| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JTabbedPane.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/JSplitPane.AccessibleJSplitPane.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JTabbedPane.AccessibleJTabbedPane.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JTabbedPane.html)    [**NO FRAMES**](http://docs.google.com/JTabbedPane.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#lnxbz9) | DETAIL: [FIELD](#z337ya) | [CONSTR](#2bn6wsx) | [METHOD](#49x2ik5) |

## **javax.swing**

Class JTabbedPane

[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.JTabbedPane**

**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), [SwingConstants](http://docs.google.com/javax/swing/SwingConstants.html)

public class **JTabbedPane**extends [JComponent](http://docs.google.com/javax/swing/JComponent.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html), [Accessible](http://docs.google.com/javax/accessibility/Accessible.html), [SwingConstants](http://docs.google.com/javax/swing/SwingConstants.html)

A component that lets the user switch between a group of components by clicking on a tab with a given title and/or icon. For examples and information on using tabbed panes see [How to Use Tabbed Panes](http://java.sun.com/docs/books/tutorial/uiswing/components/tabbedpane.html), a section in *The Java Tutorial*.

Tabs/components are added to a TabbedPane object by using the addTab and insertTab methods. A tab is represented by an index corresponding to the position it was added in, where the first tab has an index equal to 0 and the last tab has an index equal to the tab count minus 1.

The TabbedPane uses a SingleSelectionModel to represent the set of tab indices and the currently selected index. If the tab count is greater than 0, then there will always be a selected index, which by default will be initialized to the first tab. If the tab count is 0, then the selected index will be -1.

The tab title can be rendered by a Component. For example, the following produce similar results:

// In this case the look and feel renders the title for the tab.  
 tabbedPane.addTab("Tab", myComponent);  
 // In this case the custom component is responsible for rendering the  
 // title of the tab.  
 tabbedPane.addTab(null, myComponent);  
 tabbedPane.setTabComponentAt(0, new JLabel("Tab"));

The latter is typically used when you want a more complex user interaction that requires custom components on the tab. For example, you could provide a custom component that animates or one that has widgets for closing the tab.

If you specify a component for a tab, the JTabbedPane will not render any text or icon you have specified for the tab.

**Note:** Do not use setVisible directly on a tab component to make it visible, use setSelectedComponent or setSelectedIndex methods instead.

**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).

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

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**JTabbedPane.AccessibleJTabbedPane**](http://docs.google.com/javax/swing/JTabbedPane.AccessibleJTabbedPane.html)            This class implements accessibility support for the JTabbedPane class. |
| protected  class | [**JTabbedPane.ModelListener**](http://docs.google.com/javax/swing/JTabbedPane.ModelListener.html)            We pass ModelChanged events along to the listeners with the tabbedpane (instead of the model itself) as the event source. |

| **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** | |
| --- | --- |
| protected  [ChangeEvent](http://docs.google.com/javax/swing/event/ChangeEvent.html) | [**changeEvent**](http://docs.google.com/javax/swing/JTabbedPane.html#changeEvent)            Only one ChangeEvent is needed per TabPane instance since the event's only (read-only) state is the source property. |
| protected  [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) | [**changeListener**](http://docs.google.com/javax/swing/JTabbedPane.html#changeListener)            The changeListener is the listener we add to the model. |
| protected  [SingleSelectionModel](http://docs.google.com/javax/swing/SingleSelectionModel.html) | [**model**](http://docs.google.com/javax/swing/JTabbedPane.html#model)            The default selection model |
| static int | [**SCROLL\_TAB\_LAYOUT**](http://docs.google.com/javax/swing/JTabbedPane.html#SCROLL_TAB_LAYOUT)            Tab layout policy for providing a subset of available tabs when all the tabs will not fit within a single run. |
| protected  int | [**tabPlacement**](http://docs.google.com/javax/swing/JTabbedPane.html#tabPlacement)            Where the tabs are placed. |
| static int | [**WRAP\_TAB\_LAYOUT**](http://docs.google.com/javax/swing/JTabbedPane.html#WRAP_TAB_LAYOUT)            The tab layout policy for wrapping tabs in multiple runs when all tabs will not fit within a single run. |

| **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 javax.swing.**[**SwingConstants**](http://docs.google.com/javax/swing/SwingConstants.html) |
| --- |
| [BOTTOM](http://docs.google.com/javax/swing/SwingConstants.html#BOTTOM), [CENTER](http://docs.google.com/javax/swing/SwingConstants.html#CENTER), [EAST](http://docs.google.com/javax/swing/SwingConstants.html#EAST), [HORIZONTAL](http://docs.google.com/javax/swing/SwingConstants.html#HORIZONTAL), [LEADING](http://docs.google.com/javax/swing/SwingConstants.html#LEADING), [LEFT](http://docs.google.com/javax/swing/SwingConstants.html#LEFT), [NEXT](http://docs.google.com/javax/swing/SwingConstants.html#NEXT), [NORTH](http://docs.google.com/javax/swing/SwingConstants.html#NORTH), [NORTH\_EAST](http://docs.google.com/javax/swing/SwingConstants.html#NORTH_EAST), [NORTH\_WEST](http://docs.google.com/javax/swing/SwingConstants.html#NORTH_WEST), [PREVIOUS](http://docs.google.com/javax/swing/SwingConstants.html#PREVIOUS), [RIGHT](http://docs.google.com/javax/swing/SwingConstants.html#RIGHT), [SOUTH](http://docs.google.com/javax/swing/SwingConstants.html#SOUTH), [SOUTH\_EAST](http://docs.google.com/javax/swing/SwingConstants.html#SOUTH_EAST), [SOUTH\_WEST](http://docs.google.com/javax/swing/SwingConstants.html#SOUTH_WEST), [TOP](http://docs.google.com/javax/swing/SwingConstants.html#TOP), [TRAILING](http://docs.google.com/javax/swing/SwingConstants.html#TRAILING), [VERTICAL](http://docs.google.com/javax/swing/SwingConstants.html#VERTICAL), [WEST](http://docs.google.com/javax/swing/SwingConstants.html#WEST) |

