一些有用的 emacs 配置

de an raccoon@gmail.com

2011年12月13日

目录

1	配置	emacs	2
	1.1	emacs 的滚动条放到右侧	2
	1.2	打开同名的 2 个不同文件	2
	1.3	gdb 和 term 退出后删除窗口	2
	1.4	隐藏菜单栏和工具栏	2
	1.5	快速切换 buffer	3
	1.6	不产生备份和修改状态栏	3
	1.7	标题栏显示文件路径	3
	1.8	成对显示括号, 但不来回弹跳	3
	1.9	使用 X 剪贴板	3
	1.10	自动 reload 文件	4
	1.11	搜索光标下字符串	4
2	ting	of emacs	6
4	-		6
			6
	2.2		6
		•	6
		2.2.3 emacs 集成 cscope	7
		2.2.4 hack xcscope.el 文件	7
		2 May 10 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	7
	2.3		8
			8
			8
			8
			9
	2.6	emacs 的一些启动项	9
3	关于	本文	9

1 配置 emacs $\cdot 2 \cdot$

1 配置 emacs

emacs 毕竟是老古董了,很多的默认配置总是各种别扭,个人总结了一些配置的方法,修改那些 烦人奇怪的配置

1.1 emacs 的滚动条放到右侧

```
emacs 默认的滚动条是在左侧, 现在放到右侧 (customize-set-variable 'scroll-bar-mode 'right)
```

1.2 打开同名的 2 个不同文件

如果打开 2 个同名的文件(在不同文件夹下),比如 django 的 url.py 文件,emacs 显示的名称分别是 url.py 和 url.py:1,通常分不清楚哪个是哪个文件可以用插件 uniquify

```
(require 'uniquify)
(setq
  uniquify-buffer-name-style 'post-forward
  uniquify-separator ":")
```

1.3 gdb 和 term 退出后删除窗口

在 emacs 中调用 term,gdb 退出后, emacs 不会自动关闭窗口, 这时候要才敲一下 C-x k 关闭这个 buffer, 是在是太傻了。这个配置可以让 gdb 和 term 退出时, 自动关闭

```
(defun kill-buffer-when-exit ()

"Close assotiated buffer when a process exited"

(let ((current-process (ignore-errors (get-buffer-process (current-buffer)))))

(when current-process

(set-process-sentinel current-process

(lambda (watch-process change-state)

(when (string-match "\\(finished\\|exited\\)" change-state)

(kill-buffer (process-buffer watch-process))))))))))
(add-hook 'gdb-mode-hook 'kill-buffer-when-exit)
(add-hook 'term-mode-hook 'kill-buffer-when-exit)
```

1.4 隐藏菜单栏和工具栏

```
; 隐藏菜单栏
(menu-bar-mode -1)
; 隐藏工具栏
(tool-bar-mode -1)
```

需要说明的是,如果你在运行中需要菜单栏,只要用命令

1 配置 emacs $\cdot 3 \cdot$

M-x menu-bar-mode

就可以显示菜单栏了,工具栏也同理

1.5 快速切换 buffer

用 C-x C-b 切换不同 buffer 实在是太麻烦了, 用 ibuffer 轻松实现 buffer 切换

```
;;ibuffer
(require 'ibuffer)
(global-set-key [(f4)] 'ibuffer)
把 ibuffer 绑定到 f4 上,轻松切换! TODO:picture
```

1.6 不产生备份和修改状态栏

```
(display-time-mode 1);显示时间
(setq display-time-24hr-format t); 24 小时格式
(setq display-time-day-and-date t);显示日期
(fset 'yes-or-no-p 'y-or-n-p);将 yes/no 替换为 y/n
(column-number-mode t);显示列号
(setq backup-inhibited t);;不产生备份
(setq auto-save-default nil); stop creating those #autosave# files
```

1.7 标题栏显示文件路径

```
(setq frame-title-format
'("%S" (buffer-file-name "%f"
(dired-directory dired-directory "%b"))))
```

1.8 成对显示括号,但不来回弹跳

```
(show-paren-mode t)
(setq show-paren-style 'parentheses)
```

1.9 使用 X 剪贴板

emacs 对中文的支持越来越好,对于 emacs23.2 来说一般的发行版的默认语言都是 utf8,以前配置 emacs 还需要配置剪贴版的语言,比如要配成 utf8 或者是 gbk(windows 还需要配置成 GBK),而在 opensuse 中,不需要特殊的命令,就可以保证中文的拷贝没有乱码

