| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RowSorter.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/RowFilter.Entry.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/RowSorter.SortKey.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/RowSorter.html)    [**NO FRAMES**](http://docs.google.com/RowSorter.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#2s8eyo1) |

## **javax.swing**

Class RowSorter<M>

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.swing.RowSorter<M>**

**Type Parameters:**M - the type of the underlying model **Direct Known Subclasses:** [DefaultRowSorter](http://docs.google.com/javax/swing/DefaultRowSorter.html)

public abstract class **RowSorter<M>**extends [Object](http://docs.google.com/java/lang/Object.html)

RowSorter provides the basis for sorting and filtering. Beyond creating and installing a RowSorter, you very rarely need to interact with one directly. Refer to [TableRowSorter](http://docs.google.com/javax/swing/table/TableRowSorter.html) for a concrete implementation of RowSorter for JTable.

RowSorter's primary role is to provide a mapping between two coordinate systems: that of the view (for example a JTable) and that of the underlying data source, typically a model.

The view invokes the following methods on the RowSorter:

* toggleSortOrder — The view invokes this when the appropriate user gesture has occurred to trigger a sort. For example, the user clicked a column header in a table.
* One of the model change methods — The view invokes a model change method when the underlying model has changed. There may be order dependencies in how the events are delivered, so a RowSorter should not update its mapping until one of these methods is invoked.

Because the view makes extensive use of the convertRowIndexToModel, convertRowIndexToView and getViewRowCount methods, these methods need to be fast.

RowSorter provides notification of changes by way of RowSorterListener. Two types of notification are sent:

* RowSorterEvent.Type.SORT\_ORDER\_CHANGED — notifies listeners that the sort order has changed. This is typically followed by a notification that the sort has changed.
* RowSorterEvent.Type.SORTED — notifies listeners that the mapping maintained by the RowSorter has changed in some way.

RowSorter implementations typically don't have a one-to-one mapping with the underlying model, but they can. For example, if a database does the sorting, toggleSortOrder might call through to the database (on a background thread), and override the mapping methods to return the argument that is passed in.

Concrete implementations of RowSorter need to reference a model such as TableModel or ListModel. The view classes, such as JTable and JList, will also have a reference to the model. To avoid ordering dependencies, RowSorter implementations should not install a listener on the model. Instead the view class will call into the RowSorter when the model changes. For example, if a row is updated in a TableModel JTable invokes rowsUpdated. When the model changes, the view may call into any of the following methods: modelStructureChanged, allRowsChanged, rowsInserted, rowsDeleted and rowsUpdated.

**Since:** 1.6 **See Also:**[TableRowSorter](http://docs.google.com/javax/swing/table/TableRowSorter.html)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**RowSorter.SortKey**](http://docs.google.com/javax/swing/RowSorter.SortKey.html)            SortKey describes the sort order for a particular column. |

| **Constructor Summary** | |
| --- | --- |
| [**RowSorter**](http://docs.google.com/javax/swing/RowSorter.html#RowSorter())()            Creates a RowSorter. |

