| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AbstractTableModel.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   [**NEXT CLASS**](http://docs.google.com/javax/swing/table/DefaultTableCellRenderer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/table/AbstractTableModel.html)    [**NO FRAMES**](http://docs.google.com/AbstractTableModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#4d34og8) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

## **javax.swing.table**

Class AbstractTableModel

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

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Direct Known Subclasses:** [DefaultTableModel](http://docs.google.com/javax/swing/table/DefaultTableModel.html)

public abstract class **AbstractTableModel**extends [Object](http://docs.google.com/java/lang/Object.html)implements [TableModel](http://docs.google.com/javax/swing/table/TableModel.html), [Serializable](http://docs.google.com/java/io/Serializable.html)

This abstract class provides default implementations for most of the methods in the TableModel interface. It takes care of the management of listeners and provides some conveniences for generating TableModelEvents and dispatching them to the listeners. To create a concrete TableModel as a subclass of AbstractTableModel you need only provide implementations for the following three methods:

public int getRowCount();  
 public int getColumnCount();  
 public Object getValueAt(int row, int column);

**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).

| **Field Summary** | |
| --- | --- |
| protected  [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html) | [**listenerList**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#listenerList)            List of listeners |

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

| **Method Summary** | |
| --- | --- |
| void | [**addTableModelListener**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#addTableModelListener(javax.swing.event.TableModelListener))([TableModelListener](http://docs.google.com/javax/swing/event/TableModelListener.html) l)            Adds a listener to the list that's notified each time a change to the data model occurs. |
| int | [**findColumn**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#findColumn(java.lang.String))([String](http://docs.google.com/java/lang/String.html) columnName)            Returns a column given its name. |
| void | [**fireTableCellUpdated**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableCellUpdated(int,%20int))(int row, int column)            Notifies all listeners that the value of the cell at [row, column] has been updated. |
| void | [**fireTableChanged**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableChanged(javax.swing.event.TableModelEvent))([TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html) e)            Forwards the given notification event to all TableModelListeners that registered themselves as listeners for this table model. |
| void | [**fireTableDataChanged**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableDataChanged())()            Notifies all listeners that all cell values in the table's rows may have changed. |
| void | [**fireTableRowsDeleted**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableRowsDeleted(int,%20int))(int firstRow, int lastRow)            Notifies all listeners that rows in the range [firstRow, lastRow], inclusive, have been deleted. |
| void | [**fireTableRowsInserted**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableRowsInserted(int,%20int))(int firstRow, int lastRow)            Notifies all listeners that rows in the range [firstRow, lastRow], inclusive, have been inserted. |
| void | [**fireTableRowsUpdated**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableRowsUpdated(int,%20int))(int firstRow, int lastRow)            Notifies all listeners that rows in the range [firstRow, lastRow], inclusive, have been updated. |
| void | [**fireTableStructureChanged**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#fireTableStructureChanged())()            Notifies all listeners that the table's structure has changed. |
| [Class](http://docs.google.com/java/lang/Class.html)<?> | [**getColumnClass**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#getColumnClass(int))(int columnIndex)            Returns Object.class regardless of columnIndex. |
| [String](http://docs.google.com/java/lang/String.html) | [**getColumnName**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#getColumnName(int))(int column)            Returns a default name for the column using spreadsheet conventions: A, B, C, ... |
| | <T extends [EventListener](http://docs.google.com/java/util/EventListener.html)>  T[] | | --- | | [**getListeners**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#getListeners(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<T> listenerType)            Returns an array of all the objects currently registered as *Foo*Listeners upon this AbstractTableModel. |
| [TableModelListener](http://docs.google.com/javax/swing/event/TableModelListener.html)[] | [**getTableModelListeners**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#getTableModelListeners())()            Returns an array of all the table model listeners registered on this model. |
| boolean | [**isCellEditable**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#isCellEditable(int,%20int))(int rowIndex, int columnIndex)            Returns false. |
| void | [**removeTableModelListener**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#removeTableModelListener(javax.swing.event.TableModelListener))([TableModelListener](http://docs.google.com/javax/swing/event/TableModelListener.html) l)            Removes a listener from the list that's notified each time a change to the data model occurs. |
| void | [**setValueAt**](http://docs.google.com/javax/swing/table/AbstractTableModel.html#setValueAt(java.lang.Object,%20int,%20int))([Object](http://docs.google.com/java/lang/Object.html) aValue, int rowIndex, int columnIndex)            This empty implementation is provided so users don't have to implement this method if their data model is not editable. |