在 Linux 桌面中,有 2 种剪贴板,一种是 X 的剪贴板,另一种是 gnome 的剪贴板,gnome 剪贴板与 windows 类似,右键拷贝或者 Ctrl-c,Ctrl-v

X 的剪贴板就更加有意思,只要是鼠标选定高亮,就完成了复制,用鼠标的中键粘贴,完全省去了多余的按 Ctrl-c 的操作,也是传统的 Unix 拷贝和粘贴的方式.emacs 使用 X 的剪贴板,

1 配置 emacs $\cdot 4 \cdot$

与 X 共享一个剪贴板	使用键盘的对比如下在	emacs 中也可以用 X	的按键方式实现拷贝粘贴
3 41 / 1 1 / J / H 1 / A			

e	macs/X 按键对应关系	emacs 按键	X 按键
	选中文本	Mark-activate	用鼠标左键选中高亮
	复制文本	M-w	高亮的部分默认复制
	粘贴文本	С-у	点击鼠标中键, 粘贴到从鼠标指针开始的位置

;; 使用 X 剪贴板

(setq x-select-enable-clipboard t)

1.10 自动 reload 文件

如果 emacs 正在打开的文件,被其他程序修改了,为了保证在 emacs 中编辑的不会丢失,默认是不会自动 reload 新文件的,当需要 reload 新文件时,用命令

M-x revert-buffer

如果要配置 emacs 永远自动 reload 文件,增加

(global-auto-revert-mode)

1.11 搜索光标下字符串

在 vim 中,按*键可以搜索光标下文件,在 emacs 里面也有类似的按键:

Ctrl-s Ctrl-w

用过几次就会发现这个自动搜索有点儿傻,只认从当前光标到单词结尾,生生把一个完整的单词截断了,修改如下,让这个搜索聪明些!

(defun my-isearch-yank-word-or-char-from-beginning ()

"Move to beginning of word before yanking word in isearch-mode."

(interactive)

- ;; Making this work after a search string is entered by user
- ;; is too hard to do, so work only when search string is empty.
- (if (= 0 (length isearch-string))

(beginning-of-thing 'word))

(isearch-yank-word-or-char)

;; Revert to 'isearch-yank-word-or-char for subsequent calls

(substitute-key-definition 'my-isearch-yank-word-or-char-from-beginning

'isearch-yank-word-or-char

isearch-mode-map))

(add-hook 'isearch-mode-hook

(lambda ()

"Activate my customized Isearch word yank command."

(substitute-key-definition 'isearch-yank-word-or-char

'my-isearch-yank-word-or-char-from-beginning

1 配置 emacs $\cdot 5 \cdot$

isearch-mode-map)))

2 tips of emacs $\cdot 6 \cdot$

2 tips of emacs

2.1 emacs 的列模式

总的来说,感觉 emacs 的列模式没有 vim 的强大好用,不过基本的功能还是具备的,我常用的是下面两个快捷键 Ctrl-x r k 和 Ctrl-x r t。

- Ctrl-x r k 删除列模式高亮的部分
- Ctrl-x r t 在列模式高亮的部分插入文本

2.2 emacs 和 cscope

emacs 配合 cscope 浏览代码下相当方便,比 source insight 或者 slickedit 也不逊色,下面就简单介绍一下安装 cscope 和 emacs 的 cscope 插件

2.2.1 安装 cscope

用命令 zypper 安装 cscope

zypper in cscope

2.2.2 cscope 生成 index 文件

最一般的方式是在源代码目录运行 cscope -bR, -b 代表 build 源码的 index 数据库, -R 表示包括 子文件夹,cscope 在源代码的根目录生成 index 文件 cscope.out

但是在默认情况下, cscope 在建库时不会引入 cpp 文件, 如果是 cpp 的项目需要现找到所有的 c, cpp 文件, 然后再用 cscope 建立 index

```
#找到所有的 c,cpp,h 文件
find . -name "*.cpp" -o -name "*.c" -o -name "*.h" > cscope.files
# 根据 cscope.files 生成 database
cscope -b -i cscope.files
```

