| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JTree.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/swing/JToolTip.AccessibleJToolTip.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JTree.AccessibleJTree.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JTree.html)    [**NO FRAMES**](http://docs.google.com/JTree.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#4d34og8) | [CONSTR](#26in1rg) | [METHOD](#lnxbz9) | DETAIL: [FIELD](#z337ya) | [CONSTR](#46r0co2) | [METHOD](#3ygebqi) |

## **javax.swing**

Class JTree

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.awt.Component](http://docs.google.com/java/awt/Component.html)  
 [java.awt.Container](http://docs.google.com/java/awt/Container.html)  
 [javax.swing.JComponent](http://docs.google.com/javax/swing/JComponent.html)  
 **javax.swing.JTree**

**All Implemented Interfaces:** [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html), [MenuContainer](http://docs.google.com/java/awt/MenuContainer.html), [Serializable](http://docs.google.com/java/io/Serializable.html), [Accessible](http://docs.google.com/javax/accessibility/Accessible.html), [Scrollable](http://docs.google.com/javax/swing/Scrollable.html)

public class **JTree**extends [JComponent](http://docs.google.com/javax/swing/JComponent.html)implements [Scrollable](http://docs.google.com/javax/swing/Scrollable.html), [Accessible](http://docs.google.com/javax/accessibility/Accessible.html)

A control that displays a set of hierarchical data as an outline. You can find task-oriented documentation and examples of using trees in [How to Use Trees](http://java.sun.com/docs/books/tutorial/uiswing/components/tree.html), a section in *The Java Tutorial.*

A specific node in a tree can be identified either by a TreePath (an object that encapsulates a node and all of its ancestors), or by its display row, where each row in the display area displays one node. An *expanded* node is a non-leaf node (as identified by TreeModel.isLeaf(node) returning false) that will displays its children when all its ancestors are *expanded*. A *collapsed* node is one which hides them. A *hidden* node is one which is under a collapsed ancestor. All of a *viewable* nodes parents are expanded, but may or may not be displayed. A *displayed* node is both viewable and in the display area, where it can be seen.

The following JTree methods use "visible" to mean "displayed":

* isRootVisible()
* setRootVisible()
* scrollPathToVisible()
* scrollRowToVisible()
* getVisibleRowCount()
* setVisibleRowCount()

The next group of JTree methods use "visible" to mean "viewable" (under an expanded parent):

* isVisible()
* makeVisible()

If you are interested in knowing when the selection changes implement the TreeSelectionListener interface and add the instance using the method addTreeSelectionListener. valueChanged will be invoked when the selection changes, that is if the user clicks twice on the same node valueChanged will only be invoked once.

If you are interested in detecting either double-click events or when a user clicks on a node, regardless of whether or not it was selected, we recommend you do the following:

final JTree tree = ...;  
  
 MouseListener ml = new MouseAdapter() {  
 public void **mousePressed**(MouseEvent e) {  
 int selRow = tree.getRowForLocation(e.getX(), e.getY());  
 TreePath selPath = tree.getPathForLocation(e.getX(), e.getY());  
 if(selRow != -1) {  
 if(e.getClickCount() == 1) {  
 mySingleClick(selRow, selPath);  
 }  
 else if(e.getClickCount() == 2) {  
 myDoubleClick(selRow, selPath);  
 }  
 }  
 }  
 };  
 tree.addMouseListener(ml);

NOTE: This example obtains both the path and row, but you only need to get the one you're interested in.

To use JTree to display compound nodes (for example, nodes containing both a graphic icon and text), subclass [TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) and use [setCellRenderer(javax.swing.tree.TreeCellRenderer)](http://docs.google.com/javax/swing/JTree.html#setCellRenderer(javax.swing.tree.TreeCellRenderer)) to tell the tree to use it. To edit such nodes, subclass [TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) and use [setCellEditor(javax.swing.tree.TreeCellEditor)](http://docs.google.com/javax/swing/JTree.html#setCellEditor(javax.swing.tree.TreeCellEditor)).

Like all JComponent classes, you can use [InputMap](http://docs.google.com/javax/swing/InputMap.html) and [ActionMap](http://docs.google.com/javax/swing/ActionMap.html) to associate an [Action](http://docs.google.com/javax/swing/Action.html) object with a [KeyStroke](http://docs.google.com/javax/swing/KeyStroke.html) and execute the action under specified conditions.

**Warning:** Swing is not thread safe. For more information see [Swing's Threading Policy](http://docs.google.com/package-summary.html#threading).

**Warning:** Serialized objects of this class will not be compatible with future Swing releases. The current serialization support is appropriate for short term storage or RMI between applications running the same version of Swing. As of 1.4, support for long term storage of all JavaBeansTM has been added to the java.beans package. Please see [XMLEncoder](http://docs.google.com/java/beans/XMLEncoder.html).

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**JTree.AccessibleJTree**](http://docs.google.com/javax/swing/JTree.AccessibleJTree.html)            This class implements accessibility support for the JTree class. |
| static class | [**JTree.DropLocation**](http://docs.google.com/javax/swing/JTree.DropLocation.html)            A subclass of TransferHandler.DropLocation representing a drop location for a JTree. |
| static class | [**JTree.DynamicUtilTreeNode**](http://docs.google.com/javax/swing/JTree.DynamicUtilTreeNode.html)            DynamicUtilTreeNode can wrap vectors/hashtables/arrays/strings and create the appropriate children tree nodes as necessary. |
| protected static class | [**JTree.EmptySelectionModel**](http://docs.google.com/javax/swing/JTree.EmptySelectionModel.html)            EmptySelectionModel is a TreeSelectionModel that does not allow anything to be selected. |
| protected  class | [**JTree.TreeModelHandler**](http://docs.google.com/javax/swing/JTree.TreeModelHandler.html)            Listens to the model and updates the expandedState accordingly when nodes are removed, or changed. |
| protected  class | [**JTree.TreeSelectionRedirector**](http://docs.google.com/javax/swing/JTree.TreeSelectionRedirector.html)            Handles creating a new TreeSelectionEvent with the JTree as the source and passing it off to all the listeners. |

| **Nested classes/interfaces inherited from class javax.swing.**[**JComponent**](http://docs.google.com/javax/swing/JComponent.html) |
| --- |
| [JComponent.AccessibleJComponent](http://docs.google.com/javax/swing/JComponent.AccessibleJComponent.html) |

| **Nested classes/interfaces inherited from class java.awt.**[**Container**](http://docs.google.com/java/awt/Container.html) |
| --- |
| [Container.AccessibleAWTContainer](http://docs.google.com/java/awt/Container.AccessibleAWTContainer.html) |

| **Nested classes/interfaces inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [Component.AccessibleAWTComponent](http://docs.google.com/java/awt/Component.AccessibleAWTComponent.html), [Component.BaselineResizeBehavior](http://docs.google.com/java/awt/Component.BaselineResizeBehavior.html), [Component.BltBufferStrategy](http://docs.google.com/java/awt/Component.BltBufferStrategy.html), [Component.FlipBufferStrategy](http://docs.google.com/java/awt/Component.FlipBufferStrategy.html) |

| **Field Summary** | |
| --- | --- |
| static [String](http://docs.google.com/java/lang/String.html) | [**ANCHOR\_SELECTION\_PATH\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#ANCHOR_SELECTION_PATH_PROPERTY)            Bound property name for anchor selection path. |
| static [String](http://docs.google.com/java/lang/String.html) | [**CELL\_EDITOR\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#CELL_EDITOR_PROPERTY)            Bound property name for cellEditor. |
| static [String](http://docs.google.com/java/lang/String.html) | [**CELL\_RENDERER\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#CELL_RENDERER_PROPERTY)            Bound property name for cellRenderer. |
| protected  [TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) | [**cellEditor**](http://docs.google.com/javax/swing/JTree.html#cellEditor)            Editor for the entries. |
| protected  [TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) | [**cellRenderer**](http://docs.google.com/javax/swing/JTree.html#cellRenderer)            The cell used to draw nodes. |
| protected  boolean | [**editable**](http://docs.google.com/javax/swing/JTree.html#editable)            Is the tree editable? Default is false. |
| static [String](http://docs.google.com/java/lang/String.html) | [**EDITABLE\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#EDITABLE_PROPERTY)            Bound property name for editable. |
| static [String](http://docs.google.com/java/lang/String.html) | [**EXPANDS\_SELECTED\_PATHS\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#EXPANDS_SELECTED_PATHS_PROPERTY)            Bound property name for expands selected paths property |
| static [String](http://docs.google.com/java/lang/String.html) | [**INVOKES\_STOP\_CELL\_EDITING\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#INVOKES_STOP_CELL_EDITING_PROPERTY)            Bound property name for messagesStopCellEditing. |
| protected  boolean | [**invokesStopCellEditing**](http://docs.google.com/javax/swing/JTree.html#invokesStopCellEditing)            If true, when editing is to be stopped by way of selection changing, data in tree changing or other means stopCellEditing is invoked, and changes are saved. |
| static [String](http://docs.google.com/java/lang/String.html) | [**LARGE\_MODEL\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#LARGE_MODEL_PROPERTY)            Bound property name for largeModel. |
| protected  boolean | [**largeModel**](http://docs.google.com/javax/swing/JTree.html#largeModel)            Is this tree a large model? This is a code-optimization setting. |
| static [String](http://docs.google.com/java/lang/String.html) | [**LEAD\_SELECTION\_PATH\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#LEAD_SELECTION_PATH_PROPERTY)            Bound property name for leadSelectionPath. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ROOT\_VISIBLE\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#ROOT_VISIBLE_PROPERTY)            Bound property name for rootVisible. |
| protected  boolean | [**rootVisible**](http://docs.google.com/javax/swing/JTree.html#rootVisible)            True if the root node is displayed, false if its children are the highest visible nodes. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ROW\_HEIGHT\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#ROW_HEIGHT_PROPERTY)            Bound property name for rowHeight. |
| protected  int | [**rowHeight**](http://docs.google.com/javax/swing/JTree.html#rowHeight)            Height to use for each display row. |
| static [String](http://docs.google.com/java/lang/String.html) | [**SCROLLS\_ON\_EXPAND\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#SCROLLS_ON_EXPAND_PROPERTY)            Bound property name for scrollsOnExpand. |
| protected  boolean | [**scrollsOnExpand**](http://docs.google.com/javax/swing/JTree.html#scrollsOnExpand)            If true, when a node is expanded, as many of the descendants are scrolled to be visible. |
| static [String](http://docs.google.com/java/lang/String.html) | [**SELECTION\_MODEL\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#SELECTION_MODEL_PROPERTY)            Bound property name for selectionModel. |
| protected  [TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html) | [**selectionModel**](http://docs.google.com/javax/swing/JTree.html#selectionModel)            Models the set of selected nodes in this tree. |
| protected  [JTree.TreeSelectionRedirector](http://docs.google.com/javax/swing/JTree.TreeSelectionRedirector.html) | [**selectionRedirector**](http://docs.google.com/javax/swing/JTree.html#selectionRedirector)            Creates a new event and passed it off the selectionListeners. |
| static [String](http://docs.google.com/java/lang/String.html) | [**SHOWS\_ROOT\_HANDLES\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#SHOWS_ROOT_HANDLES_PROPERTY)            Bound property name for showsRootHandles. |
| protected  boolean | [**showsRootHandles**](http://docs.google.com/javax/swing/JTree.html#showsRootHandles)            True if handles are displayed at the topmost level of the tree. |
| static [String](http://docs.google.com/java/lang/String.html) | [**TOGGLE\_CLICK\_COUNT\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#TOGGLE_CLICK_COUNT_PROPERTY)            Bound property name for toggleClickCount. |
| protected  int | [**toggleClickCount**](http://docs.google.com/javax/swing/JTree.html#toggleClickCount)            Number of mouse clicks before a node is expanded. |
| static [String](http://docs.google.com/java/lang/String.html) | [**TREE\_MODEL\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#TREE_MODEL_PROPERTY)            Bound property name for treeModel. |
| protected  [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) | [**treeModel**](http://docs.google.com/javax/swing/JTree.html#treeModel)            The model that defines the tree displayed by this object. |
| protected  [TreeModelListener](http://docs.google.com/javax/swing/event/TreeModelListener.html) | [**treeModelListener**](http://docs.google.com/javax/swing/JTree.html#treeModelListener)            Updates the expandedState. |
| static [String](http://docs.google.com/java/lang/String.html) | [**VISIBLE\_ROW\_COUNT\_PROPERTY**](http://docs.google.com/javax/swing/JTree.html#VISIBLE_ROW_COUNT_PROPERTY)            Bound property name for visibleRowCount. |
| protected  int | [**visibleRowCount**](http://docs.google.com/javax/swing/JTree.html#visibleRowCount)            Number of rows to make visible at one time. |

| **Fields inherited from class javax.swing.**[**JComponent**](http://docs.google.com/javax/swing/JComponent.html) |
| --- |
| [accessibleContext](http://docs.google.com/javax/swing/JComponent.html#accessibleContext), [listenerList](http://docs.google.com/javax/swing/JComponent.html#listenerList), [TOOL\_TIP\_TEXT\_KEY](http://docs.google.com/javax/swing/JComponent.html#TOOL_TIP_TEXT_KEY), [ui](http://docs.google.com/javax/swing/JComponent.html#ui), [UNDEFINED\_CONDITION](http://docs.google.com/javax/swing/JComponent.html#UNDEFINED_CONDITION), [WHEN\_ANCESTOR\_OF\_FOCUSED\_COMPONENT](http://docs.google.com/javax/swing/JComponent.html#WHEN_ANCESTOR_OF_FOCUSED_COMPONENT), [WHEN\_FOCUSED](http://docs.google.com/javax/swing/JComponent.html#WHEN_FOCUSED), [WHEN\_IN\_FOCUSED\_WINDOW](http://docs.google.com/javax/swing/JComponent.html#WHEN_IN_FOCUSED_WINDOW) |

| **Fields inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [BOTTOM\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#BOTTOM_ALIGNMENT), [CENTER\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#CENTER_ALIGNMENT), [LEFT\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#LEFT_ALIGNMENT), [RIGHT\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#RIGHT_ALIGNMENT), [TOP\_ALIGNMENT](http://docs.google.com/java/awt/Component.html#TOP_ALIGNMENT) |

| **Fields inherited from interface java.awt.image.**[**ImageObserver**](http://docs.google.com/java/awt/image/ImageObserver.html) |
| --- |
| [ABORT](http://docs.google.com/java/awt/image/ImageObserver.html#ABORT), [ALLBITS](http://docs.google.com/java/awt/image/ImageObserver.html#ALLBITS), [ERROR](http://docs.google.com/java/awt/image/ImageObserver.html#ERROR), [FRAMEBITS](http://docs.google.com/java/awt/image/ImageObserver.html#FRAMEBITS), [HEIGHT](http://docs.google.com/java/awt/image/ImageObserver.html#HEIGHT), [PROPERTIES](http://docs.google.com/java/awt/image/ImageObserver.html#PROPERTIES), [SOMEBITS](http://docs.google.com/java/awt/image/ImageObserver.html#SOMEBITS), [WIDTH](http://docs.google.com/java/awt/image/ImageObserver.html#WIDTH) |

| **Constructor Summary** | |
| --- | --- |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree())()            Returns a JTree with a sample model. |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree(java.util.Hashtable))([Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> value)            Returns a JTree created from a Hashtable which does not display with root. |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree(java.lang.Object%5B%5D))([Object](http://docs.google.com/java/lang/Object.html)[] value)            Returns a JTree with each element of the specified array as the child of a new root node which is not displayed. |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree(javax.swing.tree.TreeModel))([TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) newModel)            Returns an instance of JTree which displays the root node -- the tree is created using the specified data model. |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree(javax.swing.tree.TreeNode))([TreeNode](http://docs.google.com/javax/swing/tree/TreeNode.html) root)            Returns a JTree with the specified TreeNode as its root, which displays the root node. |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree(javax.swing.tree.TreeNode,%20boolean))([TreeNode](http://docs.google.com/javax/swing/tree/TreeNode.html) root, boolean asksAllowsChildren)            Returns a JTree with the specified TreeNode as its root, which displays the root node and which decides whether a node is a leaf node in the specified manner. |
| [**JTree**](http://docs.google.com/javax/swing/JTree.html#JTree(java.util.Vector))([Vector](http://docs.google.com/java/util/Vector.html)<?> value)            Returns a JTree with each element of the specified Vector as the child of a new root node which is not displayed. |

