectItem = new MenuItem(memorymenu, SWT.PUSH);  
 connectItem.setText("操作项2");  
 connectItem.addListener(SWT.Selection, new Listener() {  
 @Override  
 public void handleEvent(Event event) {  
   
 }  
 });  
   
 // 菜单监听  
 memorymenu.addMenuListener(new MenuListener() {  
 // 菜单显示前  
 @Override  
 public void menuShown(MenuEvent e) {  
 // 设置操作项不可显示  
 memoryItem.setEnabled(true);  
 connectItem.setEnabled(false);  
 // 这里可以设置规则，控制操作项的显示逻辑  
 // \*\*\*  
 }  
   
 // 菜单隐藏后  
 @Override  
 public void menuHidden(MenuEvent e) {  
 }  
 });  
 table.setMenu(memorymenu);  
   
 /\* MenuItem 可重复添加 \*/  
 }  
   
 /\*\*  
 \* StyledText 显示数据  
 \* @param message  
 \*/  
 public void putStyledText(String message) {  
 styledText.getDisplay().asyncExec(new Runnable() {  
 @Override  
 public void run() {  
 StyleRange styleRange1 = new StyleRange();  
 styleRange1.start = styledText.getCharCount();  
 styleRange1.length = message.length();  
 styleRange1.fontStyle = SWT.BOLD;  
 Shell shell = new Shell();  
 styleRange1.foreground = shell.getDisplay().getSystemColor(SWT.COLOR\_RED);// Display.getCurrent().getActiveShell()  
 styledText.append(message + "\r\n"); //$NON-NLS-1$  
 styledText.setStyleRange(styleRange1);  
 styledText.setSelection(styledText.getCharCount());  
 }  
 });  
 }  
   
 public static long kb = 1024;  
 public static long mb = kb \* 1024;  
 public static long gb = mb \* 1024;  
 public static DecimalFormat df = new DecimalFormat("#,###"); //$NON-NLS-1$  
 String strlocpath = "";// 当前地址  
 /\*  
 \* 在table显示 本地路径 数据  
 \*/  
 private void listLocalFiles(File selDisk, Table table) {  
 if (selDisk == null || selDisk.isFile()) {  
 return;  
 }  
 strlocpath = selDisk.toString();  
 combo.setText(selDisk.toString());// 当前地址  
 table.removeAll();// 移除旧数据  
 File[] listFiles = selDisk.listFiles(); // 获取磁盘文件列表  
 List<File> fileList = new ArrayList<File>();  
 for (File f : listFiles) {  
 fileList.add(f);  
 }  
 Collections.sort(fileList, new Comparator<File>() {  
 @Override  
 public int compare(File o1, File o2) {  
 if (o1.isDirectory() && o2.isFile()) {  
 return -1;  
 }  
 if (o1.isFile() && o2.isDirectory()) {  
 return 1;  
 }  
 return o1.getName().compareTo(o2.getName());  
 }  
 });  
 // 上层目录  
 if (selDisk.getParentFile() != null) {  
   
 File file = new File(".."); //$NON-NLS-1$  
 TableItem itema = new TableItem(table, SWT.NONE);  
 //itema.setImage(DeviceUtil.getImage("icons/newfolder.png")); //$NON-NLS-1$  
 itema.setText(new String[] { "..", "", "" }); //$NON-NLS-1$ //$NON-NLS-2$ //$NON-NLS-3$  
 itema.setData(file);  
 } else {  
   
 }  
   
 // 遍历磁盘根文件夹的内容，添加到表格中  
 for (File file : fileList) {  
 TableItem itemsa = new TableItem(table, SWT.NONE);  
 String length = ""; // 获取文件大小 //$NON-NLS-1$  
 long lsize = file.length();  
 if (lsize >= gb) {  
 length = String.format("%.1fGB", (float) lsize / gb) + " (" + df.format(lsize) + ")"; //$NON-NLS-1$ //$NON-NLS-2$ //$NON-NLS-3$  
 } else if (lsize >= mb) {  
 float f = (float) lsize / mb;  
 length = String.format(f > 100 ? "%.0fMB" : "%.1fMB", f) + " (" + df.format(lsize) + ")"; //$NON-NLS-1$ //$NON-NLS-2$ //$NON-NLS-3$ //$NON-NLS-4$  
 } else if (lsize >= kb) {  
 float f = (float) lsize / kb;  
 length = String.format(f > 100 ? "%.0fKB" : "%.1fKB", f) + " (" + df.format(lsize) + ")"; //$NON-NLS-1$ //$NON-NLS-2$ //$NON-NLS-3$ //$NON-NLS-4$  
 } else {  
 length = String.format("%dB", lsize); //$NON-NLS-1$  
 }  
 if (file.isDirectory()) { // 显示文件夹标志  
 length = ""; //$NON-NLS-1$  
 //itemsa.setImage(DeviceUtil.getIcon(file));  
 } else {  
 //itemsa.setImage(DeviceUtil.getIcon(file));  
 }  
 // 获取文件的最后修改日期  
 //SimpleDateFormat myFmt = new SimpleDateFormat(  
 //ConstantClass.RemoteSystemFtpView\_DateFormat);  
 //String modifDate = myFmt.format(new Date(file.lastModified()));  
 itemsa.setText(new String[] { file.getName(), length, "日期" });  
 itemsa.setData(file);  
 }  
 }  
   
 // 添加 数据奥 table 列表  
 /\*添加数据到 tab 列表中\*/  
 // 给table 添加数据  
 public void setTableData\_table(String[] strtitle, String[][] strcontent) {  
 setTableData(table, tableViewer, strcontent, strtitle);  
 }  
   
 /\*\*  
 \* 给table添加信息公共方法  
 \*  
 \* @param table  
 \* @param tableViewer  
 \*/  
 void setTableData(Table table, TableViewer tableViewer, String[][] strcontent,  
 String[] strtitle) {  
 Display.getDefault().syncExec(new Runnable() {  
 @Override  
 public void run() {  
 // 记录table当前位置  
 int iscrollbarlocation = table.getTopIndex();  
 int iSelection = table.getSelectionIndex();  
 // 清空table内容  
 table.removeAll();  
   
 // 重新加载table内容  
 // if(tableViewer.getTable().getColumnCount()==0){  
 TableColumn[] columns = table.getColumns(); // 清除表格原有表头  
 for (TableColumn tc : columns) {  
 tc.dispose();  
 }  
 // 添加表头  
 for (int i = 0; i < strtitle.length; i++) {  
 TableColumn TableContent = new TableColumn(table, SWT.NONE, i);  
 TableContent.setWidth(140);  
 TableContent.setText(strtitle[i]);  
 }  
 // }  
   
 for (int j = 0; j < strcontent.length; j++) {  
 // 添加数据  
 TableItem TableTitle = new TableItem(table, SWT.NONE);  
 String strcont[] = new String[strcontent[j].length];  
 for (int k = 0; k < strcontent[j].length; k++) {  
 strcont[k] = strcontent[j][k];  
 }  
 TableTitle.setText(strcont);  
 }  
 // 设置当前选中信息位置  
 table.setTopIndex(iscrollbarlocation);  
 }  
 });  
 }  
   
 @Override  
 public void setFocus() {  
 // TODO Auto-generated method stub  
 }  
 void putMsg(String msg)  
 {  
 MessageDialog  
 .openInformation(  
 Display.getCurrent().getActiveShell(),  
 "标题",  
 msg);  
 }  
   
}

* 说明 还有部分节点未涉及，暂不说明