| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/FlowView.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/FieldView.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/FlowView.FlowStrategy.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/FlowView.html)    [**NO FRAMES**](http://docs.google.com/FlowView.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#2et92p0) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: [FIELD](#lnxbz9) | [CONSTR](#2jxsxqh) | [METHOD](#3j2qqm3) |

## **javax.swing.text**

Class FlowView

[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**

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

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

A View that tries to flow it's children into some partially constrained space. This can be used to build things like paragraphs, pages, etc. The flow is made up of the following pieces of functionality.

* A logical set of child views, which as used as a layout pool from which a physical view is formed.
* A strategy for translating the logical view to a physical (flowed) view.
* Constraints for the strategy to work against.
* A physical structure, that represents the flow. The children of this view are where the pieces of of the logical views are placed to create the flow.

**Since:** 1.3 **See Also:**[View](http://docs.google.com/javax/swing/text/View.html)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**FlowView.FlowStrategy**](http://docs.google.com/javax/swing/text/FlowView.FlowStrategy.html)            Strategy for maintaining the physical form of the flow. |

| **Field Summary** | |
| --- | --- |
| protected  [View](http://docs.google.com/javax/swing/text/View.html) | [**layoutPool**](http://docs.google.com/javax/swing/text/FlowView.html#layoutPool)            These are the views that represent the child elements of the element this view represents (The logical view to translate to a physical view). |
| protected  int | [**layoutSpan**](http://docs.google.com/javax/swing/text/FlowView.html#layoutSpan)            Default constraint against which the flow is created against. |
| protected  [FlowView.FlowStrategy](http://docs.google.com/javax/swing/text/FlowView.FlowStrategy.html) | [**strategy**](http://docs.google.com/javax/swing/text/FlowView.html#strategy)            The behavior for keeping the flow updated. |

| **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** | |
| --- | --- |
| [**FlowView**](http://docs.google.com/javax/swing/text/FlowView.html#FlowView(javax.swing.text.Element,%20int))([Element](http://docs.google.com/javax/swing/text/Element.html) elem, int axis)            Constructs a FlowView for the given element. |

| **Method Summary** | |
| --- | --- |
| protected  [SizeRequirements](http://docs.google.com/javax/swing/SizeRequirements.html) | [**calculateMinorAxisRequirements**](http://docs.google.com/javax/swing/text/FlowView.html#calculateMinorAxisRequirements(int,%20javax.swing.SizeRequirements))(int axis, [SizeRequirements](http://docs.google.com/javax/swing/SizeRequirements.html) r)            Calculate equirements along the minor axis. |
| void | [**changedUpdate**](http://docs.google.com/javax/swing/text/FlowView.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 abstract  [View](http://docs.google.com/javax/swing/text/View.html) | [**createRow**](http://docs.google.com/javax/swing/text/FlowView.html#createRow())()            Create a View that should be used to hold a a rows worth of children in a flow. |
| int | [**getFlowAxis**](http://docs.google.com/javax/swing/text/FlowView.html#getFlowAxis())()            Fetches the axis along which views should be flowed. |
| int | [**getFlowSpan**](http://docs.google.com/javax/swing/text/FlowView.html#getFlowSpan(int))(int index)            Fetch the constraining span to flow against for the given child index. |
| int | [**getFlowStart**](http://docs.google.com/javax/swing/text/FlowView.html#getFlowStart(int))(int index)            Fetch the location along the flow axis that the flow span will start at. |
| protected  int | [**getViewIndexAtPosition**](http://docs.google.com/javax/swing/text/FlowView.html#getViewIndexAtPosition(int))(int pos)            Fetches the child view index representing the given position in the model. |
| void | [**insertUpdate**](http://docs.google.com/javax/swing/text/FlowView.html#insertUpdate(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 that something was inserted into the document in a location that this view is responsible for. |
| protected  void | [**layout**](http://docs.google.com/javax/swing/text/FlowView.html#layout(int,%20int))(int width, int height)            Lays out the children. |
| protected  void | [**loadChildren**](http://docs.google.com/javax/swing/text/FlowView.html#loadChildren(javax.swing.text.ViewFactory))([ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)            Loads all of the children to initialize the view. |
| void | [**removeUpdate**](http://docs.google.com/javax/swing/text/FlowView.html#removeUpdate(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 that something was removed from the document in a location that this view is responsible for. |
| void | [**setParent**](http://docs.google.com/javax/swing/text/FlowView.html#setParent(javax.swing.text.View))([View](http://docs.google.com/javax/swing/text/View.html) parent)            Sets the parent of the 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)), [flipEastAndWestAtEnds](http://docs.google.com/javax/swing/text/BoxView.html#flipEastAndWestAtEnds(int,%20javax.swing.text.Position.Bias)), [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)), [getAlignment](http://docs.google.com/javax/swing/text/BoxView.html#getAlignment(int)), [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)), [paint](http://docs.google.com/javax/swing/text/BoxView.html#paint(java.awt.Graphics,%20java.awt.Shape)), [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)), [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)), [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** |
| --- |