| **Method Summary** | |
| --- | --- |
| void | [**addSelectionInterval**](http://docs.google.com/javax/swing/JTree.html#addSelectionInterval(int,%20int))(int index0, int index1)            Adds the paths between index0 and index1, inclusive, to the selection. |
| void | [**addSelectionPath**](http://docs.google.com/javax/swing/JTree.html#addSelectionPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Adds the node identified by the specified TreePath to the current selection. |
| void | [**addSelectionPaths**](http://docs.google.com/javax/swing/JTree.html#addSelectionPaths(javax.swing.tree.TreePath%5B%5D))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] paths)            Adds each path in the array of paths to the current selection. |
| void | [**addSelectionRow**](http://docs.google.com/javax/swing/JTree.html#addSelectionRow(int))(int row)            Adds the path at the specified row to the current selection. |
| void | [**addSelectionRows**](http://docs.google.com/javax/swing/JTree.html#addSelectionRows(int%5B%5D))(int[] rows)            Adds the paths at each of the specified rows to the current selection. |
| void | [**addTreeExpansionListener**](http://docs.google.com/javax/swing/JTree.html#addTreeExpansionListener(javax.swing.event.TreeExpansionListener))([TreeExpansionListener](http://docs.google.com/javax/swing/event/TreeExpansionListener.html) tel)            Adds a listener for TreeExpansion events. |
| void | [**addTreeSelectionListener**](http://docs.google.com/javax/swing/JTree.html#addTreeSelectionListener(javax.swing.event.TreeSelectionListener))([TreeSelectionListener](http://docs.google.com/javax/swing/event/TreeSelectionListener.html) tsl)            Adds a listener for TreeSelection events. |
| void | [**addTreeWillExpandListener**](http://docs.google.com/javax/swing/JTree.html#addTreeWillExpandListener(javax.swing.event.TreeWillExpandListener))([TreeWillExpandListener](http://docs.google.com/javax/swing/event/TreeWillExpandListener.html) tel)            Adds a listener for TreeWillExpand events. |
| void | [**cancelEditing**](http://docs.google.com/javax/swing/JTree.html#cancelEditing())()            Cancels the current editing session. |
| void | [**clearSelection**](http://docs.google.com/javax/swing/JTree.html#clearSelection())()            Clears the selection. |
| protected  void | [**clearToggledPaths**](http://docs.google.com/javax/swing/JTree.html#clearToggledPaths())()            Clears the cache of toggled tree paths. |
| void | [**collapsePath**](http://docs.google.com/javax/swing/JTree.html#collapsePath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Ensures that the node identified by the specified path is collapsed and viewable. |
| void | [**collapseRow**](http://docs.google.com/javax/swing/JTree.html#collapseRow(int))(int row)            Ensures that the node in the specified row is collapsed. |
| [String](http://docs.google.com/java/lang/String.html) | [**convertValueToText**](http://docs.google.com/javax/swing/JTree.html#convertValueToText(java.lang.Object,%20boolean,%20boolean,%20boolean,%20int,%20boolean))([Object](http://docs.google.com/java/lang/Object.html) value, boolean selected, boolean expanded, boolean leaf, int row, boolean hasFocus)            Called by the renderers to convert the specified value to text. |
| protected static [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) | [**createTreeModel**](http://docs.google.com/javax/swing/JTree.html#createTreeModel(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) value)            Returns a TreeModel wrapping the specified object. |
| protected  [TreeModelListener](http://docs.google.com/javax/swing/event/TreeModelListener.html) | [**createTreeModelListener**](http://docs.google.com/javax/swing/JTree.html#createTreeModelListener())()            Creates and returns an instance of TreeModelHandler. |
| void | [**expandPath**](http://docs.google.com/javax/swing/JTree.html#expandPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Ensures that the node identified by the specified path is expanded and viewable. |
| void | [**expandRow**](http://docs.google.com/javax/swing/JTree.html#expandRow(int))(int row)            Ensures that the node in the specified row is expanded and viewable. |
| void | [**fireTreeCollapsed**](http://docs.google.com/javax/swing/JTree.html#fireTreeCollapsed(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Notifies all listeners that have registered interest for notification on this event type. |
| void | [**fireTreeExpanded**](http://docs.google.com/javax/swing/JTree.html#fireTreeExpanded(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Notifies all listeners that have registered interest for notification on this event type. |
| void | [**fireTreeWillCollapse**](http://docs.google.com/javax/swing/JTree.html#fireTreeWillCollapse(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Notifies all listeners that have registered interest for notification on this event type. |
| void | [**fireTreeWillExpand**](http://docs.google.com/javax/swing/JTree.html#fireTreeWillExpand(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Notifies all listeners that have registered interest for notification on this event type. |
| protected  void | [**fireValueChanged**](http://docs.google.com/javax/swing/JTree.html#fireValueChanged(javax.swing.event.TreeSelectionEvent))([TreeSelectionEvent](http://docs.google.com/javax/swing/event/TreeSelectionEvent.html) e)            Notifies all listeners that have registered interest for notification on this event type. |
| [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) | [**getAccessibleContext**](http://docs.google.com/javax/swing/JTree.html#getAccessibleContext())()            Gets the AccessibleContext associated with this JTree. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getAnchorSelectionPath**](http://docs.google.com/javax/swing/JTree.html#getAnchorSelectionPath())()            Returns the path identified as the anchor. |
| [TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) | [**getCellEditor**](http://docs.google.com/javax/swing/JTree.html#getCellEditor())()            Returns the editor used to edit entries in the tree. |
| [TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) | [**getCellRenderer**](http://docs.google.com/javax/swing/JTree.html#getCellRenderer())()            Returns the current TreeCellRenderer that is rendering each cell. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getClosestPathForLocation**](http://docs.google.com/javax/swing/JTree.html#getClosestPathForLocation(int,%20int))(int x, int y)            Returns the path to the node that is closest to x,y. |
| int | [**getClosestRowForLocation**](http://docs.google.com/javax/swing/JTree.html#getClosestRowForLocation(int,%20int))(int x, int y)            Returns the row to the node that is closest to x,y. |
| protected static [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) | [**getDefaultTreeModel**](http://docs.google.com/javax/swing/JTree.html#getDefaultTreeModel())()            Creates and returns a sample TreeModel. |
| protected  [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> | [**getDescendantToggledPaths**](http://docs.google.com/javax/swing/JTree.html#getDescendantToggledPaths(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) parent)            Returns an Enumeration of TreePaths that have been expanded that are descendants of parent. |
| boolean | [**getDragEnabled**](http://docs.google.com/javax/swing/JTree.html#getDragEnabled())()            Returns whether or not automatic drag handling is enabled. |
| [JTree.DropLocation](http://docs.google.com/javax/swing/JTree.DropLocation.html) | [**getDropLocation**](http://docs.google.com/javax/swing/JTree.html#getDropLocation())()            Returns the location that this component should visually indicate as the drop location during a DnD operation over the component, or null if no location is to currently be shown. |
| [DropMode](http://docs.google.com/javax/swing/DropMode.html) | [**getDropMode**](http://docs.google.com/javax/swing/JTree.html#getDropMode())()            Returns the drop mode for this component. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getEditingPath**](http://docs.google.com/javax/swing/JTree.html#getEditingPath())()            Returns the path to the element that is currently being edited. |
| [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> | [**getExpandedDescendants**](http://docs.google.com/javax/swing/JTree.html#getExpandedDescendants(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) parent)            Returns an Enumeration of the descendants of the path parent that are currently expanded. |
| boolean | [**getExpandsSelectedPaths**](http://docs.google.com/javax/swing/JTree.html#getExpandsSelectedPaths())()            Returns the expandsSelectedPaths property. |
| boolean | [**getInvokesStopCellEditing**](http://docs.google.com/javax/swing/JTree.html#getInvokesStopCellEditing())()            Returns the indicator that tells what happens when editing is interrupted. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getLastSelectedPathComponent**](http://docs.google.com/javax/swing/JTree.html#getLastSelectedPathComponent())()            Returns the last path component in the first node of the current selection. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getLeadSelectionPath**](http://docs.google.com/javax/swing/JTree.html#getLeadSelectionPath())()            Returns the path identified as the lead. |
| int | [**getLeadSelectionRow**](http://docs.google.com/javax/swing/JTree.html#getLeadSelectionRow())()            Returns the row index corresponding to the lead path. |
| int | [**getMaxSelectionRow**](http://docs.google.com/javax/swing/JTree.html#getMaxSelectionRow())()            Returns the last selected row. |
| int | [**getMinSelectionRow**](http://docs.google.com/javax/swing/JTree.html#getMinSelectionRow())()            Gets the first selected row. |
| [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) | [**getModel**](http://docs.google.com/javax/swing/JTree.html#getModel())()            Returns the TreeModel that is providing the data. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getNextMatch**](http://docs.google.com/javax/swing/JTree.html#getNextMatch(java.lang.String,%20int,%20javax.swing.text.Position.Bias))([String](http://docs.google.com/java/lang/String.html) prefix, int startingRow, [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) bias)            Returns the TreePath to the next tree element that begins with a prefix. |
| protected  [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] | [**getPathBetweenRows**](http://docs.google.com/javax/swing/JTree.html#getPathBetweenRows(int,%20int))(int index0, int index1)            Returns JTreePath instances representing the path between index0 and index1 (including index1). |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getPathBounds**](http://docs.google.com/javax/swing/JTree.html#getPathBounds(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns the Rectangle that the specified node will be drawn into. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getPathForLocation**](http://docs.google.com/javax/swing/JTree.html#getPathForLocation(int,%20int))(int x, int y)            Returns the path for the node at the specified location. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getPathForRow**](http://docs.google.com/javax/swing/JTree.html#getPathForRow(int))(int row)            Returns the path for the specified row. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredScrollableViewportSize**](http://docs.google.com/javax/swing/JTree.html#getPreferredScrollableViewportSize())()            Returns the preferred display size of a JTree. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getRowBounds**](http://docs.google.com/javax/swing/JTree.html#getRowBounds(int))(int row)            Returns the Rectangle that the node at the specified row is drawn in. |
| int | [**getRowCount**](http://docs.google.com/javax/swing/JTree.html#getRowCount())()            Returns the number of rows that are currently being displayed. |
| int | [**getRowForLocation**](http://docs.google.com/javax/swing/JTree.html#getRowForLocation(int,%20int))(int x, int y)            Returns the row for the specified location. |
| int | [**getRowForPath**](http://docs.google.com/javax/swing/JTree.html#getRowForPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns the row that displays the node identified by the specified path. |
| int | [**getRowHeight**](http://docs.google.com/javax/swing/JTree.html#getRowHeight())()            Returns the height of each row. |
| int | [**getScrollableBlockIncrement**](http://docs.google.com/javax/swing/JTree.html#getScrollableBlockIncrement(java.awt.Rectangle,%20int,%20int))([Rectangle](http://docs.google.com/java/awt/Rectangle.html) visibleRect, int orientation, int direction)            Returns the amount for a block increment, which is the height or width of visibleRect, based on orientation. |
| boolean | [**getScrollableTracksViewportHeight**](http://docs.google.com/javax/swing/JTree.html#getScrollableTracksViewportHeight())()            Returns false to indicate that the height of the viewport does not determine the height of the table, unless the preferred height of the tree is smaller than the viewports height. |
| boolean | [**getScrollableTracksViewportWidth**](http://docs.google.com/javax/swing/JTree.html#getScrollableTracksViewportWidth())()            Returns false to indicate that the width of the viewport does not determine the width of the table, unless the preferred width of the tree is smaller than the viewports width. |
| int | [**getScrollableUnitIncrement**](http://docs.google.com/javax/swing/JTree.html#getScrollableUnitIncrement(java.awt.Rectangle,%20int,%20int))([Rectangle](http://docs.google.com/java/awt/Rectangle.html) visibleRect, int orientation, int direction)            Returns the amount to increment when scrolling. |
| boolean | [**getScrollsOnExpand**](http://docs.google.com/javax/swing/JTree.html#getScrollsOnExpand())()            Returns the value of the scrollsOnExpand property. |
| int | [**getSelectionCount**](http://docs.google.com/javax/swing/JTree.html#getSelectionCount())()            Returns the number of nodes selected. |
| [TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html) | [**getSelectionModel**](http://docs.google.com/javax/swing/JTree.html#getSelectionModel())()            Returns the model for selections. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getSelectionPath**](http://docs.google.com/javax/swing/JTree.html#getSelectionPath())()            Returns the path to the first selected node. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] | [**getSelectionPaths**](http://docs.google.com/javax/swing/JTree.html#getSelectionPaths())()            Returns the paths of all selected values. |
| int[] | [**getSelectionRows**](http://docs.google.com/javax/swing/JTree.html#getSelectionRows())()            Returns all of the currently selected rows. |
| boolean | [**getShowsRootHandles**](http://docs.google.com/javax/swing/JTree.html#getShowsRootHandles())()            Returns the value of the showsRootHandles property. |
| int | [**getToggleClickCount**](http://docs.google.com/javax/swing/JTree.html#getToggleClickCount())()            Returns the number of mouse clicks needed to expand or close a node. |
| [String](http://docs.google.com/java/lang/String.html) | [**getToolTipText**](http://docs.google.com/javax/swing/JTree.html#getToolTipText(java.awt.event.MouseEvent))([MouseEvent](http://docs.google.com/java/awt/event/MouseEvent.html) event)            Overrides JComponent's getToolTipText method in order to allow renderer's tips to be used if it has text set. |
| [TreeExpansionListener](http://docs.google.com/javax/swing/event/TreeExpansionListener.html)[] | [**getTreeExpansionListeners**](http://docs.google.com/javax/swing/JTree.html#getTreeExpansionListeners())()            Returns an array of all the TreeExpansionListeners added to this JTree with addTreeExpansionListener(). |
| [TreeSelectionListener](http://docs.google.com/javax/swing/event/TreeSelectionListener.html)[] | [**getTreeSelectionListeners**](http://docs.google.com/javax/swing/JTree.html#getTreeSelectionListeners())()            Returns an array of all the TreeSelectionListeners added to this JTree with addTreeSelectionListener(). |
| [TreeWillExpandListener](http://docs.google.com/javax/swing/event/TreeWillExpandListener.html)[] | [**getTreeWillExpandListeners**](http://docs.google.com/javax/swing/JTree.html#getTreeWillExpandListeners())()            Returns an array of all the TreeWillExpandListeners added to this JTree with addTreeWillExpandListener(). |
| [TreeUI](http://docs.google.com/javax/swing/plaf/TreeUI.html) | [**getUI**](http://docs.google.com/javax/swing/JTree.html#getUI())()            Returns the L&F object that renders this component. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUIClassID**](http://docs.google.com/javax/swing/JTree.html#getUIClassID())()            Returns the name of the L&F class that renders this component. |
| int | [**getVisibleRowCount**](http://docs.google.com/javax/swing/JTree.html#getVisibleRowCount())()            Returns the number of rows that are displayed in the display area. |
| boolean | [**hasBeenExpanded**](http://docs.google.com/javax/swing/JTree.html#hasBeenExpanded(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns true if the node identified by the path has ever been expanded. |
| boolean | [**isCollapsed**](http://docs.google.com/javax/swing/JTree.html#isCollapsed(int))(int row)            Returns true if the node at the specified display row is collapsed. |
| boolean | [**isCollapsed**](http://docs.google.com/javax/swing/JTree.html#isCollapsed(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns true if the value identified by path is currently collapsed, this will return false if any of the values in path are currently not being displayed. |
| boolean | [**isEditable**](http://docs.google.com/javax/swing/JTree.html#isEditable())()            Returns true if the tree is editable. |
| boolean | [**isEditing**](http://docs.google.com/javax/swing/JTree.html#isEditing())()            Returns true if the tree is being edited. |
| boolean | [**isExpanded**](http://docs.google.com/javax/swing/JTree.html#isExpanded(int))(int row)            Returns true if the node at the specified display row is currently expanded. |
| boolean | [**isExpanded**](http://docs.google.com/javax/swing/JTree.html#isExpanded(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns true if the node identified by the path is currently expanded, |
| boolean | [**isFixedRowHeight**](http://docs.google.com/javax/swing/JTree.html#isFixedRowHeight())()            Returns true if the height of each display row is a fixed size. |
| boolean | [**isLargeModel**](http://docs.google.com/javax/swing/JTree.html#isLargeModel())()            Returns true if the tree is configured for a large model. |
| boolean | [**isPathEditable**](http://docs.google.com/javax/swing/JTree.html#isPathEditable(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns isEditable. |
| boolean | [**isPathSelected**](http://docs.google.com/javax/swing/JTree.html#isPathSelected(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns true if the item identified by the path is currently selected. |
| boolean | [**isRootVisible**](http://docs.google.com/javax/swing/JTree.html#isRootVisible())()            Returns true if the root node of the tree is displayed. |
| boolean | [**isRowSelected**](http://docs.google.com/javax/swing/JTree.html#isRowSelected(int))(int row)            Returns true if the node identified by row is selected. |
| boolean | [**isSelectionEmpty**](http://docs.google.com/javax/swing/JTree.html#isSelectionEmpty())()            Returns true if the selection is currently empty. |
| boolean | [**isVisible**](http://docs.google.com/javax/swing/JTree.html#isVisible(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns true if the value identified by path is currently viewable, which means it is either the root or all of its parents are expanded. |
| void | [**makeVisible**](http://docs.google.com/javax/swing/JTree.html#makeVisible(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Ensures that the node identified by path is currently viewable. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**paramString**](http://docs.google.com/javax/swing/JTree.html#paramString())()            Returns a string representation of this JTree. |
| protected  boolean | [**removeDescendantSelectedPaths**](http://docs.google.com/javax/swing/JTree.html#removeDescendantSelectedPaths(javax.swing.tree.TreePath,%20boolean))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path, boolean includePath)            Removes any paths in the selection that are descendants of path. |
| protected  void | [**removeDescendantToggledPaths**](http://docs.google.com/javax/swing/JTree.html#removeDescendantToggledPaths(java.util.Enumeration))([Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> toRemove)            Removes any descendants of the TreePaths in toRemove that have been expanded. |
| void | [**removeSelectionInterval**](http://docs.google.com/javax/swing/JTree.html#removeSelectionInterval(int,%20int))(int index0, int index1)            Removes the nodes between index0 and index1, inclusive, from the selection. |
| void | [**removeSelectionPath**](http://docs.google.com/javax/swing/JTree.html#removeSelectionPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Removes the node identified by the specified path from the current selection. |
| void | [**removeSelectionPaths**](http://docs.google.com/javax/swing/JTree.html#removeSelectionPaths(javax.swing.tree.TreePath%5B%5D))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] paths)            Removes the nodes identified by the specified paths from the current selection. |
| void | [**removeSelectionRow**](http://docs.google.com/javax/swing/JTree.html#removeSelectionRow(int))(int row)            Removes the row at the index row from the current selection. |
| void | [**removeSelectionRows**](http://docs.google.com/javax/swing/JTree.html#removeSelectionRows(int%5B%5D))(int[] rows)            Removes the rows that are selected at each of the specified rows. |
| void | [**removeTreeExpansionListener**](http://docs.google.com/javax/swing/JTree.html#removeTreeExpansionListener(javax.swing.event.TreeExpansionListener))([TreeExpansionListener](http://docs.google.com/javax/swing/event/TreeExpansionListener.html) tel)            Removes a listener for TreeExpansion events. |
| void | [**removeTreeSelectionListener**](http://docs.google.com/javax/swing/JTree.html#removeTreeSelectionListener(javax.swing.event.TreeSelectionListener))([TreeSelectionListener](http://docs.google.com/javax/swing/event/TreeSelectionListener.html) tsl)            Removes a TreeSelection listener. |
| void | [**removeTreeWillExpandListener**](http://docs.google.com/javax/swing/JTree.html#removeTreeWillExpandListener(javax.swing.event.TreeWillExpandListener))([TreeWillExpandListener](http://docs.google.com/javax/swing/event/TreeWillExpandListener.html) tel)            Removes a listener for TreeWillExpand events. |
| void | [**scrollPathToVisible**](http://docs.google.com/javax/swing/JTree.html#scrollPathToVisible(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Makes sure all the path components in path are expanded (except for the last path component) and scrolls so that the node identified by the path is displayed. |
| void | [**scrollRowToVisible**](http://docs.google.com/javax/swing/JTree.html#scrollRowToVisible(int))(int row)            Scrolls the item identified by row until it is displayed. |
| void | [**setAnchorSelectionPath**](http://docs.google.com/javax/swing/JTree.html#setAnchorSelectionPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) newPath)            Sets the path identified as the anchor. |
| void | [**setCellEditor**](http://docs.google.com/javax/swing/JTree.html#setCellEditor(javax.swing.tree.TreeCellEditor))([TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) cellEditor)            Sets the cell editor. |
| void | [**setCellRenderer**](http://docs.google.com/javax/swing/JTree.html#setCellRenderer(javax.swing.tree.TreeCellRenderer))([TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) x)            Sets the TreeCellRenderer that will be used to draw each cell. |
| void | [**setDragEnabled**](http://docs.google.com/javax/swing/JTree.html#setDragEnabled(boolean))(boolean b)            Turns on or off automatic drag handling. |
| void | [**setDropMode**](http://docs.google.com/javax/swing/JTree.html#setDropMode(javax.swing.DropMode))([DropMode](http://docs.google.com/javax/swing/DropMode.html) dropMode)            Sets the drop mode for this component. |
| void | [**setEditable**](http://docs.google.com/javax/swing/JTree.html#setEditable(boolean))(boolean flag)            Determines whether the tree is editable. |
| protected  void | [**setExpandedState**](http://docs.google.com/javax/swing/JTree.html#setExpandedState(javax.swing.tree.TreePath,%20boolean))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path, boolean state)            Sets the expanded state of this JTree. |
| void | [**setExpandsSelectedPaths**](http://docs.google.com/javax/swing/JTree.html#setExpandsSelectedPaths(boolean))(boolean newValue)            Configures the expandsSelectedPaths property. |
| void | [**setInvokesStopCellEditing**](http://docs.google.com/javax/swing/JTree.html#setInvokesStopCellEditing(boolean))(boolean newValue)            Determines what happens when editing is interrupted by selecting another node in the tree, a change in the tree's data, or by some other means. |
| void | [**setLargeModel**](http://docs.google.com/javax/swing/JTree.html#setLargeModel(boolean))(boolean newValue)            Specifies whether the UI should use a large model. |
| void | [**setLeadSelectionPath**](http://docs.google.com/javax/swing/JTree.html#setLeadSelectionPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) newPath)            Sets the path identifies as the lead. |
| void | [**setModel**](http://docs.google.com/javax/swing/JTree.html#setModel(javax.swing.tree.TreeModel))([TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) newModel)            Sets the TreeModel that will provide the data. |
| void | [**setRootVisible**](http://docs.google.com/javax/swing/JTree.html#setRootVisible(boolean))(boolean rootVisible)            Determines whether or not the root node from the TreeModel is visible. |
| void | [**setRowHeight**](http://docs.google.com/javax/swing/JTree.html#setRowHeight(int))(int rowHeight)            Sets the height of each cell, in pixels. |
| void | [**setScrollsOnExpand**](http://docs.google.com/javax/swing/JTree.html#setScrollsOnExpand(boolean))(boolean newValue)            Sets the scrollsOnExpand property, which determines whether the tree might scroll to show previously hidden children. |
| void | [**setSelectionInterval**](http://docs.google.com/javax/swing/JTree.html#setSelectionInterval(int,%20int))(int index0, int index1)            Selects the nodes between index0 and index1, inclusive. |
| void | [**setSelectionModel**](http://docs.google.com/javax/swing/JTree.html#setSelectionModel(javax.swing.tree.TreeSelectionModel))([TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html) selectionModel)            Sets the tree's selection model. |
| void | [**setSelectionPath**](http://docs.google.com/javax/swing/JTree.html#setSelectionPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Selects the node identified by the specified path. |
| void | [**setSelectionPaths**](http://docs.google.com/javax/swing/JTree.html#setSelectionPaths(javax.swing.tree.TreePath%5B%5D))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] paths)            Selects the nodes identified by the specified array of paths. |
| void | [**setSelectionRow**](http://docs.google.com/javax/swing/JTree.html#setSelectionRow(int))(int row)            Selects the node at the specified row in the display. |
| void | [**setSelectionRows**](http://docs.google.com/javax/swing/JTree.html#setSelectionRows(int%5B%5D))(int[] rows)            Selects the nodes corresponding to each of the specified rows in the display. |
| void | [**setShowsRootHandles**](http://docs.google.com/javax/swing/JTree.html#setShowsRootHandles(boolean))(boolean newValue)            Sets the value of the showsRootHandles property, which specifies whether the node handles should be displayed. |
| void | [**setToggleClickCount**](http://docs.google.com/javax/swing/JTree.html#setToggleClickCount(int))(int clickCount)            Sets the number of mouse clicks before a node will expand or close. |
| void | [**setUI**](http://docs.google.com/javax/swing/JTree.html#setUI(javax.swing.plaf.TreeUI))([TreeUI](http://docs.google.com/javax/swing/plaf/TreeUI.html) ui)            Sets the L&F object that renders this component. |
| void | [**setVisibleRowCount**](http://docs.google.com/javax/swing/JTree.html#setVisibleRowCount(int))(int newCount)            Sets the number of rows that are to be displayed. |
| void | [**startEditingAtPath**](http://docs.google.com/javax/swing/JTree.html#startEditingAtPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Selects the node identified by the specified path and initiates editing. |
| boolean | [**stopEditing**](http://docs.google.com/javax/swing/JTree.html#stopEditing())()            Ends the current editing session. |
| void | [**treeDidChange**](http://docs.google.com/javax/swing/JTree.html#treeDidChange())()            Sent when the tree has changed enough that we need to resize the bounds, but not enough that we need to remove the expanded node set (e.g nodes were expanded or collapsed, or nodes were inserted into the tree). |
| void | [**updateUI**](http://docs.google.com/javax/swing/JTree.html#updateUI())()            Notification from the UIManager that the L&F has changed. |

