| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JEditorPane.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/JDialog.AccessibleJDialog.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JEditorPane.AccessibleJEditorPane.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JEditorPane.html)    [**NO FRAMES**](http://docs.google.com/JEditorPane.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#4d34og8) | [CONSTR](#lnxbz9) | [METHOD](#35nkun2) | DETAIL: [FIELD](#1y810tw) | [CONSTR](#1ci93xb) | [METHOD](#1pxezwc) |

## **javax.swing**

Class JEditorPane

[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.text.JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html)  
 **javax.swing.JEditorPane**

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

public class **JEditorPane**extends [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html)

A text component to edit various kinds of content. You can find how-to information and examples of using editor panes in [Using Text Components](http://java.sun.com/docs/books/tutorial/uiswing/components/text.html), a section in *The Java Tutorial.*

This component uses implementations of the EditorKit to accomplish its behavior. It effectively morphs into the proper kind of text editor for the kind of content it is given. The content type that editor is bound to at any given time is determined by the EditorKit currently installed. If the content is set to a new URL, its type is used to determine the EditorKit that should be used to load the content.

By default, the following types of content are known:

**text/plain** Plain text, which is the default the type given isn't recognized. The kit used in this case is an extension of DefaultEditorKit that produces a wrapped plain text view. **text/html** HTML text. The kit used in this case is the class javax.swing.text.html.HTMLEditorKit which provides HTML 3.2 support. **text/rtf** RTF text. The kit used in this case is the class javax.swing.text.rtf.RTFEditorKit which provides a limited support of the Rich Text Format.

There are several ways to load content into this component.

1. The [setText](http://docs.google.com/javax/swing/JEditorPane.html#setText(java.lang.String)) method can be used to initialize the component from a string. In this case the current EditorKit will be used, and the content type will be expected to be of this type.
2. The [read](http://docs.google.com/javax/swing/JEditorPane.html#read(java.io.InputStream,%20java.lang.Object)) method can be used to initialize the component from a Reader. Note that if the content type is HTML, relative references (e.g. for things like images) can't be resolved unless the <base> tag is used or the *Base* property on HTMLDocument is set. In this case the current EditorKit will be used, and the content type will be expected to be of this type.
3. The [setPage](http://docs.google.com/javax/swing/JEditorPane.html#setPage(java.net.URL)) method can be used to initialize the component from a URL. In this case, the content type will be determined from the URL, and the registered EditorKit for that content type will be set.

Some kinds of content may provide hyperlink support by generating hyperlink events. The HTML EditorKit will generate hyperlink events if the JEditorPane is *not editable* (JEditorPane.setEditable(false); has been called). If HTML frames are embedded in the document, the typical response would be to change a portion of the current document. The following code fragment is a possible hyperlink listener implementation, that treats HTML frame events specially, and simply displays any other activated hyperlinks.

  class Hyperactive implements HyperlinkListener {  
   
  public void hyperlinkUpdate(HyperlinkEvent e) {  
  if (e.getEventType() == HyperlinkEvent.EventType.ACTIVATED) {  
  JEditorPane pane = (JEditorPane) e.getSource();  
  if (e instanceof HTMLFrameHyperlinkEvent) {  
  HTMLFrameHyperlinkEvent evt = (HTMLFrameHyperlinkEvent)e;  
  HTMLDocument doc = (HTMLDocument)pane.getDocument();  
  doc.processHTMLFrameHyperlinkEvent(evt);  
  } else {  
  try {  
  pane.setPage(e.getURL());  
  } catch (Throwable t) {  
  t.printStackTrace();  
  }  
  }  
  }  
  }  
  }

For information on customizing how **text/html** is rendered please see [W3C\_LENGTH\_UNITS](http://docs.google.com/javax/swing/JEditorPane.html#W3C_LENGTH_UNITS) and [HONOR\_DISPLAY\_PROPERTIES](http://docs.google.com/javax/swing/JEditorPane.html#HONOR_DISPLAY_PROPERTIES)

Culturally dependent information in some documents is handled through a mechanism called character encoding. Character encoding is an unambiguous mapping of the members of a character set (letters, ideographs, digits, symbols, or control functions) to specific numeric code values. It represents the way the file is stored. Example character encodings are ISO-8859-1, ISO-8859-5, Shift-jis, Euc-jp, and UTF-8. When the file is passed to an user agent (JEditorPane) it is converted to the document character set (ISO-10646 aka Unicode).

There are multiple ways to get a character set mapping to happen with JEditorPane.

1. One way is to specify the character set as a parameter of the MIME type. This will be established by a call to the [setContentType](#1rvwp1q) method. If the content is loaded by the [setPage](#4bvk7pj) method the content type will have been set according to the specification of the URL. It the file is loaded directly, the content type would be expected to have been set prior to loading.
2. Another way the character set can be specified is in the document itself. This requires reading the document prior to determining the character set that is desired. To handle this, it is expected that the EditorKit.read operation throw a ChangedCharSetException which will be caught. The read is then restarted with a new Reader that uses the character set specified in the ChangedCharSetException (which is an IOException).

**Newlines** For a discussion on how newlines are handled, see [DefaultEditorKit](http://docs.google.com/text/DefaultEditorKit.html).

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

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

| **Nested Class Summary** | |
| --- | --- |
| protected  class | [**JEditorPane.AccessibleJEditorPane**](http://docs.google.com/javax/swing/JEditorPane.AccessibleJEditorPane.html)            This class implements accessibility support for the JEditorPane class. |
| protected  class | [**JEditorPane.AccessibleJEditorPaneHTML**](http://docs.google.com/javax/swing/JEditorPane.AccessibleJEditorPaneHTML.html)            This class provides support for AccessibleHypertext, and is used in instances where the EditorKit installed in this JEditorPane is an instance of HTMLEditorKit. |
| protected  class | [**JEditorPane.JEditorPaneAccessibleHypertextSupport**](http://docs.google.com/javax/swing/JEditorPane.JEditorPaneAccessibleHypertextSupport.html)            What's returned by AccessibleJEditorPaneHTML.getAccessibleText. |

| **Nested classes/interfaces inherited from class javax.swing.text.**[**JTextComponent**](http://docs.google.com/javax/swing/text/JTextComponent.html) |
| --- |
| [JTextComponent.AccessibleJTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.AccessibleJTextComponent.html), [JTextComponent.DropLocation](http://docs.google.com/javax/swing/text/JTextComponent.DropLocation.html), [JTextComponent.KeyBinding](http://docs.google.com/javax/swing/text/JTextComponent.KeyBinding.html) |