### layoutSpan

protected int **layoutSpan**

Default constraint against which the flow is created against.

### layoutPool

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

These are the views that represent the child elements of the element this view represents (The logical view to translate to a physical view). These are not directly children of this view. These are either placed into the rows directly or used for the purpose of breaking into smaller chunks, to form the physical view.

### strategy

protected [FlowView.FlowStrategy](http://docs.google.com/javax/swing/text/FlowView.FlowStrategy.html) **strategy**

The behavior for keeping the flow updated. By default this is a singleton shared by all instances of FlowView (FlowStrategy is stateless). Subclasses can create an alternative strategy, which might keep state.

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

### FlowView

public **FlowView**([Element](http://docs.google.com/javax/swing/text/Element.html) elem,  
 int axis)

Constructs a FlowView for the given element.

**Parameters:**elem - the element that this view is responsible foraxis - may be either View.X\_AXIS or View.Y\_AXIS

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

### getFlowAxis

public int **getFlowAxis**()

Fetches the axis along which views should be flowed. By default, this will be the axis orthogonal to the axis along which the flow rows are tiled (the axis of the default flow rows themselves). This is typically used by the FlowStrategy.

### getFlowSpan

public int **getFlowSpan**(int index)

Fetch the constraining span to flow against for the given child index. This is called by the FlowStrategy while it is updating the flow. A flow can be shaped by providing different values for the row constraints. By default, the entire span inside of the insets along the flow axis is returned.

**Parameters:**index - the index of the row being updated. This should be a value >= 0 and < getViewCount().**See Also:**[getFlowStart(int)](http://docs.google.com/javax/swing/text/FlowView.html#getFlowStart(int))

### getFlowStart

public int **getFlowStart**(int index)

Fetch the location along the flow axis that the flow span will start at. This is called by the FlowStrategy while it is updating the flow. A flow can be shaped by providing different values for the row constraints.

**Parameters:**index - the index of the row being updated. This should be a value >= 0 and < getViewCount().**See Also:**[getFlowSpan(int)](http://docs.google.com/javax/swing/text/FlowView.html#getFlowSpan(int))

### createRow

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

Create a View that should be used to hold a a rows worth of children in a flow. This is called by the FlowStrategy when new children are added or removed (i.e. rows are added or removed) in the process of updating the flow.

### loadChildren

protected void **loadChildren**([ViewFactory](http://docs.google.com/javax/swing/text/ViewFactory.html) f)

Loads all of the children to initialize the view. This is called by the setParent method. This is reimplemented to not load any children directly (as they are created in the process of formatting). If the layoutPool variable is null, an instance of LogicalView is created to represent the logical view that is used in the process of formatting.