| **Methods inherited from class javax.swing.**[**JComponent**](http://docs.google.com/javax/swing/JComponent.html) |
| --- |
| [addAncestorListener](http://docs.google.com/javax/swing/JComponent.html#addAncestorListener(javax.swing.event.AncestorListener)), [addNotify](http://docs.google.com/javax/swing/JComponent.html#addNotify()), [addVetoableChangeListener](http://docs.google.com/javax/swing/JComponent.html#addVetoableChangeListener(java.beans.VetoableChangeListener)), [computeVisibleRect](http://docs.google.com/javax/swing/JComponent.html#computeVisibleRect(java.awt.Rectangle)), [contains](http://docs.google.com/javax/swing/JComponent.html#contains(int,%20int)), [createToolTip](http://docs.google.com/javax/swing/JComponent.html#createToolTip()), [disable](http://docs.google.com/javax/swing/JComponent.html#disable()), [enable](http://docs.google.com/javax/swing/JComponent.html#enable()), [firePropertyChange](http://docs.google.com/javax/swing/JComponent.html#firePropertyChange(java.lang.String,%20boolean,%20boolean)), [firePropertyChange](http://docs.google.com/javax/swing/JComponent.html#firePropertyChange(java.lang.String,%20char,%20char)), [firePropertyChange](http://docs.google.com/javax/swing/JComponent.html#firePropertyChange(java.lang.String,%20int,%20int)), [fireVetoableChange](http://docs.google.com/javax/swing/JComponent.html#fireVetoableChange(java.lang.String,%20java.lang.Object,%20java.lang.Object)), [getActionForKeyStroke](http://docs.google.com/javax/swing/JComponent.html#getActionForKeyStroke(javax.swing.KeyStroke)), [getActionMap](http://docs.google.com/javax/swing/JComponent.html#getActionMap()), [getAlignmentX](http://docs.google.com/javax/swing/JComponent.html#getAlignmentX()), [getAlignmentY](http://docs.google.com/javax/swing/JComponent.html#getAlignmentY()), [getAncestorListeners](http://docs.google.com/javax/swing/JComponent.html#getAncestorListeners()), [getAutoscrolls](http://docs.google.com/javax/swing/JComponent.html#getAutoscrolls()), [getBaseline](http://docs.google.com/javax/swing/JComponent.html#getBaseline(int,%20int)), [getBaselineResizeBehavior](http://docs.google.com/javax/swing/JComponent.html#getBaselineResizeBehavior()), [getBorder](http://docs.google.com/javax/swing/JComponent.html#getBorder()), [getBounds](http://docs.google.com/javax/swing/JComponent.html#getBounds(java.awt.Rectangle)), [getClientProperty](http://docs.google.com/javax/swing/JComponent.html#getClientProperty(java.lang.Object)), [getComponentGraphics](http://docs.google.com/javax/swing/JComponent.html#getComponentGraphics(java.awt.Graphics)), [getComponentPopupMenu](http://docs.google.com/javax/swing/JComponent.html#getComponentPopupMenu()), [getConditionForKeyStroke](http://docs.google.com/javax/swing/JComponent.html#getConditionForKeyStroke(javax.swing.KeyStroke)), [getDebugGraphicsOptions](http://docs.google.com/javax/swing/JComponent.html#getDebugGraphicsOptions()), [getDefaultLocale](http://docs.google.com/javax/swing/JComponent.html#getDefaultLocale()), [getFontMetrics](http://docs.google.com/javax/swing/JComponent.html#getFontMetrics(java.awt.Font)), [getGraphics](http://docs.google.com/javax/swing/JComponent.html#getGraphics()), [getHeight](http://docs.google.com/javax/swing/JComponent.html#getHeight()), [getInheritsPopupMenu](http://docs.google.com/javax/swing/JComponent.html#getInheritsPopupMenu()), [getInputMap](http://docs.google.com/javax/swing/JComponent.html#getInputMap()), [getInputMap](http://docs.google.com/javax/swing/JComponent.html#getInputMap(int)), [getInputVerifier](http://docs.google.com/javax/swing/JComponent.html#getInputVerifier()), [getInsets](http://docs.google.com/javax/swing/JComponent.html#getInsets()), [getInsets](http://docs.google.com/javax/swing/JComponent.html#getInsets(java.awt.Insets)), [getListeners](http://docs.google.com/javax/swing/JComponent.html#getListeners(java.lang.Class)), [getLocation](http://docs.google.com/javax/swing/JComponent.html#getLocation(java.awt.Point)), [getMaximumSize](http://docs.google.com/javax/swing/JComponent.html#getMaximumSize()), [getMinimumSize](http://docs.google.com/javax/swing/JComponent.html#getMinimumSize()), [getNextFocusableComponent](http://docs.google.com/javax/swing/JComponent.html#getNextFocusableComponent()), [getPopupLocation](http://docs.google.com/javax/swing/JComponent.html#getPopupLocation(java.awt.event.MouseEvent)), [getPreferredSize](http://docs.google.com/javax/swing/JComponent.html#getPreferredSize()), [getRegisteredKeyStrokes](http://docs.google.com/javax/swing/JComponent.html#getRegisteredKeyStrokes()), [getRootPane](http://docs.google.com/javax/swing/JComponent.html#getRootPane()), [getSize](http://docs.google.com/javax/swing/JComponent.html#getSize(java.awt.Dimension)), [getToolTipLocation](http://docs.google.com/javax/swing/JComponent.html#getToolTipLocation(java.awt.event.MouseEvent)), [getToolTipText](http://docs.google.com/javax/swing/JComponent.html#getToolTipText()), [getTopLevelAncestor](http://docs.google.com/javax/swing/JComponent.html#getTopLevelAncestor()), [getTransferHandler](http://docs.google.com/javax/swing/JComponent.html#getTransferHandler()), [getVerifyInputWhenFocusTarget](http://docs.google.com/javax/swing/JComponent.html#getVerifyInputWhenFocusTarget()), [getVetoableChangeListeners](http://docs.google.com/javax/swing/JComponent.html#getVetoableChangeListeners()), [getVisibleRect](http://docs.google.com/javax/swing/JComponent.html#getVisibleRect()), [getWidth](http://docs.google.com/javax/swing/JComponent.html#getWidth()), [getX](http://docs.google.com/javax/swing/JComponent.html#getX()), [getY](http://docs.google.com/javax/swing/JComponent.html#getY()), [grabFocus](http://docs.google.com/javax/swing/JComponent.html#grabFocus()), [isDoubleBuffered](http://docs.google.com/javax/swing/JComponent.html#isDoubleBuffered()), [isLightweightComponent](http://docs.google.com/javax/swing/JComponent.html#isLightweightComponent(java.awt.Component)), [isManagingFocus](http://docs.google.com/javax/swing/JComponent.html#isManagingFocus()), [isOpaque](http://docs.google.com/javax/swing/JComponent.html#isOpaque()), [isOptimizedDrawingEnabled](http://docs.google.com/javax/swing/JComponent.html#isOptimizedDrawingEnabled()), [isPaintingForPrint](http://docs.google.com/javax/swing/JComponent.html#isPaintingForPrint()), [isPaintingTile](http://docs.google.com/javax/swing/JComponent.html#isPaintingTile()), [isRequestFocusEnabled](http://docs.google.com/javax/swing/JComponent.html#isRequestFocusEnabled()), [isValidateRoot](http://docs.google.com/javax/swing/JComponent.html#isValidateRoot()), [paint](http://docs.google.com/javax/swing/JComponent.html#paint(java.awt.Graphics)), [paintBorder](http://docs.google.com/javax/swing/JComponent.html#paintBorder(java.awt.Graphics)), [paintChildren](http://docs.google.com/javax/swing/JComponent.html#paintChildren(java.awt.Graphics)), [paintComponent](http://docs.google.com/javax/swing/JComponent.html#paintComponent(java.awt.Graphics)), [paintImmediately](http://docs.google.com/javax/swing/JComponent.html#paintImmediately(int,%20int,%20int,%20int)), [paintImmediately](http://docs.google.com/javax/swing/JComponent.html#paintImmediately(java.awt.Rectangle)), [print](http://docs.google.com/javax/swing/JComponent.html#print(java.awt.Graphics)), [printAll](http://docs.google.com/javax/swing/JComponent.html#printAll(java.awt.Graphics)), [printBorder](http://docs.google.com/javax/swing/JComponent.html#printBorder(java.awt.Graphics)), [printChildren](http://docs.google.com/javax/swing/JComponent.html#printChildren(java.awt.Graphics)), [printComponent](http://docs.google.com/javax/swing/JComponent.html#printComponent(java.awt.Graphics)), [processComponentKeyEvent](http://docs.google.com/javax/swing/JComponent.html#processComponentKeyEvent(java.awt.event.KeyEvent)), [processKeyBinding](http://docs.google.com/javax/swing/JComponent.html#processKeyBinding(javax.swing.KeyStroke,%20java.awt.event.KeyEvent,%20int,%20boolean)), [processKeyEvent](http://docs.google.com/javax/swing/JComponent.html#processKeyEvent(java.awt.event.KeyEvent)), [processMouseEvent](http://docs.google.com/javax/swing/JComponent.html#processMouseEvent(java.awt.event.MouseEvent)), [processMouseMotionEvent](http://docs.google.com/javax/swing/JComponent.html#processMouseMotionEvent(java.awt.event.MouseEvent)), [putClientProperty](http://docs.google.com/javax/swing/JComponent.html#putClientProperty(java.lang.Object,%20java.lang.Object)), [registerKeyboardAction](http://docs.google.com/javax/swing/JComponent.html#registerKeyboardAction(java.awt.event.ActionListener,%20javax.swing.KeyStroke,%20int)), [registerKeyboardAction](http://docs.google.com/javax/swing/JComponent.html#registerKeyboardAction(java.awt.event.ActionListener,%20java.lang.String,%20javax.swing.KeyStroke,%20int)), [removeAncestorListener](http://docs.google.com/javax/swing/JComponent.html#removeAncestorListener(javax.swing.event.AncestorListener)), [removeNotify](http://docs.google.com/javax/swing/JComponent.html#removeNotify()), [removeVetoableChangeListener](http://docs.google.com/javax/swing/JComponent.html#removeVetoableChangeListener(java.beans.VetoableChangeListener)), [repaint](http://docs.google.com/javax/swing/JComponent.html#repaint(long,%20int,%20int,%20int,%20int)), [repaint](http://docs.google.com/javax/swing/JComponent.html#repaint(java.awt.Rectangle)), [requestDefaultFocus](http://docs.google.com/javax/swing/JComponent.html#requestDefaultFocus()), [requestFocus](http://docs.google.com/javax/swing/JComponent.html#requestFocus()), [requestFocus](http://docs.google.com/javax/swing/JComponent.html#requestFocus(boolean)), [requestFocusInWindow](http://docs.google.com/javax/swing/JComponent.html#requestFocusInWindow()), [requestFocusInWindow](http://docs.google.com/javax/swing/JComponent.html#requestFocusInWindow(boolean)), [resetKeyboardActions](http://docs.google.com/javax/swing/JComponent.html#resetKeyboardActions()), [reshape](http://docs.google.com/javax/swing/JComponent.html#reshape(int,%20int,%20int,%20int)), [revalidate](http://docs.google.com/javax/swing/JComponent.html#revalidate()), [scrollRectToVisible](http://docs.google.com/javax/swing/JComponent.html#scrollRectToVisible(java.awt.Rectangle)), [setActionMap](http://docs.google.com/javax/swing/JComponent.html#setActionMap(javax.swing.ActionMap)), [setAlignmentX](http://docs.google.com/javax/swing/JComponent.html#setAlignmentX(float)), [setAlignmentY](http://docs.google.com/javax/swing/JComponent.html#setAlignmentY(float)), [setAutoscrolls](http://docs.google.com/javax/swing/JComponent.html#setAutoscrolls(boolean)), [setBackground](http://docs.google.com/javax/swing/JComponent.html#setBackground(java.awt.Color)), [setBorder](http://docs.google.com/javax/swing/JComponent.html#setBorder(javax.swing.border.Border)), [setComponentPopupMenu](http://docs.google.com/javax/swing/JComponent.html#setComponentPopupMenu(javax.swing.JPopupMenu)), [setDebugGraphicsOptions](http://docs.google.com/javax/swing/JComponent.html#setDebugGraphicsOptions(int)), [setDefaultLocale](http://docs.google.com/javax/swing/JComponent.html#setDefaultLocale(java.util.Locale)), [setDoubleBuffered](http://docs.google.com/javax/swing/JComponent.html#setDoubleBuffered(boolean)), [setEnabled](http://docs.google.com/javax/swing/JComponent.html#setEnabled(boolean)), [setFocusTraversalKeys](http://docs.google.com/javax/swing/JComponent.html#setFocusTraversalKeys(int,%20java.util.Set)), [setFont](http://docs.google.com/javax/swing/JComponent.html#setFont(java.awt.Font)), [setForeground](http://docs.google.com/javax/swing/JComponent.html#setForeground(java.awt.Color)), [setInheritsPopupMenu](http://docs.google.com/javax/swing/JComponent.html#setInheritsPopupMenu(boolean)), [setInputMap](http://docs.google.com/javax/swing/JComponent.html#setInputMap(int,%20javax.swing.InputMap)), [setInputVerifier](http://docs.google.com/javax/swing/JComponent.html#setInputVerifier(javax.swing.InputVerifier)), [setMaximumSize](http://docs.google.com/javax/swing/JComponent.html#setMaximumSize(java.awt.Dimension)), [setMinimumSize](http://docs.google.com/javax/swing/JComponent.html#setMinimumSize(java.awt.Dimension)), [setNextFocusableComponent](http://docs.google.com/javax/swing/JComponent.html#setNextFocusableComponent(java.awt.Component)), [setOpaque](http://docs.google.com/javax/swing/JComponent.html#setOpaque(boolean)), [setPreferredSize](http://docs.google.com/javax/swing/JComponent.html#setPreferredSize(java.awt.Dimension)), [setRequestFocusEnabled](http://docs.google.com/javax/swing/JComponent.html#setRequestFocusEnabled(boolean)), [setToolTipText](http://docs.google.com/javax/swing/JComponent.html#setToolTipText(java.lang.String)), [setTransferHandler](http://docs.google.com/javax/swing/JComponent.html#setTransferHandler(javax.swing.TransferHandler)), [setUI](http://docs.google.com/javax/swing/JComponent.html#setUI(javax.swing.plaf.ComponentUI)), [setVerifyInputWhenFocusTarget](http://docs.google.com/javax/swing/JComponent.html#setVerifyInputWhenFocusTarget(boolean)), [setVisible](http://docs.google.com/javax/swing/JComponent.html#setVisible(boolean)), [unregisterKeyboardAction](http://docs.google.com/javax/swing/JComponent.html#unregisterKeyboardAction(javax.swing.KeyStroke)), [update](http://docs.google.com/javax/swing/JComponent.html#update(java.awt.Graphics)) |

