

# Java Swing – Drag-n-Drop – One Trace Use Full Trace (No Slicing)

Supporting Applications : [Java2s DnD]

```
import java.awt.dnd.DragGestureListener;
import java.awt.Insets;
import java.awt.dnd.DropTargetDropEvent;
import java.awt.dnd.DropTargetDragEvent;
import java.awt.Color;
import java.awt.Insets;
import java.awt.Point;
import java.awt.dnd.DropTargetListener;
import java.awt.dnd.DropTarget;
import javax.swing.JFrame;
import java.awt.dnd.DragGestureEvent;
import javax.swing.tree.TreePath;
import javax.swing.JMenuBar;
import java.awt.Component;
import java.awt.dnd.DragSourceDropEvent;
import java.awt.dnd.DragSourceDragEvent;
import java.awt.dnd.DropTargetDragEvent;
import java.awt.datatransfer.Transferable;
import java.awt.dnd.DragGestureRecognizer;
import java.awt.Rectangle;
import java.awt.dnd.Autoscroll;
import javax.swing.JTree;
import java.awt.Dimension;
import javax.swing.JMenuItem;
import java.awt.GraphicsConfiguration;
import javax.swing.tree.TreeModel;
import java.awt.datatransfer.DataFlavor;
import javax.swing.JScrollPane;
import java.awt.dnd.DragSourceListener;
import java.awt.dnd.DropTargetEvent;
import java.awt.dnd.DropTargetContext;
import java.awt.Graphics;
import java.awt.Container;
import javax.swing.JMenuBar;

public class AppropTargetListener
implements DropTargetListener {

    public AppropTargetListener() {
        DropTarget dropTarget = new DropTarget(appJTree && appropTargetListener);
    }

    public void drop(DropTargetDropEvent) {
        Point point = DropTargetDropEvent.getLocation();
        DropTargetContext dropTargetContext = DropTargetDropEvent.getDropTargetContext();
        Transferable transferable = DropTargetDropEvent.getTransferable();
        int app_int2 = DropTargetDropEvent.getDropAction();
        DropTargetDropEvent.acceptDrop(app_int2);
        DropTargetDropEvent.dropComplete(boolean);
        DataFlavor[] dataFlavorArray = transferable.getTransferDataFlavors();
        boolean app_boolean1 = transferable.isDataFlavorSupported(dataFlavorArray);
        Object object1 = transferable.getTransferData(dataFlavorArray);
        Component component = dropTargetContext.getComponent();
        TreePath treePath = component.getClosestPathForLocation(point);
        TreeModel treeModel = component.getModel();
        Object object = treePath.getLastPathComponent(); // REPEATED!
        treeModel.insertNodeInto(object);
        boolean app_boolean = object.isLeaf();
    }

    public void dragEnter(DropTargetDragEvent) {
        Point point1 = DropTargetDragEvent.getLocation();
        DropTargetContext dropTargetContext = DropTargetEvent.getDropTargetContext();
        DropTargetDragEvent.rejectDrag();
        Component component = dropTargetContext.getComponent();
        TreePath treePath = component.getClosestPathForLocation(point1);
        Object object = treePath.getLastPathComponent();
        boolean app_boolean = object.isLeaf();
    }

    public void dragOver(DropTargetDragEvent) {
        Point point1 = DropTargetDragEvent.getLocation(); // REPEATED!
        DropTargetContext dropTargetContext = DropTargetEvent.getDropTargetContext(); // REPEATED!
        int app_int = DropTargetDragEvent.getDropAction(); // REPEATED!
        DropTargetDragEvent.acceptDrag(app_int); // REPEATED!
        DropTargetDragEvent.rejectDrag(); // REPEATED!
        Component component = dropTargetContext.getComponent(); // REPEATED!
        TreePath treePath = component.getClosestPathForLocation(point1); // REPEATED!
        Object object = treePath.getLastPathComponent(); // REPEATED!
