| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/SpinnerDateModel.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/SortOrder.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/SpinnerListModel.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/SpinnerDateModel.html)    [**NO FRAMES**](http://docs.google.com/SpinnerDateModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#2s8eyo1) | [METHOD](#26in1rg) |

## **javax.swing**

Class SpinnerDateModel

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

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

public class **SpinnerDateModel**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 Dates. The upper and lower bounds of the sequence are defined by properties called start and end and the size of the increase or decrease computed by the nextValue and previousValue methods is defined by a property called calendarField. The start and end properties can be null to indicate that the sequence has no lower or upper limit.

The value of the calendarField property must be one of the java.util.Calendar constants that specify a field within a Calendar. The getNextValue and getPreviousValue methods change the date forward or backwards by this amount. For example, if calendarField is Calendar.DAY\_OF\_WEEK, then nextValue produces a Date that's 24 hours after the current value, and previousValue produces a Date that's 24 hours earlier.

The legal values for calendarField are:

* Calendar.ERA
* Calendar.YEAR
* Calendar.MONTH
* Calendar.WEEK\_OF\_YEAR
* Calendar.WEEK\_OF\_MONTH
* Calendar.DAY\_OF\_MONTH
* Calendar.DAY\_OF\_YEAR
* Calendar.DAY\_OF\_WEEK
* Calendar.DAY\_OF\_WEEK\_IN\_MONTH
* Calendar.AM\_PM
* Calendar.HOUR
* Calendar.HOUR\_OF\_DAY
* Calendar.MINUTE
* Calendar.SECOND
* Calendar.MILLISECOND

However some UIs may set the calendarField before commiting the edit to spin the field under the cursor. If you only want one field to spin you can subclass and ignore the setCalendarField calls.

This model inherits a ChangeListener. The ChangeListeners are notified whenever the models value, calendarField, start, or end 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), [SpinnerNumberModel](http://docs.google.com/javax/swing/SpinnerNumberModel.html), [Calendar.add(int, int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int))

| **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** | |
| --- | --- |
| [**SpinnerDateModel**](http://docs.google.com/javax/swing/SpinnerDateModel.html#SpinnerDateModel())()            Constructs a SpinnerDateModel whose initial value is the current date, calendarField is equal to Calendar.DAY\_OF\_MONTH, and for which there are no start/end limits. |
| [**SpinnerDateModel**](http://docs.google.com/javax/swing/SpinnerDateModel.html#SpinnerDateModel(java.util.Date,%20java.lang.Comparable,%20java.lang.Comparable,%20int))([Date](http://docs.google.com/java/util/Date.html) value, [Comparable](http://docs.google.com/java/lang/Comparable.html) start, [Comparable](http://docs.google.com/java/lang/Comparable.html) end, int calendarField)            Creates a SpinnerDateModel that represents a sequence of dates between start and end. |

| **Method Summary** | |
| --- | --- |
| int | [**getCalendarField**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getCalendarField())()            Returns the Calendar field that is added to or subtracted from by the nextValue and previousValue methods. |
| [Date](http://docs.google.com/java/util/Date.html) | [**getDate**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getDate())()            Returns the current element in this sequence of Dates. |
| [Comparable](http://docs.google.com/java/lang/Comparable.html) | [**getEnd**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getEnd())()            Returns the last Date in the sequence. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getNextValue**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getNextValue())()            Returns the next Date in the sequence, or null if the next date is after end. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getPreviousValue**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getPreviousValue())()            Returns the previous Date in the sequence, or null if the previous date is before start. |
| [Comparable](http://docs.google.com/java/lang/Comparable.html) | [**getStart**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getStart())()            Returns the first Date in the sequence. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getValue**](http://docs.google.com/javax/swing/SpinnerDateModel.html#getValue())()            Returns the current element in this sequence of Dates. |
| void | [**setCalendarField**](http://docs.google.com/javax/swing/SpinnerDateModel.html#setCalendarField(int))(int calendarField)            Changes the size of the date value change computed by the nextValue and previousValue methods. |
| void | [**setEnd**](http://docs.google.com/javax/swing/SpinnerDateModel.html#setEnd(java.lang.Comparable))([Comparable](http://docs.google.com/java/lang/Comparable.html) end)            Changes the upper limit for Dates in this sequence. |
| void | [**setStart**](http://docs.google.com/javax/swing/SpinnerDateModel.html#setStart(java.lang.Comparable))([Comparable](http://docs.google.com/java/lang/Comparable.html) start)            Changes the lower limit for Dates in this sequence. |
| void | [**setValue**](http://docs.google.com/javax/swing/SpinnerDateModel.html#setValue(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) value)            Sets the current Date 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** |
| --- |