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

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

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

| **Field Summary** | |
| --- | --- |
| static [String](http://docs.google.com/java/lang/String.html) | [**HONOR\_DISPLAY\_PROPERTIES**](http://docs.google.com/javax/swing/JEditorPane.html#HONOR_DISPLAY_PROPERTIES)            Key for a client property used to indicate whether the default font and foreground color from the component are used if a font or foreground color is not specified in the styled text. |
| static [String](http://docs.google.com/java/lang/String.html) | [**W3C\_LENGTH\_UNITS**](http://docs.google.com/javax/swing/JEditorPane.html#W3C_LENGTH_UNITS)            Key for a client property used to indicate whether  [w3c compliant](http://www.w3.org/TR/CSS21/syndata.html#length-units) length units are used for html rendering. |

| **Fields inherited from class javax.swing.text.**[**JTextComponent**](http://docs.google.com/javax/swing/text/JTextComponent.html) |
| --- |
| [DEFAULT\_KEYMAP](http://docs.google.com/javax/swing/text/JTextComponent.html#DEFAULT_KEYMAP), [FOCUS\_ACCELERATOR\_KEY](http://docs.google.com/javax/swing/text/JTextComponent.html#FOCUS_ACCELERATOR_KEY) |

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

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

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

| **Constructor Summary** | |
| --- | --- |
| [**JEditorPane**](http://docs.google.com/javax/swing/JEditorPane.html#JEditorPane())()            Creates a new JEditorPane. |
| [**JEditorPane**](http://docs.google.com/javax/swing/JEditorPane.html#JEditorPane(java.lang.String))([String](http://docs.google.com/java/lang/String.html) url)            Creates a JEditorPane based on a string containing a URL specification. |
| [**JEditorPane**](http://docs.google.com/javax/swing/JEditorPane.html#JEditorPane(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) type, [String](http://docs.google.com/java/lang/String.html) text)            Creates a JEditorPane that has been initialized to the given text. |
| [**JEditorPane**](http://docs.google.com/javax/swing/JEditorPane.html#JEditorPane(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) initialPage)            Creates a JEditorPane based on a specified URL for input. |

| **Method Summary** | |
| --- | --- |
| void | [**addHyperlinkListener**](http://docs.google.com/javax/swing/JEditorPane.html#addHyperlinkListener(javax.swing.event.HyperlinkListener))([HyperlinkListener](http://docs.google.com/javax/swing/event/HyperlinkListener.html) listener)            Adds a hyperlink listener for notification of any changes, for example when a link is selected and entered. |
| protected  [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) | [**createDefaultEditorKit**](http://docs.google.com/javax/swing/JEditorPane.html#createDefaultEditorKit())()            Creates the default editor kit (PlainEditorKit) for when the component is first created. |
| static [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) | [**createEditorKitForContentType**](http://docs.google.com/javax/swing/JEditorPane.html#createEditorKitForContentType(java.lang.String))([String](http://docs.google.com/java/lang/String.html) type)            Creates a handler for the given type from the default registry of editor kits. |
| void | [**fireHyperlinkUpdate**](http://docs.google.com/javax/swing/JEditorPane.html#fireHyperlinkUpdate(javax.swing.event.HyperlinkEvent))([HyperlinkEvent](http://docs.google.com/javax/swing/event/HyperlinkEvent.html) e)            Notifies all listeners that have registered interest for notification on this event type. |
| [AccessibleContext](http://docs.google.com/javax/accessibility/AccessibleContext.html) | [**getAccessibleContext**](http://docs.google.com/javax/swing/JEditorPane.html#getAccessibleContext())()            Gets the AccessibleContext associated with this JEditorPane. |
| [String](http://docs.google.com/java/lang/String.html) | [**getContentType**](http://docs.google.com/javax/swing/JEditorPane.html#getContentType())()            Gets the type of content that this editor is currently set to deal with. |
| [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) | [**getEditorKit**](http://docs.google.com/javax/swing/JEditorPane.html#getEditorKit())()            Fetches the currently installed kit for handling content. |
| static [String](http://docs.google.com/java/lang/String.html) | [**getEditorKitClassNameForContentType**](http://docs.google.com/javax/swing/JEditorPane.html#getEditorKitClassNameForContentType(java.lang.String))([String](http://docs.google.com/java/lang/String.html) type)            Returns the currently registered EditorKit class name for the type type. |
| [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) | [**getEditorKitForContentType**](http://docs.google.com/javax/swing/JEditorPane.html#getEditorKitForContentType(java.lang.String))([String](http://docs.google.com/java/lang/String.html) type)            Fetches the editor kit to use for the given type of content. |
| [HyperlinkListener](http://docs.google.com/javax/swing/event/HyperlinkListener.html)[] | [**getHyperlinkListeners**](http://docs.google.com/javax/swing/JEditorPane.html#getHyperlinkListeners())()            Returns an array of all the HyperLinkListeners added to this JEditorPane with addHyperlinkListener(). |
