MiscTableCell

Inherits From:	NSCell : NSObject
Conforms To:	NSCoding (NSObject), NSCopying (NSObject), NSObject (NSObject)
Declared In:	MiscTableCell.h

Class Description

This is the cell class used by MiscTableScroll to display either text or an image. This class provides a tag and separate text and background colors for normal and selected states. Finally, it supports the concept of an *owner*, and inherited font and color values.

Owner and Inheritence of Colors and Font

This class implements the concept of an *owner*. An owner provides the default font and colors used by the cells that it owns. This makes it fast and easy to change the appearance of an entire table: set the font and colors of the owner, and all of the cells that it owns automatically inherit the new font and colors. However, you can also set font and color values for individual cells. Values that you explicitly set supercede values implicitly inherited from the owner. You can use this feature to emphasize special cells while still using inherited values for normal cells.

Another benefit of inherited values is reduced memory usage. Inherited values don't have to be stored in the individual cells. They just ask their owner for the current value whenever it is needed. This can lead to significant savings in large tables.

Owner drawing

This class also supports a specialized concept of delegated drawing. The MiscTableScroll class will draw cells that respond with YES to the `-ownerDraw` message. This enables the MiscTableScroll object to eliminate many redundant font and color setting operations, greatly improving drawing performance. See the description of owner drawing in MiscTableScroll for a more detailed discussion.

Extensible Memory Management

To minimize memory usage, storage for optional items is allocated on an as-needed basis. Memory is allocated only for the values that are actually set. If no optional items are set, then no extra memory at all is allocated.

The optional allocation scheme is extensible. Subclasses of MiscTableCell can extend this allocation scheme by appending their own dynamic variables to the end of the list used by this class. There is a strict ordering for the information stored in the variable length data. This ordering is enumerated by the *tc1_flags*. Subclasses should append their data after the last of the information stored by this class. Methods are provided to determine whether or not a piece of information is stored in the data field as well as for finding its address. Convenience methods for extracting a given piece of information complete the interface.

The methods which return the address of the piece of data in *tc1_data* are dynamic in nature. For instance, text color is stored prior to background color in *tc1_data* so if both text color and background color have been set then the address of the storage for the background color will follow the text color. However, if text color is never set ± that is it is inherited from the owner ± and the background color has been set then the address of the background color will be at a different location. If text color gets set later then the storage for background color is shifted up to make room for the text color. Ergo, the strict ordering is maintained.

For example, code to locate the address of the selected background color is stored in the cell would look like this:

```
- (unsigned int)tc1SelectBackgroundColorPos
   {
   unsigned int pos = [self tc1SelectedTextColorPos];
   if (tc1_flags & MISC_TC1_SELF_TEXT_COLOR_H)
```

```
      pos += [self tc1SelectedTextColorLen];
    return pos;
    }
```

Since the selected background color is stored following the selected text color it first determines the location of the selected text color then adds the length of the selected text color to that address if the selected text color is in fact being stored locally. The actual shifting of the dynamic memory is done in the `-setUseOwner`... methods. For instance:

```
- (void)setUseOwnerSelectedBackgroundColor:(BOOL)flag
    {
    if ([self useOwnerSelectedBackgroundColor] != flag)
        {
        unsigned int const pos = [self tc1SelectedBackgroundColorPos];
        unsigned int const len = [self tc1SelectedBackgroundColorLen];
        if (flag)
            {
            [[self selectedBackgroundColor] release];
            [self tc1DeleteDataPos:pos len:len];
            tc1_flags &= ~(MISC_TC1_SELF_BACKGROUND_COLOR_H);
            }
        else // (!flag)
            {
            NSColor* color = [[[self class] defaultSelectedBackgroundColor] retain];
            [self tc1InsertData:&color pos:pos len:len];
            tc1_flags |= MISC_TC1_SELF_BACKGROUND_COLOR_H;
            }
        }
    }
```

