| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JSplitPane.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/JSpinner.NumberEditor.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JSplitPane.AccessibleJSplitPane.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JSplitPane.html)    [**NO FRAMES**](http://docs.google.com/JSplitPane.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#1t3h5sf) | [CONSTR](#3rdcrjn) | [METHOD](#26in1rg) | DETAIL: [FIELD](#2jxsxqh) | [CONSTR](#vx1227) | [METHOD](#3tbugp1) |

## **javax.swing**

Class JSplitPane

[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.JSplitPane**

**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)

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

JSplitPane is used to divide two (and only two) Components. The two Components are graphically divided based on the look and feel implementation, and the two Components can then be interactively resized by the user. Information on using JSplitPane is in [How to Use Split Panes](http://java.sun.com/docs/books/tutorial/uiswing/components/splitpane.html) in *The Java Tutorial*.

The two Components in a split pane can be aligned left to right using JSplitPane.HORIZONTAL\_SPLIT, or top to bottom using JSplitPane.VERTICAL\_SPLIT. The preferred way to change the size of the Components is to invoke setDividerLocation where location is either the new x or y position, depending on the orientation of the JSplitPane.

To resize the Components to their preferred sizes invoke resetToPreferredSizes.

When the user is resizing the Components the minimum size of the Components is used to determine the maximum/minimum position the Components can be set to. If the minimum size of the two components is greater than the size of the split pane the divider will not allow you to resize it. To alter the minimum size of a JComponent, see [JComponent.setMinimumSize(java.awt.Dimension)](http://docs.google.com/javax/swing/JComponent.html#setMinimumSize(java.awt.Dimension)).

When the user resizes the split pane the new space is distributed between the two components based on the resizeWeight property. A value of 0, the default, indicates the right/bottom component gets all the space, where as a value of 1 indicates the left/top component gets all the space.

**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:**[setDividerLocation(double)](http://docs.google.com/javax/swing/JSplitPane.html#setDividerLocation(double)), [resetToPreferredSizes()](http://docs.google.com/javax/swing/JSplitPane.html#resetToPreferredSizes())

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**JSplitPane.AccessibleJSplitPane**](http://docs.google.com/javax/swing/JSplitPane.AccessibleJSplitPane.html)            This class implements accessibility support for the JSplitPane class. |