### SpinnerDateModel

public **SpinnerDateModel**([Date](http://docs.google.com/java/util/Date.html) value,  
 [Comparable](http://docs.google.com/java/lang/Comparable.html) start,  
 [Comparable](http://docs.google.com/java/lang/Comparable.html) end,  
 int calendarField)

Creates a SpinnerDateModel that represents a sequence of dates between start and end. The nextValue and previousValue methods compute elements of the sequence by advancing or reversing the current date value by the calendarField time unit. For a precise description of what it means to increment or decrement a Calendar field, see the add method in java.util.Calendar.

The start and end parameters can be null to indicate that the range doesn't have an upper or lower bound. If value or calendarField is null, or if both start and end 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 modelstart - the first date in the sequence or nullend - the last date in the sequence or nullcalendarField - one of

* Calendar.ERA
* Calendar.YEAR
* Calendar.MONTH
* Calendar.WEEK\_OF\_YEAR
* Calendar.WEEK\_OF\_MONTH
* Calendar.DAY\_OF\_MONTH
* Calendar.DAY\_OF\_YEAR
* Calendar.DAY\_OF\_WEEK
* Calendar.DAY\_OF\_WEEK\_IN\_MONTH
* Calendar.AM\_PM
* Calendar.HOUR
* Calendar.HOUR\_OF\_DAY
* Calendar.MINUTE
* Calendar.SECOND
* Calendar.MILLISECOND

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if value or calendarField are null, if calendarField isn't valid, or if the following expression is false: (start <= value <= end).**See Also:**[Calendar.add(int, int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)), [setValue(java.lang.Object)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setValue(java.lang.Object)), [setStart(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setStart(java.lang.Comparable)), [setEnd(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setEnd(java.lang.Comparable)), [setCalendarField(int)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setCalendarField(int))

### SpinnerDateModel

public **SpinnerDateModel**()

Constructs a SpinnerDateModel whose initial value is the current date, calendarField is equal to Calendar.DAY\_OF\_MONTH, and for which there are no start/end limits.

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

### setStart

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

Changes the lower limit for Dates in this sequence. If start is null, then there is no lower limit. No bounds checking is done here: the new start value may invalidate the (start <= value <= end) invariant enforced by the constructors. This is to simplify updating the model. Naturally one should ensure that the invariant is true before calling the nextValue, previousValue, or setValue methods.

Typically this property is a Date however it's possible to use a Comparable with a compareTo method for Dates. For example start might be an instance of a class like this:

MyStartDate implements Comparable {   
 long t = 12345;  
 public int compareTo(Date d) {  
 return (t < d.getTime() ? -1 : (t == d.getTime() ? 0 : 1));  
 }  
 public int compareTo(Object o) {  
 return compareTo((Date)o);  
 }  
 }

Note that the above example will throw a ClassCastException if the Object passed to compareTo(Object) is not a Date.

This method fires a ChangeEvent if the start has changed.

**Parameters:**start - defines the first date in the sequence**See Also:**[getStart()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getStart()), [setEnd(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setEnd(java.lang.Comparable)), [AbstractSpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getStart

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

Returns the first Date in the sequence.

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

### setEnd

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

Changes the upper limit for Dates in this sequence. If start is null, then there is no upper limit. No bounds checking is done here: the new start value may invalidate the (start <= value <= end) invariant enforced by the constructors. This is to simplify updating the model. Naturally, one should ensure that the invariant is true before calling the nextValue, previousValue, or setValue methods.

Typically this property is a Date however it's possible to use Comparable with a compareTo method for Dates. See setStart for an example.

This method fires a ChangeEvent if the end has changed.

**Parameters:**end - defines the last date in the sequence**See Also:**[getEnd()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getEnd()), [setStart(java.lang.Comparable)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setStart(java.lang.Comparable)), [AbstractSpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getEnd

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

Returns the last Date in the sequence.