| [URL](http://docs.google.com/java/net/URL.html) | [**getPage**](http://docs.google.com/javax/swing/JEditorPane.html#getPage())()            Gets the current URL being displayed. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**getPreferredSize**](http://docs.google.com/javax/swing/JEditorPane.html#getPreferredSize())()            Returns the preferred size for the JEditorPane. |
| boolean | [**getScrollableTracksViewportHeight**](http://docs.google.com/javax/swing/JEditorPane.html#getScrollableTracksViewportHeight())()            Returns true if a viewport should always force the height of this Scrollable to match the height of the viewport. |
| boolean | [**getScrollableTracksViewportWidth**](http://docs.google.com/javax/swing/JEditorPane.html#getScrollableTracksViewportWidth())()            Returns true if a viewport should always force the width of this Scrollable to match the width of the viewport. |
| protected  [InputStream](http://docs.google.com/java/io/InputStream.html) | [**getStream**](http://docs.google.com/javax/swing/JEditorPane.html#getStream(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) page)            Fetches a stream for the given URL, which is about to be loaded by the setPage method. |
| [String](http://docs.google.com/java/lang/String.html) | [**getText**](http://docs.google.com/javax/swing/JEditorPane.html#getText())()            Returns the text contained in this TextComponent in terms of the content type of this editor. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUIClassID**](http://docs.google.com/javax/swing/JEditorPane.html#getUIClassID())()            Gets the class ID for the UI. |
| protected  [String](http://docs.google.com/java/lang/String.html) | [**paramString**](http://docs.google.com/javax/swing/JEditorPane.html#paramString())()            Returns a string representation of this JEditorPane. |
| void | [**read**](http://docs.google.com/javax/swing/JEditorPane.html#read(java.io.InputStream,%20java.lang.Object))([InputStream](http://docs.google.com/java/io/InputStream.html) in, [Object](http://docs.google.com/java/lang/Object.html) desc)            This method initializes from a stream. |
| static void | [**registerEditorKitForContentType**](http://docs.google.com/javax/swing/JEditorPane.html#registerEditorKitForContentType(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) type, [String](http://docs.google.com/java/lang/String.html) classname)            Establishes the default bindings of type to classname. |
| static void | [**registerEditorKitForContentType**](http://docs.google.com/javax/swing/JEditorPane.html#registerEditorKitForContentType(java.lang.String,%20java.lang.String,%20java.lang.ClassLoader))([String](http://docs.google.com/java/lang/String.html) type, [String](http://docs.google.com/java/lang/String.html) classname, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)            Establishes the default bindings of type to classname. |
| void | [**removeHyperlinkListener**](http://docs.google.com/javax/swing/JEditorPane.html#removeHyperlinkListener(javax.swing.event.HyperlinkListener))([HyperlinkListener](http://docs.google.com/javax/swing/event/HyperlinkListener.html) listener)            Removes a hyperlink listener. |
| void | [**replaceSelection**](http://docs.google.com/javax/swing/JEditorPane.html#replaceSelection(java.lang.String))([String](http://docs.google.com/java/lang/String.html) content)            Replaces the currently selected content with new content represented by the given string. |
| void | [**scrollToReference**](http://docs.google.com/javax/swing/JEditorPane.html#scrollToReference(java.lang.String))([String](http://docs.google.com/java/lang/String.html) reference)            Scrolls the view to the given reference location (that is, the value returned by the UL.getRef method for the URL being displayed). |
| void | [**setContentType**](http://docs.google.com/javax/swing/JEditorPane.html#setContentType(java.lang.String))([String](http://docs.google.com/java/lang/String.html) type)            Sets the type of content that this editor handles. |
| void | [**setEditorKit**](http://docs.google.com/javax/swing/JEditorPane.html#setEditorKit(javax.swing.text.EditorKit))([EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) kit)            Sets the currently installed kit for handling content. |
| void | [**setEditorKitForContentType**](http://docs.google.com/javax/swing/JEditorPane.html#setEditorKitForContentType(java.lang.String,%20javax.swing.text.EditorKit))([String](http://docs.google.com/java/lang/String.html) type, [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) k)            Directly sets the editor kit to use for the given type. |
| void | [**setPage**](http://docs.google.com/javax/swing/JEditorPane.html#setPage(java.lang.String))([String](http://docs.google.com/java/lang/String.html) url)            Sets the current URL being displayed. |
| void | [**setPage**](http://docs.google.com/javax/swing/JEditorPane.html#setPage(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) page)            Sets the current URL being displayed. |
| void | [**setText**](http://docs.google.com/javax/swing/JEditorPane.html#setText(java.lang.String))([String](http://docs.google.com/java/lang/String.html) t)            Sets the text of this TextComponent to the specified content, which is expected to be in the format of the content type of this editor. |

