| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ParagraphView.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/NumberFormatter.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/PasswordView.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/ParagraphView.html)    [**NO FRAMES**](http://docs.google.com/ParagraphView.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#tyjcwt) | [CONSTR](#2s8eyo1) | [METHOD](#17dp8vu) | DETAIL: [FIELD](#44sinio) | [CONSTR](#z337ya) | [METHOD](#1y810tw) |

## **javax.swing.text**

Class ParagraphView

[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.CompositeView](http://docs.google.com/javax/swing/text/CompositeView.html)  
 [javax.swing.text.BoxView](http://docs.google.com/javax/swing/text/BoxView.html)  
 [javax.swing.text.FlowView](http://docs.google.com/javax/swing/text/FlowView.html)  
 **javax.swing.text.ParagraphView**

**All Implemented Interfaces:** [SwingConstants](http://docs.google.com/javax/swing/SwingConstants.html), [TabExpander](http://docs.google.com/javax/swing/text/TabExpander.html) **Direct Known Subclasses:** [ParagraphView](http://docs.google.com/javax/swing/text/html/ParagraphView.html)

public class **ParagraphView**extends [FlowView](http://docs.google.com/javax/swing/text/FlowView.html)implements [TabExpander](http://docs.google.com/javax/swing/text/TabExpander.html)

View of a simple line-wrapping paragraph that supports multiple fonts, colors, components, icons, etc. It is basically a vertical box with a margin around it. The contents of the box are a bunch of rows which are special horizontal boxes. This view creates a collection of views that represent the child elements of the paragraph element. Each of these views are placed into a row directly if they will fit, otherwise the breakView method is called to try and carve the view into pieces that fit.

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

| **Nested Class Summary** | |
| --- | --- |

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

| **Field Summary** | |
| --- | --- |
| protected  int | [**firstLineIndent**](http://docs.google.com/javax/swing/text/ParagraphView.html#firstLineIndent)            Indentation for the first line, from the left inset. |

| **Fields inherited from class javax.swing.text.**[**FlowView**](http://docs.google.com/javax/swing/text/FlowView.html) |
| --- |
| [layoutPool](http://docs.google.com/javax/swing/text/FlowView.html#layoutPool), [layoutSpan](http://docs.google.com/javax/swing/text/FlowView.html#layoutSpan), [strategy](http://docs.google.com/javax/swing/text/FlowView.html#strategy) |

| **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** | |
| --- | --- |
| [**ParagraphView**](http://docs.google.com/javax/swing/text/ParagraphView.html#ParagraphView(javax.swing.text.Element))([Element](http://docs.google.com/javax/swing/text/Element.html) elem)            Constructs a ParagraphView for the given element. |