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

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

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

| **Field Summary** | |
| --- | --- |
| static [String](http://docs.google.com/java/lang/String.html) | [**BOTTOM**](http://docs.google.com/javax/swing/JSplitPane.html#BOTTOM)            Used to add a Component below the other Component. |
| static [String](http://docs.google.com/java/lang/String.html) | [**CONTINUOUS\_LAYOUT\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#CONTINUOUS_LAYOUT_PROPERTY)            Bound property name for continuousLayout. |
| protected  boolean | [**continuousLayout**](http://docs.google.com/javax/swing/JSplitPane.html#continuousLayout)            Whether or not the views are continuously redisplayed while resizing. |
| static [String](http://docs.google.com/java/lang/String.html) | [**DIVIDER**](http://docs.google.com/javax/swing/JSplitPane.html#DIVIDER)            Used to add a Component that will represent the divider. |
| static [String](http://docs.google.com/java/lang/String.html) | [**DIVIDER\_LOCATION\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#DIVIDER_LOCATION_PROPERTY)            Bound property for the dividerLocation. |
| static [String](http://docs.google.com/java/lang/String.html) | [**DIVIDER\_SIZE\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#DIVIDER_SIZE_PROPERTY)            Bound property name for border. |
| protected  int | [**dividerSize**](http://docs.google.com/javax/swing/JSplitPane.html#dividerSize)            Size of the divider. |
| static int | [**HORIZONTAL\_SPLIT**](http://docs.google.com/javax/swing/JSplitPane.html#HORIZONTAL_SPLIT)            Horizontal split indicates the Components are split along the x axis. |
| static [String](http://docs.google.com/java/lang/String.html) | [**LAST\_DIVIDER\_LOCATION\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#LAST_DIVIDER_LOCATION_PROPERTY)            Bound property for lastLocation. |
| protected  int | [**lastDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#lastDividerLocation)            Previous location of the split pane. |
| static [String](http://docs.google.com/java/lang/String.html) | [**LEFT**](http://docs.google.com/javax/swing/JSplitPane.html#LEFT)            Used to add a Component to the left of the other Component. |
| protected  [Component](http://docs.google.com/java/awt/Component.html) | [**leftComponent**](http://docs.google.com/javax/swing/JSplitPane.html#leftComponent)            The left or top component. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ONE\_TOUCH\_EXPANDABLE\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#ONE_TOUCH_EXPANDABLE_PROPERTY)            Bound property for oneTouchExpandable. |
| protected  boolean | [**oneTouchExpandable**](http://docs.google.com/javax/swing/JSplitPane.html#oneTouchExpandable)            Is a little widget provided to quickly expand/collapse the split pane? |
| protected  int | [**orientation**](http://docs.google.com/javax/swing/JSplitPane.html#orientation)            How the views are split. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ORIENTATION\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#ORIENTATION_PROPERTY)            Bound property name for orientation (horizontal or vertical). |
| static [String](http://docs.google.com/java/lang/String.html) | [**RESIZE\_WEIGHT\_PROPERTY**](http://docs.google.com/javax/swing/JSplitPane.html#RESIZE_WEIGHT_PROPERTY)            Bound property for weight. |
| static [String](http://docs.google.com/java/lang/String.html) | [**RIGHT**](http://docs.google.com/javax/swing/JSplitPane.html#RIGHT)            Used to add a Component to the right of the other Component. |
| protected  [Component](http://docs.google.com/java/awt/Component.html) | [**rightComponent**](http://docs.google.com/javax/swing/JSplitPane.html#rightComponent)            The right or bottom component. |
| static [String](http://docs.google.com/java/lang/String.html) | [**TOP**](http://docs.google.com/javax/swing/JSplitPane.html#TOP)            Used to add a Component above the other Component. |
| static int | [**VERTICAL\_SPLIT**](http://docs.google.com/javax/swing/JSplitPane.html#VERTICAL_SPLIT)            Vertical split indicates the Components are split along the y axis. |

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

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

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

| **Constructor Summary** | |
| --- | --- |
| [**JSplitPane**](http://docs.google.com/javax/swing/JSplitPane.html#JSplitPane())()            Creates a new JSplitPane configured to arrange the child components side-by-side horizontally with no continuous layout, using two buttons for the components. |
| [**JSplitPane**](http://docs.google.com/javax/swing/JSplitPane.html#JSplitPane(int))(int newOrientation)            Creates a new JSplitPane configured with the specified orientation and no continuous layout. |
| [**JSplitPane**](http://docs.google.com/javax/swing/JSplitPane.html#JSplitPane(int,%20boolean))(int newOrientation, boolean newContinuousLayout)            Creates a new JSplitPane with the specified orientation and redrawing style. |
| [**JSplitPane**](http://docs.google.com/javax/swing/JSplitPane.html#JSplitPane(int,%20boolean,%20java.awt.Component,%20java.awt.Component))(int newOrientation, boolean newContinuousLayout, [Component](http://docs.google.com/java/awt/Component.html) newLeftComponent, [Component](http://docs.google.com/java/awt/Component.html) newRightComponent)            Creates a new JSplitPane with the specified orientation and redrawing style, and with the specified components. |
| [**JSplitPane**](http://docs.google.com/javax/swing/JSplitPane.html#JSplitPane(int,%20java.awt.Component,%20java.awt.Component))(int newOrientation, [Component](http://docs.google.com/java/awt/Component.html) newLeftComponent, [Component](http://docs.google.com/java/awt/Component.html) newRightComponent)            Creates a new JSplitPane with the specified orientation and with the specified components that do not do continuous redrawing. |

