| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AbstractButton.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/AbstractAction.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/AbstractButton.AccessibleAbstractButton.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/AbstractButton.html)    [**NO FRAMES**](http://docs.google.com/AbstractButton.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#lnxbz9) | DETAIL: [FIELD](#z337ya) | [CONSTR](#2u6wntf) | [METHOD](#3tbugp1) |

## **javax.swing**

Class AbstractButton

[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.AbstractButton**

**All Implemented Interfaces:** [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html), [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html), [MenuContainer](http://docs.google.com/java/awt/MenuContainer.html), [Serializable](http://docs.google.com/java/io/Serializable.html), [SwingConstants](http://docs.google.com/javax/swing/SwingConstants.html) **Direct Known Subclasses:** [JButton](http://docs.google.com/javax/swing/JButton.html), [JMenuItem](http://docs.google.com/javax/swing/JMenuItem.html), [JToggleButton](http://docs.google.com/javax/swing/JToggleButton.html)

public abstract class **AbstractButton**extends [JComponent](http://docs.google.com/javax/swing/JComponent.html)implements [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html), [SwingConstants](http://docs.google.com/javax/swing/SwingConstants.html)

Defines common behaviors for buttons and menu items.

Buttons can be configured, and to some degree controlled, by [Action](http://docs.google.com/Action.html)s. Using an Action with a button has many benefits beyond directly configuring a button. Refer to  [Swing Components Supporting Action](http://docs.google.com/Action.html#buttonActions) for more details, and you can find more information in [How to Use Actions](http://java.sun.com/docs/books/tutorial/uiswing/misc/action.html), a section in *The Java Tutorial*.

For further information see [How to Use Buttons, Check Boxes, and Radio Buttons](http://java.sun.com/docs/books/tutorial/uiswing/components/button.html), a section in *The Java Tutorial*.

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

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**AbstractButton.AccessibleAbstractButton**](http://docs.google.com/javax/swing/AbstractButton.AccessibleAbstractButton.html)            This class implements accessibility support for the AbstractButton class. |
| protected  class | [**AbstractButton.ButtonChangeListener**](http://docs.google.com/javax/swing/AbstractButton.ButtonChangeListener.html)            Extends ChangeListener to be serializable. |

| **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  [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) | [**actionListener**](http://docs.google.com/javax/swing/AbstractButton.html#actionListener)            The button model's ActionListener. |
| static [String](http://docs.google.com/java/lang/String.html) | [**BORDER\_PAINTED\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#BORDER_PAINTED_CHANGED_PROPERTY)            Identifies a change to having the border drawn, or having it not drawn. |
| protected  [ChangeEvent](http://docs.google.com/javax/swing/event/ChangeEvent.html) | [**changeEvent**](http://docs.google.com/javax/swing/AbstractButton.html#changeEvent)            Only one ChangeEvent is needed per button instance since the event's only state is the source property. |
| protected  [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) | [**changeListener**](http://docs.google.com/javax/swing/AbstractButton.html#changeListener)            The button model's changeListener. |
| static [String](http://docs.google.com/java/lang/String.html) | [**CONTENT\_AREA\_FILLED\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#CONTENT_AREA_FILLED_CHANGED_PROPERTY)            Identifies a change to having the button paint the content area. |
| static [String](http://docs.google.com/java/lang/String.html) | [**DISABLED\_ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#DISABLED_ICON_CHANGED_PROPERTY)            Identifies a change to the icon used when the button has been disabled. |
| static [String](http://docs.google.com/java/lang/String.html) | [**DISABLED\_SELECTED\_ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#DISABLED_SELECTED_ICON_CHANGED_PROPERTY)            Identifies a change to the icon used when the button has been disabled and selected. |
| static [String](http://docs.google.com/java/lang/String.html) | [**FOCUS\_PAINTED\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#FOCUS_PAINTED_CHANGED_PROPERTY)            Identifies a change to having the border highlighted when focused, or not. |
| static [String](http://docs.google.com/java/lang/String.html) | [**HORIZONTAL\_ALIGNMENT\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#HORIZONTAL_ALIGNMENT_CHANGED_PROPERTY)            Identifies a change in the button's horizontal alignment. |
| static [String](http://docs.google.com/java/lang/String.html) | [**HORIZONTAL\_TEXT\_POSITION\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#HORIZONTAL_TEXT_POSITION_CHANGED_PROPERTY)            Identifies a change in the button's horizontal text position. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#ICON_CHANGED_PROPERTY)            Identifies a change to the icon that represents the button. |
| protected  [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) | [**itemListener**](http://docs.google.com/javax/swing/AbstractButton.html#itemListener)            The button model's ItemListener. |
| static [String](http://docs.google.com/java/lang/String.html) | [**MARGIN\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#MARGIN_CHANGED_PROPERTY)            Identifies a change in the button's margins. |
| static [String](http://docs.google.com/java/lang/String.html) | [**MNEMONIC\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#MNEMONIC_CHANGED_PROPERTY)            Identifies a change to the button's mnemonic. |
| protected  [ButtonModel](http://docs.google.com/javax/swing/ButtonModel.html) | [**model**](http://docs.google.com/javax/swing/AbstractButton.html#model)            The data model that determines the button's state. |
| static [String](http://docs.google.com/java/lang/String.html) | [**MODEL\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#MODEL_CHANGED_PROPERTY)            Identifies a change in the button model. |
| static [String](http://docs.google.com/java/lang/String.html) | [**PRESSED\_ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#PRESSED_ICON_CHANGED_PROPERTY)            Identifies a change to the icon used when the button has been pressed. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ROLLOVER\_ENABLED\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#ROLLOVER_ENABLED_CHANGED_PROPERTY)            Identifies a change from rollover enabled to disabled or back to enabled. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ROLLOVER\_ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#ROLLOVER_ICON_CHANGED_PROPERTY)            Identifies a change to the icon used when the cursor is over the button. |
| static [String](http://docs.google.com/java/lang/String.html) | [**ROLLOVER\_SELECTED\_ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#ROLLOVER_SELECTED_ICON_CHANGED_PROPERTY)            Identifies a change to the icon used when the cursor is over the button and it has been selected. |
| static [String](http://docs.google.com/java/lang/String.html) | [**SELECTED\_ICON\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#SELECTED_ICON_CHANGED_PROPERTY)            Identifies a change to the icon used when the button has been selected. |
| static [String](http://docs.google.com/java/lang/String.html) | [**TEXT\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#TEXT_CHANGED_PROPERTY)            Identifies a change in the button's text. |
| static [String](http://docs.google.com/java/lang/String.html) | [**VERTICAL\_ALIGNMENT\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#VERTICAL_ALIGNMENT_CHANGED_PROPERTY)            Identifies a change in the button's vertical alignment. |
| static [String](http://docs.google.com/java/lang/String.html) | [**VERTICAL\_TEXT\_POSITION\_CHANGED\_PROPERTY**](http://docs.google.com/javax/swing/AbstractButton.html#VERTICAL_TEXT_POSITION_CHANGED_PROPERTY)            Identifies a change in the button's vertical text position. |

| **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** | |
| --- | --- |
| [**AbstractButton**](http://docs.google.com/javax/swing/AbstractButton.html#AbstractButton())() |

