

fvwm-mode

Bert 'theBlackDragon' Geens

24 April 2006

Contents

1	Introduction	2
2	Features	2
3	Initial setup	2
4	Usage	3
4.1	Completing commands and keywords	3
4.2	Executing commands	3
4.3	Inserting skeletons	4
5	Configuration	5
5.1	Completion	5
5.2	Executing commands	5
5.3	Syntax hilighting	6
5.4	Indentation	6
5.5	Automatic timestamp insertion	6
6	Thanks to...	7

1 Introduction

Fvwm-mode is an Emacs major mode to help with the editing of FVWM configuration files.

If you searched a bit before bumping into this mode you will no doubt have found at least one other Emacs major mode to support editing FVWM configuration files, so why did I write a new one? Basically I was dissatisfied with the syntax highlighting this already existing mode provided and since I wanted to do something with Lisp anyway I decided to write my own. I think you will agree that, although not perfect, fvwm-mode's syntax highlighting is pretty good.

Why didn't I just enhance that other mode then? Well, that's a simple one, because I didn't understand everything used in there. Thus the chances of breaking things in ways that would make me lose days just to figure out what was going on were pretty big. I thus decided to start from scratch and implement only the thing I craved the most at that time: syntax highlighting. But it didn't end there...

2 Features

Fvwm-mode can do a lot of things, some of which are pretty essential, others you might barely use.

- syntax highlighting
- FVWM keyword completion
- insertion of skeletons for various FVWM constructs (functions, FvwmScripts, FvwmScript widgets,...)
- automatically adding of a timestamp to your configuration file
- indentation (still experimental and turned off by default)
- execution of individual FVWM commands from the minibuffer
- execution of marked regions
- execution of an external file containing FVWM commands

If you think of a feature that might be useful don't hesitate to drop me a line.

3 Initial setup

Before you can use fvwm-mode you'll have to "install" it, I will suppose that you already have some version of Emacs installed, fvwm-mode should work with GNU Emacs 21 and XEmacs 21, I haven't tested with older versions so those might work, or they might not.

You'll then need to acquire a version of fvwm-mode which is normally available here, you can either get the stable release or the latest "released" development version, the devel version should normally work on at least the latest (CVS) version of GNU Emacs as that is the version I generally use myself. The devel versions are released in the hopes of getting feedback from users as I can't test all possible configurations with all possible Emacs versions.

Once you have a version of fvwm-mode.el you'll need to put it somewhere, I keep all my Emacs extensions in a */.elisp* directory, you would thus add the following to automatically load fvwm-mode for FVWM configuration files:

```
(add-to-list 'load-path "~/elisp")

;; Load fvwm-mode on Emacs startup
(require 'fvwm-mode)

;; Automatically load fvwm-mode for fvwm configuration files
(setq auto-mode-alist
      (cons ("config" . fvwm-mode)
            (cons ("FvwmApplet-" . fvwm-mode)
                  (cons ("FvwmScript-" . fvwm-mode)
                        auto-mode-alist))))
```

4 Usage

4.1 Completing commands and keywords

You can complete the current expression by pressing M-Tab (or Esc-Tab if M-Tab is already bound to something else), this will either complete the keyword or popup a temporary buffer with available completions, you can press M-Tab multiple times while adding letters until you get a unique match, which will then be inserted at point in the current buffer.

4.2 Executing commands

There are three ways to execute commands in fvwm-mode:

<i>C-c e c</i>	fvwm-execute-command
<i>C-c e f</i>	fvwm-execute-file
<i>C-c e r</i>	fvwm-execute-region

All of these require you to have FvwmCommand running in FVWM, see *man FvwmCommand* for details, but basic setup goes like this:

```
AddToFunc StartFunction I Module FvwmCommandS
```

Note that the 'S' is *not* a typo, it really is *FvwmCommandS*, even though the module is called *FvwmCommand*.

The function *fvwm-execute-command* will prompt you for an FVWM command to execute in the minibuffer and will run that, there unfortunately is no way to know if your command succeeded or failed, you will have to open the FVWM log (by default in */.xsession-errors*) for any output your command might produce.

The *fvwm-execute-file* function will prompt you for a file to run in the minibuffer and then attempt to run the FVWM commands in that file.

The last function to execute commands, *fvwm-execute-region*, is probably the most used of the three, it allows you to execute an arbitrary region of FVWM commands all at once right from your FVWM file, keep in mind that this command always executes whole lines, so putting the mark or point halfway a line will result in that entire line being executed (which is what you would normally want anyway)

4.3 Inserting skeletons

Fvwm-mode proved the possibility to insert skeletons for much used, or not much used functionality, either saving you typework or sparing you the trouble of looking the exact syntax of an FvwmScript up yet again.

<i>C-c i b</i>	<i>fvwm-insert-buttons</i>
<i>C-c i f</i>	<i>fvwm-insert-function</i>
<i>C-c i m</i>	<i>fvwm-insert-menu</i>
	<i>fvwm-script-insert-skeleton</i>
<i>C-c i w</i>	<i>fvwm-script-insert-widget</i>

The result of using *fvwm-script-insert-skeleton*, inputting "Test" as the title and leaving all the other options at their defaults looks something like this:

```
#--fvwm--
WindowTitle { Test }
Init
Begin

End

PeriodicTasks
Begin

End
```

There is no keybinding for this function as it's something you won't be using that much.

