NAME
> XkbSetMap − Send a complete new set of values for entire components to the server.

SYNOPSIS

> **Bool XkbSetMap** (**Display ****dpy*, **unsigned int** *which*, **XkbDescPtr** *xkb*);

ARGUMENTS
> *dpy*   connection to X server
>
> *which*   mask selecting subcomponents to update
>
> *xkb*   description from which new values are taken

DESCRIPTION
> There are two ways to make changes to map components: either change a local copy of the keyboard map
> and call *XkbSetMap* to send the modified map to the server, or, to reduce network traffic, use an
> XkbMapChangesRec structure and call *XkbChangeMap.*
>
> Use *XkbSetMap* to send a complete new set of values for entire components (for example, all symbols, all
> actions, and so on) to the server. The *which* parameter specifies the components to be sent to the server, and
> is a bitwise inclusive OR of the masks listed in Table 1. The *xkb* parameter is a pointer to an XkbDescRec
> structure and contains the information to be copied to the server. For each bit set in the *which* parameter,
> *XkbSetMap* takes the corresponding structure values from the *xkb* parameter and sends it to the server spec-
> ified by *dpy*.
>
> If any components specified by *which* are not present in the *xkb* parameter, *XkbSetMap* returns False. Oth-
> erwise, it sends the update request to the server and returns True. *XkbSetMap* can generate BadAlloc,
> BadLength, and BadValue protocol errors.
>
> Key types, symbol maps, and actions are all interrelated; changes in one require changes in the others. Xkb
> provides functions to make it easier to edit these components and handle the interdependencies. Table 1
> lists these helper functions and provides a pointer to where they are defined.

<div align="center">

Table 1 Xkb Mapping Component Masks
and Convenience Functions

</div>

Mask	Value	Map	Fields	Convenience Functions
XkbKeyTypesMask	(1<<0)	client	types size_types num_types	XkbGetKeyTypes XkbResizeKeyType XkbCopyKeyType XkbCopyKeyTypes
XkbKeySymsMask	(1<<1)	client	syms size_syms num_syms key_sym_map	XkbGetKeySyms XkbResizeKeySyms XkbChangeTypesOfKey
XkbModifierMapMask	(1<<2)	client	modmap	XkbGetKeyModifierMap
XkbExplicitComponentsMask	(1<<3)	server	explicit	XkbGetKeyExplicitComponents
XkbKeyActionsMask	(1<<4)	server	key_acts acts num_acts size_acts	XkbGetKeyActions XkbResizeKeyActions
XkbKeyBehaviorsMask	(1<<5)	server	behaviors	XkbGetKeyBehaviors
XkbVirtualModsMask	(1<<6)	server	vmods	XkbGetVirtualMods
XkbVirtualModMapMask	(1<<7)	server	vmodmap	XkbGetVirtualModMap

> The *changed* field identifies the map components that have changed in an XkbDescRec structure and may

contain any of the bits in Table 1, which are also shown in Table 2. Every 1 bit in *changed* also identifies which other fields in the XkbMapChangesRec structure contain valid values, as indicated in Table 2. The *min_key_code* and *max_key_code* fields are for reference only; they are ignored on any requests sent to the server and are always updated by the server whenever it returns the data for an XkbMapChangesRec.