| **Methods inherited from class javax.swing.text.**[**JTextComponent**](http://docs.google.com/javax/swing/text/JTextComponent.html) |
| --- |
| [addCaretListener](http://docs.google.com/javax/swing/text/JTextComponent.html#addCaretListener(javax.swing.event.CaretListener)), [addInputMethodListener](http://docs.google.com/javax/swing/text/JTextComponent.html#addInputMethodListener(java.awt.event.InputMethodListener)), [addKeymap](http://docs.google.com/javax/swing/text/JTextComponent.html#addKeymap(java.lang.String,%20javax.swing.text.Keymap)), [copy](http://docs.google.com/javax/swing/text/JTextComponent.html#copy()), [cut](http://docs.google.com/javax/swing/text/JTextComponent.html#cut()), [fireCaretUpdate](http://docs.google.com/javax/swing/text/JTextComponent.html#fireCaretUpdate(javax.swing.event.CaretEvent)), [getActions](http://docs.google.com/javax/swing/text/JTextComponent.html#getActions()), [getCaret](http://docs.google.com/javax/swing/text/JTextComponent.html#getCaret()), [getCaretColor](http://docs.google.com/javax/swing/text/JTextComponent.html#getCaretColor()), [getCaretListeners](http://docs.google.com/javax/swing/text/JTextComponent.html#getCaretListeners()), [getCaretPosition](http://docs.google.com/javax/swing/text/JTextComponent.html#getCaretPosition()), [getDisabledTextColor](http://docs.google.com/javax/swing/text/JTextComponent.html#getDisabledTextColor()), [getDocument](http://docs.google.com/javax/swing/text/JTextComponent.html#getDocument()), [getDragEnabled](http://docs.google.com/javax/swing/text/JTextComponent.html#getDragEnabled()), [getDropLocation](http://docs.google.com/javax/swing/text/JTextComponent.html#getDropLocation()), [getDropMode](http://docs.google.com/javax/swing/text/JTextComponent.html#getDropMode()), [getFocusAccelerator](http://docs.google.com/javax/swing/text/JTextComponent.html#getFocusAccelerator()), [getHighlighter](http://docs.google.com/javax/swing/text/JTextComponent.html#getHighlighter()), [getInputMethodRequests](http://docs.google.com/javax/swing/text/JTextComponent.html#getInputMethodRequests()), [getKeymap](http://docs.google.com/javax/swing/text/JTextComponent.html#getKeymap()), [getKeymap](http://docs.google.com/javax/swing/text/JTextComponent.html#getKeymap(java.lang.String)), [getMargin](http://docs.google.com/javax/swing/text/JTextComponent.html#getMargin()), [getNavigationFilter](http://docs.google.com/javax/swing/text/JTextComponent.html#getNavigationFilter()), [getPreferredScrollableViewportSize](http://docs.google.com/javax/swing/text/JTextComponent.html#getPreferredScrollableViewportSize()), [getPrintable](http://docs.google.com/javax/swing/text/JTextComponent.html#getPrintable(java.text.MessageFormat,%20java.text.MessageFormat)), [getScrollableBlockIncrement](http://docs.google.com/javax/swing/text/JTextComponent.html#getScrollableBlockIncrement(java.awt.Rectangle,%20int,%20int)), [getScrollableUnitIncrement](http://docs.google.com/javax/swing/text/JTextComponent.html#getScrollableUnitIncrement(java.awt.Rectangle,%20int,%20int)), [getSelectedText](http://docs.google.com/javax/swing/text/JTextComponent.html#getSelectedText()), [getSelectedTextColor](http://docs.google.com/javax/swing/text/JTextComponent.html#getSelectedTextColor()), [getSelectionColor](http://docs.google.com/javax/swing/text/JTextComponent.html#getSelectionColor()), [getSelectionEnd](http://docs.google.com/javax/swing/text/JTextComponent.html#getSelectionEnd()), [getSelectionStart](http://docs.google.com/javax/swing/text/JTextComponent.html#getSelectionStart()), [getText](http://docs.google.com/javax/swing/text/JTextComponent.html#getText(int,%20int)), [getToolTipText](http://docs.google.com/javax/swing/text/JTextComponent.html#getToolTipText(java.awt.event.MouseEvent)), [getUI](http://docs.google.com/javax/swing/text/JTextComponent.html#getUI()), [isEditable](http://docs.google.com/javax/swing/text/JTextComponent.html#isEditable()), [loadKeymap](http://docs.google.com/javax/swing/text/JTextComponent.html#loadKeymap(javax.swing.text.Keymap,%20javax.swing.text.JTextComponent.KeyBinding%5B%5D,%20javax.swing.Action%5B%5D)), [modelToView](http://docs.google.com/javax/swing/text/JTextComponent.html#modelToView(int)), [moveCaretPosition](http://docs.google.com/javax/swing/text/JTextComponent.html#moveCaretPosition(int)), [paste](http://docs.google.com/javax/swing/text/JTextComponent.html#paste()), [print](http://docs.google.com/javax/swing/text/JTextComponent.html#print()), [print](http://docs.google.com/javax/swing/text/JTextComponent.html#print(java.text.MessageFormat,%20java.text.MessageFormat)), [print](http://docs.google.com/javax/swing/text/JTextComponent.html#print(java.text.MessageFormat,%20java.text.MessageFormat,%20boolean,%20javax.print.PrintService,%20javax.print.attribute.PrintRequestAttributeSet,%20boolean)), [processInputMethodEvent](http://docs.google.com/javax/swing/text/JTextComponent.html#processInputMethodEvent(java.awt.event.InputMethodEvent)), [read](http://docs.google.com/javax/swing/text/JTextComponent.html#read(java.io.Reader,%20java.lang.Object)), [removeCaretListener](http://docs.google.com/javax/swing/text/JTextComponent.html#removeCaretListener(javax.swing.event.CaretListener)), [removeKeymap](http://docs.google.com/javax/swing/text/JTextComponent.html#removeKeymap(java.lang.String)), [removeNotify](http://docs.google.com/javax/swing/text/JTextComponent.html#removeNotify()), [select](http://docs.google.com/javax/swing/text/JTextComponent.html#select(int,%20int)), [selectAll](http://docs.google.com/javax/swing/text/JTextComponent.html#selectAll()), [setCaret](http://docs.google.com/javax/swing/text/JTextComponent.html#setCaret(javax.swing.text.Caret)), [setCaretColor](http://docs.google.com/javax/swing/text/JTextComponent.html#setCaretColor(java.awt.Color)), [setCaretPosition](http://docs.google.com/javax/swing/text/JTextComponent.html#setCaretPosition(int)), [setComponentOrientation](http://docs.google.com/javax/swing/text/JTextComponent.html#setComponentOrientation(java.awt.ComponentOrientation)), [setDisabledTextColor](http://docs.google.com/javax/swing/text/JTextComponent.html#setDisabledTextColor(java.awt.Color)), [setDocument](http://docs.google.com/javax/swing/text/JTextComponent.html#setDocument(javax.swing.text.Document)), [setDragEnabled](http://docs.google.com/javax/swing/text/JTextComponent.html#setDragEnabled(boolean)), [setDropMode](http://docs.google.com/javax/swing/text/JTextComponent.html#setDropMode(javax.swing.DropMode)), [setEditable](http://docs.google.com/javax/swing/text/JTextComponent.html#setEditable(boolean)), [setFocusAccelerator](http://docs.google.com/javax/swing/text/JTextComponent.html#setFocusAccelerator(char)), [setHighlighter](http://docs.google.com/javax/swing/text/JTextComponent.html#setHighlighter(javax.swing.text.Highlighter)), [setKeymap](http://docs.google.com/javax/swing/text/JTextComponent.html#setKeymap(javax.swing.text.Keymap)), [setMargin](http://docs.google.com/javax/swing/text/JTextComponent.html#setMargin(java.awt.Insets)), [setNavigationFilter](http://docs.google.com/javax/swing/text/JTextComponent.html#setNavigationFilter(javax.swing.text.NavigationFilter)), [setSelectedTextColor](http://docs.google.com/javax/swing/text/JTextComponent.html#setSelectedTextColor(java.awt.Color)), [setSelectionColor](http://docs.google.com/javax/swing/text/JTextComponent.html#setSelectionColor(java.awt.Color)), [setSelectionEnd](http://docs.google.com/javax/swing/text/JTextComponent.html#setSelectionEnd(int)), [setSelectionStart](http://docs.google.com/javax/swing/text/JTextComponent.html#setSelectionStart(int)), [setUI](http://docs.google.com/javax/swing/text/JTextComponent.html#setUI(javax.swing.plaf.TextUI)), [updateUI](http://docs.google.com/javax/swing/text/JTextComponent.html#updateUI()), [viewToModel](http://docs.google.com/javax/swing/text/JTextComponent.html#viewToModel(java.awt.Point)), [write](http://docs.google.com/javax/swing/text/JTextComponent.html#write(java.io.Writer)) |

