FC_Display

Olivier LAVIALE 2004

COLLABORATORS			
	TITLE : FC_Display		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Olivier LAVIALE 2004	January 13, 2023	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

FC_Display

1.1 Feelin : FC_Display

FC_Display (03.00)

IDs: Dynamic Super: FC_Object Include: <libraries/feelin.h>

This class manages display environnements. A display for Feelin is a screen for Intuition. Like public screens, displays are sharable by nature. Many FC_Display objects can share the same display (or screen).

Each application uses a FC_Display object to create its custom color scheme, and a display environnements for its windows. If several applications request the same kind of display, only one is opened, applications share the same environnement. Displays are closed automatically when their user count drops to zero. This is the hidden part.

The class also manages display's colors. A color is an ARGB value : Alpha, Red, Green and Blue. Each component is described with 8 bits e.g. 0xFF0000 for full red, the alpha component is ignored. A pen is the result of this color allocated from the colormap of a display (or screen). In other words, a pen is an indexed value within a table of 2 to 256 entries.

You should pay attention to the limited number of colors available, because even Hi- and True-color display have this limit. If you want to remap a picture, check display's depth, and use colors only for depth inferior or equal to 8 bits.

NOTE

Although stated as using Dynamic IDs, which is generaly true, the class also uses some static IDs. These static IDs are easy to recognize because they don't have the 'Display_' part e.g. 'FM_CreateColor'.

This was decided for convenience, because FM_CreateColor or FM_CreateColorScheme are oftenly used.

METHODS

FM_Display_Find FM_Display_Create

FM_Display_Delete

FM_Display_AddColor FM_Display_RemColor

FM_Display_AddPalette FM_Display_RemPalette

FM_CreateColor

FM_DeleteColor FM_CreateColorScheme

FM_DeleteColorScheme

ATTRIBUTES

FA_Display_Background FA_Display_ColorMap

FA_Display_Depth FA_Display_BitMap

FA_Display_DrawInfo FA_Display_Height

FA_Display_Name FA_Display_Screen

FA_Display_ScreenMode FA_Display_Spec

FA_Display_Title FA_Display_Width

1.2 FC_Display / FM_DeleteColor

NAME

FM_DeleteColor -- (01.00)

SYNOPSIS

F_Do(Obj,FM_DeleteColor,FColor *Color);

FUNCTION

An interface to the FM_Display_RemColor method.

SEE ALSO

FM_CreateColor

1.3 FC_Display / FM_DeleteColorScheme

NAME

FM_DeleteColorScheme -- (01.00)

SYNOPSIS

F_DoA(Obj,FM_DeleteColorScheme,FS_DeleteColorScheme);

F_Do(Obj,FM_DeleteColorScheme,FPalette *Palette);

FUNCTION

An interface to the FM_Display_RemPalette method.

FM_CreateColorScheme

1.4 FC_Display / FM_Display_AddColor

NAME

FM_Display_AddColor -- (00.00)

SYNOPSIS

F_DoA(Obj,FM_Display_AddColor,struct * FS_AddColor);

F_Do(Obj,FM_Display_AddColor,ULONG ARGB);

FUNCTION

Figures out what pen to use to represent a given color.

A color is an ARGB value : Alpha, Red, Green and Blue. Each component is described with 8 bits e.g. 0xFF0000 for full red, the alpha component is ignored. A pen is the result of this color allocated from the colormap of a display (or screen). In other words, a pen is an indexed value within a table of 2 to 256 entries.

You should pay attention to the limited number of colors available, because even Hi- and True-color display have this limit. If you want to remap a picture, check display's depth, and use colors only for depth inferior or equal to 8 bits.

The method will try to find the pen closest to the specified color. If there is no pen within a good tolerence, then a new one will be allocated, if available. If none is available, then the closest one found will be returned.

INPUTS

ARGB - An unsigned long word describing the color to allocate e.g. 0x00FF0000 for full red. The alpha component (bits 24 to 31) is ignored. e.g. 0x00FFFFFF and 0xFFFFFFFF are full dark.

RESULT

A FColor .

Use this pointer to remove the color with the FM_Display_RemColor method.

SEE ALSO

FM_CreateColor FM_Display_AddPalette

1.5 FC_Display / FM_Display_AddPalette

NAME

FM_Display_AddPalette -- (01.00)

SYNOPSIS

F_DoA(Obj,FM_Display_AddPalette,struct * FS_AddPalette);

F_Do(Obj,FM_Display_AddPalette,ULONG Count,ULONG *ARGBs);

FUNCTION

Allocate several colors, called a palette, from the display's colormap.