| **Method Summary** | |
| --- | --- |
| protected  void | [**addImpl**](http://docs.google.com/javax/swing/JSplitPane.html#addImpl(java.awt.Component,%20java.lang.Object,%20int))([Component](http://docs.google.com/java/awt/Component.html) comp, [Object](http://docs.google.com/java/lang/Object.html) constraints, int index)            Adds the specified component to this split pane. |
| [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) | [**getAccessibleContext**](http://docs.google.com/javax/swing/JSplitPane.html#getAccessibleContext())()            Gets the AccessibleContext associated with this JSplitPane. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getBottomComponent**](http://docs.google.com/javax/swing/JSplitPane.html#getBottomComponent())()            Returns the component below, or to the right of the divider. |
| int | [**getDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#getDividerLocation())()            Returns the last value passed to setDividerLocation. |
| int | [**getDividerSize**](http://docs.google.com/javax/swing/JSplitPane.html#getDividerSize())()            Returns the size of the divider. |
| int | [**getLastDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#getLastDividerLocation())()            Returns the last location the divider was at. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getLeftComponent**](http://docs.google.com/javax/swing/JSplitPane.html#getLeftComponent())()            Returns the component to the left (or above) the divider. |
| int | [**getMaximumDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#getMaximumDividerLocation())()            Returns the maximum location of the divider from the look and feel implementation. |
| int | [**getMinimumDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#getMinimumDividerLocation())()            Returns the minimum location of the divider from the look and feel implementation. |
| int | [**getOrientation**](http://docs.google.com/javax/swing/JSplitPane.html#getOrientation())()            Returns the orientation. |
| double | [**getResizeWeight**](http://docs.google.com/javax/swing/JSplitPane.html#getResizeWeight())()            Returns the number that determines how extra space is distributed. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getRightComponent**](http://docs.google.com/javax/swing/JSplitPane.html#getRightComponent())()            Returns the component to the right (or below) the divider. |
| [Component](http://docs.google.com/java/awt/Component.html) | [**getTopComponent**](http://docs.google.com/javax/swing/JSplitPane.html#getTopComponent())()            Returns the component above, or to the left of the divider. |
| [SplitPaneUI](http://docs.google.com/javax/swing/plaf/SplitPaneUI.html) | [**getUI**](http://docs.google.com/javax/swing/JSplitPane.html#getUI())()            Returns the SplitPaneUI that is providing the current look and feel. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUIClassID**](http://docs.google.com/javax/swing/JSplitPane.html#getUIClassID())()            Returns the name of the L&F class that renders this component. |
| boolean | [**isContinuousLayout**](http://docs.google.com/javax/swing/JSplitPane.html#isContinuousLayout())()            Gets the continuousLayout property. |
| boolean | [**isOneTouchExpandable**](http://docs.google.com/javax/swing/JSplitPane.html#isOneTouchExpandable())()            Gets the oneTouchExpandable property. |
| boolean | [**isValidateRoot**](http://docs.google.com/javax/swing/JSplitPane.html#isValidateRoot())()            Returns true, so that calls to revalidate on any descendant of this JSplitPane will cause a request to be queued that will validate the JSplitPane and all its descendants. |
| protected  void | [**paintChildren**](http://docs.google.com/javax/swing/JSplitPane.html#paintChildren(java.awt.Graphics))([Graphics](http://docs.google.com/java/awt/Graphics.html) g)            Subclassed to message the UI with finishedPaintingChildren after super has been messaged, as well as painting the border. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**paramString**](http://docs.google.com/javax/swing/JSplitPane.html#paramString())()            Returns a string representation of this JSplitPane. |
