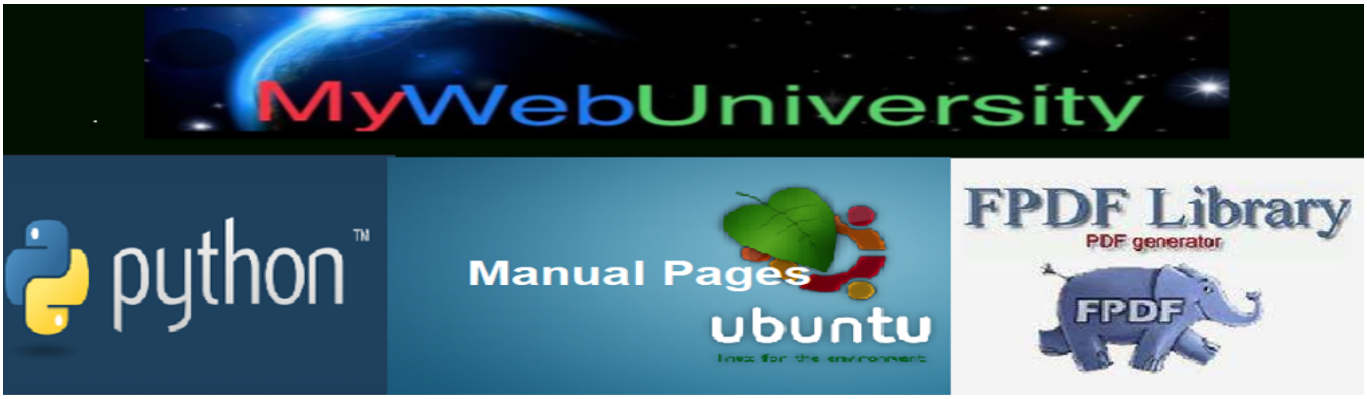

Rocky Enterprise Linux 9.2 Manual Pages on command 'XtDispatchEvent.3'

$ man XtDispatchEvent.3

XtAppNextEvent(3) XT FUNCTIONS XtAppNextEvent(3)

NAME

XtAppNextEvent, XtAppPending, XtAppPeekEvent, XtAppProcessEvent, XtDispatchEvent, XtApp?

MainLoop - query and process events and input

SYNTAX

```
#include <X11/Intrinsic.h>
void XtAppNextEvent(XtAppContext app_context, XEvent *event_return);
Boolean XtAppPeekEvent(XtAppContext app_context, XEvent *event_return);
XtInputMask XtAppPending(XtAppContext app_context);
void XtAppProcessEvent(XtAppContext app_context, XtInputMask mask);
Boolean XtDispatchEvent(XEvent *event);
void XtAppMainLoop(XtAppContext app_context);
```

ARGUMENTS

app_context

Specifies the application context that identifies the application.

event Specifies a pointer to the event structure that is to be dispatched to the ap?

propriate event handler.

event_return

Returns the event information to the specified event structure.

mask Specifies what types of events to process. The mask is the bitwise inclusive OR

of any combination of XtIMXEvent, XtIMTimer, XtIMAlternateInput, and XtIMSignal.

As a convenience, the X Toolkit defines the symbolic name XtIMAll to be the bit?

wise inclusive OR of all event types.

DESCRIPTION

If the X event queue is empty, XtAppNextEvent flushes the X output buffers of each Display in the application context and waits for an event while looking at the other input sources, timeout timeout values, and signal handlers and calling any callback procedures triggered by them. This wait time can be used for background processing (see Section 7.8).

If there is an event in the queue, XtAppPeekEvent fills in the event and returns a nonzero value. If no X input is on the queue, XtAppPeekEvent flushes the output buffer and blocks until input is available (possibly calling some timeout callbacks in the process). If the input is an event, XtAppPeekEvent fills in the event and returns a nonzero value. Other? wise, the input is for an alternate input source, and XtAppPeekEvent returns zero.

The XtAppPending function returns a nonzero value if there are events pending from the X server, timer pending, or other input sources pending. The value returned is a bit mask that is the OR of XtIMXEvent, XtIMTimer, XtIMAlternateInput, and XtIMSignal (see XtApp? ProcessEvent). If there are no events pending, XtAppPending flushes the output buffer and returns zero.

The XtAppProcessEvent function processes one timer, alternate input, signal source, or X event. If there is nothing of the appropriate type to process, XtAppProcessEvent blocks until there is. If there is more than one type of thing available to process, it is unde? fined which will get processed. Usually, this procedure is not called by client applica? tions (see XtAppMainLoop). XtAppProcessEvent processes timer events by calling any appro? priate timer callbacks, alternate input by calling any appropriate alternate input call? backs, signal source by calling any appropriate signal callbacks, and X events by calling XtDispatchEvent.

When an X event is received, it is passed to XtDispatchEvent, which calls the appropriate event handlers and passes them the widget, the event, and client-specific data registered with each procedure. If there are no handlers for that event registered, the event is ig? nored and the dispatcher simply returns. The order in which the handlers are called is undefined.

The XtDispatchEvent function sends those events to the event handler functions that have been previously registered with the dispatch routine. XtDispatchEvent returns True if it dispatched the event to some handler and False if it found no handler to dispatch the event to. The most common use of XtDispatchEvent is to dispatch events acquired with the

XtAppNextEvent procedure. However, it also can be used to dispatch user-constructed events. XtDispatchEvent also is responsible for implementing the grab semantics for XtAd? dGrab.

The XtAppMainLoop function processes events using XtAppProcessEvent, varying the mask pa? rameter and using XtAppPending to ensure that it has a chance to handle events of all types, i.e., X events, timer events, input events and signal sources. This constitutes the main loop of X Toolkit applications, and, as such, it does not return unless XtAppSe? tExitFlag is called. Applications are expected to exit in response to some user action.

There is nothing special about XtAppMainLoop; it is simply an loop that processes events until XtAppGetExitFlag() returns true.

Applications can provide their own version of this loop, which tests some global termina? tion flag or tests that the number of top-level widgets is larger than zero before cir? cling back for the next event.

SEE ALSO

X Toolkit Intrinsics - C Language Interface

Xlib - C Language X Interface