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

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

| **Methods inherited from class java.awt.**[**Component**](http://docs.google.com/java/awt/Component.html) |
| --- |
| [action](http://docs.google.com/java/awt/Component.html#action(java.awt.Event,%20java.lang.Object)), [add](http://docs.google.com/java/awt/Component.html#add(java.awt.PopupMenu)), [addComponentListener](http://docs.google.com/java/awt/Component.html#addComponentListener(java.awt.event.ComponentListener)), [addFocusListener](http://docs.google.com/java/awt/Component.html#addFocusListener(java.awt.event.FocusListener)), [addHierarchyBoundsListener](http://docs.google.com/java/awt/Component.html#addHierarchyBoundsListener(java.awt.event.HierarchyBoundsListener)), [addHierarchyListener](http://docs.google.com/java/awt/Component.html#addHierarchyListener(java.awt.event.HierarchyListener)), [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()), [getKeyListeners](http://docs.google.com/java/awt/Component.html#getKeyListeners()), [getLocale](http://docs.google.com/java/awt/Component.html#getLocale()), [getLocation](http://docs.google.com/java/awt/Component.html#getLocation()), [getLocationOnScreen](http://docs.google.com/java/awt/Component.html#getLocationOnScreen()), [getMouseListeners](http://docs.google.com/java/awt/Component.html#getMouseListeners()), [getMouseMotionListeners](http://docs.google.com/java/awt/Component.html#getMouseMotionListeners()), [getMousePosition](http://docs.google.com/java/awt/Component.html#getMousePosition()), [getMouseWheelListeners](http://docs.google.com/java/awt/Component.html#getMouseWheelListeners()), [getName](http://docs.google.com/java/awt/Component.html#getName()), [getParent](http://docs.google.com/java/awt/Component.html#getParent()), [getPeer](http://docs.google.com/java/awt/Component.html#getPeer()), [getPropertyChangeListeners](http://docs.google.com/java/awt/Component.html#getPropertyChangeListeners()), [getPropertyChangeListeners](http://docs.google.com/java/awt/Component.html#getPropertyChangeListeners(java.lang.String)), [getSize](http://docs.google.com/java/awt/Component.html#getSize()), [getToolkit](http://docs.google.com/java/awt/Component.html#getToolkit()), [getTreeLock](http://docs.google.com/java/awt/Component.html#getTreeLock()), [gotFocus](http://docs.google.com/java/awt/Component.html#gotFocus(java.awt.Event,%20java.lang.Object)), [handleEvent](http://docs.google.com/java/awt/Component.html#handleEvent(java.awt.Event)), [hasFocus](http://docs.google.com/java/awt/Component.html#hasFocus()), [hide](http://docs.google.com/java/awt/Component.html#hide()), [imageUpdate](http://docs.google.com/java/awt/Component.html#imageUpdate(java.awt.Image,%20int,%20int,%20int,%20int,%20int)), [inside](http://docs.google.com/java/awt/Component.html#inside(int,%20int)), [isBackgroundSet](http://docs.google.com/java/awt/Component.html#isBackgroundSet()), [isCursorSet](http://docs.google.com/java/awt/Component.html#isCursorSet()), [isDisplayable](http://docs.google.com/java/awt/Component.html#isDisplayable()), [isEnabled](http://docs.google.com/java/awt/Component.html#isEnabled()), [isFocusable](http://docs.google.com/java/awt/Component.html#isFocusable()), [isFocusOwner](http://docs.google.com/java/awt/Component.html#isFocusOwner()), [isFocusTraversable](http://docs.google.com/java/awt/Component.html#isFocusTraversable()), [isFontSet](http://docs.google.com/java/awt/Component.html#isFontSet()), [isForegroundSet](http://docs.google.com/java/awt/Component.html#isForegroundSet()), [isLightweight](http://docs.google.com/java/awt/Component.html#isLightweight()), [isMaximumSizeSet](http://docs.google.com/java/awt/Component.html#isMaximumSizeSet()), [isMinimumSizeSet](http://docs.google.com/java/awt/Component.html#isMinimumSizeSet()), [isPreferredSizeSet](http://docs.google.com/java/awt/Component.html#isPreferredSizeSet()), [isShowing](http://docs.google.com/java/awt/Component.html#isShowing()), [isValid](http://docs.google.com/java/awt/Component.html#isValid()), [isVisible](http://docs.google.com/java/awt/Component.html#isVisible()), [keyDown](http://docs.google.com/java/awt/Component.html#keyDown(java.awt.Event,%20int)), [keyUp](http://docs.google.com/java/awt/Component.html#keyUp(java.awt.Event,%20int)), [list](http://docs.google.com/java/awt/Component.html#list()), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintStream)), [list](http://docs.google.com/java/awt/Component.html#list(java.io.PrintWriter)), [location](http://docs.google.com/java/awt/Component.html#location()), [lostFocus](http://docs.google.com/java/awt/Component.html#lostFocus(java.awt.Event,%20java.lang.Object)), [mouseDown](http://docs.google.com/java/awt/Component.html#mouseDown(java.awt.Event,%20int,%20int)), [mouseDrag](http://docs.google.com/java/awt/Component.html#mouseDrag(java.awt.Event,%20int,%20int)), [mouseEnter](http://docs.google.com/java/awt/Component.html#mouseEnter(java.awt.Event,%20int,%20int)), [mouseExit](http://docs.google.com/java/awt/Component.html#mouseExit(java.awt.Event,%20int,%20int)), [mouseMove](http://docs.google.com/java/awt/Component.html#mouseMove(java.awt.Event,%20int,%20int)), [mouseUp](http://docs.google.com/java/awt/Component.html#mouseUp(java.awt.Event,%20int,%20int)), [move](http://docs.google.com/java/awt/Component.html#move(int,%20int)), [nextFocus](http://docs.google.com/java/awt/Component.html#nextFocus()), [paintAll](http://docs.google.com/java/awt/Component.html#paintAll(java.awt.Graphics)), [postEvent](http://docs.google.com/java/awt/Component.html#postEvent(java.awt.Event)), [prepareImage](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20java.awt.image.ImageObserver)), [prepareImage](http://docs.google.com/java/awt/Component.html#prepareImage(java.awt.Image,%20int,%20int,%20java.awt.image.ImageObserver)), [processComponentEvent](http://docs.google.com/java/awt/Component.html#processComponentEvent(java.awt.event.ComponentEvent)), [processFocusEvent](http://docs.google.com/java/awt/Component.html#processFocusEvent(java.awt.event.FocusEvent)), [processHierarchyBoundsEvent](http://docs.google.com/java/awt/Component.html#processHierarchyBoundsEvent(java.awt.event.HierarchyEvent)), [processHierarchyEvent](http://docs.google.com/java/awt/Component.html#processHierarchyEvent(java.awt.event.HierarchyEvent)), [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)), [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** |
| --- |