| **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** | |
| --- | --- |
| [**JTabbedPane**](http://docs.google.com/javax/swing/JTabbedPane.html#JTabbedPane())()            Creates an empty TabbedPane with a default tab placement of JTabbedPane.TOP. |
| [**JTabbedPane**](http://docs.google.com/javax/swing/JTabbedPane.html#JTabbedPane(int))(int tabPlacement)            Creates an empty TabbedPane with the specified tab placement of either: JTabbedPane.TOP, JTabbedPane.BOTTOM, JTabbedPane.LEFT, or JTabbedPane.RIGHT. |
| [**JTabbedPane**](http://docs.google.com/javax/swing/JTabbedPane.html#JTabbedPane(int,%20int))(int tabPlacement, int tabLayoutPolicy)            Creates an empty TabbedPane with the specified tab placement and tab layout policy. |

| **Method Summary** | |
| --- | --- |
| [Component](http://docs.google.com/java/awt/Component.html) | [**add**](http://docs.google.com/javax/swing/JTabbedPane.html#add(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) component)            Adds a component with a tab title defaulting to the name of the component which is the result of calling component.getName. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**add**](http://docs.google.com/javax/swing/JTabbedPane.html#add(java.awt.Component,%20int))([Component](http://docs.google.com/java/awt/Component.html) component, int index)            Adds a component at the specified tab index with a tab title defaulting to the name of the component. |
| void | [**add**](http://docs.google.com/javax/swing/JTabbedPane.html#add(java.awt.Component,%20java.lang.Object))([Component](http://docs.google.com/java/awt/Component.html) component, [Object](http://docs.google.com/java/lang/Object.html) constraints)            Adds a component to the tabbed pane. |
| void | [**add**](http://docs.google.com/javax/swing/JTabbedPane.html#add(java.awt.Component,%20java.lang.Object,%20int))([Component](http://docs.google.com/java/awt/Component.html) component, [Object](http://docs.google.com/java/lang/Object.html) constraints, int index)            Adds a component at the specified tab index. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**add**](http://docs.google.com/javax/swing/JTabbedPane.html#add(java.lang.String,%20java.awt.Component))([String](http://docs.google.com/java/lang/String.html) title, [Component](http://docs.google.com/java/awt/Component.html) component)            Adds a component with the specified tab title. |
| void | [**addChangeListener**](http://docs.google.com/javax/swing/JTabbedPane.html#addChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)            Adds a ChangeListener to this tabbedpane. |
| void | [**addTab**](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20java.awt.Component))([String](http://docs.google.com/java/lang/String.html) title, [Component](http://docs.google.com/java/awt/Component.html) component)            Adds a component represented by a title and no icon. |
| void | [**addTab**](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component))([String](http://docs.google.com/java/lang/String.html) title, [Icon](http://docs.google.com/javax/swing/Icon.html) icon, [Component](http://docs.google.com/java/awt/Component.html) component)            Adds a component represented by a title and/or icon, either of which can be null. |
| void | [**addTab**](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) title, [Icon](http://docs.google.com/javax/swing/Icon.html) icon, [Component](http://docs.google.com/java/awt/Component.html) component, [String](http://docs.google.com/java/lang/String.html) tip)            Adds a component and tip represented by a title and/or icon, either of which can be null. |
| protected  [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) | [**createChangeListener**](http://docs.google.com/javax/swing/JTabbedPane.html#createChangeListener())()            Subclasses that want to handle ChangeEvents differently can override this to return a subclass of ModelListener or another ChangeListener implementation. |
| protected  void | [**fireStateChanged**](http://docs.google.com/javax/swing/JTabbedPane.html#fireStateChanged())()            Sends a ChangeEvent, with this JTabbedPane as the source, to each registered listener. |
| [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) | [**getAccessibleContext**](http://docs.google.com/javax/swing/JTabbedPane.html#getAccessibleContext())()            Gets the AccessibleContext associated with this JTabbedPane. |
| [Color](http://docs.google.com/java/awt/Color.html) | [**getBackgroundAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getBackgroundAt(int))(int index)            Returns the tab background color at index. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getBoundsAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getBoundsAt(int))(int index)            Returns the tab bounds at index. |
| [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html)[] | [**getChangeListeners**](http://docs.google.com/javax/swing/JTabbedPane.html#getChangeListeners())()            Returns an array of all the ChangeListeners added to this JTabbedPane with addChangeListener. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getComponentAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getComponentAt(int))(int index)            Returns the component at index. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getDisabledIconAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getDisabledIconAt(int))(int index)            Returns the tab disabled icon at index. |
| int | [**getDisplayedMnemonicIndexAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getDisplayedMnemonicIndexAt(int))(int tabIndex)            Returns the character, as an index, that the look and feel should provide decoration for as representing the mnemonic character. |
| [Color](http://docs.google.com/java/awt/Color.html) | [**getForegroundAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getForegroundAt(int))(int index)            Returns the tab foreground color at index. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getIconAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getIconAt(int))(int index)            Returns the tab icon at index. |
| int | [**getMnemonicAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getMnemonicAt(int))(int tabIndex)            Returns the keyboard mnemonic for accessing the specified tab. |
| [SingleSelectionModel](http://docs.google.com/javax/swing/SingleSelectionModel.html) | [**getModel**](http://docs.google.com/javax/swing/JTabbedPane.html#getModel())()            Returns the model associated with this tabbedpane. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getSelectedComponent**](http://docs.google.com/javax/swing/JTabbedPane.html#getSelectedComponent())()            Returns the currently selected component for this tabbedpane. |
| int | [**getSelectedIndex**](http://docs.google.com/javax/swing/JTabbedPane.html#getSelectedIndex())()            Returns the currently selected index for this tabbedpane. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getTabComponentAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getTabComponentAt(int))(int index)            Returns the tab component at index. |
| int | [**getTabCount**](http://docs.google.com/javax/swing/JTabbedPane.html#getTabCount())()            Returns the number of tabs in this tabbedpane. |
| int | [**getTabLayoutPolicy**](http://docs.google.com/javax/swing/JTabbedPane.html#getTabLayoutPolicy())()            Returns the policy used by the tabbedpane to layout the tabs when all the tabs will not fit within a single run. |
| int | [**getTabPlacement**](http://docs.google.com/javax/swing/JTabbedPane.html#getTabPlacement())()            Returns the placement of the tabs for this tabbedpane. |
| int | [**getTabRunCount**](http://docs.google.com/javax/swing/JTabbedPane.html#getTabRunCount())()            Returns the number of tab runs currently used to display the tabs. |
| [String](http://docs.google.com/java/lang/String.html) | [**getTitleAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getTitleAt(int))(int index)            Returns the tab title at index. |