        boolean app_boolean = object.isLeaf(); // REPEATED!
    }

}

public class AppJTree
implements Autoscroll
extends JTree {

    public Insets getAutoscrollInsets() {
        Container container = appJTree.getParent(); // REPEATED!
        container.getBounds(Rectangle):(Rectangle)|():(Rectangle); // REPEATED!
        Insets insets = new Insets(int, int, int, int); // REPEATED!
    }

    public void paintComponent(Graphics) {
        Graphics.setColor(Color); // REPEATED!
        Graphics.drawRect(int,int,int,int); // REPEATED!
        Container container = appJTree.getParent(); // REPEATED!
        container.getBounds(Rectangle):(Rectangle)|():(Rectangle); // REPEATED!
    }

}

public class Appransferable
implements Transferable {

    public Appransferable() {
        DataFlavor dataFlavor = new DataFlavor(String,String,ClassLoader)| |(String,String,MimeTypeParameterList,Class,String)| |()| |(String)| |(Class,String)| |(String,String);
    }

}

public class AppragSourceListener
implements DragSourceListener, DragGestureListener {

    public AppragSourceListener() {
        DragSource dragSource = new DragSource();
        DragGestureRecognizer dragGestureRecognizer = dragSource.createDefaultDragGestureRecognizer(appJTree &&
        appragSourceListener);
    }

}
```

```

public void dragOver(DragSourceDragEvent) {
}

public void dragEnter(DragSourceDragEvent) {
}

public void dragGestureRecognized(DragGestureEvent) {
    TreePath treePath1 = appJTree.getSelectionPath();
    TransferableTreeNode transferableTreeNode = new TransferableTreeNode(treePath1);
    appDragSource.startDrag(appragSourceListener && transferableTreeNode);
    int app_int3 = treePath1.getPathCount();
    Object object = treePath1.getLastPathComponent();
}

public void dragDropEnd(DragSourceDropEvent) {
    TreeModel treeModel = appJTree.getModel();
    treeModel.removeNodeFromParent(object);
    /*          Cyclic Statements          */
    int app_int1 = DragSourceDropEvent.getDropAction(); // REPEATED!
    boolean app_boolean2 = DragSourceDropEvent.getDropSuccess();
}
}

public class SomeClass {
    public void someMethod() {
        JFrame jFrame = new JFrame()|(GraphicsConfiguration)|(String)|(String,GraphicsConfiguration);
        AppJTree appJTree = new AppJTree();
        AppropTargetListener apppropTargetListener = new ApppropTargetListener(appJTree);
        AppragSourceListener appragSourceListener = new AppragSourceListener(appJTree);
        JFrame.setIconImage(Image);
        JMenuBar jMenuBar = new JMenuBar();
        JFrame.setJMenuBar(jMenuBar);
        JScrollPane jScrollPane = new JScrollPane(appJTree);
        JFrame.setSize(Dimension:()|(int,int):());
        JFrame.setDefaultCloseOperation(int);
        JMenu jMenu = new JMenu()|(String)|(Action)|(String,boolean);
        JMenuBar.add(jMenu);
        Container container1 = jFrame.getContentPane();
        JFrame.setVisible(boolean);
        container1.add(jScrollPane);
        Class class = DataFlavor.getRepresentationClass();
        /*          Cyclic Statements          */
        JMenuItem jMenuItem = new JMenuItem(imageIcon); // REPEATED!
        JMenu.add(jMenuItem); // REPEATED!
        DragSourceContext dragSourceContext = DragSourceEvent.getDragSourceContext();
        dragSourceContext.setCursor(Cursor);
        Appransferable appransferable = new Appransferable();
        DragGestureEvent.startDrag(appransferable && appragSourceListener);
        container1.getLocation():(Point)|(Point):(Point);
        int app_int1 = container1.getHeight();
        int app_int = container1.getWidth();
    }
}

```