| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SpinnerNumberModel.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/SpinnerModel.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/Spring.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/SpinnerNumberModel.html)    [**NO FRAMES**](http://docs.google.com/SpinnerNumberModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#2s8eyo1) | [METHOD](#35nkun2) |

## **javax.swing**

Class SpinnerNumberModel

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

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html)

public class **SpinnerNumberModel**extends [AbstractSpinnerModel](http://docs.google.com/javax/swing/AbstractSpinnerModel.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html)

A SpinnerModel for sequences of numbers. The upper and lower bounds of the sequence are defined by properties called minimum and maximum. The size of the increase or decrease computed by the nextValue and previousValue methods is defined by a property called stepSize. The minimum and maximum properties can be null to indicate that the sequence has no lower or upper limit. All of the properties in this class are defined in terms of two generic types: Number and Comparable, so that all Java numeric types may be accommodated. Internally, there's only support for values whose type is one of the primitive Number types: Double, Float, Long, Integer, Short, or Byte.

To create a SpinnerNumberModel for the integer range zero to one hundred, with fifty as the initial value, one could write:

Integer value = new Integer(50);   
 Integer min = new Integer(0);  
 Integer max = new Integer(100);   
 Integer step = new Integer(1);   
 SpinnerNumberModel model = new SpinnerNumberModel(value, min, max, step);   
 int fifty = model.getNumber().intValue();

Spinners for integers and doubles are common, so special constructors for these cases are provided. For example to create the model in the previous example, one could also write:

SpinnerNumberModel model = new SpinnerNumberModel(50, 0, 100, 1);

This model inherits a ChangeListener. The ChangeListeners are notified whenever the model's value, stepSize, minimum, or maximum properties changes.

**Since:** 1.4 **See Also:**[JSpinner](http://docs.google.com/javax/swing/JSpinner.html), [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html), [AbstractSpinnerModel](http://docs.google.com/javax/swing/AbstractSpinnerModel.html), [SpinnerListModel](http://docs.google.com/javax/swing/SpinnerListModel.html), [SpinnerDateModel](http://docs.google.com/javax/swing/SpinnerDateModel.html)

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

| **Fields inherited from class javax.swing.**[**AbstractSpinnerModel**](http://docs.google.com/javax/swing/AbstractSpinnerModel.html) |
| --- |
| [listenerList](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#listenerList) |

| **Constructor Summary** | |
| --- | --- |
| [**SpinnerNumberModel**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#SpinnerNumberModel())()            Constructs a SpinnerNumberModel with no minimum or maximum value, stepSize equal to one, and an initial value of zero. |
| [**SpinnerNumberModel**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#SpinnerNumberModel(double,%20double,%20double,%20double))(double value, double minimum, double maximum, double stepSize)            Constructs a SpinnerNumberModel with the specified value, minimum/maximum bounds, and stepSize. |
| [**SpinnerNumberModel**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#SpinnerNumberModel(int,%20int,%20int,%20int))(int value, int minimum, int maximum, int stepSize)            Constructs a SpinnerNumberModel with the specified value, minimum/maximum bounds, and stepSize. |
| [**SpinnerNumberModel**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#SpinnerNumberModel(java.lang.Number,%20java.lang.Comparable,%20java.lang.Comparable,%20java.lang.Number))([Number](http://docs.google.com/java/lang/Number.html) value, [Comparable](http://docs.google.com/java/lang/Comparable.html) minimum, [Comparable](http://docs.google.com/java/lang/Comparable.html) maximum, [Number](http://docs.google.com/java/lang/Number.html) stepSize)            Constructs a SpinnerModel that represents a closed sequence of numbers from minimum to maximum. |

| **Method Summary** | |
| --- | --- |
| [Comparable](http://docs.google.com/java/lang/Comparable.html) | [**getMaximum**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getMaximum())()            Returns the last number in the sequence. |
| [Comparable](http://docs.google.com/java/lang/Comparable.html) | [**getMinimum**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getMinimum())()            Returns the first number in this sequence. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getNextValue**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getNextValue())()            Returns the next number in the sequence. |
| [Number](http://docs.google.com/java/lang/Number.html) | [**getNumber**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getNumber())()            Returns the value of the current element of the sequence. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getPreviousValue**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getPreviousValue())()            Returns the previous number in the sequence. |
| [Number](http://docs.google.com/java/lang/Number.html) | [**getStepSize**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getStepSize())()            Returns the size of the value change computed by the getNextValue and getPreviousValue methods. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getValue**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getValue())()            Returns the value of the current element of the sequence. |
| void | [**setMaximum**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setMaximum(java.lang.Comparable))([Comparable](http://docs.google.com/java/lang/Comparable.html) maximum)            Changes the upper bound for numbers in this sequence. |
| void | [**setMinimum**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setMinimum(java.lang.Comparable))([Comparable](http://docs.google.com/java/lang/Comparable.html) minimum)            Changes the lower bound for numbers in this sequence. |
| void | [**setStepSize**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setStepSize(java.lang.Number))([Number](http://docs.google.com/java/lang/Number.html) stepSize)            Changes the size of the value change computed by the getNextValue and getPreviousValue methods. |
| void | [**setValue**](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setValue(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) value)            Sets the current value for this sequence. |