| **Methods inherited from class java.awt.**[**Container**](http://docs.google.com/java/awt/Container.html) |
| --- |
| [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)), [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20int)), [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object)), [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object,%20int)), [add](http://docs.google.com/java/awt/Container.html#add(java.lang.String,%20java.awt.Component)), [addContainerListener](http://docs.google.com/java/awt/Container.html#addContainerListener(java.awt.event.ContainerListener)), [addImpl](http://docs.google.com/java/awt/Container.html#addImpl(java.awt.Component,%20java.lang.Object,%20int)), [addPropertyChangeListener](http://docs.google.com/java/awt/Container.html#addPropertyChangeListener(java.beans.PropertyChangeListener)), [addPropertyChangeListener](http://docs.google.com/java/awt/Container.html#addPropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [applyComponentOrientation](http://docs.google.com/java/awt/Container.html#applyComponentOrientation(java.awt.ComponentOrientation)), [areFocusTraversalKeysSet](http://docs.google.com/java/awt/Container.html#areFocusTraversalKeysSet(int)), [countComponents](http://docs.google.com/java/awt/Container.html#countComponents()), [deliverEvent](http://docs.google.com/java/awt/Container.html#deliverEvent(java.awt.Event)), [doLayout](http://docs.google.com/java/awt/Container.html#doLayout()), [findComponentAt](http://docs.google.com/java/awt/Container.html#findComponentAt(int,%20int)), [findComponentAt](http://docs.google.com/java/awt/Container.html#findComponentAt(java.awt.Point)), [getComponent](http://docs.google.com/java/awt/Container.html#getComponent(int)), [getComponentAt](http://docs.google.com/java/awt/Container.html#getComponentAt(int,%20int)), [getComponentAt](http://docs.google.com/java/awt/Container.html#getComponentAt(java.awt.Point)), [getComponentCount](http://docs.google.com/java/awt/Container.html#getComponentCount()), [getComponents](http://docs.google.com/java/awt/Container.html#getComponents()), [getComponentZOrder](http://docs.google.com/java/awt/Container.html#getComponentZOrder(java.awt.Component)), [getContainerListeners](http://docs.google.com/java/awt/Container.html#getContainerListeners()), [getFocusTraversalKeys](http://docs.google.com/java/awt/Container.html#getFocusTraversalKeys(int)), [getFocusTraversalPolicy](http://docs.google.com/java/awt/Container.html#getFocusTraversalPolicy()), [getLayout](http://docs.google.com/java/awt/Container.html#getLayout()), [getMousePosition](http://docs.google.com/java/awt/Container.html#getMousePosition(boolean)), [insets](http://docs.google.com/java/awt/Container.html#insets()), [invalidate](http://docs.google.com/java/awt/Container.html#invalidate()), [isAncestorOf](http://docs.google.com/java/awt/Container.html#isAncestorOf(java.awt.Component)), [isFocusCycleRoot](http://docs.google.com/java/awt/Container.html#isFocusCycleRoot()), [isFocusCycleRoot](http://docs.google.com/java/awt/Container.html#isFocusCycleRoot(java.awt.Container)), [isFocusTraversalPolicyProvider](http://docs.google.com/java/awt/Container.html#isFocusTraversalPolicyProvider()), [isFocusTraversalPolicySet](http://docs.google.com/java/awt/Container.html#isFocusTraversalPolicySet()), [layout](http://docs.google.com/java/awt/Container.html#layout()), [list](http://docs.google.com/java/awt/Container.html#list(java.io.PrintStream,%20int)), [list](http://docs.google.com/java/awt/Container.html#list(java.io.PrintWriter,%20int)), [locate](http://docs.google.com/java/awt/Container.html#locate(int,%20int)), [minimumSize](http://docs.google.com/java/awt/Container.html#minimumSize()), [paintComponents](http://docs.google.com/java/awt/Container.html#paintComponents(java.awt.Graphics)), [preferredSize](http://docs.google.com/java/awt/Container.html#preferredSize()), [printComponents](http://docs.google.com/java/awt/Container.html#printComponents(java.awt.Graphics)), [processContainerEvent](http://docs.google.com/java/awt/Container.html#processContainerEvent(java.awt.event.ContainerEvent)), [processEvent](http://docs.google.com/java/awt/Container.html#processEvent(java.awt.AWTEvent)), [remove](http://docs.google.com/java/awt/Container.html#remove(java.awt.Component)), [remove](http://docs.google.com/java/awt/Container.html#remove(int)), [removeAll](http://docs.google.com/java/awt/Container.html#removeAll()), [removeContainerListener](http://docs.google.com/java/awt/Container.html#removeContainerListener(java.awt.event.ContainerListener)), [setComponentZOrder](http://docs.google.com/java/awt/Container.html#setComponentZOrder(java.awt.Component,%20int)), [setFocusCycleRoot](http://docs.google.com/java/awt/Container.html#setFocusCycleRoot(boolean)), [setFocusTraversalPolicy](http://docs.google.com/java/awt/Container.html#setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)), [setFocusTraversalPolicyProvider](http://docs.google.com/java/awt/Container.html#setFocusTraversalPolicyProvider(boolean)), [setLayout](http://docs.google.com/java/awt/Container.html#setLayout(java.awt.LayoutManager)), [transferFocusBackward](http://docs.google.com/java/awt/Container.html#transferFocusBackward()), [transferFocusDownCycle](http://docs.google.com/java/awt/Container.html#transferFocusDownCycle()), [validate](http://docs.google.com/java/awt/Container.html#validate()), [validateTree](http://docs.google.com/java/awt/Container.html#validateTree()) |