| void | [**remove**](http://docs.google.com/javax/swing/JSplitPane.html#remove(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) component)            Removes the child component, component from the pane. |
| void | [**remove**](http://docs.google.com/javax/swing/JSplitPane.html#remove(int))(int index)            Removes the Component at the specified index. |
| void | [**removeAll**](http://docs.google.com/javax/swing/JSplitPane.html#removeAll())()            Removes all the child components from the split pane. |
| void | [**resetToPreferredSizes**](http://docs.google.com/javax/swing/JSplitPane.html#resetToPreferredSizes())()            Lays out the JSplitPane layout based on the preferred size of the children components. |
| void | [**setBottomComponent**](http://docs.google.com/javax/swing/JSplitPane.html#setBottomComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) comp)            Sets the component below, or to the right of the divider. |
| void | [**setContinuousLayout**](http://docs.google.com/javax/swing/JSplitPane.html#setContinuousLayout(boolean))(boolean newContinuousLayout)            Sets the value of the continuousLayout property, which must be true for the child components to be continuously redisplayed and laid out during user intervention. |
| void | [**setDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#setDividerLocation(double))(double proportionalLocation)            Sets the divider location as a percentage of the JSplitPane's size. |
| void | [**setDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#setDividerLocation(int))(int location)            Sets the location of the divider. |
| void | [**setDividerSize**](http://docs.google.com/javax/swing/JSplitPane.html#setDividerSize(int))(int newSize)            Sets the size of the divider. |
| void | [**setLastDividerLocation**](http://docs.google.com/javax/swing/JSplitPane.html#setLastDividerLocation(int))(int newLastLocation)            Sets the last location the divider was at to newLastLocation. |
| void | [**setLeftComponent**](http://docs.google.com/javax/swing/JSplitPane.html#setLeftComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) comp)            Sets the component to the left (or above) the divider. |
| void | [**setOneTouchExpandable**](http://docs.google.com/javax/swing/JSplitPane.html#setOneTouchExpandable(boolean))(boolean newValue)            Sets the value of the oneTouchExpandable property, which must be true for the JSplitPane to provide a UI widget on the divider to quickly expand/collapse the divider. |
| void | [**setOrientation**](http://docs.google.com/javax/swing/JSplitPane.html#setOrientation(int))(int orientation)            Sets the orientation, or how the splitter is divided. |
| void | [**setResizeWeight**](http://docs.google.com/javax/swing/JSplitPane.html#setResizeWeight(double))(double value)            Specifies how to distribute extra space when the size of the split pane changes. |
| void | [**setRightComponent**](http://docs.google.com/javax/swing/JSplitPane.html#setRightComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) comp)            Sets the component to the right (or below) the divider. |
| void | [**setTopComponent**](http://docs.google.com/javax/swing/JSplitPane.html#setTopComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) comp)            Sets the component above, or to the left of the divider. |
| void | [**setUI**](http://docs.google.com/javax/swing/JSplitPane.html#setUI(javax.swing.plaf.SplitPaneUI))([SplitPaneUI](http://docs.google.com/javax/swing/plaf/SplitPaneUI.html) ui)            Sets the L&F object that renders this component. |
| void | [**updateUI**](http://docs.google.com/javax/swing/JSplitPane.html#updateUI())()            Notification from the UIManager that the L&F has changed. |

