XSetWMProperties, XmbSetWMProperties – set standard window properties

void XSetWMProperties(*display, w, window_name, icon_name, argv, argc, normal_hints, wm_hints, class_hints***)**
Display ***display;**
Window *w**;**
XTextProperty ***window_name;**
XTextProperty ***icon_name;**
char ****argv;**
int *argc**;**
XSizeHints ***normal_hints;**
XWMHints ***wm_hints;**
XClassHint ***class_hints;**

void XmbSetWMProperties(*display*, *w*, *window_name*, *icon_name*, *argv*, *argc*,
normal_hints, *wm_hints*, *class_hints*)
Display **display*;
Window *w*;
char **window_name*;
char **icon_name*;
char **argv*[];
int *argc*;
XSizeHints **normal_hints*;
XWMHints **wm_hints*;
XClassHint **class_hints*;

argc	Specifies the number of arguments.
argv	Specifies the application's argument list.
class_hints	Specifies the **XClassHint** structure to be used.
display	Specifies the connection to the X server.
icon_name	Specifies the icon name, which should be a null-terminated string.
normal_hints	Specifies the size hints for the window in its normal state.
w	Specifies the window.
window_name	Specifies the window name, which should be a null-terminated string.
wm_hints	Specifies the **XWMHints** structure to be used.

The XSetWMProperties convenience function provides a single programming interface for setting those essential window properties that are used for communicating with other clients (particularly window and session managers).

If the window_name argument is non-NULL, **XSetWMProperties** calls **XSetWMName**, which in turn, sets the WM_NAME property (see section 14.1.4). If the icon_name argument is non-NULL, **XSetWM-Properties** calls **XSetWMIconName**, which sets the WM_ICON_NAME property (see section 14.1.5). If the argv argument is non-NULL, **XSetWMProperties** calls **XSetCommand**, which sets the WM_COMMAND property (see section 14.2.1). Note that an argc of zero is allowed to indicate a zero-length command. Note also that the hostname of this machine is stored using **XSetWMClientMachine** (see section 14.2.2).

If the normal_hints argument is non-NULL, **XSetWMProperties** calls **XSetWMNormalHints**, which sets the WM_NORMAL_HINTS property (see section 14.1.7). If the wm_hints argument is non-NULL, **XSetWMProperties** calls **XSetWMHints**, which sets the WM_HINTS property (see section 14.1.6).

If the class_hints argument is non-NULL, **XSetWMProperties** calls **XSetClassHint**, which sets the WM_CLASS property (see section 14.1.8). If the res_name member in the **XClassHint** structure is set to the NULL pointer and the RESOURCE_NAME environment variable is set, then the value of the

environment variable is substituted for res_name. If the res_name member is NULL, the environment variable is not set, and argv and argv[0] are set, then the value of argv[0], stripped of any directory prefixes, is substituted for res_name.

The **XmbSetWMProperties** convenience function provides a simple programming interface for setting those essential window properties that are used for communicating with other clients (particularly window and session managers).

If the window_name argument is non-NULL, **XmbSetWMProperties** sets the WM_NAME property. If the icon_name argument is non-NULL, **XmbSetWMProperties** sets the WM_ICON_NAME property. The window_name and icon_name arguments are null-terminated strings in the encoding of the current locale. If the arguments can be fully converted to the STRING encoding, the properties are created with type ‘‘STRING’’; otherwise, the arguments are converted to Compound Text, and the properties are created with type ‘‘COMPOUND_TEXT’’.

If the normal_hints argument is non-NULL, **XmbSetWMProperties** calls **XSetWMNormalHints**, which sets the WM_NORMAL_HINTS property (see section 14.1.7). If the wm_hints argument is non-NULL, **XmbSetWMProperties** calls **XSetWMHints**, which sets the WM_HINTS property (see section 14.1.6).

If the argv argument is non-NULL, **XmbSetWMProperties** sets the WM_COMMAND property from argv and argc. An argc of zero indicates a zero-length command.

The hostname of the machine is stored using **XSetWMClientMachine** (see section 14.2.2).

If the class_hints argument is non-NULL, **XmbSetWMProperties** sets the WM_CLASS property. If the res_name member in the **XClassHint** structure is set to the NULL pointer and the RESOURCE_NAME environment variable is set, the value of the environment variable is substituted for res_name. If the res_name member is NULL, the environment variable is not set, and argv and argv[0] are set, then the value of argv[0], stripped of any directory prefixes, is substituted for res_name.

It is assumed that the supplied class_hints.res_name and argv, the RESOURCE_NAME environment variable, and the hostname of the machine are in the encoding of the locale announced for the LC_CTYPE category (on POSIX-compliant systems, the LC_CTYPE, else LANG environment variable). The corresponding WM_CLASS, WM_COMMAND, and WM_CLIENT_MACHINE properties are typed according to the local host locale announcer. No encoding conversion is performed prior to storage in the properties.

For clients that need to process the property text in a locale, **XmbSetWMProperties** sets the WM_LOCALE_NAME property to be the name of the current locale. The name is assumed to be in the Host Portable Character Encoding and is converted to STRING for storage in the property.

XSetWMProperties and **XmbSetWMProperties** can generate **BadAlloc** and **BadWindow** errors.

WM_CLASS Set by application programs to allow window and session managers to obtain the application’s resources from the resource database. WM_CLIENT_MACHINE The string name of the machine on which the client application is running. WM_COMMAND The command and arguments, null-separated, used to invoke the application. WM_HINTS Additional hints set by the client for use by the window manager. The C type of this property is XWMHints. WM_ICON_NAME The name to be used in an icon. WM_NAME The name of the application. WM_NORMAL_HINTS Size hints for a window in its normal state. The C type of this property is **XSizeHints**.

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.