| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/VariableHeightLayoutCache.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/tree/TreeSelectionModel.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/swing/tree/VariableHeightLayoutCache.html)    [**NO FRAMES**](http://docs.google.com/VariableHeightLayoutCache.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#3dy6vkm) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#lnxbz9) |

## **javax.swing.tree**

Class VariableHeightLayoutCache

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [javax.swing.tree.AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html)  
 **javax.swing.tree.VariableHeightLayoutCache**

**All Implemented Interfaces:** [RowMapper](http://docs.google.com/javax/swing/tree/RowMapper.html)

public class **VariableHeightLayoutCache**extends [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html)

NOTE: This will become more open in a future release.

**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** | |
| --- | --- |

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

| **Field Summary** | |
| --- | --- |

| **Fields inherited from class javax.swing.tree.**[**AbstractLayoutCache**](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) |
| --- |
| [nodeDimensions](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#nodeDimensions), [rootVisible](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#rootVisible), [rowHeight](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#rowHeight), [treeModel](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#treeModel), [treeSelectionModel](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#treeSelectionModel) |

| **Constructor Summary** | |
| --- | --- |
| [**VariableHeightLayoutCache**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#VariableHeightLayoutCache())() |

| **Method Summary** | |
| --- | --- |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getBounds**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getBounds(javax.swing.tree.TreePath,%20java.awt.Rectangle))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path, [Rectangle](http://docs.google.com/java/awt/Rectangle.html) placeIn)            Returns the Rectangle enclosing the label portion into which the item identified by path will be drawn. |
| boolean | [**getExpandedState**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getExpandedState(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns true if the path is expanded, and visible. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getPathClosestTo**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getPathClosestTo(int,%20int))(int x, int y)            Returns the path to the node that is closest to x,y. |
| [TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) | [**getPathForRow**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getPathForRow(int))(int row)            Returns the path for row. |
| int | [**getPreferredHeight**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getPreferredHeight())()            Returns the preferred height. |
| int | [**getPreferredWidth**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getPreferredWidth(java.awt.Rectangle))([Rectangle](http://docs.google.com/java/awt/Rectangle.html) bounds)            Returns the preferred width and height for the region in visibleRegion. |
| int | [**getRowCount**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getRowCount())()            Returns the number of visible rows. |
| int | [**getRowForPath**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getRowForPath(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns the row where the last item identified in path is visible. |
| int | [**getVisibleChildCount**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getVisibleChildCount(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns the number of visible children for path. |
| [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[TreePath](http://docs.google.com/javax/swing/tree/TreePath.html)> | [**getVisiblePathsFrom**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#getVisiblePathsFrom(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Returns an Enumerator that increments over the visible paths starting at the passed in location. |
| void | [**invalidatePathBounds**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#invalidatePathBounds(javax.swing.tree.TreePath))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path)            Instructs the LayoutCache that the bounds for path are invalid, and need to be updated. |
| void | [**invalidateSizes**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#invalidateSizes())()            Informs the TreeState that it needs to recalculate all the sizes it is referencing. |
| boolean | [**isExpanded**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#isExpanded(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 expanded. |
| void | [**setExpandedState**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#setExpandedState(javax.swing.tree.TreePath,%20boolean))([TreePath](http://docs.google.com/javax/swing/tree/TreePath.html) path, boolean isExpanded)            Marks the path path expanded state to isExpanded. |
| void | [**setModel**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.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 | [**setNodeDimensions**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#setNodeDimensions(javax.swing.tree.AbstractLayoutCache.NodeDimensions))([AbstractLayoutCache.NodeDimensions](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.NodeDimensions.html) nd)            Sets the renderer that is responsible for drawing nodes in the tree. |
| void | [**setRootVisible**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.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/tree/VariableHeightLayoutCache.html#setRowHeight(int))(int rowHeight)            Sets the height of each cell. |
| void | [**treeNodesChanged**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#treeNodesChanged(javax.swing.event.TreeModelEvent))([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)            Invoked after a node (or a set of siblings) has changed in some way. |
| void | [**treeNodesInserted**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#treeNodesInserted(javax.swing.event.TreeModelEvent))([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)            Invoked after nodes have been inserted into the tree. |
| void | [**treeNodesRemoved**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#treeNodesRemoved(javax.swing.event.TreeModelEvent))([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)            Invoked after nodes have been removed from the tree. |
| void | [**treeStructureChanged**](http://docs.google.com/javax/swing/tree/VariableHeightLayoutCache.html#treeStructureChanged(javax.swing.event.TreeModelEvent))([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)            Invoked after the tree has drastically changed structure from a given node down. |