| **Method Summary** | |
| --- | --- |
| protected  void | [**adjustRow**](http://docs.google.com/javax/swing/text/ParagraphView.html#adjustRow(javax.swing.text.ParagraphView.Row,%20int,%20int))(javax.swing.text.ParagraphView.Row r, int desiredSpan, int x)            Adjusts the given row if possible to fit within the layout span. |
| [View](http://docs.google.com/javax/swing/text/View.html) | [**breakView**](http://docs.google.com/javax/swing/text/ParagraphView.html#breakView(int,%20float,%20java.awt.Shape))(int axis, float len, [Shape](http://docs.google.com/java/awt/Shape.html) a)            Breaks this view on the given axis at the given length. |
| void | [**changedUpdate**](http://docs.google.com/javax/swing/text/ParagraphView.html#changedUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory))([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) changes, [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  [View](http://docs.google.com/javax/swing/text/View.html) | [**createRow**](http://docs.google.com/javax/swing/text/ParagraphView.html#createRow())()            Create a View that should be used to hold a a row's worth of children in a flow. |
| protected  int | [**findOffsetToCharactersInString**](http://docs.google.com/javax/swing/text/ParagraphView.html#findOffsetToCharactersInString(char%5B%5D,%20int))(char[] string, int start)            Finds the next character in the document with a character in string, starting at offset start. |
| protected  boolean | [**flipEastAndWestAtEnds**](http://docs.google.com/javax/swing/text/ParagraphView.html#flipEastAndWestAtEnds(int,%20javax.swing.text.Position.Bias))(int position, [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) bias)            Determines in which direction the next view lays. |
| float | [**getAlignment**](http://docs.google.com/javax/swing/text/ParagraphView.html#getAlignment(int))(int axis)            Determines the desired alignment for this view along an axis. |
| int | [**getBreakWeight**](http://docs.google.com/javax/swing/text/ParagraphView.html#getBreakWeight(int,%20float))(int axis, float len)            Gets the break weight for a given location. |
| protected  int | [**getClosestPositionTo**](http://docs.google.com/javax/swing/text/ParagraphView.html#getClosestPositionTo(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D,%20int,%20int))(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, int rowIndex, int x)            Returns the closest model position to x. |
| int | [**getFlowSpan**](http://docs.google.com/javax/swing/text/ParagraphView.html#getFlowSpan(int))(int index)            Fetches the constraining span to flow against for the given child index. |
| int | [**getFlowStart**](http://docs.google.com/javax/swing/text/ParagraphView.html#getFlowStart(int))(int index)            Fetches the location along the flow axis that the flow span will start at. |
| protected  [View](http://docs.google.com/javax/swing/text/View.html) | [**getLayoutView**](http://docs.google.com/javax/swing/text/ParagraphView.html#getLayoutView(int))(int index)            Returns the view at a given index. |
| protected  int | [**getLayoutViewCount**](http://docs.google.com/javax/swing/text/ParagraphView.html#getLayoutViewCount())()            Returns the number of views that this view is responsible for. |
| protected  int | [**getNextNorthSouthVisualPositionFrom**](http://docs.google.com/javax/swing/text/ParagraphView.html#getNextNorthSouthVisualPositionFrom(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)            Returns the next visual position for the cursor, in either the east or west direction. |
| protected  float | [**getPartialSize**](http://docs.google.com/javax/swing/text/ParagraphView.html#getPartialSize(int,%20int))(int startOffset, int endOffset)            Returns the size used by the views between startOffset and endOffset. |
| protected  float | [**getTabBase**](http://docs.google.com/javax/swing/text/ParagraphView.html#getTabBase())()            Returns where the tabs are calculated from. |
| protected  [TabSet](http://docs.google.com/javax/swing/text/TabSet.html) | [**getTabSet**](http://docs.google.com/javax/swing/text/ParagraphView.html#getTabSet())()            Gets the Tabset to be used in calculating tabs. |
| float | [**nextTabStop**](http://docs.google.com/javax/swing/text/ParagraphView.html#nextTabStop(float,%20int))(float x, int tabOffset)            Returns the next tab stop position given a reference position. |
| void | [**paint**](http://docs.google.com/javax/swing/text/ParagraphView.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 using the given rendering surface and area on that surface. |
| protected  void | [**setFirstLineIndent**](http://docs.google.com/javax/swing/text/ParagraphView.html#setFirstLineIndent(float))(float fi)            Sets the indent on the first line. |
| protected  void | [**setJustification**](http://docs.google.com/javax/swing/text/ParagraphView.html#setJustification(int))(int j)            Sets the type of justification. |
| protected  void | [**setLineSpacing**](http://docs.google.com/javax/swing/text/ParagraphView.html#setLineSpacing(float))(float ls)            Sets the line spacing. |
| protected  void | [**setPropertiesFromAttributes**](http://docs.google.com/javax/swing/text/ParagraphView.html#setPropertiesFromAttributes())()            Set the cached properties from the attributes. |

