MiscSliderCell

Inherits From: TextFieldCell : ActionCell : Cell : Object

Declared In: MiscSliderCell.h

Class Description

A subclass of TextFieldCell that adds a SliderCell to the mix. This cell functions the same way as a TextField/Slider combination connected together in Interface Builder. The main differences are that it allows you to now send an action method to an outside object when the value changes, it is much easier to get a consistent layout, there are fewer connections to be made, and it can be put in a Matrix easily (the same as any other cell). A **MiscStringArray** or a **StringList** (found as a MiniExample) can be attached to provide text strings that get shown instead of numbers.

A valid string list object responds to:

-(char *) **stringAt:**(int)item;

-(unsigned int) **count**;

The value sent to **stringAt:** will be in the range 0 - **count**. When the **stringList** instance variable is set, the ranges as set in IB (or wherever) are ignored and the string list defines the range of its values. Of course, **stringAt:** can create its return value any way it likes.

It also implements expandable boundaries. This allows you to set up the slider to have a reasonable range when it first appears, but also allows the user to type in values out of the range. The slider will then adjust to the new range automatically.

The cell can be configured to either send a single action message to it's target when the slider is released, or it can still send continuous messages. This allows you to easily skip dealing with the intermediate values, but still respond quickly to any change.

Instance Variables

id	idSlider;
id	stringList;
int	layout;
float	split;
double	actualValue;
double	limitMinValue;
double	limitMaxValue;

```
double    limitMinBound;
double    limitMaxBound;
BOOL      fExpandLow;
BOOL      fExpandHigh;
BOOL      fInt;
BOOL      fLoop;
BOOL      fFeedback;
BOOL      fValidRect;
NXRect    rectFrame;
```

idSlider	The SliderCell.
stringList	An object to provide strings in place of values.
layout	The relative positioning of the sub-cells.
split	The size of the Text Cell relative to the size of the MiscSliderCell.
actualValue	The numeric value since the display may be a string.
limitMinValue	The absolute lower limit of allowable values.
limitMaxValue	The absolute upper limit of allowable values.
limitMinBound	The adjustable lower limit for stopping the value when using the slider.
limitMaxBound	The adjustable upper limit for stopping the value when using the slider.
fExpandLow	Declares that limitMinBound is actually being used.

fExpandHigh	Declares that limitMaxBound is actually being used.
fInt	YES means the value it truncated to integral values.
fLoop	Used to keep things sane in the setXXXValue: methods. (Stops endless looping).
fFeedback	Used when the slider is being moved to keep from looping.
fValidRect	Used by the drawing methods to keep track of the width adjustment.
rectFrame	Used by the drawing methods to keep track of the width adjustment.

Method Types

Initializing	± initTextCell: ± copyFromZone:
Changing a MiscSliderCell	± setEditable:
Manipulating a MiscSliderCell	± setMinValue: ± setMaxValue: ± setMinBoundary: ± setMaxBoundary:

	± setExpandMin: ± setExpandMax: ± setPosition: ± setSplit: ± setStringList: ± setIntegerOnly:
Querying values	± minValue ± maxValue ± minBoundary ± maxBoundary ± expandMin ± expandMax ± position ± split ± stringList ± integerOnly
Archiving	± read: ± write: ± awake ± awakeFromNib

Instance Methods

awake
± awake

Sends a message so the value will be properly displayed. Returns **self**.

See also: **-awakeFromNib**

awakeFromNib
± **awakeFromNib**

Sends self an awake message. Returns **self**.

See also: ± **awake**

copyFromZone:
± **copyFromZone:**(NXZone *)*zone*

Copies the MiscSliderCell and also copies the ButtonCells inside. Returns the new object.

expandMax
±(BOOL) **expandMax**

Returns YES if **maxBoundary** is in effect stopping the arrow buttons from reaching **maxValue**.

See also: **± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± setExpandMax, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

expandMin

±(BOOL) **expandMin**

Returns YES if **minBoundary** is in effect stopping the arrow buttons from reaching **minValue**.

See also: **± expandMax, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± setExpandMax, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

initTextCell

± **initTextCell:**(const char *)*sz*

Initializes the receiver to contain the string *sz*. The hard limits are set to 0 and 100 and the soft limits are not active. the step size is set to 1 and the Alternate step size is set to 10. The Cell has the Bezeled style, is Editable and is not Continuous. Returns **self**.

integerOnly

±(BOOL) **integerOnly**

Returns YES if the MiscSliderCell is displaying only integral values.

See also: **± setIntegerOnly:**

maxBoundary

±(double) **maxBoundary**

Returns the current limit that the value will stop at when the up arrow button is being used to change the value. This limit can be changed by entering a larger value using the keyboard. This limit will be set the highest value entered that is still within the bounds of **maxValue**. It cannot be reduced except programmatically.

See also: **± expandMax, ± expandMin, ± maxValue, ± minBoundary, ± minValue, ± setExpandMax, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

maxValue

±(double) **maxValue**

Returns the highest value this field is allowed to reach. This limit cannot be exceeded in any way. **maxBoundary** can match this value and then have no effect.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± minBoundary, ± minValue, ± setExpandMax, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

minBoundary

±(double) **minBoundary**

Returns the current limit that the value will stop at when the down arrow button is being used to change the

value. This limit can be changed by entering a smaller value using the keyboard. This limit will be set the lowest value entered that is still within the bounds of **minValue**. It cannot be increased except programmatically.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minValue, ± setExpandMax, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

minValue

±(double) **minValue**

Returns the lowest value this field is allowed to reach. This value cannot be exceeded in any way. **minBoundary** can match this value and then have no effect.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± setExpandMax, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

position

±(int) **position**

Returns the position of the TextFieldCell relative to the SliderCell. See **setPosition:** for the values.

