| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/BoundedRangeModel.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/BorderFactory.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/Box.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/BoundedRangeModel.html)    [**NO FRAMES**](http://docs.google.com/BoundedRangeModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#2et92p0) |

## **javax.swing**

Interface BoundedRangeModel

**All Known Implementing Classes:** [DefaultBoundedRangeModel](http://docs.google.com/javax/swing/DefaultBoundedRangeModel.html)

public interface **BoundedRangeModel**

Defines the data model used by components like Sliders and ProgressBars. Defines four interrelated integer properties: minimum, maximum, extent and value. These four integers define two nested ranges like this:

minimum <= value <= value+extent <= maximum

The outer range is minimum,maximum and the inner range is value,value+extent. The inner range must lie within the outer one, i.e. value must be less than or equal to maximum and value+extent must greater than or equal to minimum, and maximum must be greater than or equal to minimum. There are a few features of this model that one might find a little surprising. These quirks exist for the convenience of the Swing BoundedRangeModel clients, such as Slider and ScrollBar.

* The minimum and maximum set methods "correct" the other three properties to accommodate their new value argument. For example setting the model's minimum may change its maximum, value, and extent properties (in that order), to maintain the constraints specified above.
* The value and extent set methods "correct" their argument to fit within the limits defined by the other three properties. For example if value == maximum, setExtent(10) would change the extent (back) to zero.
* The four BoundedRangeModel values are defined as Java Beans properties however Swing ChangeEvents are used to notify clients of changes rather than PropertyChangeEvents. This was done to keep the overhead of monitoring a BoundedRangeModel low. Changes are often reported at MouseDragged rates.

For an example of specifying custom bounded range models used by sliders, see [The Anatomy of a Swing-Based Program](http://java.sun.com/docs/books/tutorial/uiswing/overview/anatomy.html) in *The Java Tutorial.*

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

| **Method Summary** | |
| --- | --- |
| void | [**addChangeListener**](http://docs.google.com/javax/swing/BoundedRangeModel.html#addChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) x)            Adds a ChangeListener to the model's listener list. |
| int | [**getExtent**](http://docs.google.com/javax/swing/BoundedRangeModel.html#getExtent())()            Returns the model's extent, the length of the inner range that begins at the model's value. |
| int | [**getMaximum**](http://docs.google.com/javax/swing/BoundedRangeModel.html#getMaximum())()            Returns the model's maximum. |
| int | [**getMinimum**](http://docs.google.com/javax/swing/BoundedRangeModel.html#getMinimum())()            Returns the minimum acceptable value. |
| int | [**getValue**](http://docs.google.com/javax/swing/BoundedRangeModel.html#getValue())()            Returns the model's current value. |
| boolean | [**getValueIsAdjusting**](http://docs.google.com/javax/swing/BoundedRangeModel.html#getValueIsAdjusting())()            Returns true if the current changes to the value property are part of a series of changes. |
| void | [**removeChangeListener**](http://docs.google.com/javax/swing/BoundedRangeModel.html#removeChangeListener(javax.swing.event.ChangeListener))([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) x)            Removes a ChangeListener from the model's listener list. |
| void | [**setExtent**](http://docs.google.com/javax/swing/BoundedRangeModel.html#setExtent(int))(int newExtent)            Sets the model's extent. |
| void | [**setMaximum**](http://docs.google.com/javax/swing/BoundedRangeModel.html#setMaximum(int))(int newMaximum)            Sets the model's maximum to *newMaximum*. |
| void | [**setMinimum**](http://docs.google.com/javax/swing/BoundedRangeModel.html#setMinimum(int))(int newMinimum)            Sets the model's minimum to *newMinimum*. |
| void | [**setRangeProperties**](http://docs.google.com/javax/swing/BoundedRangeModel.html#setRangeProperties(int,%20int,%20int,%20int,%20boolean))(int value, int extent, int min, int max, boolean adjusting)            This method sets all of the model's data with a single method call. |
| void | [**setValue**](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValue(int))(int newValue)            Sets the model's current value to newValue if newValue satisfies the model's constraints. |
| void | [**setValueIsAdjusting**](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValueIsAdjusting(boolean))(boolean b)            This attribute indicates that any upcoming changes to the value of the model should be considered a single event. |

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

### getMinimum

int **getMinimum**()

Returns the minimum acceptable value.

**Returns:**the value of the minimum property**See Also:**[setMinimum(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setMinimum(int))

### setMinimum

void **setMinimum**(int newMinimum)

Sets the model's minimum to *newMinimum*. The other three properties may be changed as well, to ensure that:

minimum <= value <= value+extent <= maximum

Notifies any listeners if the model changes.

**Parameters:**newMinimum - the model's new minimum**See Also:**[getMinimum()](http://docs.google.com/javax/swing/BoundedRangeModel.html#getMinimum()), [addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/BoundedRangeModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getMaximum

int **getMaximum**()

Returns the model's maximum. Note that the upper limit on the model's value is (maximum - extent).

