Colormap Utilization Policy and Extension

Version 1.0
X Project Team Standard
X Version 11, Release 6.4

Kaleb S. KEITHLEY
The Open Group
1. Overview

This extension has three purposes: a) to provide mechanism for a special application (a colormap manager) to discover any special colormap requirements, e.g. the colormap entries that are nominally reserved for desktop colors in the MS-Windows environment and initialize the default colormap so that it can be more easily shared; and b) to encourage colormap sharing and reduce colormap flashing on low-end 8-bit frame buffers by providing a policy for sharing; and c) when colormaps aren't shared, define a behavior in the X server color allocation scheme to reduce colormap flashing.

To encourage colormap sharing and accommodate special colormap requirements two new protocols are defined: the first provides a way to query the server for a list of reserved colormap entries, and the second is a way to initialize read-only (shareable) colormap entries at specific locations in a colormap.

To minimize colormap flashing when the root window's default visual is one of GrayScale, PseudoColor, or DirectColor, and a private colormap for the default visual is being used, a minor (but compatible) change to the server implementation of the AllocColor and AllocNamedColor requests is required. Where the core protocol says nothing about the pixel values returned, when this extension is in effect, the AllocColor and AllocNamedColor requests will first look for a matching color in the default colormap, and, if a match is found and the same cell in the private colormap has not already been allocated, the color will be allocated in the private colormap at the same location as in the default colormap (instead of in the first available location.)

2. Requests

(QueryVersion)

client_major_version: CARD16
client_minor_version: CARD16

=>

server_major_version: CARD16
server_minor_version: CARD16

If supplied, the client_major_version and client_minor_version indicate what version of the protocol the client wants the server to implement. The server version numbers returned indicate the protocol this extension actually supports. This might not equal the version sent by the client. An implementation can (but need not) support more than one version simultaneously. The server_major_version and the server_minor_version are a mechanism to support future revisions of the TOG-CUP protocol that may be necessary. In general, the major version would increment for incompatible changes, and the minor version would increment for small upward-compatible changes. Servers that support the protocol defined in this document will return a server_major_version of one (1), and a server_minor_version of zero (0).

(GetReservedColormapEntries)

screen: CARD32

=>

entries: LISTofCOLORITEM

This request returns a list of colormap entries (pixels) that are reserved by the system, e.g. MS-Windows reserved desktop colors. This list will, at a minimum, contain entries for the BlackPixel and WhitePixel of the specified screen. The do-red, do-green, and do-blue elements of the COLORITEMs are unused in this reply.

Rationale: There are colormap entries (pixels) that, e.g., MS-Windows desktop reserves as desktop colors, that should not be altered. If they are altered then X programs will cause colormap flashing between X and MS-Windows applications running/displaying on the same desktop.
StoreColors

cmap: COLORMAP
items: LISTofCOLORITEM
=>
items: LISTofCOLORITEM

This request changes the colormap entries of the specified pixels. The colormap entries are allocated as if by an AllocColor request. The do-red, do-green, and do-blue elements of the COLORITEMs are unused in this request. A boolean alloc-ok element (a bit) is returned indicating whether the particular pixel was successfully allocated or not. If successfully allocated the RGB and pixel are returned.

A Value error is generated if a pixel is not a valid index into cmap. A BadMatch error is generated if cmap does not belong to a GrayScale, PseudoColor, or DirectColor visual.

3. Events and Errors

No new events or errors are defined by this extension.

4. Changes to existing protocol.

None.

5. Encoding

The name of this extension is "TOG-CUP".
The conventions used here are the same as those for the core X11 Protocol Encoding.