### W3C\_LENGTH\_UNITS

public static final [String](http://docs.google.com/java/lang/String.html) **W3C\_LENGTH\_UNITS**

Key for a client property used to indicate whether  [w3c compliant](http://www.w3.org/TR/CSS21/syndata.html#length-units) length units are used for html rendering.

By default this is not enabled; to enable it set the client [property](http://docs.google.com/javax/swing/JComponent.html#putClientProperty(java.lang.Object,%20java.lang.Object)) with this name to Boolean.TRUE.

**Since:** 1.5 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JEditorPane.W3C_LENGTH_UNITS)

### HONOR\_DISPLAY\_PROPERTIES

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

Key for a client property used to indicate whether the default font and foreground color from the component are used if a font or foreground color is not specified in the styled text.

The default varies based on the look and feel; to enable it set the client [property](http://docs.google.com/javax/swing/JComponent.html#putClientProperty(java.lang.Object,%20java.lang.Object)) with this name to Boolean.TRUE.

**Since:** 1.5 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.swing.JEditorPane.HONOR_DISPLAY_PROPERTIES)

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

### JEditorPane

public **JEditorPane**()

Creates a new JEditorPane. The document model is set to null.

### JEditorPane

public **JEditorPane**([URL](http://docs.google.com/java/net/URL.html) initialPage)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Creates a JEditorPane based on a specified URL for input.

**Parameters:**initialPage - the URL **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if the URL is null or cannot be accessed

### JEditorPane

public **JEditorPane**([String](http://docs.google.com/java/lang/String.html) url)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Creates a JEditorPane based on a string containing a URL specification.

**Parameters:**url - the URL **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - if the URL is null or cannot be accessed

### JEditorPane

public **JEditorPane**([String](http://docs.google.com/java/lang/String.html) type,  
 [String](http://docs.google.com/java/lang/String.html) text)

Creates a JEditorPane that has been initialized to the given text. This is a convenience constructor that calls the setContentType and setText methods.

**Parameters:**type - mime type of the given texttext - the text to initialize with; may be null **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the type parameter is null

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

### addHyperlinkListener

public void **addHyperlinkListener**([HyperlinkListener](http://docs.google.com/javax/swing/event/HyperlinkListener.html) listener)

Adds a hyperlink listener for notification of any changes, for example when a link is selected and entered.

**Parameters:**listener - the listener

### removeHyperlinkListener

public void **removeHyperlinkListener**([HyperlinkListener](http://docs.google.com/javax/swing/event/HyperlinkListener.html) listener)

Removes a hyperlink listener.

**Parameters:**listener - the listener

### getHyperlinkListeners

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

Returns an array of all the HyperLinkListeners added to this JEditorPane with addHyperlinkListener().

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

### fireHyperlinkUpdate

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

Notifies all listeners that have registered interest for notification on this event type. This is normally called by the currently installed EditorKit if a content type that supports hyperlinks is currently active and there was activity with a link. The listener list is processed last to first.

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

### setPage

public void **setPage**([URL](http://docs.google.com/java/net/URL.html) page)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Sets the current URL being displayed. The content type of the pane is set, and if the editor kit for the pane is non-null, then a new default document is created and the URL is read into it. If the URL contains and reference location, the location will be scrolled to by calling the scrollToReference method. If the desired URL is the one currently being displayed, the document will not be reloaded. To force a document reload it is necessary to clear the stream description property of the document. The following code shows how this can be done:

Document doc = jEditorPane.getDocument();  
 doc.putProperty(Document.StreamDescriptionProperty, null);

If the desired URL is not the one currently being displayed, the getStream method is called to give subclasses control over the stream provided.

This may load either synchronously or asynchronously depending upon the document returned by the EditorKit. If the Document is of type AbstractDocument and has a value returned by AbstractDocument.getAsynchronousLoadPriority that is greater than or equal to zero, the page will be loaded on a separate thread using that priority.

If the document is loaded synchronously, it will be filled in with the stream prior to being installed into the editor with a call to setDocument, which is bound and will fire a property change event. If an IOException is thrown the partially loaded document will be discarded and neither the document or page property change events will be fired. If the document is successfully loaded and installed, a view will be built for it by the UI which will then be scrolled if necessary, and then the page property change event will be fired.

If the document is loaded asynchronously, the document will be installed into the editor immediately using a call to setDocument which will fire a document property change event, then a thread will be created which will begin doing the actual loading. In this case, the page property change event will not be fired by the call to this method directly, but rather will be fired when the thread doing the loading has finished. It will also be fired on the event-dispatch thread. Since the calling thread can not throw an IOException in the event of failure on the other thread, the page property change event will be fired when the other thread is done whether the load was successful or not.

**Parameters:**page - the URL of the page **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - for a null or invalid page specification, or exception from the stream being read**See Also:**[getPage()](http://docs.google.com/javax/swing/JEditorPane.html#getPage())

### read

public void **read**([InputStream](http://docs.google.com/java/io/InputStream.html) in,  
 [Object](http://docs.google.com/java/lang/Object.html) desc)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

This method initializes from a stream. If the kit is set to be of type HTMLEditorKit, and the desc parameter is an HTMLDocument, then it invokes the HTMLEditorKit to initiate the read. Otherwise it calls the superclass method which loads the model as plain text.

