

**NAME**

XGrabDeviceButton, XUngrabDeviceButton - grab/ungrab extension input device buttons

**SYNOPSIS**

```
#include <X11/extensions/XInput.h>
```

```
int XGrabDeviceButton( Display *display,
                      XDevice *device,
                      unsigned int button,
                      unsigned int modifiers,
                      XDevice* modifier_device,
                      Window grab_window,
                      Bool owner_events,
                      unsigned int event_count,
                      XEventClass *event_list,
                      int this_device_mode,
                      int other_devices_mode);
```

```
int XUngrabDeviceButton( Display *display,
                        XDevice *device,
                        unsigned int button,
                        unsigned int modifiers,
                        XDevice* modifier_device,
                        Window grab_window);
```

display

Specifies the connection to the X server.

device

Specifies the device that is to be grabbed or released

button

Specifies the device button that is to be grabbed or released or AnyButton.

modifiers

Specifies the set of keymasks or AnyModifier. The mask is the bitwise inclusive OR of the valid keymask bits. Valid bits are: ShiftMask, LockMask, ControlMask, Mod1Mask, Mod2Mask, Mod3Mask, Mod4Mask, Mod5Mask.



**modifier\_device**

specifies the device whose modifiers are to be used. If the `modifier_device` specified is `NULL`, the X keyboard will be used as the `modifier_device`.

**grab\_window**

Specifies the grab window.

**owner\_events**

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

**event\_count**

Specifies the number of event classes in the event list.

**event\_list**

Specifies which events are reported to the client.

**this\_device\_mode**

Specifies further processing of events from this device. You can pass `GrabModeSync` or `GrabModeAsync`.

**other\_devices\_mode**

Specifies further processing of events from all other devices. You can pass `GrabModeSync` or `GrabModeAsync`.

**DESCRIPTION**

The `XGrabDeviceButton` request establishes a passive grab. In the future, the device is actively grabbed (as for `XGrabDevice`, the last-grab time is set to the time at which the button was pressed (as transmitted in the `DeviceButtonPress` event), and the `DeviceButtonPress` event is reported if all of the following conditions are true:

- \* The device is not grabbed, and the specified button is logically pressed when the specified modifier keys are logically down on the specified modifier device and no other buttons or modifier keys are logically down.
- \* Either the grab window is an ancestor of (or is) the focus



window, OR the grab window is a descendent of the focus window and contains the device.

- \* A passive grab on the same button/modifier combination does not exist on any ancestor of grab\_window.

The interpretation of the remaining arguments is as for XGrabDevice. The active grab is terminated automatically when the logical state of the device has all buttons released (independent of the logical state of the modifier keys).

Note that the logical state of a device (as seen by client applications) may lag the physical state if device event processing is frozen.

This request overrides all previous grabs by the same client on the same button/modifier combinations on the same window. A modifiers of AnyModifier is equivalent to issuing the grab request for all possible modifier combinations (including the combination of no modifiers). It is not required that all modifiers specified have currently assigned KeyCodes. A button of AnyButton is equivalent to issuing the request for all possible buttons. Otherwise, it is not required that the specified button currently be assigned to a physical button.

A modifier\_device of NULL indicates that the X keyboard is to be used as the modifier\_device.

If some other client has already issued a XGrabDeviceButton with the same button/modifier combination on the same window, a BadAccess error results. When using AnyModifier or AnyButton , the request fails completely, and a BadAccess error results (no grabs are established) if there is a conflicting grab for any combination. XGrabDeviceButton has no effect on an active grab.

XGrabDeviceButton can generate BadClass, BadDevice, BadMatch, BadValue, and BadWindow errors.

The XUngrabDeviceButton request releases the passive grab for a button/modifier combination on the specified window if it was grabbed by this client. A modifier of AnyModifier is equivalent to



issuing the ungrab request for all possible modifier combinations, including the combination of no modifiers. A button of AnyButton is equivalent to issuing the request for all possible buttons. XUngrabDeviceButton has no effect on an active grab.

A modifier\_device of NULL indicates that the X keyboard should be used as the modifier\_device.

XUngrabDeviceButton can generate BadDevice, BadMatch, BadValue and BadWindow errors.

## DIAGNOSTICS

### BadDevice

An invalid device was specified. The specified device does not exist or has not been opened by this client via XOpenInputDevice. This error may also occur if the specified device is the X keyboard or X pointer device.

### BadMatch

This error may occur if an XGrabDeviceButton request was made specifying a device that has no buttons, or specifying a modifier device that has no keys.

### 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

XAllowDeviceEvents(3), XGrabDevice(3), XGrabDeviceKey(3)