| [String](http://docs.google.com/java/lang/String.html) | [**getToolTipText**](http://docs.google.com/javax/swing/JTabbedPane.html#getToolTipText(java.awt.event.MouseEvent))([MouseEvent](http://docs.google.com/java/awt/event/MouseEvent.html) event)            Returns the tooltip text for the component determined by the mouse event location. |
| [String](http://docs.google.com/java/lang/String.html) | [**getToolTipTextAt**](http://docs.google.com/javax/swing/JTabbedPane.html#getToolTipTextAt(int))(int index)            Returns the tab tooltip text at index. |
| [TabbedPaneUI](http://docs.google.com/javax/swing/plaf/TabbedPaneUI.html) | [**getUI**](http://docs.google.com/javax/swing/JTabbedPane.html#getUI())()            Returns the UI object which implements the L&F for this component. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUIClassID**](http://docs.google.com/javax/swing/JTabbedPane.html#getUIClassID())()            Returns the name of the UI class that implements the L&F for this component. |
| int | [**indexAtLocation**](http://docs.google.com/javax/swing/JTabbedPane.html#indexAtLocation(int,%20int))(int x, int y)            Returns the tab index corresponding to the tab whose bounds intersect the specified location. |
| int | [**indexOfComponent**](http://docs.google.com/javax/swing/JTabbedPane.html#indexOfComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) component)            Returns the index of the tab for the specified component. |
| int | [**indexOfTab**](http://docs.google.com/javax/swing/JTabbedPane.html#indexOfTab(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) icon)            Returns the first tab index with a given icon, or -1 if no tab has this icon. |
| int | [**indexOfTab**](http://docs.google.com/javax/swing/JTabbedPane.html#indexOfTab(java.lang.String))([String](http://docs.google.com/java/lang/String.html) title)            Returns the first tab index with a given title, or -1 if no tab has this title. |
| int | [**indexOfTabComponent**](http://docs.google.com/javax/swing/JTabbedPane.html#indexOfTabComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) tabComponent)            Returns the index of the tab for the specified tab component. |
| void | [**insertTab**](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) title, [Icon](http://docs.google.com/javax/swing/Icon.html) icon, [Component](http://docs.google.com/java/awt/Component.html) component, [String](http://docs.google.com/java/lang/String.html) tip, int index)            Inserts a component, at index, represented by a title and/or icon, either of which may be null. |
| boolean | [**isEnabledAt**](http://docs.google.com/javax/swing/JTabbedPane.html#isEnabledAt(int))(int index)            Returns whether or not the tab at index is currently enabled. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**paramString**](http://docs.google.com/javax/swing/JTabbedPane.html#paramString())()            Returns a string representation of this JTabbedPane. |
| void | [**remove**](http://docs.google.com/javax/swing/JTabbedPane.html#remove(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) component)            Removes the specified Component from the JTabbedPane. |
| void | [**remove**](http://docs.google.com/javax/swing/JTabbedPane.html#remove(int))(int index)            Removes the tab and component which corresponds to the specified index. |
| void | [**removeAll**](http://docs.google.com/javax/swing/JTabbedPane.html#removeAll())()            Removes all the tabs and their corresponding components from the tabbedpane. |
| void | [**removeChangeListener**](http://docs.google.com/javax/swing/JTabbedPane.html#removeChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)            Removes a ChangeListener from this tabbedpane. |
| void | [**removeTabAt**](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))(int index)            Removes the tab at index. |
| void | [**setBackgroundAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setBackgroundAt(int,%20java.awt.Color))(int index, [Color](http://docs.google.com/java/awt/Color.html) background)            Sets the background color at index to background which can be null, in which case the tab's background color will default to the background color of the tabbedpane. |
| void | [**setComponentAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setComponentAt(int,%20java.awt.Component))(int index, [Component](http://docs.google.com/java/awt/Component.html) component)            Sets the component at index to component. |
| void | [**setDisabledIconAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setDisabledIconAt(int,%20javax.swing.Icon))(int index, [Icon](http://docs.google.com/javax/swing/Icon.html) disabledIcon)            Sets the disabled icon at index to icon which can be null. |
| void | [**setDisplayedMnemonicIndexAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setDisplayedMnemonicIndexAt(int,%20int))(int tabIndex, int mnemonicIndex)            Provides a hint to the look and feel as to which character in the text should be decorated to represent the mnemonic. |
| void | [**setEnabledAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setEnabledAt(int,%20boolean))(int index, boolean enabled)            Sets whether or not the tab at index is enabled. |
| void | [**setForegroundAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setForegroundAt(int,%20java.awt.Color))(int index, [Color](http://docs.google.com/java/awt/Color.html) foreground)            Sets the foreground color at index to foreground which can be null, in which case the tab's foreground color will default to the foreground color of this tabbedpane. |
| void | [**setIconAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setIconAt(int,%20javax.swing.Icon))(int index, [Icon](http://docs.google.com/javax/swing/Icon.html) icon)            Sets the icon at index to icon which can be null. |
| void | [**setMnemonicAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setMnemonicAt(int,%20int))(int tabIndex, int mnemonic)            Sets the keyboard mnemonic for accessing the specified tab. |
| void | [**setModel**](http://docs.google.com/javax/swing/JTabbedPane.html#setModel(javax.swing.SingleSelectionModel))([SingleSelectionModel](http://docs.google.com/javax/swing/SingleSelectionModel.html) model)            Sets the model to be used with this tabbedpane. |
| void | [**setSelectedComponent**](http://docs.google.com/javax/swing/JTabbedPane.html#setSelectedComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) c)            Sets the selected component for this tabbedpane. |
| void | [**setSelectedIndex**](http://docs.google.com/javax/swing/JTabbedPane.html#setSelectedIndex(int))(int index)            Sets the selected index for this tabbedpane. |
| void | [**setTabComponentAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setTabComponentAt(int,%20java.awt.Component))(int index, [Component](http://docs.google.com/java/awt/Component.html) component)            Sets the component that is responsible for rendering the title for the specified tab. |
| void | [**setTabLayoutPolicy**](http://docs.google.com/javax/swing/JTabbedPane.html#setTabLayoutPolicy(int))(int tabLayoutPolicy)            Sets the policy which the tabbedpane will use in laying out the tabs when all the tabs will not fit within a single run. |
| void | [**setTabPlacement**](http://docs.google.com/javax/swing/JTabbedPane.html#setTabPlacement(int))(int tabPlacement)            Sets the tab placement for this tabbedpane. |
| void | [**setTitleAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setTitleAt(int,%20java.lang.String))(int index, [String](http://docs.google.com/java/lang/String.html) title)            Sets the title at index to title which can be null. |
| void | [**setToolTipTextAt**](http://docs.google.com/javax/swing/JTabbedPane.html#setToolTipTextAt(int,%20java.lang.String))(int index, [String](http://docs.google.com/java/lang/String.html) toolTipText)            Sets the tooltip text at index to toolTipText which can be null. |
| void | [**setUI**](http://docs.google.com/javax/swing/JTabbedPane.html#setUI(javax.swing.plaf.TabbedPaneUI))([TabbedPaneUI](http://docs.google.com/javax/swing/plaf/TabbedPaneUI.html) ui)            Sets the UI object which implements the L&F for this component. |
| void | [**updateUI**](http://docs.google.com/javax/swing/JTabbedPane.html#updateUI())()            Resets the UI property to a value from the current look and feel. |