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

| **Methods inherited from class java.awt.**[**Container**](http://docs.google.com/java/awt/Container.html) |
| --- |
| [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)), [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20int)), [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object)), [add](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object,%20int)), [add](http://docs.google.com/java/awt/Container.html#add(java.lang.String,%20java.awt.Component)), [addContainerListener](http://docs.google.com/java/awt/Container.html#addContainerListener(java.awt.event.ContainerListener)), [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** |
| --- |

### VERTICAL\_SPLIT

public static final int **VERTICAL\_SPLIT**

Vertical split indicates the Components are split along the y axis. For example the two Components will be split one on top of the other.

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

### HORIZONTAL\_SPLIT

public static final int **HORIZONTAL\_SPLIT**

Horizontal split indicates the Components are split along the x axis. For example the two Components will be split one to the left of the other.

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

### LEFT

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

Used to add a Component to the left of the other Component.

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

### RIGHT

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

Used to add a Component to the right of the other Component.

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

### TOP

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

Used to add a Component above the other Component.

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

### BOTTOM

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

Used to add a Component below the other Component.

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

### DIVIDER

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

Used to add a Component that will represent the divider.

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

### ORIENTATION\_PROPERTY

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

Bound property name for orientation (horizontal or vertical).

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

### CONTINUOUS\_LAYOUT\_PROPERTY

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

Bound property name for continuousLayout.

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

### DIVIDER\_SIZE\_PROPERTY

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

Bound property name for border.

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

### ONE\_TOUCH\_EXPANDABLE\_PROPERTY

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

Bound property for oneTouchExpandable.

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

### LAST\_DIVIDER\_LOCATION\_PROPERTY

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

Bound property for lastLocation.

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

### DIVIDER\_LOCATION\_PROPERTY

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

Bound property for the dividerLocation.

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

### RESIZE\_WEIGHT\_PROPERTY

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

Bound property for weight.

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

### orientation

protected int **orientation**

How the views are split.

### continuousLayout

protected boolean **continuousLayout**

Whether or not the views are continuously redisplayed while resizing.

### leftComponent

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

The left or top component.

### rightComponent

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

The right or bottom component.

### dividerSize

protected int **dividerSize**

Size of the divider.

### oneTouchExpandable

protected boolean **oneTouchExpandable**

Is a little widget provided to quickly expand/collapse the split pane?

### lastDividerLocation

protected int **lastDividerLocation**

Previous location of the split pane.

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

### JSplitPane

public **JSplitPane**()

Creates a new JSplitPane configured to arrange the child components side-by-side horizontally with no continuous layout, using two buttons for the components.

### JSplitPane

public **JSplitPane**(int newOrientation)

Creates a new JSplitPane configured with the specified orientation and no continuous layout.

**Parameters:**newOrientation - JSplitPane.HORIZONTAL\_SPLIT or JSplitPane.VERTICAL\_SPLIT **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if orientation is not one of HORIZONTAL\_SPLIT or VERTICAL\_SPLIT.

### JSplitPane

public **JSplitPane**(int newOrientation,  
 boolean newContinuousLayout)

Creates a new JSplitPane with the specified orientation and redrawing style.

**Parameters:**newOrientation - JSplitPane.HORIZONTAL\_SPLIT or JSplitPane.VERTICAL\_SPLITnewContinuousLayout - a boolean, true for the components to redraw continuously as the divider changes position, false to wait until the divider position stops changing to redraw **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if orientation is not one of HORIZONTAL\_SPLIT or VERTICAL\_SPLIT

### JSplitPane

public **JSplitPane**(int newOrientation,  
 [Component](http://docs.google.com/java/awt/Component.html) newLeftComponent,  
 [Component](http://docs.google.com/java/awt/Component.html) newRightComponent)

Creates a new JSplitPane with the specified orientation and with the specified components that do not do continuous redrawing.

**Parameters:**newOrientation - JSplitPane.HORIZONTAL\_SPLIT or JSplitPane.VERTICAL\_SPLITnewLeftComponent - the Component that will appear on the left of a horizontally-split pane, or at the top of a vertically-split panenewRightComponent - the Component that will appear on the right of a horizontally-split pane, or at the bottom of a vertically-split pane **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if orientation is not one of: HORIZONTAL\_SPLIT or VERTICAL\_SPLIT

### JSplitPane

public **JSplitPane**(int newOrientation,  
 boolean newContinuousLayout,  
 [Component](http://docs.google.com/java/awt/Component.html) newLeftComponent,  
 [Component](http://docs.google.com/java/awt/Component.html) newRightComponent)

Creates a new JSplitPane with the specified orientation and redrawing style, and with the specified components.