| **Methods inherited from class javax.swing.tree.**[**AbstractLayoutCache**](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) |
| --- |
| [getModel](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getModel()), [getNodeDimensions](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getNodeDimensions()), [getNodeDimensions](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getNodeDimensions(java.lang.Object,%20int,%20int,%20boolean,%20java.awt.Rectangle)), [getRowHeight](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getRowHeight()), [getRowsForPaths](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getRowsForPaths(javax.swing.tree.TreePath%5B%5D)), [getSelectionModel](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getSelectionModel()), [isFixedRowHeight](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#isFixedRowHeight()), [isRootVisible](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#isRootVisible()), [setSelectionModel](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#setSelectionModel(javax.swing.tree.TreeSelectionModel)) |

| **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()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [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)) |

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

### VariableHeightLayoutCache

public **VariableHeightLayoutCache**()

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

### setModel

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

Sets the TreeModel that will provide the data.

**Overrides:**[setModel](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#setModel(javax.swing.tree.TreeModel)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**newModel - the TreeModel that is to provide the data

### setRootVisible

public void **setRootVisible**(boolean rootVisible)

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

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

### setRowHeight

public void **setRowHeight**(int rowHeight)

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

**Overrides:**[setRowHeight](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#setRowHeight(int)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**rowHeight - the height of each cell, in pixels

### setNodeDimensions

public void **setNodeDimensions**([AbstractLayoutCache.NodeDimensions](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.NodeDimensions.html) nd)

Sets the renderer that is responsible for drawing nodes in the tree.

**Overrides:**[setNodeDimensions](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#setNodeDimensions(javax.swing.tree.AbstractLayoutCache.NodeDimensions)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**nd - the renderer

### setExpandedState

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

Marks the path path expanded state to isExpanded.

**Specified by:**[setExpandedState](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#setExpandedState(javax.swing.tree.TreePath,%20boolean)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the TreePath of interestisExpanded - true if the path should be expanded, otherwise false

### getExpandedState

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

Returns true if the path is expanded, and visible.

**Specified by:**[getExpandedState](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getExpandedState(javax.swing.tree.TreePath)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the path being queried **Returns:**true if the path is expanded and visible, otherwise false

### getBounds

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

Returns the Rectangle enclosing the label portion into which the item identified by path will be drawn.

**Specified by:**[getBounds](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getBounds(javax.swing.tree.TreePath,%20java.awt.Rectangle)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the path to be drawnplaceIn - the bounds of the enclosing rectangle **Returns:**the bounds of the enclosing rectangle or null if the node could not be ascertained

### getPathForRow

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

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

**Specified by:**[getPathForRow](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getPathForRow(int)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**row - the location of interest **Returns:**the path for row, or null if row is not visible

### getRowForPath

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

Returns the row where the last item identified in path is visible. Will return -1 if any of the elements in path are not currently visible.

**Specified by:**[getRowForPath](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getRowForPath(javax.swing.tree.TreePath)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the TreePath of interest **Returns:**the row where the last item in path is visible

### getRowCount

public int **getRowCount**()

Returns the number of visible rows.

**Specified by:**[getRowCount](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getRowCount()) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Returns:**the number of visible rows

### invalidatePathBounds

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

Instructs the LayoutCache that the bounds for path are invalid, and need to be updated.

**Specified by:**[invalidatePathBounds](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#invalidatePathBounds(javax.swing.tree.TreePath)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the TreePath which is now invalid

### getPreferredHeight

public int **getPreferredHeight**()

Returns the preferred height.