| **Methods inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [action](http://docs.google.com/java/awt/Component.html#action(java.awt.Event,%20java.lang.Object)), [add](http://docs.google.com/java/awt/Component.html#add(java.awt.PopupMenu)), [addComponentListener](http://docs.google.com/java/awt/Component.html#addComponentListener(java.awt.event.ComponentListener)), [addFocusListener](http://docs.google.com/java/awt/Component.html#addFocusListener(java.awt.event.FocusListener)), [addHierarchyBoundsListener](http://docs.google.com/java/awt/Component.html#addHierarchyBoundsListener(java.awt.event.HierarchyBoundsListener)), [addHierarchyListener](http://docs.google.com/java/awt/Component.html#addHierarchyListener(java.awt.event.HierarchyListener)), [addInputMethodListener](http://docs.google.com/java/awt/Component.html#addInputMethodListener(java.awt.event.InputMethodListener)), [addKeyListener](http://docs.google.com/java/awt/Component.html#addKeyListener(java.awt.event.KeyListener)), [addMouseListener](http://docs.google.com/java/awt/Component.html#addMouseListener(java.awt.event.MouseListener)), [addMouseMotionListener](http://docs.google.com/java/awt/Component.html#addMouseMotionListener(java.awt.event.MouseMotionListener)), [addMouseWheelListener](http://docs.google.com/java/awt/Component.html#addMouseWheelListener(java.awt.event.MouseWheelListener)), [bounds](http://docs.google.com/java/awt/Component.html#bounds()), [checkImage](http://docs.google.com/java/awt/Component.html#checkImage(java.awt.Image,%20java.awt.image.ImageObserver)), [checkImage](http://docs.google.com/java/awt/Component.html#checkImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [coalesceEvents](http://docs.google.com/java/awt/Component.html#coalesceEvents(java.awt.AWTEvent,%20java.awt.AWTEvent)), [contains](http://docs.google.com/java/awt/Component.html#contains(java.awt.Point)), [createImage](http://docs.google.com/java/awt/Component.html#createImage(java.awt.image.ImageProducer)), [createImage](http://docs.google.com/java/awt/Component.html#createImage(int,%20int)), [createVolatileImage](http://docs.google.com/java/awt/Component.html#createVolatileImage(int,%20int)), [createVolatileImage](http://docs.google.com/java/awt/Component.html#createVolatileImage(int,%20int,%20java.awt.ImageCapabilities)), [disableEvents](http://docs.google.com/java/awt/Component.html#disableEvents(long)), [dispatchEvent](http://docs.google.com/java/awt/Component.html#dispatchEvent(java.awt.AWTEvent)), [enable](http://docs.google.com/java/awt/Component.html#enable(boolean)), [enableEvents](http://docs.google.com/java/awt/Component.html#enableEvents(long)), [enableInputMethods](http://docs.google.com/java/awt/Component.html#enableInputMethods(boolean)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20byte,%20byte)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20double,%20double)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20float,%20float)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20long,%20long)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20java.lang.Object,%20java.lang.Object)), [firePropertyChange](http://docs.google.com/java/awt/Component.html#firePropertyChange(java.lang.String,%20short,%20short)), [getBackground](http://docs.google.com/java/awt/Component.html#getBackground()), [getBounds](http://docs.google.com/java/awt/Component.html#getBounds()), [getColorModel](http://docs.google.com/java/awt/Component.html#getColorModel()), [getComponentListeners](http://docs.google.com/java/awt/Component.html#getComponentListeners()), [getComponentOrientation](http://docs.google.com/java/awt/Component.html#getComponentOrientation()), [getCursor](http://docs.google.com/java/awt/Component.html#getCursor()), [getDropTarget](http://docs.google.com/java/awt/Component.html#getDropTarget()), [getFocusCycleRootAncestor](http://docs.google.com/java/awt/Component.html#getFocusCycleRootAncestor()), [getFocusListeners](http://docs.google.com/java/awt/Component.html#getFocusListeners()), [getFocusTraversalKeysEnabled](http://docs.google.com/java/awt/Component.html#getFocusTraversalKeysEnabled()), [getFont](http://docs.google.com/java/awt/Component.html#getFont()), [getForeground](http://docs.google.com/java/awt/Component.html#getForeground()), [getGraphicsConfiguration](http://docs.google.com/java/awt/Component.html#getGraphicsConfiguration()), [getHierarchyBoundsListeners](http://docs.google.com/java/awt/Component.html#getHierarchyBoundsListeners()), [getHierarchyListeners](http://docs.google.com/java/awt/Component.html#getHierarchyListeners()), [getIgnoreRepaint](http://docs.google.com/java/awt/Component.html#getIgnoreRepaint()), [getInputContext](http://docs.google.com/java/awt/Component.html#getInputContext()), [getInputMethodListeners](http://docs.google.com/java/awt/Component.html#getInputMethodListeners()), [getInputMethodRequests](http://docs.google.com/java/awt/Component.html#getInputMethodRequests()), [getKeyListeners](http://docs.google.com/java/awt/Component.html#getKeyListeners()), [getLocale](http://docs.google.com/java/awt/Component.html#getLocale()), [getLocation](http://docs.google.com/java/awt/Component.html#getLocation()), [getLocationOnScreen](http://docs.google.com/java/awt/Component.html#getLocationOnScreen()), [getMouseListeners](http://docs.google.com/java/awt/Component.html#getMouseListeners()), [getMouseMotionListeners](http://docs.google.com/java/awt/Component.html#getMouseMotionListeners()), [getMousePosition](http://docs.google.com/java/awt/Component.html#getMousePosition()), [getMouseWheelListeners](http://docs.google.com/java/awt/Component.html#getMouseWheelListeners()), [getName](http://docs.google.com/java/awt/Component.html#getName()), [getParent](http://docs.google.com/java/awt/Component.html#getParent()), [getPeer](http://docs.google.com/java/awt/Component.html#getPeer()), [getPropertyChangeListeners](http://docs.google.com/java/awt/Component.html#getPropertyChangeListeners()), [getPropertyChangeListeners](http://docs.google.com/java/awt/Component.html#getPropertyChangeListeners(java.lang.String)), [getSize](http://docs.google.com/java/awt/Component.html#getSize()), [getToolkit](http://docs.google.com/java/awt/Component.html#getToolkit()), [getTreeLock](http://docs.google.com/java/awt/Component.html#getTreeLock()), [gotFocus](http://docs.google.com/java/awt/Component.html#gotFocus(java.awt.Event,%20java.lang.Object)), [handleEvent](http://docs.google.com/java/awt/Component.html#handleEvent(java.awt.Event)), [hasFocus](http://docs.google.com/java/awt/Component.html#hasFocus()), [hide](http://docs.google.com/java/awt/Component.html#hide()), [imageUpdate](http://docs.google.com/java/awt/Component.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int)), [inside](http://docs.google.com/java/awt/Component.html#inside(int,%20int)), [isBackgroundSet](http://docs.google.com/java/awt/Component.html#isBackgroundSet()), [isCursorSet](http://docs.google.com/java/awt/Component.html#isCursorSet()), [isDisplayable](http://docs.google.com/java/awt/Component.html#isDisplayable()), [isEnabled](http://docs.google.com/java/awt/Component.html#isEnabled()), [isFocusable](http://docs.google.com/java/awt/Component.html#isFocusable()), [isFocusOwner](http://docs.google.com/java/awt/Component.html#isFocusOwner()), [isFocusTraversable](http://docs.google.com/java/awt/Component.html#isFocusTraversable()), [isFontSet](http://docs.google.com/java/awt/Component.html#isFontSet()), [isForegroundSet](http://docs.google.com/java/awt/Component.html#isForegroundSet()), [isLightweight](http://docs.google.com/java/awt/Component.html#isLightweight()), [isMaximumSizeSet](http://docs.google.com/java/awt/Component.html#isMaximumSizeSet()), [isMinimumSizeSet](http://docs.google.com/java/awt/Component.html#isMinimumSizeSet()), [isPreferredSizeSet](http://docs.google.com/java/awt/Component.html#isPreferredSizeSet()), [isShowing](http://docs.google.com/java/awt/Component.html#isShowing()), [isValid](http://docs.google.com/java/awt/Component.html#isValid()), [isVisible](http://docs.google.com/java/awt/Component.html#isVisible()), [keyDown](http://docs.google.com/java/awt/Component.html#keyDown(java.awt.Event,%20int)), [keyUp](http://docs.google.com/java/awt/Component.html#keyUp(java.awt.Event,%20int)), [list](http://docs.google.com/java/awt/Component.html#list()), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintStream)), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintWriter)), [location](http://docs.google.com/java/awt/Component.html#location()), [lostFocus](http://docs.google.com/java/awt/Component.html#lostFocus(java.awt.Event,%20java.lang.Object)), [mouseDown](http://docs.google.com/java/awt/Component.html#mouseDown(java.awt.Event,%20int,%20int)), [mouseDrag](http://docs.google.com/java/awt/Component.html#mouseDrag(java.awt.Event,%20int,%20int)), [mouseEnter](http://docs.google.com/java/awt/Component.html#mouseEnter(java.awt.Event,%20int,%20int)), [mouseExit](http://docs.google.com/java/awt/Component.html#mouseExit(java.awt.Event,%20int,%20int)), [mouseMove](http://docs.google.com/java/awt/Component.html#mouseMove(java.awt.Event,%20int,%20int)), [mouseUp](http://docs.google.com/java/awt/Component.html#mouseUp(java.awt.Event,%20int,%20int)), [move](http://docs.google.com/java/awt/Component.html#move(int,%20int)), [nextFocus](http://docs.google.com/java/awt/Component.html#nextFocus()), [paintAll](http://docs.google.com/java/awt/Component.html#paintAll(java.awt.Graphics)), [postEvent](http://docs.google.com/java/awt/Component.html#postEvent(java.awt.Event)), [prepareImage](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20java.awt.image.ImageObserver)), [prepareImage](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [processComponentEvent](http://docs.google.com/java/awt/Component.html#processComponentEvent(java.awt.event.ComponentEvent)), [processFocusEvent](http://docs.google.com/java/awt/Component.html#processFocusEvent(java.awt.event.FocusEvent)), [processHierarchyBoundsEvent](http://docs.google.com/java/awt/Component.html#processHierarchyBoundsEvent(java.awt.event.HierarchyEvent)), [processHierarchyEvent](http://docs.google.com/java/awt/Component.html#processHierarchyEvent(java.awt.event.HierarchyEvent)), [processInputMethodEvent](http://docs.google.com/java/awt/Component.html#processInputMethodEvent(java.awt.event.InputMethodEvent)), [processMouseWheelEvent](http://docs.google.com/java/awt/Component.html#processMouseWheelEvent(java.awt.event.MouseWheelEvent)), [remove](http://docs.google.com/java/awt/Component.html#remove(java.awt.MenuComponent)), [removeComponentListener](http://docs.google.com/java/awt/Component.html#removeComponentListener(java.awt.event.ComponentListener)), [removeFocusListener](http://docs.google.com/java/awt/Component.html#removeFocusListener(java.awt.event.FocusListener)), [removeHierarchyBoundsListener](http://docs.google.com/java/awt/Component.html#removeHierarchyBoundsListener(java.awt.event.HierarchyBoundsListener)), [removeHierarchyListener](http://docs.google.com/java/awt/Component.html#removeHierarchyListener(java.awt.event.HierarchyListener)), [removeInputMethodListener](http://docs.google.com/java/awt/Component.html#removeInputMethodListener(java.awt.event.InputMethodListener)), [removeKeyListener](http://docs.google.com/java/awt/Component.html#removeKeyListener(java.awt.event.KeyListener)), [removeMouseListener](http://docs.google.com/java/awt/Component.html#removeMouseListener(java.awt.event.MouseListener)), [removeMouseMotionListener](http://docs.google.com/java/awt/Component.html#removeMouseMotionListener(java.awt.event.MouseMotionListener)), [removeMouseWheelListener](http://docs.google.com/java/awt/Component.html#removeMouseWheelListener(java.awt.event.MouseWheelListener)), [removePropertyChangeListener](http://docs.google.com/java/awt/Component.html#removePropertyChangeListener(java.beans.PropertyChangeListener)), [removePropertyChangeListener](http://docs.google.com/java/awt/Component.html#removePropertyChangeListener(java.lang.String,%20java.beans.PropertyChangeListener)), [repaint](http://docs.google.com/java/awt/Component.html#repaint()), [repaint](http://docs.google.com/java/awt/Component.html#repaint(int,%20int,%20int,%20int)), [repaint](http://docs.google.com/java/awt/Component.html#repaint(long)), [resize](http://docs.google.com/java/awt/Component.html#resize(java.awt.Dimension)), [resize](http://docs.google.com/java/awt/Component.html#resize(int,%20int)), [setBounds](http://docs.google.com/java/awt/Component.html#setBounds(int,%20int,%20int,%20int)), [setBounds](http://docs.google.com/java/awt/Component.html#setBounds(java.awt.Rectangle)), [setComponentOrientation](http://docs.google.com/java/awt/Component.html#setComponentOrientation(java.awt.ComponentOrientation)), [setCursor](http://docs.google.com/java/awt/Component.html#setCursor(java.awt.Cursor)), [setDropTarget](http://docs.google.com/java/awt/Component.html#setDropTarget(java.awt.dnd.DropTarget)), [setFocusable](http://docs.google.com/java/awt/Component.html#setFocusable(boolean)), [setFocusTraversalKeysEnabled](http://docs.google.com/java/awt/Component.html#setFocusTraversalKeysEnabled(boolean)), [setIgnoreRepaint](http://docs.google.com/java/awt/Component.html#setIgnoreRepaint(boolean)), [setLocale](http://docs.google.com/java/awt/Component.html#setLocale(java.util.Locale)), [setLocation](http://docs.google.com/java/awt/Component.html#setLocation(int,%20int)), [setLocation](http://docs.google.com/java/awt/Component.html#setLocation(java.awt.Point)), [setName](http://docs.google.com/java/awt/Component.html#setName(java.lang.String)), [setSize](http://docs.google.com/java/awt/Component.html#setSize(java.awt.Dimension)), [setSize](http://docs.google.com/java/awt/Component.html#setSize(int,%20int)), [show](http://docs.google.com/java/awt/Component.html#show()), [show](http://docs.google.com/java/awt/Component.html#show(boolean)), [size](http://docs.google.com/java/awt/Component.html#size()), [toString](http://docs.google.com/java/awt/Component.html#toString()), [transferFocus](http://docs.google.com/java/awt/Component.html#transferFocus()), [transferFocusUpCycle](http://docs.google.com/java/awt/Component.html#transferFocusUpCycle()) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Field Detail** |
| --- |

### treeModel

protected transient [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) **treeModel**

The model that defines the tree displayed by this object.

### selectionModel

protected transient [TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html) **selectionModel**

Models the set of selected nodes in this tree.

### rootVisible

protected boolean **rootVisible**

True if the root node is displayed, false if its children are the highest visible nodes.

### cellRenderer

protected transient [TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) **cellRenderer**

The cell used to draw nodes. If null, the UI uses a default cellRenderer.

### rowHeight

protected int **rowHeight**

Height to use for each display row. If this is <= 0 the renderer determines the height for each row.

### showsRootHandles

protected boolean **showsRootHandles**

True if handles are displayed at the topmost level of the tree.

A handle is a small icon that displays adjacent to the node which allows the user to click once to expand or collapse the node. A common interface shows a plus sign (+) for a node which can be expanded and a minus sign (-) for a node which can be collapsed. Handles are always shown for nodes below the topmost level.

If the rootVisible setting specifies that the root node is to be displayed, then that is the only node at the topmost level. If the root node is not displayed, then all of its children are at the topmost level of the tree. Handles are always displayed for nodes other than the topmost.

If the root node isn't visible, it is generally a good to make this value true. Otherwise, the tree looks exactly like a list, and users may not know that the "list entries" are actually tree nodes.

**See Also:**[rootVisible](http://docs.google.com/javax/swing/JTree.html#rootVisible)

### selectionRedirector

protected transient [JTree.TreeSelectionRedirector](http://docs.google.com/javax/swing/JTree.TreeSelectionRedirector.html) **selectionRedirector**

Creates a new event and passed it off the selectionListeners.

### cellEditor

protected transient [TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) **cellEditor**

Editor for the entries. Default is null (tree is not editable).

### editable

protected boolean **editable**

Is the tree editable? Default is false.

### largeModel

protected boolean **largeModel**

Is this tree a large model? This is a code-optimization setting. A large model can be used when the cell height is the same for all nodes. The UI will then cache very little information and instead continually message the model. Without a large model the UI caches most of the information, resulting in fewer method calls to the model.

This value is only a suggestion to the UI. Not all UIs will take advantage of it. Default value is false.

### visibleRowCount

protected int **visibleRowCount**

Number of rows to make visible at one time. This value is used for the Scrollable interface. It determines the preferred size of the display area.

### invokesStopCellEditing

protected boolean **invokesStopCellEditing**

If true, when editing is to be stopped by way of selection changing, data in tree changing or other means stopCellEditing is invoked, and changes are saved. If false, cancelCellEditing is invoked, and changes are discarded. Default is false.

### scrollsOnExpand

protected boolean **scrollsOnExpand**

If true, when a node is expanded, as many of the descendants are scrolled to be visible.

### toggleClickCount

protected int **toggleClickCount**

Number of mouse clicks before a node is expanded.

### treeModelListener

protected transient [TreeModelListener](http://docs.google.com/javax/swing/event/TreeModelListener.html) **treeModelListener**

Updates the expandedState.