The number of FColor that can be created in a single step is limited to 256. This is not a conception problem, nor an internal limitation, but there is no valuable reason to allocate more colors. The purpose of FC_Display is to provide a simple color managing system usable regardless display's depth. Its purpose is not to allocate 32000 colors because you are lazy remapping a picture. There are other techniques for this, and I want to preserve the few available pens (even true-color displays cannot share more than 256 colors).

INPUTS Count (ULONG)

The number of FColor to create.

ARGBs (ULONG *)

An array of unsigned long words describing colors to allocate e.g. 0x00FF0000 for full red. The alpha component (bits 24 to 31) is ignored. e.g. 0x00FFFFFF and 0xFFFFFFFF are full dark.

RESULT

A FPalette , that should be delete with the FM_Display_RemPalette method.

NOTE

FPalette s are shared by nature (as the FColor s they hold). If a similar palette already exists, its user count is incremented and a pointer to this palette is returned. So, don't be suprised if you obtain several same pointers.

SEE ALSO

FM_CreateColorScheme

1.6 FC_Display / FM_Display_Create

NAME

FM_Display_Create -- (01.00)

SYNOPSIS

F_Do(Class,FM_Display_Create);

FUNCTION

Class' method, not documented yet.

1.7 FC_Display / FM_Display_Delete

NAME

FM_Display_Delete -- (01.00)

SYNOPSIS

F_Do(Class,FM_Display_Delete);

FUNCTION

Class' method, not documented yet.

1.8 FC_Display / FM_Display_Find

NAME

FM_Display_Find -- (01.00)

SYNOPSIS

F_Do(Class,FM_Display_Find);

FUNCTION

Class' method, not documented yet.

1.9 FC_Display / FM_Display_RemColor

NAME

FM_Display_RemColor -- (00.00)

SYNOPSIS

F_Do(Obj,FM_Display_RemColor,FColor *);

FUNCTION

Returns the palette entry for use by other objects.

INPUTS

Color - A FColor returned by FM_Display_AddColor .

SEE ALSO

FM_CreateColor

1.10 FC_Display / FM_Display_RemPalette

NAME

FM_Display_RemPalette -- (01.00)

SYNOPSIS

F_DoA(Obj,FM_Display_RemPalette,struct FS_Display_RemPalette);

F_Do(Obj,FM_Display_RemPalette,FPalette *Palette);

FUNCTION

Deletes a FPalette created by the FM_Display_AddPalette method.

Each FColor in the palette is deleted with the FM_Display_RemColor method.

INPUTS

Palette - The FPalette to delete.

NOTE

FPalette s are shared by nature (as the FColor s they hold). If a similar palette already exists, its user count is incremented and a pointer to this palette is returned. So, don't be suprised if colors are not freed when you delete a palette.

SEE ALSO

FM_Display_AddPalette

1.11 FC_Display / FM_CreateColor

NAME

FM_CreateColor -- (01.00)

SYNOPSIS

F_Do(Obj,FM_CreateColor,STRPTR Spec,FPalette * Reference);

FUNCTION

This method is an interface to the FM_Display_AddColor method, extending the method by handling complex color decoding with reference.

INPUTS Spec (STRPTR)

Pointer to a string describing the color to create :

s:<n> - where <n> is a color from the color scheme. e.g. "s:1" will be Shine. r:<n> - where <n> is a register number from the Display's colormap. e.g. "r:3" will be pen 3 of the colormap. c:<r,g,b> - where <r>, <g> and <b> are hexadecimal values of the color to create. e.g. "c:FF0000" will be full red.

Reference (FPalette *)

Pointer to a FPalette to use as reference.

In an application tree, each child creates its color scheme using its parent's color scheme as reference. Using exactly the same specification 's:2,s:4', objects get darker and darker as the application tree goes deep. So don't forget to give your parent's color scheme as reference when you create colors or color schemes.

RESULT

A pointer to a FColor .

Don't forget to release the color using FM_DeleteColor .

SEE ALSO

FM_Display_AddColor FM_CreateColorScheme

1.12 FC_Display / FM_CreateColorScheme

NAME

FM_CreateColorScheme -- (01.00)

SYNOPSIS

F_DoA(Obj,FM_CreateColorScheme,struct FS_CreateColorScheme *);

F_Do(Obj,FM_CreateColorScheme,STRPTR Spec,FPalette *Reference);

FUNCTION

Creates a color scheme.

