| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/BoxLayout.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/Box.Filler.AccessibleBoxFiller.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/ButtonGroup.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/BoxLayout.html)    [**NO FRAMES**](http://docs.google.com/BoxLayout.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#35nkun2) |

## **javax.swing**

Class BoxLayout

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

**All Implemented Interfaces:** [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html), [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html), [Serializable](http://docs.google.com/java/io/Serializable.html) **Direct Known Subclasses:** [DefaultMenuLayout](http://docs.google.com/javax/swing/plaf/basic/DefaultMenuLayout.html)

public class **BoxLayout**extends [Object](http://docs.google.com/java/lang/Object.html)implements [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html), [Serializable](http://docs.google.com/java/io/Serializable.html)

A layout manager that allows multiple components to be laid out either vertically or horizontally. The components will not wrap so, for example, a vertical arrangement of components will stay vertically arranged when the frame is resized.

|  |
| --- |

Nesting multiple panels with different combinations of horizontal and vertical gives an effect similar to GridBagLayout, without the complexity. The diagram shows two panels arranged horizontally, each of which contains 3 components arranged vertically.

The BoxLayout manager is constructed with an axis parameter that specifies the type of layout that will be done. There are four choices:

**X\_AXIS** - Components are laid out horizontally from left to right.**Y\_AXIS** - Components are laid out vertically from top to bottom.**LINE\_AXIS** - Components are laid out the way words are laid out in a line, based on the container's ComponentOrientation property. If the container's ComponentOrientation is horizontal then components are laid out horizontally, otherwise they are laid out vertically. For horizontal orientations, if the container's ComponentOrientation is left to right then components are laid out left to right, otherwise they are laid out right to left. For vertical orientations components are always laid out from top to bottom.**PAGE\_AXIS** - Components are laid out the way text lines are laid out on a page, based on the container's ComponentOrientation property. If the container's ComponentOrientation is horizontal then components are laid out vertically, otherwise they are laid out horizontally. For horizontal orientations, if the container's ComponentOrientation is left to right then components are laid out left to right, otherwise they are laid out right to left.  For vertical orientations components are always laid out from top to bottom.

For all directions, components are arranged in the same order as they were added to the container.

BoxLayout attempts to arrange components at their preferred widths (for horizontal layout) or heights (for vertical layout). For a horizontal layout, if not all the components are the same height, BoxLayout attempts to make all the components as high as the highest component. If that's not possible for a particular component, then BoxLayout aligns that component vertically, according to the component's Y alignment. By default, a component has a Y alignment of 0.5, which means that the vertical center of the component should have the same Y coordinate as the vertical centers of other components with 0.5 Y alignment.

Similarly, for a vertical layout, BoxLayout attempts to make all components in the column as wide as the widest component. If that fails, it aligns them horizontally according to their X alignments. For PAGE\_AXIS layout, horizontal alignment is done based on the leading edge of the component. In other words, an X alignment value of 0.0 means the left edge of a component if the container's ComponentOrientation is left to right and it means the right edge of the component otherwise.

Instead of using BoxLayout directly, many programs use the Box class. The Box class is a lightweight container that uses a BoxLayout. It also provides handy methods to help you use BoxLayout well. Adding components to multiple nested boxes is a powerful way to get the arrangement you want.

For further information and examples see [How to Use BoxLayout](http://java.sun.com/docs/books/tutorial/uiswing/layout/box.html), a section in *The Java Tutorial.*

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

**See Also:**[Box](http://docs.google.com/javax/swing/Box.html), [ComponentOrientation](http://docs.google.com/java/awt/ComponentOrientation.html), [JComponent.getAlignmentX()](http://docs.google.com/javax/swing/JComponent.html#getAlignmentX()), [JComponent.getAlignmentY()](http://docs.google.com/javax/swing/JComponent.html#getAlignmentY())

| **Field Summary** | |
| --- | --- |
| static int | [**LINE\_AXIS**](http://docs.google.com/javax/swing/BoxLayout.html#LINE_AXIS)            Specifies that components should be laid out in the direction of a line of text as determined by the target container's ComponentOrientation property. |
| static int | [**PAGE\_AXIS**](http://docs.google.com/javax/swing/BoxLayout.html#PAGE_AXIS)            Specifies that components should be laid out in the direction that lines flow across a page as determined by the target container's ComponentOrientation property. |
| static int | [**X\_AXIS**](http://docs.google.com/javax/swing/BoxLayout.html#X_AXIS)            Specifies that components should be laid out left to right. |
| static int | [**Y\_AXIS**](http://docs.google.com/javax/swing/BoxLayout.html#Y_AXIS)            Specifies that components should be laid out top to bottom. |

| **Constructor Summary** | |
| --- | --- |
| [**BoxLayout**](http://docs.google.com/javax/swing/BoxLayout.html#BoxLayout(java.awt.Container,%20int))([Container](http://docs.google.com/java/awt/Container.html) target, int axis)            Creates a layout manager that will lay out components along the given axis. |

