X Nonrectangular Window Shape Extension Library

X Consortium Standard

Keith Packard

X Nonrectangular Window Shape Extension Library: X Consortium Standard

by Keith Packard

X Version 11, Release 6.4

Version 1.0

Copyright © 1989 X Consortium

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE X CONSORTIUM BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of the X Consortium shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from the X Consortium.

Table of Contents

Chapter 1. Overview

This extension provides arbitrary window and border shapes within the X11 protocol.

The restriction of rectangular windows within the X protocol is a significant limitation in the implementation of many styles of user interface. For example, many transient windows would like to display a “drop shadow” to give the illusion of 3 dimensions. As another example, some user interface style guides call for buttons with rounded corners; the full simulation of a nonrectangular shape, particularly with respect to event distribution and cursor shape, is not possible within the core X protocol. As a final example, round clocks and nonrectangular icons are desirable visual addition to the desktop.

This extension provides mechanisms for changing the visible shape of a window to an arbitrary, possibly disjoint, nonrectangular form. The intent of the extension is to supplement the existing semantics, not replace them. In particular, it is desirable for clients that are unaware of the extension to still be able to cope reasonably with shaped windows. For example, window managers should still be able to negotiate screen real estate in rectangular pieces. Toward this end, any shape specified for a window is clipped by the bounding rectangle for the window as specified by the window's geometry in the core protocol. An expected convention would be that client programs expand their shape to fill the area offered by the window manager.

Chapter 2. Description

Each window (even with no shapes specified) is defined by two regions: the *bounding region* and the *clip region*. The bounding region is the area of the parent window that the window will occupy (including border). The clip region is the subset of the bounding region that is available for subwindows and graphics. The area between the bounding region and the clip region is defined to be the border of the window.

A nonshaped window will have a bounding region that is a rectangle spanning the window, including its border; the clip region will be a rectangle filling the inside dimensions (not including the border). In this document, these areas are referred to as the *default bounding region* and the *default clip region*. For a window with inside size of *width* by *height* and border width *bwidth*, the default bounding and clip regions are the rectangles (relative to the window origin):

bounding.x = -*bwidth*
bounding.y = -*bwidth*
bounding.width = *width* + 2 * *bwidth*
bounding.height = *height* + 2 * *bwidth*

clip.x = 0
clip.y = 0
clip.width = *width*
clip.height = *height*

This extension allows a client to modify either or both of the bounding or clip regions by specifying new regions that combine with the default regions. These new regions are called the *client bounding region* and the *client clip region*. They are specified relative to the origin of the window and are always defined by offsets relative to the window origin (that is, region adjustments are not required when the window is moved). Three mechanisms for specifying regions are provided: a list of rectangles, a bitmap, and an existing bounding or clip region from a window. This is modeled on the specification of regions in graphics contexts in the core protocol and allows a variety of different uses of the extension.

When using an existing window shape as an operand in specifying a new shape, the client region is used, unless none has been set, in which case the default region is used instead.

The *effective bounding region* of a window is defined to be the intersection of the client bounding region with the default bounding region. Any portion of the client bounding region that is not included in the default bounding region will not be included in the effective bounding region on the screen. This means that window managers (or other geometry managers) used to dealing with rectangular client windows will be able to constrain the client to a rectangular area of the screen.

Construction of the effective bounding region is dynamic; the client bounding region is not mutated to obtain the effective bounding region. If a client bounding region is specified that extends beyond the current default bounding region, and the window is later enlarged, the effective bounding region will be enlarged to include more of the client bounding region.

The *effective clip region* of a window is defined to be the intersection of the client clip region with both the default clip region and the client bounding region. Any portion of the client clip region that is not included in both the default clip region and the client bounding region will not be included in the effective clip region on the screen.

Construction of the effective clip region is dynamic; the client clip region is not mutated to obtain the effective clip region. If a client clip region is specified that extends beyond the current default clip region and the window or its bounding region is later enlarged, the effective clip region will be enlarged to include more of the client clip region if it is included in the effective bounding region.

