NAME

XGrabPointer, XUngrabPointer, XChangeActivePointerGrab - grab the pointer

SYNTAX

int XGrabPointer(Display **display*, Window *grab_window*, Bool *owner_events*, unsigned int *event_mask*, int *pointer_mode*, int *keyboard_mode*, Window *confine_to*, Cursor *cursor*, Time *time*);

int XUngrabPointer(Display **display*, Time *time*);

int XChangeActivePointerGrab(Display **display*, unsigned int *event_mask*, Cursor *cursor*, Time *time*);

ARGUMENTS

confine_to Specifies the window to confine the pointer in or **None**.

cursor Specifies the cursor that is to be displayed during the grab or **None**.

display Specifies the connection to the X server.

event_mask Specifies which pointer events are reported to the client. The mask is the bitwise inclusive OR of the valid pointer event mask bits.

grab_window Specifies the grab window.

keyboard_mode
Specifies further processing of keyboard events. You can pass **GrabModeSync** or **GrabModeAsync**.

owner_events Specifies a Boolean value that indicates whether the pointer events are to be reported as usual or reported with respect to the grab window if selected by the event mask.

pointer_mode Specifies further processing of pointer events. You can pass **GrabModeSync** or **GrabModeAsync**.

time Specifies the time. You can pass either a timestamp or **CurrentTime**.

DESCRIPTION

The **XGrabPointer** function actively grabs control of the pointer and returns **GrabSuccess** if the grab was successful. Further pointer events are reported only to the grabbing client. **XGrabPointer** overrides any active pointer grab by this client. If owner_events is **False**, all generated pointer events

are reported with respect to grab_window and are reported only if selected by event_mask. If owner_events is **True** and if a generated pointer event would normally be reported to this client, it is reported as usual. Otherwise, the event is reported with respect to the grab_window and is reported only if selected by event_mask. For either value of owner_events, unreported events are discarded.

If the pointer_mode is **GrabModeAsync**, pointer event processing continues as usual. If the pointer is currently frozen by this client, the processing of events for the pointer is resumed. If the pointer_mode is **GrabModeSync**, the state of the pointer, as seen by client applications, appears to freeze, and the X server generates no further pointer events until the grabbing client calls **XAllowEvents** or until the pointer grab is released. Actual pointer changes are not lost while the pointer is frozen; they are simply queued in the server for later processing.

If the keyboard_mode is **GrabModeAsync**, keyboard event processing is unaffected by activation of the grab. If the keyboard_mode is **GrabModeSync**, the state of the keyboard, as seen by client applications, appears to freeze, and the X server generates no further keyboard events until the grabbing client calls **XAllowEvents** or until the pointer grab is released. Actual keyboard changes are not lost while the pointer is frozen; they are simply queued in the server for later processing.

If a cursor is specified, it is displayed regardless of what window the pointer is in. If **None** is specified, the normal cursor for that window is displayed when the pointer is in grab_window or one of its subwindows; otherwise, the cursor for grab_window is displayed.

If a confine_to window is specified, the pointer is restricted to stay contained in that window. The confine_to window need have no relationship to the grab_window. If the pointer is not initially in the confine_to window, it is warped automatically to the closest edge just before the grab activates and enter/leave events are generated as usual. If the confine_to window is subsequently reconfigured, the pointer is warped automatically, as necessary, to keep it contained in the window.

The time argument allows you to avoid certain circumstances that come up if applications take a long time to respond or if there are long network delays. Consider a situation where you have two applications, both of which normally grab the pointer when clicked on. If both applications specify the timestamp from the event, the second application may wake up faster and successfully grab the pointer before the first application. The first application then will get an indication that the other application grabbed the pointer before its request was processed.

XGrabPointer generates **EnterNotify** and **LeaveNotify** events.

Either if grab_window or confine_to window is not viewable or if the confine_to window lies completely outside the boundaries of the root window, **XGrabPointer** fails and returns **GrabNotViewable**. If the pointer is actively grabbed by some other client, it fails and returns

AlreadyGrabbed. If the pointer is frozen by an active grab of another client, it fails and returns **GrabFrozen**. If the specified time is earlier than the last-pointer-grab time or later than the current X server time, it fails and returns **GrabInvalidTime**. Otherwise, the last-pointer-grab time is set to the specified time (**CurrentTime** is replaced by the current X server time).

XGrabPointer can generate **BadCursor**, **BadValue**, and **BadWindow** errors.

The **XUngrabPointer** function releases the pointer and any queued events if this client has actively grabbed the pointer from **XGrabPointer**, **XGrabButton**, or from a normal button press. **XUngrabPointer** does not release the pointer if the specified time is earlier than the last-pointer-grab time or is later than the current X server time. It also generates **EnterNotify** and **LeaveNotify** events. The X server performs an **UngrabPointer** request automatically if the event window or confine_to window for an active pointer grab becomes not viewable or if window reconfiguration causes the confine_to window to lie completely outside the boundaries of the root window.

The **XChangeActivePointerGrab** function changes the specified dynamic parameters if the pointer is actively grabbed by the client and if the specified time is no earlier than the last-pointer-grab time and no later than the current X server time. This function has no effect on the passive parameters of a **XGrabButton**. The interpretation of event_mask and cursor is the same as described in **XGrabPointer**.

XChangeActivePointerGrab can generate a **BadCursor** and **BadValue** error.

DIAGNOSTICS

BadCursor A value for a Cursor argument does not name a defined Cursor.

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

XAllowEvents(3), XGrabButton(3), XGrabKey(3), XGrabKeyboard(3)
Xlib - C Language X Interface