| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/GlyphView.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/text/GapContent.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/GlyphView.html)    [**NO FRAMES**](http://docs.google.com/GlyphView.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#tyjcwt) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#lnxbz9) |

## **javax.swing.text**

Class GlyphView

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [javax.swing.text.View](http://docs.google.com/javax/swing/text/View.html)  
 **javax.swing.text.GlyphView**

**All Implemented Interfaces:** [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [SwingConstants](http://docs.google.com/javax/swing/SwingConstants.html), [TabableView](http://docs.google.com/javax/swing/text/TabableView.html) **Direct Known Subclasses:** [LabelView](http://docs.google.com/javax/swing/text/LabelView.html)

public class **GlyphView**extends [View](http://docs.google.com/javax/swing/text/View.html)implements [TabableView](http://docs.google.com/javax/swing/text/TabableView.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html)

A GlyphView is a styled chunk of text that represents a view mapped over an element in the text model. This view is generally responsible for displaying text glyphs using character level attributes in some way. An implementation of the GlyphPainter class is used to do the actual rendering and model/view translations. This separates rendering from layout and management of the association with the model.

The view supports breaking for the purpose of formatting. The fragments produced by breaking share the view that has primary responsibility for the element (i.e. they are nested classes and carry only a small amount of state of their own) so they can share its resources.

Since this view represents text that may have tabs embedded in it, it implements the TabableView interface. Tabs will only be expanded if this view is embedded in a container that does tab expansion. ParagraphView is an example of a container that does tab expansion.

**Since:** 1.3

| **Nested Class Summary** | |
| --- | --- |
| static class | [**GlyphView.GlyphPainter**](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html)            A class to perform rendering of the glyphs. |

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

| **Fields inherited from class javax.swing.text.**[**View**](http://docs.google.com/javax/swing/text/View.html) |
| --- |
| [BadBreakWeight](http://docs.google.com/javax/swing/text/View.html#BadBreakWeight), [ExcellentBreakWeight](http://docs.google.com/javax/swing/text/View.html#ExcellentBreakWeight), [ForcedBreakWeight](http://docs.google.com/javax/swing/text/View.html#ForcedBreakWeight), [GoodBreakWeight](http://docs.google.com/javax/swing/text/View.html#GoodBreakWeight), [X\_AXIS](http://docs.google.com/javax/swing/text/View.html#X_AXIS), [Y\_AXIS](http://docs.google.com/javax/swing/text/View.html#Y_AXIS) |

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

| **Constructor Summary** | |
| --- | --- |
| [**GlyphView**](http://docs.google.com/javax/swing/text/GlyphView.html#GlyphView(javax.swing.text.Element))([Element](http://docs.google.com/javax/swing/text/Element.html) elem)            Constructs a new view wrapped on an element. |