A color scheme is a FPalette , composed of 9 (FV_PEN_COUNT) colors, created with the FM_Display_AddPalette method. Colors' specifications are decoded using the same functions as the FM_CreateColor method.

Only Shine, Fill, Dark, Text and Highlight can be defined. HalfShine, HalfShadow and HalfDark are computed. Omitted values can replaced with values from the Reference, or obtained from the display (actually Intuition) if the Reference is NULL.

Pen values can be accessed from the 'Pens' field of FPalette . The special values FV_Pen_Shine, FV_Pen_HalfShine, FV_Pen_Fill, FV_Pen_HelfShadow, FV_Pen_Shadow, FV_Pen_HalfDark, FV_Pen_Dark, FV_Pen_Text and FV_Pen_Highlight should be used to obtain pen values.

INPUTS Spec (STRPTR)

Pointer to a string describing the color scheme to create. The full template is: "<shine_spec>,<fill_spec>,<dark_spec>,<text_spec>,<high

Omitted values (e.g. ",,") are replaced by values defined in Reference, or default values (in that priority).

Reference (FPalette *)

A FPalette create by FM_Display_AddPalette (or FM_CreateColorScheme). If not NULL, these colors are used as references for creating the new ones. Omitted values (e.g. ",,") are replaced by RGB values from this palette (mainly used by "s:").

In an application tree, each child creates its color scheme using its parent's color scheme as reference. Using exactly the same specification 's:2,s:4', objects get darker and darker as the application tree goes deep. So don't forget to give your parent's color scheme as reference when you create colors or color schemes.

RESULT

A FPalette composed of 9 (FV_PEN_COUNT) colors.

EXAMPLE

FPalette *pal_0 = (FPalette *) F_Do(Obj,FM_CreateColorScheme,"c:FFFFFF,c:BEBEBE,c:000000,c:000000,c:FF000000",NULL);
FPalette *pal_1 = (FPalette *) F_Do(Obj,FM_CreateColorScheme,"s:2,s:4",pal_0);

...

FPalette *pal_intuition = (FPalette *) F_Do(Obj,FM_CreateColorScheme,NULL,NULL);

...

SEE ALSO

FM_CreateColor FM_DeleteColorScheme

FM_Display_AddPalette

1.13 FC_Display / FColor

NAME

FColor -- (01.00)

STRUCT

struct FeelinColor { ULONG _priv0; ULONG _priv1; ULONG Pen; ULONG ARGB; };

FUNCTION

A color allocated on a display.

The structure is created by the FM_Display_AddColor method, and deleted by the FM_Display_RemColor method.

FIELDS Pen

Color's register number (pen) allocated from the display's colormap (or screen). Use its value to set a pen to draw with.

ARGB

An hexadecimal value representing the color value. e.g 0xFF0000 is full red. The alpha component ('A') is ignored.

EXAMPLE

FColor * col;

ULONG id_AddColor = F_DynamicFindID("FM_Display_AddColor"); ULONG id_RemColor = F_DynamicFindID("FM_Display_Re

col = F_Do(display,id_AddColor,0xFF00FF);

_APen(col -> Pen); _Boxf(10,10,109,109);

F_Do(display,id_RemColor,col);

NOTE

FColors are shared by nature. If a similar color already exists, its user count is incremented and a pointer to this color is returned. So, don't be suprised if you obtain several same pointers. Or if a color is not freed when your remove it.

SEE ALSO

FPalette

1.14 FC_Display / FPalette

NAME

FPalette -- (01.00)

STRUCT

struct FeelinPalette { ULONG _priv0; ULONG _priv1; ULONG _priv2; ULONG ColorCount; ULONG *Pens; ULONG *ARGB; FPen **Colors; };

FUNCTION

Several colors, called a palette, allocated on a display.

The structure is created by the FM_Display_AddPalette method, and deleted by the FM_Display_RemPalette method.

FIELDS ColorCount

The number of colors stored is the palette.

*Pens

Pointer to an array of color's register number (pens) allocated from the display's colormap (or screen). Use these values to set pens to draw with.

*ARGB

Hexadecimal values representing the colors values. e.g 0xFF0000 is full red. Alpha components ('A') are ignored.

**Colors

Pointer to an array of FColor , one for each color in the palette. These FColor s are created by the FM_Display_AddColor method.

NOTE

FPalettes are shared by nature (as the FColor s they hold). If a similar palette already exists, its user count is incremented and a pointer to this palette is returned. So, don't be suprised if you obtain several same pointers. Or if colors are not freed when your delete a palette.

SEE ALSO

FPalette