| **Methods inherited from class javax.swing.**[**AbstractSpinnerModel**](http://docs.google.com/javax/swing/AbstractSpinnerModel.html) |
| --- |
| [addChangeListener](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener)), [fireStateChanged](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#fireStateChanged()), [getChangeListeners](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#getChangeListeners()), [getListeners](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#getListeners(java.lang.Class)), [removeChangeListener](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#removeChangeListener(javax.swing.event.ChangeListener)) |

| **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)) |

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

### SpinnerNumberModel

public **SpinnerNumberModel**([Number](http://docs.google.com/java/lang/Number.html) value,  
 [Comparable](http://docs.google.com/java/lang/Comparable.html) minimum,  
 [Comparable](http://docs.google.com/java/lang/Comparable.html) maximum,  
 [Number](http://docs.google.com/java/lang/Number.html) stepSize)

Constructs a SpinnerModel that represents a closed sequence of numbers from minimum to maximum. The nextValue and previousValue methods compute elements of the sequence by adding or subtracting stepSize respectively. All of the parameters must be mutually Comparable, value and stepSize must be instances of Integer Long, Float, or Double.

The minimum and maximum parameters can be null to indicate that the range doesn't have an upper or lower bound. If value or stepSize is null, or if both minimum and maximum are specified and mininum > maximum then an IllegalArgumentException is thrown. Similarly if (minimum <= value <= maximum) is false, an IllegalArgumentException is thrown.

**Parameters:**value - the current (non null) value of the modelminimum - the first number in the sequence or nullmaximum - the last number in the sequence or nullstepSize - the difference between elements of the sequence **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if stepSize or value is null or if the following expression is false: minimum <= value <= maximum

### SpinnerNumberModel

public **SpinnerNumberModel**(int value,  
 int minimum,  
 int maximum,  
 int stepSize)

Constructs a SpinnerNumberModel with the specified value, minimum/maximum bounds, and stepSize.

**Parameters:**value - the current value of the modelminimum - the first number in the sequencemaximum - the last number in the sequencestepSize - the difference between elements of the sequence **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the following expression is false: minimum <= value <= maximum

### SpinnerNumberModel

public **SpinnerNumberModel**(double value,  
 double minimum,  
 double maximum,  
 double stepSize)

Constructs a SpinnerNumberModel with the specified value, minimum/maximum bounds, and stepSize.

**Parameters:**value - the current value of the modelminimum - the first number in the sequencemaximum - the last number in the sequencestepSize - the difference between elements of the sequence **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the following expression is false: minimum <= value <= maximum

### SpinnerNumberModel

public **SpinnerNumberModel**()

Constructs a SpinnerNumberModel with no minimum or maximum value, stepSize equal to one, and an initial value of zero.

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

### setMinimum

public void **setMinimum**([Comparable](http://docs.google.com/java/lang/Comparable.html) minimum)

Changes the lower bound for numbers in this sequence. If minimum is null, then there is no lower bound. No bounds checking is done here; the new minimum value may invalidate the (minimum <= value <= maximum) invariant enforced by the constructors. This is to simplify updating the model, naturally one should ensure that the invariant is true before calling the getNextValue, getPreviousValue, or setValue methods.

Typically this property is a Number of the same type as the value however it's possible to use any Comparable with a compareTo method for a Number with the same type as the value. For example if value was a Long, minimum might be a Date subclass defined like this:

MyDate extends Date { // Date already implements Comparable  
 public int compareTo(Long o) {  
 long t = getTime();  
 return (t < o.longValue() ? -1 : (t == o.longValue() ? 0 : 1));  
 }  
 }

This method fires a ChangeEvent if the minimum has changed.

**Parameters:**minimum - a Comparable that has a compareTo method for Numbers with the same type as value**See Also:**[getMinimum()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getMinimum()), [setMaximum(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setMaximum(java.lang.Comparable)), [SpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/SpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getMinimum

public [Comparable](http://docs.google.com/java/lang/Comparable.html) **getMinimum**()

Returns the first number in this sequence.

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

### setMaximum

public void **setMaximum**([Comparable](http://docs.google.com/java/lang/Comparable.html) maximum)

Changes the upper bound for numbers in this sequence. If maximum is null, then there is no upper bound. No bounds checking is done here; the new maximum value may invalidate the (minimum <= value < maximum) invariant enforced by the constructors. This is to simplify updating the model, naturally one should ensure that the invariant is true before calling the next, previous, or setValue methods.