| **Method Summary** | |
| --- | --- |
| [View](http://docs.google.com/javax/swing/text/View.html) | [**breakView**](http://docs.google.com/javax/swing/text/GlyphView.html#breakView(int,%20int,%20float,%20float))(int axis, int p0, float pos, float len)            Breaks this view on the given axis at the given length. |
| void | [**changedUpdate**](http://docs.google.com/javax/swing/text/GlyphView.html#changedUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) e, [Shape](http://docs.google.com/java/awt/Shape.html) a, [ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)            Gives notification from the document that attributes were changed in a location that this view is responsible for. |
| protected  void | [**checkPainter**](http://docs.google.com/javax/swing/text/GlyphView.html#checkPainter())()            Check to see that a glyph painter exists. |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/javax/swing/text/GlyphView.html#clone())()            Creates a shallow copy. |
| [View](http://docs.google.com/javax/swing/text/View.html) | [**createFragment**](http://docs.google.com/javax/swing/text/GlyphView.html#createFragment(int,%20int))(int p0, int p1)            Creates a view that represents a portion of the element. |
| float | [**getAlignment**](http://docs.google.com/javax/swing/text/GlyphView.html#getAlignment(int))(int axis)            Determines the desired alignment for this view along an axis. |
| [Color](http://docs.google.com/java/awt/Color.html) | [**getBackground**](http://docs.google.com/javax/swing/text/GlyphView.html#getBackground())()            Fetch the background color to use to render the glyphs. |
| int | [**getBreakWeight**](http://docs.google.com/javax/swing/text/GlyphView.html#getBreakWeight(int,%20float,%20float))(int axis, float pos, float len)            Determines how attractive a break opportunity in this view is. |
| int | [**getEndOffset**](http://docs.google.com/javax/swing/text/GlyphView.html#getEndOffset())()            Fetches the portion of the model that this view is responsible for. |
| [Font](http://docs.google.com/java/awt/Font.html) | [**getFont**](http://docs.google.com/javax/swing/text/GlyphView.html#getFont())()            Fetch the font that the glyphs should be based upon. |
| [Color](http://docs.google.com/java/awt/Color.html) | [**getForeground**](http://docs.google.com/javax/swing/text/GlyphView.html#getForeground())()            Fetch the foreground color to use to render the glyphs. |
| [GlyphView.GlyphPainter](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html) | [**getGlyphPainter**](http://docs.google.com/javax/swing/text/GlyphView.html#getGlyphPainter())()            Fetch the currently installed glyph painter. |
| int | [**getNextVisualPositionFrom**](http://docs.google.com/javax/swing/text/GlyphView.html#getNextVisualPositionFrom(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D))(int pos, [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) b, [Shape](http://docs.google.com/java/awt/Shape.html) a, int direction, [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html)[] biasRet)            Provides a way to determine the next visually represented model location that one might place a caret. |
| float | [**getPartialSpan**](http://docs.google.com/javax/swing/text/GlyphView.html#getPartialSpan(int,%20int))(int p0, int p1)            Determines the span along the same axis as tab expansion for a portion of the view. |
| float | [**getPreferredSpan**](http://docs.google.com/javax/swing/text/GlyphView.html#getPreferredSpan(int))(int axis)            Determines the preferred span for this view along an axis. |
| int | [**getStartOffset**](http://docs.google.com/javax/swing/text/GlyphView.html#getStartOffset())()            Fetches the portion of the model that this view is responsible for. |
| float | [**getTabbedSpan**](http://docs.google.com/javax/swing/text/GlyphView.html#getTabbedSpan(float,%20javax.swing.text.TabExpander))(float x, [TabExpander](http://docs.google.com/javax/swing/text/TabExpander.html) e)            Determines the desired span when using the given tab expansion implementation. |
| [TabExpander](http://docs.google.com/javax/swing/text/TabExpander.html) | [**getTabExpander**](http://docs.google.com/javax/swing/text/GlyphView.html#getTabExpander())()            Fetch the TabExpander to use if tabs are present in this view. |
| [Segment](http://docs.google.com/javax/swing/text/Segment.html) | [**getText**](http://docs.google.com/javax/swing/text/GlyphView.html#getText(int,%20int))(int p0, int p1)            Fetch a reference to the text that occupies the given range. |
| void | [**insertUpdate**](http://docs.google.com/javax/swing/text/GlyphView.html#insertUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) e, [Shape](http://docs.google.com/java/awt/Shape.html) a, [ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)            Gives notification that something was inserted into the document in a location that this view is responsible for. |
| boolean | [**isStrikeThrough**](http://docs.google.com/javax/swing/text/GlyphView.html#isStrikeThrough())()            Determine if the glyphs should have a strikethrough line. |
| boolean | [**isSubscript**](http://docs.google.com/javax/swing/text/GlyphView.html#isSubscript())()            Determine if the glyphs should be rendered as superscript. |
| boolean | [**isSuperscript**](http://docs.google.com/javax/swing/text/GlyphView.html#isSuperscript())()            Determine if the glyphs should be rendered as subscript. |
| boolean | [**isUnderline**](http://docs.google.com/javax/swing/text/GlyphView.html#isUnderline())()            Determine if the glyphs should be underlined. |
| [Shape](http://docs.google.com/java/awt/Shape.html) | [**modelToView**](http://docs.google.com/javax/swing/text/GlyphView.html#modelToView(int,%20java.awt.Shape,%20javax.swing.text.Position.Bias))(int pos, [Shape](http://docs.google.com/java/awt/Shape.html) a, [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) b)            Provides a mapping from the document model coordinate space to the coordinate space of the view mapped to it. |
| void | [**paint**](http://docs.google.com/javax/swing/text/GlyphView.html#paint(java.awt.Graphics,%20java.awt.Shape))([Graphics](http://docs.google.com/java/awt/Graphics.html) g, [Shape](http://docs.google.com/java/awt/Shape.html) a)            Renders a portion of a text style run. |
| void | [**removeUpdate**](http://docs.google.com/javax/swing/text/GlyphView.html#removeUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) e, [Shape](http://docs.google.com/java/awt/Shape.html) a, [ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)            Gives notification that something was removed from the document in a location that this view is responsible for. |
| void | [**setGlyphPainter**](http://docs.google.com/javax/swing/text/GlyphView.html#setGlyphPainter(javax.swing.text.GlyphView.GlyphPainter))([GlyphView.GlyphPainter](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html) p)            Sets the painter to use for rendering glyphs. |
| int | [**viewToModel**](http://docs.google.com/javax/swing/text/GlyphView.html#viewToModel(float,%20float,%20java.awt.Shape,%20javax.swing.text.Position.Bias%5B%5D))(float x, float y, [Shape](http://docs.google.com/java/awt/Shape.html) a, [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html)[] biasReturn)            Provides a mapping from the view coordinate space to the logical coordinate space of the model. |

