JTree JSON

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package yan.t3;

import java.awt.*;
import java.awt.event.*;

import javax.swing.*;
import javax.swing.event.*;
import javax.swing.tree.*;

public class JTreeTest implements ActionListener, TreeModelListener {

    JLabel label = null;
    JTree tree = null;
    DefaultTreeModel treeModel = null;
    String nodeName = null;// 原有节点名称

    public JTreeTest() {
        JFrame f = new JFrame("");
        Container contentPane = f.getContentPane();

        DefaultMutableTreeNode root = new DefaultMutableTreeNode("JSON 结构");

        tree = new JTree(root);
        tree.setEditable(true);
        tree.addMouseListener(new MouseHandle());
        treeModel = (DefaultTreeModel) tree.getModel();
        treeModel.addTreeModelListener(this);
        // add this listener for edit the tree node
        tree.getCellEditor().addCellEditorListener(new Tree_CellEditorAction());
        JScrollPane scrollPane = new JScrollPane();
        scrollPane.setViewportView(tree);

        JPanel panel = new JPanel();
        JButton b = new JButton("新增节点");
        b.addActionListener(this);
        panel.add(b);
```

```
        b = new JButton("删除节点");
        b.addActionListener(this);
        panel.add(b);
        b = new JButton("清除所有节点");
        b.addActionListener(this);
        panel.add(b);

        label = new JLabel("Action");
        contentPane.add(panel, BorderLayout.NORTH);
        contentPane.add(scrollPane, BorderLayout.CENTER);
        contentPane.add(label, BorderLayout.SOUTH);
        f.pack();
        f.setVisible(true);
        f.requestFocus();
        f.setSize(400, 300);
        f.setLocationRelativeTo(null);
        f.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }

            public void windowLostFocus(WindowEvent e) {
                System.out.println("ggg");
            }
        });
    }

    // 本方法运行新增、删除、清除所有节点的程序代码.
    public void actionPerformed(ActionEvent ae) {
        if (ae.getActionCommand().equals("新增节点")) {
            DefaultMutableTreeNode parentNode = null;
            DefaultMutableTreeNode newNode = new DefaultMutableTreeNode("新节点");
            newNode.setAllowsChildren(true);
            TreePath parentPath = tree.getSelectionPath();
            if (parentPath == null) {
                return;
            }
            // 取得新节点的父节点
            parentNode = (DefaultMutableTreeNode) (parentPath.getLastPathComponent());

            // 由 DefaultTreeModel 的 insertNodeInto（）方法增加新节点
            treeModel.insertNodeInto(newNode, parentNode, parentNode.getChildCount());

            // tree 的 scrollPathToVisible()方法在使 Tree 会自动展开文件夹以便显示所加入
```

```
的新节点。若没加这行则加入的新节点
                // 会被 包在文件夹中，你必须自行展开文件夹才看得到。
                tree.scrollPathToVisible(new TreePath(newNode.getPath()));
                label.setText("新增节点成功");
            }
            if (ae.getActionCommand().equals("删除节点")) {
                TreePath treepath = tree.getSelectionPath();
                if (treepath != null) {
                    // 下面两行取得选取节点的父节点.
                    DefaultMutableTreeNode selectionNode = (DefaultMutableTreeNode)
treepath.getLastPathComponent();
                    TreeNode parent = (TreeNode) selectionNode.getParent();
                    if (parent != null) {
                        // 由 DefaultTreeModel 的 removeNodeFromParent()方法删除节点，包
含它的子节点。
                        treeModel.removeNodeFromParent(selectionNode);
                        label.setText("删除节点成功");
                    }
                }
            }
            if (ae.getActionCommand().equals("清除所有节点")) {

                // 下面一行，由 DefaultTreeModel 的 getRoot()方法取得根节点.
                DefaultMutableTreeNode rootNode = (DefaultMutableTreeNode)
treeModel.getRoot();

                // 下面一行删除所有子节点.
                rootNode.removeAllChildren();

                // 删除完后务必运行 DefaultTreeModel 的 reload()操作，整个 Tree 的节点才会
真正被删除.
                treeModel.reload();
                label.setText("清除所有节点成功");
            }
        }

        public void treeNodesChanged(TreeModelEvent e) {
            TreePath treePath = e.getTreePath();
            DefaultMutableTreeNode node = (DefaultMutableTreeNode)
treePath.getLastPathComponent();
            try {
                int[] index = e.getChildIndices();
                node = (DefaultMutableTreeNode) node.getChildAt(index[0]);
            } catch (NullPointerException exc) {
```

```
        }
        label.setText(nodeName + "更改数据为:" + (String) node.getUserObject());
    }

    public void treeNodesInserted(TreeModelEvent e) {
        System.out.println("new node insered");
    }

    public void treeNodesRemoved(TreeModelEvent e) {
        System.out.println("node deleted");
    }

    public void treeStructureChanged(TreeModelEvent e) {
        System.out.println("Structrue changed");
    }

    public static void main(String[] args) {
        new JTreeTest();
    }

    class MouseHandle extends MouseAdapter {

        public void mousePressed(MouseEvent e) {
            try {
                JTree tree = (JTree) e.getSource();
                int rowLocation = tree.getRowForLocation(e.getX(), e.getY());
                TreePath treepath = tree.getPathForRow(rowLocation);
                TreeNode treenode = (TreeNode) treepath.getLastPathComponent();
                nodeName = treenode.toString();
            } catch (NullPointerException ne) {
            }
        }
    }

    class Tree_CellEditorAction implements CellEditorListener {

        public void editingStopped(ChangeEvent e) {
            Object selectnode = tree.getLastSelectedPathComponent();
            DefaultMutableTreeNode node = (DefaultMutableTreeNode) selectnode;
            CellEditor cellEditor = (CellEditor) e.getSource();
            String newName = (String) cellEditor.getCellEditorValue();
            node.setUserObject(newName);

            DefaultTreeModel model = (DefaultTreeModel) tree.getModel();
```

```
                model.nodeStructureChanged(node);
            }

            public void editingCanceled(ChangeEvent e) {
                editingStopped(e);
            }
        }
    }
```