**Parameters:**in - the stream from which to readdesc - an object describing the stream **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - as thrown by the stream being used to initialize**See Also:**[JTextComponent.read(java.io.Reader, java.lang.Object)](http://docs.google.com/javax/swing/text/JTextComponent.html#read(java.io.Reader,%20java.lang.Object)), [JTextComponent.setDocument(javax.swing.text.Document)](http://docs.google.com/javax/swing/text/JTextComponent.html#setDocument(javax.swing.text.Document))

### getStream

protected [InputStream](http://docs.google.com/java/io/InputStream.html) **getStream**([URL](http://docs.google.com/java/net/URL.html) page)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Fetches a stream for the given URL, which is about to be loaded by the setPage method. By default, this simply opens the URL and returns the stream. This can be reimplemented to do useful things like fetch the stream from a cache, monitor the progress of the stream, etc.

This method is expected to have the the side effect of establishing the content type, and therefore setting the appropriate EditorKit to use for loading the stream.

If this the stream was an http connection, redirects will be followed and the resulting URL will be set as the Document.StreamDescriptionProperty so that relative URL's can be properly resolved.

**Parameters:**page - the URL of the page **Throws:** [IOException](http://docs.google.com/java/io/IOException.html)

### scrollToReference

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

Scrolls the view to the given reference location (that is, the value returned by the UL.getRef method for the URL being displayed). By default, this method only knows how to locate a reference in an HTMLDocument. The implementation calls the scrollRectToVisible method to accomplish the actual scrolling. If scrolling to a reference location is needed for document types other than HTML, this method should be reimplemented. This method will have no effect if the component is not visible.

**Parameters:**reference - the named location to scroll to

### getPage

public [URL](http://docs.google.com/java/net/URL.html) **getPage**()

Gets the current URL being displayed. If a URL was not specified in the creation of the document, this will return null, and relative URL's will not be resolved.

**Returns:**the URL, or null if none

### setPage

public void **setPage**([String](http://docs.google.com/java/lang/String.html) url)  
 throws [IOException](http://docs.google.com/java/io/IOException.html)

Sets the current URL being displayed.

**Parameters:**url - the URL for display **Throws:** [IOException](http://docs.google.com/java/io/IOException.html) - for a null or invalid URL specification

### getUIClassID

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

Gets the class ID for the UI.

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

### createDefaultEditorKit

protected [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) **createDefaultEditorKit**()

Creates the default editor kit (PlainEditorKit) for when the component is first created.

**Returns:**the editor kit

### getEditorKit

public [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) **getEditorKit**()

Fetches the currently installed kit for handling content. createDefaultEditorKit is called to set up a default if necessary.

**Returns:**the editor kit

### getContentType

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

Gets the type of content that this editor is currently set to deal with. This is defined to be the type associated with the currently installed EditorKit.

**Returns:**the content type, null if no editor kit set

### setContentType

public final void **setContentType**([String](http://docs.google.com/java/lang/String.html) type)

Sets the type of content that this editor handles. This calls getEditorKitForContentType, and then setEditorKit if an editor kit can be successfully located. This is mostly convenience method that can be used as an alternative to calling setEditorKit directly.

If there is a charset definition specified as a parameter of the content type specification, it will be used when loading input streams using the associated EditorKit. For example if the type is specified as text/html; charset=EUC-JP the content will be loaded using the EditorKit registered for text/html and the Reader provided to the EditorKit to load unicode into the document will use the EUC-JP charset for translating to unicode. If the type is not recognized, the content will be loaded using the EditorKit registered for plain text, text/plain.

**Parameters:**type - the non-null mime type for the content editing support **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the type parameter is null**See Also:**[getContentType()](http://docs.google.com/javax/swing/JEditorPane.html#getContentType())

### setEditorKit

public void **setEditorKit**([EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) kit)

Sets the currently installed kit for handling content. This is the bound property that establishes the content type of the editor. Any old kit is first deinstalled, then if kit is non-null, the new kit is installed, and a default document created for it. A PropertyChange event ("editorKit") is always fired when setEditorKit is called.

*NOTE: This has the side effect of changing the model, because the EditorKit is the source of how a particular type of content is modeled. This method will cause setDocument to be called on behalf of the caller to ensure integrity of the internal state.*

**Parameters:**kit - the desired editor behavior**See Also:**[getEditorKit()](http://docs.google.com/javax/swing/JEditorPane.html#getEditorKit())

### getEditorKitForContentType

public [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) **getEditorKitForContentType**([String](http://docs.google.com/java/lang/String.html) type)

Fetches the editor kit to use for the given type of content. This is called when a type is requested that doesn't match the currently installed type. If the component doesn't have an EditorKit registered for the given type, it will try to create an EditorKit from the default EditorKit registry. If that fails, a PlainEditorKit is used on the assumption that all text documents can be represented as plain text.

This method can be reimplemented to use some other kind of type registry. This can be reimplemented to use the Java Activation Framework, for example.

**Parameters:**type - the non-null content type **Returns:**the editor kit

### setEditorKitForContentType

public void **setEditorKitForContentType**([String](http://docs.google.com/java/lang/String.html) type,  
 [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) k)

Directly sets the editor kit to use for the given type. A look-and-feel implementation might use this in conjunction with createEditorKitForContentType to install handlers for content types with a look-and-feel bias.

**Parameters:**type - the non-null content typek - the editor kit to be set

### replaceSelection

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

Replaces the currently selected content with new content represented by the given string. If there is no selection this amounts to an insert of the given text. If there is no replacement text (i.e. the content string is empty or null) this amounts to a removal of the current selection. The replacement text will have the attributes currently defined for input. If the component is not editable, beep and return.

This method is thread safe, although most Swing methods are not. Please see [How to Use Threads](http://java.sun.com/docs/books/tutorial/uiswing/misc/threads.html) for more information.

**Overrides:**[replaceSelection](http://docs.google.com/javax/swing/text/JTextComponent.html#replaceSelection(java.lang.String)) in class [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html) **Parameters:**content - the content to replace the selection with. This value can be null

### createEditorKitForContentType

public static [EditorKit](http://docs.google.com/javax/swing/text/EditorKit.html) **createEditorKitForContentType**([String](http://docs.google.com/java/lang/String.html) type)

Creates a handler for the given type from the default registry of editor kits. The registry is created if necessary. If the registered class has not yet been loaded, an attempt is made to dynamically load the prototype of the kit for the given type. If the type was registered with a ClassLoader, that ClassLoader will be used to load the prototype. If there was no registered ClassLoader, Class.forName will be used to load the prototype.