| **Methods inherited from class javax.swing.text.**[**View**](http://docs.google.com/javax/swing/text/View.html) |
| --- |
| [append](http://docs.google.com/javax/swing/text/View.html#append(javax.swing.text.View)), [forwardUpdate](http://docs.google.com/javax/swing/text/View.html#forwardUpdate(javax.swing.event.DocumentEvent.ElementChange,%20javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)), [forwardUpdateToView](http://docs.google.com/javax/swing/text/View.html#forwardUpdateToView(javax.swing.text.View,%20javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)), [getAttributes](http://docs.google.com/javax/swing/text/View.html#getAttributes()), [getChildAllocation](http://docs.google.com/javax/swing/text/View.html#getChildAllocation(int,%20java.awt.Shape)), [getContainer](http://docs.google.com/javax/swing/text/View.html#getContainer()), [getDocument](http://docs.google.com/javax/swing/text/View.html#getDocument()), [getElement](http://docs.google.com/javax/swing/text/View.html#getElement()), [getGraphics](http://docs.google.com/javax/swing/text/View.html#getGraphics()), [getMaximumSpan](http://docs.google.com/javax/swing/text/View.html#getMaximumSpan(int)), [getMinimumSpan](http://docs.google.com/javax/swing/text/View.html#getMinimumSpan(int)), [getParent](http://docs.google.com/javax/swing/text/View.html#getParent()), [getResizeWeight](http://docs.google.com/javax/swing/text/View.html#getResizeWeight(int)), [getToolTipText](http://docs.google.com/javax/swing/text/View.html#getToolTipText(float,%20float,%20java.awt.Shape)), [getView](http://docs.google.com/javax/swing/text/View.html#getView(int)), [getViewCount](http://docs.google.com/javax/swing/text/View.html#getViewCount()), [getViewFactory](http://docs.google.com/javax/swing/text/View.html#getViewFactory()), [getViewIndex](http://docs.google.com/javax/swing/text/View.html#getViewIndex(float,%20float,%20java.awt.Shape)), [getViewIndex](http://docs.google.com/javax/swing/text/View.html#getViewIndex(int,%20javax.swing.text.Position.Bias)), [insert](http://docs.google.com/javax/swing/text/View.html#insert(int,%20javax.swing.text.View)), [isVisible](http://docs.google.com/javax/swing/text/View.html#isVisible()), [modelToView](http://docs.google.com/javax/swing/text/View.html#modelToView(int,%20javax.swing.text.Position.Bias,%20int,%20javax.swing.text.Position.Bias,%20java.awt.Shape)), [modelToView](http://docs.google.com/javax/swing/text/View.html#modelToView(int,%20java.awt.Shape)), [preferenceChanged](http://docs.google.com/javax/swing/text/View.html#preferenceChanged(javax.swing.text.View,%20boolean,%20boolean)), [remove](http://docs.google.com/javax/swing/text/View.html#remove(int)), [removeAll](http://docs.google.com/javax/swing/text/View.html#removeAll()), [replace](http://docs.google.com/javax/swing/text/View.html#replace(int,%20int,%20javax.swing.text.View%5B%5D)), [setParent](http://docs.google.com/javax/swing/text/View.html#setParent(javax.swing.text.View)), [setSize](http://docs.google.com/javax/swing/text/View.html#setSize(float,%20float)), [updateChildren](http://docs.google.com/javax/swing/text/View.html#updateChildren(javax.swing.event.DocumentEvent.ElementChange,%20javax.swing.event.DocumentEvent,%20javax.swing.text.ViewFactory)), [updateLayout](http://docs.google.com/javax/swing/text/View.html#updateLayout(javax.swing.event.DocumentEvent.ElementChange,%20javax.swing.event.DocumentEvent,%20java.awt.Shape)), [viewToModel](http://docs.google.com/javax/swing/text/View.html#viewToModel(float,%20float,%20java.awt.Shape)) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

