XGrabDevice, XUngrabDevice – grab/release the specified extension device

int XGrabDevice(*display***,** *device***,** *grab_window***,** *owner_events***,** *event_count***,** *event_list***,**
*this_device_mode***,** *other_devices_mode***,** *time***)**
  Display ****display*;**
  XDevice ****device*;**
  Window *grab_window***;**
  Bool *owner_events***;**
  int *event_count***;**
  XEventClass ****event_list*;**
  int *this_device_mode***,** *other_devices_mode***;**
  Time *time***;**

XUngrabDevice(*display***,** *device***,** *time***)**
  Display ****display*;**
  XDevice ****device*;**
  Time *time***;**

display **Specifies the connection to the X server.** *device* **Specifies the device to be grabbed or released.** *grab_window* **Specifies the id of a window to be associated with the device.** *owner_events* **Specifies a Boolean value that indicates whether the events from the device 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 elements in the event_list array.** *event_list* **Specifies a pointer to a list of event classes that indicates which events the client wishes to receive. These event classes must have been obtained specifying the device being grabbed.** *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 other devices. You can pass** *GrabModeSync* **or** *GrabModeAsync***.** *time* **Specifies the time. You can pass either a timestamp or** *CurrentTime***.**

The *XGrabDevice* **request actively grabs control of the device and generates** *DeviceFocusIn* **and** *DeviceFocusOut* **events. Further device events are reported only to the grabbing client.** *XGrabDevice* **overrides any active device grab by this client. event_list is a pointer to a list of event classes. This list indicates which events the client wishes to receive while the grab is active. If owner_events is** *False* **, all generated device events are reported with respect to grab_window if selected. If owner_events is** *True* **and if a generated device event would normally be reported to this client, it is reported normally; otherwise, the event is reported with respect to the grab_window, and is only reported if specified in the event_list.**

If the this_device_mode argument is *GrabModeAsync* , device event processing continues as usual. If the device is currently frozen by this client, then processing of device events is resumed. If the this_device_mode argument is *GrabModeSync* , the state of the device (as seen by client applications) appears to freeze, and the X server generates no further device events until the grabbing client issues a releasing *XAllowDeviceEvents* call or until the device grab is released. Actual device changes are not lost while the device is frozen; they are simply queued in the server for later processing.

If other_devices_mode is *GrabModeAsync* , processing of events from other devices is unaffected by activation of the grab. If other_devices_mode is *GrabModeSync*, the state of all devices except the grabbed device
(as seen by client applications) appears to freeze, and the X server generates no further events from those devices until the grabbing client issues a releasing *XAllowDeviceEvents* call or until the device grab is released. Actual events are not lost while the devices are frozen; they are simply queued in the server for later processing.

If the device is actively grabbed by some other client, *XGrabDevice* fails and returns *AlreadyGrabbed.* If grab_window is not viewable, it fails and returns *GrabNotViewable*. If the device is frozen by an active grab of another client, it fails and returns *GrabFrozen*. If the specified time is earlier than the last-device-grab time or later than the current X server time, it fails and returns *GrabInvalidTime*. Otherwise, the last-device-grab time is set to the specified time *(CurrentTime* is replaced by the current X server time).

If a grabbed device is closed by a client while an active grab by that client is in effect, the active grab is released. If the device is frozen only by an active grab of the requesting client, it is thawed.

XGrabDevice can generate *BadClass*, *BadDevice*, *BadValue*, and *BadWindow* errors.

The *XUngrabDevice* request releases the device and any queued events if this client has it actively grabbed from either *XGrabDevice* or *XGrabDeviceKey*. If other devices are frozen by the grab, *XUngrabDevice* thaws them. *XUngrabDevice* does not release the device and any queued events if the specified time is earlier than the last-device-grab time or is later than the current X server time. It also generates *DeviceFocusIn* and *DeviceFocusOut* events. The X server automatically performs an *UngrabDevice* request if the event window for an active device grab becomes not viewable.

XUngrabDevice can generate a *BadDevice* error.

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.** *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.**

XAllowDeviceEvents(3X), XGrabDeviceButton(3X), XGrabDeviceKey(3X),
Programming With Xlib