

**NAME**

xcb\_send\_event - send an event

**SYNOPSIS**

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

**Request function**

```
xcb_void_cookie_t xcb_send_event(xcb_connection_t *conn, uint8_t propagate,  
xcb_window_t destination, uint32_t event_mask, const char *event);
```

**REQUEST ARGUMENTS**

*conn*           The XCB connection to X11.

*propagate*       If *propagate* is true and no clients have selected any event on *destination*, the destination is replaced with the closest ancestor of *destination* for which some client has selected a type in *event\_mask* and for which no intervening window has that type in its do-not-propagate-mask. If no such window exists or if the window is an ancestor of the focus window and *InputFocus* was originally specified as the destination, the event is not sent to any clients. Otherwise, the event is reported to every client selecting on the final destination any of the types specified in *event\_mask*.

*destination*     The window to send this event to. Every client which selects any event within *event\_mask* on *destination* will get the event.

The special value *XCB\_SEND\_EVENT\_DEST\_POINTER\_WINDOW* refers to the window that contains the mouse pointer.

The special value *XCB\_SEND\_EVENT\_DEST\_ITEM\_FOCUS* refers to the window which has the keyboard focus.

*event\_mask*      Event\_mask for determining which clients should receive the specified event. See *destination* and *propagate*.

*event*           The event to send to the specified *destination*.

**DESCRIPTION**

Identifies the *destination* window, determines which clients should receive the specified event and ignores any active grabs.

The *event* must be one of the core events or an event defined by an extension, so that the X server can

correctly byte-swap the contents as necessary. The contents of *event* are otherwise unaltered and unchecked except for the *send\_event* field which is forced to 'true'.

## RETURN VALUE

Returns an *xcb\_void\_cookie\_t*. Errors (if any) have to be handled in the event loop.

If you want to handle errors directly with *xcb\_request\_check* instead, use *xcb\_send\_event\_checked*. See **xcb-requests(3)** for details.

## ERRORS

*xcb\_value\_error\_t*

The given *event* is neither a core event nor an event defined by an extension.

*xcb\_window\_error\_t*

The specified *destination* window does not exist.

## EXAMPLE

```

/*
 * Tell the given window that it was configured to a size of 800x600 pixels.
 *
 */
void my_example(xcb_connection_t *conn, xcb_window_t window) {
    /* Every X11 event is 32 bytes long. Therefore, XCB will copy 32 bytes.
     * In order to properly initialize these bytes, we allocate 32 bytes even
     * though we only need less for an xcb_configure_notify_event_t */
    xcb_configure_notify_event_t *event = calloc(32, 1);

    event->event = window;
    event->window = window;
    event->response_type = XCB_CONFIGURE_NOTIFY;

    event->x = 0;
    event->y = 0;
    event->width = 800;
    event->height = 600;

    event->border_width = 0;
    event->above_sibling = XCB_NONE;
    event->override_redirect = false;

```

```
xcb_send_event(conn, false, window, XCB_EVENT_MASK_STRUCTURE_NOTIFY,  
              (char*)event);  
xcb_flush(conn);  
free(event);  
}
```

**SEE ALSO**

[xcb-requests\(3\)](#), [xcb-examples\(3\)](#), [xcb\\_configure\\_notify\\_event\\_t\(3\)](#)

**AUTHOR**

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