### CELL\_RENDERER\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **CELL\_RENDERER\_PROPERTY**

Bound property name for cellRenderer.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.CELL_RENDERER_PROPERTY)

### TREE\_MODEL\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **TREE\_MODEL\_PROPERTY**

Bound property name for treeModel.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.TREE_MODEL_PROPERTY)

### ROOT\_VISIBLE\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **ROOT\_VISIBLE\_PROPERTY**

Bound property name for rootVisible.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.ROOT_VISIBLE_PROPERTY)

### SHOWS\_ROOT\_HANDLES\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **SHOWS\_ROOT\_HANDLES\_PROPERTY**

Bound property name for showsRootHandles.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.SHOWS_ROOT_HANDLES_PROPERTY)

### ROW\_HEIGHT\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **ROW\_HEIGHT\_PROPERTY**

Bound property name for rowHeight.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.ROW_HEIGHT_PROPERTY)

### CELL\_EDITOR\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **CELL\_EDITOR\_PROPERTY**

Bound property name for cellEditor.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.CELL_EDITOR_PROPERTY)

### EDITABLE\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **EDITABLE\_PROPERTY**

Bound property name for editable.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.EDITABLE_PROPERTY)

### LARGE\_MODEL\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **LARGE\_MODEL\_PROPERTY**

Bound property name for largeModel.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.LARGE_MODEL_PROPERTY)

### SELECTION\_MODEL\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **SELECTION\_MODEL\_PROPERTY**

Bound property name for selectionModel.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.SELECTION_MODEL_PROPERTY)

### VISIBLE\_ROW\_COUNT\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **VISIBLE\_ROW\_COUNT\_PROPERTY**

Bound property name for visibleRowCount.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.VISIBLE_ROW_COUNT_PROPERTY)

### INVOKES\_STOP\_CELL\_EDITING\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **INVOKES\_STOP\_CELL\_EDITING\_PROPERTY**

Bound property name for messagesStopCellEditing.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.INVOKES_STOP_CELL_EDITING_PROPERTY)

### SCROLLS\_ON\_EXPAND\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **SCROLLS\_ON\_EXPAND\_PROPERTY**

Bound property name for scrollsOnExpand.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.SCROLLS_ON_EXPAND_PROPERTY)

### TOGGLE\_CLICK\_COUNT\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **TOGGLE\_CLICK\_COUNT\_PROPERTY**

Bound property name for toggleClickCount.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.TOGGLE_CLICK_COUNT_PROPERTY)

### LEAD\_SELECTION\_PATH\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **LEAD\_SELECTION\_PATH\_PROPERTY**

Bound property name for leadSelectionPath.

**Since:** 1.3 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.LEAD_SELECTION_PATH_PROPERTY)

### ANCHOR\_SELECTION\_PATH\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **ANCHOR\_SELECTION\_PATH\_PROPERTY**

Bound property name for anchor selection path.

**Since:** 1.3 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.ANCHOR_SELECTION_PATH_PROPERTY)

### EXPANDS\_SELECTED\_PATHS\_PROPERTY

public static final [String](http://docs.google.com/java/lang/String.html) **EXPANDS\_SELECTED\_PATHS\_PROPERTY**

Bound property name for expands selected paths property

**Since:** 1.3 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JTree.EXPANDS_SELECTED_PATHS_PROPERTY)

| **Constructor Detail** |
| --- |

### JTree

public **JTree**()

Returns a JTree with a sample model. The default model used by the tree defines a leaf node as any node without children.

**See Also:**[DefaultTreeModel.asksAllowsChildren](http://docs.google.com/javax/swing/tree/DefaultTreeModel.html#asksAllowsChildren)

### JTree

public **JTree**([Object](http://docs.google.com/java/lang/Object.html)[] value)

Returns a JTree with each element of the specified array as the child of a new root node which is not displayed. By default, the tree defines a leaf node as any node without children.

**Parameters:**value - an array of Objects**See Also:**[DefaultTreeModel.asksAllowsChildren](http://docs.google.com/javax/swing/tree/DefaultTreeModel.html#asksAllowsChildren)

### JTree

public **JTree**([Vector](http://docs.google.com/java/util/Vector.html)<?> value)

Returns a JTree with each element of the specified Vector as the child of a new root node which is not displayed. By default, the tree defines a leaf node as any node without children.

**Parameters:**value - a Vector**See Also:**[DefaultTreeModel.asksAllowsChildren](http://docs.google.com/javax/swing/tree/DefaultTreeModel.html#asksAllowsChildren)

### JTree

public **JTree**([Hashtable](http://docs.google.com/java/util/Hashtable.html)<?,?> value)

Returns a JTree created from a Hashtable which does not display with root. Each value-half of the key/value pairs in the HashTable becomes a child of the new root node. By default, the tree defines a leaf node as any node without children.

**Parameters:**value - a Hashtable**See Also:**[DefaultTreeModel.asksAllowsChildren](http://docs.google.com/javax/swing/tree/DefaultTreeModel.html#asksAllowsChildren)

### JTree

public **JTree**([TreeNode](http://docs.google.com/javax/swing/tree/TreeNode.html) root)

Returns a JTree with the specified TreeNode as its root, which displays the root node. By default, the tree defines a leaf node as any node without children.

**Parameters:**root - a TreeNode object**See Also:**[DefaultTreeModel.asksAllowsChildren](http://docs.google.com/javax/swing/tree/DefaultTreeModel.html#asksAllowsChildren)

### JTree

public **JTree**([TreeNode](http://docs.google.com/javax/swing/tree/TreeNode.html) root,  
 boolean asksAllowsChildren)

Returns a JTree with the specified TreeNode as its root, which displays the root node and which decides whether a node is a leaf node in the specified manner.

**Parameters:**root - a TreeNode objectasksAllowsChildren - if false, any node without children is a leaf node; if true, only nodes that do not allow children are leaf nodes**See Also:**[DefaultTreeModel.asksAllowsChildren](http://docs.google.com/javax/swing/tree/DefaultTreeModel.html#asksAllowsChildren)

### JTree

public **JTree**([TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) newModel)

Returns an instance of JTree which displays the root node -- the tree is created using the specified data model.

**Parameters:**newModel - the TreeModel to use as the data model

| **Method Detail** |
| --- |

### getDefaultTreeModel

protected static [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) **getDefaultTreeModel**()

Creates and returns a sample TreeModel. Used primarily for beanbuilders to show something interesting.

**Returns:**the default TreeModel

### createTreeModel

protected static [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) **createTreeModel**([Object](http://docs.google.com/java/lang/Object.html) value)

Returns a TreeModel wrapping the specified object. If the object is:

* an array of Objects,
* a Hashtable, or
* a Vector

then a new root node is created with each of the incoming objects as children. Otherwise, a new root is created with the specified object as its value.

**Parameters:**value - the Object used as the foundation for the TreeModel **Returns:**a TreeModel wrapping the specified object

### getUI

public [TreeUI](http://docs.google.com/javax/swing/plaf/TreeUI.html) **getUI**()

Returns the L&F object that renders this component.

**Returns:**the TreeUI object that renders this component

### setUI

public void **setUI**([TreeUI](http://docs.google.com/javax/swing/plaf/TreeUI.html) ui)

Sets the L&F object that renders this component.

**Parameters:**ui - the TreeUI L&F object**See Also:**[UIDefaults.getUI(javax.swing.JComponent)](http://docs.google.com/javax/swing/UIDefaults.html#getUI(javax.swing.JComponent))

### updateUI

public void **updateUI**()

Notification from the UIManager that the L&F has changed. Replaces the current UI object with the latest version from the UIManager.

**Overrides:**[updateUI](http://docs.google.com/javax/swing/JComponent.html#updateUI()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **See Also:**[JComponent.updateUI()](http://docs.google.com/javax/swing/JComponent.html#updateUI())

### getUIClassID

public [String](http://docs.google.com/java/lang/String.html) **getUIClassID**()

Returns the name of the L&F class that renders this component.

**Overrides:**[getUIClassID](http://docs.google.com/javax/swing/JComponent.html#getUIClassID()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Returns:**the string "TreeUI"**See Also:**[JComponent.getUIClassID()](http://docs.google.com/javax/swing/JComponent.html#getUIClassID()), [UIDefaults.getUI(javax.swing.JComponent)](http://docs.google.com/javax/swing/UIDefaults.html#getUI(javax.swing.JComponent))

### getCellRenderer

public [TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) **getCellRenderer**()

Returns the current TreeCellRenderer that is rendering each cell.

**Returns:**the TreeCellRenderer that is rendering each cell

### setCellRenderer

public void **setCellRenderer**([TreeCellRenderer](http://docs.google.com/javax/swing/tree/TreeCellRenderer.html) x)

Sets the TreeCellRenderer that will be used to draw each cell.

**Parameters:**x - the TreeCellRenderer that is to render each cell

### setEditable

public void **setEditable**(boolean flag)

Determines whether the tree is editable. Fires a property change event if the new setting is different from the existing setting.

**Parameters:**flag - a boolean value, true if the tree is editable

### isEditable

public boolean **isEditable**()

Returns true if the tree is editable.

**Returns:**true if the tree is editable

### setCellEditor

public void **setCellEditor**([TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) cellEditor)

Sets the cell editor. A null value implies that the tree cannot be edited. If this represents a change in the cellEditor, the propertyChange method is invoked on all listeners.

**Parameters:**cellEditor - the TreeCellEditor to use

### getCellEditor

public [TreeCellEditor](http://docs.google.com/javax/swing/tree/TreeCellEditor.html) **getCellEditor**()

Returns the editor used to edit entries in the tree.

**Returns:**the TreeCellEditor in use, or null if the tree cannot be edited

### getModel

public [TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) **getModel**()

Returns the TreeModel that is providing the data.

**Returns:**the TreeModel that is providing the data

### setModel

public void **setModel**([TreeModel](http://docs.google.com/javax/swing/tree/TreeModel.html) newModel)

Sets the TreeModel that will provide the data.

**Parameters:**newModel - the TreeModel that is to provide the data

### isRootVisible

public boolean **isRootVisible**()

Returns true if the root node of the tree is displayed.

**Returns:**true if the root node of the tree is displayed**See Also:**[rootVisible](http://docs.google.com/javax/swing/JTree.html#rootVisible)

### setRootVisible

public void **setRootVisible**(boolean rootVisible)

Determines whether or not the root node from the TreeModel is visible.

**Parameters:**rootVisible - true if the root node of the tree is to be displayed**See Also:**[rootVisible](http://docs.google.com/javax/swing/JTree.html#rootVisible)

### setShowsRootHandles

public void **setShowsRootHandles**(boolean newValue)

Sets the value of the showsRootHandles property, which specifies whether the node handles should be displayed. The default value of this property depends on the constructor used to create the JTree. Some look and feels might not support handles; they will ignore this property.

**Parameters:**newValue - true if root handles should be displayed; otherwise, false**See Also:**[showsRootHandles](http://docs.google.com/javax/swing/JTree.html#showsRootHandles), [getShowsRootHandles()](http://docs.google.com/javax/swing/JTree.html#getShowsRootHandles())

### getShowsRootHandles

public boolean **getShowsRootHandles**()

Returns the value of the showsRootHandles property.

**Returns:**the value of the showsRootHandles property**See Also:**[showsRootHandles](http://docs.google.com/javax/swing/JTree.html#showsRootHandles)

### setRowHeight

public void **setRowHeight**(int rowHeight)

Sets the height of each cell, in pixels. If the specified value is less than or equal to zero the current cell renderer is queried for each row's height.

**Parameters:**rowHeight - the height of each cell, in pixels

### getRowHeight

public int **getRowHeight**()

Returns the height of each row. If the returned value is less than or equal to 0 the height for each row is determined by the renderer.

### isFixedRowHeight

public boolean **isFixedRowHeight**()

Returns true if the height of each display row is a fixed size.

**Returns:**true if the height of each row is a fixed size

### setLargeModel

public void **setLargeModel**(boolean newValue)

Specifies whether the UI should use a large model. (Not all UIs will implement this.) Fires a property change for the LARGE\_MODEL\_PROPERTY.

**Parameters:**newValue - true to suggest a large model to the UI**See Also:**[largeModel](http://docs.google.com/javax/swing/JTree.html#largeModel)

### isLargeModel

public boolean **isLargeModel**()

Returns true if the tree is configured for a large model.

**Returns:**true if a large model is suggested**See Also:**[largeModel](http://docs.google.com/javax/swing/JTree.html#largeModel)

### setInvokesStopCellEditing

public void **setInvokesStopCellEditing**(boolean newValue)

Determines what happens when editing is interrupted by selecting another node in the tree, a change in the tree's data, or by some other means. Setting this property to true causes the changes to be automatically saved when editing is interrupted.

Fires a property change for the INVOKES\_STOP\_CELL\_EDITING\_PROPERTY.

**Parameters:**newValue - true means that stopCellEditing is invoked when editing is interrupted, and data is saved; false means that cancelCellEditing is invoked, and changes are lost

### getInvokesStopCellEditing

public boolean **getInvokesStopCellEditing**()

Returns the indicator that tells what happens when editing is interrupted.

**Returns:**the indicator that tells what happens when editing is interrupted**See Also:**[setInvokesStopCellEditing(boolean)](http://docs.google.com/javax/swing/JTree.html#setInvokesStopCellEditing(boolean))

### setScrollsOnExpand

public void **setScrollsOnExpand**(boolean newValue)

Sets the scrollsOnExpand property, which determines whether the tree might scroll to show previously hidden children. If this property is true (the default), when a node expands the tree can use scrolling to make the maximum possible number of the node's descendants visible. In some look and feels, trees might not need to scroll when expanded; those look and feels will ignore this property.

**Parameters:**newValue - false to disable scrolling on expansion; true to enable it**See Also:**[getScrollsOnExpand()](http://docs.google.com/javax/swing/JTree.html#getScrollsOnExpand())

### getScrollsOnExpand

public boolean **getScrollsOnExpand**()

Returns the value of the scrollsOnExpand property.

**Returns:**the value of the scrollsOnExpand property

### setToggleClickCount

public void **setToggleClickCount**(int clickCount)

Sets the number of mouse clicks before a node will expand or close. The default is two.

**Since:** 1.3

### getToggleClickCount

public int **getToggleClickCount**()

Returns the number of mouse clicks needed to expand or close a node.

**Returns:**number of mouse clicks before node is expanded**Since:** 1.3

### setExpandsSelectedPaths

public void **setExpandsSelectedPaths**(boolean newValue)

Configures the expandsSelectedPaths property. If true, any time the selection is changed, either via the TreeSelectionModel, or the cover methods provided by JTree, the TreePaths parents will be expanded to make them visible (visible meaning the parent path is expanded, not necessarily in the visible rectangle of the JTree). If false, when the selection changes the nodes parent is not made visible (all its parents expanded). This is useful if you wish to have your selection model maintain paths that are not always visible (all parents expanded).

**Parameters:**newValue - the new value for expandsSelectedPaths**Since:** 1.3

### getExpandsSelectedPaths

public boolean **getExpandsSelectedPaths**()

Returns the expandsSelectedPaths property.

**Returns:**true if selection changes result in the parent path being expanded**Since:** 1.3 **See Also:**[setExpandsSelectedPaths(boolean)](http://docs.google.com/javax/swing/JTree.html#setExpandsSelectedPaths(boolean))

### setDragEnabled

public void **setDragEnabled**(boolean b)

Turns on or off automatic drag handling. In order to enable automatic drag handling, this property should be set to true, and the tree's TransferHandler needs to be non-null. The default value of the dragEnabled property is false.

The job of honoring this property, and recognizing a user drag gesture, lies with the look and feel implementation, and in particular, the tree's TreeUI. When automatic drag handling is enabled, most look and feels (including those that subclass BasicLookAndFeel) begin a drag and drop operation whenever the user presses the mouse button over an item and then moves the mouse a few pixels. Setting this property to true can therefore have a subtle effect on how selections behave.

If a look and feel is used that ignores this property, you can still begin a drag and drop operation by calling exportAsDrag on the tree's TransferHandler.

**Parameters:**b - whether or not to enable automatic drag handling **Throws:** [HeadlessException](http://docs.google.com/java/awt/HeadlessException.html) - if b is true and GraphicsEnvironment.isHeadless() returns true**Since:** 1.4 **See Also:**[GraphicsEnvironment.isHeadless()](http://docs.google.com/java/awt/GraphicsEnvironment.html#isHeadless()), [getDragEnabled()](http://docs.google.com/javax/swing/JTree.html#getDragEnabled()), [JComponent.setTransferHandler(javax.swing.TransferHandler)](http://docs.google.com/javax/swing/JComponent.html#setTransferHandler(javax.swing.TransferHandler)), [TransferHandler](http://docs.google.com/javax/swing/TransferHandler.html)

### getDragEnabled

public boolean **getDragEnabled**()

Returns whether or not automatic drag handling is enabled.

**Returns:**the value of the dragEnabled property**Since:** 1.4 **See Also:**[setDragEnabled(boolean)](http://docs.google.com/javax/swing/JTree.html#setDragEnabled(boolean))

### setDropMode

public final void **setDropMode**([DropMode](http://docs.google.com/javax/swing/DropMode.html) dropMode)

Sets the drop mode for this component. For backward compatibility, the default for this property is DropMode.USE\_SELECTION. Usage of one of the other modes is recommended, however, for an improved user experience. DropMode.ON, for instance, offers similar behavior of showing items as selected, but does so without affecting the actual selection in the tree.