**Overrides:**[getPreferredHeight](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getPreferredHeight()) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Returns:**the preferred height

### getPreferredWidth

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

Returns the preferred width and height for the region in visibleRegion.

**Overrides:**[getPreferredWidth](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getPreferredWidth(java.awt.Rectangle)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**bounds - the region being queried **Returns:**the preferred width for the passed in region

### getPathClosestTo

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

Returns the path to the node that is closest to x,y. If there is nothing currently visible this will return null, otherwise it will always return a valid path. If you need to test if the returned object is exactly at x, y you should get the bounds for the returned path and test x, y against that.

**Specified by:**[getPathClosestTo](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getPathClosestTo(int,%20int)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**x - the x-coordinatey - the y-coordinate **Returns:**the path to the node that is closest to x, y

### getVisiblePathsFrom

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

Returns an Enumerator that increments over the visible paths starting at the passed in location. The ordering of the enumeration is based on how the paths are displayed.

**Specified by:**[getVisiblePathsFrom](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getVisiblePathsFrom(javax.swing.tree.TreePath)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the location in the TreePath to start **Returns:**an Enumerator that increments over the visible paths

### getVisibleChildCount

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

Returns the number of visible children for path.

**Specified by:**[getVisibleChildCount](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#getVisibleChildCount(javax.swing.tree.TreePath)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**path - the path being queried **Returns:**the number of visible children for path

### invalidateSizes

public void **invalidateSizes**()

Informs the TreeState that it needs to recalculate all the sizes it is referencing.

**Specified by:**[invalidateSizes](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#invalidateSizes()) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html)

### isExpanded

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

Returns true if the value identified by path is currently expanded.

**Specified by:**[isExpanded](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#isExpanded(javax.swing.tree.TreePath)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Returns:**true if the value identified by path is currently expanded

### treeNodesChanged

public void **treeNodesChanged**([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)

Invoked after a node (or a set of siblings) has changed in some way. The node(s) have not changed locations in the tree or altered their children arrays, but other attributes have changed and may affect presentation. Example: the name of a file has changed, but it is in the same location in the file system.

e.path returns the path the parent of the changed node(s).

e.childIndices returns the index(es) of the changed node(s).

**Specified by:**[treeNodesChanged](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#treeNodesChanged(javax.swing.event.TreeModelEvent)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**e - the TreeModelEvent of interest

### treeNodesInserted

public void **treeNodesInserted**([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)

Invoked after nodes have been inserted into the tree.

e.path returns the parent of the new nodes.

e.childIndices returns the indices of the new nodes in ascending order.

**Specified by:**[treeNodesInserted](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#treeNodesInserted(javax.swing.event.TreeModelEvent)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**e - the TreeModelEvent of interest

### treeNodesRemoved

public void **treeNodesRemoved**([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)

Invoked after nodes have been removed from the tree. Note that if a subtree is removed from the tree, this method may only be invoked once for the root of the removed subtree, not once for each individual set of siblings removed.

e.path returns the former parent of the deleted nodes.

e.childIndices returns the indices the nodes had before they were deleted in ascending order.

**Specified by:**[treeNodesRemoved](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#treeNodesRemoved(javax.swing.event.TreeModelEvent)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**e - the TreeModelEvent of interest

### treeStructureChanged

public void **treeStructureChanged**([TreeModelEvent](http://docs.google.com/javax/swing/event/TreeModelEvent.html) e)

Invoked after the tree has drastically changed structure from a given node down. If the path returned by e.getPath is of length one and the first element does not identify the current root node the first element should become the new root of the tree.

e.path holds the path to the node.

e.childIndices returns null.

**Specified by:**[treeStructureChanged](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html#treeStructureChanged(javax.swing.event.TreeModelEvent)) in class [AbstractLayoutCache](http://docs.google.com/javax/swing/tree/AbstractLayoutCache.html) **Parameters:**e - the TreeModelEvent of interest

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/VariableHeightLayoutCache.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/tree/TreeSelectionModel.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?javax/swing/tree/VariableHeightLayoutCache.html)    [**NO FRAMES**](http://docs.google.com/VariableHeightLayoutCache.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#3dy6vkm) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#lnxbz9) |

[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).