NAME

XkbChangeTypesOfKey - Change the number of groups and the types bound to a key

SYNOPSIS

Status XkbChangeTypesOfKey (XkbDescPtr *xkb*, **int** *key*, **int** *n_groups*, **unsigned int** *groups*, **int ****new_types_in*, **XkbMapChangesPtr** *p_changes***);**

ARGUMENTS

xkb keyboard description to be changed

key keycode for key of interest

n_groups
new number of groups for key

groups
mask indicating groups to change

new_types_in
indices for new groups specified in groups

p_changes
notes changes made to xkb

DESCRIPTION

XkbChangeTypesOfKey reallocates the symbols and actions bound to the key, if necessary, and initializes any new symbols or actions to NoSymbol or NoAction, as appropriate. If the p_changes parameter is not NULL, *XkbChangeTypesOfKey* adds the XkbKeySymsMask to the changes field of *p_changes* and modifies the *first_key_sym* and *num_key_syms* fields of *p_changes* to include the *key* that was changed. See STRUCTURE for more information on the XkbMapChangesPtr structure. If successful, *XkbChangeTypesOfKey* returns Success.

The *n_groups* parameter specifies the new number of groups for the key. The *groups* parameter is a mask specifying the groups for which new types are supplied and is a bitwise inclusive OR of the following masks: XkbGroup1Mask, XkbGroup2Mask, XkbGroup3Mask, and XkbGroup4Mask.

The *new_types_in* parameter is an integer array of length *n_groups.* Each entry represents the type to use for the associated group and is an index into *xkb->map->types.* The *new_types_in* array is indexed by group index; if *n_groups* is four and *groups* only has Group1Mask and Group3Mask set, *new_types_in* looks like this:

```
new_types_in[0] = type for Group1
new_types_in[1] = ignored
new_types_in[2] = type for Group3
new_types_in[3] = ignored
```

For convenience, Xkb provides the following constants to use as indices to the groups:

Table 1 Group Index Constants

Constant Name	Value
XkbGroup1Index0	
XkbGroup2Index1	
XkbGroup3Index2	
XkbGroup4Index3	

If the Xkb extension has not been properly initialized, *XkbChangeTypesOfKey* returns BadAccess. If the *xkb* parameter it not valid (that is, it is NULL or it does not contain a valid client map), *XkbChangeTypesOfKey* returns BadMatch. If the *key* is not a valid keycode, *n_groups* is greater than XkbNumKbdGroups, or the *groups* mask does not contain any of the valid group mask bits, *XkbChangeTypesOfKey* returns BadValue. If it is necessary to resize the key symbols or key actions arrays and any allocation errors occur, *XkbChangeTypesOfKey* returns BadAlloc.

STRUCTURES

Use the XkbMapChangesRec structure to identify and track partial modifications to the mapping components and to reduce the amount of traffic between the server and clients.

```
typedef struct _XkbMapChanges {
   unsigned short   changed;           /* identifies valid components in structure */
   KeyCode          min_key_code;      /* lowest numbered keycode for device */
   KeyCode          max_key_code;      /* highest numbered keycode for device */
   unsigned char    first_type;        /* index of first key type modified */
   unsigned char    num_types;         /* # types modified */
   KeyCode          first_key_sym;     /* first key whose key_sym_map changed */
   unsigned char    num_key_syms;      /* # key_sym_map entries changed */
   KeyCode          first_key_act;     /* first key whose key_acts entry changed */
   unsigned char    num_key_acts;      /* # key_acts entries changed */
   KeyCode          first_key_behavior; /* first key whose behaviors changed */
```

```
    unsigned char    num_key_behaviors;  /* # behaviors entries changed */
    KeyCode          first_key_explicit; /* first key whose explicit entry changed */
    unsigned char    num_key_explicit;   /* # explicit entries changed */
    KeyCode          first_modmap_key;   /* first key whose modmap entry changed */
    unsigned char    num_modmap_keys;    /* # modmap entries changed */
    KeyCode          first_vmodmap_key;  /* first key whose vmodmap changed */
    unsigned char    num_vmodmap_keys;   /* # vmodmap entries changed */
    unsigned char    pad1;               /* reserved */
    unsigned short   vmods;              /* mask indicating which vmods changed */
} XkbMapChangesRec,*XkbMapChangesPtr;
```

DIAGNOSTICS

BadAccess The Xkb extension has not been properly initialized

BadAlloc Unable to allocate storage

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