See also: **± setPosition:, ± setSplit:, ± split**

read:

± **read:**(NXTypedStream *)*stream*

Reads a MiscSliderCell from *stream*. This also reads in the ButtonCells used as subcells. Returns **self**.

See also: ± **write:**

setEditable:
± **setEditable:**(BOOL)*flag*

If *flag* is YES, sets the TextFieldCell to normal editable mode. If *flag* is NO, sets non-editable, but also sends **setSelectable:YES** so the buttons will still work. Returns **self**.

setExpandMax:
± **setExpandMax:**(BOOL)*flag*

If *flag* is YES then **maxBoundary** has an effect on the upper limit of the field value when it's adjusted with the arrow buttons. Returns **self**.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± €setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

setExpandMin:
± **setExpandMin:**(BOOL)*flag*

If *flag* is YES then **minBoundary** has an effect on the lower limit of the field value when it's adjusted with the arrow buttons. Returns **self**.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± €setExpandMax, ± setMaxBoundary, ± setMaxValue, ± setMinBoundary, ± setMinValue**

setIntegerOnly

± **setIntegerOnly:**(BOOL)*flag*

If *flag* is YES only the integral part of the value will be used. The fractional part of the value is truncated when the value is set, **doubleValue** will return the same value as **intValue** when this flag is set to YES. Returns **self**.

See also: **± integerOnly**

setMaxBoundary

± **setMaxBoundary:**(double)*value*

Sets the highest value the field will allow when using the arrow buttons to adjust the value. This value can be exceeded and altered by typing a value greater than this limit. This value will be adjusted to match the entered amount with an additional limitation that it will stop at **maxValue**. Returns **self**.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± €setExpandMax, ± setExpandMin, ± setMaxValue, ± setMinBoundary, ± setMinValue**

setMaxValue

± **setMaxValue:**(double)*value*

Sets the highest value the field will allow. There is no way to exceed this limit from the interface. **maxBoundary** will stop expanding when it gets to this value. Returns **self**.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± €setExpandMax, ± setExpandMin, ± setMaxBoundary, ± setMinBoundary, ± setMinValue**

setMinBoundary

± **setMinBoundary:**(double)*value*

Sets the lowest value the field will allow when using the arrow buttons to adjust the value. This value can be exceeded and altered by typing a value lower than this limit. This value will be adjusted to match the entered amount with an additional limitation that it will stop at **minValue**. Returns **self**.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ± €setExpandMax, ± setExpandMin, ± setMaxBoundary, ± setMaxValue, ± setMinValue**

setMinValue

± **setMinValue:**(double)*value*

Sets the lowest value the field will allow. There is no way to exceed this limit from the interface. **minBoundary** will stop expanding when it gets to this value. Returns **self**.

See also: **± expandMax, ± expandMin, ± maxBoundary, ± maxValue, ± minBoundary, ± minValue, ±**

setPosition

± **setPosition:**(int)*where*

Sets the position of the TextFieldCell relative to the SliderCell. Constants are defined in MiscSliderCell.h and can be selected from MSC_ABOVELEFT, MSC_ABOVECENTER, MSC_ABOVERIGHT, MSC_LEFT, MSC_RIGHT, MSC_BELOWLEFT, MSC_BELOWCENTER and MSC_BELOWRIGHT. See **setSplit:** for further control of the display. Returns **self**.

See also: **± position, ± setSplit:, ± split**

setSplit

± **setSplit:**(int)*percent*

Sets the width of the TextFieldCell as a percentage of the width of the entire MiscSliderCell. When the position is set to MSC_LEFT or MSC_RIGHT the SliderCell is adjusted to take up the remaining horizontal space. In all the other position settings the SliderCell takes the full width of the MiscSliderCell and the TextFieldCell is still sized as a percentage of the total length and positioned appropriately. Valid values are 0-100 but the minimum size of both the TextFieldCell and the SliderCell are set. Returns **self**.

See also: **± position, ± setPosition:, ± split**

setStringList:

± **setStringList:***anObject*

Sets the object to be asked for display strings. *anObject* should respond to **stringAt:** and **count** messages. When a string list is set the range limits are all ignored and the field is set to non-editable. The value of the field will only fit within a range defined by *anObject's* **count** method. Returns **self**.

anObject's **stringAt:** method should take an **int** as its argument and return a **char *** with the correct string to be displayed for the given value. The value of the argument will range from 0 to **count** - 1. This arrangement has been created to allow the MiscValueField to be connected to a StringList (found in the MiniExamples) object in Interface Builder so lists may be created, displayed and dealt with with no additional code.

See also: ± **stringList**

split

±(int) **split**

Returns the relative width of the TextFieldCell in relation to the size of the MiscSliderCell.

See also: **± position, ± setPosition:, ± setSplit:**

stringList

± **stringList**

Returns the object that is taking the task of providing the display strings.

See also: ± **setStringList:**

write:

± **write:**(NXTypedStream *)*stream*

Writes the MiscSliderCell and its ButtonCells to *stream*. The images in the ButtonCells only get archived once so redundancy is kept down. Returns **self**.

See also: ± **read:**