| **Method Summary** | |
| --- | --- |
| protected  void | [**actionPropertyChanged**](http://docs.google.com/javax/swing/AbstractButton.html#actionPropertyChanged(javax.swing.Action,%20java.lang.String))([Action](http://docs.google.com/javax/swing/Action.html) action, [String](http://docs.google.com/java/lang/String.html) propertyName)            Updates the button's state in response to property changes in the associated action. |
| void | [**addActionListener**](http://docs.google.com/javax/swing/AbstractButton.html#addActionListener(java.awt.event.ActionListener))([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)            Adds an ActionListener to the button. |
| void | [**addChangeListener**](http://docs.google.com/javax/swing/AbstractButton.html#addChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)            Adds a ChangeListener to the button. |
| protected  void | [**addImpl**](http://docs.google.com/javax/swing/AbstractButton.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 container at the specified index, refer to [Container.addImpl(Component, Object, int)](http://docs.google.com/java/awt/Container.html#addImpl(java.awt.Component,%20java.lang.Object,%20int)) for a complete description of this method. |
| void | [**addItemListener**](http://docs.google.com/javax/swing/AbstractButton.html#addItemListener(java.awt.event.ItemListener))([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)            Adds an ItemListener to the checkbox. |
| protected  int | [**checkHorizontalKey**](http://docs.google.com/javax/swing/AbstractButton.html#checkHorizontalKey(int,%20java.lang.String))(int key, [String](http://docs.google.com/java/lang/String.html) exception)            Verify that the key argument is a legal value for the horizontalAlignment and horizontalTextPosition properties. |
| protected  int | [**checkVerticalKey**](http://docs.google.com/javax/swing/AbstractButton.html#checkVerticalKey(int,%20java.lang.String))(int key, [String](http://docs.google.com/java/lang/String.html) exception)            Verify that the key argument is a legal value for the vertical properties. |
| protected  void | [**configurePropertiesFromAction**](http://docs.google.com/javax/swing/AbstractButton.html#configurePropertiesFromAction(javax.swing.Action))([Action](http://docs.google.com/javax/swing/Action.html) a)            Sets the properties on this button to match those in the specified Action. |
| protected  [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) | [**createActionListener**](http://docs.google.com/javax/swing/AbstractButton.html#createActionListener())() |
| protected  [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) | [**createActionPropertyChangeListener**](http://docs.google.com/javax/swing/AbstractButton.html#createActionPropertyChangeListener(javax.swing.Action))([Action](http://docs.google.com/javax/swing/Action.html) a)            Creates and returns a PropertyChangeListener that is responsible for listening for changes from the specified Action and updating the appropriate properties. |
| protected  [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) | [**createChangeListener**](http://docs.google.com/javax/swing/AbstractButton.html#createChangeListener())()            Subclasses that want to handle ChangeEvents differently can override this to return another ChangeListener implementation. |
| protected  [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) | [**createItemListener**](http://docs.google.com/javax/swing/AbstractButton.html#createItemListener())() |
| void | [**doClick**](http://docs.google.com/javax/swing/AbstractButton.html#doClick())()            Programmatically perform a "click". |
| void | [**doClick**](http://docs.google.com/javax/swing/AbstractButton.html#doClick(int))(int pressTime)            Programmatically perform a "click". |
| protected  void | [**fireActionPerformed**](http://docs.google.com/javax/swing/AbstractButton.html#fireActionPerformed(java.awt.event.ActionEvent))([ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html) event)            Notifies all listeners that have registered interest for notification on this event type. |
| protected  void | [**fireItemStateChanged**](http://docs.google.com/javax/swing/AbstractButton.html#fireItemStateChanged(java.awt.event.ItemEvent))([ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html) event)            Notifies all listeners that have registered interest for notification on this event type. |
| protected  void | [**fireStateChanged**](http://docs.google.com/javax/swing/AbstractButton.html#fireStateChanged())()            Notifies all listeners that have registered interest for notification on this event type. |
| [Action](http://docs.google.com/javax/swing/Action.html) | [**getAction**](http://docs.google.com/javax/swing/AbstractButton.html#getAction())()            Returns the currently set Action for this ActionEvent source, or null if no Action is set. |
| [String](http://docs.google.com/java/lang/String.html) | [**getActionCommand**](http://docs.google.com/javax/swing/AbstractButton.html#getActionCommand())()            Returns the action command for this button. |
| [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)[] | [**getActionListeners**](http://docs.google.com/javax/swing/AbstractButton.html#getActionListeners())()            Returns an array of all the ActionListeners added to this AbstractButton with addActionListener(). |
| [ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html)[] | [**getChangeListeners**](http://docs.google.com/javax/swing/AbstractButton.html#getChangeListeners())()            Returns an array of all the ChangeListeners added to this AbstractButton with addChangeListener(). |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getDisabledIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getDisabledIcon())()            Returns the icon used by the button when it's disabled. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getDisabledSelectedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getDisabledSelectedIcon())()            Returns the icon used by the button when it's disabled and selected. |
| int | [**getDisplayedMnemonicIndex**](http://docs.google.com/javax/swing/AbstractButton.html#getDisplayedMnemonicIndex())()            Returns the character, as an index, that the look and feel should provide decoration for as representing the mnemonic character. |
| boolean | [**getHideActionText**](http://docs.google.com/javax/swing/AbstractButton.html#getHideActionText())()            Returns the value of the hideActionText property, which determines whether the button displays text from the Action. |
| int | [**getHorizontalAlignment**](http://docs.google.com/javax/swing/AbstractButton.html#getHorizontalAlignment())()            Returns the horizontal alignment of the icon and text. |
| int | [**getHorizontalTextPosition**](http://docs.google.com/javax/swing/AbstractButton.html#getHorizontalTextPosition())()            Returns the horizontal position of the text relative to the icon. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getIcon())()            Returns the default icon. |
| int | [**getIconTextGap**](http://docs.google.com/javax/swing/AbstractButton.html#getIconTextGap())()            Returns the amount of space between the text and the icon displayed in this button. |
| [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)[] | [**getItemListeners**](http://docs.google.com/javax/swing/AbstractButton.html#getItemListeners())()            Returns an array of all the ItemListeners added to this AbstractButton with addItemListener(). |
| [String](http://docs.google.com/java/lang/String.html) | [**getLabel**](http://docs.google.com/javax/swing/AbstractButton.html#getLabel())()  **Deprecated.** *- Replaced by getText* |
| [Insets](http://docs.google.com/java/awt/Insets.html) | [**getMargin**](http://docs.google.com/javax/swing/AbstractButton.html#getMargin())()            Returns the margin between the button's border and the label. |
| int | [**getMnemonic**](http://docs.google.com/javax/swing/AbstractButton.html#getMnemonic())()            Returns the keyboard mnemonic from the the current model. |
| [ButtonModel](http://docs.google.com/javax/swing/ButtonModel.html) | [**getModel**](http://docs.google.com/javax/swing/AbstractButton.html#getModel())()            Returns the model that this button represents. |
| long | [**getMultiClickThreshhold**](http://docs.google.com/javax/swing/AbstractButton.html#getMultiClickThreshhold())()            Gets the amount of time (in milliseconds) required between mouse press events for the button to generate the corresponding action events. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getPressedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getPressedIcon())()            Returns the pressed icon for the button. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getRolloverIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getRolloverIcon())()            Returns the rollover icon for the button. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getRolloverSelectedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getRolloverSelectedIcon())()            Returns the rollover selection icon for the button. |
| [Icon](http://docs.google.com/javax/swing/Icon.html) | [**getSelectedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#getSelectedIcon())()            Returns the selected icon for the button. |
| [Object](http://docs.google.com/java/lang/Object.html)[] | [**getSelectedObjects**](http://docs.google.com/javax/swing/AbstractButton.html#getSelectedObjects())()            Returns an array (length 1) containing the label or null if the button is not selected. |
| [String](http://docs.google.com/java/lang/String.html) | [**getText**](http://docs.google.com/javax/swing/AbstractButton.html#getText())()            Returns the button's text. |
| [ButtonUI](http://docs.google.com/javax/swing/plaf/ButtonUI.html) | [**getUI**](http://docs.google.com/javax/swing/AbstractButton.html#getUI())()            Returns the L&F object that renders this component. |
| int | [**getVerticalAlignment**](http://docs.google.com/javax/swing/AbstractButton.html#getVerticalAlignment())()            Returns the vertical alignment of the text and icon. |
| int | [**getVerticalTextPosition**](http://docs.google.com/javax/swing/AbstractButton.html#getVerticalTextPosition())()            Returns the vertical position of the text relative to the icon. |
| boolean | [**imageUpdate**](http://docs.google.com/javax/swing/AbstractButton.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int))([Image](http://docs.google.com/java/awt/Image.html) img, int infoflags, int x, int y, int w, int h)            This is overridden to return false if the current Icon's Image is not equal to the passed in Image img. |
