

**NAME**

`xkbevd` – XKB event daemon

**SYNOPSIS**

`xkbevd` [ options ]

**DESCRIPTION**

This command is very raw and is therefore only partially implemented; we present it here as a rough prototype for developers, not as a general purpose tool for end users. Something like this might make a suitable replacement for `xev`; I'm not signing up, mind you, but it's an interesting idea.

The `xkbevd` event daemon listens for specified XKB events and executes requested commands if they occur. The configuration file consists of a list of event specification/action pairs and/or variable definitions.

An event specification consists of a short XKB event name followed by a string or identifier which serves as a qualifier in parentheses; empty parenthesis indicate no qualification and serve to specify the default command which is applied to events which do not match any of the other specifications. The interpretation of the qualifier depends on the type of the event: Bell events match using the name of the bell, message events match on the contents of the message string and slow key events accept any of *press*, *release*, *accept*, or *reject*. No other events are currently recognized.

An action consists of an optional keyword followed by an optional string argument. Currently, `xkbev` recognizes the actions: *none*, *ignore*, *echo*, *printEvent*, *sound*, and *shell*. If the action is not specified, the string is taken as the name of a sound file to be played unless it begins with an exclamation point, in which case it is taken as a shell command.

Variable definitions in the argument string are expanded with fields from the event in question before the argument string is passed to the action processor. The general syntax for a variable is either `$c` or `$(str)`, where `c` is a single character and `str` is a string of arbitrary length. All parameters have both single-character and long names.

The list of recognized parameters varies from event to event and is too long to list here right now. This is a developer release anyway, so you can be expected to look at the source code (`evargs.c` is of particular interest).

The *ignore*, *echo*, *printEvent*, *sound*, and *shell* actions do what you would expect commands named *ignore*, *echo*, *printEvent*, *sound*, and *shell* to do, except that the sound command has only been implemented and tested for SGI machines. It launches an external program right now, so it should be pretty easy to adapt, especially if you like audio cues that arrive about a half-second after you expect them.

The only currently recognized variables are *soundDirectory* and *soundCmd*. I'm sure you can figure out what they do.

**OPTIONS**

- help** Prints a usage message that is far more up-to-date than anything in this man page.
- cfg file** Specifies the configuration file to read. If no configuration file is specified, `xkbevd` looks for `~/xkb/xkbevd.cf` and `$(LIBDIR)/xkb/xkbevd.cf` in that order.
- sc cmd** Specifies the command used to play sounds.
- sd directory**  
Specifies a top-level directory for sound files.
- display display**  
Specifies the display to use. If not present, `xkbevd` uses `$DISPLAY`.
- bg** Tells `xkbevd` to fork itself (and run in the background).
- synch** Forces synchronization of all X requests. Slow.
- v** Print more information, including debugging messages. Multiple specifications of `-v` cause more output, to a point.

**-version**

Prints the program version and exits.

**SEE ALSO**

**xev(1)**, **xkbwatch(1)**, **X(7)**.

**COPYRIGHT**

Copyright 1995, Silicon Graphics Computer Systems  
Copyright 1995, 1998 The Open Group  
See *X(7)* for a full statement of rights and permissions.

**AUTHOR**

Erik Fortune, Silicon Graphics