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

### GlyphView

public **GlyphView**([Element](http://docs.google.com/javax/swing/text/Element.html) elem)

Constructs a new view wrapped on an element.

**Parameters:**elem - the element

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

### clone

protected final [Object](http://docs.google.com/java/lang/Object.html) **clone**()

Creates a shallow copy. This is used by the createFragment and breakView methods.

**Overrides:**[clone](http://docs.google.com/java/lang/Object.html#clone()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**the copy**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### getGlyphPainter

public [GlyphView.GlyphPainter](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html) **getGlyphPainter**()

Fetch the currently installed glyph painter. If a painter has not yet been installed, and a default was not yet needed, null is returned.

### setGlyphPainter

public void **setGlyphPainter**([GlyphView.GlyphPainter](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html) p)

Sets the painter to use for rendering glyphs.

### getText

public [Segment](http://docs.google.com/javax/swing/text/Segment.html) **getText**(int p0,  
 int p1)

Fetch a reference to the text that occupies the given range. This is normally used by the GlyphPainter to determine what characters it should render glyphs for.

**Parameters:**p0 - the starting document offset >= 0p1 - the ending document offset >= p0 **Returns:**the Segment containing the text

### getBackground

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

Fetch the background color to use to render the glyphs. If there is no background color, null should be returned. This is implemented to call StyledDocument.getBackground if the associated document is a styled document, otherwise it returns null.

### getForeground

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

Fetch the foreground color to use to render the glyphs. If there is no foreground color, null should be returned. This is implemented to call StyledDocument.getBackground if the associated document is a StyledDocument. If the associated document is not a StyledDocument, the associated components foreground color is used. If there is no associated component, null is returned.

### getFont

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

Fetch the font that the glyphs should be based upon. This is implemented to call StyledDocument.getFont if the associated document is a StyledDocument. If the associated document is not a StyledDocument, the associated components font is used. If there is no associated component, null is returned.

### isUnderline

public boolean **isUnderline**()

Determine if the glyphs should be underlined. If true, an underline should be drawn through the baseline.

### isStrikeThrough

public boolean **isStrikeThrough**()

Determine if the glyphs should have a strikethrough line. If true, a line should be drawn through the center of the glyphs.

### isSubscript

public boolean **isSubscript**()

Determine if the glyphs should be rendered as superscript.

### isSuperscript

public boolean **isSuperscript**()

Determine if the glyphs should be rendered as subscript.

### getTabExpander

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

Fetch the TabExpander to use if tabs are present in this view.

### checkPainter

protected void **checkPainter**()

Check to see that a glyph painter exists. If a painter doesn't exist, a default glyph painter will be installed.

### getTabbedSpan

public float **getTabbedSpan**(float x,  
 [TabExpander](http://docs.google.com/javax/swing/text/TabExpander.html) e)

Determines the desired span when using the given tab expansion implementation.