| protected  void | [**init**](http://docs.google.com/javax/swing/AbstractButton.html#init(java.lang.String,%20javax.swing.Icon))([String](http://docs.google.com/java/lang/String.html) text, [Icon](http://docs.google.com/javax/swing/Icon.html) icon) |
| boolean | [**isBorderPainted**](http://docs.google.com/javax/swing/AbstractButton.html#isBorderPainted())()            Gets the borderPainted property. |
| boolean | [**isContentAreaFilled**](http://docs.google.com/javax/swing/AbstractButton.html#isContentAreaFilled())()            Gets the contentAreaFilled property. |
| boolean | [**isFocusPainted**](http://docs.google.com/javax/swing/AbstractButton.html#isFocusPainted())()            Gets the paintFocus property. |
| boolean | [**isRolloverEnabled**](http://docs.google.com/javax/swing/AbstractButton.html#isRolloverEnabled())()            Gets the rolloverEnabled property. |
| boolean | [**isSelected**](http://docs.google.com/javax/swing/AbstractButton.html#isSelected())()            Returns the state of the button. |
| protected  void | [**paintBorder**](http://docs.google.com/javax/swing/AbstractButton.html#paintBorder(java.awt.Graphics))([Graphics](http://docs.google.com/java/awt/Graphics.html) g)            Paint the button's border if BorderPainted property is true and the button has a border. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**paramString**](http://docs.google.com/javax/swing/AbstractButton.html#paramString())()            Returns a string representation of this AbstractButton. |
| void | [**removeActionListener**](http://docs.google.com/javax/swing/AbstractButton.html#removeActionListener(java.awt.event.ActionListener))([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)            Removes an ActionListener from the button. |
| void | [**removeChangeListener**](http://docs.google.com/javax/swing/AbstractButton.html#removeChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) l)            Removes a ChangeListener from the button. |
| void | [**removeItemListener**](http://docs.google.com/javax/swing/AbstractButton.html#removeItemListener(java.awt.event.ItemListener))([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)            Removes an ItemListener from the button. |
| void | [**removeNotify**](http://docs.google.com/javax/swing/AbstractButton.html#removeNotify())()            Notifies this component that it no longer has a parent component. |
| void | [**setAction**](http://docs.google.com/javax/swing/AbstractButton.html#setAction(javax.swing.Action))([Action](http://docs.google.com/javax/swing/Action.html) a)            Sets the Action. |
| void | [**setActionCommand**](http://docs.google.com/javax/swing/AbstractButton.html#setActionCommand(java.lang.String))([String](http://docs.google.com/java/lang/String.html) actionCommand)            Sets the action command for this button. |
| void | [**setBorderPainted**](http://docs.google.com/javax/swing/AbstractButton.html#setBorderPainted(boolean))(boolean b)            Sets the borderPainted property. |
| void | [**setContentAreaFilled**](http://docs.google.com/javax/swing/AbstractButton.html#setContentAreaFilled(boolean))(boolean b)            Sets the contentAreaFilled property. |
| void | [**setDisabledIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setDisabledIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) disabledIcon)            Sets the disabled icon for the button. |
| void | [**setDisabledSelectedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setDisabledSelectedIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) disabledSelectedIcon)            Sets the disabled selection icon for the button. |
| void | [**setDisplayedMnemonicIndex**](http://docs.google.com/javax/swing/AbstractButton.html#setDisplayedMnemonicIndex(int))(int index)            Provides a hint to the look and feel as to which character in the text should be decorated to represent the mnemonic. |
| void | [**setEnabled**](http://docs.google.com/javax/swing/AbstractButton.html#setEnabled(boolean))(boolean b)            Enables (or disables) the button. |
| void | [**setFocusPainted**](http://docs.google.com/javax/swing/AbstractButton.html#setFocusPainted(boolean))(boolean b)            Sets the paintFocus property, which must be true for the focus state to be painted. |
| void | [**setHideActionText**](http://docs.google.com/javax/swing/AbstractButton.html#setHideActionText(boolean))(boolean hideActionText)            Sets the hideActionText property, which determines whether the button displays text from the Action. |
| void | [**setHorizontalAlignment**](http://docs.google.com/javax/swing/AbstractButton.html#setHorizontalAlignment(int))(int alignment)            Sets the horizontal alignment of the icon and text. |
| void | [**setHorizontalTextPosition**](http://docs.google.com/javax/swing/AbstractButton.html#setHorizontalTextPosition(int))(int textPosition)            Sets the horizontal position of the text relative to the icon. |
| void | [**setIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) defaultIcon)            Sets the button's default icon. |
| void | [**setIconTextGap**](http://docs.google.com/javax/swing/AbstractButton.html#setIconTextGap(int))(int iconTextGap)            If both the icon and text properties are set, this property defines the space between them. |
| void | [**setLabel**](http://docs.google.com/javax/swing/AbstractButton.html#setLabel(java.lang.String))([String](http://docs.google.com/java/lang/String.html) label)  **Deprecated.** *- Replaced by setText(text)* |
| void | [**setLayout**](http://docs.google.com/javax/swing/AbstractButton.html#setLayout(java.awt.LayoutManager))([LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) mgr)            Sets the layout manager for this container, refer to [Container.setLayout(LayoutManager)](http://docs.google.com/java/awt/Container.html#setLayout(java.awt.LayoutManager)) for a complete description of this method. |
| void | [**setMargin**](http://docs.google.com/javax/swing/AbstractButton.html#setMargin(java.awt.Insets))([Insets](http://docs.google.com/java/awt/Insets.html) m)            Sets space for margin between the button's border and the label. |
| void | [**setMnemonic**](http://docs.google.com/javax/swing/AbstractButton.html#setMnemonic(char))(char mnemonic)            This method is now obsolete, please use setMnemonic(int) to set the mnemonic for a button. |
| void | [**setMnemonic**](http://docs.google.com/javax/swing/AbstractButton.html#setMnemonic(int))(int mnemonic)            Sets the keyboard mnemonic on the current model. |
| void | [**setModel**](http://docs.google.com/javax/swing/AbstractButton.html#setModel(javax.swing.ButtonModel))([ButtonModel](http://docs.google.com/javax/swing/ButtonModel.html) newModel)            Sets the model that this button represents. |
| void | [**setMultiClickThreshhold**](http://docs.google.com/javax/swing/AbstractButton.html#setMultiClickThreshhold(long))(long threshhold)            Sets the amount of time (in milliseconds) required between mouse press events for the button to generate the corresponding action events. |
| void | [**setPressedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setPressedIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) pressedIcon)            Sets the pressed icon for the button. |
| void | [**setRolloverEnabled**](http://docs.google.com/javax/swing/AbstractButton.html#setRolloverEnabled(boolean))(boolean b)            Sets the rolloverEnabled property, which must be true for rollover effects to occur. |
| void | [**setRolloverIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setRolloverIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) rolloverIcon)            Sets the rollover icon for the button. |
| void | [**setRolloverSelectedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setRolloverSelectedIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) rolloverSelectedIcon)            Sets the rollover selected icon for the button. |
| void | [**setSelected**](http://docs.google.com/javax/swing/AbstractButton.html#setSelected(boolean))(boolean b)            Sets the state of the button. |
| void | [**setSelectedIcon**](http://docs.google.com/javax/swing/AbstractButton.html#setSelectedIcon(javax.swing.Icon))([Icon](http://docs.google.com/javax/swing/Icon.html) selectedIcon)            Sets the selected icon for the button. |
| void | [**setText**](http://docs.google.com/javax/swing/AbstractButton.html#setText(java.lang.String))([String](http://docs.google.com/java/lang/String.html) text)            Sets the button's text. |
| void | [**setUI**](http://docs.google.com/javax/swing/AbstractButton.html#setUI(javax.swing.plaf.ButtonUI))([ButtonUI](http://docs.google.com/javax/swing/plaf/ButtonUI.html) ui)            Sets the L&F object that renders this component. |
| void | [**setVerticalAlignment**](http://docs.google.com/javax/swing/AbstractButton.html#setVerticalAlignment(int))(int alignment)            Sets the vertical alignment of the icon and text. |
| void | [**setVerticalTextPosition**](http://docs.google.com/javax/swing/AbstractButton.html#setVerticalTextPosition(int))(int textPosition)            Sets the vertical position of the text relative to the icon. |
| void | [**updateUI**](http://docs.google.com/javax/swing/AbstractButton.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)), [getAccessibleContext](http://docs.google.com/javax/swing/JComponent.html#getAccessibleContext()), [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()), [getUIClassID](http://docs.google.com/javax/swing/JComponent.html#getUIClassID()), [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)), [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)), [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)), [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)), [remove](http://docs.google.com/java/awt/Container.html#remove(java.awt.Component)), [remove](http://docs.google.com/java/awt/Container.html#remove(int)), [removeAll](http://docs.google.com/java/awt/Container.html#removeAll()), [removeContainerListener](http://docs.google.com/java/awt/Container.html#removeContainerListener(java.awt.event.ContainerListener)), [setComponentZOrder](http://docs.google.com/java/awt/Container.html#setComponentZOrder(java.awt.Component,%20int)), [setFocusCycleRoot](http://docs.google.com/java/awt/Container.html#setFocusCycleRoot(boolean)), [setFocusTraversalPolicy](http://docs.google.com/java/awt/Container.html#setFocusTraversalPolicy(java.awt.FocusTraversalPolicy)), [setFocusTraversalPolicyProvider](http://docs.google.com/java/awt/Container.html#setFocusTraversalPolicyProvider(boolean)), [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()), [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** |
| --- |