**Parameters:**newOrientation - JSplitPane.HORIZONTAL\_SPLIT or JSplitPane.VERTICAL\_SPLITnewContinuousLayout - a boolean, true for the components to redraw continuously as the divider changes position, false to wait until the divider position stops changing to redrawnewLeftComponent - the Component that will appear on the left of a horizontally-split pane, or at the top of a vertically-split panenewRightComponent - the Component that will appear on the right of a horizontally-split pane, or at the bottom of a vertically-split pane **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if orientation is not one of HORIZONTAL\_SPLIT or VERTICAL\_SPLIT

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

### setUI

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

Sets the L&F object that renders this component.

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

### getUI

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

Returns the SplitPaneUI that is providing the current look and feel.

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

### updateUI

public void **updateUI**()

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

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

### getUIClassID

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

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

**Overrides:**[getUIClassID](http://docs.google.com/javax/swing/JComponent.html#getUIClassID()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Returns:**the string "SplitPaneUI"**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))

### setDividerSize

public void **setDividerSize**(int newSize)

Sets the size of the divider.

**Parameters:**newSize - an integer giving the size of the divider in pixels

### getDividerSize

public int **getDividerSize**()

Returns the size of the divider.

**Returns:**an integer giving the size of the divider in pixels

### setLeftComponent

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

Sets the component to the left (or above) the divider.

**Parameters:**comp - the Component to display in that position

### getLeftComponent

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

Returns the component to the left (or above) the divider.

**Returns:**the Component displayed in that position

### setTopComponent

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

Sets the component above, or to the left of the divider.

**Parameters:**comp - the Component to display in that position

### getTopComponent

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

Returns the component above, or to the left of the divider.

**Returns:**the Component displayed in that position

### setRightComponent

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

Sets the component to the right (or below) the divider.

**Parameters:**comp - the Component to display in that position

### getRightComponent

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

Returns the component to the right (or below) the divider.

**Returns:**the Component displayed in that position

### setBottomComponent

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

Sets the component below, or to the right of the divider.

**Parameters:**comp - the Component to display in that position

### getBottomComponent

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

Returns the component below, or to the right of the divider.

**Returns:**the Component displayed in that position

### setOneTouchExpandable

public void **setOneTouchExpandable**(boolean newValue)

Sets the value of the oneTouchExpandable property, which must be true for the JSplitPane to provide a UI widget on the divider to quickly expand/collapse the divider. The default value of this property is false. Some look and feels might not support one-touch expanding; they will ignore this property.

**Parameters:**newValue - true to specify that the split pane should provide a collapse/expand widget**See Also:**[isOneTouchExpandable()](http://docs.google.com/javax/swing/JSplitPane.html#isOneTouchExpandable())

### isOneTouchExpandable

public boolean **isOneTouchExpandable**()

Gets the oneTouchExpandable property.

**Returns:**the value of the oneTouchExpandable property**See Also:**[setOneTouchExpandable(boolean)](http://docs.google.com/javax/swing/JSplitPane.html#setOneTouchExpandable(boolean))

### setLastDividerLocation

public void **setLastDividerLocation**(int newLastLocation)

Sets the last location the divider was at to newLastLocation.

**Parameters:**newLastLocation - an integer specifying the last divider location in pixels, from the left (or upper) edge of the pane to the left (or upper) edge of the divider

### getLastDividerLocation

public int **getLastDividerLocation**()

Returns the last location the divider was at.

**Returns:**an integer specifying the last divider location as a count of pixels from the left (or upper) edge of the pane to the left (or upper) edge of the divider

### setOrientation

public void **setOrientation**(int orientation)

Sets the orientation, or how the splitter is divided. The options are:

* JSplitPane.VERTICAL\_SPLIT (above/below orientation of components)
* JSplitPane.HORIZONTAL\_SPLIT (left/right orientation of components)

**Parameters:**orientation - an integer specifying the orientation **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if orientation is not one of: HORIZONTAL\_SPLIT or VERTICAL\_SPLIT.