Finally, the `-setSelectedBackgroundColor:` method makes room for the data and then stores it:

```
- (void)setSelectedBackgroundColor:(NSColor*)c
    {
    NSColor** p;
    [self setUseOwnerSelectedBackgroundColor:NO];
    p = [self tc1SelectedBackgroundColorPtr];
    [*p autorelease];
    *p = [c retain];
    }
```

A subclass adding more data to the end of *tc1_data* would provide similar methods. Its counterpart to `-tc1SelectedBackgroundColorPos` would first call `-tc1SelectedBackgroundColorPos` to determine its address and then add the correct offset returned by `-tc1SelectedBackgroundColorLen` as appropriate.

Instance Variables

```
id owner;
int tag;
unsigned tc1_flags;
void* tc1_data;
```

owner	The owner of this cell; queried for font and color information.
tag	A general purpose slot for your use.
tc1_flags	Bit flags describing the contents of *tc1_data*.
tc1_data	Variable length data which is allocated only as needed for storage of color values.

Method Types

Initializing, freeing, and copying	± initImageCell: ± initTextCell: ± dealloc ± copyWithZone:

Drawing	± drawInteriorWithFrame:inView:
	± drawWithFrame:inView:
	± isOpaque
	± bgColor
	± fgColor
	± setSelected:
	± isSelected
	± ownerDraw
	± setOwnerDraw:
Tag manipulation	± setTag:
	± tag
Font manipulation	± font
	± setFont:
Setting and querying colors	+ defaultBackgroundColor
	+ defaultFont
	+ defaultSelectedBackgroundColor
	+ defaultSelectedTextColor
	+ defaultTextColor
	± backgroundColor

	± selectedBackgroundColor
	± selectedTextColor
	± setBackgroundColor:
	± setSelectedBackgroundColor:
	± setSelectedTextColor:
	± setTextColor:
	± textColor
Setting and querying owner	± owner
	± setOwner:
Colors and owner inheritence	± setOwnerBackgroundColor:
	± setOwnerFont:
	± setOwnerSelectedBackgroundColor:
	± setOwnerSelectedTextColor:
	± setOwnerTextColor:
	± setUseOwnerBackgroundColor:
	± setUseOwnerFont:
	± setUseOwnerSelectedBackgroundColor:
	± setUseOwnerSelectedTextColor:
	± setUseOwnerTextColor:

	± useOwnerBackgroundColor ± useOwnerFont ± useOwnerSelectedBackgroundColor ± useOwnerSelectedTextColor ± useOwnerTextColor
Archiving	± initWithCoder: ± encodeWithCoder:
Extensible conditional allocations	± tc1BackgroundColorLen ± tc1BackgroundColorPos ± tc1BackgroundColorPtr ± tc1DataSize ± tc1DeleteDataPos:len: ± tc1DestroyData ± tc1Flags ± tc1SelectedBackgroundColorLen ± tc1SelectedBackgroundColorPos ± tc1SelectedBackgroundColorPtr ± tc1SelectedTextColorLen ± tc1SelectedTextColorPos

± tc1SelectedTextColorPtr
± tc1InsertData:pos:len:
± tc1TextColorLen
± tc1TextColorPos
± tc1TextColorPtr

Class Methods

defaultBackgroundColor

+ (NSColor*)**defaultBackgroundColor**

Returns `[NSColor lightGrayColor]` as the default background color. This is the color which is returned by `-backgroundColor` if a custom color has not been set for this cell and no owner has been set or the owner does not respond to the `-backgroundColor` message. This is also the color used upon receipt of a `-setUseOwnerBackgroundColor:NO` message. Subclasses should override this method if this color is inappropriate.

See also: **± backgroundColor, + defaultSelectedBackgroundColor, + defaultSelectedTextColor, + €defaultTextColor, ± setBackgroundColor:, ± setUseOwnerBackgroundColor:**

defaultFont

+ (NSFont*)**defaultFont**

Returns the user-font at point-size 12.0 as the default font. This is the font which is used when initializing the cell as a text cell with **±**`initTextCell:`. Subclasses should override this method if this value is inappropriate.