### MODEL\_CHANGED\_PROPERTY

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

Identifies a change in the button model.

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

### TEXT\_CHANGED\_PROPERTY

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

Identifies a change in the button's text.

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

### MNEMONIC\_CHANGED\_PROPERTY

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

Identifies a change to the button's mnemonic.

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

### MARGIN\_CHANGED\_PROPERTY

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

Identifies a change in the button's margins.

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

### VERTICAL\_ALIGNMENT\_CHANGED\_PROPERTY

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

Identifies a change in the button's vertical alignment.

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

### HORIZONTAL\_ALIGNMENT\_CHANGED\_PROPERTY

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

Identifies a change in the button's horizontal alignment.

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

### VERTICAL\_TEXT\_POSITION\_CHANGED\_PROPERTY

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

Identifies a change in the button's vertical text position.

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

### HORIZONTAL\_TEXT\_POSITION\_CHANGED\_PROPERTY

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

Identifies a change in the button's horizontal text position.

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

### BORDER\_PAINTED\_CHANGED\_PROPERTY

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

Identifies a change to having the border drawn, or having it not drawn.

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

### FOCUS\_PAINTED\_CHANGED\_PROPERTY

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

Identifies a change to having the border highlighted when focused, or not.

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

### ROLLOVER\_ENABLED\_CHANGED\_PROPERTY

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

Identifies a change from rollover enabled to disabled or back to enabled.

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

### CONTENT\_AREA\_FILLED\_CHANGED\_PROPERTY

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

Identifies a change to having the button paint the content area.

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

### ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon that represents the button.

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

### PRESSED\_ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon used when the button has been pressed.

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

### SELECTED\_ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon used when the button has been selected.

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

### ROLLOVER\_ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon used when the cursor is over the button.

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

### ROLLOVER\_SELECTED\_ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon used when the cursor is over the button and it has been selected.

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

### DISABLED\_ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon used when the button has been disabled.

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

### DISABLED\_SELECTED\_ICON\_CHANGED\_PROPERTY

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

Identifies a change to the icon used when the button has been disabled and selected.

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

### model

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

The data model that determines the button's state.

### changeListener

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

The button model's changeListener.

### actionListener

protected [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) **actionListener**

The button model's ActionListener.

### itemListener

protected [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) **itemListener**

The button model's ItemListener.

### changeEvent

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

Only one ChangeEvent is needed per button instance since the event's only state is the source property. The source of events generated is always "this".

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

### AbstractButton

public **AbstractButton**()

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

### setHideActionText

public void **setHideActionText**(boolean hideActionText)

Sets the hideActionText property, which determines whether the button displays text from the Action. This is useful only if an Action has been installed on the button.

**Parameters:**hideActionText - true if the button's text property should not reflect that of the Action; the default is false**Since:** 1.6 **See Also:**[Swing Components Supporting Action](http://docs.google.com/Action.html#buttonActions)

### getHideActionText

public boolean **getHideActionText**()

Returns the value of the hideActionText property, which determines whether the button displays text from the Action. This is useful only if an Action has been installed on the button.

**Returns:**true if the button's text property should not reflect that of the Action; the default is false**Since:** 1.6

### getText

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

Returns the button's text.

**Returns:**the buttons text**See Also:**[setText(java.lang.String)](http://docs.google.com/javax/swing/AbstractButton.html#setText(java.lang.String))

### setText

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

Sets the button's text.

**Parameters:**text - the string used to set the text**See Also:**[getText()](http://docs.google.com/javax/swing/AbstractButton.html#getText())

### isSelected

public boolean **isSelected**()

Returns the state of the button. True if the toggle button is selected, false if it's not.

**Returns:**true if the toggle button is selected, otherwise false

### setSelected

public void **setSelected**(boolean b)

Sets the state of the button. Note that this method does not trigger an actionEvent. Call doClick to perform a programatic action change.

**Parameters:**b - true if the button is selected, otherwise false

### doClick

public void **doClick**()

Programmatically perform a "click". This does the same thing as if the user had pressed and released the button.

