NAME

XAllocWMHints, XSetWMHints, XGetWMHints, XWMHints − allocate window manager hints structure and set or read a window's WM_HINTS property

SYNTAX

XWMHints *XAllocWMHints(void);

int XSetWMHints(Display **display*, Window *w*, XWMHints **wmhints*);

XWMHints *XGetWMHints(Display **display*, Window *w*);

ARGUMENTS

display	Specifies the connection to the X server.
w	Specifies the window.
wmhints	Specifies the **XWMHints** structure to be used.

DESCRIPTION

The **XAllocWMHints** function allocates and returns a pointer to a **XWMHints** structure. Note that all fields in the **XWMHints** structure are initially set to zero. If insufficient memory is available, **XAllocWMHints** returns NULL. To free the memory allocated to this structure, use **XFree**.

The **XSetWMHints** function sets the window manager hints that include icon information and location, the initial state of the window, and whether the application relies on the window manager to get keyboard input.

XSetWMHints can generate **BadAlloc** and **BadWindow** errors.

The **XGetWMHints** function reads the window manager hints and returns NULL if no WM_HINTS property was set on the window or returns a pointer to a **XWMHints** structure if it succeeds. When finished with the data, free the space used for it by calling **XFree**.

XGetWMHints can generate a **BadWindow** error.

PROPERTIES

WM_HINTS	Additional hints set by the client for use by the window manager. The C type of this property is **XWMHints**.

STRUCTURES

The **XWMHints** structure contains:

/* Window manager hints mask bits */

#define	**InputHint**	(1L << 0)
#define	**StateHint**	(1L << 1)
#define	**IconPixmapHint**	(1L << 2)
#define	**IconWindowHint**	(1L << 3)
#define	**IconPositionHint**	(1L << 4)
#define	**IconMaskHint**	(1L << 5)
#define	**WindowGroupHint**	(1L << 6)
#define	**XUrgencyHint**	(1L << 8)

#define **AllHints** (InputHint|
StateHint|
IconPixmapHint|
IconWindowHint|
IconPositionHint|
IconMaskHint|
WindowGroupHint)

```
/* Values */

typedef struct {
        long flags;     /* marks which fields in this structure are defined */
        Bool input;     /* does this application rely on the window manager to
                        get keyboard input? */
        int initial_state;      /* see below */
        Pixmap icon_pixmap;     /* pixmap to be used as icon */
        Window icon_window;     /* window to be used as icon */
        int icon_x, icon_y;     /* initial position of icon */
        Pixmap icon_mask;       /* pixmap to be used as mask for icon_pixmap */
        XID window_group;       /* id of related window group */
        /* this structure may be extended in the future */
} XWMHints;
```

The input member is used to communicate to the window manager the input focus model used by the application. Applications that expect input but never explicitly set focus to any of their subwindows (that is, use the push model of focus management), such as X Version 10 style applications that use real-estate driven focus, should set this member to **True**. Similarly, applications that set input focus to their subwindows only when it is given to their top-level window by a window manager should also set this member to **True**. Applications that manage their own input focus by explicitly setting focus to one of their subwindows whenever they want keyboard input (that is, use the pull model of focus management) should set this member to **False**. Applications that never expect any keyboard input also should set this member to **False**.

Pull model window managers should make it possible for push model applications to get input by setting input focus to the top-level windows of applications whose input member is **True**. Push model window managers should make sure that pull model applications do not break them by resetting input focus to **PointerRoot** when it is appropriate (for example, whenever an application whose input member is **False** sets input focus to one of its subwindows).

The definitions for the initial_state flag are:

#define **WithdrawnState** 0

#define **NormalState** 1 /* most applications start this way */

#define **IconicState** 3 /* application wants to start as an icon */

The icon_mask specifies which pixels of the icon_pixmap should be used as the icon. This allows for nonrectangular icons. Both icon_pixmap and icon_mask must be bitmaps. The icon_window lets an application provide a window for use as an icon for window managers that support such use. The window_group lets you specify that this window belongs to a group of other windows. For example, if a single application manipulates multiple top-level windows, this allows you to provide enough information that a window manager can iconify all of the windows rather than just the one window.

The **UrgencyHint** flag, if set in the flags field, indicates that the client deems the window contents to be urgent, requiring the timely response of the user. The window manager will make some effort to draw the user's attention to this window while this flag is set. The client must provide some means by which the user can cause the urgency flag to be cleared (either mitigating the condition that made the window urgent or merely shutting off the alarm) or the window to be withdrawn.

DIAGNOSTICS

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

BadWindow A value for a Window argument does not name a defined Window.

SEE ALSO

XAllocClassHint(3), XAllocIconSize(3), XAllocSizeHints(3), XFree(3), XSetCommand(3), XSetTransientForHint(3), XSetTextProperty(3), XSetWMClientMachine(3), XSetWMColormapWindows(3), XSetWMIconName(3), XSetWMName(3), XSetWMProperties(3), XSetWMProtocols(3), XStringListToTextProperty(3)

Xlib − C Language X Interface, O'Reilly and Associates, Sebastopol, 1991.