The border of a window is defined to be the difference between the effective bounding region and the effective clip region. If this region is empty, no border is displayed. If this region is nonempty, the border is filled using the border-tile or border-pixel of the window as specified in the core protocol. Note that a window with a nonzero border width will never be able to draw beyond the default clip region of the window. Also note that a zero border width does not prevent a window from having a border, since the clip shape can still be made smaller than the bounding shape.

All output to the window and visible regions of any subwindows will be clipped to the effective clip region. The server must not retain window contents beyond the effective bounding region with backing store. The window's origin (for graphics operations, background tiling, and subwindow placement) is not affected by the existence of a bounding region or clip region.

Areas that are inside the default bounding region but outside the effective bounding region are not part of the window; these areas of the screen will be occupied by other windows. Input events that occur within the default bounding region but outside the effective bounding region will be delivered as if the window was not occluding the event position. Events that occur in a nonrectangular border of a window will be delivered to that window, just as for events that occur in a normal rectangular border.

An `InputOnly` window can have its bounding region set, but it is a `Match` error to attempt to set a clip region on an `InputOnly` window or to specify its clip region as a source to a request in this extension.

The server must accept changes to the clip region of a root window, but the server is permitted to ignore requested changes to the bounding region of a root window. If the server accepts bounding region changes, the contents of the screen outside the bounding region are implementation dependent.

Chapter 3. C Language Binding

The C functions provide direct access to the protocol and add no additional semantics.

The include file for this extension is <X11/extensions/shape.h>. The defined shape kinds are `ShapeBounding` and `ShapeClip` The defined region operations are `ShapeSet` `ShapeUnion` `ShapeIntersect` `ShapeSubtract` and `ShapeInvert`.

`Bool` **`XShapeQueryExtension`**`(`*`*display`*`,` *`*event_base`*`,` *`*error_base`*`);`

`XShapeQueryExtension` returns `True` if the specified display supports the SHAPE extension else `False` If the extension is supported, *event_base is set to the event number for `ShapeNotify` events and *error_base would be set to the error number for the first error for this extension. Because no errors are defined for this version of the extension, the value returned here is not defined (nor useful).

`Status` **`XShapeQueryVersion`**`(` *`*display`*`,` *`*major_version`*`,` *`*minor_version`*`);`

If the extension is supported, `XShapeQueryVersion` sets the major and minor version numbers of the extension supported by the display and returns a nonzero value. Otherwise, the arguments are not set and zero is returned.

`XShapeCombineRegion``(` *`*display`*`,` *`dest`*`,` *`dest_kind`*`,` *`x_off`*`,` *`y_off`*`,` *`region`*`,` *`op`*`,` *`*region`*`);`

`XShapeCombineRegion` converts the specified region into a list of rectangles and calls `XShapeCombineRectangles`

`XShapeCombineRectangles``(` *`*display`*`,` *`dest`*`,` *`dest_kind`*`,` *`x_off`*`,` *`y_off`*`,` *`*rectangles`*`,` *`n_rects`*`,` *`op`*`,` *`ordering`*`);`

If the extension is supported, `XShapeCombineRectangles` performs a `ShapeRectangles` operation; otherwise, the request is ignored.

`XShapeCombineMask``(` *`*display`*`,` *`dest`*`,` *`dest_kind`*`,` *`x_off`*`,` *`y_off`*`,` *`src`*`,` *`op`*`);`

If the extension is supported, `XShapeCombineMask` performs a `ShapeMask` operation; otherwise, the request is ignored.

`XShapeCombineShape``(` *`*display`*`,` *`dest`*`,` *`dest_kind`*`,` *`x_off`*`,` *`y_off`*`,` *`src`*`,` *`src_kind`*`,` *`op`*`);`

If the extension is supported, `XShapeCombineShape` performs a `ShapeCombine` operation; otherwise, the request is ignored.