### doClick

public void **doClick**(int pressTime)

Programmatically perform a "click". This does the same thing as if the user had pressed and released the button. The button stays visually "pressed" for pressTime milliseconds.

**Parameters:**pressTime - the time to "hold down" the button, in milliseconds

### setMargin

public void **setMargin**([Insets](http://docs.google.com/java/awt/Insets.html) m)

Sets space for margin between the button's border and the label. Setting to null will cause the button to use the default margin. The button's default Border object will use this value to create the proper margin. However, if a non-default border is set on the button, it is that Border object's responsibility to create the appropriate margin space (else this property will effectively be ignored).

**Parameters:**m - the space between the border and the label

### getMargin

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

Returns the margin between the button's border and the label.

**Returns:**an Insets object specifying the margin between the botton's border and the label**See Also:**[setMargin(java.awt.Insets)](http://docs.google.com/javax/swing/AbstractButton.html#setMargin(java.awt.Insets))

### getIcon

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

Returns the default icon.

**Returns:**the default Icon**See Also:**[setIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setIcon(javax.swing.Icon))

### setIcon

public void **setIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) defaultIcon)

Sets the button's default icon. This icon is also used as the "pressed" and "disabled" icon if there is no explicitly set pressed icon.

**Parameters:**defaultIcon - the icon used as the default image**See Also:**[getIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getIcon()), [setPressedIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setPressedIcon(javax.swing.Icon))

### getPressedIcon

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

Returns the pressed icon for the button.

**Returns:**the pressedIcon property**See Also:**[setPressedIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setPressedIcon(javax.swing.Icon))

### setPressedIcon

public void **setPressedIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) pressedIcon)

Sets the pressed icon for the button.

**Parameters:**pressedIcon - the icon used as the "pressed" image**See Also:**[getPressedIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getPressedIcon())

### getSelectedIcon

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

Returns the selected icon for the button.

**Returns:**the selectedIcon property**See Also:**[setSelectedIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setSelectedIcon(javax.swing.Icon))

### setSelectedIcon

public void **setSelectedIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) selectedIcon)

Sets the selected icon for the button.

**Parameters:**selectedIcon - the icon used as the "selected" image**See Also:**[getSelectedIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getSelectedIcon())

### getRolloverIcon

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

Returns the rollover icon for the button.

**Returns:**the rolloverIcon property**See Also:**[setRolloverIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setRolloverIcon(javax.swing.Icon))

### setRolloverIcon

public void **setRolloverIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) rolloverIcon)

Sets the rollover icon for the button.

**Parameters:**rolloverIcon - the icon used as the "rollover" image**See Also:**[getRolloverIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getRolloverIcon())

### getRolloverSelectedIcon

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

Returns the rollover selection icon for the button.

**Returns:**the rolloverSelectedIcon property**See Also:**[setRolloverSelectedIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setRolloverSelectedIcon(javax.swing.Icon))

### setRolloverSelectedIcon

public void **setRolloverSelectedIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) rolloverSelectedIcon)

Sets the rollover selected icon for the button.

**Parameters:**rolloverSelectedIcon - the icon used as the "selected rollover" image**See Also:**[getRolloverSelectedIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getRolloverSelectedIcon())

### getDisabledIcon

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

Returns the icon used by the button when it's disabled. If no disabled icon has been set this will forward the call to the look and feel to construct an appropriate disabled Icon.

Some look and feels might not render the disabled Icon, in which case they will ignore this.

**Returns:**the disabledIcon property**See Also:**[getPressedIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getPressedIcon()), [setDisabledIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setDisabledIcon(javax.swing.Icon)), [LookAndFeel.getDisabledIcon(javax.swing.JComponent, javax.swing.Icon)](http://docs.google.com/javax/swing/LookAndFeel.html#getDisabledIcon(javax.swing.JComponent,%20javax.swing.Icon))

### setDisabledIcon

public void **setDisabledIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) disabledIcon)

Sets the disabled icon for the button.

**Parameters:**disabledIcon - the icon used as the disabled image**See Also:**[getDisabledIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getDisabledIcon())

### getDisabledSelectedIcon

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

Returns the icon used by the button when it's disabled and selected. If no disabled selection icon has been set, this will forward the call to the LookAndFeel to construct an appropriate disabled Icon from the selection icon if it has been set and to getDisabledIcon() otherwise.

Some look and feels might not render the disabled selected Icon, in which case they will ignore this.

**Returns:**the disabledSelectedIcon property**See Also:**[getDisabledIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getDisabledIcon()), [setDisabledSelectedIcon(javax.swing.Icon)](http://docs.google.com/javax/swing/AbstractButton.html#setDisabledSelectedIcon(javax.swing.Icon)), [LookAndFeel.getDisabledSelectedIcon(javax.swing.JComponent, javax.swing.Icon)](http://docs.google.com/javax/swing/LookAndFeel.html#getDisabledSelectedIcon(javax.swing.JComponent,%20javax.swing.Icon))

### setDisabledSelectedIcon

public void **setDisabledSelectedIcon**([Icon](http://docs.google.com/javax/swing/Icon.html) disabledSelectedIcon)

Sets the disabled selection icon for the button.

**Parameters:**disabledSelectedIcon - the icon used as the disabled selection image**See Also:**[getDisabledSelectedIcon()](http://docs.google.com/javax/swing/AbstractButton.html#getDisabledSelectedIcon())

### getVerticalAlignment

public int **getVerticalAlignment**()

Returns the vertical alignment of the text and icon.

**Returns:**the verticalAlignment property, one of the following values:

* SwingConstants.CENTER (the default)
* SwingConstants.TOP
* SwingConstants.BOTTOM

### setVerticalAlignment

public void **setVerticalAlignment**(int alignment)

Sets the vertical alignment of the icon and text.

**Parameters:**alignment - one of the following values:

* SwingConstants.CENTER (the default)
* SwingConstants.TOP
* SwingConstants.BOTTOM

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the alignment is not one of the legal values listed above

### getHorizontalAlignment

public int **getHorizontalAlignment**()

Returns the horizontal alignment of the icon and text. AbstractButton's default is SwingConstants.CENTER, but subclasses such as JCheckBox may use a different default.

**Returns:**the horizontalAlignment property, one of the following values:

* SwingConstants.RIGHT
* SwingConstants.LEFT
* SwingConstants.CENTER
* SwingConstants.LEADING
* SwingConstants.TRAILING

### setHorizontalAlignment

public void **setHorizontalAlignment**(int alignment)

Sets the horizontal alignment of the icon and text. AbstractButton's default is SwingConstants.CENTER, but subclasses such as JCheckBox may use a different default.

**Parameters:**alignment - the alignment value, one of the following values:

* SwingConstants.RIGHT
* SwingConstants.LEFT
* SwingConstants.CENTER
* SwingConstants.LEADING
* SwingConstants.TRAILING

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the alignment is not one of the valid values

### getVerticalTextPosition

public int **getVerticalTextPosition**()

Returns the vertical position of the text relative to the icon.

**Returns:**the verticalTextPosition property, one of the following values:

* SwingConstants.CENTER (the default)
* SwingConstants.TOP
* SwingConstants.BOTTOM

### setVerticalTextPosition

public void **setVerticalTextPosition**(int textPosition)

Sets the vertical position of the text relative to the icon.