JTree supports the following drop modes:

* DropMode.USE\_SELECTION
* DropMode.ON
* DropMode.INSERT
* DropMode.ON\_OR\_INSERT

The drop mode is only meaningful if this component has a TransferHandler that accepts drops.

**Parameters:**dropMode - the drop mode to use **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the drop mode is unsupported or null**Since:** 1.6 **See Also:**[getDropMode()](http://docs.google.com/javax/swing/JTree.html#getDropMode()), [getDropLocation()](http://docs.google.com/javax/swing/JTree.html#getDropLocation()), [JComponent.setTransferHandler(javax.swing.TransferHandler)](http://docs.google.com/javax/swing/JComponent.html#setTransferHandler(javax.swing.TransferHandler)), [TransferHandler](http://docs.google.com/javax/swing/TransferHandler.html)

### getDropMode

public final [DropMode](http://docs.google.com/javax/swing/DropMode.html) **getDropMode**()

Returns the drop mode for this component.

**Returns:**the drop mode for this component**Since:** 1.6 **See Also:**[setDropMode(javax.swing.DropMode)](http://docs.google.com/javax/swing/JTree.html#setDropMode(javax.swing.DropMode))

### getDropLocation

public final [JTree.DropLocation](http://docs.google.com/javax/swing/JTree.DropLocation.html) **getDropLocation**()

Returns the location that this component should visually indicate as the drop location during a DnD operation over the component, or null if no location is to currently be shown.

This method is not meant for querying the drop location from a TransferHandler, as the drop location is only set after the TransferHandler's canImport has returned and has allowed for the location to be shown.

When this property changes, a property change event with name "dropLocation" is fired by the component.

**Returns:**the drop location**Since:** 1.6 **See Also:**[setDropMode(javax.swing.DropMode)](http://docs.google.com/javax/swing/JTree.html#setDropMode(javax.swing.DropMode)), [TransferHandler.canImport(TransferHandler.TransferSupport)](http://docs.google.com/javax/swing/TransferHandler.html#canImport(javax.swing.TransferHandler.TransferSupport))

### isPathEditable

public boolean **isPathEditable**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns isEditable. This is invoked from the UI before editing begins to insure that the given path can be edited. This is provided as an entry point for subclassers to add filtered editing without having to resort to creating a new editor.

**Returns:**true if every parent node and the node itself is editable**See Also:**[isEditable()](http://docs.google.com/javax/swing/JTree.html#isEditable())

### getToolTipText

public [String](http://docs.google.com/java/lang/String.html) **getToolTipText**([MouseEvent](http://docs.google.com/java/awt/event/MouseEvent.html) event)

Overrides JComponent's getToolTipText method in order to allow renderer's tips to be used if it has text set.

NOTE: For JTree to properly display tooltips of its renderers, JTree must be a registered component with the ToolTipManager. This can be done by invoking ToolTipManager.sharedInstance().registerComponent(tree). This is not done automatically!

**Overrides:**[getToolTipText](http://docs.google.com/javax/swing/JComponent.html#getToolTipText(java.awt.event.MouseEvent)) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Parameters:**event - the MouseEvent that initiated the ToolTip display **Returns:**a string containing the tooltip or null if event is null

### convertValueToText

public [String](http://docs.google.com/java/lang/String.html) **convertValueToText**([Object](http://docs.google.com/java/lang/Object.html) value,  
 boolean selected,  
 boolean expanded,  
 boolean leaf,  
 int row,  
 boolean hasFocus)

Called by the renderers to convert the specified value to text. This implementation returns value.toString, ignoring all other arguments. To control the conversion, subclass this method and use any of the arguments you need.

**Parameters:**value - the Object to convert to textselected - true if the node is selectedexpanded - true if the node is expandedleaf - true if the node is a leaf noderow - an integer specifying the node's display row, where 0 is the first row in the displayhasFocus - true if the node has the focus **Returns:**the String representation of the node's value

### getRowCount

public int **getRowCount**()

Returns the number of rows that are currently being displayed.

**Returns:**the number of rows that are being displayed

### setSelectionPath

public void **setSelectionPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Selects the node identified by the specified path. If any component of the path is hidden (under a collapsed node), and getExpandsSelectedPaths is true it is exposed (made viewable).

**Parameters:**path - the TreePath specifying the node to select

### setSelectionPaths

public void **setSelectionPaths**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] paths)

Selects the nodes identified by the specified array of paths. If any component in any of the paths is hidden (under a collapsed node), and getExpandsSelectedPaths is true it is exposed (made viewable).

**Parameters:**paths - an array of TreePath objects that specifies the nodes to select

### setLeadSelectionPath

public void **setLeadSelectionPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) newPath)

Sets the path identifies as the lead. The lead may not be selected. The lead is not maintained by JTree, rather the UI will update it.

**Parameters:**newPath - the new lead path**Since:** 1.3

### setAnchorSelectionPath

public void **setAnchorSelectionPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) newPath)

Sets the path identified as the anchor. The anchor is not maintained by JTree, rather the UI will update it.

**Parameters:**newPath - the new anchor path**Since:** 1.3

### setSelectionRow

public void **setSelectionRow**(int row)

Selects the node at the specified row in the display.

**Parameters:**row - the row to select, where 0 is the first row in the display

### setSelectionRows

public void **setSelectionRows**(int[] rows)

Selects the nodes corresponding to each of the specified rows in the display. If a particular element of rows is < 0 or >= getRowCount, it will be ignored. If none of the elements in rows are valid rows, the selection will be cleared. That is it will be as if clearSelection was invoked.

**Parameters:**rows - an array of ints specifying the rows to select, where 0 indicates the first row in the display

### addSelectionPath

public void **addSelectionPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Adds the node identified by the specified TreePath to the current selection. If any component of the path isn't viewable, and getExpandsSelectedPaths is true it is made viewable.

Note that JTree does not allow duplicate nodes to exist as children under the same parent -- each sibling must be a unique object.

**Parameters:**path - the TreePath to add

### addSelectionPaths

public void **addSelectionPaths**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] paths)

Adds each path in the array of paths to the current selection. If any component of any of the paths isn't viewable and getExpandsSelectedPaths is true, it is made viewable.

Note that JTree does not allow duplicate nodes to exist as children under the same parent -- each sibling must be a unique object.

**Parameters:**paths - an array of TreePath objects that specifies the nodes to add

### addSelectionRow

public void **addSelectionRow**(int row)

Adds the path at the specified row to the current selection.

**Parameters:**row - an integer specifying the row of the node to add, where 0 is the first row in the display

### addSelectionRows

public void **addSelectionRows**(int[] rows)

Adds the paths at each of the specified rows to the current selection.

**Parameters:**rows - an array of ints specifying the rows to add, where 0 indicates the first row in the display

### getLastSelectedPathComponent

public [Object](http://docs.google.com/java/lang/Object.html) **getLastSelectedPathComponent**()

Returns the last path component in the first node of the current selection.

**Returns:**the last Object in the first selected node's TreePath, or null if nothing is selected**See Also:**[TreePath.getLastPathComponent()](http://docs.google.com/javax/swing/tree/TreePath.html#getLastPathComponent())

### getLeadSelectionPath

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getLeadSelectionPath**()

Returns the path identified as the lead.

**Returns:**path identified as the lead

### getAnchorSelectionPath

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getAnchorSelectionPath**()

Returns the path identified as the anchor.

**Returns:**path identified as the anchor**Since:** 1.3

### getSelectionPath

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getSelectionPath**()

Returns the path to the first selected node.

**Returns:**the TreePath for the first selected node, or null if nothing is currently selected

### getSelectionPaths

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] **getSelectionPaths**()

Returns the paths of all selected values.

**Returns:**an array of TreePath objects indicating the selected nodes, or null if nothing is currently selected

### getSelectionRows

public int[] **getSelectionRows**()

Returns all of the currently selected rows. This method is simply forwarded to the TreeSelectionModel. If nothing is selected null or an empty array will be returned, based on the TreeSelectionModel implementation.

**Returns:**an array of integers that identifies all currently selected rows where 0 is the first row in the display

### getSelectionCount

public int **getSelectionCount**()

Returns the number of nodes selected.

**Returns:**the number of nodes selected

### getMinSelectionRow

public int **getMinSelectionRow**()

Gets the first selected row.

**Returns:**an integer designating the first selected row, where 0 is the first row in the display

### getMaxSelectionRow

public int **getMaxSelectionRow**()

Returns the last selected row.

**Returns:**an integer designating the last selected row, where 0 is the first row in the display

### getLeadSelectionRow

public int **getLeadSelectionRow**()

Returns the row index corresponding to the lead path.

**Returns:**an integer giving the row index of the lead path, where 0 is the first row in the display; or -1 if leadPath is null

### isPathSelected

public boolean **isPathSelected**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns true if the item identified by the path is currently selected.

**Parameters:**path - a TreePath identifying a node **Returns:**true if the node is selected

### isRowSelected

public boolean **isRowSelected**(int row)

Returns true if the node identified by row is selected.

**Parameters:**row - an integer specifying a display row, where 0 is the first row in the display **Returns:**true if the node is selected

### getExpandedDescendants

public [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> **getExpandedDescendants**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) parent)

Returns an Enumeration of the descendants of the path parent that are currently expanded. If parent is not currently expanded, this will return null. If you expand/collapse nodes while iterating over the returned Enumeration this may not return all the expanded paths, or may return paths that are no longer expanded.

**Parameters:**parent - the path which is to be examined **Returns:**an Enumeration of the descendents of parent, or null if parent is not currently expanded

### hasBeenExpanded

public boolean **hasBeenExpanded**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns true if the node identified by the path has ever been expanded.

**Returns:**true if the path has ever been expanded

### isExpanded

public boolean **isExpanded**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns true if the node identified by the path is currently expanded,

**Parameters:**path - the TreePath specifying the node to check **Returns:**false if any of the nodes in the node's path are collapsed, true if all nodes in the path are expanded

### isExpanded

public boolean **isExpanded**(int row)

Returns true if the node at the specified display row is currently expanded.

**Parameters:**row - the row to check, where 0 is the first row in the display **Returns:**true if the node is currently expanded, otherwise false

### isCollapsed

public boolean **isCollapsed**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns true if the value identified by path is currently collapsed, this will return false if any of the values in path are currently not being displayed.

**Parameters:**path - the TreePath to check **Returns:**true if any of the nodes in the node's path are collapsed, false if all nodes in the path are expanded

### isCollapsed

public boolean **isCollapsed**(int row)

Returns true if the node at the specified display row is collapsed.

**Parameters:**row - the row to check, where 0 is the first row in the display **Returns:**true if the node is currently collapsed, otherwise false

### makeVisible

public void **makeVisible**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Ensures that the node identified by path is currently viewable.

**Parameters:**path - the TreePath to make visible

### isVisible

public boolean **isVisible**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns true if the value identified by path is currently viewable, which means it is either the root or all of its parents are expanded. Otherwise, this method returns false.

**Returns:**true if the node is viewable, otherwise false

### getPathBounds

public [Rectangle](http://docs.google.com/java/awt/Rectangle.html) **getPathBounds**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns the Rectangle that the specified node will be drawn into. Returns null if any component in the path is hidden (under a collapsed parent).

Note:

This method returns a valid rectangle, even if the specified node is not currently displayed.

**Parameters:**path - the TreePath identifying the node **Returns:**the Rectangle the node is drawn in, or null

### getRowBounds

public [Rectangle](http://docs.google.com/java/awt/Rectangle.html) **getRowBounds**(int row)

Returns the Rectangle that the node at the specified row is drawn in.

**Parameters:**row - the row to be drawn, where 0 is the first row in the display **Returns:**the Rectangle the node is drawn in

### scrollPathToVisible

public void **scrollPathToVisible**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Makes sure all the path components in path are expanded (except for the last path component) and scrolls so that the node identified by the path is displayed. Only works when this JTree is contained in a JScrollPane.

**Parameters:**path - the TreePath identifying the node to bring into view

### scrollRowToVisible

public void **scrollRowToVisible**(int row)

Scrolls the item identified by row until it is displayed. The minimum of amount of scrolling necessary to bring the row into view is performed. Only works when this JTree is contained in a JScrollPane.

**Parameters:**row - an integer specifying the row to scroll, where 0 is the first row in the display

### getPathForRow

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getPathForRow**(int row)

Returns the path for the specified row. If row is not visible, null is returned.

**Parameters:**row - an integer specifying a row **Returns:**the TreePath to the specified node, null if row < 0 or row > getRowCount()

### getRowForPath

public int **getRowForPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Returns the row that displays the node identified by the specified path.

**Parameters:**path - the TreePath identifying a node **Returns:**an integer specifying the display row, where 0 is the first row in the display, or -1 if any of the elements in path are hidden under a collapsed parent.

### expandPath

public void **expandPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Ensures that the node identified by the specified path is expanded and viewable. If the last item in the path is a leaf, this will have no effect.

**Parameters:**path - the TreePath identifying a node

### expandRow

public void **expandRow**(int row)

Ensures that the node in the specified row is expanded and viewable.

If row is < 0 or >= getRowCount this will have no effect.

**Parameters:**row - an integer specifying a display row, where 0 is the first row in the display

### collapsePath

public void **collapsePath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Ensures that the node identified by the specified path is collapsed and viewable.

**Parameters:**path - the TreePath identifying a node

### collapseRow

public void **collapseRow**(int row)

Ensures that the node in the specified row is collapsed.

If row is < 0 or >= getRowCount this will have no effect.

**Parameters:**row - an integer specifying a display row, where 0 is the first row in the display

### getPathForLocation

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getPathForLocation**(int x,  
 int y)

Returns the path for the node at the specified location.

**Parameters:**x - an integer giving the number of pixels horizontally from the left edge of the display area, minus any left marginy - an integer giving the number of pixels vertically from the top of the display area, minus any top margin **Returns:**the TreePath for the node at that location

### getRowForLocation

public int **getRowForLocation**(int x,  
 int y)

Returns the row for the specified location.

**Parameters:**x - an integer giving the number of pixels horizontally from the left edge of the display area, minus any left marginy - an integer giving the number of pixels vertically from the top of the display area, minus any top margin **Returns:**the row corresponding to the location, or -1 if the location is not within the bounds of a displayed cell**See Also:**[getClosestRowForLocation(int, int)](http://docs.google.com/javax/swing/JTree.html#getClosestRowForLocation(int,%20int))

### getClosestPathForLocation

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getClosestPathForLocation**(int x,  
 int y)

Returns the path to the node that is closest to x,y. If no nodes are currently viewable, or there is no model, returns null, otherwise it always returns a valid path. To test if the node is exactly at x, y, get the node's bounds and test x, y against that.

**Parameters:**x - an integer giving the number of pixels horizontally from the left edge of the display area, minus any left marginy - an integer giving the number of pixels vertically from the top of the display area, minus any top margin **Returns:**the TreePath for the node closest to that location, null if nothing is viewable or there is no model**See Also:**[getPathForLocation(int, int)](http://docs.google.com/javax/swing/JTree.html#getPathForLocation(int,%20int)), [getPathBounds(javax.swing.tree.TreePath)](http://docs.google.com/javax/swing/JTree.html#getPathBounds(javax.swing.tree.TreePath))

### getClosestRowForLocation

public int **getClosestRowForLocation**(int x,  
 int y)

Returns the row to the node that is closest to x,y. If no nodes are viewable or there is no model, returns -1. Otherwise, it always returns a valid row. To test if the returned object is exactly at x, y, get the bounds for the node at the returned row and test x, y against that.

**Parameters:**x - an integer giving the number of pixels horizontally from the left edge of the display area, minus any left marginy - an integer giving the number of pixels vertically from the top of the display area, minus any top margin **Returns:**the row closest to the location, -1 if nothing is viewable or there is no model**See Also:**[getRowForLocation(int, int)](http://docs.google.com/javax/swing/JTree.html#getRowForLocation(int,%20int)), [getRowBounds(int)](http://docs.google.com/javax/swing/JTree.html#getRowBounds(int))

### isEditing

public boolean **isEditing**()

Returns true if the tree is being edited. The item that is being edited can be obtained using getSelectionPath.