| **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) |
| --- |
| [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)), [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** |
| --- |

### WRAP\_TAB\_LAYOUT

public static final int **WRAP\_TAB\_LAYOUT**

The tab layout policy for wrapping tabs in multiple runs when all tabs will not fit within a single run.

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

### SCROLL\_TAB\_LAYOUT

public static final int **SCROLL\_TAB\_LAYOUT**

Tab layout policy for providing a subset of available tabs when all the tabs will not fit within a single run. If all the tabs do not fit within a single run the look and feel will provide a way to navigate to hidden tabs.

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

### tabPlacement

protected int **tabPlacement**

Where the tabs are placed.

**See Also:**[setTabPlacement(int)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabPlacement(int))

### model

protected [SingleSelectionModel](http://docs.google.com/javax/swing/SingleSelectionModel.html) **model**

The default selection model

### changeListener

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

The changeListener is the listener we add to the model.

### changeEvent

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

Only one ChangeEvent is needed per TabPane instance since the event's only (read-only) state is the source property. The source of events generated here is always "this".

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

### JTabbedPane

public **JTabbedPane**()

Creates an empty TabbedPane with a default tab placement of JTabbedPane.TOP.

**See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String))

### JTabbedPane

public **JTabbedPane**(int tabPlacement)

Creates an empty TabbedPane with the specified tab placement of either: JTabbedPane.TOP, JTabbedPane.BOTTOM, JTabbedPane.LEFT, or JTabbedPane.RIGHT.

**Parameters:**tabPlacement - the placement for the tabs relative to the content**See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String))

### JTabbedPane

public **JTabbedPane**(int tabPlacement,  
 int tabLayoutPolicy)

Creates an empty TabbedPane with the specified tab placement and tab layout policy. Tab placement may be either: JTabbedPane.TOP, JTabbedPane.BOTTOM, JTabbedPane.LEFT, or JTabbedPane.RIGHT. Tab layout policy may be either: JTabbedPane.WRAP\_TAB\_LAYOUT or JTabbedPane.SCROLL\_TAB\_LAYOUT.

**Parameters:**tabPlacement - the placement for the tabs relative to the contenttabLayoutPolicy - the policy for laying out tabs when all tabs will not fit on one run **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if tab placement or tab layout policy are not one of the above supported values**Since:** 1.4 **See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String))

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

### getUI

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

Returns the UI object which implements the L&F for this component.

**Returns:**a TabbedPaneUI object**See Also:**[setUI(javax.swing.plaf.TabbedPaneUI)](http://docs.google.com/javax/swing/JTabbedPane.html#setUI(javax.swing.plaf.TabbedPaneUI))

### setUI

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

Sets the UI object which implements the L&F for this component.

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

### updateUI

public void **updateUI**()

Resets the UI property to a value from the current look and feel.

**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 UI class that implements the L&F for 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 "TabbedPaneUI"**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))

### createChangeListener

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

Subclasses that want to handle ChangeEvents differently can override this to return a subclass of ModelListener or another ChangeListener implementation.

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

### addChangeListener

public void **addChangeListener**([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)

Adds a ChangeListener to this tabbedpane.

**Parameters:**l - the ChangeListener to add**See Also:**[fireStateChanged()](http://docs.google.com/javax/swing/JTabbedPane.html#fireStateChanged()), [removeChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/JTabbedPane.html#removeChangeListener(javax.swing.event.ChangeListener))

### removeChangeListener

public void **removeChangeListener**([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)

Removes a ChangeListener from this tabbedpane.

**Parameters:**l - the ChangeListener to remove**See Also:**[fireStateChanged()](http://docs.google.com/javax/swing/JTabbedPane.html#fireStateChanged()), [addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/JTabbedPane.html#addChangeListener(javax.swing.event.ChangeListener))

### getChangeListeners

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

Returns an array of all the ChangeListeners added to this JTabbedPane with addChangeListener.

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

### fireStateChanged

protected void **fireStateChanged**()

Sends a ChangeEvent, with this JTabbedPane as the source, to each registered listener. This method is called each time there is a change to either the selected index or the selected tab in the JTabbedPane. Usually, the selected index and selected tab change together. However, there are some cases, such as tab addition, where the selected index changes and the same tab remains selected. There are other cases, such as deleting the selected tab, where the index remains the same, but a new tab moves to that index. Events are fired for all of these cases.