| **Methods inherited from class javax.swing.text.**[**FlowView**](http://docs.google.com/javax/swing/text/FlowView.html) |
| --- |
| [calculateMinorAxisRequirements](http://docs.google.com/javax/swing/text/FlowView.html#calculateMinorAxisRequirements(int,%20javax.swing.SizeRequirements)), [getFlowAxis](http://docs.google.com/javax/swing/text/FlowView.html#getFlowAxis()), [getViewIndexAtPosition](http://docs.google.com/javax/swing/text/FlowView.html#getViewIndexAtPosition(int)), [insertUpdate](http://docs.google.com/javax/swing/text/FlowView.html#insertUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)), [layout](http://docs.google.com/javax/swing/text/FlowView.html#layout(int,%20int)), [loadChildren](http://docs.google.com/javax/swing/text/FlowView.html#loadChildren(javax.swing.text.ViewFactory)), [removeUpdate](http://docs.google.com/javax/swing/text/FlowView.html#removeUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)), [setParent](http://docs.google.com/javax/swing/text/FlowView.html#setParent(javax.swing.text.View)) |

| **Methods inherited from class javax.swing.text.**[**BoxView**](http://docs.google.com/javax/swing/text/BoxView.html) |
| --- |
| [baselineLayout](http://docs.google.com/javax/swing/text/BoxView.html#baselineLayout(int,%20int,%20int%5B%5D,%20int%5B%5D)), [baselineRequirements](http://docs.google.com/javax/swing/text/BoxView.html#baselineRequirements(int,%20javax.swing.SizeRequirements)), [calculateMajorAxisRequirements](http://docs.google.com/javax/swing/text/BoxView.html#calculateMajorAxisRequirements(int,%20javax.swing.SizeRequirements)), [childAllocation](http://docs.google.com/javax/swing/text/BoxView.html#childAllocation(int,%20java.awt.Rectangle)), [forwardUpdate](http://docs.google.com/javax/swing/text/BoxView.html#forwardUpdate(javax.swing.event.DocumentEvent.ElementChange,%20javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)), [getAxis](http://docs.google.com/javax/swing/text/BoxView.html#getAxis()), [getChildAllocation](http://docs.google.com/javax/swing/text/BoxView.html#getChildAllocation(int,%20java.awt.Shape)), [getHeight](http://docs.google.com/javax/swing/text/BoxView.html#getHeight()), [getMaximumSpan](http://docs.google.com/javax/swing/text/BoxView.html#getMaximumSpan(int)), [getMinimumSpan](http://docs.google.com/javax/swing/text/BoxView.html#getMinimumSpan(int)), [getOffset](http://docs.google.com/javax/swing/text/BoxView.html#getOffset(int,%20int)), [getPreferredSpan](http://docs.google.com/javax/swing/text/BoxView.html#getPreferredSpan(int)), [getResizeWeight](http://docs.google.com/javax/swing/text/BoxView.html#getResizeWeight(int)), [getSpan](http://docs.google.com/javax/swing/text/BoxView.html#getSpan(int,%20int)), [getViewAtPoint](http://docs.google.com/javax/swing/text/BoxView.html#getViewAtPoint(int,%20int,%20java.awt.Rectangle)), [getWidth](http://docs.google.com/javax/swing/text/BoxView.html#getWidth()), [isAfter](http://docs.google.com/javax/swing/text/BoxView.html#isAfter(int,%20int,%20java.awt.Rectangle)), [isAllocationValid](http://docs.google.com/javax/swing/text/BoxView.html#isAllocationValid()), [isBefore](http://docs.google.com/javax/swing/text/BoxView.html#isBefore(int,%20int,%20java.awt.Rectangle)), [isLayoutValid](http://docs.google.com/javax/swing/text/BoxView.html#isLayoutValid(int)), [layoutChanged](http://docs.google.com/javax/swing/text/BoxView.html#layoutChanged(int)), [layoutMajorAxis](http://docs.google.com/javax/swing/text/BoxView.html#layoutMajorAxis(int,%20int,%20int%5B%5D,%20int%5B%5D)), [layoutMinorAxis](http://docs.google.com/javax/swing/text/BoxView.html#layoutMinorAxis(int,%20int,%20int%5B%5D,%20int%5B%5D)), [modelToView](http://docs.google.com/javax/swing/text/BoxView.html#modelToView(int,%20java.awt.Shape,%20javax.swing.text.Position.Bias)), [paintChild](http://docs.google.com/javax/swing/text/BoxView.html#paintChild(java.awt.Graphics,%20java.awt.Rectangle,%20int)), [preferenceChanged](http://docs.google.com/javax/swing/text/BoxView.html#preferenceChanged(javax.swing.text.View,%20boolean,%20boolean)), [replace](http://docs.google.com/javax/swing/text/BoxView.html#replace(int,%20int,%20javax.swing.text.View%5B%5D)), [setAxis](http://docs.google.com/javax/swing/text/BoxView.html#setAxis(int)), [setSize](http://docs.google.com/javax/swing/text/BoxView.html#setSize(float,%20float)), [viewToModel](http://docs.google.com/javax/swing/text/BoxView.html#viewToModel(float,%20float,%20java.awt.Shape,%20javax.swing.text.Position.Bias%5B%5D)) |