Inserting a new FvwmScript widget with *fvwm-script-insert-widget* (*C-c i w*) yields the following

```
Widget 1
```

```

Property
  Position 0 0
  Size 100 50
  Type ItemDraw
  Title {}
Main
  Case message of
    SingleClick :
      Begin

      End
End

```

The other commands give similar output for their respective subjects, the exact output depends on the input you do (or don't) supply.

5 Configuration

5.1 Completion

If you are using a version of GNU Emacs that supports pcomplete you can safely skip this section ¹

If you are using XEmacs and don't have pcomplete support ² fvwm-mode will fall back to the old completion mechanism which has some drawbacks that might make it worthwhile to either switch to GNU Emacs or get pcomplete from somewhere (if that's possible), one of these is that the generation of the completion list takes some time, this can be very annoying when you are busy editing a configuration file. You can make this more bearable by setting the variable *fvwm-preload-completions* to *t* which will generate the completions list when the mode gets loaded and not when you first use completion as is the default.

You can do it by adding this to your .emacs file:

```
(setq fvwm-preload-completions t)
```

It should be noted that the completion lists are built each time you open a file with fvwm-mode, which isn't exactly optimal. If anybody knows a fix for this, or even better, a decent completion mechanism for XEmacs please let me know.

5.2 Executing commands

This should normally work out of the box, but should you have installed FVWM in a location different from the default you will need to make sure that the variable *fvwm-fvwmcommand-path* points to the correct binary.

¹GNU Emacs 21 and 22 fall in this category, I don't know about older releases of GNU Emacs

²You can check by evaluating (*featurep 'pcomplete*), if it returns nil you don't have it

You can set this variable like this:

```
(setq fvwm-fvwmcommand-path "/somepath/FvwmCommand")
```

5.3 Syntax highlighting

There is one thing you can customize about the syntax highlighting without diving into fvwm-mode's internals and that's if keywords are highlighted in a case sensitive or case insensitive way. FVWM doesn't differentiate between the two but using CamelCase is seen as good style, but if you want to you can turn it off:

```
(setq fvwm-keywords-force-case nil)
```

5.4 Indentation

Indentation is still pretty experimental and incomplete, it might behave in unexpected ways, as such it is not turned on by default, should you want to turn it on to see how well it works add this to your *.emacs*:

```
(add-hook 'fvwm-mode-hook
          '(lambda ()
            (fvwm-enable-indentation)))
```

5.5 Automatic timestamp insertion

You can insert a timestamp in your file which gets updated everytime you save the modified file (nothing happens if the file hasn't been modified), there are a couple of steps involved to make this work and I'll try to describe those as exact as possible.

First you'll need to figure out what prefix you want to use for your timestamp, the prefix is a string that's *unique* (this is important) in your file and might look like this: "# Last edited on", which, in fact, is the default, this will be used to find the place the timestamp has to be inserted so it has to *exactly* match the text in your configuration file (your best bet is to just copy it over), you need to define it in the variable *fvwm-last-updated-prefix* like this:

```
(defvar fvwm-last-updated-prefix "# Last edited on")
```

You can also modify the format used to output the date/time in the timestamp by setting the variable *fvwm-time-format-string* like this:

```
(setq fvwm-time-format-string "%Y/%m/%d - %X")
```

The possible characters you can use in this time string can be found in the documentation of *format-time-string*, which you can read by typing *C-h f format-time-string*.

It's also possible to set an optional suffix, this for example allows you to create a box.

This code

```
(setq fvwm-last-updated-prefix "# Last edited on ")
(setq fvwm-time-format-string "%Y/%m/%d - %X")
(setq fvwm-last-updated-suffix
      "                                     #")
```

results in this in your FVWM configuration file:

```
#####
# Fvwm2 configuration file #
# written by Bert 'theBlackDragon' Geens <bert@NoSPAM.lair.be> #
# Last edited on 2006/04/24 - 23:40:04 #
#####
```

which is actually what I use in my configuration file (hence why they are the fvwm-mode defaults ;-)). You can clearly see the line with the timestamp that gets updated every time we edit and save our configuration file.

6 Thanks to...

Thomas Adam (*thomasadam*) for his excellent Vim syntax file from which I have borrowed most of the highlighting keywords.

Hannes Klas (*Hun*) for testing this mode with XEmacs and providing (E)Lisp coding advice.

Gramphos on the FVWM forums for pointing out shortcomings in the text.

Also thanks to the people in #emacs on Freenode for putting up with my sometimes really green ELisp questions.

And thanks to the people in #fvwm just for being such nice folks to talk to (now all of you download Emacs and enjoy this mode! ;)).

And thanks to all the people that use this piece of software, especially thanks to those who report bugs or suggest improvements, your input is very much appreciated!