**Specified by:**[getTabbedSpan](http://docs.google.com/javax/swing/text/TabableView.html#getTabbedSpan(float,%20javax.swing.text.TabExpander)) in interface [TabableView](http://docs.google.com/javax/swing/text/TabableView.html) **Parameters:**x - the position the view would be located at for the purpose of tab expansion >= 0.e - how to expand the tabs when encountered. **Returns:**the desired span >= 0**See Also:**[TabableView.getTabbedSpan(float, javax.swing.text.TabExpander)](http://docs.google.com/javax/swing/text/TabableView.html#getTabbedSpan(float,%20javax.swing.text.TabExpander))

### getPartialSpan

public float **getPartialSpan**(int p0,  
 int p1)

Determines the span along the same axis as tab expansion for a portion of the view. This is intended for use by the TabExpander for cases where the tab expansion involves aligning the portion of text that doesn't have whitespace relative to the tab stop. There is therefore an assumption that the range given does not contain tabs.

This method can be called while servicing the getTabbedSpan or getPreferredSize. It has to arrange for its own text buffer to make the measurements.

**Specified by:**[getPartialSpan](http://docs.google.com/javax/swing/text/TabableView.html#getPartialSpan(int,%20int)) in interface [TabableView](http://docs.google.com/javax/swing/text/TabableView.html) **Parameters:**p0 - the starting document offset >= 0p1 - the ending document offset >= p0 **Returns:**the span >= 0

### getStartOffset

public int **getStartOffset**()

Fetches the portion of the model that this view is responsible for.

**Overrides:**[getStartOffset](http://docs.google.com/javax/swing/text/View.html#getStartOffset()) in class [View](http://docs.google.com/javax/swing/text/View.html) **Returns:**the starting offset into the model**See Also:**[View.getStartOffset()](http://docs.google.com/javax/swing/text/View.html#getStartOffset())

### getEndOffset

public int **getEndOffset**()

Fetches the portion of the model that this view is responsible for.

**Overrides:**[getEndOffset](http://docs.google.com/javax/swing/text/View.html#getEndOffset()) in class [View](http://docs.google.com/javax/swing/text/View.html) **Returns:**the ending offset into the model**See Also:**[View.getEndOffset()](http://docs.google.com/javax/swing/text/View.html#getEndOffset())

### paint

public void **paint**([Graphics](http://docs.google.com/java/awt/Graphics.html) g,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a)

Renders a portion of a text style run.

**Specified by:**[paint](http://docs.google.com/javax/swing/text/View.html#paint(java.awt.Graphics,%20java.awt.Shape)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**g - the rendering surface to usea - the allocated region to render into

### getPreferredSpan

public float **getPreferredSpan**(int axis)

Determines the preferred span for this view along an axis.

**Specified by:**[getPreferredSpan](http://docs.google.com/javax/swing/text/View.html#getPreferredSpan(int)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**axis - may be either View.X\_AXIS or View.Y\_AXIS **Returns:**the span the view would like to be rendered into >= 0. Typically the view is told to render into the span that is returned, although there is no guarantee. The parent may choose to resize or break the view.**See Also:**[View.getPreferredSpan(int)](http://docs.google.com/javax/swing/text/View.html#getPreferredSpan(int))

### getAlignment

public float **getAlignment**(int axis)

Determines the desired alignment for this view along an axis. For the label, the alignment is along the font baseline for the y axis, and the superclasses alignment along the x axis.

**Overrides:**[getAlignment](http://docs.google.com/javax/swing/text/View.html#getAlignment(int)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**axis - may be either View.X\_AXIS or View.Y\_AXIS **Returns:**the desired alignment. This should be a value between 0.0 and 1.0 inclusive, where 0 indicates alignment at the origin and 1.0 indicates alignment to the full span away from the origin. An alignment of 0.5 would be the center of the view.

### modelToView

public [Shape](http://docs.google.com/java/awt/Shape.html) **modelToView**(int pos,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a,  
 [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) b)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Provides a mapping from the document model coordinate space to the coordinate space of the view mapped to it.