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

### setCalendarField

public void **setCalendarField**(int calendarField)

Changes the size of the date value change computed by the nextValue and previousValue methods. The calendarField parameter must be one of the Calendar field constants like Calendar.MONTH or Calendar.MINUTE. The nextValue and previousValue methods simply move the specified Calendar field forward or backward by one unit with the Calendar.add method. You should use this method with care as some UIs may set the calendarField before commiting the edit to spin the field under the cursor. If you only want one field to spin you can subclass and ignore the setCalendarField calls.

**Parameters:**calendarField - one of

* Calendar.ERA
* Calendar.YEAR
* Calendar.MONTH
* Calendar.WEEK\_OF\_YEAR
* Calendar.WEEK\_OF\_MONTH
* Calendar.DAY\_OF\_MONTH
* Calendar.DAY\_OF\_YEAR
* Calendar.DAY\_OF\_WEEK
* Calendar.DAY\_OF\_WEEK\_IN\_MONTH
* Calendar.AM\_PM
* Calendar.HOUR
* Calendar.HOUR\_OF\_DAY
* Calendar.MINUTE
* Calendar.SECOND
* Calendar.MILLISECOND

This method fires a ChangeEvent if the calendarField has changed.

**See Also:**[getCalendarField()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getCalendarField()), [getNextValue()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getNextValue()), [getPreviousValue()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getPreviousValue()), [Calendar.add(int, int)](http://docs.google.com/java/util/Calendar.html#add(int,%20int)), [AbstractSpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/AbstractSpinnerModel.html#addChangeListener(javax.swing.event.ChangeListener))

### getCalendarField

public int **getCalendarField**()

Returns the Calendar field that is added to or subtracted from by the nextValue and previousValue methods.

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

### getNextValue

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

Returns the next Date in the sequence, or null if the next date is after end.

**Specified by:**[getNextValue](http://docs.google.com/javax/swing/SpinnerModel.html#getNextValue()) in interface [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html) **Returns:**the next Date in the sequence, or null if the next date is after end.**See Also:**[SpinnerModel.getNextValue()](http://docs.google.com/javax/swing/SpinnerModel.html#getNextValue()), [getPreviousValue()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getPreviousValue()), [setCalendarField(int)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setCalendarField(int))

### getPreviousValue

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

Returns the previous Date in the sequence, or null if the previous date is before start.

**Specified by:**[getPreviousValue](http://docs.google.com/javax/swing/SpinnerModel.html#getPreviousValue()) in interface [SpinnerModel](http://docs.google.com/javax/swing/SpinnerModel.html) **Returns:**the previous Date in the sequence, or null if the previous date is before start**See Also:**[SpinnerModel.getPreviousValue()](http://docs.google.com/javax/swing/SpinnerModel.html#getPreviousValue()), [getNextValue()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getNextValue()), [setCalendarField(int)](http://docs.google.com/javax/swing/SpinnerDateModel.html#setCalendarField(int))

### getDate

public [Date](http://docs.google.com/java/util/Date.html) **getDate**()

Returns the current element in this sequence of Dates. This method is equivalent to (Date)getValue.

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

### getValue

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

Returns the current element in this sequence of Dates.

**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/SpinnerDateModel.html#setValue(java.lang.Object)), [getDate()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getDate())

### setValue

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

Sets the current Date for this sequence. If value is null, an IllegalArgumentException is thrown. No bounds checking is done here: the new value may invalidate the (start <= value < end) invariant enforced by the constructors. Naturally, one should ensure that the (start <= value <= maximum) invariant is true before calling the nextValue, previousValue, 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) Date for this sequence **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if value is null or not a Date**See Also:**[getDate()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getDate()), [getValue()](http://docs.google.com/javax/swing/SpinnerDateModel.html#getValue()), [AbstractSpinnerModel.addChangeListener(javax.swing.event.ChangeListener)](http://docs.google.com/javax/swing/AbstractSpinnerModel.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/SpinnerDateModel.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/SortOrder.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/SpinnerListModel.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/SpinnerDateModel.html)    [**NO FRAMES**](http://docs.google.com/SpinnerDateModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#2s8eyo1) | [METHOD](#26in1rg) |

[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).