NAME

XkbSetDeviceInfo - Modify some or all of the characteristics of an X Input Extension device

SYNOPSIS

Bool XkbSetDeviceInfo (Display ****dpy*, unsigned int** *which***, XkbDeviceInfoPtr** *device_info***);**

ARGUMENTS

dpy connection to X server

which

mask indicating characteristics to modify

device_info

structure defining the device and modifications

DESCRIPTION

To change characteristics of an X Input Extension device in the server, first modify a local copy of the device structure and then use either *XkbSetDeviceInfo*, or, to save network traffic, use an XkbDeviceChangesRec structure and call *XkbChangeDeviceInfo* to download the changes to the server.

XkbSetDeviceInfo sends a request to the server to modify the characteristics of the device specified in the *device_info* structure. The particular characteristics modified are identified by the bits set in *which* and take their values from the relevant fields in device_info (see Table 1). XkbSetDeviceInfo returns True if the request was successfully sent to the server. If the X server implementation does not allow interaction between the X input extension and the Xkb Extension, the function does nothing and returns False.

Table 1 XkbDeviceInfoRec Mask Bits

Name	XkbDeviceInfoRec Value Fields Effected	Capability If Set
XkbXI_KeyboardsMask		(1L <<0) Clients can use all Xkb requests and events with KeyClass devices supported by the input device extension.
XkbXI_ButtonActionsMask	num_btns btn_acts	(1L <<1) Clients can assign key actions to buttons

non-KeyClass input
extension devices.

XkbXI_IndicatorNamesMask leds->names (1L <<2) Clients can assign
names to indicators on
non-KeyClass input
extension devices.

XkbXI_IndicatorMapsMask leds->maps (1L <<3) Clients can assign
indicator maps to
indicators on
non-KeyClass input
extension devices.

XkbXI_IndicatorStateMask leds->state (1L <<4) Clients can request
the status of indicators
on non-KeyClass input
extension devices.

XkbXI_IndicatorsMask sz_leds (0x1c) XkbXI_IndicatorNamesMask |
num_leds XkbXI_IndicatorMapsMask |
leds->* XkbXI_IndicatorStateMask

XkbXI_UnsupportedFeaturesMask unsupported (1L <<15)

XkbXI_AllDeviceFeaturesMask Those selected (0x1e) XkbXI_IndicatorsMask |
by Value Column XkbSI_ButtonActionsMask
masks

XkbXI_AllFeaturesMask Those selected (0x1f) XkbSI_AllDeviceFeaturesMask |
by Value Column XkbSI_KeyboardsMask
masks

XkbXI_AllDetailsMask Those selected (0x801f) XkbXI_AllFeaturesMask |
by Value column XkbXI_UnsupportedFeaturesMask
masks

The *which* parameter specifies which aspects of the device should be changed and is a bitmask composed of an inclusive OR or one or more of the following bits: XkbXI_ButtonActionsMask, XkbXI_IndicatorNamesMask, XkbXI_IndicatorMapsMask. If the features requested to be manipulated

in *which* are valid for the device, but the server does not support assignment of one or more of them, that particular portion of the request is ignored.

If the device specified in *device_info->device_spec* does not contain buttons and a request affecting buttons is made, or the device does not contain indicators and a request affecting indicators is made, a BadMatch protocol error results.

If the XkbXI_ButtonActionsMask bit is set in the *supported* mask returned by *XkbGetDeviceInfo,* the Xkb extension allows applications to assign key actions to buttons on input extension devices other than the core keyboard device. If the XkbXI_ButtonActionsMask is set in *which,* the actions for all buttons specified in *device_info* are set to the XkbActions specified in *device_info->btn_acts.* If the number of buttons requested to be updated is not valid for the device, *XkbSetDeviceInfo* returns False and a BadValue protocol error results.

If the XkbXI_IndicatorMaps and / or XkbXI_IndicatorNamesMask bit is set in the *supported* mask returned by *XkbGetDeviceInfo,* the Xkb extension allows applications to assign maps and / or names to the indicators of nonkeyboard extension devices. If supported, maps and / or names can be assigned to all extension device indicators, whether they are part of a keyboard feedback or part of an indicator feedback.

If the XkbXI_IndicatorMapsMask and / or XkbXI_IndicatorNamesMask flag is set in *which,* the indicator maps and / or names for all *device_info->num_leds* indicator devices specified in *device_info->leds* are set to the maps and / or names specified in *device_info->leds.* *device_info->leds->led_class* and *led_id* specify the input extension class and device ID for each indicator device to modify; if they have invalid values, a BadValue protocol error results and *XkbSetDeviceInfo* returns False. If they have legal values but do not specify a keyboard or indicator class feedback for the device in question, a BadMatch error results. If any of the values in *device_info->leds->names* are not a valid Atom or None, a BadAtom protocol error results.

RETURN VALUES

True — The *XkbSetDeviceInfo* function returns True if the request was successfully sent to the server.

False — The *XkbSetDeviceInfo* function returns False if the X server implementation does not allow interaction between the X input extension and the Xkb Extension.

STRUCTURES

Changes to an Xkb extension device may be tracked by listening to XkbDeviceExtensionNotify events and accumulating the changes in an XkbDeviceChangesRec structure. The changes noted in the structure may then be used in subsequent operations to update either a server configuration or a local

copy of an Xkb extension device configuration. The changes structure is defined as follows:

```
typedef struct _XkbDeviceChanges {
   unsigned int    changed;    /* bits indicating what has changed */
   unsigned short  first_btn;  /* number of first button which changed, if any */
   unsigned short  num_btns;   /* number of buttons that have changed */
   XkbDeviceLedChangesRec leds;
} XkbDeviceChangesRec,*XkbDeviceChangesPtr;
```

DIAGNOSTICS

BadAtom A name is neither a valid Atom or None

BadMatch A compatible version of Xkb was not available in the server or an argument has correct type and range, but is otherwise invalid

BadValue An argument is out of range

SEE ALSO

XkbChangeDeviceInfo(3), **XkbGetDeviceInfo**(3)