**Specified by:**[modelToView](http://docs.google.com/javax/swing/text/View.html#modelToView(int,%20java.awt.Shape,%20javax.swing.text.Position.Bias)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**pos - the position to convert >= 0a - the allocated region to render intob - either Position.Bias.Forward or Position.Bias.Backward **Returns:**the bounding box of the given position **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) - if the given position does not represent a valid location in the associated document**See Also:**[View.modelToView(int, java.awt.Shape, javax.swing.text.Position.Bias)](http://docs.google.com/javax/swing/text/View.html#modelToView(int,%20java.awt.Shape,%20javax.swing.text.Position.Bias))

### viewToModel

public int **viewToModel**(float x,  
 float y,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a,  
 [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html)[] biasReturn)

Provides a mapping from the view coordinate space to the logical coordinate space of the model.

**Specified by:**[viewToModel](http://docs.google.com/javax/swing/text/View.html#viewToModel(float,%20float,%20java.awt.Shape,%20javax.swing.text.Position.Bias%5B%5D)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**x - the X coordinate >= 0y - the Y coordinate >= 0a - the allocated region to render intobiasReturn - either Position.Bias.Forward or Position.Bias.Backward is returned as the zero-th element of this array **Returns:**the location within the model that best represents the given point of view >= 0**See Also:**[View.viewToModel(float, float, java.awt.Shape, javax.swing.text.Position.Bias[])](http://docs.google.com/javax/swing/text/View.html#viewToModel(float,%20float,%20java.awt.Shape,%20javax.swing.text.Position.Bias%5B%5D))

### getBreakWeight

public int **getBreakWeight**(int axis,  
 float pos,  
 float len)

Determines how attractive a break opportunity in this view is. This can be used for determining which view is the most attractive to call breakView on in the process of formatting. The higher the weight, the more attractive the break. A value equal to or lower than View.BadBreakWeight should not be considered for a break. A value greater than or equal to View.ForcedBreakWeight should be broken.

This is implemented to forward to the superclass for the Y\_AXIS. Along the X\_AXIS the following values may be returned.

**View.ExcellentBreakWeight** if there is whitespace proceeding the desired break location. **View.BadBreakWeight** if the desired break location results in a break location of the starting offset. **View.GoodBreakWeight** if the other conditions don't occur. This will normally result in the behavior of breaking on a whitespace location if one can be found, otherwise breaking between characters.

**Overrides:**[getBreakWeight](http://docs.google.com/javax/swing/text/View.html#getBreakWeight(int,%20float,%20float)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**axis - may be either View.X\_AXIS or View.Y\_AXISpos - the potential location of the start of the broken view >= 0. This may be useful for calculating tab positions.len - specifies the relative length from *pos* where a potential break is desired >= 0. **Returns:**the weight, which should be a value between View.ForcedBreakWeight and View.BadBreakWeight.**See Also:**[LabelView](http://docs.google.com/javax/swing/text/LabelView.html), [ParagraphView](http://docs.google.com/javax/swing/text/ParagraphView.html), [View.BadBreakWeight](http://docs.google.com/javax/swing/text/View.html#BadBreakWeight), [View.GoodBreakWeight](http://docs.google.com/javax/swing/text/View.html#GoodBreakWeight), [View.ExcellentBreakWeight](http://docs.google.com/javax/swing/text/View.html#ExcellentBreakWeight), [View.ForcedBreakWeight](http://docs.google.com/javax/swing/text/View.html#ForcedBreakWeight)

### breakView

public [View](http://docs.google.com/javax/swing/text/View.html) **breakView**(int axis,  
 int p0,  
 float pos,  
 float len)

Breaks this view on the given axis at the given length. This is implemented to attempt to break on a whitespace location, and returns a fragment with the whitespace at the end. If a whitespace location can't be found, the nearest character is used.

**Overrides:**[breakView](http://docs.google.com/javax/swing/text/View.html#breakView(int,%20int,%20float,%20float)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**axis - may be either View.X\_AXIS or View.Y\_AXISp0 - the location in the model where the fragment should start it's representation >= 0.pos - the position along the axis that the broken view would occupy >= 0. This may be useful for things like tab calculations.len - specifies the distance along the axis where a potential break is desired >= 0. **Returns:**the fragment of the view that represents the given span, if the view can be broken. If the view doesn't support breaking behavior, the view itself is returned.**See Also:**[View.breakView(int, int, float, float)](http://docs.google.com/javax/swing/text/View.html#breakView(int,%20int,%20float,%20float))

### createFragment

public [View](http://docs.google.com/javax/swing/text/View.html) **createFragment**(int p0,  
 int p1)

Creates a view that represents a portion of the element. This is potentially useful during formatting operations for taking measurements of fragments of the view. If the view doesn't support fragmenting (the default), it should return itself.