| **Method Summary** | |
| --- | --- |
| void | [**addRowSorterListener**](http://docs.google.com/javax/swing/RowSorter.html#addRowSorterListener(javax.swing.event.RowSorterListener))([RowSorterListener](http://docs.google.com/javax/swing/event/RowSorterListener.html) l)            Adds a RowSorterListener to receive notification about this RowSorter. |
| abstract  void | [**allRowsChanged**](http://docs.google.com/javax/swing/RowSorter.html#allRowsChanged())()            Invoked when the contents of the underlying model have completely changed. |
| abstract  int | [**convertRowIndexToModel**](http://docs.google.com/javax/swing/RowSorter.html#convertRowIndexToModel(int))(int index)            Returns the location of index in terms of the underlying model. |
| abstract  int | [**convertRowIndexToView**](http://docs.google.com/javax/swing/RowSorter.html#convertRowIndexToView(int))(int index)            Returns the location of index in terms of the view. |
| protected  void | [**fireRowSorterChanged**](http://docs.google.com/javax/swing/RowSorter.html#fireRowSorterChanged(int%5B%5D))(int[] lastRowIndexToModel)            Notifies listener that the mapping has changed. |
| protected  void | [**fireSortOrderChanged**](http://docs.google.com/javax/swing/RowSorter.html#fireSortOrderChanged())()            Notifies listener that the sort order has changed. |
| abstract  [M](http://docs.google.com/javax/swing/RowSorter.html) | [**getModel**](http://docs.google.com/javax/swing/RowSorter.html#getModel())()            Returns the underlying model. |
| abstract  int | [**getModelRowCount**](http://docs.google.com/javax/swing/RowSorter.html#getModelRowCount())()            Returns the number of rows in the underlying model. |
| abstract  [List](http://docs.google.com/java/util/List.html)<? extends [RowSorter.SortKey](http://docs.google.com/javax/swing/RowSorter.SortKey.html)> | [**getSortKeys**](http://docs.google.com/javax/swing/RowSorter.html#getSortKeys())()            Returns the current sort keys. |
| abstract  int | [**getViewRowCount**](http://docs.google.com/javax/swing/RowSorter.html#getViewRowCount())()            Returns the number of rows in the view. |
| abstract  void | [**modelStructureChanged**](http://docs.google.com/javax/swing/RowSorter.html#modelStructureChanged())()            Invoked when the underlying model structure has completely changed. |
| void | [**removeRowSorterListener**](http://docs.google.com/javax/swing/RowSorter.html#removeRowSorterListener(javax.swing.event.RowSorterListener))([RowSorterListener](http://docs.google.com/javax/swing/event/RowSorterListener.html) l)            Removes a RowSorterListener. |
| abstract  void | [**rowsDeleted**](http://docs.google.com/javax/swing/RowSorter.html#rowsDeleted(int,%20int))(int firstRow, int endRow)            Invoked when rows have been deleted from the underlying model in the specified range (inclusive). |
| abstract  void | [**rowsInserted**](http://docs.google.com/javax/swing/RowSorter.html#rowsInserted(int,%20int))(int firstRow, int endRow)            Invoked when rows have been inserted into the underlying model in the specified range (inclusive). |
| abstract  void | [**rowsUpdated**](http://docs.google.com/javax/swing/RowSorter.html#rowsUpdated(int,%20int))(int firstRow, int endRow)            Invoked when rows have been changed in the underlying model between the specified range (inclusive). |
| abstract  void | [**rowsUpdated**](http://docs.google.com/javax/swing/RowSorter.html#rowsUpdated(int,%20int,%20int))(int firstRow, int endRow, int column)            Invoked when the column in the rows have been updated in the underlying model between the specified range. |
| abstract  void | [**setSortKeys**](http://docs.google.com/javax/swing/RowSorter.html#setSortKeys(java.util.List))([List](http://docs.google.com/java/util/List.html)<? extends [RowSorter.SortKey](http://docs.google.com/javax/swing/RowSorter.SortKey.html)> keys)            Sets the current sort keys. |
| abstract  void | [**toggleSortOrder**](http://docs.google.com/javax/swing/RowSorter.html#toggleSortOrder(int))(int column)            Reverses the sort order of the specified column. |

| **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** |
| --- |

### RowSorter

public **RowSorter**()

Creates a RowSorter.

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

### getModel

public abstract [M](http://docs.google.com/javax/swing/RowSorter.html) **getModel**()

Returns the underlying model.

**Returns:**the underlying model

### toggleSortOrder

public abstract void **toggleSortOrder**(int column)

Reverses the sort order of the specified column. It is up to subclasses to provide the exact behavior when invoked. Typically this will reverse the sort order from ascending to descending (or descending to ascending) if the specified column is already the primary sorted column; otherwise, makes the specified column the primary sorted column, with an ascending sort order. If the specified column is not sortable, this method has no effect.

If this results in changing the sort order and sorting, the appropriate RowSorterListener notification will be sent.

**Parameters:**column - the column to toggle the sort ordering of, in terms of the underlying model **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if column is outside the range of the underlying model

### convertRowIndexToModel

public abstract int **convertRowIndexToModel**(int index)

Returns the location of index in terms of the underlying model. That is, for the row index in the coordinates of the view this returns the row index in terms of the underlying model.

**Parameters:**index - the row index in terms of the underlying view **Returns:**row index in terms of the view **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is outside the range of the view

### convertRowIndexToView

public abstract int **convertRowIndexToView**(int index)

Returns the location of index in terms of the view. That is, for the row index in the coordinates of the underlying model this returns the row index in terms of the view.

**Parameters:**index - the row index in terms of the underlying model **Returns:**row index in terms of the view, or -1 if index has been filtered out of the view **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if index is outside the range of the model

### setSortKeys

public abstract void **setSortKeys**([List](http://docs.google.com/java/util/List.html)<? extends [RowSorter.SortKey](http://docs.google.com/javax/swing/RowSorter.SortKey.html)> keys)

Sets the current sort keys.

**Parameters:**keys - the new SortKeys; null is a shorthand for specifying an empty list, indicating that the view should be unsorted

### getSortKeys

public abstract [List](http://docs.google.com/java/util/List.html)<? extends [RowSorter.SortKey](http://docs.google.com/javax/swing/RowSorter.SortKey.html)> **getSortKeys**()

Returns the current sort keys. This must return a non-null List and may return an unmodifiable List. If you need to change the sort keys, make a copy of the returned List, mutate the copy and invoke setSortKeys with the new list.