**See Also:**[addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/JTabbedPane.html#addChangeListener(javax.swing.event.ChangeListener)), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### getModel

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

Returns the model associated with this tabbedpane.

**See Also:**[setModel(javax.swing.SingleSelectionModel)](http://docs.google.com/javax/swing/JTabbedPane.html#setModel(javax.swing.SingleSelectionModel))

### setModel

public void **setModel**([SingleSelectionModel](http://docs.google.com/javax/swing/SingleSelectionModel.html) model)

Sets the model to be used with this tabbedpane.

**Parameters:**model - the model to be used**See Also:**[getModel()](http://docs.google.com/javax/swing/JTabbedPane.html#getModel())

### getTabPlacement

public int **getTabPlacement**()

Returns the placement of the tabs for this tabbedpane.

**See Also:**[setTabPlacement(int)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabPlacement(int))

### setTabPlacement

public void **setTabPlacement**(int tabPlacement)

Sets the tab placement for this tabbedpane. Possible values are:

* JTabbedPane.TOP
* JTabbedPane.BOTTOM
* JTabbedPane.LEFT
* JTabbedPane.RIGHT

The default value, if not set, is SwingConstants.TOP.

**Parameters:**tabPlacement - the placement for the tabs relative to the content **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if tab placement value isn't one of the above valid values

### getTabLayoutPolicy

public int **getTabLayoutPolicy**()

Returns the policy used by the tabbedpane to layout the tabs when all the tabs will not fit within a single run.

**Since:** 1.4 **See Also:**[setTabLayoutPolicy(int)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabLayoutPolicy(int))

### setTabLayoutPolicy

public void **setTabLayoutPolicy**(int tabLayoutPolicy)

Sets the policy which the tabbedpane will use in laying out the tabs when all the tabs will not fit within a single run. Possible values are:

* JTabbedPane.WRAP\_TAB\_LAYOUT
* JTabbedPane.SCROLL\_TAB\_LAYOUT

The default value, if not set by the UI, is JTabbedPane.WRAP\_TAB\_LAYOUT.

Some look and feels might only support a subset of the possible layout policies, in which case the value of this property may be ignored.

**Parameters:**tabLayoutPolicy - the policy used to layout the tabs **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if layoutPolicy value isn't one of the above valid values**Since:** 1.4 **See Also:**[getTabLayoutPolicy()](http://docs.google.com/javax/swing/JTabbedPane.html#getTabLayoutPolicy())

### getSelectedIndex

public int **getSelectedIndex**()

Returns the currently selected index for this tabbedpane. Returns -1 if there is no currently selected tab.

**Returns:**the index of the selected tab**See Also:**[setSelectedIndex(int)](http://docs.google.com/javax/swing/JTabbedPane.html#setSelectedIndex(int))

### setSelectedIndex

public void **setSelectedIndex**(int index)

Sets the selected index for this tabbedpane. The index must be a valid tab index or -1, which indicates that no tab should be selected (can also be used when there are no tabs in the tabbedpane). If a -1 value is specified when the tabbedpane contains one or more tabs, then the results will be implementation defined.

**Parameters:**index - the index to be selected **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < -1 || index >= tab count)**See Also:**[getSelectedIndex()](http://docs.google.com/javax/swing/JTabbedPane.html#getSelectedIndex()), [SingleSelectionModel.setSelectedIndex(int)](http://docs.google.com/javax/swing/SingleSelectionModel.html#setSelectedIndex(int))

### getSelectedComponent

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

Returns the currently selected component for this tabbedpane. Returns null if there is no currently selected tab.

**Returns:**the component corresponding to the selected tab**See Also:**[setSelectedComponent(java.awt.Component)](http://docs.google.com/javax/swing/JTabbedPane.html#setSelectedComponent(java.awt.Component))

### setSelectedComponent

public void **setSelectedComponent**([Component](http://docs.google.com/java/awt/Component.html) c)

Sets the selected component for this tabbedpane. This will automatically set the selectedIndex to the index corresponding to the specified component.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if component not found in tabbed pane**See Also:**[getSelectedComponent()](http://docs.google.com/javax/swing/JTabbedPane.html#getSelectedComponent())

### insertTab

public void **insertTab**([String](http://docs.google.com/java/lang/String.html) title,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) icon,  
 [Component](http://docs.google.com/java/awt/Component.html) component,  
 [String](http://docs.google.com/java/lang/String.html) tip,  
 int index)

Inserts a component, at index, represented by a title and/or icon, either of which may be null. Uses java.util.Vector internally, see insertElementAt for details of insertion conventions.

**Parameters:**title - the title to be displayed in this tabicon - the icon to be displayed in this tabcomponent - The component to be displayed when this tab is clicked.tip - the tooltip to be displayed for this tabindex - the position to insert this new tab**See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### addTab

public void **addTab**([String](http://docs.google.com/java/lang/String.html) title,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) icon,  
 [Component](http://docs.google.com/java/awt/Component.html) component,  
 [String](http://docs.google.com/java/lang/String.html) tip)

Adds a component and tip represented by a title and/or icon, either of which can be null. Cover method for insertTab.

**Parameters:**title - the title to be displayed in this tabicon - the icon to be displayed in this tabcomponent - the component to be displayed when this tab is clickedtip - the tooltip to be displayed for this tab**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### addTab

public void **addTab**([String](http://docs.google.com/java/lang/String.html) title,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) icon,  
 [Component](http://docs.google.com/java/awt/Component.html) component)

Adds a component represented by a title and/or icon, either of which can be null. Cover method for insertTab.

**Parameters:**title - the title to be displayed in this tabicon - the icon to be displayed in this tabcomponent - the component to be displayed when this tab is clicked**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### addTab

public void **addTab**([String](http://docs.google.com/java/lang/String.html) title,  
 [Component](http://docs.google.com/java/awt/Component.html) component)

Adds a component represented by a title and no icon. Cover method for insertTab.

**Parameters:**title - the title to be displayed in this tabcomponent - the component to be displayed when this tab is clicked**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### add

public [Component](http://docs.google.com/java/awt/Component.html) **add**([Component](http://docs.google.com/java/awt/Component.html) component)

Adds a component with a tab title defaulting to the name of the component which is the result of calling component.getName. Cover method for insertTab.