### getOrientation

public int **getOrientation**()

Returns the orientation.

**Returns:**an integer giving the orientation**See Also:**[setOrientation(int)](http://docs.google.com/javax/swing/JSplitPane.html#setOrientation(int))

### setContinuousLayout

public void **setContinuousLayout**(boolean newContinuousLayout)

Sets the value of the continuousLayout property, which must be true for the child components to be continuously redisplayed and laid out during user intervention. The default value of this property is false. Some look and feels might not support continuous layout; they will ignore this property.

**Parameters:**newContinuousLayout - true if the components should continuously be redrawn as the divider changes position**See Also:**[isContinuousLayout()](http://docs.google.com/javax/swing/JSplitPane.html#isContinuousLayout())

### isContinuousLayout

public boolean **isContinuousLayout**()

Gets the continuousLayout property.

**Returns:**the value of the continuousLayout property**See Also:**[setContinuousLayout(boolean)](http://docs.google.com/javax/swing/JSplitPane.html#setContinuousLayout(boolean))

### setResizeWeight

public void **setResizeWeight**(double value)

Specifies how to distribute extra space when the size of the split pane changes. A value of 0, the default, indicates the right/bottom component gets all the extra space (the left/top component acts fixed), where as a value of 1 specifies the left/top component gets all the extra space (the right/bottom component acts fixed). Specifically, the left/top component gets (weight \* diff) extra space and the right/bottom component gets (1 - weight) \* diff extra space.

**Parameters:**value - as described above **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if value is < 0 or > 1**Since:** 1.3

### getResizeWeight

public double **getResizeWeight**()

Returns the number that determines how extra space is distributed.

**Returns:**how extra space is to be distributed on a resize of the split pane**Since:** 1.3

### resetToPreferredSizes

public void **resetToPreferredSizes**()

Lays out the JSplitPane layout based on the preferred size of the children components. This will likely result in changing the divider location.

### setDividerLocation

public void **setDividerLocation**(double proportionalLocation)

Sets the divider location as a percentage of the JSplitPane's size.

This method is implemented in terms of setDividerLocation(int). This method immediately changes the size of the split pane based on its current size. If the split pane is not correctly realized and on screen, this method will have no effect (new divider location will become (current size \* proportionalLocation) which is 0).

**Parameters:**proportionalLocation - a double-precision floating point value that specifies a percentage, from zero (top/left) to 1.0 (bottom/right) **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the specified location is < 0 or > 1.0

### setDividerLocation

public void **setDividerLocation**(int location)

Sets the location of the divider. This is passed off to the look and feel implementation, and then listeners are notified. A value less than 0 implies the divider should be reset to a value that attempts to honor the preferred size of the left/top component. After notifying the listeners, the last divider location is updated, via setLastDividerLocation.

**Parameters:**location - an int specifying a UI-specific value (typically a pixel count)

### getDividerLocation

public int **getDividerLocation**()

Returns the last value passed to setDividerLocation. The value returned from this method may differ from the actual divider location (if setDividerLocation was passed a value bigger than the curent size).

**Returns:**an integer specifying the location of the divider

### getMinimumDividerLocation

public int **getMinimumDividerLocation**()

Returns the minimum location of the divider from the look and feel implementation.

**Returns:**an integer specifying a UI-specific value for the minimum location (typically a pixel count); or -1 if the UI is null

### getMaximumDividerLocation

public int **getMaximumDividerLocation**()

Returns the maximum location of the divider from the look and feel implementation.

**Returns:**an integer specifying a UI-specific value for the maximum location (typically a pixel count); or -1 if the UI is null

### remove

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

Removes the child component, component from the pane. Resets the leftComponent or rightComponent instance variable, as necessary.