**Overrides:**[loadChildren](http://docs.google.com/javax/swing/text/CompositeView.html#loadChildren(javax.swing.text.ViewFactory)) in class [CompositeView](http://docs.google.com/javax/swing/text/CompositeView.html) **Parameters:**f - the view factory**See Also:**[CompositeView.setParent(javax.swing.text.View)](http://docs.google.com/javax/swing/text/CompositeView.html#setParent(javax.swing.text.View))

### getViewIndexAtPosition

protected int **getViewIndexAtPosition**(int pos)

Fetches the child view index representing the given position in the model.

**Overrides:**[getViewIndexAtPosition](http://docs.google.com/javax/swing/text/CompositeView.html#getViewIndexAtPosition(int)) in class [CompositeView](http://docs.google.com/javax/swing/text/CompositeView.html) **Parameters:**pos - the position >= 0 **Returns:**index of the view representing the given position, or -1 if no view represents that position

### layout

protected void **layout**(int width,  
 int height)

Lays out the children. If the span along the flow axis has changed, layout is marked as invalid which which will cause the superclass behavior to recalculate the layout along the box axis. The FlowStrategy.layout method will be called to rebuild the flow rows as appropriate. If the height of this view changes (determined by the perferred size along the box axis), a preferenceChanged is called. Following all of that, the normal box layout of the superclass is performed.

**Overrides:**[layout](http://docs.google.com/javax/swing/text/BoxView.html#layout(int,%20int)) in class [BoxView](http://docs.google.com/javax/swing/text/BoxView.html) **Parameters:**width - the width to lay out against >= 0. This is the width inside of the inset area.height - the height to lay out against >= 0 This is the height inside of the inset area.

### calculateMinorAxisRequirements

protected [SizeRequirements](http://docs.google.com/javax/swing/SizeRequirements.html) **calculateMinorAxisRequirements**(int axis,  
 [SizeRequirements](http://docs.google.com/javax/swing/SizeRequirements.html) r)

Calculate equirements along the minor axis. This is implemented to forward the request to the logical view by calling getMinimumSpan, getPreferredSpan, and getMaximumSpan on it.

**Overrides:**[calculateMinorAxisRequirements](http://docs.google.com/javax/swing/text/BoxView.html#calculateMinorAxisRequirements(int,%20javax.swing.SizeRequirements)) in class [BoxView](http://docs.google.com/javax/swing/text/BoxView.html) **Parameters:**axis - the axis being studiedr - the SizeRequirements object; if null one will be created **Returns:**the newly initialized SizeRequirements object**See Also:**[SizeRequirements](http://docs.google.com/javax/swing/SizeRequirements.html)

### insertUpdate

public void **insertUpdate**([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 that something was inserted into the document in a location that this view is responsible for.

**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:**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.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) changes,  
 [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.

**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:**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.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) 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/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:**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))

### setParent

public void **setParent**([View](http://docs.google.com/javax/swing/text/View.html) parent)

Sets the parent of the view. This is reimplemented to provide the superclass behavior as well as calling the loadChildren method if this view does not already have children. The children should not be loaded in the constructor because the act of setting the parent may cause them to try to search up the hierarchy (to get the hosting Container for example). If this view has children (the view is being moved from one place in the view hierarchy to another), the loadChildren method will not be called.

**Overrides:**[setParent](http://docs.google.com/javax/swing/text/CompositeView.html#setParent(javax.swing.text.View)) in class [CompositeView](http://docs.google.com/javax/swing/text/CompositeView.html) **Parameters:**parent - the parent of the view, null if none

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/FlowView.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/FieldView.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/FlowView.FlowStrategy.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/FlowView.html)    [**NO FRAMES**](http://docs.google.com/FlowView.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#2et92p0) | [CONSTR](#1t3h5sf) | [METHOD](#4d34og8) | DETAIL: [FIELD](#lnxbz9) | [CONSTR](#2jxsxqh) | [METHOD](#3j2qqm3) |

[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).