

**NAME**

xcb\_grab\_keyboard - Grab the keyboard

**SYNOPSIS**

```
#include <xcb/xproto.h>
```

**Request function**

```
xcb_grab_keyboard_cookie_t xcb_grab_keyboard(xcb_connection_t *conn, uint8_t owner_events,
      xcb_window_t grab_window, xcb_timestamp_t time, uint8_t pointer_mode,
      uint8_t keyboard_mode);
```

**Reply datastructure**

```
typedef struct xcb_grab_keyboard_reply_t {
    uint8_t response_type;
    uint8_t status;
    uint16_t sequence;
    uint32_t length;
} xcb_grab_keyboard_reply_t;
```

**Reply function**

```
xcb_grab_keyboard_reply_t *xcb_grab_keyboard_reply(xcb_connection_t *conn,
      xcb_grab_keyboard_cookie_t cookie, xcb_generic_error_t **e);
```

**REQUEST ARGUMENTS**

*conn*            The XCB connection to X11.

*owner\_events*   If 1, the *grab\_window* will still get the pointer events. If 0, events are not reported to the *grab\_window*.

*grab\_window*    Specifies the window on which the pointer should be grabbed.

*time*            Timestamp to avoid race conditions when running X over the network.

The special value *XCB\_CURRENT\_TIME* will be replaced with the current server time.

*pointer\_mode*   One of the following values:

*XCB\_GRAB\_MODE\_SYNC*

The state of the keyboard appears to freeze: No further keyboard events are generated by the server until the grabbing client issues a releasing *AllowEvents* request or until the keyboard grab is released.

#### *XCB\_GRAB\_MODE\_ASYNC*

Keyboard event processing continues normally.

#### *keyboard\_mode*

One of the following values:

#### *XCB\_GRAB\_MODE\_SYNC*

The state of the keyboard appears to freeze: No further keyboard events are generated by the server until the grabbing client issues a releasing *AllowEvents* request or until the keyboard grab is released.

#### *XCB\_GRAB\_MODE\_ASYNC*

Keyboard event processing continues normally.

### REPLY FIELDS

*response\_type* The type of this reply, in this case *XCB\_GRAB\_KEYBOARD*. This field is also present in the *xcb\_generic\_reply\_t* and can be used to tell replies apart from each other.

*sequence* The sequence number of the last request processed by the X11 server.

*length* The length of the reply, in words (a word is 4 bytes).

*status* One of the following values:

#### *XCB\_GRAB\_STATUS\_SUCCESS*

TODO: NOT YET DOCUMENTED.

#### *XCB\_GRAB\_STATUS\_ALREADY\_GRABBED*

TODO: NOT YET DOCUMENTED.

#### *XCB\_GRAB\_STATUS\_INVALID\_TIME*

TODO: NOT YET DOCUMENTED.

*XCB\_GRAB\_STATUS\_NOT\_VIEWABLE*

TODO: NOT YET DOCUMENTED.

*XCB\_GRAB\_STATUS\_FROZEN*

TODO: NOT YET DOCUMENTED.

TODO: NOT YET DOCUMENTED.

## DESCRIPTION

Actively grabs control of the keyboard and generates FocusIn and FocusOut events. Further key events are reported only to the grabbing client.

Any active keyboard grab by this client is overridden. If the keyboard is actively grabbed by some other client, *AlreadyGrabbed* is returned. If *grab\_window* is not viewable, *GrabNotViewable* is returned. If the keyboard is frozen by an active grab of another client, *GrabFrozen* is returned. If the specified *time* is earlier than the last-keyboard-grab time or later than the current X server time, *GrabInvalidTime* is returned. Otherwise, the last-keyboard-grab time is set to the specified time.

## RETURN VALUE

Returns an *xcb\_grab\_keyboard\_cookie\_t*. Errors have to be handled when calling the reply function *xcb\_grab\_keyboard\_reply*.

If you want to handle errors in the event loop instead, use *xcb\_grab\_keyboard\_unchecked*. See **xcb-requests(3)** for details.

## ERRORS

*xcb\_value\_error\_t*

TODO: reasons?

*xcb\_window\_error\_t*

The specified *window* does not exist.

## EXAMPLE

```
/*
 * Grabs the keyboard actively
 *
 */
void my_example(xcb_connection_t *conn, xcb_screen_t *screen) {
```

```
xcb_grab_keyboard_cookie_t cookie;
xcb_grab_keyboard_reply_t *reply;

cookie = xcb_grab_keyboard(
    conn,
    true,          /* report events */
    screen->root,   /* grab the root window */
    XCB_CURRENT_TIME,
    XCB_GRAB_MODE_ASYNC, /* process events as normal, do not require sync */
    XCB_GRAB_MODE_ASYNC
);

if ((reply = xcb_grab_keyboard_reply(conn, cookie, NULL))) {
    if (reply->status == XCB_GRAB_STATUS_SUCCESS)
        printf("successfully grabbed the keyboard\n");

    free(reply);
}
}
```

**SEE ALSO**

**xcb-requests(3)**, **xcb-examples(3)**, **xcb\_grab\_pointer(3)**

**AUTHOR**

Generated from xproto.xml. Contact [xcb@lists.freedesktop.org](mailto:xcb@lists.freedesktop.org) for corrections and improvements.