Once a prototype EditorKit instance is successfully located, it is cloned and the clone is returned.

**Parameters:**type - the content type **Returns:**the editor kit, or null if there is nothing registered for the given type

### registerEditorKitForContentType

public static void **registerEditorKitForContentType**([String](http://docs.google.com/java/lang/String.html) type,  
 [String](http://docs.google.com/java/lang/String.html) classname)

Establishes the default bindings of type to classname. The class will be dynamically loaded later when actually needed, and can be safely changed before attempted uses to avoid loading unwanted classes. The prototype EditorKit will be loaded with Class.forName when registered with this method.

**Parameters:**type - the non-null content typeclassname - the class to load later

### registerEditorKitForContentType

public static void **registerEditorKitForContentType**([String](http://docs.google.com/java/lang/String.html) type,  
 [String](http://docs.google.com/java/lang/String.html) classname,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)

Establishes the default bindings of type to classname. The class will be dynamically loaded later when actually needed using the given ClassLoader, and can be safely changed before attempted uses to avoid loading unwanted classes.

**Parameters:**type - the non-null content typeclassname - the class to load laterloader - the ClassLoader to use to load the name

### getEditorKitClassNameForContentType

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

Returns the currently registered EditorKit class name for the type type.

**Parameters:**type - the non-null content type**Since:** 1.3

### getPreferredSize

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

Returns the preferred size for the JEditorPane. The preferred size for JEditorPane is slightly altered from the preferred size of the superclass. If the size of the viewport has become smaller than the minimum size of the component, the scrollable definition for tracking width or height will turn to false. The default viewport layout will give the preferred size, and that is not desired in the case where the scrollable is tracking. In that case the *normal* preferred size is adjusted to the minimum size. This allows things like HTML tables to shrink down to their minimum size and then be laid out at their minimum size, refusing to shrink any further.

**Overrides:**[getPreferredSize](http://docs.google.com/javax/swing/JComponent.html#getPreferredSize()) in class [JComponent](http://docs.google.com/javax/swing/JComponent.html) **Returns:**a Dimension containing the preferred size**See Also:**[JComponent.setPreferredSize(java.awt.Dimension)](http://docs.google.com/javax/swing/JComponent.html#setPreferredSize(java.awt.Dimension)), [ComponentUI](http://docs.google.com/javax/swing/plaf/ComponentUI.html)

### setText

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

Sets the text of this TextComponent to the specified content, which is expected to be in the format of the content type of this editor. For example, if the type is set to text/html the string should be specified in terms of HTML.

This is implemented to remove the contents of the current document, and replace them by parsing the given string using the current EditorKit. This gives the semantics of the superclass by not changing out the model, while supporting the content type currently set on this component. The assumption is that the previous content is relatively small, and that the previous content doesn't have side effects. Both of those assumptions can be violated and cause undesirable results. To avoid this, create a new document, getEditorKit().createDefaultDocument(), and replace the existing Document with the new one. You are then assured the previous Document won't have any lingering state.

1. Leaving the existing model in place means that the old view will be torn down, and a new view created, where replacing the document would avoid the tear down of the old view.
2. Some formats (such as HTML) can install things into the document that can influence future contents. HTML can have style information embedded that would influence the next content installed unexpectedly.

An alternative way to load this component with a string would be to create a StringReader and call the read method. In this case the model would be replaced after it was initialized with the contents of the string.

This method is thread safe, although most Swing methods are not. Please see [How to Use Threads](http://java.sun.com/docs/books/tutorial/uiswing/misc/threads.html) for more information.

**Overrides:**[setText](http://docs.google.com/javax/swing/text/JTextComponent.html#setText(java.lang.String)) in class [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html) **Parameters:**t - the new text to be set; if null the old text will be deleted**See Also:**[getText()](http://docs.google.com/javax/swing/JEditorPane.html#getText())

### getText

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

Returns the text contained in this TextComponent in terms of the content type of this editor. If an exception is thrown while attempting to retrieve the text, null will be returned. This is implemented to call JTextComponent.write with a StringWriter.

**Overrides:**[getText](http://docs.google.com/javax/swing/text/JTextComponent.html#getText()) in class [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html) **Returns:**the text**See Also:**[setText(java.lang.String)](http://docs.google.com/javax/swing/JEditorPane.html#setText(java.lang.String))

### getScrollableTracksViewportWidth

public boolean **getScrollableTracksViewportWidth**()

Returns true if a viewport should always force the width of this Scrollable to match the width of the viewport.

**Specified by:**[getScrollableTracksViewportWidth](http://docs.google.com/javax/swing/Scrollable.html#getScrollableTracksViewportWidth()) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html)**Overrides:**[getScrollableTracksViewportWidth](http://docs.google.com/javax/swing/text/JTextComponent.html#getScrollableTracksViewportWidth()) in class [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html) **Returns:**true if a viewport should force the Scrollables width to match its own, false otherwise

### getScrollableTracksViewportHeight

public boolean **getScrollableTracksViewportHeight**()

Returns true if a viewport should always force the height of this Scrollable to match the height of the viewport.

**Specified by:**[getScrollableTracksViewportHeight](http://docs.google.com/javax/swing/Scrollable.html#getScrollableTracksViewportHeight()) in interface [Scrollable](http://docs.google.com/javax/swing/Scrollable.html)**Overrides:**[getScrollableTracksViewportHeight](http://docs.google.com/javax/swing/text/JTextComponent.html#getScrollableTracksViewportHeight()) in class [JTextComponent](http://docs.google.com/javax/swing/text/JTextComponent.html) **Returns:**true if a viewport should force the Scrollable's height to match its own, false otherwise

### paramString

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

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

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

### getAccessibleContext

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

Gets the AccessibleContext associated with this JEditorPane. For editor panes, the AccessibleContext takes the form of an AccessibleJEditorPane. A new AccessibleJEditorPane instance is created if necessary.

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

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JEditorPane.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/JDialog.AccessibleJDialog.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/JEditorPane.AccessibleJEditorPane.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/JEditorPane.html)    [**NO FRAMES**](http://docs.google.com/JEditorPane.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#4d34og8) | [CONSTR](#lnxbz9) | [METHOD](#35nkun2) | DETAIL: [FIELD](#1y810tw) | [CONSTR](#1ci93xb) | [METHOD](#1pxezwc) |

[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).