NSTableColumn

Inherits From:	NSObject
Conforms To:	NSObject (NSObject)
Declared In:	AppKit/NSTableColumn.h

Class Description

An NSTableColumn stores the display characteristics and attribute identifier for a column in an NSTableView. The NSTableColumn determines the width and width limits, resizability, and editability of its column in the NSTableView. It also stores two NSCell objects: the *header cell*, which is used to draw the column header, and the *data cell*, used to draw the values for each row. You can control the display of the column by setting the subclasses of NSCell used and by setting the font and other display characteristics for these NSCells. For example, you can use the default NSTextFieldCell for displaying string values or substitute an NSImageCell to display pictures.

See the NSTableView class specification for a general overview.

Method Types

Creating an NSTableColumn instance	– initWithIdentifier:
Setting the identifier	– setIdentifier: – identifier
Setting the NSTableView	– setTableView: – tableView
Controlling size	– setWidth: – width – setMinWidth: – minWidth – setMaxWidth: – maxWidth – setResizable: – isResizable – sizeToFit

Controlling editability	– setEditable: – isEditable
Setting component cells	– setHeaderCell: – headerCell – setDataCell: – dataCell

Instance Methods

dataCell

– (id)**dataCell**

Returns the NSCell object used by the NSTableView to draw values for the NSTableColumn.

See also: – **setDataCell:**

headerCell

– (id)**headerCell**

Returns the NSTableHeaderCell object used to draw the header of the NSTableColumn. You can set the column title by sending **setStringValue:** to this object.

See also: – **setHeaderCell:**

initWithIdentifier:

– (id)**initWithIdentifier:***anObject*

Initializes a newly created NSTableColumn with *anObject* as its identifier and with an NSTextFieldCell as its data cell. Send **setStringValue:** to the header cell to set the column title. This is the designated initializer for the NSTableColumn class. Returns **self**.

See the NSTableView class specification for information on identifiers.

See also: – **setIdentifier:**

identifier

– (id)**identifier**

Returns the object used by the data source to identify the attribute corresponding to the NSTableColumn.

See also: – **setIdentifier:**

isEditable

– (BOOL)**isEditable**

Returns YES if the user can edit cells associated with the NSTableColumn by double-clicking the column in the NSTableView, NO otherwise. You can initiate editing programmatically regardless of this setting with NSTableView's **editColumn:row:withEvent:select:** method.

See also: – **setEditable:**

isResizable

– (BOOL)**isResizable**

Returns YES if the user is allowed to resize the NSTableColumn in its NSTableView, NO otherwise. You can change the size programmatically regardless of this setting.

See also: – **setWidth:**, – **setMinWidth:**, – **setMaxWidth:**, – **setResizable:**

maxWidth

– (float)**maxWidth**

Returns the maximum width for the NSTableColumn. The NSTableColumn's width can't be made larger than this either by the user or programmatically.

See also: – **minWidth**, – **width**, – **setMaxWidth:**, – **sizeToFit** (NSTableView),
– **autoresizesAllColumnsToFit** (NSTableView)

minWidth

– (float)**minWidth**

Returns the minimum width for the NSTableColumn. The NSTableColumn's width can't be made less than this either by the user or programmatically.

See also: – **maxWidth**, – **width**, – **setMinWidth:**, – **sizeToFit** (NSTableView),
– **autoresizesAllColumnsToFit** (NSTableView)

setDataCell:

– (void)**setDataCell:**(NSCell *)*aCell*

Sets the NSCell used by the NSTableView to draw individual values for the NSTableColumn to *aCell*. You can use this method to control the font, alignment, and other text attributes for an NSTableColumn. You can also assign a cell to display things other than text—for example, an NSImageCell to display images.

See also: – **dataCell**

setEditable:

– (void)**setEditable:**(BOOL)*flag*

Controls whether the user can edit cells in the receiver by double-clicking them. If *flag* is YES a double click initiates editing; if *flag* is NO it merely sends the double action to the NSTableView's target. You can initiate editing programmatically regardless of this setting with NSTableView's **editColumn:row:withEvent:select:** method.

See also: – **isEditable**

setHeaderCell:

– (void)**setHeaderCell:**(NSCell *)*aCell*

Sets the NSCell used to draw the NSTableColumn's header to *aCell*. *aCell* should never be **nil**.

See also: – **headerCell**

setIdentifier:

– (void)**setIdentifier:**(id)*anObject*

Sets the NSTableColumn's identifier to *anObject*. This object is used by the data source to identify the attribute corresponding to the NSTableColumn.

See also: – **identifier**

setMaxWidth:

– (void)**setMaxWidth:**(float)*maxWidth*

Sets the NSTableColumn's maximum width to *maxWidth*, also adjusting the current width if it's greater than this value. The NSTableView can be made no wider than this, either by the user or programmatically.

See also: – **setMinWidth:**, – **setWidth:**, – **maxWidth**, – **sizeToFit** (NSTableView), – **autoresizesAllColumnsToFit** (NSTableView)

setMinWidth:

– (void)**setMinWidth:**(float)*minWidth*

Sets the NSTableColumn's minimum width to *minWidth*, also adjusting the current width if it's less than this value. The NSTableView can be made no less wide than this, either by the user or programmatically.

See also: – **setMaxWidth:**, – **setWidth:**, – **minWidth**, – **sizeToFit** (NSTableView), – **autoresizesAllColumnsToFit** (NSTableView)

setResizable:

– (void)**setResizable:**(BOOL)*flag*

Sets whether the user can resize the receiver in its NSTableView. If *flag* is YES the user can resize the receiver; if *flag* is NO the user can't resize it (though you can set the size programmatically).

See also: – **isResizable**, – **setWidth:**, – **setMinWidth:**, – **setMaxWidth:**

setTableView:

– (void)**setTableView:**(NSTableView *)*aTableView*

Sets *aTableView* as the NSTableColumn's NSTableView. You should never need to invoke this method; it's invoked automatically when you add an NSTableColumn to an NSTableView.

See also: – **tableView**, – **addTableColumn:** (NSTableView)

setWidth:

– (void)**setWidth:**(float)*newWidth*

Sets the NSTableColumn's width to *newWidth*. If *newWidth* exceeds the minimum or maximum width, it's adjusted to the appropriate limiting value. Marks the NSTableView as needing display.

This method posts NSTableViewColumnDidResizeNotification on behalf of the NSTableColumn's NSTableView.

See also: – **width**, – **setMinWidth:**, – **setMaxWidth:**, – **sizeToFit** (NSTableView), – **autoresizesAllColumnsToFit** (NSTableView)

sizeToFit

– (void)**sizeToFit**

Resizes the NSTableColumn to fit the width of its header cell. If the maximum width is less than the width of the header, the maximum is increased to the header's width. Similarly, if the minimum width is greater than the width of the header, the minimum is reduced to the header's width. Marks the NSTableView as needing display if the width actually changes.

See also: – **width**, – **minWidth**, – **maxWidth**, – **sizeToFit** (NSTableView), – **autoresizesAllColumnsToFit** (NSTableView)

tableView

– (NSTableView *)**tableView**

Returns the NSTableView that the NSTableColumn belongs to.

See also: – **setTableView:**

width

– (float)**width**

Returns the width of the NSTableColumn.

See also: – **width**