| **Methods inherited from class javax.swing.text.**[**CompositeView**](http://docs.google.com/javax/swing/text/CompositeView.html) |
| --- |
| [getBottomInset](http://docs.google.com/javax/swing/text/CompositeView.html#getBottomInset()), [getInsideAllocation](http://docs.google.com/javax/swing/text/CompositeView.html#getInsideAllocation(java.awt.Shape)), [getLeftInset](http://docs.google.com/javax/swing/text/CompositeView.html#getLeftInset()), [getNextEastWestVisualPositionFrom](http://docs.google.com/javax/swing/text/CompositeView.html#getNextEastWestVisualPositionFrom(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D)), [getNextVisualPositionFrom](http://docs.google.com/javax/swing/text/CompositeView.html#getNextVisualPositionFrom(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D)), [getRightInset](http://docs.google.com/javax/swing/text/CompositeView.html#getRightInset()), [getTopInset](http://docs.google.com/javax/swing/text/CompositeView.html#getTopInset()), [getView](http://docs.google.com/javax/swing/text/CompositeView.html#getView(int)), [getViewAtPosition](http://docs.google.com/javax/swing/text/CompositeView.html#getViewAtPosition(int,%20java.awt.Rectangle)), [getViewCount](http://docs.google.com/javax/swing/text/CompositeView.html#getViewCount()), [getViewIndex](http://docs.google.com/javax/swing/text/CompositeView.html#getViewIndex(int,%20javax.swing.text.Position.Bias)), [modelToView](http://docs.google.com/javax/swing/text/CompositeView.html#modelToView(int,%20javax.swing.text.Position.Bias,%20int,%20javax.swing.text.Position.Bias,%20java.awt.Shape)), [setInsets](http://docs.google.com/javax/swing/text/CompositeView.html#setInsets(short,%20short,%20short,%20short)), [setParagraphInsets](http://docs.google.com/javax/swing/text/CompositeView.html#setParagraphInsets(javax.swing.text.AttributeSet)) |

| **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)), [breakView](http://docs.google.com/javax/swing/text/View.html#breakView(int,%20int,%20float,%20float)), [createFragment](http://docs.google.com/javax/swing/text/View.html#createFragment(int,%20int)), [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()), [getBreakWeight](http://docs.google.com/javax/swing/text/View.html#getBreakWeight(int,%20float,%20float)), [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()), [getEndOffset](http://docs.google.com/javax/swing/text/View.html#getEndOffset()), [getGraphics](http://docs.google.com/javax/swing/text/View.html#getGraphics()), [getParent](http://docs.google.com/javax/swing/text/View.html#getParent()), [getStartOffset](http://docs.google.com/javax/swing/text/View.html#getStartOffset()), [getToolTipText](http://docs.google.com/javax/swing/text/View.html#getToolTipText(float,%20float,%20java.awt.Shape)), [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)), [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,%20java.awt.Shape)), [remove](http://docs.google.com/javax/swing/text/View.html#remove(int)), [removeAll](http://docs.google.com/javax/swing/text/View.html#removeAll()), [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) |
| --- |
| [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()), [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)) |