| **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)) |

| **Methods inherited from interface javax.swing.table.**[**TableModel**](http://docs.google.com/javax/swing/table/TableModel.html) |
| --- |
| [getColumnCount](http://docs.google.com/javax/swing/table/TableModel.html#getColumnCount()), [getRowCount](http://docs.google.com/javax/swing/table/TableModel.html#getRowCount()), [getValueAt](http://docs.google.com/javax/swing/table/TableModel.html#getValueAt(int,%20int)) |

| **Field Detail** |
| --- |

### listenerList

protected [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html) **listenerList**

List of listeners

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

### AbstractTableModel

public **AbstractTableModel**()

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

### getColumnName

public [String](http://docs.google.com/java/lang/String.html) **getColumnName**(int column)

Returns a default name for the column using spreadsheet conventions: A, B, C, ... Z, AA, AB, etc. If column cannot be found, returns an empty string.

**Specified by:**[getColumnName](http://docs.google.com/javax/swing/table/TableModel.html#getColumnName(int)) in interface [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Parameters:**column - the column being queried **Returns:**a string containing the default name of column

### findColumn

public int **findColumn**([String](http://docs.google.com/java/lang/String.html) columnName)

Returns a column given its name. Implementation is naive so this should be overridden if this method is to be called often. This method is not in the TableModel interface and is not used by the JTable.

**Parameters:**columnName - string containing name of column to be located **Returns:**the column with columnName, or -1 if not found

### getColumnClass

public [Class](http://docs.google.com/java/lang/Class.html)<?> **getColumnClass**(int columnIndex)

Returns Object.class regardless of columnIndex.

**Specified by:**[getColumnClass](http://docs.google.com/javax/swing/table/TableModel.html#getColumnClass(int)) in interface [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Parameters:**columnIndex - the column being queried **Returns:**the Object.class

### isCellEditable

public boolean **isCellEditable**(int rowIndex,  
 int columnIndex)

Returns false. This is the default implementation for all cells.

**Specified by:**[isCellEditable](http://docs.google.com/javax/swing/table/TableModel.html#isCellEditable(int,%20int)) in interface [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Parameters:**rowIndex - the row being queriedcolumnIndex - the column being queried **Returns:**false**See Also:**[TableModel.setValueAt(java.lang.Object, int, int)](http://docs.google.com/javax/swing/table/TableModel.html#setValueAt(java.lang.Object,%20int,%20int))

### setValueAt

public void **setValueAt**([Object](http://docs.google.com/java/lang/Object.html) aValue,  
 int rowIndex,  
 int columnIndex)

This empty implementation is provided so users don't have to implement this method if their data model is not editable.

**Specified by:**[setValueAt](http://docs.google.com/javax/swing/table/TableModel.html#setValueAt(java.lang.Object,%20int,%20int)) in interface [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Parameters:**aValue - value to assign to cellrowIndex - row of cellcolumnIndex - column of cell**See Also:**[TableModel.getValueAt(int, int)](http://docs.google.com/javax/swing/table/TableModel.html#getValueAt(int,%20int)), [TableModel.isCellEditable(int, int)](http://docs.google.com/javax/swing/table/TableModel.html#isCellEditable(int,%20int))

### addTableModelListener

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

Adds a listener to the list that's notified each time a change to the data model occurs.

**Specified by:**[addTableModelListener](http://docs.google.com/javax/swing/table/TableModel.html#addTableModelListener(javax.swing.event.TableModelListener)) in interface [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Parameters:**l - the TableModelListener

### removeTableModelListener

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

Removes a listener from the list that's notified each time a change to the data model occurs.

**Specified by:**[removeTableModelListener](http://docs.google.com/javax/swing/table/TableModel.html#removeTableModelListener(javax.swing.event.TableModelListener)) in interface [TableModel](http://docs.google.com/javax/swing/table/TableModel.html) **Parameters:**l - the TableModelListener

### getTableModelListeners

public [TableModelListener](http://docs.google.com/javax/swing/event/TableModelListener.html)[] **getTableModelListeners**()

Returns an array of all the table model listeners registered on this model.

