

**NAME**

XIAllowEvents - Release queued events

**SYNOPSIS**

```
#include <X11/extensions/XInput2.h>
```

```
Status XIAllowEvents( Display *display,  
                      int deviceid,  
                      int event_mode,  
                      Time time );
```

```
Status XIAllowTouchEvents( Display *dpy,  
                           int deviceid,  
                           unsigned int touchid,  
                           Window grab_window,  
                           int event_mode );
```

display

Specifies the connection to the X server.

device

Specifies the device that is to be grabbed or released

event\_mode

Specifies whether a device is to be thawed and events are to be replayed, or how to handle a grabbed touch sequence.

time

A valid server time or CurrentTime.

touchid

The ID of the touch sequence to accept or reject. The value is ignored for event modes other than AcceptTouch and RejectTouch.

grab\_window

The window on which to accept or reject a touch sequence grab. The value is ignored for event modes other than AcceptTouch and RejectTouch.

**DESCRIPTION**

The XIAllowEvents request releases some queued events if the client

has caused a device to freeze. It also is used to handle touch grab and ownership processing.

The function has no effect if the specified time is earlier than the last-grab time of the most recent active grab for the client, or if the specified time is later than the current X server time. The time parameter must be `CurrentTime` for requests with event modes of `XIAcceptTouch` and `XIRejectTouch`.

The following describes the processing that occurs depending on what constant you pass to the `event_mode` argument:

#### `XIAsyncDevice`:

If the specified device is frozen by the client, event processing for that device continues as usual. If the device is frozen multiple times by the client on behalf of multiple separate grabs, `XIAsyncDevice` thaws for all. `XIAsyncDevice` has no effect if the specified device is not frozen by the client, but the device need not be grabbed by the client.

#### `XISyncDevice`:

If the specified device is frozen and actively grabbed by the client, event processing for that device continues normally until the next button press or release, or key press or release, or a gesture begin or end event (depending on the grab) is reported to the client.

At this time, the specified device again appears to freeze.

However, if the reported event causes the grab to be released, the specified device does not freeze.

`XISyncDevice` has no effect if the specified device is not frozen by the client or is not grabbed by the client.

#### `XIReplayDevice`:

If the specified device is actively grabbed by the client and is frozen as the result of an event having been sent to the client (either from the activation of a `XIGrabButton` or from a previous `XIAllowEvents` with mode `SyncDevice`, but not from a `Grab`), the grab is released and that event is completely reprocessed. This time, however, the request ignores any passive grabs at or above (towards the root) the grab window of the grab just released.

The request has no effect if the specified device is not grabbed by the client or if it is not frozen as the result of an event.

In case of gesture begin event being replayed, the original grabbing

client will receive a `XI_GesturePinchEnd` or `XI_GestureSwipeEnd` event.

#### `XIAsyncPairedDevice`:

If the paired master device is frozen by the client, event processing for it continues as usual. If the paired device is frozen multiple times by the client on behalf of multiple separate grabs,

`XIAsyncPairedDevice` thaws for all.

`XIAsyncPairedDevice` has no effect if the device is not frozen by the client, but those devices need not be grabbed by the client.

`XIAsyncPairedDevice` has no effect if `deviceid` specifies a slave device.

#### `XISyncPair`:

If both the device and the paired master device are frozen by the client, event processing (for both devices) continues normally until the next `XI_ButtonPress`, `XI_ButtonRelease`, `XI_KeyPress`, or `XI_KeyRelease` event is reported to the client for a grabbed device (button event for a pointer, key event for a keyboard), at which time the devices again appear to freeze. However, if the reported event causes the grab to be released, then the devices do not freeze (but if the other device is still grabbed, then a subsequent event for it will still cause both devices to freeze).

`XISyncPair` has no effect unless both the device and the paired master device are frozen by the client. If the device or paired master device is frozen twice by the client on behalf of two separate grabs, `XISyncPair` thaws for both (but a subsequent freeze for `XISyncPair` will only freeze each device once).

`XISyncPair` has no effect if `deviceid` specifies a slave device.

#### `XIAsyncPair`:

If the device and the paired master device are frozen by the client, event processing for both devices continues normally. If a device is frozen twice by the client on behalf of two separate grabs, `AsyncBoth` thaws for both. `XIAsyncPair` has no effect unless both the device and the paired master device are frozen by the client.

`XIAsyncPair` has no effect if `deviceid` specifies a slave device.

#### `XIAcceptTouch`:

The client is deemed to have taken control of the touch sequence once it owns the sequence. `TouchEnd` events will be sent to all clients listening to the touch sequence that have either grabbed the touch sequence on a

child window of the grab\_window or have received events for the touch sequence through event selection. These clients will no longer receive any TouchUpdate events.

**XIRejectTouch:**

The client is no longer interested in the touch sequence, and will receive a XI\_TouchEnd event. If the client is the current owner of the sequence, ownership will be passed on to the next listener.

**DIAGNOSTICS****BadDevice**

An invalid deviceid was specified.

**BadAccess**

This error may occur if event\_mode is XIAcceptTouch and this client is not the current or potential owner of the specified touch ID.

**BadValue**

This error may occur if event\_mode is XIAcceptTouch and touch ID is invalid.

**BadWindow**

A value for a grab\_window argument does not name a defined Window.

**SEE ALSO**

XIGrabButton(3)