| **Field Detail** |
| --- |

### firstLineIndent

protected int **firstLineIndent**

Indentation for the first line, from the left inset.

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

### ParagraphView

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

Constructs a ParagraphView for the given element.

**Parameters:**elem - the element that this view is responsible for

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

### setJustification

protected void **setJustification**(int j)

Sets the type of justification.

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

* StyleConstants.ALIGN\_LEFT
* StyleConstants.ALIGN\_CENTER
* StyleConstants.ALIGN\_RIGHT

### setLineSpacing

protected void **setLineSpacing**(float ls)

Sets the line spacing.

**Parameters:**ls - the value is a factor of the line hight

### setFirstLineIndent

protected void **setFirstLineIndent**(float fi)

Sets the indent on the first line.

**Parameters:**fi - the value in points

### setPropertiesFromAttributes

protected void **setPropertiesFromAttributes**()

Set the cached properties from the attributes.

### getLayoutViewCount

protected int **getLayoutViewCount**()

Returns the number of views that this view is responsible for. The child views of the paragraph are rows which have been used to arrange pieces of the Views that represent the child elements. This is the number of views that have been tiled in two dimensions, and should be equivalent to the number of child elements to the element this view is responsible for.

**Returns:**the number of views that this ParagraphView is responsible for

### getLayoutView

protected [View](http://docs.google.com/javax/swing/text/View.html) **getLayoutView**(int index)

Returns the view at a given index. The child views of the paragraph are rows which have been used to arrange pieces of the Views that represent the child elements. This methods returns the view responsible for the child element index (prior to breaking). These are the Views that were produced from a factory (to represent the child elements) and used for layout.

**Parameters:**index - the index of the desired view **Returns:**the view at index

### adjustRow

protected void **adjustRow**(javax.swing.text.ParagraphView.Row r,  
 int desiredSpan,  
 int x)

Adjusts the given row if possible to fit within the layout span. By default this will try to find the highest break weight possible nearest the end of the row. If a forced break is encountered, the break will be positioned there.

This is meant for internal usage, and should not be used directly.

**Parameters:**r - the row to adjust to the current layout spandesiredSpan - the current layout span >= 0x - the location r starts at

### getNextNorthSouthVisualPositionFrom

protected int **getNextNorthSouthVisualPositionFrom**(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)

Returns the next visual position for the cursor, in either the east or west direction. Overridden from CompositeView.

**Overrides:**[getNextNorthSouthVisualPositionFrom](http://docs.google.com/javax/swing/text/CompositeView.html#getNextNorthSouthVisualPositionFrom(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D)) in class [CompositeView](http://docs.google.com/javax/swing/text/CompositeView.html) **Parameters:**pos - position into the modelb - either Position.Bias.Forward or Position.Bias.Backwarda - the allocated region to render intodirection - either SwingConstants.NORTH or SwingConstants.SOUTHbiasRet - an array containing the bias that were checked in this method **Returns:**the location in the model that represents the next location visual position **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)**See Also:**[CompositeView.getNextVisualPositionFrom(int, javax.swing.text.Position.Bias, java.awt.Shape, int, javax.swing.text.Position.Bias[])](http://docs.google.com/javax/swing/text/CompositeView.html#getNextVisualPositionFrom(int,%20javax.swing.text.Position.Bias,%20java.awt.Shape,%20int,%20javax.swing.text.Position.Bias%5B%5D))

### getClosestPositionTo

protected int **getClosestPositionTo**(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,  
 int rowIndex,  
 int x)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Returns the closest model position to x. rowIndex gives the index of the view that corresponds that should be looked in.

