NAME

tknewsbiff − pop up a window when news appears

SYNOPSIS

tknewsbiff [*server or config-file*]

INTRODUCTION

tknewsbiff pops up a window when there is unread news in your favorite newsgroups and removes the window after you've read the news. tknewsbiff can optionally play a sound, start your newsreader, etc.

SELECTING NEWSGROUPS

By default, the configuration file ~/.tknewsbiff describes how tknewsbiff behaves. The syntax observes the usual Tcl rules - however, even if you don't know Tcl, all but the most esoteric configurations will be obvious.

Each newsgroup (or set of newsgroups) to be watched is described by using the "watch" command. For example:

```
watch dc.dining
watch nist.*
watch comp.unix.wizard  -threshold 3
watch *.sources.*       -threshold 20
```

For each newsgroup pattern, any newsgroup that matches it and which you are subscribed to (according to your newsrc file) is eligible for reporting. By default, tknewsbiff reports on the newsgroup if there is at least one unread article. The "-threshold" flag changes the threshold to the following number. For example, "-threshold 3" means there must be at least three articles unread before tknewsbiff will report the newsgroup.

If no watch commands are given (or no configuration file exists), all groups which are subscribed to are watched.

To suppress newsgroups that would otherwise be reported, use the "ignore" command. For example, the following matches all comp.* and nist.* newgroups except for nist.posix or .d (discussion) groups:

```
watch comp.*
watch nist.*
ignore nist.posix.*
ignore *.d
```

The flag "-new" describes a command to be executed when the newsgroup is first reported as having unread news. For example, the following lines invoke the UNIX command "play" to play a sound.

```
watch dc.dining -new "exec play /usr/local/sounds/yumyum.au"
watch rec.auto* -new "exec play /usr/local/sounds/vroom.au"
```

You can cut down on the verbosity of actions by defining procedures. For example, if you have many -new flags that all play sound files, you could define a sound procedure. This would allow the -new specification to be much shorter.

```
proc play {sound} {
        exec play /usr/local/sounds/$sound.au
}

watch dc.dining -new "play yumyum"
watch rec.auto* -new "play vroom"
```

As an aside, you can put an "&" at the end of an "exec" command to get commands to execute asynchronously. However, it's probably not a good idea to do this when playing sound files anyway.

"newsgroup" is a read-only variable which contains the name of the newsgroup that is being reported. This is useful when the action is triggered by a pattern. For example, the following line could run the newsgroup name through a speech synthesizer:

```
watch * -new {
        exec play herald.au
        exec speak "New news has arrived in $newsgroup."
}
```

The flag "−display" describes a command to be executed every time the newsgroup is reported as having unread news. The special command "display" is the default command. It schedules $newsgroup to be written to tknewsbiff's display when it is rewritten. For example, by explicitly providing a -display flag that omits the display command, you can disable the display of newsgroups that are already reported via -new.

```
watch dc.dining -new {exec play yumyum.au} -display {}
```

If you want to execute an action repeatedly and *still* display the newsgroup in the default manner, explicitly invoke the display command via the -display flag. For example:

```
watch *security* -display {
        exec play red-alert.au
        display
}
```

Actions associated with the -new and -display flags are executed only once for each matching newsgroup. The command executed is the one associated with the first pattern in the configuration file that matches and observes the given threshold.

Any command that is simply listed in the configuration file is executed each time before the update loop in tknewsbiff. The reserved (but user-defined) procedure "user" is run immediately after the newsgroups are scheduled to be written to the display and before they are actually written.

For example, suppose unread articles appear in several rec.auto groups and you play the same sound for each one. To prevent playing the sound several times in a row, make the -new command simply set a flag. In the user procedure, play the sound if the flag is set (and then reset the flag).

The user procedure could also be used to start a newsreader. This would avoid the possibility of starting

multiple newsreaders just because multiple newsgroups contained unread articles. (A check should, of course, be made to make sure that a newsreader is not already running.)

