

**NAME**

xcb\_randr\_get\_provider\_info -

**SYNOPSIS**

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

**Request function**

```
xcb_randr_get_provider_info_cookie_t xcb_randr_get_provider_info(xcb_connection_t *conn,
    xcb_randr_provider_t provider, xcb_timestamp_t config_timestamp);
```

**Reply datastructure**

```
typedef struct xcb_randr_get_provider_info_reply_t {
    uint8_t    response_type;
    uint8_t    status;
    uint16_t   sequence;
    uint32_t   length;
    xcb_timestamp_t timestamp;
    uint32_t   capabilities;
    uint16_t   num_crtcs;
    uint16_t   num_outputs;
    uint16_t   num_associated_providers;
    uint16_t   name_len;
    uint8_t    pad0[8];
} xcb_randr_get_provider_info_reply_t;
```

**Reply function**

```
xcb_randr_get_provider_info_reply_t *xcb_randr_get_provider_info_reply(xcb_connection_t *conn,
    xcb_randr_get_provider_info_cookie_t cookie, xcb_generic_error_t **e);
```

**Reply accessors**

```
xcb_randr_crtc_t *xcb_randr_get_provider_info_crtcs(const xcb_randr_get_provider_info_request_t
    *reply);
```

```
int xcb_randr_get_provider_info_crtcs_length(const xcb_randr_get_provider_info_reply_t *reply);
```

```
xcb_generic_iterator_t xcb_randr_get_provider_info_crtcs_end(const
    xcb_randr_get_provider_info_reply_t *reply);
```

```
xcb_randr_output_t *xcb_randr_get_provider_info_outputs(const
```

```

xcb_randr_get_provider_info_request_t *reply);

int xcb_randr_get_provider_info_outputs_length(const xcb_randr_get_provider_info_reply_t *reply);

xcb_generic_iterator_t xcb_randr_get_provider_info_outputs_end(const
xcb_randr_get_provider_info_reply_t *reply);

xcb_randr_provider_t *xcb_randr_get_provider_info_associated_providers(const
xcb_randr_get_provider_info_request_t *reply);

int xcb_randr_get_provider_info_associated_providers_length(const
xcb_randr_get_provider_info_reply_t *reply);

xcb_generic_iterator_t xcb_randr_get_provider_info_associated_providers_end(const
xcb_randr_get_provider_info_reply_t *reply);

uint32_t *xcb_randr_get_provider_info_associated_capability(const
xcb_randr_get_provider_info_request_t *reply);

int xcb_randr_get_provider_info_associated_capability_length(const
xcb_randr_get_provider_info_reply_t *reply);

xcb_generic_iterator_t xcb_randr_get_provider_info_associated_capability_end(const
xcb_randr_get_provider_info_reply_t *reply);

char *xcb_randr_get_provider_info_name(const xcb_randr_get_provider_info_request_t *reply);

int xcb_randr_get_provider_info_name_length(const xcb_randr_get_provider_info_reply_t *reply);

xcb_generic_iterator_t xcb_randr_get_provider_info_name_end(const
xcb_randr_get_provider_info_reply_t *reply);

```

## REQUEST ARGUMENTS

*conn*           The XCB connection to X11.

*provider*       TODO: NOT YET DOCUMENTED.

*config\_timestamp*  
                  TODO: NOT YET DOCUMENTED.

**REPLY FIELDS**

<i>response_type</i>	The type of this reply, in this case <i>XCB_RANDR_GET_PROVIDER_INFO</i> . This field is also present in the <i>xcb_generic_reply_t</i> and can be used to tell replies apart from each other.
<i>sequence</i>	The sequence number of the last request processed by the X11 server.
<i>length</i>	The length of the reply, in words (a word is 4 bytes).
<i>status</i>	TODO: NOT YET DOCUMENTED.
<i>timestamp</i>	TODO: NOT YET DOCUMENTED.
<i>capabilities</i>	TODO: NOT YET DOCUMENTED.
<i>num_crtcs</i>	TODO: NOT YET DOCUMENTED.
<i>num_outputs</i>	TODO: NOT YET DOCUMENTED.
<i>num_associated_providers</i>	TODO: NOT YET DOCUMENTED.
<i>name_len</i>	TODO: NOT YET DOCUMENTED.

**DESCRIPTION****RETURN VALUE**

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

If you want to handle errors in the event loop instead, use *xcb\_randr\_get\_provider\_info\_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.