这样也太麻烦了把?! 其实大家都这么想,在 cscope 的源代码包中,在 contrib/xcscope 中有一个文件 cscope-indexer,就完成了上述功能,搜索文件夹,并且建 cscope 的 index,比如 opensuse 并没有把这个文件打到 cscope 的包中,所以建议大家自己去下载源代码包,把这个文件拷贝到/bin 下面,就可以直接用了。

这个文件比较简单,可以根据自己的需要 hack 它,比如我通常用 quilt 管理 patch,会在源代码目录生成/.pc/目录,里面是拷贝的 c 文件,我并不希望 cscope-indexer 去搜索这些文件,看了一下这个文件,无非就多加一个规则,虑除.pc 下所有文件

```
@@ -138,7 +138,7 @@
    fi
) | \
    egrep -i '\.([chly](xx|pp)*|cc|hh)$' | \
-    sed -e '/\/CVS\//d' -e '/\/RCS\//d' -e 's/^\.\//' | \
+    sed -e '/\/CVS\//d' -e '/\/RCS\//d' -e '/\/.pc\//d' -e 's/^\.\//'| \
```

2 tips of emacs $\cdot 7 \cdot$

2.2.3 emacs 集成 cscope

说了这么多,终于回到 emacs 了,直接从 contrib/xcscope/中拷贝 xcsope.el 到 Žemacs.d, 修改Žemacs

```
(add-to-list 'load-path "~/.emacs.d/")
(require 'xcscope)
(glabal-set=key [f1] 'cscope-index-files)
(global-set-key [f6] 'cscope-find-global-definition-no-prompting)
(global-set-key [f7] 'cscope-find-functions-calling-this-function)
(global-set-key [f8] 'cscope-find-this-symbol)
(global-set-key [S-f8] 'cscope-pop-mark)
```

快捷键可以根据自己的喜好定义, 先运行 M-x cscope-index-files, 生成代码的 index。用 cscope-find-global-definition-no-prompting 跳转到定义,用 cscope-find-this-symbol 搜索所有的引用

2.2.4 hack xcscope.el 文件

xcscope.el 也比较简单,很多配置项都写死了,没法修改,比如在跳出的 cscope 窗口中,选择要跳转到哪一个位置,有2种方式一个是光标移动到函数位置,按回车。令一个是用鼠标中键点击。

第 2 种方式是在是太奇怪了,我想把它改到鼠标左键上,修改 xcscope.el 文件,mouse-2 改成 mouse-1 就大功告成

```
(if cscope-running-in-xemacs
   (define-key cscope-list-entry-keymap [button2] 'cscope-mouse-select-entry-other-window)
- (define-key cscope-list-entry-keymap [mouse-2] 'cscope-mouse-select-entry-other-window))
+ (define-key cscope-list-entry-keymap [mouse-1] 'cscope-mouse-select-entry-other-window))
```

2.2.5 集成 cscope 窗口到 ECB 窗口

每次用 cscope 搜索的时候, 总是会新开一个窗口, 用完了也不会自动关闭, 对于屏幕小的同学, 很不方便, 白白占用了宝贵的空间, 配置 emacs, 让 cscope 的窗口始终在 ECB 窗口中显示, 既完成功能又节约空间假设现在你已经装了 ECB 和 CEDET(ECB 部分功能依赖 CEDET), 现在只需要修改.emacs

