)initWithFrame:(NSRect)frameRect	Initializes a new NSMatrix object in frameRect.
)initWithFrame:(NSRect)frameRect mode:(int)aMode cellClass:(Class)classId numberOfRows:(int)rowsHigh numberOfColumns:(int)colsWide	Initializes a new NSMatrix object in frameRect, with aMode as the selection mode, classId as the class used to make new cells, and having rowsHigh rows and colsWide columns.
)initWithFrame:(NSRect)frameRect mode:(int)aMode prototype:(NSCell *)aCell numberOfRows:(int)rowsHigh numberOfColumns:(int)colsWide	Initializes a new NSMatrix object with the given values with aMode as the selection mode, aCell as the prototype copied to make new cells, and having rowsHigh rows and colsWide columns.
SMatrixMode)mode	Returns the selection mode of the matrix.
id)setMode:(NSMatrixMode)aMode	Sets the selection mode of the matrix.
OOL)allowsEmptySelection	Returns whether it's possible to have no cells selected.
OOL)isSelectionByRect	Returns whether a user can drag a rectangular selection.
id)setAllowsEmptySelection:(BOOL)flag	Sets whether it's possible to have no cells selected.
id)setSelectionByRect:(BOOL)flag	Sets whether a user can drag a rectangular selection (the default is YES). If flag is NO, selection is on a row-by-row basis.
ass)cellClass	Returns the subclass of NSCell used to make new cells.

id)addColumnWithCells:(NSArray *)cellArray	Adds a new column of cells, using those contained in cellA
id)addRow	Adds a new row of cells below the last row.
id)addRowWithCells:(NSArray *)cellArray	Adds a new row of cells, using those contained in cellArra
SRect)cellFrameAtRow:(int)row column:(int)column	Returns the frame rectangle of the cell at row and column.
SSize)cellSize	Returns the width and height of cells in the matrix.
id)getNumberOfRows:(int *)rowCount columns:(int *)columnCount	Gets the number of rows and columns in the matrix.
id)insertColumn:(int)column	Inserts a new column of cells at column, creating as many matrix column columns wide.
id)insertColumn:(int)column withCells:(NSArray *)cellArray	Inserts a new row of cells at column, using those containe
id)insertRow:(int)row	Inserts a new row of cells at row, creating as many as nee row rows wide.
id)insertRow:(int)row withCells:(NSArray *)cellArray	Inserts a new row of cells at row, using those contained in
SSize)intercellSpacing	Returns the vertical and horizontal spacing between cells
SCell *)makeCellAtRow:(int)row column:(int)column	Creates a new cell at row, column in the matrix and return it.
id)putCell:(NSCell *)newCell atRow:(int)row column:(int)column	Replaces the cell at row and column with newCell.
id)removeColumn:(int)column	Removes the column at column, releasing the cells.
id)removeRow:(int)row	Removes the row at row, releasing the cells.
id)renewRows:(int)newRows columns:(int)newColumns	Changes the number of rows and columns in the receiver without freeing any cells.
id)setCellSize:(NSSize)aSize	Sets the width and height of all cells in the matrix.
id)setIntercellSpacing:(NSSize)aSize	Sets the vertical and horizontal spacing between cells.
id)sortUsingFunction:(int (*)(id element1, id element2, void *userData))comparator context:(void *)context	Sorts the receiver's cells in ascending order as defined by function comparator. context is passed as the function's
id)sortUsingSelector:(SEL)comparator	Sorts the receiver's cells in ascending order as defined by comparator.
OOL)getRow:(int *)row column:(int *)column forPoint:(NSPoint)aPoint	Gets the row and column position corresponding to aPoint Returns YES if aPoint is within the matrix NO otherwise.
OOL)getRow:(int *)row column:(int *)column ofCell:(NSCell *)aCell	Gets the row and column position of aCell. Returns YES if aCell is in the matrix NO otherwise.