See also: ± font, ± initTextCell:, ± setFont:

defaultSelectedBackgroundColor

+ (NSColor*)**defaultSelectedBackgroundColor**

Returns `[NSColor whiteColor]` as the default selected background color. This is the color which is returned by `-selectedBackgroundColor` if a custom color has not been set for this cell and no owner has been set or the owner does not respond to the `-selectedBackgroundColor` message. This is also the color used upon receipt of a `-setUseOwnerSelectedBackgroundColor:NO` message. Subclasses should override this method if this color is inappropriate.

See also: + defaultBackgroundColor, + defaultSelectedTextColor, + defaultTextColor, ± €selectedBackgroundColor, ± setSelectedBackgroundColor:, ± €setUseOwnerSelectedBackgroundColor:

defaultSelectedTextColor
+ (NSColor*)**defaultSelectedTextColor**

Returns `[NSColor blackColor]` as the default selected text color. This is the color which is returned by `-selectedTextColor` if a custom color has not been set for this cell and no owner has been set or the owner does not respond to the `-selectedTextColor` message. This is also the color used upon receipt of a `-setUseOwnerSelectedTextColor:NO` message. Subclasses should override this method if this color is inappropriate.

See also: **+ defaultBackgroundColor, + defaultSelectedBackgroundColor, + defaultTextColor, ±selectedTextColor, ± setSelectedBackgroundColor:, ± setUseOwnerSelectedBackgroundColor:**

defaultTextColor
+ (NSColor*)**defaultTextColor**

Returns `[NSColor blackColor]` as the default text color. This is the color which is returned by `-textColor` if a custom color has not been set for this cell and no owner has been set or the owner does not respond to the `-textColor` message. This is also the color used upon receipt of a `-setUseOwnerTextColor:NO` message. Subclasses should override this method if this color is inappropriate.

See also: **+ defaultBackgroundColor, +€defaultSelectedBackgroundColor, + defaultSelectedTextColor, ± €setSelectedBackgroundColor:, ±€setUseOwnerSelectedBackgroundColor:, ± textColor**

Instance Methods

backgroundColor

- (NSColor*)**backgroundColor**

Returns the color that is used to draw the background of the cell in its normal (unselected) state. This is the color which has been set with **±**`setBackgroundColor:` if it was ever called. If not, and the owner has been set and responds to **±**`backgroundColor` then it is queried and that color is returned. If both of the above fail then the value from **+**`defaultBackgroundColor` is returned.

See also: **+ defaultBackgroundColor, ± selectedBackgroundColor, ± selectedTextColor, ± setBackgroundColor:, ±€textColor**

bgColor

- (NSColor*)**bgColor**

This method is used by cells that draw themselves rather than letting the owner draw them. It returns the color that is used to fill the background of the cell during drawing. If the cell `-isSelected`, then this method returns the value from `-selectedBackgroundColor`, otherwise it returns the value from `-backgroundColor`. Subclasses should override this method if they have different criteria for determining the cell's background color during drawing.

See also: **± backgroundColor, ± fgColor, ± selectedBackgroundColor, ± isSelected**

copyWithZone:

- (id)**copyWithZone:**(NSZone*)*zone*

Allocates, initializes, and returns a copy of the receiving cell. The copy is allocated from *zone* and is assigned the same contents as the receiver.

See also: **± copyWithZone:** (NSCopying)**, ± initImageCell:, ± initTextCell:**

dealloc

- (void)**dealloc**

Frees the memory used by the cell.

See also: **± copyWithZone:, ± initImageCell:, ± initTextCell:**

drawInteriorWithFrame:inView:

- (void)**drawInteriorWithFrame:**(NSRect)*cellFrame* **inView:**(NSView*)*controlView*

Draws the inside of the cell, but not the border, in *cellFrame* within *controlView*. *cellFrame* should be the same rectangle passed to `-drawWithFrame:inView:`. The PostScript focus must be locked on *controlView* when this message is sent.