**Overrides:**[add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**component - the component to be displayed when this tab is clicked **Returns:**the component**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### add

public [Component](http://docs.google.com/java/awt/Component.html) **add**([String](http://docs.google.com/java/lang/String.html) title,  
 [Component](http://docs.google.com/java/awt/Component.html) component)

Adds a component with the specified tab title. Cover method for insertTab.

**Overrides:**[add](http://docs.google.com/java/awt/Container.html#add(java.lang.String,%20java.awt.Component)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**title - the title to be displayed in this tabcomponent - the component to be displayed when this tab is clicked **Returns:**the component**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### add

public [Component](http://docs.google.com/java/awt/Component.html) **add**([Component](http://docs.google.com/java/awt/Component.html) component,  
 int index)

Adds a component at the specified tab index with a tab title defaulting to the name of the component. Cover method for insertTab.

**Overrides:**[add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20int)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**component - the component to be displayed when this tab is clickedindex - the position to insert this new tab **Returns:**the component**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### add

public void **add**([Component](http://docs.google.com/java/awt/Component.html) component,  
 [Object](http://docs.google.com/java/lang/Object.html) constraints)

Adds a component to the tabbed pane. If constraints is a String or an Icon, it will be used for the tab title, otherwise the component's name will be used as the tab title. Cover method for insertTab.

**Overrides:**[add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**component - the component to be displayed when this tab is clickedconstraints - the object to be displayed in the tab**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### add

public void **add**([Component](http://docs.google.com/java/awt/Component.html) component,  
 [Object](http://docs.google.com/java/lang/Object.html) constraints,  
 int index)

Adds a component at the specified tab index. If constraints is a String or an Icon, it will be used for the tab title, otherwise the component's name will be used as the tab title. Cover method for insertTab.

**Overrides:**[add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object,%20int)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**component - the component to be displayed when this tab is clickedconstraints - the object to be displayed in the tabindex - the position to insert this new tab**See Also:**[insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### removeTabAt

public void **removeTabAt**(int index)

Removes the tab at index. After the component associated with index is removed, its visibility is reset to true to ensure it will be visible if added to other containers.

**Parameters:**index - the index of the tab to be removed **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String)), [insertTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String, int)](http://docs.google.com/javax/swing/JTabbedPane.html#insertTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String,%20int))

### remove

public void **remove**([Component](http://docs.google.com/java/awt/Component.html) component)

Removes the specified Component from the JTabbedPane. The method does nothing if the component is null.

**Overrides:**[remove](http://docs.google.com/java/awt/Container.html#remove(java.awt.Component)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**component - the component to remove from the tabbedpane**See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### remove

public void **remove**(int index)

Removes the tab and component which corresponds to the specified index.

**Overrides:**[remove](http://docs.google.com/java/awt/Container.html#remove(int)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**index - the index of the component to remove from the tabbedpane **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### removeAll

public void **removeAll**()

Removes all the tabs and their corresponding components from the tabbedpane.

**Overrides:**[removeAll](http://docs.google.com/java/awt/Container.html#removeAll()) in class [Container](http://docs.google.com/java/awt/Container.html) **See Also:**[addTab(java.lang.String, javax.swing.Icon, java.awt.Component, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#addTab(java.lang.String,%20javax.swing.Icon,%20java.awt.Component,%20java.lang.String)), [removeTabAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#removeTabAt(int))

### getTabCount

public int **getTabCount**()

Returns the number of tabs in this tabbedpane.

**Returns:**an integer specifying the number of tabbed pages

### getTabRunCount

public int **getTabRunCount**()

Returns the number of tab runs currently used to display the tabs.

**Returns:**an integer giving the number of rows if the tabPlacement is TOP or BOTTOM and the number of columns if tabPlacement is LEFT or RIGHT, or 0 if there is no UI set on this tabbedpane

### getTitleAt

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

Returns the tab title at index.

**Parameters:**index - the index of the item being queried **Returns:**the title at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setTitleAt(int, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#setTitleAt(int,%20java.lang.String))

### getIconAt

public [Icon](http://docs.google.com/javax/swing/Icon.html) **getIconAt**(int index)

Returns the tab icon at index.

**Parameters:**index - the index of the item being queried **Returns:**the icon at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setIconAt(int, javax.swing.Icon)](http://docs.google.com/javax/swing/JTabbedPane.html#setIconAt(int,%20javax.swing.Icon))

### getDisabledIconAt

public [Icon](http://docs.google.com/javax/swing/Icon.html) **getDisabledIconAt**(int index)

Returns the tab disabled icon at index. If the tab disabled icon doesn't exist at index this will forward the call to the look and feel to construct an appropriate disabled Icon from the corresponding enabled Icon. Some look and feels might not render the disabled Icon, in which case it won't be created.

**Parameters:**index - the index of the item being queried **Returns:**the icon at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setDisabledIconAt(int, javax.swing.Icon)](http://docs.google.com/javax/swing/JTabbedPane.html#setDisabledIconAt(int,%20javax.swing.Icon))

### getToolTipTextAt

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

Returns the tab tooltip text at index.

**Parameters:**index - the index of the item being queried **Returns:**a string containing the tool tip text at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**Since:** 1.3 **See Also:**[setToolTipTextAt(int, java.lang.String)](http://docs.google.com/javax/swing/JTabbedPane.html#setToolTipTextAt(int,%20java.lang.String))

### getBackgroundAt

public [Color](http://docs.google.com/java/awt/Color.html) **getBackgroundAt**(int index)

Returns the tab background color at index.

**Parameters:**index - the index of the item being queried **Returns:**the Color of the tab background at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setBackgroundAt(int, java.awt.Color)](http://docs.google.com/javax/swing/JTabbedPane.html#setBackgroundAt(int,%20java.awt.Color))

### getForegroundAt

public [Color](http://docs.google.com/java/awt/Color.html) **getForegroundAt**(int index)

Returns the tab foreground color at index.

**Parameters:**index - the index of the item being queried **Returns:**the Color of the tab foreground at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setForegroundAt(int, java.awt.Color)](http://docs.google.com/javax/swing/JTabbedPane.html#setForegroundAt(int,%20java.awt.Color))

### isEnabledAt

public boolean **isEnabledAt**(int index)

Returns whether or not the tab at index is currently enabled.

