NAME

XTestQueryExtension, XTestCompareCursorWithWindow, XTestCompareCurrentCursorWithWindow, XTestFakeKeyEvent, XTestFakeButtonEvent, XTestFakeMotionEvent, XTestFakeRelativeMotionEvent, XTestGrabControl, XTestSetGContextOfGC, XTestSetVisualIDOfVisual, XTestDiscard - XTest extension functions

SYNOPSIS

cc [flag ...] file ... -lXtst [library ...]

#include <X11/extensions/XTest.h>

Bool XTestQueryExtension(*display***,** *event_base_return***,** *error_base_return***,** *major_version_return***,** *minor_version_return***);**

Display *display;
int *event_base_return;
int *error_base_return;
int *major_version_return;
int *minor_version_return;

Bool XTestCompareCursorWithWindow(*display***,** *window***,** *cursor***);**

Display *display;
Window window;
Cursor cursor;

Bool XTestCompareCurrentCursorWithWindow(*display***,** *window***);**

Display *display;
Window window;

int XTestFakeKeyEvent(*display***,** *keycode***,** *is_press***,** *delay***);**

Display *display;
unsigned int keycode;
Bool is_press;
unsigned long delay;

int XTestFakeButtonEvent(*display***,** *button***,** *is_press***,** *delay***);**

Display *display;
unsigned int button;
Bool is_press;
unsigned long delay;

int XTestFakeMotionEvent(*display***,** *screen_number***,** *x***,** *y***,** *delay***);**

Display *display;
int screen_number;
int x, y;
unsigned long delay;

int XTestFakeRelativeMotionEvent(*display***,** *screen_number***,** *x***,** *y***,** *delay***);**

Display *display;
int screen_number;
int x, y;
unsigned long delay;

int XTestGrabControl(*display***,** *impervious***);**

Display *display;
Bool impervious;

void XTestSetGContextOfGC(*gc***,** *gid***);**

GC gc;
GContext gid;

void XTestSetVisualIDOfVisual(*visual***,** *visualid***);**

Visual *visual;
VisualID visualid;

Status XTestDiscard(*display***);**

Display *display;

DESCRIPTION

This extension is a minimal set of client and server extensions required to completely test the X11 server with no user intervention. This extension is not intended to support general journaling and playback of user actions.

The functions provided by this extension fall into two groups:

Client Operations

These routines manipulate otherwise hidden client-side behavior. The actual implementation will depend on the details of the actual language binding and what degree of request buffering, GContext caching, and so on, is provided. In the C binding, routines are provided to access the internals of two opaque data structures -- GCs and Visuals -- and to discard any requests pending within the output buffer of a connection. The exact details can be expected to differ for other language bindings.

Server Requests

The first of these requests is similar to that provided in most extensions: it allows a client to specify a major and minor version number to the server and for the server to respond with major and minor versions of its own. The remaining two requests allow the following:

- to an otherwise *write-only* server resource: the cursor associated with a given window

- most importantly, limited synthesis of input device events, almost as if a cooperative user had moved the pointing device or pressed a key or button.

All XTEST extension functions and procedures, and all manifest constants and macros, will start with the string *XTest*. All operations are classified as server/client (Server) or client-only (Client).

XTestQueryExtension returns True if the specified display supports the XTEST extension, else False. If the extension is supported, *event_base would be set to the event number for the first event for this extension and *error_base would be set to the error number for the first error for this extension. As no errors or events are defined for this version of the extension, the values returned here are not defined (nor useful). If the extension is supported, *major_version and *minor_version are set to the major and minor version numbers of the extension supported by the display. Otherwise, none of the arguments are set.

If the extension is supported, **XTestCompareCursorWithWindow** performs a comparison of the cursor whose ID is specified by cursor (which may be **None**) with the cursor of the window specified by window returning True if they are the same and False otherwise. If the extension is not supported, then the request is ignored and zero is returned.

If the extension is supported, **XTestCompareCurrentCursorWithWindow** performs a comparison of the current cursor with the cursor of the specified window returning True if they are the same and False

otherwise. If the extension is not supported, then the request is ignored and zero is returned.

If the extension is supported, **XTestFakeKeyEvent** requests the server to simulate either a **KeyPress** (if is_press is True) or a **KeyRelease** (if is_press is False) of the key with the specified keycode; otherwise, the request is ignored.

If the extension is supported, the simulated event will not be processed until delay milliseconds after the request is received (if delay is **CurrentTime**, then this is interpreted as no delay at all). No other requests from this client will be processed until this delay, if any, has expired and subsequent processing of the simulated event has been completed.

If the extension is supported, **XTestFakeButtonEvent** requests the server to simulate either a **ButtonPress** (if is_press is True) or a **ButtonRelease** (if is_press is False) of the logical button numbered by the specified button; otherwise, the request is ignored.

If the extension is supported, the simulated event will not be processed until delay milliseconds after the request is received (if delay is **CurrentTime**, then this is interpreted as no delay at all). No other requests from this client will be processed until this delay, if any, has expired and subsequent processing of the simulated event has been completed.

If the extension is supported, **XTestFakeMotionEvent** requests the server to simulate a movement of the pointer to the specified position (x, y) on the root window of screen_number; otherwise, the request is ignored. If screen_number is -1, the current screen (that the pointer is on) is used.

If the extension is supported, the simulated event will not be processed until delay milliseconds after the request is received (if delay is **CurrentTime**, then this is interpreted as no delay at all). No other requests from this client will be processed until this delay, if any, has expired and subsequent processing of the simulated event has been completed.

If the extension is supported, **XTestFakeRelativeMotionEvent** requests the server to simulate a movement of the pointer by the specified offsets (x, y) relative to the current pointer position on screen_number; otherwise, the request is ignored. If screen_number is -1, the current screen (that the pointer is on) is used.

If the extension is supported, the simulated event will not be processed until delay milliseconds after the request is received (if delay is **CurrentTime**, then this is interpreted as no delay at all). No other requests from this client will be processed until this delay, if any, has expired and subsequent processing of the simulated event has been completed.

If impervious is True, then the executing client becomes impervious to server grabs. If impervious is False, then the executing client returns to the normal state of being susceptible to server grabs.

XTestSetGContextOfGC sets the GContext within the opaque datatype referenced by gc to be that specified by gid.

XTestSetVisualIDOfVisual sets the VisualID within the opaque datatype referenced by visual to be that specified by visualid.

XTestDiscard discards any requests within the output buffer for the specified display. It returns True if any requests were discarded; otherwise, it returns False.

RETURN VALUES

All routines that have return type Status will return nonzero for success and zero for failure. Even if the XTEST extension is supported, the server may withdraw such facilities arbitrarily; in which case they will subsequently return zero.

AUTHOR

Kieron Drake
UniSoft Ltd.
Author.