See also: **- drawInteriorWithFrame:inView:** (NSCell), **- drawWithFrame:inView:, - lockFocus** (NSView)

drawWithFrame:inView:

- (void)**drawWithFrame:**(NSRect)*cellFrame* **inView:**(NSView*)*controlView*

Displays the cell in *cellFrame* within *controlView*. The PostScript focus must be locked on *controlView* when this message is sent. Draws the border of the cell, then invokes `±drawInteriorWithFrame:inView:`.

See also: - **drawInteriorWithFrame:inView:**, ± **drawWithFrame:inView:** (NSCell), - **lockFocus** (NSView)

encodeWithCoder:

- (void)**encodeWithCoder:**(NSCoder€*)*encoder*

Encodes the receiver using *encoder*.

fgColor

- (NSColor*)**fgColor**

This method is used by cells that draw themselves rather than letting the owner draw them. It returns the color that is used for the text in the cell during drawing. If the cell `-isSelected`, then this method returns the value from `-selectedTextColor`, otherwise it returns the value from `-textColor`. Subclasses should override this method if they have different criteria for determining the cell's text color during drawing.

See also: ± bgColor, ± selectedTextColor, ± isSelected, ± textColor

font

- (NSFont*)**font**

Returns the Font used to display text in the cell. Returns **nil** if the receiver isn't a text-type cell.

See also: **+ defaultFont, ± setFont:**

initImageCell:

- (id)**initImageCell:**(NSImage*)*image*

Prepare a newly allocated cell that will display *image*. Returns **self**.

initTextCell:

- (id)**initTextCell:**(NSString*)*text*

Prepare a newly allocated cell that will display *text*. The font and colors will use inherited or default values until explicitly set. Returns **self**.

initWithCoder:

- (id)**initWithCoder:**(NSCoder*)*decoder*

Initializes a newly allocated instance from data in *decoder*. Returns **self**.

isOpaque

- (BOOL)**isOpaque**

Returns NO if the cell uses the owner's background color, or YES if it does not.

isSelected

- (BOOL)**isSelected**

Returns the cell's MISC_TC1_IS_SELECTED flag.

See also: **± bgColor**, **± fgColor, ± setSelected:**

owner

- (id)**owner**

Returns the *owner* of the cell, or **nil** if there is none.

ownerDraw;¬ownerDraw

- (BOOL)**ownerDraw**

Returns YES if the owner should perform the actual drawing for this cell, otherwise NO.

selectedBackgroundColor

- (NSColor*)**selectedBackgroundColor**

Returns the color that is used to draw the background of the cell in its selected state. This is the color which has been set with **±**`setSelectedBackgroundColor:` if it was ever called. If not, and the owner has been set and responds to **±**`selectedBackgroundColor` then it is queried and that color is returned. If both of the above fail then the value from `+defaultSelectedBackgroundColor` is returned.

See also: **± backgroundColor, + defaultSelectedBackgroundColor, ± selectedTextColor, ± €setSelectedBackgroundColor:, ± textColor**

selectedTextColor

- (NSColor*)**selectedTextColor**

Returns the color that is used to draw the text in the cell in its selected state. This is the color which has been set with **±**`setSelectedTextColor:` if it was ever called. If not, and the owner has been set and responds to **±**`selectedTextColor` then it is queried and that color is returned. If both of the above fail then the value from `+defaultSelectedTextColor` is returned.