**Parameters:**index - the index of the item being queried **Returns:**true if the tab at index is enabled; false otherwise **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setEnabledAt(int, boolean)](http://docs.google.com/javax/swing/JTabbedPane.html#setEnabledAt(int,%20boolean))

### getComponentAt

public [Component](http://docs.google.com/java/awt/Component.html) **getComponentAt**(int index)

Returns the component at index.

**Parameters:**index - the index of the item being queried **Returns:**the Component at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setComponentAt(int, java.awt.Component)](http://docs.google.com/javax/swing/JTabbedPane.html#setComponentAt(int,%20java.awt.Component))

### getMnemonicAt

public int **getMnemonicAt**(int tabIndex)

Returns the keyboard mnemonic for accessing the specified tab. The mnemonic is the key which when combined with the look and feel's mouseless modifier (usually Alt) will activate the specified tab.

**Parameters:**tabIndex - the index of the tab that the mnemonic refers to **Returns:**the key code which represents the mnemonic; -1 if a mnemonic is not specified for the tab **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (tabIndex < 0 || tabIndex >= tab count)**Since:** 1.4 **See Also:**[setDisplayedMnemonicIndexAt(int,int)](http://docs.google.com/javax/swing/JTabbedPane.html#setDisplayedMnemonicIndexAt(int,%20int)), [setMnemonicAt(int,int)](http://docs.google.com/javax/swing/JTabbedPane.html#setMnemonicAt(int,%20int))

### getDisplayedMnemonicIndexAt

public int **getDisplayedMnemonicIndexAt**(int tabIndex)

Returns the character, as an index, that the look and feel should provide decoration for as representing the mnemonic character.

**Parameters:**tabIndex - the index of the tab that the mnemonic refers to **Returns:**index representing mnemonic character if one exists; otherwise returns -1 **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (tabIndex < 0 || tabIndex >= tab count)**Since:** 1.4 **See Also:**[setDisplayedMnemonicIndexAt(int,int)](http://docs.google.com/javax/swing/JTabbedPane.html#setDisplayedMnemonicIndexAt(int,%20int)), [setMnemonicAt(int,int)](http://docs.google.com/javax/swing/JTabbedPane.html#setMnemonicAt(int,%20int))

### getBoundsAt

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

Returns the tab bounds at index. If the tab at this index is not currently visible in the UI, then returns null. If there is no UI set on this tabbedpane, then returns null.

**Parameters:**index - the index to be queried **Returns:**a Rectangle containing the tab bounds at index, or null if tab at index is not currently visible in the UI, or if there is no UI set on this tabbedpane **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)

### setTitleAt

public void **setTitleAt**(int index,  
 [String](http://docs.google.com/java/lang/String.html) title)

Sets the title at index to title which can be null. The title is not shown if a tab component for this tab was specified. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where the title should be settitle - the title to be displayed in the tab **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[getTitleAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getTitleAt(int)), [setTabComponentAt(int, java.awt.Component)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabComponentAt(int,%20java.awt.Component))

### setIconAt

public void **setIconAt**(int index,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) icon)

Sets the icon at index to icon which can be null. This does not set disabled icon at icon. If the new Icon is different than the current Icon and disabled icon is not explicitly set, the LookAndFeel will be asked to generate a disabled Icon. To explicitly set disabled icon, use setDisableIconAt(). The icon is not shown if a tab component for this tab was specified. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where the icon should be seticon - the icon to be displayed in the tab **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[setDisabledIconAt(int, javax.swing.Icon)](http://docs.google.com/javax/swing/JTabbedPane.html#setDisabledIconAt(int,%20javax.swing.Icon)), [getIconAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getIconAt(int)), [getDisabledIconAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getDisabledIconAt(int)), [setTabComponentAt(int, java.awt.Component)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabComponentAt(int,%20java.awt.Component))

### setDisabledIconAt

public void **setDisabledIconAt**(int index,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) disabledIcon)

Sets the disabled icon at index to icon which can be null. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where the disabled icon should be setdisabledIcon - the icon to be displayed in the tab when disabled **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[getDisabledIconAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getDisabledIconAt(int))

### setToolTipTextAt

public void **setToolTipTextAt**(int index,  
 [String](http://docs.google.com/java/lang/String.html) toolTipText)

Sets the tooltip text at index to toolTipText which can be null. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where the tooltip text should be settoolTipText - the tooltip text to be displayed for the tab **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**Since:** 1.3 **See Also:**[getToolTipTextAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getToolTipTextAt(int))

### setBackgroundAt

public void **setBackgroundAt**(int index,  
 [Color](http://docs.google.com/java/awt/Color.html) background)

Sets the background color at index to background which can be null, in which case the tab's background color will default to the background color of the tabbedpane. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where the background should be setbackground - the color to be displayed in the tab's background **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[getBackgroundAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getBackgroundAt(int))

### setForegroundAt

public void **setForegroundAt**(int index,  
 [Color](http://docs.google.com/java/awt/Color.html) foreground)

Sets the foreground color at index to foreground which can be null, in which case the tab's foreground color will default to the foreground color of this tabbedpane. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where the foreground should be setforeground - the color to be displayed as the tab's foreground **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[getForegroundAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getForegroundAt(int))

### setEnabledAt

public void **setEnabledAt**(int index,  
 boolean enabled)

Sets whether or not the tab at index is enabled. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index which should be enabled/disabledenabled - whether or not the tab should be enabled **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[isEnabledAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#isEnabledAt(int))

### setComponentAt

public void **setComponentAt**(int index,  
 [Component](http://docs.google.com/java/awt/Component.html) component)

Sets the component at index to component. An internal exception is raised if there is no tab at that index.

**Parameters:**index - the tab index where this component is being placedcomponent - the component for the tab **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**See Also:**[getComponentAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getComponentAt(int))

### setDisplayedMnemonicIndexAt

public void **setDisplayedMnemonicIndexAt**(int tabIndex,  
 int mnemonicIndex)

Provides a hint to the look and feel as to which character in the text should be decorated to represent the mnemonic. Not all look and feels may support this. A value of -1 indicates either there is no mnemonic for this tab, or you do not wish the mnemonic to be displayed for this tab.