`XShapeOffsetShape``(`*`display`*`,` *`dest`*`,` *`dest_kind`*`,` *`x_off`*`,` *`y_off`*`);`

If the extension is supported, `XShapeOffsetShape` performs a `ShapeOffset` operation; otherwise, the request is ignored.

`Status` **`XShapeQueryExtents`**`(` *`*display`*`,` *`window`*`,` *`*bounding_shaped`*`,` *`*x_bounding`*`,` *`*y_bounding`*`,` *`*w_bounding`*`,` *`*h_bounding`*`,` *`*clip_shaped`*`,` *`*x_clip`*`,` *`*y_clip`*`,` *`*w_clip`*`,` *`*h_clip`*`);`

If the extension is supported, `XShapeQueryExtents` sets x_bounding, y_bounding, w_bounding, h_bounding to the extents of the bounding shape and sets x_clip, y_clip, w_clip, h_clip to the extents of the clip shape. For unspecified client regions, the extents of the corresponding default region are used.

If the extension is supported, a nonzero value is returned; otherwise, zero is returned.

XShapeSelectInput(**display*, *window*, *mask*);

To make this extension more compatible with other interfaces, although only one event type can be selected via the extension, `XShapeSelectInput` provides a general mechanism similar to the standard Xlib binding for window events. A mask value has been defined, `ShapeNotifyMask` that is the only valid bit in mask that may be specified. The structure for this event is defined as follows:

```
typedef struct {
    int type;     /* of event */
    unsigned long serial;     /* # of last request processed by server */
    Bool send_event;     /* true if this came frome a SendEvent request */
    Display *display;     /* Display the event was read from */
    Window window;     /* window of event */
    int kind;     /* ShapeBounding or ShapeClip */
    int x, y;     /* extents of new region */
    unsigned width, height;
    Time time;     /* server timestamp when region changed */
    Bool shaped;     /* true if the region exists */
} XShapeEvent;
```

unsigned long **XShapeInputSelected**(**display*, *window*);

`XShapeInputSelected` returns the current input mask for extension events on the specified window; the value returned if `ShapeNotify` is selected for is `ShapeNotifyMask` otherwise, it returns zero. If the extension is not supported, it returns zero.

XRectangle ***XShapeGetRectangles**(**display*, *window*, *kind*, **count*, **ordering*);

If the extension is not supported, `XShapeGetRectangles` returns NULL. Otherwise, it returns a list of rectangles that describe the region specified by kind.

Glossary

bounding region	The area of the parent window that this window will occupy. This area is divided into two parts: the border and the interior.
clip region	The interior of the window, as a subset of the bounding region. This region describes the area that will be painted with the window background when the window is cleared, will contain all graphics output to the window, and will clip any subwindows.
default bounding region	The rectangular area, as described by the core protocol window size, that covers the interior of the window and its border.
default clip region	The rectangular area, as described by the core protocol window size, that covers the interior of the window and excludes the border.
client bounding region	The region associated with a window that is directly modified via this extension when specified by `ShapeBounding` This region is used in conjunction with the default bounding region to produce the effective bounding region.
client clip region	The region associated with a window that is directly modified via this extension when specified by `ShapeClip` This region is used in conjunction with the default clip region and the client bounding region to produce the effective clip region.
effective bounding region	The actual shape of the window on the screen, including border and interior (but excluding the effects of overlapping windows). When a window has a client bounding region, the effective bounding region is the intersection of the default bounding region and the client bounding region. Otherwise, the effective bounding region is the same as the default bounding region.
effective clip region	The actual shape of the interior of the window on the screen (excluding the effects of overlapping windows). When a window has a client clip region or a client bounding region, the effective clip region is the intersection of the default clip region, the client clip region (if any) and the client bounding region (if any). Otherwise, the effective clip region is the same as the default clip region.