**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**See Also:**[Container.add(java.awt.Component)](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)), [Container.validate()](http://docs.google.com/java/awt/Container.html#validate()), [Container.remove(int)](http://docs.google.com/java/awt/Container.html#remove(int))

### remove

public void **remove**(int index)

Removes the Component at the specified index. Updates the leftComponent and rightComponent instance variables as necessary, and then messages super.

**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 - an integer specifying the component to remove, where 1 specifies the left/top component and 2 specifies the bottom/right component**See Also:**[Container.add(java.awt.Component)](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)), [Container.validate()](http://docs.google.com/java/awt/Container.html#validate()), [Container.getComponentCount()](http://docs.google.com/java/awt/Container.html#getComponentCount())

### removeAll

public void **removeAll**()

Removes all the child components from the split pane. Resets the leftComonent and rightComponent instance variables.

**Overrides:**[removeAll](http://docs.google.com/java/awt/Container.html#removeAll()) in class [Container](http://docs.google.com/java/awt/Container.html) **See Also:**[Container.add(java.awt.Component)](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)), [Container.remove(int)](http://docs.google.com/java/awt/Container.html#remove(int))

### isValidateRoot

public boolean **isValidateRoot**()

Returns true, so that calls to revalidate on any descendant of this JSplitPane will cause a request to be queued that will validate the JSplitPane and all its descendants.

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

### addImpl

protected void **addImpl**([Component](http://docs.google.com/java/awt/Component.html) comp,  
 [Object](http://docs.google.com/java/lang/Object.html) constraints,  
 int index)

Adds the specified component to this split pane. If constraints identifies the left/top or right/bottom child component, and a component with that identifier was previously added, it will be removed and then comp will be added in its place. If constraints is not one of the known identifiers the layout manager may throw an IllegalArgumentException.

The possible constraints objects (Strings) are:

* JSplitPane.TOP
* JSplitPane.LEFT
* JSplitPane.BOTTOM
* JSplitPane.RIGHT

If the constraints object is null, the component is added in the first available position (left/top if open, else right/bottom).

**Overrides:**[addImpl](http://docs.google.com/java/awt/Container.html#addImpl(java.awt.Component,%20java.lang.Object,%20int)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**comp - the component to addconstraints - an Object specifying the layout constraints (position) for this componentindex - an integer specifying the index in the container's list. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the constraints object does not match an existing component**See Also:**[Container.addImpl(Component, Object, int)](http://docs.google.com/java/awt/Container.html#addImpl(java.awt.Component,%20java.lang.Object,%20int))

### paintChildren

protected void **paintChildren**([Graphics](http://docs.google.com/java/awt/Graphics.html) g)

Subclassed to message the UI with finishedPaintingChildren after super has been messaged, as well as painting the border.

**Overrides:**[paintChildren](http://docs.google.com/javax/swing/JComponent.html#paintChildren(java.awt.Graphics)) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Parameters:**g - the Graphics context within which to paint**See Also:**[JComponent.paint(java.awt.Graphics)](http://docs.google.com/javax/swing/JComponent.html#paint(java.awt.Graphics)), [Container.paint(java.awt.Graphics)](http://docs.google.com/java/awt/Container.html#paint(java.awt.Graphics))

### paramString

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

Returns a string representation of this JSplitPane. 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 JSplitPane.

### getAccessibleContext

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

Gets the AccessibleContext associated with this JSplitPane. For split panes, the AccessibleContext takes the form of an AccessibleJSplitPane. A new AccessibleJSplitPane 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 AccessibleJSplitPane that serves as the AccessibleContext of this JSplitPane

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JSplitPane.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/JSpinner.NumberEditor.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JSplitPane.AccessibleJSplitPane.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JSplitPane.html)    [**NO FRAMES**](http://docs.google.com/JSplitPane.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#1t3h5sf) | [CONSTR](#3rdcrjn) | [METHOD](#26in1rg) | DETAIL: [FIELD](#2jxsxqh) | [CONSTR](#vx1227) | [METHOD](#3tbugp1) |

[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).