QueryVersion

1    CARD8          opcode
1    0              TOG-CUP opcode
2    2              request length
2    CARD16         client_major_version
2    CARD16         client_minor_version

=>
1    1              reply
1    unused
2    CARD16         sequence number
4    0              length
2    CARD16         server_major_version
2    CARD16         server_minor_number
20   unused

GetReservedColormapEntries

1    CARD8          opcode
1    1              TOG-CUP opcode
2    2              request length
4    CARD32         screen

=>
1    1              reply
1    unused
2    CARD16         sequence number
4    3n             length
X11 Colormap Utilization Policy and Extension

24 unused
12n LISTofCOLORITEM items

StoreColors
1  CARD8 opcode
1  2 TOG-CUP opcode
2  2+3n request length
4  COLORMAP cmap
12n LISTofCOLORITEM items

=>
1  1 reply
1 unused
2  CARD16 sequence number
4  3n length
24 unused
12n LISTofCOLORITEM items

(The definition of COLORITEM here is only for the purpose of defining the additional alloc-ok member in
the CUPStoreColors reply.)

COLORITEM
4  CARD32 pixel
2  CARD16 red
2  CARD16 green
2  CARD16 blue
1 alloc-ok
    #x07 unused
    #x08 alloc-ok (1 is True, 0 is False)
    #xF0 unused
1 unused

6. C Language Binding
The C functions provide direct access to the protocol and add no additional semantics. For complete details
on the effects of these functions, refer to the appropriate protocol request, which can be derived by deleting
X Cup at the start of the function. All functions that have return type Status will return nonzero for success
and zero for failure.
The include file for this extension is <X11/extensions/Xcup.h>.

StatusX CupQueryVersion(
    Display* display,
    int* major_version_return,
    int* minor_version_return)

display Specifies the connection to the X server.
major_version_return Returns the major version supported by the server.
minor_version_return Returns the minor version supported by the server.

X CupQueryVersions sets major_version_return and minor_version_return to the major and minor TOG-
CUP protocol version supported by the server. If the TOG-CUP library is compatible with the version
returned by the server, it returns nonzero. If dpy does not support the TOG-CUP extension, or if there was 
an error during communication with the server, or if the server and library protocol versions are incompati-
ble, it returns zero. No other XCup functions may be called before this function. If a client violates this 
rule, the effects of all subsequent XCup calls that it makes are undefined.

To get the list of reserved colormap entries, use XCupGetReservedColormapEntries.

**Status**

```c
XStatus XCupGetReservedColormapEntries(
    Display* display,
    int screen,
    XColor** colors_out,
    int* ncolors)
```

*display* Specifies the connection to the X server.
*colors_out* Returns the values reserved by the server.
*ncolors* Returns the number of items in *colors_out.*

The XCupGetReservedColormapEntries function gets system specific colormap entries. E.g. the MS-Win-
dows desktop uses *N* colormap entries at the beginning (0..*N*) and end (256-*N*..255) of the colormap. Use 
XFree to free *colors_out.*

To allocate one or more read-only color cells with RGB values, use XCupStoreColors.

**Status**

```c
XStatus XCupStoreColors(
    Display* display,
    Colormap colormap,
    XColor* colors_in_out,
    int ncolors)
```

*display* Specifies the connection to the X server.
*colormap* Specifies the colormap.
*colors_in_out* Specifies and returns the values actually used in the colormap.
*ncolors* Specifies the number of items in *colors_in_out.*

The XCupStoreColors function changes the colormap entries of the pixel values specified in the pixel mem-
bers of the XColor structures. The colormap entries are allocated as if an AllocColor had been used instead,
i.e. the colors are read-only (shareable). XCupStoreColors returns the number of colors that were success-
fully allocated in the colormap.

### 7. Using the TOG-CUP extension and Colormap Utilization Policy

The X server preallocates any hardware or desktop special colors in the default colormap; e.g. UNIX X 
servers preallocate Black and White pixels. PC X servers should also preallocate the MS-Windows desktop 
colors. (Note to implementors: in the Sample Implementation special colors are allocated in the default col-
ormap in cfbCreateDefColormap for dumb memory framebuffers.)

To minimize colormap flash an application which installs its own private colormap should query the special 
colors by calling XCupGetReservedColormapEntries, and can then store those entries (in the proper loca-
tion) in its private colormap using XCupStoreColors.

Applications which allocate many colors in a screen’s default colormap, e.g. a color-cube or a gray-ramp, 
should allocate them with XCupStoreColors. By using XCupStoreColors the colors will be allocated 
shareable (read-only) and any other application which allocates the same color will share that color cell.
X11 Colormap Utilization Policy and Extension