See also: ± backgroundColor, + defaultSelectedTextColor, ± selectedBackgroundColor, ± setSelectedTextColor:, ± textColor

setBackgroundColor:

- (void)**setBackgroundColor:**(NSColor*)*c*

Sets the color that the cell will use to fill its background. This color will override any inherited value.

setFont:

- (void)**setFont:**(NSFont*)*font*

Sets the font of a text cell to *font*. This will override any inherited value.

setOwner:

- (void)**setOwner:**(id)*obj*

Sets the *owner* of the cell to *obj*. The cell will inherit font and color values from *obj*.

setOwnerBackgroundColor:

- (void)**setOwnerBackgroundColor:**(NSColor*)*c*

Informs the cell that the owner's *backgroundColor* value has changed to ***c.*** The current implementation does nothing, since inherited owner values are not stored in the cells. However, the MiscTableScroll class tests for, and sends this message in preference to the `-setBackgroundColor:` message when the table distributes colors to the cells, so that cells can distinguish between cell-specific color assignments and global color assignments.

setOwnerDraw;¬setOwnerDraw:

- (void)**setOwnerDraw:**(BOOL)*flag*

Sets a flag indicating whether or not the owner should perform the drawing for this object. The default value for MiscTableCell objects is YES.

setOwnerFont:

- (void)**setOwnerFont:**(NSFont*)*font*

Informs the cell that the owner's *font* value has changed to *font*. If the cell is using the font value inherited from its *owner*, it will change its font in response to this message. Otherwise, it will ignore the message. The MiscTableScroll class tests for, and sends this message in preference to the `-setFont:` message when the table distributes fonts to the cells, so that cells can distinguish between cell-specific font assignments and global font assignments.

setOwnerSelectedBackgroundColor:

- (void)**setOwnerSelectedBackgroundColor:**(NSColor*)*c*

Informs the cell that the owner's *selectedBackgroundColor* value has changed to ***c.*** The current implementation does nothing, since inherited owner values are not stored in the cells. However, the MiscTableScroll class tests for, and sends this message in preference to the `-setSelectedBackgroundColor:` message when the table distributes colors to the cells, so that cells can distinguish between cell-specific color assignments and global color assignments.

setOwnerSelectedTextColor:

- (void)**setOwnerSelectedTextColor:**(NSColor*)*c*

Informs the cell that the owner's *selectedTextColor* value has changed to *c.* The current implementation does nothing, since inherited owner values are not stored in the cells. However, the MiscTableScroll class tests for, and sends this message in preference to the `-setSelectedTextColor:` message when the table distributes colors to the cells, so that cells can distinguish between cell-specific color assignments and global color assignments.

setOwnerTextColor:

- (void)**setOwnerTextColor:**(NSColor*)*c*

Informs the cell that the owner's *textColor* value has changed to *c.* The current implementation does nothing, since inherited owner values are not stored in the cells. However, the MiscTableScroll class tests for, and sends this message in preference to the `-setTextColor:` message when the table distributes colors to the cells, so that cells can distinguish between cell-specific color assignments and global color assignments.

setSelected:

- (void)**setSelected:**(BOOL)*flag*

Sets the cell's MISC_TC1_IS_SELECTED flag.

See also: **± bgColor**, **± fgColor, ± isSelected**

setSelectedBackgroundColor:
- (void)**setSelectedBackgroundColor:**(NSColor*)*c*

Sets the color that will be used to fill the background of a text cell when the cell is selected. This will override any inherited value.

setSelectedTextColor:
- (void)**setSelectedTextColor:**(NSColor*)*c*

Sets the color that will be used to draw the text of a text cell. This will override any inherited value.

setTag:
- (void)**setTag:**(int)*x*

Sets the cell's *tag* value to *x*.