| **Method Summary** | |
| --- | --- |
| void | [**addLayoutComponent**](http://docs.google.com/javax/swing/BoxLayout.html#addLayoutComponent(java.awt.Component,%20java.lang.Object))([Component](http://docs.google.com/java/awt/Component.html) comp, [Object](http://docs.google.com/java/lang/Object.html) constraints)            Not used by this class. |
| void | [**addLayoutComponent**](http://docs.google.com/javax/swing/BoxLayout.html#addLayoutComponent(java.lang.String,%20java.awt.Component))([String](http://docs.google.com/java/lang/String.html) name, [Component](http://docs.google.com/java/awt/Component.html) comp)            Not used by this class. |
| int | [**getAxis**](http://docs.google.com/javax/swing/BoxLayout.html#getAxis())()            Returns the axis that was used to lay out components. |
| float | [**getLayoutAlignmentX**](http://docs.google.com/javax/swing/BoxLayout.html#getLayoutAlignmentX(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Returns the alignment along the X axis for the container. |
| float | [**getLayoutAlignmentY**](http://docs.google.com/javax/swing/BoxLayout.html#getLayoutAlignmentY(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Returns the alignment along the Y axis for the container. |
| [Container](http://docs.google.com/java/awt/Container.html) | [**getTarget**](http://docs.google.com/javax/swing/BoxLayout.html#getTarget())()            Returns the container that uses this layout manager. |
| void | [**invalidateLayout**](http://docs.google.com/javax/swing/BoxLayout.html#invalidateLayout(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Indicates that a child has changed its layout related information, and thus any cached calculations should be flushed. |
| void | [**layoutContainer**](http://docs.google.com/javax/swing/BoxLayout.html#layoutContainer(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Called by the AWT when the specified container needs to be laid out. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**maximumLayoutSize**](http://docs.google.com/javax/swing/BoxLayout.html#maximumLayoutSize(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Returns the maximum dimensions the target container can use to lay out the components it contains. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**minimumLayoutSize**](http://docs.google.com/javax/swing/BoxLayout.html#minimumLayoutSize(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Returns the minimum dimensions needed to lay out the components contained in the specified target container. |
| [Dimension](http://docs.google.com/java/awt/Dimension.html) | [**preferredLayoutSize**](http://docs.google.com/javax/swing/BoxLayout.html#preferredLayoutSize(java.awt.Container))([Container](http://docs.google.com/java/awt/Container.html) target)            Returns the preferred dimensions for this layout, given the components in the specified target container. |
| void | [**removeLayoutComponent**](http://docs.google.com/javax/swing/BoxLayout.html#removeLayoutComponent(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) comp)            Not used by this class. |

| **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** |
| --- |

### X\_AXIS

public static final int **X\_AXIS**

Specifies that components should be laid out left to right.

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

### Y\_AXIS

public static final int **Y\_AXIS**

Specifies that components should be laid out top to bottom.

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

### LINE\_AXIS

public static final int **LINE\_AXIS**

Specifies that components should be laid out in the direction of a line of text as determined by the target container's ComponentOrientation property.

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

### PAGE\_AXIS

public static final int **PAGE\_AXIS**

Specifies that components should be laid out in the direction that lines flow across a page as determined by the target container's ComponentOrientation property.

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

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

### BoxLayout

public **BoxLayout**([Container](http://docs.google.com/java/awt/Container.html) target,  
 int axis)

Creates a layout manager that will lay out components along the given axis.

**Parameters:**target - the container that needs to be laid outaxis - the axis to lay out components along. Can be one of: BoxLayout.X\_AXIS, BoxLayout.Y\_AXIS, BoxLayout.LINE\_AXIS or BoxLayout.PAGE\_AXIS **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the value of axis is invalid

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

### getTarget

public final [Container](http://docs.google.com/java/awt/Container.html) **getTarget**()

Returns the container that uses this layout manager.

**Returns:**the container that uses this layout manager**Since:** 1.6

### getAxis

public final int **getAxis**()

Returns the axis that was used to lay out components. Returns one of: BoxLayout.X\_AXIS, BoxLayout.Y\_AXIS, BoxLayout.LINE\_AXIS or BoxLayout.PAGE\_AXIS

**Returns:**the axis that was used to lay out components**Since:** 1.6

### invalidateLayout

public void **invalidateLayout**([Container](http://docs.google.com/java/awt/Container.html) target)

Indicates that a child has changed its layout related information, and thus any cached calculations should be flushed.

This method is called by AWT when the invalidate method is called on the Container. Since the invalidate method may be called asynchronously to the event thread, this method may be called asynchronously.

**Specified by:**[invalidateLayout](http://docs.google.com/java/awt/LayoutManager2.html#invalidateLayout(java.awt.Container)) in interface [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html) **Parameters:**target - the affected container **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor

### addLayoutComponent

public void **addLayoutComponent**([String](http://docs.google.com/java/lang/String.html) name,  
 [Component](http://docs.google.com/java/awt/Component.html) comp)

Not used by this class.

**Specified by:**[addLayoutComponent](http://docs.google.com/java/awt/LayoutManager.html#addLayoutComponent(java.lang.String,%20java.awt.Component)) in interface [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) **Parameters:**name - the name of the componentcomp - the component

### removeLayoutComponent

public void **removeLayoutComponent**([Component](http://docs.google.com/java/awt/Component.html) comp)

Not used by this class.