**Parameters:**pos - position into the modela - the allocated region to render intodirection - one of the following values:

* SwingConstants.NORTH
* SwingConstants.SOUTH

biasRet - an array containing the bias that were checked in this methodrowIndex - the index of the viewx - the x coordinate of interest **Returns:**the closest model position to x **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

### flipEastAndWestAtEnds

protected boolean **flipEastAndWestAtEnds**(int position,  
 [Position.Bias](http://docs.google.com/javax/swing/text/Position.Bias.html) bias)

Determines in which direction the next view lays. Consider the View at index n. Typically the Views are layed out from left to right, so that the View to the EAST will be at index n + 1, and the View to the WEST will be at index n - 1. In certain situations, such as with bidirectional text, it is possible that the View to EAST is not at index n + 1, but rather at index n - 1, or that the View to the WEST is not at index n - 1, but index n + 1. In this case this method would return true, indicating the Views are layed out in descending order.

This will return true if the text is layed out right to left at position, otherwise false.

**Overrides:**[flipEastAndWestAtEnds](http://docs.google.com/javax/swing/text/BoxView.html#flipEastAndWestAtEnds(int,%20javax.swing.text.Position.Bias)) in class [BoxView](http://docs.google.com/javax/swing/text/BoxView.html) **Parameters:**position - position into the modelbias - either Position.Bias.Forward or Position.Bias.Backward **Returns:**true if the text is layed out right to left at position, otherwise false.

### getFlowSpan

public int **getFlowSpan**(int index)

Fetches the constraining span to flow against for the given child index.

**Overrides:**[getFlowSpan](http://docs.google.com/javax/swing/text/FlowView.html#getFlowSpan(int)) in class [FlowView](http://docs.google.com/javax/swing/text/FlowView.html) **Parameters:**index - the index of the view being queried **Returns:**the constraining span for the given view at index**Since:** 1.3 **See Also:**[FlowView.getFlowStart(int)](http://docs.google.com/javax/swing/text/FlowView.html#getFlowStart(int))

### getFlowStart

public int **getFlowStart**(int index)

Fetches the location along the flow axis that the flow span will start at.

**Overrides:**[getFlowStart](http://docs.google.com/javax/swing/text/FlowView.html#getFlowStart(int)) in class [FlowView](http://docs.google.com/javax/swing/text/FlowView.html) **Parameters:**index - the index of the view being queried **Returns:**the location for the given view at index**Since:** 1.3 **See Also:**[FlowView.getFlowSpan(int)](http://docs.google.com/javax/swing/text/FlowView.html#getFlowSpan(int))

### createRow

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

Create a View that should be used to hold a a row's worth of children in a flow.

**Specified by:**[createRow](http://docs.google.com/javax/swing/text/FlowView.html#createRow()) in class [FlowView](http://docs.google.com/javax/swing/text/FlowView.html) **Returns:**the new View**Since:** 1.3

### nextTabStop

public float **nextTabStop**(float x,  
 int tabOffset)

Returns the next tab stop position given a reference position. This view implements the tab coordinate system, and calls getTabbedSpan on the logical children in the process of layout to determine the desired span of the children. The logical children can delegate their tab expansion upward to the paragraph which knows how to expand tabs. LabelView is an example of a view that delegates its tab expansion needs upward to the paragraph.

This is implemented to try and locate a TabSet in the paragraph element's attribute set. If one can be found, its settings will be used, otherwise a default expansion will be provided. The base location for for tab expansion is the left inset from the paragraphs most recent allocation (which is what the layout of the children is based upon).