See also: ± tag

setTextColor:

- (void)**setTextColor:**(NSColor*)*c*

Causes the cell to render its text with the color, *c*. This overrides any *textColor* value inherited from the *owner*.

setUseOwnerBackgroundColor:

- (void)**setUseOwnerBackgroundColor:**(BOOL)*flag*

If *flag* is YES, the cell will discard any *backgroundColor* value that it is currently storing, and begin using the backgroundColor value provided by the *owner*. If the owner has not been set, or the owner does not respond to the `-backgroundColor` message, then the value returned by `+defaultBackgroundColor` will be used. If *flag* is NO, the cell will make room to store a *backgroundColor* value, and initialize it to the value returned by the `+defaultBackgroundColor` method.

setUseOwnerFont:

- (void)**setUseOwnerFont:**(BOOL)*flag*

If *flag* is YES, the cell will use the *font* value inherited from its *owner*. If *flag* is NO, the cell will continue using its current font, or fonts set via the `-setFont:` message rather than the font inherited from the owner. This method also updates the `MISC_TC1_SELF_FONT` bit of the *tc1_flags* bit mask.

setUseOwnerSelectedBackgroundColor:

- (void)**setUseOwnerSelectedBackgroundColor:**(BOOL)*flag*

If *flag* is YES, the cell will discard any *selectedBackgroundColor* value that it is currently storing, and begin using the *selectedBackgroundColor* value provided by the *owner*. If the owner has not been set, or the owner does not respond to the `-selectedBackgroundColor` message, then the value returned by `+defaultSelectedBackgroundColor` will be used. If *flag* is NO, the cell will make room to store a *selectedBackgroundColor* value, and initialize it to the value returned by the `+defaultSelectedBackgroundColor` method.

setUseOwnerSelectedTextColor:

- (void)**setUseOwnerSelectedTextColor:**(BOOL)*flag*

If *flag* is YES, the cell will discard any *selectedTextColor* value that it is currently storing, and begin using the *selectedTextColor* value provided by the *owner*. If the owner has not been set, or the owner does not respond to the `-selectedTextColor` message, then the value returned by `+defaultSelectedTextColor` will be used. If *flag* is NO, the cell will make room to store a *selectedTextColor* value, and initialize it to the value returned by the `+defaultSelectedTextColor` method.

setUseOwnerTextColor:

- (void)**setUseOwnerTextColor:**(BOOL)*flag*

If *flag* is YES, the cell will discard any *textColor* value that it is currently storing, and begin using the *textColor* value provided by the *owner*. If the owner has not been set, or the owner does not respond to the `-textColor` message, then the value returned by `+defaultTextColor` will be used. If *flag* is NO, the cell will make room to store a *textColor* value, and initialize it to the value returned by the `+defaultTextColor` method.

tableScroll:retireAtRow:column:

- (id)**tableScroll:**(MiscTableScroll*)*scroll* **retireAtRow:**(int)*row* **column:**(int)*col*

MiscTableCell's implementation of the MiscTableScroll call-back method. This method is called whenever the cell is removed from active use by the MiscTableScroll as the result of any of the methods that reduce the number of active rows (ex: `-renewRows:`, `-empty`, etc.). The current implementation clears the string value of the cell. Unfortunately, since the overhead of performing these reset operations is quite high, the behavior of this method might change in the future. Returns **self**.

tableScroll:reviveAtRow:column:

- (id)**tableScroll:**(MiscTableScroll*)*scroll* **reviveAtRow:**(int)*row* **column:**(int)*col*

MiscTableCell's implementation of the MiscTableScroll call-back method. This method is called whenever the cell is brought into active use by the MiscTableScroll as the result of any of the methods that expand the number of active rows (ex: `-renewRows:`, `-addRow`, etc.). This method sets *scroll* as the *owner* of the cell, then ensures that all attributes are set to the *use-owner* values. Unfortunately, since the overhead of performing these initializations is quite high, the behavior of this method might change in the future. Returns **self**.

tag

- (int)**tag**

Returns the cell's *tag* value.

See also: ± setTag:

tc1BackgroundColorLen

- (unsigned int)**tc1BackgroundColorLen**

Returns the length of the *backgroundColor* value in the variable-length *tc1_data* field. This is currently sizeof(NSColor*).

tc1BackgroundColorPos

- (unsigned int)**tc1BackgroundColorPos**

Returns the current offset of the *backgroundColor* value in the variable-length *tc1_data* field. This value changes whenever preceeding values are added or removed.

tc1BackgroundColorPtr

- (NSColor**)**tc1BackgroundColorPtr**

Returns a pointer to the *backgroundColor* value in the variable-length *tc1_data* field which is, itself, an NSColor*.