http://code.google.com/p/opensuse-topics/

2 tips of emacs $\cdot 8 \cdot$

ECB 有很多 layout, 现在选择自定义 layout 的方式, 把 cscope 窗口放到 ecb 的窗口中, 在左侧的右下角就是 cscope 专用窗口

```
$ ()
     ⊞ Includes
                                                                                                                                                               switch (state)
         B 🏡 _NMDevicePriv
→ 🖶 🗂 Attributes
                                                                                                                                                              case NM DEVICE STATE UNMANAGED:
                                                                                                                                2421
               □ Attributes
□ At
                                                                                                                                                                          if (old_state > NM_DEVICE_STATE_UNMANAGED)
    nm_device_take_down (device, TRUE, reason);
m@act_source_id: guint
apaipd_addr: guint
apaipd_pid: GPid
m@aipd_timeout: guint
apaipd_watch: guint
m@aipd_timeout: guint
m@aipd_timeout: guint
apaipd_config: NNOHCP4
apaipd_config: 
                                                                                                                                                               case NM DEVICE STATE UNAVAILABLE:
                                                                                                                             242€
242₹
242₹
                                                                                                                                                                          if (old_state == NM_DEVICE_STATE_UNMANAGED) {
    if (inm_device_bring_up (device, TRUE, &no_firmware) && no_firmware)
        nm_warning ("%s: firmware may be missing.", nm_device_get_iface (device));
                                                                                                                              2429
                                                                                                                                                                                                                                            so when the device needs to be
anges we actually deactivate it
                                                                                                                                                              case NM DEVICE STATE DISCONNECTED:
                                                                                                                               2432
                                                                                                                                                                         if (old_state |= NM_DEVICE_STATE_UNAVAILABLE)
nm_device_interface_deactivate (NM_DEVICE_INTERFACE (device), reason);
                                                                                                                               243:
                                                                                                                                                                           break;
                                                                                                                             243€
                                                                                                                                                              default:
   NetworkManager.h
NetworkManagerPolicy.c
nm-device-ethernet.c
                                                                                                                                                           }
                                                                                                                                                             g_object_notify (G_OBJECT (device), NM_DEVICE_INTERFACE_STATE);
g_signal_emit_by_name (device, "state-changed", state, old_state, reason);
    nm-device-interface.c
                                                                                                                               244€
                                                                                                                             2441
2442
2443
2444
2445
2446
2447
  nm-device.h
nm-serial-device.c
                                                                                                                                                             switch (state) {
case NM_DEVICE_STATE_ACTIVATED:
                                                                                                                                                                         nm_info ("Activation (%s) successful, device activated.", nm_device_get_iface (device));
nm_utils_call_dispatcher ("up", nm_act_request_get_connection (req), device, NULL);
                                                                                                                               2448
                                                                                                                                                              case NM_DEVICE_STATE_FAILED:
W-1 History
                                                                                                                                                                           nm_info ("Activation (%s) failed.", nm_device_get_iface (device));
                                                                                                                                 2451
                                                                                                                                                             priv-failed_to_disconnected_id = g_idle_add (failed_to_disconnected, device);
break;
default:
    Finding symbol: NMDeviceSta+→
    Database directory: /home/zl>
                                                                                                                                2455
                                                                                                                                                                         break;
         ** include/NetworkManager.l
                                                                                                                                                            }
                                                                                                                               2457
  global>[352]
                                                                                                                                                             if (old_state == NM_DEVICE_STATE_ACTIVATED)
 *** libnm-glib/nm-device.h:
                                                                                                                             2459
2466
                                                                                                                                                                           nm\_utils\_call\_dispatcher~("down",~nm\_act\_request\_get\_connection~(req),~device,~NULL);\\
                                                                                                                                                                                                                    the cached activation request *.
                                                                                                                                                             if (req)
                                                                                                                            2462
     <global>[40]
                                                                                                                             2463
                                                                                                                                                                           g_object_unref (req);
                                                                                                                               2464 }
                                              2465
2466 NMDeviceState
                                 >[85]
           global>[91]
nm-device.c 97% (2462,12) (C/l yas Abbrev)--四 5月 12 18:15 Mail-
```

2.3 cedect

2.3.1 基本 cedect 配置

2.3.2 其他

- M-x eassist-switch-h-cpp 快速切换 h, cpp 文件
- M-x eassist-list-methods 显示函数

2.4 frame looks

配置 frame 的默认属性,包括背景色,前景色,窗口位置大小,字体等等。

(setq default-frame-alist

```
'((background-color . "black")
(top . 20) (left . 40) (height . 40) (width . 128)
(font . "monospace-11")
(foreground-color . "#ffffff")))
```

3 关于本文 · 9 ·

2.5 只读模式

C-x,C-q

转为只读模式, 在看代码的时候很有用

2.6 emacs 的一些启动项

不带图形界面的启动

emacs -nw

默认启动,不加载用户的配置文件

emacs -q

启动 emacs 的 server 模式, emacs 在后台运行

emacs --daemon

以后用 emacsclient 启动 emacs 前端, 启动速度飞快, 赶上 vim

3 关于本文

项目地址	http://code.google.com/p/opensuse-topics
版权	GPLv2

欢迎访问项目网页浏览更多文档, 欢迎加入项目撰写文档!