**Returns:**the current sort order

### getViewRowCount

public abstract int **getViewRowCount**()

Returns the number of rows in the view. If the contents have been filtered this might differ from the row count of the underlying model.

**Returns:**number of rows in the view**See Also:**[getModelRowCount()](http://docs.google.com/javax/swing/RowSorter.html#getModelRowCount())

### getModelRowCount

public abstract int **getModelRowCount**()

Returns the number of rows in the underlying model.

**Returns:**number of rows in the underlying model**See Also:**[getViewRowCount()](http://docs.google.com/javax/swing/RowSorter.html#getViewRowCount())

### modelStructureChanged

public abstract void **modelStructureChanged**()

Invoked when the underlying model structure has completely changed. For example, if the number of columns in a TableModel changed, this method would be invoked.

You normally do not call this method. This method is public to allow view classes to call it.

### allRowsChanged

public abstract void **allRowsChanged**()

Invoked when the contents of the underlying model have completely changed. The structure of the table is the same, only the contents have changed. This is typically sent when it is too expensive to characterize the change in terms of the other methods.

You normally do not call this method. This method is public to allow view classes to call it.

### rowsInserted

public abstract void **rowsInserted**(int firstRow,  
 int endRow)

Invoked when rows have been inserted into the underlying model in the specified range (inclusive).

The arguments give the indices of the effected range. The first argument is in terms of the model before the change, and must be less than or equal to the size of the model before the change. The second argument is in terms of the model after the change and must be less than the size of the model after the change. For example, if you have a 5-row model and add 3 items to the end of the model the indices are 5, 7.

You normally do not call this method. This method is public to allow view classes to call it.

**Parameters:**firstRow - the first rowendRow - the last row **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if either argument is invalid, or firstRow > endRow

### rowsDeleted

public abstract void **rowsDeleted**(int firstRow,  
 int endRow)

Invoked when rows have been deleted from the underlying model in the specified range (inclusive).

The arguments give the indices of the effected range and are in terms of the model **before** the change. For example, if you have a 5-row model and delete 3 items from the end of the model the indices are 2, 4.

You normally do not call this method. This method is public to allow view classes to call it.

**Parameters:**firstRow - the first rowendRow - the last row **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if either argument is outside the range of the model before the change, or firstRow > endRow

### rowsUpdated

public abstract void **rowsUpdated**(int firstRow,  
 int endRow)

Invoked when rows have been changed in the underlying model between the specified range (inclusive).

You normally do not call this method. This method is public to allow view classes to call it.

**Parameters:**firstRow - the first row, in terms of the underlying modelendRow - the last row, in terms of the underlying model **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if either argument is outside the range of the underlying model, or firstRow > endRow

### rowsUpdated

public abstract void **rowsUpdated**(int firstRow,  
 int endRow,  
 int column)

Invoked when the column in the rows have been updated in the underlying model between the specified range.

You normally do not call this method. This method is public to allow view classes to call it.

**Parameters:**firstRow - the first row, in terms of the underlying modelendRow - the last row, in terms of the underlying modelcolumn - the column that has changed, in terms of the underlying model **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if either argument is outside the range of the underlying model after the change, firstRow > endRow, or column is outside the range of the underlying model

### addRowSorterListener

public void **addRowSorterListener**([RowSorterListener](http://docs.google.com/javax/swing/event/RowSorterListener.html) l)

Adds a RowSorterListener to receive notification about this RowSorter. If the same listener is added more than once it will receive multiple notifications. If l is null nothing is done.

**Parameters:**l - the RowSorterListener

### removeRowSorterListener

public void **removeRowSorterListener**([RowSorterListener](http://docs.google.com/javax/swing/event/RowSorterListener.html) l)

Removes a RowSorterListener. If l is null nothing is done.

**Parameters:**l - the RowSorterListener

### fireSortOrderChanged

protected void **fireSortOrderChanged**()

Notifies listener that the sort order has changed.

### fireRowSorterChanged

protected void **fireRowSorterChanged**(int[] lastRowIndexToModel)

Notifies listener that the mapping has changed.

**Parameters:**lastRowIndexToModel - the mapping from model indices to view indices prior to the sort, may be null

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/RowSorter.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/RowFilter.Entry.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/RowSorter.SortKey.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/RowSorter.html)    [**NO FRAMES**](http://docs.google.com/RowSorter.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | FIELD | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: FIELD | [CONSTR](#1t3h5sf) | [METHOD](#2s8eyo1) |

[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).