The value returned is only valid if `-useOwnerBackgroundColor` returns NO.

tc1DataSize

- (unsigned int)**tc1DataSize**

Returns the current total size of the variable-length *tc1_data* field. This value changes as optional values are added and removed.

tc1DeleteDataPos:len:

- (void)**tc1DeleteDataPos:**(unsigned int)*pos* **len:**(unsigned int)*len*

Deletes *len* bytes at offset *pos* from the variable-length *tc1_data* field. All following data is shifted down by *len* bytes. This method is called internally whenever an optional value no longer needs to be stored because the inherited value from the owner is going to be used.

See also: **± tc1InsertData:pos:len:**

tc1DestroyData

- (void)**tc1DestroyData**

This method is provided as a hook for subclasses that need to perform special actions (destructors) before the variable-length *tc1_data* field is freed. The MiscTableCell implementation of this class clears the *tc1_flags* bit mask to zero.

See also: ± tc1FreeData

tc1Flags

- (unsigned int)**tc1Flags**

Returns the current value of the *tc1_flags* field. This is a bit-mask indicating which optional values are currently stored in the variable-length *tc1_data* field. The flags also indicate whether or not the corresponding values inherited from the owner are used, since any value that has been set (and stored) in the cell overrides the inherited value. The values of the individual bits are declared in `MiscTableCell.h` as follows:

```
#define MISC_TC1_HAS_TAG                     (1 << 0)  /* obsolete */
#define MISC_TC1_SELF_FONT                   (1 << 1)
#define MISC_TC1_SELF_TEXT_COLOR             (1 << 2)
#define MISC_TC1_SELF_BACKGROUND_COLOR       (1 << 3)
#define MISC_TC1_SELF_TEXT_COLOR_H           (1 << 4)
```

```
#define MISC_TC1_SELF_BACKGROUND_COLOR_H  (1 << 5)
#define MISC_TC1_IS_SELECTED              (1 << 6)
#define MISC_TC1_SELF_DRAW                (1 << 7)  /* !ownerDraw */
#define MISC_TC1_LAST_BIT                 (1 << 7)
```

tc1FreeData

- (void)**tc1FreeData**

Frees the variable-length *tc1_data* variable. This method is invoked from within the `-dealloc` method, and also within the `-initWithCoder:` method.

See also: ± tc1DestroyData

tc1InsertData:pos:len:

- (void*)**tc1InsertData:**(void const*)*data* **pos:**(unsigned int)*pos* **len:**(unsigned int)*len*

Inserts *data* at offset *pos* in the variable-length *tc1_data* field, shifting all following data by *len* bytes. This method is used internally to allocate the storage for optional values that do not already have storage allocated.

See also: ± tc1DeleteDataPos:len:

tc1SelectedBackgroundColorLen

- (unsigned int)**tc1SelectedBackgroundColorLen**

Returns the length of the *selectedBackgroundColor* value in the variable-length *tc1_data* field. This is currently sizeof(NSColor*).

tc1SelectedBackgroundColorPos

- (unsigned int)**tc1SelectedBackgroundColorPos**

Returns the current offset of the *selectedBackgroundColor* value in the variable-length *tc1_data* field. This value changes whenever preceeding values are added or removed.

tc1SelectedBackgroundColorPtr

- (NSColor**)**tc1SelectedBackgroundColorPtr**

Returns a pointer to the *selectedBackgroundColor* value in the variable-length *tc1_data* field which is, itself, an NSColor*. The value returned is only valid if `-useOwnerSelectedBackgroundColor` returns NO.

