

**NAME**

XkbAllocDeviceInfo - Obtain an XkbDeviceInfoRec structure

**SYNOPSIS**

```
XkbDeviceInfoPtr XkbAllocDeviceInfo (unsigned int device_spec, unsigned int n_buttons, unsigned int sz_leds);
```

**ARGUMENTS**

*device\_spec*

device ID with which structure will be used

*n\_buttons*

number of button actions to allocate space for

*sz\_leds*

number of LED feedbacks to allocate space for

**DESCRIPTION**

*XkbAllocDeviceInfo* allocates space for an XkbDeviceInfoRec structure and initializes that structure's *device\_spec* field with the device ID specified by *device\_spec*. If *n\_buttons* is nonzero, *n\_buttons* XkbActions are linked into the XkbDeviceInfoRec structure and initialized to zero. If *sz\_leds* is nonzero, *sz\_leds* XkbDeviceLedInfoRec structures are also allocated and linked into the XkbDeviceInfoRec structure. If you request XkbDeviceLedInfoRec structures be allocated using this request, you must initialize them explicitly, by using **XkbAddDeviceLedInfo(3)**.

**STRUCTURES**

Information about X Input Extension devices is transferred between a client program and the Xkb extension in an XkbDeviceInfoRec structure:

```
typedef struct {
    char *      name;      /* name for device */
    Atom       type;      /* name for class of devices */
    unsigned short  device_spec; /* device of interest */
    Bool       has_own_state; /* True=>this device has its own state */
    unsigned short  supported; /* bits indicating supported capabilities */
    unsigned short  unsupported; /* bits indicating unsupported capabilities */
    unsigned short  num_btns; /* number of entries in btn_acts */
    XkbAction *   btn_acts; /* button actions */
    unsigned short  sz_leds; /* total number of entries in LEDs vector */
    unsigned short  num_leds; /* number of valid entries in LEDs vector */
}
```

```
    unsigned short    dflt_kbd_fb; /* input extension ID of default (core kbd) indicator */
    unsigned short    dflt_led_fb; /* input extension ID of default indicator feedback */
    XkbDeviceLedInfoPtr leds;      /* LED descriptions */
} XkbDeviceInfoRec, *XkbDeviceInfoPtr;
```

```
typedef struct {
    unsigned short    led_class;    /* class for this LED device*/
    unsigned short    led_id;       /* ID for this LED device */
    unsigned int      phys_indicators; /* bits for which LEDs physically present */
    unsigned int      maps_present;  /* bits for which LEDs have maps in maps */
    unsigned int      names_present; /* bits for which LEDs are in names */
    unsigned int      state;         /* 1 bit => corresponding LED is on */
    Atom              names[XkbNumIndicators]; /* names for LEDs */
    XkbIndicatorMapRec maps;         /* indicator maps for each LED */
} XkbDeviceLedInfoRec, *XkbDeviceLedInfoPtr;
```

**SEE ALSO****XkbAddDeviceLedInfo(3)**