This view does support fragmenting. It is implemented to return a nested class that shares state in this view representing only a portion of the view.

**Overrides:**[createFragment](http://docs.google.com/javax/swing/text/View.html#createFragment(int,%20int)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**p0 - the starting offset >= 0. This should be a value greater or equal to the element starting offset and less than the element ending offset.p1 - the ending offset > p0. This should be a value less than or equal to the elements end offset and greater than the elements starting offset. **Returns:**the view fragment, or itself if the view doesn't support breaking into fragments**See Also:**[LabelView](http://docs.google.com/javax/swing/text/LabelView.html)

### getNextVisualPositionFrom

public int **getNextVisualPositionFrom**(int pos,  
 [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) b,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a,  
 int direction,  
 [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html)[] biasRet)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Provides a way to determine the next visually represented model location that one might place a caret. Some views may not be visible, they might not be in the same order found in the model, or they just might not allow access to some of the locations in the model.

**Overrides:**[getNextVisualPositionFrom](http://docs.google.com/javax/swing/text/View.html#getNextVisualPositionFrom(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**pos - the position to convert >= 0a - the allocated region to render intodirection - the direction from the current position that can be thought of as the arrow keys typically found on a keyboard. This may be SwingConstants.WEST, SwingConstants.EAST, SwingConstants.NORTH, or SwingConstants.SOUTH. **Returns:**the location within the model that best represents the next location visual position. **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - for an invalid direction

### insertUpdate

public void **insertUpdate**([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) e,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a,  
 [ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)

Gives notification that something was inserted into the document in a location that this view is responsible for. This is implemented to call preferenceChanged along the axis the glyphs are rendered.

**Overrides:**[insertUpdate](http://docs.google.com/javax/swing/text/View.html#insertUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**e - the change information from the associated documenta - the current allocation of the viewf - the factory to use to rebuild if the view has children**See Also:**[View.insertUpdate(javax.swing.event.DocumentEvent, java.awt.Shape, javax.swing.text.ViewFactory)](http://docs.google.com/javax/swing/text/View.html#insertUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))

### removeUpdate

public void **removeUpdate**([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) e,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a,  
 [ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)

Gives notification that something was removed from the document in a location that this view is responsible for. This is implemented to call preferenceChanged along the axis the glyphs are rendered.

**Overrides:**[removeUpdate](http://docs.google.com/javax/swing/text/View.html#removeUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**e - the change information from the associated documenta - the current allocation of the viewf - the factory to use to rebuild if the view has children**See Also:**[View.removeUpdate(javax.swing.event.DocumentEvent, java.awt.Shape, javax.swing.text.ViewFactory)](http://docs.google.com/javax/swing/text/View.html#removeUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))

### changedUpdate

public void **changedUpdate**([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) e,  
 [Shape](http://docs.google.com/java/awt/Shape.html) a,  
 [ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)

Gives notification from the document that attributes were changed in a location that this view is responsible for. This is implemented to call preferenceChanged along both the horizontal and vertical axis.

**Overrides:**[changedUpdate](http://docs.google.com/javax/swing/text/View.html#changedUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)) in class [View](http://docs.google.com/javax/swing/text/View.html) **Parameters:**e - the change information from the associated documenta - the current allocation of the viewf - the factory to use to rebuild if the view has children**See Also:**[View.changedUpdate(javax.swing.event.DocumentEvent, java.awt.Shape, javax.swing.text.ViewFactory)](http://docs.google.com/javax/swing/text/View.html#changedUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/GlyphView.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/text/GapContent.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/GlyphView.GlyphPainter.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/GlyphView.html)    [**NO FRAMES**](http://docs.google.com/GlyphView.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#tyjcwt) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#lnxbz9) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).