tc1SelectedTextColorLen

- (unsigned int)**tc1SelectedTextColorLen**

Returns the length of the *selectedTextColor* value in the variable-length *tc1_data* field.   This is currently sizeof(NSColor*).

tc1SelectedTextColorPos

- (unsigned int)**tc1SelectedTextColorPos**

Returns the current offset of the *selectedTextColor* value in the variable-length *tc1_data* field.   This value changes whenever preceeding values are added or removed.

tc1SelectedTextColorPtr

- (NSColor**)**tc1SelectedTextColorPtr**

Returns a pointer to the *selectedTextColor* value in the variable-length *tc1_data* field which is, itself, an NSColor*. The value returned is only valid if `-useOwnerSelectedTextColor` returns NO.

tc1TextColorLen

- (unsigned int)**tc1TextColorLen**

Returns the length of the *textColor* value in the variable-length *tc1_data* field.   This is currently sizeof(NSColor*).

tc1TextColorPos

- (unsigned int)**tc1TextColorPos**

Returns the current offset of the *textColor* value in the variable-length *tc1_data* field.   This offset changes as other values are set or cleared.

tc1TextColorPtr

- (NSColor**)**tc1TextColorPtr**

Returns a pointer to the location of the *textColor* value stored in the variable-length *tc1_data* field which is, itself, an NSColor*.   The value returned is only valid if `-useOwnerTextColor` returns NO.

textColor

- (NSColor*)**textColor**

Returns the color that is used to draw the text in the cell in its normal (unselected) state. This is the color which has been set with **±**`setTextColor:` if it was ever called. If not, and the owner has been set and responds to **±**`textColor` then it is queried and that color is returned. If both of the above fail then the value from `+defaultTextColor` is returned.

See also: ± backgroundColor, + defaultTextColor, ± selectedBackgroundColor, ± selectedTextColor, ± €setTextColor:

useOwnerBackgroundColor

- (BOOL)**useOwnerBackgroundColor**

Returns YES if the cell uses the *backgroundColor* inherited from its *owner*, otherwise NO.

See also: ±€setBackgroundColor:, ± setUseOwnerBackgroundColor:

useOwnerFont

- (BOOL)**useOwnerFont**

Returns YES if the cell uses the *font* inherited from its *owner*, otherwise NO.

See also: **± setFont:, ± setUseOwnerFont:**

useOwnerSelectedBackgroundColor

- (BOOL)**useOwnerSelectedBackgroundColor**

Returns YES if the cell uses the *selectedBackgroundColor* inherited from its *owner*, otherwise NO.

See also: **± setSelectedBackgroundColor:, ± setUseOwnerSelectedBackgroundColor:**

useOwnerSelectedTextColor

- (BOOL)**useOwnerSelectedTextColor**

Returns YES if the cell uses the *selectedTextColor* inherited from its *owner*, otherwise NO.

See also: **±€setSelectedTextColor:, ± setUseOwnerSelectedTextColor:**

useOwnerTextColor

- (BOOL)**useOwnerTextColor**

Returns YES if the cell uses the *textColor* value inherited from its *owner*, otherwise NO.

See also: **± setTextColor:, ± setUseOwnerTextColor:**

Constants and Defined Types

```
#define MISC_TC1_HAS_TAG                   (1 << 0)  /* obsolete */
#define MISC_TC1_SELF_FONT                 (1 << 1)
#define MISC_TC1_SELF_TEXT_COLOR           (1 << 2)
#define MISC_TC1_SELF_BACKGROUND_COLOR     (1 << 3)
#define MISC_TC1_SELF_TEXT_COLOR_H         (1 << 4)
#define MISC_TC1_SELF_BACKGROUND_COLOR_H   (1 << 5)
#define MISC_TC1_IS_SELECTED               (1 << 6)
#define MISC_TC1_SELF_DRAW                 (1 << 7)  /* !ownerDraw */
#define MISC_TC1_LAST_BIT                  (1 << 7)
```