**Returns:**the value of the maximum property.**See Also:**[setMaximum(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setMaximum(int)), [setExtent(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setExtent(int))

### setMaximum

void **setMaximum**(int newMaximum)

Sets the model's maximum to *newMaximum*. The other three properties may be changed as well, to ensure that

minimum <= value <= value+extent <= maximum

Notifies any listeners if the model changes.

**Parameters:**newMaximum - the model's new maximum**See Also:**[getMaximum()](http://docs.google.com/javax/swing/BoundedRangeModel.html#getMaximum()), [addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/BoundedRangeModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getValue

int **getValue**()

Returns the model's current value. Note that the upper limit on the model's value is maximum - extent and the lower limit is minimum.

**Returns:**the model's value**See Also:**[setValue(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValue(int))

### setValue

void **setValue**(int newValue)

Sets the model's current value to newValue if newValue satisfies the model's constraints. Those constraints are:

minimum <= value <= value+extent <= maximum

Otherwise, if newValue is less than minimum it's set to minimum, if its greater than maximum then it's set to maximum, and if it's greater than value+extent then it's set to value+extent.

When a BoundedRange model is used with a scrollbar the value specifies the origin of the scrollbar knob (aka the "thumb" or "elevator"). The value usually represents the origin of the visible part of the object being scrolled.

Notifies any listeners if the model changes.

**Parameters:**newValue - the model's new value**See Also:**[getValue()](http://docs.google.com/javax/swing/BoundedRangeModel.html#getValue())

### setValueIsAdjusting

void **setValueIsAdjusting**(boolean b)

This attribute indicates that any upcoming changes to the value of the model should be considered a single event. This attribute will be set to true at the start of a series of changes to the value, and will be set to false when the value has finished changing. Normally this allows a listener to only take action when the final value change in committed, instead of having to do updates for all intermediate values.

Sliders and scrollbars use this property when a drag is underway.

**Parameters:**b - true if the upcoming changes to the value property are part of a series

### getValueIsAdjusting

boolean **getValueIsAdjusting**()

Returns true if the current changes to the value property are part of a series of changes.

**Returns:**the valueIsAdjustingProperty.**See Also:**[setValueIsAdjusting(boolean)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValueIsAdjusting(boolean))

### getExtent

int **getExtent**()

Returns the model's extent, the length of the inner range that begins at the model's value.

**Returns:**the value of the model's extent property**See Also:**[setExtent(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setExtent(int)), [setValue(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValue(int))

### setExtent

void **setExtent**(int newExtent)

Sets the model's extent. The *newExtent* is forced to be greater than or equal to zero and less than or equal to maximum - value.

When a BoundedRange model is used with a scrollbar the extent defines the length of the scrollbar knob (aka the "thumb" or "elevator"). The extent usually represents how much of the object being scrolled is visible. When used with a slider, the extent determines how much the value can "jump", for example when the user presses PgUp or PgDn.

Notifies any listeners if the model changes.

**Parameters:**newExtent - the model's new extent**See Also:**[getExtent()](http://docs.google.com/javax/swing/BoundedRangeModel.html#getExtent()), [setValue(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValue(int))

### setRangeProperties

void **setRangeProperties**(int value,  
 int extent,  
 int min,  
 int max,  
 boolean adjusting)

This method sets all of the model's data with a single method call. The method results in a single change event being generated. This is convenient when you need to adjust all the model data simultaneously and do not want individual change events to occur.

**Parameters:**value - an int giving the current valueextent - an int giving the amount by which the value can "jump"min - an int giving the minimum valuemax - an int giving the maximum valueadjusting - a boolean, true if a series of changes are in progress**See Also:**[setValue(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValue(int)), [setExtent(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setExtent(int)), [setMinimum(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setMinimum(int)), [setMaximum(int)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setMaximum(int)), [setValueIsAdjusting(boolean)](http://docs.google.com/javax/swing/BoundedRangeModel.html#setValueIsAdjusting(boolean))

### addChangeListener

void **addChangeListener**([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) x)

Adds a ChangeListener to the model's listener list.

**Parameters:**x - the ChangeListener to add**See Also:**[removeChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/BoundedRangeModel.html#removeChangeListener(javax.swing.event.ChangeListener))

### removeChangeListener

void **removeChangeListener**([ChangeListener](http://docs.google.com/javax/swing/event/ChangeListener.html) x)

Removes a ChangeListener from the model's listener list.

**Parameters:**x - the ChangeListener to remove**See Also:**[addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/BoundedRangeModel.html#addChangeListener(javax.swing.event.ChangeListener))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/BoundedRangeModel.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/BorderFactory.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/Box.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/BoundedRangeModel.html)    [**NO FRAMES**](http://docs.google.com/BoundedRangeModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#2et92p0) |

[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).