NAME

XCreateColormap, XCopyColormapAndFree, XFreeColormap, XColor - create, copy, or destroy colormaps and color structure

SYNTAX

Colormap XCreateColormap(Display **display*, Window *w*, Visual **visual*, int *alloc*);

Colormap XCopyColormapAndFree(Display **display*, Colormap *colormap*);

int XFreeColormap(Display **display*, Colormap *colormap*);

ARGUMENTS

alloc Specifies the colormap entries to be allocated. You can pass **AllocNone** or **AllocAll**.

colormap Specifies the colormap that you want to create, copy, set, or destroy.

display Specifies the connection to the X server.

visual Specifies a visual type supported on the screen. If the visual type is not one supported by the screen, a **BadMatch** error results.

w Specifies the window on whose screen you want to create a colormap.

DESCRIPTION

The **XCreateColormap** function creates a colormap of the specified visual type for the screen on which the specified window resides and returns the colormap ID associated with it. Note that the specified window is only used to determine the screen.

The initial values of the colormap entries are undefined for the visual classes **GrayScale**, **PseudoColor**, and **DirectColor**. For **StaticGray**, **StaticColor**, and **TrueColor**, the entries have defined values, but those values are specific to the visual and are not defined by X. For **StaticGray**, **StaticColor**, and **TrueColor**, alloc must be **AllocNone**, or a **BadMatch** error results. For the other visual classes, if alloc is **AllocNone**, the colormap initially has no allocated entries, and clients can allocate them. For information about the visual types, see section 3.1.

If alloc is **AllocAll**, the entire colormap is allocated writable. The initial values of all allocated entries are undefined. For **GrayScale** and **PseudoColor**, the effect is as if an **XAllocColorCells** call returned all pixel values from zero to N - 1, where N is the colormap entries value in the specified visual. For **DirectColor**, the effect is as if an **XAllocColorPlanes** call returned a pixel value of zero and red_mask, green_mask, and blue_mask values containing the same bits as the corresponding masks in the

specified visual. However, in all cases, none of these entries can be freed by using **XFreeColors**.

XCreateColormap can generate **BadAlloc**, **BadMatch**, **BadValue**, and **BadWindow** errors.

The **XCopyColormapAndFree** function creates a colormap of the same visual type and for the same screen as the specified colormap and returns the new colormap ID. It also moves all of the client's existing allocation from the specified colormap to the new colormap with their color values intact and their read-only or writable characteristics intact and frees those entries in the specified colormap. Color values in other entries in the new colormap are undefined. If the specified colormap was created by the client with alloc set to **AllocAll**, the new colormap is also created with **AllocAll**, all color values for all entries are copied from the specified colormap, and then all entries in the specified colormap are freed. If the specified colormap was not created by the client with **AllocAll**, the allocations to be moved are all those pixels and planes that have been allocated by the client using **XAllocColor**, **XAllocNamedColor**, **XAllocColorCells**, or **XAllocColorPlanes** and that have not been freed since they were allocated.

XCopyColormapAndFree can generate **BadAlloc** and **BadColor** errors.

The **XFreeColormap** function deletes the association between the colormap resource ID and the colormap and frees the colormap storage. However, this function has no effect on the default colormap for a screen. If the specified colormap is an installed map for a screen, it is uninstalled (see **XUninstallColormap**). If the specified colormap is defined as the colormap for a window (by **XCreateWindow**, **XSetWindowColormap**, or **XChangeWindowAttributes**), **XFreeColormap** changes the colormap associated with the window to **None** and generates a **ColormapNotify** event. X does not define the colors displayed for a window with a colormap of **None**.

XFreeColormap can generate a **BadColor** error.

STRUCTURES

The **XColor** structure contains:

```
typedef struct {
    unsigned long pixel;    /* pixel value */
    unsigned short red, green, blue;        /* rgb values */
    char flags;     /* DoRed, DoGreen, DoBlue */
    char pad;
} XColor;
```

The red, green, and blue values are always in the range 0 to 65535 inclusive, independent of the number of bits actually used in the display hardware. The server scales these values down to the range

used by the hardware. Black is represented by (0,0,0), and white is represented by (65535,65535,65535). In some functions, the flags member controls which of the red, green, and blue members is used and can be the inclusive OR of zero or more of **DoRed**, **DoGreen**, and **DoBlue**.

DIAGNOSTICS

BadAlloc The server failed to allocate the requested resource or server memory.

BadColor A value for a Colormap argument does not name a defined Colormap.

BadMatch An **InputOnly** window is used as a Drawable.

BadMatch Some argument or pair of arguments has the correct type and range but fails to match in some other way required by the request.

BadValue Some numeric value falls outside the range of values accepted by the request. Unless a specific range is specified for an argument, the full range defined by the argument's type is accepted. Any argument defined as a set of alternatives can generate this error.

BadWindow A value for a Window argument does not name a defined Window.

SEE ALSO

XAllocColor(3), XChangeWindowAttributes(3), XCreateWindow(3), XQueryColor(3), XStoreColors(3)
Xlib - C Language X Interface