NAME

XListInputDevices, XFreeDeviceList - list available input devices

SYNOPSIS

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

XDeviceInfo *XListInputDevices( Display *display,
                    int *ndevices_return);

int XFreeDeviceList( XDeviceInfo *list);
```

display
: Specifies the connection to the X server.

ndevices_return
: Specifies a pointer to a variable where the number of available devices can be returned.

list
: Specifies the list of devices to free. The XFreeDeviceList function frees the list of available extension input devices.

DESCRIPTION

The XListInputDevices request lists the available input devices. This list includes the core keyboard and any physical input device currently accessible through the X server, and any input devices that are not currently accessible through the X server but could be accessed if requested.

A master pointer is a virtual pointer device that does not represent a physical device. It is visually represented through a cursor. A master keyboard is a virtual keyboard device that does not represent a physical device. It is virtually represented through a keyboard focus. A master pointer and a master keyboard are always paired (i.e. if shift is pressed on the master keyboard, a pointer click would be a shift-click). Multiple master pointer/keyboard pairs can exist.

X servers supporting the X Input Extension version 2,

XListInputDevices only returns the first master pointer, the first master keyboard and all slave devices. Additional master devices are not listed.

Physical devices (so-called slave devices) are attached to either a master pointer or a master keyboard, depending on their capabilities. If a slave device generates an event, the event is also generated by the respective master device. Multiple slave devices can be attached to a single master device.

Some server implementations may make all physical input devices available at the time the server is initialized. Others may wait until requested by a client to access an input device. In the latter case, it is possible that an input device will be listed as available at one time but not at another.

For each input device available to the server, the XListInputDevices request returns an XDeviceInfo structure. That structure contains a pointer to a list of structures, each of which contains information about one class of input supported by the device. The XDeviceInfo structure is defined as follows:

```
typedef struct _XDeviceInfo {
XID     id;
Atom    type;
char    *name;
int         num_classes;
int         use;
XAnyClassPtr inputclassinfo;
} XDeviceInfo;
```

The id is a number in the range 0-128 that uniquely identifies the device. It is assigned to the device when it is initialized by the server.

The type field is of type Atom and indicates the nature of the device. The type will correspond to one of the following strings (defined in the header file XI.h):

XI_MOUSE XI_TABLET XI_KEYBOARD XI_TOUCHSCREEN XI_TOUCHPAD XI_BUTTONBOX XI_BARCODE XI_TRACKBALL XI_QUADRATURE XI_ID_MODULE XI_ONE_KNOB XI_NINE_KNOB XI_KNOB_BOX XI_SPACEBALL XI_DATAGLOVE XI_EYETRACKER XI_CURSORKEYS XI_FOOTMOUSE XI_JOYSTICK

These strings may be used in an XInternAtom request to return an atom that can be compared with the type field of the XDeviceInfo structure.

The name field contains a pointer to a null-terminated string that serves as identifier of the device. This identifier may be user-configured or automatically assigned by the server.

The num_classes field is a number in the range 0-255 that specifies the number of input classes supported by the device for which information is returned by ListInputDevices. Some input classes, such as class Focus and class Proximity do not have any information to be returned by ListInputDevices.

All devices provide an AttachClass. This class specifies the master device a given slave device is attached to. For master devices, the class specifies the respective paired master device.

The use field specifies how the device is currently being used. If the value is IsXKeyboard, the device is a master keyboard. If the value is IsXPointer, the device is a master pointer. If the value is IsXExtensionPointer, the device is a slave pointer. If the value is IsXExtensionKeyboard, the device is a slave keyboard. If the value is IsXExtensionDevice, the device is available for use as an extension device.

The inputclassinfo field contains a pointer to the first input-class specific data. The first two fields are common to all classes.

The class field is a number in the range 0-255. It uniquely identifies the class of input for which information is returned. Currently defined classes are KeyClass, ButtonClass, and ValuatorClass.

The length field is a number in the range 0- 255. It specifies the number of bytes of data that are contained in this input class. The length includes the class and length fields.

The XKeyInfo structure describes the characteristics of the keys on the device. It is defined as follows:

```
typedef struct _XKeyInfo {
XID class;
int length;
unsigned short min_keycode;
unsigned short max_keycode;
unsigned short num_keys;
} XKeyInfo;
```

min_keycode is of type KEYCODE. It specifies the minimum keycode that the device will report. The minimum keycode will not be smaller than 8.

max_keycode is of type KEYCODE. It specifies the maximum keycode that the device will report. The maximum keycode will not be larger than 255.

num_keys specifies the number of keys that the device has.

The XButtonInfo structure defines the characteristics of the buttons on the device. It is defined as follows:

```
typedef struct _XButtonInfo {
XID class;
int length;
short num_buttons;
} XButtonInfo;
```

num_buttons specifies the number of buttons that the device has.

The XValuatorInfo structure defines the characteristics of the valuators on the device. It is defined as follows:

```
typedef struct  _XValuatorInfo {
XID class;
int length;
unsigned char num_axes;
unsigned char mode;
unsigned long motion_buffer;
XAxisInfoPtr axes;
} XValuatorInfo;
```

num_axes contains the number of axes the device supports.

mode is a constant that has one of the following values:
Absolute or Relative. Some devices allow the mode to be changed dynamically via the SetDeviceMode request.

motion_buffer_size is a cardinal number that specifies the number of elements that can be contained in the motion history buffer for the device.

The axes field contains a pointer to an XAxisInfo structure.

The XAxisInfo structure is defined as follows:

```
typedef struct _XAxisInfo {
int resolution;
int min_value;
int max_value;
} XAxisInfo;
```

The resolution contains a number in counts/meter.

The min_val field contains a number that specifies the minimum value the device reports for this axis. For devices whose mode is Relative, the min_val field will contain 0.

The max_val field contains a number that specifies the maximum value the device reports for this axis. For devices whose mode is Relative, the max_val field will contain 0.

The XAttachInfo structure is defined as follows:

```
typedef struct _XAttachInfo {
int     attached;
} XAttachInfo;
```

If the device is a slave device, attached specifies the device ID of the master device this device is attached to. If the device is not attached to a master device, attached is Floating. If the device is a master device, attached specifies the device ID of the master device this device is paired with.

RETURN VALUE

XListInputDevices returns a pointer to an array of XDeviceInfo structs and sets ndevices_return to the number of elements in that array. To free the XDeviceInfo array created by XListInputDevices, use XFreeDeviceList.

On error, XListInputDevices returns NULL and ndevices_return is left unmodified.