**Returns:**true if the user is currently editing a node**See Also:**[getSelectionPath()](http://docs.google.com/javax/swing/JTree.html#getSelectionPath())

### stopEditing

public boolean **stopEditing**()

Ends the current editing session. (The DefaultTreeCellEditor object saves any edits that are currently in progress on a cell. Other implementations may operate differently.) Has no effect if the tree isn't being edited.**Note:**

To make edit-saves automatic whenever the user changes their position in the tree, use [setInvokesStopCellEditing(boolean)](http://docs.google.com/javax/swing/JTree.html#setInvokesStopCellEditing(boolean)).

**Returns:**true if editing was in progress and is now stopped, false if editing was not in progress

### cancelEditing

public void **cancelEditing**()

Cancels the current editing session. Has no effect if the tree isn't being edited.

### startEditingAtPath

public void **startEditingAtPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Selects the node identified by the specified path and initiates editing. The edit-attempt fails if the CellEditor does not allow editing for the specified item.

**Parameters:**path - the TreePath identifying a node

### getEditingPath

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getEditingPath**()

Returns the path to the element that is currently being edited.

**Returns:**the TreePath for the node being edited

### setSelectionModel

public void **setSelectionModel**([TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html) selectionModel)

Sets the tree's selection model. When a null value is specified an empty selectionModel is used, which does not allow selections.

**Parameters:**selectionModel - the TreeSelectionModel to use, or null to disable selections**See Also:**[TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html)

### getSelectionModel

public [TreeSelectionModel](http://docs.google.com/javax/swing/tree/TreeSelectionModel.html) **getSelectionModel**()

Returns the model for selections. This should always return a non-null value. If you don't want to allow anything to be selected set the selection model to null, which forces an empty selection model to be used.

**See Also:**[setSelectionModel(javax.swing.tree.TreeSelectionModel)](http://docs.google.com/javax/swing/JTree.html#setSelectionModel(javax.swing.tree.TreeSelectionModel))

### getPathBetweenRows

protected [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] **getPathBetweenRows**(int index0,  
 int index1)

Returns JTreePath instances representing the path between index0 and index1 (including index1). Returns null if there is no tree.

**Parameters:**index0 - an integer specifying a display row, where 0 is the first row in the displayindex1 - an integer specifying a second display row **Returns:**an array of TreePath objects, one for each node between index0 and index1, inclusive; or null if there is no tree

### setSelectionInterval

public void **setSelectionInterval**(int index0,  
 int index1)

Selects the nodes between index0 and index1, inclusive.

**Parameters:**index0 - an integer specifying a display row, where 0 is the first row in the displayindex1 - an integer specifying a second display row

### addSelectionInterval

public void **addSelectionInterval**(int index0,  
 int index1)

Adds the paths between index0 and index1, inclusive, to the selection.

**Parameters:**index0 - an integer specifying a display row, where 0 is the first row in the displayindex1 - an integer specifying a second display row

### removeSelectionInterval

public void **removeSelectionInterval**(int index0,  
 int index1)

Removes the nodes between index0 and index1, inclusive, from the selection.

**Parameters:**index0 - an integer specifying a display row, where 0 is the first row in the displayindex1 - an integer specifying a second display row

### removeSelectionPath

public void **removeSelectionPath**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Removes the node identified by the specified path from the current selection.

**Parameters:**path - the TreePath identifying a node

### removeSelectionPaths

public void **removeSelectionPaths**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)[] paths)

Removes the nodes identified by the specified paths from the current selection.

**Parameters:**paths - an array of TreePath objects that specifies the nodes to remove

### removeSelectionRow

public void **removeSelectionRow**(int row)

Removes the row at the index row from the current selection.

**Parameters:**row - the row to remove

### removeSelectionRows

public void **removeSelectionRows**(int[] rows)

Removes the rows that are selected at each of the specified rows.

**Parameters:**rows - an array of ints specifying display rows, where 0 is the first row in the display

### clearSelection

public void **clearSelection**()

Clears the selection.

### isSelectionEmpty

public boolean **isSelectionEmpty**()

Returns true if the selection is currently empty.

**Returns:**true if the selection is currently empty

### addTreeExpansionListener

public void **addTreeExpansionListener**([TreeExpansionListener](http://docs.google.com/javax/swing/event/TreeExpansionListener.html) tel)

Adds a listener for TreeExpansion events.

**Parameters:**tel - a TreeExpansionListener that will be notified when a tree node is expanded or collapsed (a "negative expansion")

### removeTreeExpansionListener

public void **removeTreeExpansionListener**([TreeExpansionListener](http://docs.google.com/javax/swing/event/TreeExpansionListener.html) tel)

Removes a listener for TreeExpansion events.

**Parameters:**tel - the TreeExpansionListener to remove

### getTreeExpansionListeners

public [TreeExpansionListener](http://docs.google.com/javax/swing/event/TreeExpansionListener.html)[] **getTreeExpansionListeners**()

Returns an array of all the TreeExpansionListeners added to this JTree with addTreeExpansionListener().

**Returns:**all of the TreeExpansionListeners added or an empty array if no listeners have been added**Since:** 1.4

### addTreeWillExpandListener

public void **addTreeWillExpandListener**([TreeWillExpandListener](http://docs.google.com/javax/swing/event/TreeWillExpandListener.html) tel)

Adds a listener for TreeWillExpand events.

**Parameters:**tel - a TreeWillExpandListener that will be notified when a tree node will be expanded or collapsed (a "negative expansion")

### removeTreeWillExpandListener

public void **removeTreeWillExpandListener**([TreeWillExpandListener](http://docs.google.com/javax/swing/event/TreeWillExpandListener.html) tel)

Removes a listener for TreeWillExpand events.

**Parameters:**tel - the TreeWillExpandListener to remove

### getTreeWillExpandListeners

public [TreeWillExpandListener](http://docs.google.com/javax/swing/event/TreeWillExpandListener.html)[] **getTreeWillExpandListeners**()

Returns an array of all the TreeWillExpandListeners added to this JTree with addTreeWillExpandListener().

**Returns:**all of the TreeWillExpandListeners added or an empty array if no listeners have been added**Since:** 1.4

### fireTreeExpanded

public void **fireTreeExpanded**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Notifies all listeners that have registered interest for notification on this event type. The event instance is lazily created using the path parameter.

**Parameters:**path - the TreePath indicating the node that was expanded**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTreeCollapsed

public void **fireTreeCollapsed**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)

Notifies all listeners that have registered interest for notification on this event type. The event instance is lazily created using the path parameter.

**Parameters:**path - the TreePath indicating the node that was collapsed**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTreeWillExpand

public void **fireTreeWillExpand**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)  
 throws [ExpandVetoException](http://docs.google.com/javax/swing/tree/ExpandVetoException.html)

Notifies all listeners that have registered interest for notification on this event type. The event instance is lazily created using the path parameter.

**Parameters:**path - the TreePath indicating the node that was expanded **Throws:** [ExpandVetoException](http://docs.google.com/javax/swing/tree/ExpandVetoException.html)**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTreeWillCollapse

public void **fireTreeWillCollapse**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)  
 throws [ExpandVetoException](http://docs.google.com/javax/swing/tree/ExpandVetoException.html)

Notifies all listeners that have registered interest for notification on this event type. The event instance is lazily created using the path parameter.

**Parameters:**path - the TreePath indicating the node that was expanded **Throws:** [ExpandVetoException](http://docs.google.com/javax/swing/tree/ExpandVetoException.html)**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### addTreeSelectionListener

public void **addTreeSelectionListener**([TreeSelectionListener](http://docs.google.com/javax/swing/event/TreeSelectionListener.html) tsl)

Adds a listener for TreeSelection events.

**Parameters:**tsl - the TreeSelectionListener that will be notified when a node is selected or deselected (a "negative selection")

### removeTreeSelectionListener

public void **removeTreeSelectionListener**([TreeSelectionListener](http://docs.google.com/javax/swing/event/TreeSelectionListener.html) tsl)

Removes a TreeSelection listener.

**Parameters:**tsl - the TreeSelectionListener to remove

### getTreeSelectionListeners

public [TreeSelectionListener](http://docs.google.com/javax/swing/event/TreeSelectionListener.html)[] **getTreeSelectionListeners**()

Returns an array of all the TreeSelectionListeners added to this JTree with addTreeSelectionListener().

**Returns:**all of the TreeSelectionListeners added or an empty array if no listeners have been added**Since:** 1.4

### fireValueChanged

protected void **fireValueChanged**([TreeSelectionEvent](http://docs.google.com/javax/swing/event/TreeSelectionEvent.html) e)

Notifies all listeners that have registered interest for notification on this event type.

**Parameters:**e - the TreeSelectionEvent to be fired; generated by the TreeSelectionModel when a node is selected or deselected**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### treeDidChange

public void **treeDidChange**()

Sent when the tree has changed enough that we need to resize the bounds, but not enough that we need to remove the expanded node set (e.g nodes were expanded or collapsed, or nodes were inserted into the tree). You should never have to invoke this, the UI will invoke this as it needs to.

### setVisibleRowCount

public void **setVisibleRowCount**(int newCount)

Sets the number of rows that are to be displayed. This will only work if the tree is contained in a JScrollPane, and will adjust the preferred size and size of that scrollpane.

**Parameters:**newCount - the number of rows to display

### getVisibleRowCount

public int **getVisibleRowCount**()

Returns the number of rows that are displayed in the display area.

**Returns:**the number of rows displayed

### getNextMatch

public [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) **getNextMatch**([String](http://docs.google.com/java/lang/String.html) prefix,  
 int startingRow,  
 [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) bias)

Returns the TreePath to the next tree element that begins with a prefix. To handle the conversion of a TreePath into a String, convertValueToText is used.

**Parameters:**prefix - the string to test for a matchstartingRow - the row for starting the searchbias - the search direction, either Position.Bias.Forward or Position.Bias.Backward. **Returns:**the TreePath of the next tree element that starts with the prefix; otherwise null **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if prefix is null or startingRow is out of bounds**Since:** 1.4

### getPreferredScrollableViewportSize

public [Dimension](http://docs.google.com/java/awt/Dimension.html) **getPreferredScrollableViewportSize**()

Returns the preferred display size of a JTree. The height is determined from getVisibleRowCount and the width is the current preferred width.

**Specified by:**[getPreferredScrollableViewportSize](http://docs.google.com/javax/swing/Scrollable.html#getPreferredScrollableViewportSize()) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html) **Returns:**a Dimension object containing the preferred size**See Also:**[JComponent.getPreferredSize()](http://docs.google.com/javax/swing/JComponent.html#getPreferredSize())

### getScrollableUnitIncrement

public int **getScrollableUnitIncrement**([Rectangle](http://docs.google.com/java/awt/Rectangle.html) visibleRect,  
 int orientation,  
 int direction)

Returns the amount to increment when scrolling. The amount is the height of the first displayed row that isn't completely in view or, if it is totally displayed, the height of the next row in the scrolling direction.

**Specified by:**[getScrollableUnitIncrement](http://docs.google.com/javax/swing/Scrollable.html#getScrollableUnitIncrement(java.awt.Rectangle,%20int,%20int)) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html) **Parameters:**visibleRect - the view area visible within the viewportorientation - either SwingConstants.VERTICAL or SwingConstants.HORIZONTALdirection - less than zero to scroll up/left, greater than zero for down/right **Returns:**the "unit" increment for scrolling in the specified direction**See Also:**[JScrollBar.setUnitIncrement(int)](http://docs.google.com/javax/swing/JScrollBar.html#setUnitIncrement(int))

### getScrollableBlockIncrement

public int **getScrollableBlockIncrement**([Rectangle](http://docs.google.com/java/awt/Rectangle.html) visibleRect,  
 int orientation,  
 int direction)

Returns the amount for a block increment, which is the height or width of visibleRect, based on orientation.

**Specified by:**[getScrollableBlockIncrement](http://docs.google.com/javax/swing/Scrollable.html#getScrollableBlockIncrement(java.awt.Rectangle,%20int,%20int)) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html) **Parameters:**visibleRect - the view area visible within the viewportorientation - either SwingConstants.VERTICAL or SwingConstants.HORIZONTALdirection - less than zero to scroll up/left, greater than zero for down/right. **Returns:**the "block" increment for scrolling in the specified direction**See Also:**[JScrollBar.setBlockIncrement(int)](http://docs.google.com/javax/swing/JScrollBar.html#setBlockIncrement(int))

### getScrollableTracksViewportWidth

public boolean **getScrollableTracksViewportWidth**()

Returns false to indicate that the width of the viewport does not determine the width of the table, unless the preferred width of the tree is smaller than the viewports width. In other words: ensure that the tree is never smaller than its viewport.

**Specified by:**[getScrollableTracksViewportWidth](http://docs.google.com/javax/swing/Scrollable.html#getScrollableTracksViewportWidth()) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html) **Returns:**false**See Also:**[Scrollable.getScrollableTracksViewportWidth()](http://docs.google.com/javax/swing/Scrollable.html#getScrollableTracksViewportWidth())

### getScrollableTracksViewportHeight

public boolean **getScrollableTracksViewportHeight**()

Returns false to indicate that the height of the viewport does not determine the height of the table, unless the preferred height of the tree is smaller than the viewports height. In other words: ensure that the tree is never smaller than its viewport.

**Specified by:**[getScrollableTracksViewportHeight](http://docs.google.com/javax/swing/Scrollable.html#getScrollableTracksViewportHeight()) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html) **Returns:**false**See Also:**[Scrollable.getScrollableTracksViewportHeight()](http://docs.google.com/javax/swing/Scrollable.html#getScrollableTracksViewportHeight())

### setExpandedState

protected void **setExpandedState**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path,  
 boolean state)

Sets the expanded state of this JTree. If state is true, all parents of path and path are marked as expanded. If state is false, all parents of path are marked EXPANDED, but path itself is marked collapsed.

This will fail if a TreeWillExpandListener vetos it.

### getDescendantToggledPaths

protected [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> **getDescendantToggledPaths**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) parent)

Returns an Enumeration of TreePaths that have been expanded that are descendants of parent.

### removeDescendantToggledPaths

protected void **removeDescendantToggledPaths**([Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> toRemove)

Removes any descendants of the TreePaths in toRemove that have been expanded.

### clearToggledPaths

protected void **clearToggledPaths**()

Clears the cache of toggled tree paths. This does NOT send out any TreeExpansionListener events.

### createTreeModelListener

protected [TreeModelListener](http://docs.google.com/javax/swing/event/TreeModelListener.html) **createTreeModelListener**()

Creates and returns an instance of TreeModelHandler. The returned object is responsible for updating the expanded state when the TreeModel changes.

For more information on what expanded state means, see the [JTree description](#3znysh7) above.

### removeDescendantSelectedPaths

protected boolean **removeDescendantSelectedPaths**([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path,  
 boolean includePath)

Removes any paths in the selection that are descendants of path. If includePath is true and path is selected, it will be removed from the selection.

**Returns:**true if a descendant was selected**Since:** 1.3

### paramString

protected [String](http://docs.google.com/java/lang/String.html) **paramString**()

Returns a string representation of this JTree. This method is intended to be used only for debugging purposes, and the content and format of the returned string may vary between implementations. The returned string may be empty but may not be null.

**Overrides:**[paramString](http://docs.google.com/javax/swing/JComponent.html#paramString()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Returns:**a string representation of this JTree.

### getAccessibleContext

public [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) **getAccessibleContext**()

Gets the AccessibleContext associated with this JTree. For JTrees, the AccessibleContext takes the form of an AccessibleJTree. A new AccessibleJTree instance is created if necessary.

**Specified by:**[getAccessibleContext](http://docs.google.com/javax/accessibility/Accessible.html#getAccessibleContext()) in interface [Accessible](http://docs.google.com/javax/accessibility/Accessible.html)**Overrides:**[getAccessibleContext](http://docs.google.com/javax/swing/JComponent.html#getAccessibleContext()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Returns:**an AccessibleJTree that serves as the AccessibleContext of this JTree

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JTree.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/swing/JToolTip.AccessibleJToolTip.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JTree.AccessibleJTree.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JTree.html)    [**NO FRAMES**](http://docs.google.com/JTree.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#4d34og8) | [CONSTR](#26in1rg) | [METHOD](#lnxbz9) | DETAIL: [FIELD](#z337ya) | [CONSTR](#46r0co2) | [METHOD](#3ygebqi) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).