MORE VARIABLES

The following example lines show variables that can affect the behavior of tknewsbiff

```
set delay        120
set server       news.nist.gov
set server_timeout 60
set newsrc       ~/.newsrc
set width        40
set height       20
set active_file   /usr/news/lib/active
```

tknewsbiff alternates between checking for unread news and sleeping (kind of like many undergraduates). The "delay" variable describes how many seconds to sleep.

The "server" variable names an NNTP news-server. The default is "news". The "server" variable is only used if the "active_file" variable is not set.

The "server_timeout" variable describes how how many seconds to wait for a response from the server before giving up. -1 means wait forever or until the server itself times out. The default is 60 seconds.

The "newsrc" variable describes the name of your .newsrc file. By default, tknewsbiff looks in your home directory for a newsrc file. A server-specific newsrc is used if found. For example, if you have set server to "cubit.nist.gov", then tknewsbiff looks for ~/.newsrc-cubit.nist.gov. (This is the Emacs gnus convention - which is very convenient when you read news from multiple servers.) If there is no server-specific newsrc, tknewsbiff uses ~/.newsrc.

The "width" variable describes the width that tknewsbiff will use to display information. If any newsgroup names are long enough, they will be truncated so that the article counts can still be shown. You can manually resize the window to see what was truncated. However, if your configuration file sets the width variable, the window will be restored to that size the next time that tknewsbiff checks for unread news and updates its display.

The "height" variable describes the maximum height that tknewsbiff will use to display information. If fewer newsgroups are reported, tknewsbiff will shrink the window appropriately. You can manually resize the window but if your configuration file sets the height variable, the window will be restored to that size the next time that tknewsbiff checks for unread news and updates its display.

The "active_file" variable describes the name of the news active file. If set, the active file is read directly in preference to using NNTP (even if the "server" variable is set). This is particularly useful for testing out new configuration files since you can edit a fake active file and then click button 2 to immediately see how tknewsbiff responds (see BUTTONS below).

If the environment variable DOTDIR is set, then its value is used as a directory in which to find all dotfiles instead of from the home directory. In particular, this affects the tknewsbiff configuration file and the .newsrc file (assuming the newsrc variable is not set explicitly).

WATCHING DIFFERENT NEWS SERVERS

To watch multiple servers, run tknewsbiff multiple times. (Since you need different .newsrc files and the servers have different newsgroups and article numbers anyway, there is no point in trying to do this in a

single process.)

You can point tknewsbiff at a different server with an appropriate argument. The argument is tried both as a configuration file name and as a suffix to the string "~/.tknewsbiff-". So if you want to watch the server "kidney", store the tknewsbiff configuration information in ~/.tknewsbiff-kidney". The following two commands will both use that configuration file.

```
tknewsbiff kidney
tknewsbiff ~/.tknewsbiff-kidney
```

In both cases, the actual server to contact is set by the value of the server variable in the configuration file.

If no configuration file is found, the argument is used as the server to contact. This allows tknewsbiff to be run with no preparation whatsoever.

If the argument is the special keyword "active" (or ends in "/active"), it is used as the name of an active file. This is in turn used to initialize the variable "active_file" so that tknewsbiff reads from the active file directly rather than using NNTP.

Creating your own active file is a convenient way of testing your configuration file. For example, after running the following command, you can repeatedly edit your active file and trigger the update-now command (either by pressing button 2 or setting the delay variable very low) to see how tknewsbiff responds.

The active file must follow the format of a real active file. The format is one newsgroup per line. After the newsgroup name is the number of the highest article, the lowest article. Lastly is the letter y or m. m means the newsgroup is moderated. y means posting is allowed.

WINDOW

When unread news is found, a window is popped up. The window lists the names of the newsgroups and the number of unread articles in each (unless suppressed by the -display flag). When there is no longer any unread news, the window disappears (although the process continues to run).

BUTTONS

Button or key bindings may be assigned by bind commands. Feel free to change them. The default bind commands are:

```
bind .list <1> help
bind .list <2> update-now
bind .list <3> unmapwindow
```

By default button 1 (left) is bound to "help". The help command causes tknewsbiff to pop up a help window.