The value of this is updated as the properties relating to the mnemonic change (such as the mnemonic itself, the text...). You should only ever have to call this if you do not wish the default character to be underlined. For example, if the text at tab index 3 was 'Apple Price', with a mnemonic of 'p', and you wanted the 'P' to be decorated, as 'Apple Price', you would have to invoke setDisplayedMnemonicIndex(3, 6) after invoking setMnemonicAt(3, KeyEvent.VK\_P).

Note that it is the programmer's responsibility to ensure that each tab has a unique mnemonic or unpredictable results may occur.

**Parameters:**tabIndex - the index of the tab that the mnemonic refers tomnemonicIndex - index into the String to underline **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if tabIndex is out of range (tabIndex < 0 || tabIndex >= tab count) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - will be thrown if mnemonicIndex is >= length of the tab title , or < -1**Since:** 1.4 **See Also:**[setMnemonicAt(int,int)](http://docs.google.com/javax/swing/JTabbedPane.html#setMnemonicAt(int,%20int)), [getDisplayedMnemonicIndexAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getDisplayedMnemonicIndexAt(int))

### setMnemonicAt

public void **setMnemonicAt**(int tabIndex,  
 int mnemonic)

Sets the keyboard mnemonic for accessing the specified tab. The mnemonic is the key which when combined with the look and feel's mouseless modifier (usually Alt) will activate the specified tab.

A mnemonic must correspond to a single key on the keyboard and should be specified using one of the VK\_XXX keycodes defined in java.awt.event.KeyEvent. Mnemonics are case-insensitive, therefore a key event with the corresponding keycode would cause the button to be activated whether or not the Shift modifier was pressed.

This will update the displayed mnemonic property for the specified tab.

**Parameters:**tabIndex - the index of the tab that the mnemonic refers tomnemonic - the key code which represents the mnemonic **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if tabIndex is out of range (tabIndex < 0 || tabIndex >= tab count)**Since:** 1.4 **See Also:**[getMnemonicAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getMnemonicAt(int)), [setDisplayedMnemonicIndexAt(int,int)](http://docs.google.com/javax/swing/JTabbedPane.html#setDisplayedMnemonicIndexAt(int,%20int))

### indexOfTab

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

Returns the first tab index with a given title, or -1 if no tab has this title.

**Parameters:**title - the title for the tab **Returns:**the first tab index which matches title, or -1 if no tab has this title

### indexOfTab

public int **indexOfTab**([Icon](http://docs.google.com/javax/swing/Icon.html) icon)

Returns the first tab index with a given icon, or -1 if no tab has this icon.

**Parameters:**icon - the icon for the tab **Returns:**the first tab index which matches icon, or -1 if no tab has this icon

### indexOfComponent

public int **indexOfComponent**([Component](http://docs.google.com/java/awt/Component.html) component)

Returns the index of the tab for the specified component. Returns -1 if there is no tab for this component.

**Parameters:**component - the component for the tab **Returns:**the first tab which matches this component, or -1 if there is no tab for this component

### indexAtLocation

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

Returns the tab index corresponding to the tab whose bounds intersect the specified location. Returns -1 if no tab intersects the location.

**Parameters:**x - the x location relative to this tabbedpaney - the y location relative to this tabbedpane **Returns:**the tab index which intersects the location, or -1 if no tab intersects the location**Since:** 1.4

### getToolTipText

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

Returns the tooltip text for the component determined by the mouse event location.

**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 tells where the cursor is lingering **Returns:**the String containing the tooltip text

### paramString

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

Returns a string representation of this JTabbedPane. 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 JTabbedPane.

### getAccessibleContext

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

Gets the AccessibleContext associated with this JTabbedPane. For tabbed panes, the AccessibleContext takes the form of an AccessibleJTabbedPane. A new AccessibleJTabbedPane 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 AccessibleJTabbedPane that serves as the AccessibleContext of this JTabbedPane

### setTabComponentAt

public void **setTabComponentAt**(int index,  
 [Component](http://docs.google.com/java/awt/Component.html) component)

Sets the component that is responsible for rendering the title for the specified tab. A null value means JTabbedPane will render the title and/or icon for the specified tab. A non-null value means the component will render the title and JTabbedPane will not render the title and/or icon.

Note: The component must not be one that the developer has already added to the tabbed pane.

**Parameters:**index - the tab index where the component should be setcomponent - the component to render the title for the specified tab **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if component has already been added to this JTabbedPane**Since:** 1.6 **See Also:**[getTabComponentAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getTabComponentAt(int))

### getTabComponentAt

public [Component](http://docs.google.com/java/awt/Component.html) **getTabComponentAt**(int index)

Returns the tab component at index.

**Parameters:**index - the index of the item being queried **Returns:**the tab component at index **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is out of range (index < 0 || index >= tab count)**Since:** 1.6 **See Also:**[setTabComponentAt(int, java.awt.Component)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabComponentAt(int,%20java.awt.Component))

### indexOfTabComponent

public int **indexOfTabComponent**([Component](http://docs.google.com/java/awt/Component.html) tabComponent)

Returns the index of the tab for the specified tab component. Returns -1 if there is no tab for this tab component.

**Parameters:**tabComponent - the tab component for the tab **Returns:**the first tab which matches this tab component, or -1 if there is no tab for this tab component**Since:** 1.6 **See Also:**[setTabComponentAt(int, java.awt.Component)](http://docs.google.com/javax/swing/JTabbedPane.html#setTabComponentAt(int,%20java.awt.Component)), [getTabComponentAt(int)](http://docs.google.com/javax/swing/JTabbedPane.html#getTabComponentAt(int))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JTabbedPane.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/JSplitPane.AccessibleJSplitPane.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JTabbedPane.AccessibleJTabbedPane.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JTabbedPane.html)    [**NO FRAMES**](http://docs.google.com/JTabbedPane.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#lnxbz9) | DETAIL: [FIELD](#z337ya) | [CONSTR](#2bn6wsx) | [METHOD](#49x2ik5) |

[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).