**Parameters:**textPosition - one of the following values:

* SwingConstants.CENTER (the default)
* SwingConstants.TOP
* SwingConstants.BOTTOM

### getHorizontalTextPosition

public int **getHorizontalTextPosition**()

Returns the horizontal position of the text relative to the icon.

**Returns:**the horizontalTextPosition property, one of the following values:

* SwingConstants.RIGHT
* SwingConstants.LEFT
* SwingConstants.CENTER
* SwingConstants.LEADING
* SwingConstants.TRAILING (the default)

### setHorizontalTextPosition

public void **setHorizontalTextPosition**(int textPosition)

Sets the horizontal position of the text relative to the icon.

**Parameters:**textPosition - one of the following values:

* SwingConstants.RIGHT
* SwingConstants.LEFT
* SwingConstants.CENTER
* SwingConstants.LEADING
* SwingConstants.TRAILING (the default)

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if textPosition is not one of the legal values listed above

### getIconTextGap

public int **getIconTextGap**()

Returns the amount of space between the text and the icon displayed in this button.

**Returns:**an int equal to the number of pixels between the text and the icon.**Since:** 1.4 **See Also:**[setIconTextGap(int)](http://docs.google.com/javax/swing/AbstractButton.html#setIconTextGap(int))

### setIconTextGap

public void **setIconTextGap**(int iconTextGap)

If both the icon and text properties are set, this property defines the space between them.

The default value of this property is 4 pixels.

This is a JavaBeans bound property.

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

### checkHorizontalKey

protected int **checkHorizontalKey**(int key,  
 [String](http://docs.google.com/java/lang/String.html) exception)

Verify that the key argument is a legal value for the horizontalAlignment and horizontalTextPosition properties. Valid values are:

* SwingConstants.RIGHT
* SwingConstants.LEFT
* SwingConstants.CENTER
* SwingConstants.LEADING
* SwingConstants.TRAILING

**Parameters:**key - the property value to checkexception - the message to use in the IllegalArgumentException that is thrown for an invalid value **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if key is not one of the legal values listed above**See Also:**[setHorizontalTextPosition(int)](http://docs.google.com/javax/swing/AbstractButton.html#setHorizontalTextPosition(int)), [setHorizontalAlignment(int)](http://docs.google.com/javax/swing/AbstractButton.html#setHorizontalAlignment(int))

### checkVerticalKey

protected int **checkVerticalKey**(int key,  
 [String](http://docs.google.com/java/lang/String.html) exception)

Verify that the key argument is a legal value for the vertical properties. Valid values are:

* SwingConstants.CENTER
* SwingConstants.TOP
* SwingConstants.BOTTOM

**Parameters:**key - the property value to checkexception - the message to use in the IllegalArgumentException that is thrown for an invalid value **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if key is not one of the legal values listed above

### removeNotify

public void **removeNotify**()

Notifies this component that it no longer has a parent component. When this method is invoked, any KeyboardActions set up in the the chain of parent components are removed.

**Overrides:**[removeNotify](http://docs.google.com/javax/swing/JComponent.html#removeNotify()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Since:** 1.6 **See Also:**[JComponent.registerKeyboardAction(java.awt.event.ActionListener, java.lang.String, javax.swing.KeyStroke, int)](http://docs.google.com/javax/swing/JComponent.html#registerKeyboardAction(java.awt.event.ActionListener,%20java.lang.String,%20javax.swing.KeyStroke,%20int))

### setActionCommand

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

Sets the action command for this button.

**Parameters:**actionCommand - the action command for this button

### getActionCommand

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

Returns the action command for this button.

**Returns:**the action command for this button

### setAction

public void **setAction**([Action](http://docs.google.com/javax/swing/Action.html) a)

Sets the Action. The new Action replaces any previously set Action but does not affect ActionListeners independently added with addActionListener. If the Action is already a registered ActionListener for the button, it is not re-registered.

Setting the Action results in immediately changing all the properties described in  [Swing Components Supporting Action](http://docs.google.com/Action.html#buttonActions). Subsequently, the button's properties are automatically updated as the Action's properties change.

This method uses three other methods to set and help track the Action's property values. It uses the configurePropertiesFromAction method to immediately change the button's properties. To track changes in the Action's property values, this method registers the PropertyChangeListener returned by createActionPropertyChangeListener. The default PropertyChangeListener invokes the actionPropertyChanged method when a property in the Action changes.

**Parameters:**a - the Action for the AbstractButton, or null**Since:** 1.3 **See Also:**[Action](http://docs.google.com/javax/swing/Action.html), [getAction()](http://docs.google.com/javax/swing/AbstractButton.html#getAction()), [configurePropertiesFromAction(javax.swing.Action)](http://docs.google.com/javax/swing/AbstractButton.html#configurePropertiesFromAction(javax.swing.Action)), [createActionPropertyChangeListener(javax.swing.Action)](http://docs.google.com/javax/swing/AbstractButton.html#createActionPropertyChangeListener(javax.swing.Action)), [actionPropertyChanged(javax.swing.Action, java.lang.String)](http://docs.google.com/javax/swing/AbstractButton.html#actionPropertyChanged(javax.swing.Action,%20java.lang.String))

### getAction

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

Returns the currently set Action for this ActionEvent source, or null if no Action is set.

**Returns:**the Action for this ActionEvent source, or null**Since:** 1.3 **See Also:**[Action](http://docs.google.com/javax/swing/Action.html), [setAction(javax.swing.Action)](http://docs.google.com/javax/swing/AbstractButton.html#setAction(javax.swing.Action))

### configurePropertiesFromAction

protected void **configurePropertiesFromAction**([Action](http://docs.google.com/javax/swing/Action.html) a)

Sets the properties on this button to match those in the specified Action. Refer to  [Swing Components Supporting Action](http://docs.google.com/Action.html#buttonActions) for more details as to which properties this sets.

**Parameters:**a - the Action from which to get the properties, or null**Since:** 1.3 **See Also:**[Action](http://docs.google.com/javax/swing/Action.html), [setAction(javax.swing.Action)](http://docs.google.com/javax/swing/AbstractButton.html#setAction(javax.swing.Action))

### actionPropertyChanged

protected void **actionPropertyChanged**([Action](http://docs.google.com/javax/swing/Action.html) action,  
 [String](http://docs.google.com/java/lang/String.html) propertyName)

Updates the button's state in response to property changes in the associated action. This method is invoked from the PropertyChangeListener returned from createActionPropertyChangeListener. Subclasses do not normally need to invoke this. Subclasses that support additional Action properties should override this and configurePropertiesFromAction.

Refer to the table at  [Swing Components Supporting Action](http://docs.google.com/Action.html#buttonActions) for a list of the properties this method sets.

**Parameters:**action - the Action associated with this buttonpropertyName - the name of the property that changed**Since:** 1.6 **See Also:**[Action](http://docs.google.com/javax/swing/Action.html), [configurePropertiesFromAction(javax.swing.Action)](http://docs.google.com/javax/swing/AbstractButton.html#configurePropertiesFromAction(javax.swing.Action))

### createActionPropertyChangeListener

protected [PropertyChangeListener](http://docs.google.com/java/beans/PropertyChangeListener.html) **createActionPropertyChangeListener**([Action](http://docs.google.com/javax/swing/Action.html) a)

Creates and returns a PropertyChangeListener that is responsible for listening for changes from the specified Action and updating the appropriate properties.

**Warning:** If you subclass this do not create an anonymous inner class. If you do the lifetime of the button will be tied to that of the Action.