By default, button 2 (middle) is bound to "update-now". The update-now command causes tknewsbiff to immediately check for unread news. If your news server is slow or maintains a very large number of newsgroups, or you have a large number of patterns in your configuration file, tknewsbiff can take considerable time before actually updating the window.

By default, button 3 (right) is bound to "unmapwindow". The unmapwindow command causes tknewsbiff to remove the window from the display until the next time it finds unread news. (The mapwindow

command causes tknewsbiff to restore the window.)

As an example, here is a binding to pop up an xterm and run rn when you hold down the shift key and press button 1 in the listing window.

```
bind .list <Shift-1> {
        exec xterm -e rn &
}
```

Here is a similar binding. However it tells rn to look only at the newsgroup that is under the mouse when you pressed it. (The "display_list" variable is described later in this man page.)

```
bind .list <Shift-1> {
        exec xterm -e rn [lindex $display_list [.list nearest %y]] &
}
```

OTHER COMMANDS AND VARIABLES

Built-in commands already mentioned are: watch, ignore, display, help, update-now, unmapwindow, and mapwindow.

Any Tcl and Tk command can also be given. In particular, the list of newsgroups is stored in the list widget ".list", and the scroll bar is stored in the scrollbar widget ".scroll". So for example, if you want to change the foreground and background colors of the newsgroup list, you can say:

```
.list config -bg honeydew1 -fg orchid2
```

These can also be controlled by the X resource database as well. However, the configuration file allows arbitrarily complex commands to be evaluated rather than simple assignments.

Certain Tcl/Tk commands can disrupt proper function of tknewsbiff. These will probably be obvious to anyone who knows enough to give these commands in the first place. As a simple example, the program assumes the font in the list box is of fixed width. The newsgroups will likely not align if you use a variable-width font.

The following variables are accessible and can be used for esoteric uses. All other variables are private. Private variables and commands begin with "_" so you don’t need to worry about accidental collisions.

The array "db" is a database which maintains information about read and unread news. db($newsgroup,hi) is the highest article that exists. db($newsgroup,seen) is the highest article that you have read.

A number of lists maintain interesting information. "active_list" is a list of known newsgroups. "seen_list" is a list of newsgroups that have been seen so far as the -new and -display flags are being processed. "previous_seen_list" is "seen_list" from the previous cycle. "ignore_list" is the list of newsgroup patterns to ignore. "watch_list" is the list of newsgroup patterns to watch. "display_list" is the list of newsgroup will be displayed at the next opportunity.

UPDATING YOUR FILES

tknewsbiff automatically rereads your configuration file each time it wakes up to check for unread news. To force tknewsbiff to reread the file immediately (such as if you are testing a new configuration or have just modified your newsrc file), press button 2 in the display (see BUTTONS above).

CAVEATS

tknewsbiff defines the number of unread articles as the highest existing article minus the highest article that you've read. So if you've read the last article in the newsgroup but no others, tknewsbiff thinks there are no unread articles. (It's impossible to do any better by reading the active file and it would be very time consuming to do this more accurately via NNTP since servers provide no efficient way of reporting their own holes in the newsgroups.) Fortunately, this definition is considered a feature by most people. It allows you to read articles and then mark them "unread" but not have tknewsbiff continue telling you that they are unread.

UNWARRANTED CONCERNS

Your news administrator may wonder if many people using tknewsbiff severely impact an NNTP server. In fact, the impact is negligible even when the delay is very low. To gather all the information it needs, tknewsbiff uses a single NNTP query - it just asks for the active file. The NNTP server does no computation, formatting, etc, it just sends the file. All the interesting processing happens locally in the tknewsbiff program itself.

BUGS

The man page is longer than the program.

SEE ALSO

"Exploring Expect: A Tcl-Based Toolkit for Automating Interactive Programs" by Don Libes, O'Reilly and Associates, January 1995.

AUTHOR

Don Libes, National Institute of Standards and Technology