Table 2 XkbMapChangesRec Masks

Mask	Valid XkbMapChangesRec Fields	XkbDescRec Field Containing Changed Data
XkbKeyTypesMask	first_type, num_types	map->type[first_type] .. map->type[first_type + num_types - 1]
XkbKeySymsMask	first_key_sym, num_key_syms	map->key_sym_map[first_key_sym] .. map->key_sym_map[first_key_sym + num_key_syms - 1]
XkbModifierMapMask	first_modmap_key, num_modmap_keys	map->modmap[first_modmap_key] .. map->modmap[first_modmap_key + num_modmap_keys - 1]
XkbExplicitComponentsMask	first_key_explicit, num_key_explicit	server->explicit[first_key_explicit] .. server->explicit[first_key_explicit + num_key_explicit - 1]
XkbKeyActionsMask	first_key_act, num_key_acts	server->key_acts[first_key_act] .. server->key_acts[first_key_act + num_key_acts - 1]
XkbKeyBehaviorsMask	first_key_behavior, num_key_behaviors	server->behaviors[first_key_behavior] .. server->behaviors[first_key_behavior + num_key_behaviors - 1]
XkbVirtualModsMask	vmods	server->vmods[*]
XkbVirtualModMapMask	first_vmodmap_key, num_vmodmap_keys	server->vmodmap[first_vmodmap_key] .. server->vmodmap[first_vmodmap_key + num_vmodmap_keys - 1]

RETURN VALUES

True	The *XkbSetMap* function returns True all components specified by *which* are present in the *xkb* parameter.
False	The *XkbSetMap* function returns False if any component specified by *which* is not present in the *xkb* parameter.

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;
```

The complete description of an Xkb keyboard is given by an XkbDescRec. The component structures in the XkbDescRec represent the major Xkb components.

```
typedef struct {
    struct _XDisplay * display;     /* connection to X server */
    unsigned short   flags;         /* private to Xkb, do not modify */
    unsigned short   device_spec;   /* device of interest */
    KeyCode          min_key_code;  /* minimum keycode for device */
    KeyCode          max_key_code;  /* maximum keycode for device */
    XkbControlsPtr   ctrls;         /* controls */
    XkbServerMapPtr  server;        /* server keymap */
    XkbClientMapPtr  map;           /* client keymap */
    XkbIndicatorPtr  indicators;    /* indicator map */
    XkbNamesPtr      names;         /* names for all components */
    XkbCompatMapPtr  compat;        /* compatibility map */
    XkbGeometryPtr   geom;          /* physical geometry of keyboard */
} XkbDescRec, *XkbDescPtr;
```

The *display* field points to an X display structure. The *flags field is private to the library: modifying flags* may yield unpredictable results. The *device_spec* field specifies the device identifier of the keyboard input device, or XkbUseCoreKeyboard, which specifies the core keyboard device. The *min_key_code* and *max_key_code* fields specify the least and greatest keycode that can be returned by the keyboard.

Each structure component has a corresponding mask bit that is used in function calls to indicate that the structure should be manipulated in some manner, such as allocating it or freeing it. These masks and their relationships to the fields in the XkbDescRec are shown in Table 3.

Table 3 Mask Bits for XkbDescRec

Mask Bit	XkbDescRec Field	Value
XkbControlsMask	ctrls	(1L<<0)
XkbServerMapMask	server	(1L<<1)
XkbIClientMapMask	map	(1L<<2)
XkbIndicatorMapMask	indicators	(1L<<3)
XkbNamesMask	names	(1L<<4)
XkbCompatMapMask	compat	(1L<<5)
XkbGeometryMask	geom	(1L<<6)
XkbAllComponentsMask	All Fields	(0x7f)

DIAGNOSTICS

BadAlloc   Unable to allocate storage

BadLength   The length of a request is shorter or longer than that required to minimally contain the arguments

BadValue   An argument is out of range

SEE ALSO

XkbChangeMap(3), **XkbChangeTypesOfKey**(3), **XkbCopyKeyType**(3), **XkbCopyKeyTypes**(3), **XkbGetKeyActions**(3), **XkbGetKeyBehaviors**(3), **XkbGetKeyExplicitComponents**(3), **XkbGetKeyModifierMap**(3), **XkbGetKeySyms**(3), **XkbGetKeyTypes**(3), **XkbResizeKeyActions**(3), **XkbResizeKeySyms**(3), **XkbResizeKeyType**(3), **XkbGetVirtualModMap**(3), **XkbGetVirtualMods**(3)