Typically this property is a Number of the same type as the value however it's possible to use any Comparable with a compareTo method for a Number with the same type as the value. See  [setMinimum](#1ksv4uv) for an example.

This method fires a ChangeEvent if the maximum has changed.

**Parameters:**maximum - a Comparable that has a compareTo method for Numbers with the same type as value**See Also:**[getMaximum()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getMaximum()), [setMinimum(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setMinimum(java.lang.Comparable)), [SpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/SpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getMaximum

public [Comparable](http://docs.google.com/java/lang/Comparable.html) **getMaximum**()

Returns the last number in the sequence.

**Returns:**the value of the maximum property**See Also:**[setMaximum(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setMaximum(java.lang.Comparable))

### setStepSize

public void **setStepSize**([Number](http://docs.google.com/java/lang/Number.html) stepSize)

Changes the size of the value change computed by the getNextValue and getPreviousValue methods. An IllegalArgumentException is thrown if stepSize is null.

This method fires a ChangeEvent if the stepSize has changed.

**Parameters:**stepSize - the size of the value change computed by the getNextValue and getPreviousValue methods**See Also:**[getNextValue()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getNextValue()), [getPreviousValue()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getPreviousValue()), [getStepSize()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getStepSize()), [SpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/SpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getStepSize

public [Number](http://docs.google.com/java/lang/Number.html) **getStepSize**()

Returns the size of the value change computed by the getNextValue and getPreviousValue methods.

**Returns:**the value of the stepSize property**See Also:**[setStepSize(java.lang.Number)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setStepSize(java.lang.Number))

### getNextValue

public [Object](http://docs.google.com/java/lang/Object.html) **getNextValue**()

Returns the next number in the sequence.

**Specified by:**[getNextValue](http://docs.google.com/javax/swing/SpinnerModel.html#getNextValue()) in interface [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html) **Returns:**value + stepSize or null if the sum exceeds maximum.**See Also:**[SpinnerModel.getNextValue()](http://docs.google.com/javax/swing/SpinnerModel.html#getNextValue()), [getPreviousValue()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getPreviousValue()), [setStepSize(java.lang.Number)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setStepSize(java.lang.Number))

### getPreviousValue

public [Object](http://docs.google.com/java/lang/Object.html) **getPreviousValue**()

Returns the previous number in the sequence.

**Specified by:**[getPreviousValue](http://docs.google.com/javax/swing/SpinnerModel.html#getPreviousValue()) in interface [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html) **Returns:**value - stepSize, or null if the sum is less than minimum.**See Also:**[SpinnerModel.getPreviousValue()](http://docs.google.com/javax/swing/SpinnerModel.html#getPreviousValue()), [getNextValue()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getNextValue()), [setStepSize(java.lang.Number)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setStepSize(java.lang.Number))

### getNumber

public [Number](http://docs.google.com/java/lang/Number.html) **getNumber**()

Returns the value of the current element of the sequence.

**Returns:**the value property**See Also:**[setValue(java.lang.Object)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setValue(java.lang.Object))

### getValue

public [Object](http://docs.google.com/java/lang/Object.html) **getValue**()

Returns the value of the current element of the sequence.

**Specified by:**[getValue](http://docs.google.com/javax/swing/SpinnerModel.html#getValue()) in interface [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html) **Returns:**the value property**See Also:**[setValue(java.lang.Object)](http://docs.google.com/javax/swing/SpinnerNumberModel.html#setValue(java.lang.Object)), [getNumber()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getNumber())

### setValue

public void **setValue**([Object](http://docs.google.com/java/lang/Object.html) value)

Sets the current value for this sequence. If value is null, or not a Number, an IllegalArgumentException is thrown. No bounds checking is done here; the new value may invalidate the (minimum <= value <= maximum) invariant enforced by the constructors. It's also possible to set the value to be something that wouldn't naturally occur in the sequence, i.e. a value that's not modulo the stepSize. This is to simplify updating the model, and to accommodate spinners that don't want to restrict values that have been directly entered by the user. Naturally, one should ensure that the (minimum <= value <= maximum) invariant is true before calling the next, previous, or setValue methods.

This method fires a ChangeEvent if the value has changed.

**Specified by:**[setValue](http://docs.google.com/javax/swing/SpinnerModel.html#setValue(java.lang.Object)) in interface [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html) **Parameters:**value - the current (non null) Number for this sequence **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if value is null or not a Number**See Also:**[getNumber()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getNumber()), [getValue()](http://docs.google.com/javax/swing/SpinnerNumberModel.html#getValue()), [SpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/SpinnerModel.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/SpinnerNumberModel.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/SpinnerModel.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/Spring.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/SpinnerNumberModel.html)    [**NO FRAMES**](http://docs.google.com/SpinnerNumberModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#2s8eyo1) | [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).