id)deselectSelectedCell	Deselects the selected cell.
id)selectAll:(id)sender	Selects all the cells in the matrix.
id)selectCellAtRow:(int)row column:(int)column	Selects the cell at row and col.
OOL)selectCellWithTag:(int)anInt	Selects the cell with the tag anInt.
)selectedCell	Returns the most recently selected cell or nil if no cell has
SArray *)selectedCells	Returns an array containing the selected cells.
t)selectedColumn	Returns the column of the selected cell or 1 if no column h
t)selectedRow	Returns the row of the selected cell or 1 if no row has been
id)setSelectionFrom:(int)startPos to:(int)endPos anchor:(int)anchorPos highlight:(BOOL)flag	Selects the cells in the matrix from startPos to endPos, counting in row order from the upper left, as though anchorPos were the number of the last cell selected, an highlighting the cells according to flag.
)cellAtRow:(int)row column:(int)column	Returns the cell at row row and column col.
)cellWithTag:(int)anInt	Returns the cell having anInt as its tag.
SArray *)cells	Returns the matrix's array of cells.
SColor *)backgroundColor	Returns the color of the background between cells.
SColor *)cellBackgroundColor	Returns the color of the background within cells.
OOL)drawsBackground	Returns whether the receiver draws the background betwe
OOL)drawsCellBackground	Returns whether the receiver draws the background within
id)setBackgroundColor:(NSColor *)aColor	Sets the color of the background between cells to aColor.
id)setCellBackgroundColor:(NSColor *)aColor	Sets the color of the background within cells to aColor.
id)setDrawsBackground:(BOOL)flag	Sets whether the receiver draws the background between c
id)setDrawsCellBackground:(BOOL)flag	Sets whether the receiver draws the background within cel
id)selectText:(id)sender	Selects the text in the first or last editable cell.
)selectTextAtRow:(int)row column:(int)column	Selects the text of the cell at row, column in the matrix.
id)textDidBeginEditing:(NSNotification *)notification	Invoked when there's a change in the text after the receive status. Default behavior is pass to this message on to th method posts the NSControlTextDidBeginEditingNoti

	Invoked when text editing ends and then forwarded to the method posts the notification NSControlTextDidEndE the receiving object and, in the notification's dictionary the key NSFieldEditor) to the default notification cente
OOL)textShouldBeginEditing:(NSText *)textObject	Invoked to let the NSTextField respond to impending char forwarded to the text delegate.
OOL)textShouldEndEditing:(NSText *)textObject	Invoked to let the NSTextField respond to impending loss status and then forwarded to the text delegate.
)nextText	Returns the object to be selected when the user presses Ta text cell.
)previousText	Returns the object to be selected when the user presses Sh the first text cell.
id)setNextText:(id)anObject	Sets the object to be selected when the user presses Tab w cell.
id)setPreviousText:(id)anObject	Sets the object to be selected when user presses Shift-Tab text cell.
id)setDelegate:(id)anObject	Sets the delegate for messages from the field editor.
)delegate	Returns the delegate for messages from the field editor.
OOL)autosizesCells	Returns whether the matrix resizes its cells automatically.
id)setAutosizesCells:(BOOL)flag	Sets whether the matrix resizes its cells automatically.
id)setValidateSize:(BOOL)flag	Sets whether the cell size needs to be recalculated.
id)sizeToCells	Resizes the matrix to fit its cells exactly.
OOL)isAutoscroll	Returns whether the matrix automatically scrolls when dra
id)scrollCellToVisibleAtRow:(int)row column:(int)column	Scrolls the matrix so that the cell at row and column is visible.
id)setAutoscroll:(BOOL)flag	Sets whether the matrix automatically scrolls when dragge
id)setScrollable:(BOOL)flag	If flag is YES, makes all the cells scrollable.
id)drawCellAtRow:(int)row column:(int)column	Displays the cell at row and col.

EL)errorAction	Returns the action method for editing errors.
OOL)sendAction	Sends the selected cell's action, or the NSMatrix's action one.
id)sendAction:(SEL)aSelector to:(id)anObject forAllCells:(BOOL)flag	Sends aSelector to anObject, for all cells if flag is YES.
id)sendDoubleAction	Sends the action corresponding to a double-click.
id)setErrorAction:(SEL)aSelector	Sets the action method for editing errors to aSelector.
OOL)acceptsFirstMouse:(NSEvent *)theEvent	Returns NO only if receiver's mode is NSListModeMatrix
id)mouseDown:(NSEvent *)theEvent	Responds to a mouse-down event. A mouse-down event i editing mode. A double-click in any cell type except a double-click action of the NSMatrix (if there is one) in click action.
t)mouseDownFlags	Returns the event flags in effect at start of tracking.
OOL)performKeyEquivalent:(NSEvent *)theEvent	Simulates a mouse click in the appropriate cell.
id)resetCursorRects	Resets cursor rectangles so that the cursor becomes an I-b