**Specified by:**[nextTabStop](http://docs.google.com/javax/swing/text/TabExpander.html#nextTabStop(float,%20int)) in interface [TabExpander](http://docs.google.com/javax/swing/text/TabExpander.html) **Parameters:**x - the X reference positiontabOffset - the position within the text stream that the tab occurred at >= 0 **Returns:**the trailing end of the tab expansion >= 0**See Also:**[TabSet](http://docs.google.com/javax/swing/text/TabSet.html), [TabStop](http://docs.google.com/javax/swing/text/TabStop.html), [LabelView](http://docs.google.com/javax/swing/text/LabelView.html)

### getTabSet

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

Gets the Tabset to be used in calculating tabs.

**Returns:**the TabSet

### getPartialSize

protected float **getPartialSize**(int startOffset,  
 int endOffset)

Returns the size used by the views between startOffset and endOffset. This uses getPartialView to calculate the size if the child view implements the TabableView interface. If a size is needed and a View does not implement the TabableView interface, the preferredSpan will be used.

**Parameters:**startOffset - the starting document offset >= 0endOffset - the ending document offset >= startOffset **Returns:**the size >= 0

### findOffsetToCharactersInString

protected int **findOffsetToCharactersInString**(char[] string,  
 int start)

Finds the next character in the document with a character in string, starting at offset start. If there are no characters found, -1 will be returned.

**Parameters:**string - the string of charactersstart - where to start in the model >= 0 **Returns:**the document offset, or -1 if no characters found

### getTabBase

protected float **getTabBase**()

Returns where the tabs are calculated from.

**Returns:**where tabs are calculated from

### 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 using the given rendering surface and area on that surface. This is implemented to delgate to the superclass after stashing the base coordinate for tab calculations.

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

### getAlignment

public float **getAlignment**(int axis)

Determines the desired alignment for this view along an axis. This is implemented to give the alignment to the center of the first row along the y axis, and the default along the x axis.

**Overrides:**[getAlignment](http://docs.google.com/javax/swing/text/BoxView.html#getAlignment(int)) in class [BoxView](http://docs.google.com/javax/swing/text/BoxView.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.

### breakView

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

Breaks this view on the given axis at the given length.

ParagraphView instances are breakable along the Y\_AXIS only, and only if len is after the first line.

**Parameters:**axis - may be either View.X\_AXIS or View.Y\_AXISlen - specifies where a potential break is desired along the given axis >= 0a - the current allocation of the view **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))

### getBreakWeight

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

Gets the break weight for a given location.

ParagraphView instances are breakable along the Y\_AXIS only, and only if len is after the first row. If the length is less than one row, a value of BadBreakWeight is returned.

**Parameters:**axis - may be either View.X\_AXIS or View.Y\_AXISlen - specifies where a potential break is desired >= 0 **Returns:**a value indicating the attractiveness of breaking here; either GoodBreakWeight or BadBreakWeight**See Also:**[View.getBreakWeight(int, float, float)](http://docs.google.com/javax/swing/text/View.html#getBreakWeight(int,%20float,%20float))

### changedUpdate

public void **changedUpdate**([DocumentEvent](http://docs.google.com/javax/swing/event/DocumentEvent.html) changes,  
 [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.

**Overrides:**[changedUpdate](http://docs.google.com/javax/swing/text/FlowView.html#changedUpdate(javax.swing.event.DocumentEvent,%20java.awt.Shape,%20javax.swing.text.ViewFactory)) in class [FlowView](http://docs.google.com/javax/swing/text/FlowView.html) **Parameters:**changes - 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/ParagraphView.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/NumberFormatter.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/PasswordView.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/ParagraphView.html)    [**NO FRAMES**](http://docs.google.com/ParagraphView.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | [FIELD](#tyjcwt) | [CONSTR](#2s8eyo1) | [METHOD](#17dp8vu) | DETAIL: [FIELD](#44sinio) | [CONSTR](#z337ya) | [METHOD](#1y810tw) |

[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).