

**NAME**

xcb\_randr\_get\_screen\_resources -

**SYNOPSIS**

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

**Request function**

```
xcb_randr_get_screen_resources_cookie_t xcb_randr_get_screen_resources(xcb_connection_t *conn,
    xcb_window_t window);
```

**Reply datastructure**

```
typedef struct xcb_randr_get_screen_resources_reply_t {
    uint8_t    response_type;
    uint8_t    pad0;
    uint16_t   sequence;
    uint32_t   length;
    xcb_timestamp_t timestamp;
    xcb_timestamp_t config_timestamp;
    uint16_t   num_crtcs;
    uint16_t   num_outputs;
    uint16_t   num_modes;
    uint16_t   names_len;
    uint8_t    pad1[8];
} xcb_randr_get_screen_resources_reply_t;
```

**Reply function**

```
xcb_randr_get_screen_resources_reply_t
    *xcb_randr_get_screen_resources_reply(xcb_connection_t *conn,
    xcb_randr_get_screen_resources_cookie_t cookie, xcb_generic_error_t **e);
```

**Reply accessors**

```
xcb_randr_crtc_t *xcb_randr_get_screen_resources_crtcs(const
    xcb_randr_get_screen_resources_request_t *reply);
```

```
int xcb_randr_get_screen_resources_crtcs_length(const xcb_randr_get_screen_resources_reply_t
    *reply);
```

```
xcb_generic_iterator_t xcb_randr_get_screen_resources_crtcs_end(const
    xcb_randr_get_screen_resources_reply_t *reply);
```

```

xcb_randr_output_t *xcb_randr_get_screen_resources_outputs(const
    xcb_randr_get_screen_resources_request_t *reply);

int xcb_randr_get_screen_resources_outputs_length(const xcb_randr_get_screen_resources_reply_t
    *reply);

xcb_generic_iterator_t xcb_randr_get_screen_resources_outputs_end(const
    xcb_randr_get_screen_resources_reply_t *reply);

xcb_randr_mode_info_t *xcb_randr_get_screen_resources_modes(const
    xcb_randr_get_screen_resources_request_t *reply);

int xcb_randr_get_screen_resources_modes_length(const xcb_randr_get_screen_resources_reply_t
    *reply);

xcb_randr_mode_info_iterator_t xcb_randr_get_screen_resources_modes_iterator(const
    xcb_randr_get_screen_resources_reply_t *reply);

uint8_t *xcb_randr_get_screen_resources_names(const xcb_randr_get_screen_resources_request_t
    *reply);

int xcb_randr_get_screen_resources_names_length(const xcb_randr_get_screen_resources_reply_t
    *reply);

xcb_generic_iterator_t xcb_randr_get_screen_resources_names_end(const
    xcb_randr_get_screen_resources_reply_t *reply);

```

## REQUEST ARGUMENTS

*conn*            The XCB connection to X11.

*window*            TODO: NOT YET DOCUMENTED.

## REPLY FIELDS

*response\_type*    The type of this reply, in this case *XCB\_RANDR\_GET\_SCREEN\_RESOURCES*. 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).

*timestamp*      TODO: NOT YET DOCUMENTED.

*config\_timestamp*  
                  TODO: NOT YET DOCUMENTED.

*num\_crtcs*      TODO: NOT YET DOCUMENTED.

*num\_outputs*    TODO: NOT YET DOCUMENTED.

*num\_modes*     TODO: NOT YET DOCUMENTED.

*names\_len*      TODO: NOT YET DOCUMENTED.

## DESCRIPTION

### RETURN VALUE

Returns an *xcb\_randr\_get\_screen\_resources\_cookie\_t*. Errors have to be handled when calling the reply function *xcb\_randr\_get\_screen\_resources\_reply*.

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

### ERRORS

This request does never generate any errors.

### SEE ALSO

### AUTHOR

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