**Returns:**all of this model's TableModelListeners or an empty array if no table model listeners are currently registered**Since:** 1.4 **See Also:**[addTableModelListener(javax.swing.event.TableModelListener)](http://docs.google.com/javax/swing/table/AbstractTableModel.html#addTableModelListener(javax.swing.event.TableModelListener)), [removeTableModelListener(javax.swing.event.TableModelListener)](http://docs.google.com/javax/swing/table/AbstractTableModel.html#removeTableModelListener(javax.swing.event.TableModelListener))

### fireTableDataChanged

public void **fireTableDataChanged**()

Notifies all listeners that all cell values in the table's rows may have changed. The number of rows may also have changed and the JTable should redraw the table from scratch. The structure of the table (as in the order of the columns) is assumed to be the same.

**See Also:**[TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html), [JTable.tableChanged(TableModelEvent)](http://docs.google.com/javax/swing/JTable.html#tableChanged(javax.swing.event.TableModelEvent))

### fireTableStructureChanged

public void **fireTableStructureChanged**()

Notifies all listeners that the table's structure has changed. The number of columns in the table, and the names and types of the new columns may be different from the previous state. If the JTable receives this event and its autoCreateColumnsFromModel flag is set it discards any table columns that it had and reallocates default columns in the order they appear in the model. This is the same as calling setModel(TableModel) on the JTable.

**See Also:**[TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTableRowsInserted

public void **fireTableRowsInserted**(int firstRow,  
 int lastRow)

Notifies all listeners that rows in the range [firstRow, lastRow], inclusive, have been inserted.

**Parameters:**firstRow - the first rowlastRow - the last row**See Also:**[TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTableRowsUpdated

public void **fireTableRowsUpdated**(int firstRow,  
 int lastRow)

Notifies all listeners that rows in the range [firstRow, lastRow], inclusive, have been updated.

**Parameters:**firstRow - the first rowlastRow - the last row**See Also:**[TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTableRowsDeleted

public void **fireTableRowsDeleted**(int firstRow,  
 int lastRow)

Notifies all listeners that rows in the range [firstRow, lastRow], inclusive, have been deleted.

**Parameters:**firstRow - the first rowlastRow - the last row**See Also:**[TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTableCellUpdated

public void **fireTableCellUpdated**(int row,  
 int column)

Notifies all listeners that the value of the cell at [row, column] has been updated.

**Parameters:**row - row of cell which has been updatedcolumn - column of cell which has been updated**See Also:**[TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### fireTableChanged

public void **fireTableChanged**([TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html) e)

Forwards the given notification event to all TableModelListeners that registered themselves as listeners for this table model.

**Parameters:**e - the event to be forwarded**See Also:**[addTableModelListener(javax.swing.event.TableModelListener)](http://docs.google.com/javax/swing/table/AbstractTableModel.html#addTableModelListener(javax.swing.event.TableModelListener)), [TableModelEvent](http://docs.google.com/javax/swing/event/TableModelEvent.html), [EventListenerList](http://docs.google.com/javax/swing/event/EventListenerList.html)

### getListeners

public <T extends [EventListener](http://docs.google.com/java/util/EventListener.html)> T[] **getListeners**([Class](http://docs.google.com/java/lang/Class.html)<T> listenerType)

Returns an array of all the objects currently registered as *Foo*Listeners upon this AbstractTableModel. *Foo*Listeners are registered using the add*Foo*Listener method.

You can specify the listenerType argument with a class literal, such as *Foo*Listener.class. For example, you can query a model m for its table model listeners with the following code:

TableModelListener[] tmls = (TableModelListener[])(m.getListeners(TableModelListener.class));

If no such listeners exist, this method returns an empty array.

**Parameters:**listenerType - the type of listeners requested; this parameter should specify an interface that descends from java.util.EventListener **Returns:**an array of all objects registered as *Foo*Listeners on this component, or an empty array if no such listeners have been added **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if listenerType doesn't specify a class or interface that implements java.util.EventListener**Since:** 1.3 **See Also:**[getTableModelListeners()](http://docs.google.com/javax/swing/table/AbstractTableModel.html#getTableModelListeners())

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AbstractTableModel.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   [**NEXT CLASS**](http://docs.google.com/javax/swing/table/DefaultTableCellRenderer.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/table/AbstractTableModel.html)    [**NO FRAMES**](http://docs.google.com/AbstractTableModel.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#4d34og8) | [CONSTR](#17dp8vu) | [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).