XSetClipOrigin, XSetClipMask, XSetClipRectangles – GC convenience routines

XSetClipOrigin(*display***,** *gc***,** *clip_x_origin***,** *clip_y_origin***)**
Display ***display*;**
GC *gc***;**
int *clip_x_origin***,** *clip_y_origin***;**

XSetClipMask(*display*, *gc*, *pixmap*)
Display **display*;
GC *gc*;
Pixmap *pixmap*;

XSetClipRectangles(*display*, *gc*, *clip_x_origin*, *clip_y_origin*, *rectangles*, *n*, *ordering*)
Display **display*;
GC *gc*;
int *clip_x_origin*, *clip_y_origin*;
XRectangle *rectangles*[];
int *n*;
int *ordering*;

display	Specifies the connection to the X server.
clip_x_origin *clip_y_origin*	Specify the x and y coordinates of the clip-mask origin.
gc	Specifies the GC.
n	Specifies the number of rectangles.
ordering	Specifies the ordering relations on the rectangles. You can pass **Unsorted**, **YSorted**, **YXSorted**, or **YXBanded**.
pixmap	Specifies the pixmap or **None**.
rectangles	Specifies an array of rectangles that define the clip-mask.

The XSetClipOrigin function sets the clip origin in the specified GC. The clip-mask origin is interpreted relative to the origin of whatever destination drawable is specified in the graphics request.

XSetClipOrigin can generate **BadAlloc** and **BadGC** errors.

The **XSetClipMask** function sets the clip-mask in the specified GC to the specified pixmap. If the clip-mask is set to **None**, the pixels are are always drawn (regardless of the clip-origin).

XSetClipMask can generate **BadAlloc**, **BadGC**, **BadMatch**, and **BadValue** errors.

The **XSetClipRectangles** function changes the clip-mask in the specified GC to the specified list of rectangles and sets the clip origin. The output is clipped to remain contained within the rectangles. The clip-origin is interpreted relative to the origin of whatever destination drawable is specified in a graphics request. The rectangle coordinates are interpreted relative to the clip-origin. The rectangles should be nonintersecting, or the graphics results will be undefined. Note that the list of rectangles can be empty, which effectively disables output. This is the opposite of passing **None** as the clip-mask in **XCreateGC**, **XChangeGC**, and **XSetClipMask**.

If known by the client, ordering relations on the rectangles can be specified with the ordering argument. This may provide faster operation by the server. If an incorrect ordering is specified, the X server may generate a **BadMatch** error, but it is not required to do so. If no error is generated, the graphics results are undefined. **Unsorted** means the rectangles are in arbitrary order. **YSorted** means that the rectangles are nondecreasing in their Y origin. **YXSorted** additionally constrains **YSorted** order in that all rectangles with an equal Y origin are nondecreasing in their X origin. **YXBanded** additionally constrains **YXSorted** by requiring that, for every possible Y scanline, all rectangles that include that scanline have an identical Y origins and Y extents.

XSetClipRectangles can generate **BadAlloc**, **BadGC**, **BadMatch**, and **BadValue** errors.

BadAlloc The server failed to allocate the requested resource or server memory. **BadGC** A value for a GContext argument does not name a defined GContext. **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.

XCreateGC(3X11), XDrawRectangle(3X11), XQueryBestSize(3X11), XSetArcMode(3X11), XSetFillStyle(3X11), XSetFont(3X11), XSetLineAttributes(3X11), XSetState(3X11), XSetTile(3X11)
Xlib – C Language X Interface