**Specified by:**[removeLayoutComponent](http://docs.google.com/java/awt/LayoutManager.html#removeLayoutComponent(java.awt.Component)) in interface [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) **Parameters:**comp - the component

### addLayoutComponent

public void **addLayoutComponent**([Component](http://docs.google.com/java/awt/Component.html) comp,  
 [Object](http://docs.google.com/java/lang/Object.html) constraints)

Not used by this class.

**Specified by:**[addLayoutComponent](http://docs.google.com/java/awt/LayoutManager2.html#addLayoutComponent(java.awt.Component,%20java.lang.Object)) in interface [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html) **Parameters:**comp - the componentconstraints - constraints

### preferredLayoutSize

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

Returns the preferred dimensions for this layout, given the components in the specified target container.

**Specified by:**[preferredLayoutSize](http://docs.google.com/java/awt/LayoutManager.html#preferredLayoutSize(java.awt.Container)) in interface [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) **Parameters:**target - the container that needs to be laid out **Returns:**the dimensions >= 0 && <= Integer.MAX\_VALUE **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor**See Also:**[Container](http://docs.google.com/java/awt/Container.html), [minimumLayoutSize(java.awt.Container)](http://docs.google.com/javax/swing/BoxLayout.html#minimumLayoutSize(java.awt.Container)), [maximumLayoutSize(java.awt.Container)](http://docs.google.com/javax/swing/BoxLayout.html#maximumLayoutSize(java.awt.Container))

### minimumLayoutSize

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

Returns the minimum dimensions needed to lay out the components contained in the specified target container.

**Specified by:**[minimumLayoutSize](http://docs.google.com/java/awt/LayoutManager.html#minimumLayoutSize(java.awt.Container)) in interface [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) **Parameters:**target - the container that needs to be laid out **Returns:**the dimensions >= 0 && <= Integer.MAX\_VALUE **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor**See Also:**[preferredLayoutSize(java.awt.Container)](http://docs.google.com/javax/swing/BoxLayout.html#preferredLayoutSize(java.awt.Container)), [maximumLayoutSize(java.awt.Container)](http://docs.google.com/javax/swing/BoxLayout.html#maximumLayoutSize(java.awt.Container))

### maximumLayoutSize

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

Returns the maximum dimensions the target container can use to lay out the components it contains.

**Specified by:**[maximumLayoutSize](http://docs.google.com/java/awt/LayoutManager2.html#maximumLayoutSize(java.awt.Container)) in interface [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html) **Parameters:**target - the container that needs to be laid out **Returns:**the dimenions >= 0 && <= Integer.MAX\_VALUE **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor**See Also:**[preferredLayoutSize(java.awt.Container)](http://docs.google.com/javax/swing/BoxLayout.html#preferredLayoutSize(java.awt.Container)), [minimumLayoutSize(java.awt.Container)](http://docs.google.com/javax/swing/BoxLayout.html#minimumLayoutSize(java.awt.Container))

### getLayoutAlignmentX

public float **getLayoutAlignmentX**([Container](http://docs.google.com/java/awt/Container.html) target)

Returns the alignment along the X axis for the container. If the box is horizontal, the default alignment will be returned. Otherwise, the alignment needed to place the children along the X axis will be returned.

**Specified by:**[getLayoutAlignmentX](http://docs.google.com/java/awt/LayoutManager2.html#getLayoutAlignmentX(java.awt.Container)) in interface [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html) **Parameters:**target - the container **Returns:**the alignment >= 0.0f && <= 1.0f **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor

### getLayoutAlignmentY

public float **getLayoutAlignmentY**([Container](http://docs.google.com/java/awt/Container.html) target)

Returns the alignment along the Y axis for the container. If the box is vertical, the default alignment will be returned. Otherwise, the alignment needed to place the children along the Y axis will be returned.

**Specified by:**[getLayoutAlignmentY](http://docs.google.com/java/awt/LayoutManager2.html#getLayoutAlignmentY(java.awt.Container)) in interface [LayoutManager2](http://docs.google.com/java/awt/LayoutManager2.html) **Parameters:**target - the container **Returns:**the alignment >= 0.0f && <= 1.0f **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor

### layoutContainer

public void **layoutContainer**([Container](http://docs.google.com/java/awt/Container.html) target)

Called by the AWT when the specified container needs to be laid out.

**Specified by:**[layoutContainer](http://docs.google.com/java/awt/LayoutManager.html#layoutContainer(java.awt.Container)) in interface [LayoutManager](http://docs.google.com/java/awt/LayoutManager.html) **Parameters:**target - the container to lay out **Throws:** [AWTError](http://docs.google.com/java/awt/AWTError.html) - if the target isn't the container specified to the BoxLayout constructor

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/BoxLayout.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/Box.Filler.AccessibleBoxFiller.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/ButtonGroup.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/BoxLayout.html)    [**NO FRAMES**](http://docs.google.com/BoxLayout.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#35nkun2) |

[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).