NAME

XChangeKeyboardMapping, XGetKeyboardMapping, XDisplayKeycodes, XSetModifierMapping, XGetModifierMapping, XNewModifiermap, XInsertModifiermapEntry, XDeleteModifiermapEntry, XFreeModifiermap, XModifierKeymap - manipulate keyboard encoding and keyboard encoding structure

SYNTAX

int XChangeKeyboardMapping(Display **display*, int *first_keycode*, int *keysyms_per_keycode*, KeySym **keysyms*, int *num_codes*);

KeySym *XGetKeyboardMapping(Display **display*, KeyCode *first_keycode*, int *keycode_count*, int **keysyms_per_keycode_return*);

int XDisplayKeycodes(Display **display*, int **min_keycodes_return*, int **max_keycodes_return*);

int XSetModifierMapping(Display **display*, XModifierKeymap **modmap*);

XModifierKeymap *XGetModifierMapping(Display **display*);

XModifierKeymap *XNewModifiermap(int *max_keys_per_mod*);

XModifierKeymap *XInsertModifiermapEntry(XModifierKeymap **modmap*, KeyCode *keycode_entry*, int *modifier*);

XModifierKeymap *XDeleteModifiermapEntry(XModifierKeymap **modmap*, KeyCode *keycode_entry*, int *modifier*);

int XFreeModifiermap(XModifierKeymap **modmap*);

ARGUMENTS

display Specifies the connection to the X server.

first_keycode Specifies the first KeyCode that is to be changed or returned.

keycode_count Specifies the number of KeyCodes that are to be returned.

keycode_entry Specifies the KeyCode.

keysyms Specifies an array of KeySyms.

keysyms_per_keycode
Specifies the number of KeySyms per KeyCode.

keysyms_per_keycode_return
Returns the number of KeySyms per KeyCode.

max_keys_per_mod
Specifies the number of KeyCode entries preallocated to the modifiers in the map.

max_keycodes_return
Returns the maximum number of KeyCodes.

min_keycodes_return
Returns the minimum number of KeyCodes.

modifier Specifies the modifier.

modmap Specifies the **XModifierKeymap** structure.

num_codes Specifies the number of KeyCodes that are to be changed.

DESCRIPTION

The **XChangeKeyboardMapping** function defines the symbols for the specified number of KeyCodes starting with first_keycode. The symbols for KeyCodes outside this range remain unchanged. The number of elements in keysyms must be:

num_codes * keysyms_per_keycode

The specified first_keycode must be greater than or equal to min_keycode returned by **XDisplayKeycodes**, or a **BadValue** error results. In addition, the following expression must be less than or equal to max_keycode as returned by **XDisplayKeycodes**, or a **BadValue** error results:

first_keycode + num_codes - 1

KeySym number N, counting from zero, for KeyCode K has the following index in keysyms, counting from zero:

(K - first_keycode) * keysyms_per_keycode + N

The specified keysyms_per_keycode can be chosen arbitrarily by the client to be large enough to hold

all desired symbols. A special KeySym value of **NoSymbol** should be used to fill in unused elements for individual KeyCodes. It is legal for **NoSymbol** to appear in nontrailing positions of the effective list for a KeyCode. **XChangeKeyboardMapping** generates a **MappingNotify** event.

There is no requirement that the X server interpret this mapping. It is merely stored for reading and writing by clients.

XChangeKeyboardMapping can generate **BadAlloc** and **BadValue** errors.

The **XGetKeyboardMapping** function returns the symbols for the specified number of KeyCodes starting with first_keycode. The value specified in first_keycode must be greater than or equal to min_keycode as returned by **XDisplayKeycodes**, or a **BadValue** error results. In addition, the following expression must be less than or equal to max_keycode as returned by **XDisplayKeycodes**:

first_keycode + keycode_count - 1

If this is not the case, a **BadValue** error results. The number of elements in the KeySyms list is:

keycode_count * keysyms_per_keycode_return

KeySym number N, counting from zero, for KeyCode K has the following index in the list, counting from zero:
(K - first_code) * keysyms_per_code_return + N

The X server arbitrarily chooses the keysyms_per_keycode_return value to be large enough to report all requested symbols. A special KeySym value of **NoSymbol** is used to fill in unused elements for individual KeyCodes. To free the storage returned by **XGetKeyboardMapping**, use **XFree**.

XGetKeyboardMapping can generate a **BadValue** error.

The **XDisplayKeycodes** function returns the min-keycodes and max-keycodes supported by the specified display. The minimum number of KeyCodes returned is never less than 8, and the maximum number of KeyCodes returned is never greater than 255. Not all KeyCodes in this range are required to have corresponding keys.

The **XSetModifierMapping** function specifies the KeyCodes of the keys (if any) that are to be used as modifiers. If it succeeds, the X server generates a **MappingNotify** event, and **XSetModifierMapping** returns **MappingSuccess**. X permits at most 8 modifier keys. If more than 8 are specified in the **XModifierKeymap** structure, a **BadLength** error results.

The modifiermap member of the **XModifierKeymap** structure contains 8 sets of max_keypermod KeyCodes, one for each modifier in the order **Shift**, **Lock**, **Control**, **Mod1**, **Mod2**, **Mod3**, **Mod4**, and **Mod5**. Only nonzero KeyCodes have meaning in each set, and zero KeyCodes are ignored. In addition, all of the nonzero KeyCodes must be in the range specified by min_keycode and max_keycode in the **Display** structure, or a **BadValue** error results.

An X server can impose restrictions on how modifiers can be changed, for example, if certain keys do not generate up transitions in hardware, if auto-repeat cannot be disabled on certain keys, or if multiple modifier keys are not supported. If some such restriction is violated, the status reply is **MappingFailed**, and none of the modifiers are changed. If the new KeyCodes specified for a modifier differ from those currently defined and any (current or new) keys for that modifier are in the logically down state, **XSetModifierMapping** returns **MappingBusy**, and none of the modifiers is changed.

XSetModifierMapping can generate **BadAlloc** and **BadValue** errors.

The **XGetModifierMapping** function returns a pointer to a newly created **XModifierKeymap** structure that contains the keys being used as modifiers. The structure should be freed after use by calling **XFreeModifiermap**. If only zero values appear in the set for any modifier, that modifier is disabled.

The **XNewModifiermap** function returns a pointer to **XModifierKeymap** structure for later use.

The **XInsertModifiermapEntry** function adds the specified KeyCode to the set that controls the specified modifier and returns the resulting **XModifierKeymap** structure (expanded as needed).

The **XDeleteModifiermapEntry** function deletes the specified KeyCode from the set that controls the specified modifier and returns a pointer to the resulting **XModifierKeymap** structure.

The **XFreeModifiermap** function frees the specified **XModifierKeymap** structure.

STRUCTURES

The **XModifierKeymap** structure contains:

```
typedef struct {
    int max_keypermod;      /* This server's max number of keys per modifier */
    KeyCode *modifiermap;   /* An 8 by max_keypermod array of the modifiers */
} XModifierKeymap;
```

DIAGNOSTICS

BadAlloc The server failed to allocate the requested resource or server memory.

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.

SEE ALSO

XFree(3), XkbGetMap(3), XSetPointerMapping(3)
Xlib - C Language X Interface