MiscClipTextFieldCell

Inherits From: TextFieldCell : ActionCell : Cell : Object

Declared In: MiscClipTextFieldCell.h

Class Description

MiscClipTextFieldCell objects can be used in controls to display long string values that normally would be truncated because of short of space. Using MiscClipTextFieldCell you can easily implement a text field similar to what Workspace uses to display long pathnames in the "Attributes Inspector" (check out MiscPathName or MiscClipTextField which actually do this).

MiscClipTextFieldCells can be configured several ways. You can specify whether clipping should happen on the left or the right of the string value (**setClipOnRight:**), what string to use to substitude clipped (non-visible) part of the string (**setClipperString** :) and you can specify characters (delimiters) only by which clipping can happen (**setClipDelimiters:**). For example, when you want to display long pathnames you can set MiscClipTextFieldCells to clip on the left, use three dots ("...") as clipper string and to allow clipping to happen only by the slash ('/') characters.

NOTE: This was not meant to be a "Your code/interface/palette/documentation should look like this" kind of project. I only made it because I needed this "clipping" feature in a little project of mine - and anyway I'm not a professional NS guru (yet). This class was mostly inspired by Don Yacktman's `MiscClipTextField' class (MiscKit 1.5.0. "Temp" area). Some other code was lifted from Mike Ferris's `MORegexPalette'.

Known Bugs

I know of one bug but I don't know why it happens. When MiscClipTextFieldCells are used in Matrix and a Button's action message is set to **resetStringValue:** it will crash (or at least mess up) IB or your application. If you use a TextField instead of the button it doesn't do any harm. Check out Mess.nib in the source folder under Test. [Ed. Note: I couldn't recreate this on my 3.3 machine. -don]

Anyway it dosen't seem to have dangerous bugs in it, and should work as advertised :-)

Instance Variables

id **delegate**;
id **fullString**;
id **clipper**;
id **delimiters**;
BOOL **clipOnRight**;
BOOL **clipEnabled**;

delegate	The object that receives notification messages.
fullString	The non-clipped string value.
clipper	Substitutes for non-visible portion of the cell's string value.
delimiters	A string of characters by which clipping can occur.
clipOnRight	True if clipping happens on the right.
clipEnabled	True if clipping is enabled.

Method Types

Initializing a new MiscClipFieldCell
- init
- initTextCell:

Copying a MiscClipTextFieldCell
- copyFromZone:

Setting the MiscClipTextFieldCell's value
- setStringValue:

Setting clipping attributes
- resetStringValue:
- setClipDelimiters:

- setClipEnabled:
- setClipOnRight:
- setClipperString:

Querying MiscClipTextFieldCell attributes

- clipper
- delegate
- delimiters
- doesClipOnRight
- fullStringValue
- isClipEnabled
- isWrapped

Assigning a delegate

- setDelegate:
- delegate

Archiving

- read:
- write:

Instance Methods

clipper

- **clipper**

Returns a MiscString object that stores the string which is used to substitute non-visible parts of the reciever's string value.

See also: - **delimiters**,

copyFromZone:

- **copyFromZone:**(NXZone *)*zone*

Returns a new instance that is an exact copy of the reciever. Memory for the new instance is allocated fom *zone*.

See also:

delegate

- **delegate**

Returns the delegate of the reciever, the object which gets notified of clipping (see *Methods Implemented by the Delegate*)

See also: - **setDelegate:**,

delimiters

- **delimiters**

Returns a MiscString object that stores the characters by which clipping can happen. Returns **nil** if

delimiter characters are not specified.

See also: - **setClipDelimiters:**

doesClipOnRight

- (BOOL) **doesClipOnRight**

Returns YES if last time a string value was set by the reciever clipping happened on the right and NO otherwise

See also: - **setClipOnRight:**

fullStringValue

- (const char*) **fullStringValue**

Returns the non-clipped string value (ie. the full string value).

See also: - **setStringValue:**, - **takeStringValueFrom:**

getInspectorClassName

- (const char *)**getInspectorClassName**

Returns the string "MiscClipTextFieldInspector" (Used for implementing MiscClipText.palette)

See also:

init

- **init**

Initailizes and returns the reciever, a new instance of MiscClipTextFieldCell. Defaults are set as described in **initTextCell:** below

See also: - **initTextCell:**,

initTextCell:

- **initTextCell:**(const char*)*aString*

Initializes and returns the receiver, a new instance of MiscClipTextFieldCell, with *aString* as its text. The default text gray is NX_BLACK, and the default background gray is NX_LTGRAY. Its font is set to the user's system font, and the font size is 12.0 point.