**Parameters:**a - the button's action**Since:** 1.3 **See Also:**[Actions](#1nia2ey), [Action](http://docs.google.com/javax/swing/Action.html), [setAction(javax.swing.Action)](http://docs.google.com/javax/swing/AbstractButton.html#setAction(javax.swing.Action))

### isBorderPainted

public boolean **isBorderPainted**()

Gets the borderPainted property.

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

### setBorderPainted

public void **setBorderPainted**(boolean b)

Sets the borderPainted property. If true and the button has a border, the border is painted. The default value for the borderPainted property is true.

**Parameters:**b - if true and border property is not null, the border is painted**See Also:**[isBorderPainted()](http://docs.google.com/javax/swing/AbstractButton.html#isBorderPainted())

### paintBorder

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

Paint the button's border if BorderPainted property is true and the button has a border.

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

### isFocusPainted

public boolean **isFocusPainted**()

Gets the paintFocus property.

**Returns:**the paintFocus property**See Also:**[setFocusPainted(boolean)](http://docs.google.com/javax/swing/AbstractButton.html#setFocusPainted(boolean))

### setFocusPainted

public void **setFocusPainted**(boolean b)

Sets the paintFocus property, which must be true for the focus state to be painted. The default value for the paintFocus property is true. Some look and feels might not paint focus state; they will ignore this property.

**Parameters:**b - if true, the focus state should be painted**See Also:**[isFocusPainted()](http://docs.google.com/javax/swing/AbstractButton.html#isFocusPainted())

### isContentAreaFilled

public boolean **isContentAreaFilled**()

Gets the contentAreaFilled property.

**Returns:**the contentAreaFilled property**See Also:**[setContentAreaFilled(boolean)](http://docs.google.com/javax/swing/AbstractButton.html#setContentAreaFilled(boolean))

### setContentAreaFilled

public void **setContentAreaFilled**(boolean b)

Sets the contentAreaFilled property. If true the button will paint the content area. If you wish to have a transparent button, such as an icon only button, for example, then you should set this to false. Do not call setOpaque(false). The default value for the the contentAreaFilled property is true.

This function may cause the component's opaque property to change.

The exact behavior of calling this function varies on a component-by-component and L&F-by-L&F basis.

**Parameters:**b - if true, the content should be filled; if false the content area is not filled**See Also:**[isContentAreaFilled()](http://docs.google.com/javax/swing/AbstractButton.html#isContentAreaFilled()), [JComponent.setOpaque(boolean)](http://docs.google.com/javax/swing/JComponent.html#setOpaque(boolean))

### isRolloverEnabled

public boolean **isRolloverEnabled**()

Gets the rolloverEnabled property.

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

### setRolloverEnabled

public void **setRolloverEnabled**(boolean b)

Sets the rolloverEnabled property, which must be true for rollover effects to occur. The default value for the rolloverEnabled property is false. Some look and feels might not implement rollover effects; they will ignore this property.

**Parameters:**b - if true, rollover effects should be painted**See Also:**[isRolloverEnabled()](http://docs.google.com/javax/swing/AbstractButton.html#isRolloverEnabled())

### getMnemonic

public int **getMnemonic**()

Returns the keyboard mnemonic from the the current model.

**Returns:**the keyboard mnemonic from the model

### setMnemonic

public void **setMnemonic**(int mnemonic)

Sets the keyboard mnemonic on the current model. The mnemonic is the key which when combined with the look and feel's mouseless modifier (usually Alt) will activate this button if focus is contained somewhere within this button's ancestor window.

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.

If the character defined by the mnemonic is found within the button's label string, the first occurrence of it will be underlined to indicate the mnemonic to the user.

**Parameters:**mnemonic - the key code which represents the mnemonic**See Also:**[KeyEvent](http://docs.google.com/java/awt/event/KeyEvent.html), [setDisplayedMnemonicIndex(int)](http://docs.google.com/javax/swing/AbstractButton.html#setDisplayedMnemonicIndex(int))

### setMnemonic

public void **setMnemonic**(char mnemonic)

This method is now obsolete, please use setMnemonic(int) to set the mnemonic for a button. This method is only designed to handle character values which fall between 'a' and 'z' or 'A' and 'Z'.

**Parameters:**mnemonic - a char specifying the mnemonic value**See Also:**[setMnemonic(int)](http://docs.google.com/javax/swing/AbstractButton.html#setMnemonic(int))

### setDisplayedMnemonicIndex

public void **setDisplayedMnemonicIndex**(int index)  
 throws [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html)

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, the mnemonic character is not contained in the string, or the developer does not wish the mnemonic to be displayed.

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 was 'Save As', with a mnemonic of 'a', and you wanted the 'A' to be decorated, as 'Save As', you would have to invoke setDisplayedMnemonicIndex(5) after invoking setMnemonic(KeyEvent.VK\_A).

**Parameters:**index - Index into the String to underline **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - will be thrown if index is >= length of the text, or < -1**Since:** 1.4 **See Also:**[getDisplayedMnemonicIndex()](http://docs.google.com/javax/swing/AbstractButton.html#getDisplayedMnemonicIndex())

### getDisplayedMnemonicIndex

public int **getDisplayedMnemonicIndex**()

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

**Returns:**index representing mnemonic character**Since:** 1.4 **See Also:**[setDisplayedMnemonicIndex(int)](http://docs.google.com/javax/swing/AbstractButton.html#setDisplayedMnemonicIndex(int))

### setMultiClickThreshhold

public void **setMultiClickThreshhold**(long threshhold)

Sets the amount of time (in milliseconds) required between mouse press events for the button to generate the corresponding action events. After the initial mouse press occurs (and action event generated) any subsequent mouse press events which occur on intervals less than the threshhold will be ignored and no corresponding action event generated. By default the threshhold is 0, which means that for each mouse press, an action event will be fired, no matter how quickly the mouse clicks occur. In buttons where this behavior is not desirable (for example, the "OK" button in a dialog), this threshhold should be set to an appropriate positive value.

**Parameters:**threshhold - the amount of time required between mouse press events to generate corresponding action events **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if threshhold < 0**Since:** 1.4 **See Also:**[getMultiClickThreshhold()](http://docs.google.com/javax/swing/AbstractButton.html#getMultiClickThreshhold())

### getMultiClickThreshhold

public long **getMultiClickThreshhold**()

Gets the amount of time (in milliseconds) required between mouse press events for the button to generate the corresponding action events.

**Returns:**the amount of time required between mouse press events to generate corresponding action events**Since:** 1.4 **See Also:**[setMultiClickThreshhold(long)](http://docs.google.com/javax/swing/AbstractButton.html#setMultiClickThreshhold(long))

### getModel

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

Returns the model that this button represents.

**Returns:**the model property**See Also:**[setModel(javax.swing.ButtonModel)](http://docs.google.com/javax/swing/AbstractButton.html#setModel(javax.swing.ButtonModel))

### setModel

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

Sets the model that this button represents.

**Parameters:**newModel - the new ButtonModel**See Also:**[getModel()](http://docs.google.com/javax/swing/AbstractButton.html#getModel())

### getUI

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

Returns the L&F object that renders this component.

**Returns:**the ButtonUI object**See Also:**[setUI(javax.swing.plaf.ButtonUI)](http://docs.google.com/javax/swing/AbstractButton.html#setUI(javax.swing.plaf.ButtonUI))

### setUI

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

Sets the L&F object that renders this component.

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

### updateUI

public void **updateUI**()

Resets the UI property to a value from the current look and feel. Subtypes of AbstractButton should override this to update the UI. For example, JButton might do the following:

setUI((ButtonUI)UIManager.getUI(  
 "ButtonUI", "javax.swing.plaf.basic.BasicButtonUI", this));

**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.setUI(javax.swing.plaf.ComponentUI)](http://docs.google.com/javax/swing/JComponent.html#setUI(javax.swing.plaf.ComponentUI)), [UIManager.getLookAndFeel()](http://docs.google.com/javax/swing/UIManager.html#getLookAndFeel()), [UIManager.getUI(javax.swing.JComponent)](http://docs.google.com/javax/swing/UIManager.html#getUI(javax.swing.JComponent))

### 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 container at the specified index, refer to [Container.addImpl(Component, Object, int)](http://docs.google.com/java/awt/Container.html#addImpl(java.awt.Component,%20java.lang.Object,%20int)) for a complete description of this method.

**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 be addedconstraints - an object expressing layout constraints for this componentindex - the position in the container's list at which to insert the component, where -1 means append to the end **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if index is invalid [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if adding the container's parent to itself [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if adding a window to a container**Since:** 1.5 **See Also:**[Container.add(Component)](http://docs.google.com/java/awt/Container.html#add(java.awt.Component)), [Container.add(Component, int)](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20int)), [Container.add(Component, java.lang.Object)](http://docs.google.com/java/awt/Container.html#add(java.awt.Component,%20java.lang.Object)), [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html), [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html)

### setLayout

public void **setLayout**([LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) mgr)

Sets the layout manager for this container, refer to [Container.setLayout(LayoutManager)](http://docs.google.com/java/awt/Container.html#setLayout(java.awt.LayoutManager)) for a complete description of this method.

**Overrides:**[setLayout](http://docs.google.com/java/awt/Container.html#setLayout(java.awt.LayoutManager)) in class [Container](http://docs.google.com/java/awt/Container.html) **Parameters:**mgr - the specified layout manager**Since:** 1.5 **See Also:**[Container.doLayout()](http://docs.google.com/java/awt/Container.html#doLayout()), [Container.getLayout()](http://docs.google.com/java/awt/Container.html#getLayout())

### addChangeListener

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

Adds a ChangeListener to the button.

**Parameters:**l - the listener to be added

### removeChangeListener

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

Removes a ChangeListener from the button.

**Parameters:**l - the listener to be removed

### getChangeListeners

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

Returns an array of all the ChangeListeners added to this AbstractButton 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**()

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

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

### addActionListener

public void **addActionListener**([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)

Adds an ActionListener to the button.

**Parameters:**l - the ActionListener to be added

### removeActionListener

public void **removeActionListener**([ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) l)

Removes an ActionListener from the button. If the listener is the currently set Action for the button, then the Action is set to null.

**Parameters:**l - the listener to be removed

### getActionListeners

public [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html)[] **getActionListeners**()

Returns an array of all the ActionListeners added to this AbstractButton with addActionListener().

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

### 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 another ChangeListener implementation.

**Returns:**the new ChangeListener

### fireActionPerformed

protected void **fireActionPerformed**([ActionEvent](http://docs.google.com/java/awt/event/ActionEvent.html) event)

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

**Parameters:**event - the ActionEvent object**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireItemStateChanged

protected void **fireItemStateChanged**([ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html) event)

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

**Parameters:**event - the ItemEvent object**See Also:**[EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### createActionListener

protected [ActionListener](http://docs.google.com/java/awt/event/ActionListener.html) **createActionListener**()

### createItemListener

protected [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) **createItemListener**()

### setEnabled

public void **setEnabled**(boolean b)

Enables (or disables) the button.

**Overrides:**[setEnabled](http://docs.google.com/javax/swing/JComponent.html#setEnabled(boolean)) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Parameters:**b - true to enable the button, otherwise false**See Also:**[Component.isEnabled()](http://docs.google.com/java/awt/Component.html#isEnabled()), [Component.isLightweight()](http://docs.google.com/java/awt/Component.html#isLightweight())

### getLabel

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

**Deprecated.** *- Replaced by getText*

Returns the label text.

**Returns:**a String containing the label

### setLabel

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public void **setLabel**([String](http://docs.google.com/java/lang/String.html) label)

**Deprecated.** *- Replaced by setText(text)*

Sets the label text.

**Parameters:**label - a String containing the text

### addItemListener

public void **addItemListener**([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)

Adds an ItemListener to the checkbox.

**Specified by:**[addItemListener](http://docs.google.com/java/awt/ItemSelectable.html#addItemListener(java.awt.event.ItemListener)) in interface [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html) **Parameters:**l - the ItemListener to be added**See Also:**[ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html)

### removeItemListener

public void **removeItemListener**([ItemListener](http://docs.google.com/java/awt/event/ItemListener.html) l)

Removes an ItemListener from the button.

**Specified by:**[removeItemListener](http://docs.google.com/java/awt/ItemSelectable.html#removeItemListener(java.awt.event.ItemListener)) in interface [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html) **Parameters:**l - the ItemListener to be removed**See Also:**[ItemEvent](http://docs.google.com/java/awt/event/ItemEvent.html)

### getItemListeners

public [ItemListener](http://docs.google.com/java/awt/event/ItemListener.html)[] **getItemListeners**()

Returns an array of all the ItemListeners added to this AbstractButton with addItemListener().

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

### getSelectedObjects

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

Returns an array (length 1) containing the label or null if the button is not selected.

**Specified by:**[getSelectedObjects](http://docs.google.com/java/awt/ItemSelectable.html#getSelectedObjects()) in interface [ItemSelectable](http://docs.google.com/java/awt/ItemSelectable.html) **Returns:**an array containing 1 Object: the text of the button, if the item is selected; otherwise null

### init

protected void **init**([String](http://docs.google.com/java/lang/String.html) text,  
 [Icon](http://docs.google.com/javax/swing/Icon.html) icon)

### imageUpdate

public boolean **imageUpdate**([Image](http://docs.google.com/java/awt/Image.html) img,  
 int infoflags,  
 int x,  
 int y,  
 int w,  
 int h)

This is overridden to return false if the current Icon's Image is not equal to the passed in Image img.

**Specified by:**[imageUpdate](http://docs.google.com/java/awt/image/ImageObserver.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int)) in interface [ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html)**Overrides:**[imageUpdate](http://docs.google.com/java/awt/Component.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int)) in class [Component](http://docs.google.com/java/awt/Component.html) **Parameters:**img - the Image to be comparedinfoflags - flags used to repaint the button when the image is updated and which determine how much is to be paintedx - the x coordinatey - the y coordinatew - the widthh - the height **Returns:**false if the infoflags indicate that the image is completely loaded; true otherwise.**See Also:**[ImageObserver](http://docs.google.com/java/awt/image/ImageObserver.html), [Component.imageUpdate(java.awt.Image, int, int, int, int, int)](http://docs.google.com/java/awt/Component.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int))

### paramString

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

Returns a string representation of this AbstractButton. 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.

Overriding paramString to provide information about the specific new aspects of the JFC components.

**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 AbstractButton

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AbstractButton.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/AbstractAction.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/AbstractButton.AccessibleAbstractButton.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/AbstractButton.html)    [**NO FRAMES**](http://docs.google.com/AbstractButton.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#lnxbz9) | DETAIL: [FIELD](#z337ya) | [CONSTR](#2u6wntf) | [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).