XSetDeviceFocus, XGetDeviceFocus − control extension input device focus

XSetDeviceFocus(*display*, *device*, *focus*, *revert_to*, *time***)**
    Display ******display*;
    Display ******device*;
    Window *focus*;
    int *revert_to*;
    Time *time*;

XGetDeviceFocus(*display*, *device*, *focus_return*, *revert_to_return*, *time_return***)**
    Display ******display*;
    Display ******device*;
    Window ******focus_return*;
    int ******revert_to_return*;
    int ******time_return*;

display **Specifies the connection to the X server.** *device* **Specifies the device whose focus is to be queried or changed.** *focus* **Specifies the window,** *PointerRoot*, *FollowKeyboard*, **or** *None*. *focus_return* **Returns the focus window,** *PointerRoot*, *FollowKeyboard*, **or** *None*. *revert_to* **Specifies where the input focus reverts to if the window becomes not viewable. You can pass** *RevertToParent*, *RevertToPointerRoot*, *RevertToFollowKeyboard*, **or** *RevertToNone*. *revert_to_return* **Returns the current focus state** *RevertToParent*, *RevertToPointerRoot*, *RevertToFollowKeyboard*, **or** *RevertToNone*. *time_return* **Returns the last_focus_time for the device.** *time* **Specifies the time. You can pass either a timestamp or** *CurrentTime*.

The *XSetDeviceFocus* **request changes the focus of the specified device and its last-focus-change time. It has no effect if the specified time is earlier than the current last-focus-change time or is later than the current X server time. Otherwise, the last-focus-change time is set to the specified time** *CurrentTime* **is replaced by the current X server time).** *XSetDeviceFocus* **causes the X server to generate** *DeviceFocusIn* **and** *DeviceFocusOut* **events.**

Depending on the focus argument, the following occurs:

- If focus is *None* , all device events are discarded until a new focus window is set, and the revert_to argument is ignored.
- If focus is a window, it becomes the device’s focus window. If a generated device event would normally be reported to this window or one of its inferiors, the event is reported as usual. Otherwise, the event is reported relative to the focus window.
- If focus is *PointerRoot*, the focus window is dynamically taken to be the root window of whatever screen the pointer is on at each event from the specified device. In this case, the revert_to argument is ignored.
- If focus is *FollowKeyboard*, the focus window is dynamically taken to be the window to which the X keyboard focus is set at each input event.

The specified focus window must be viewable at the time *XSetDeviceFocus* is called, or a *BadMatch* error results. If the focus window later becomes not viewable, the X server evaluates the revert_to argument to determine the new focus window as follows:

- If revert_to is *RevertToParent*, the focus reverts to the parent (or the closest viewable ancestor), and the new revert_to value is taken to be *RevertToNone*.
- If revert_to is *RevertToPointerRoot*, *RevertToFollowKeyboard*, or *RevertToNone*, the focus reverts to *PointerRoot*, *FollowKeyboard*, or *None*, respectively.

When the focus reverts, the X server generates *DeviceFocusIn* and *DeviceFocusOut* events, but the last-focus-change time is not affected.

Input extension devices are not required to support the ability to be focused. Attempting to set the focus of a device that does not support this request will result in a *BadMatch* error. Whether or not given device can support this request can be determined by the information returned by *XOpenDevice*. For those devices that support focus, *XOpenDevice* will return an *XInputClassInfo* structure with the input_class field equal to the constant *FocusClass* (defined in the file *XI.h*).

XSetDeviceFocus can generate *BadDevice*, *BadMatch*, *BadValue*, and *BadWindow* errors.

The *XGetDeviceFocus* request returns the focus window and the current focus state.

Not all input extension devices can be focused. Attempting to query the focus state of a device that can't be focused results in a *BadMatch* error. A device that can be focused returns information for input Class Focus when an *XOpenDevice* request is made.

XGetDeviceFocus can generate *BadDevice*, and *BadMatch* errors.

BadDevice **An invalid device was specified. The specified device does not exist or has not been opened by this client via** *XOpenInputDevice*. **This error may also occur if the specified device is the X keyboard or X pointer device.** *BadValue* **Some numeric value falls outside the range of values accepted by the request. Unless a specific range is specified for an argument, the full range defined by the argument's type is accepted. Any argument defined as a set of alternatives can generate this error.** *BadWindow* **A value for a Window argument does not name a defined Window.** *BadMatch* **This error may occur if an** *XGetDeviceFocus* **or** *XSetDeviceFocus* **request was made specifying a device that the server implementation does not allow to be focused.**

Programming with Xlib