The default clipper string (the string that is displayed in place of the clipped part of the string value) is set to three dots ("..."), clipping is enabled and clipping is set to happen on the right of the string. Delimiters are not set thus clipping occurs at the last displayable character.

This method is the designated initializer for MiscTextFieldCell. Note that MiscTextFieldCell doesn't override Cell's **initIconCell:** designated initializer; your code shouldn't use that method to initialize an instance of MiscTextFieldCell.

See also: - **init**

isClipEnabled

- (BOOL) **isClipEnabled**

Returns YES if clipping is enabled and NO otherwise.

See also: - **setClipEnabled:**

isWrapped

- (BOOL) **isWrapped**

Returns YES if text in the cell is wrapped to word bounderies and returns NO otherwise (ie: text is truncated in the cell)

See also: - **setWrap:** (Cell)

read:

- **read:**(NXTypedStream *)*stream*

Reads the MiscTextFieldCell from the typed stream *stream*. Returns **self**.

See also: - **read:**

resetStringValue:

- **resetStringValue:***sender*

Resets the string value displayed in the MiscClipTextFieldCell. (This method can be used e.g. when the control in which the cell is displayed is resized. See MiscClipTextField as an example.)

See also: - **setStringValue:**

setClipDelimiters:

- **setClipDelimiters:**(const char*)*delimiters*

Sets *delimiters* as the character by which clipping has to happen. If *delimiters* is NULL clipping will happen by the last displayable character of the MiscClipTextFieldCell's string value. Returns **self**.

See also: - **delimiters**,

setClipEnabled:

- **setClipEnabled:**(BOOL)*flag*

Sets whether the next message to the reciever with selector **setSrtingValue:** should clip the given string value or not.

If *flag* is YES, next time the MiscTextFieldCell's string value is set and this value is too long to be displayed in whole the string value will be clipped so that non-visible portion of a given string will be replaced by a user defined string (*clipper*).

If *flag* is NO the string value will be displayed as TextField would do. Return **self.**

See also: - **isClipEnabled**

setClipOnRight:

- **setClipOnRight:**(BOOL)*flag*

If *flag* is YES clipping will happen on the right next time it is needed (e.g: a message to the reciever with selector **setStringValue:**) . If flag is NO cnext clipping will happen on left. Returns **self**.

See also: - **doesClipOnRight**,

setClipperString:

- **setClipperString:**(const char*)*aString*

Sets *aString* as the string that is displayed when substituting non-visible (clipped) portion of the cell's string value. Returns **self**.

See also: - **clipper**,

setDelegate:

- **setDelegate:***anObject*

Makes *anObject* the MiscClipTextFieldCell's delegate. A MiscClipTextFieldCell's delegate is given a chance to clip the string value before MiscClipTextFieldCell would do it. If the delegate clips the string value short enogh to be displayed, MiscClipTextFieldCell will not apply its own clipping. Returns **self**.

See also: - **delegate**,

setStringValue:

- **setStringValue:**(const char *)*aString*

If clipping is enabled (**setClipEnabled**) and *aString* is too long to be displayed in whole this method clips the string by replacing the clipped part with a user defined string (*clipper*) and displayes the clipped string value.

If delimiters are set (**setClipDelimiters:**) clipping will happen at the last delimiter charcater which yet can be displayed. Iff delimiters are not set, clipping will happen at the last displayable character.

See also: - **fullStringValue**

takeStringValueFrom:

- **takeStringValueFrom:***sender*

Method description here.

See also: - **myReference**

write:

- **write:**(NXTypedStream *)*stream*

Writes the recieving MiscClipTextFieldCell to the typed stream *stream*. Returns **self**.

See also: - **read:**

Methods Implemented by the Delegate

stringWillBeClipped:

- **stringWillBeClipped:***theString*

Notifies the reciever that *theString* (a MiscString instance) will be clipped. The reciever can clip *theString* and if it gets short enogh to be displayed, MiscClipTextFieldCell will not apply its own clipping. (See MiscPathField as an example of the use of this method)

See also: