

Configuration

Contents

| | | |
|----------|--|----------|
| 1 | Choix d'interface | 1 |
| 1.1 | Pas d'écran de bienvenue | 1 |
| 1.2 | Désactive des raccourcis | 1 |
| 1.3 | Augmente la mémoire pour le <i>garbage collector</i> \Rightarrow meilleures performances | 1 |
| 1.4 | <i>toolbar</i> | 1 |
| 1.5 | Réponses par y ou n | 1 |
| 1.6 | Commentaires | 1 |
| 1.7 | Désactive l'auto-save | 1 |
| 1.8 | Auto-fill | 1 |
| 1.9 | Suppression des blancs inutiles | 2 |
| 1.10 | Numéros de lignes | 2 |
| 1.11 | Try | 2 |
| 1.12 | Posframe | 2 |
| 1.13 | Which key | 2 |
| 1.14 | Gnuplot | 2 |
| 1.15 | Org mode | 3 |
| 1.16 | Ace window | 4 |
| 1.17 | Window-jump | 4 |
| 1.18 | Ivy / Counsel | 4 |
| 1.19 | Swiper | 5 |
| 1.20 | Avy | 6 |
| 1.21 | Company | 6 |
| 1.22 | Treemacs | 6 |
| 1.23 | C++ | 6 |
| 1.23.1 | Formatage automatique : clang-format | 8 |
| 1.23.2 | Coloration syntaxique (C++ moderne) | 8 |
| 1.24 | Thèmes | 9 |
| 1.25 | Modeline | 9 |
| 1.26 | Parenthèses arc-en-ciel | 9 |
| 1.27 | FlyCheck | 9 |
| 1.28 | Python | 10 |
| 1.29 | YASnippet | 10 |
| 1.30 | Divers paquets | 10 |
| 1.30.1 | Highlight line | 10 |
| 1.30.2 | Beacon | 10 |
| 1.30.3 | hungry-delete | 11 |
| 1.30.4 | Expand-region | 11 |
| 1.30.5 | Meilleure gestion du kill-ring | 11 |
| 1.30.6 | Gestion de la restauration des buffers | 11 |
| 1.31 | Powerline | 11 |
| 1.32 | iedit | 12 |
| 1.33 | Narrow/widen dwim | 12 |
| 1.34 | Web Mode | 12 |
| 1.35 | Emmet mode | 13 |
| 1.36 | Dired-dwim | 13 |
| 1.37 | L ^A T _E X | 13 |
| 1.38 | Gestion de projets | 13 |
| 1.39 | Org-mode (langages supportés) | 14 |

| | | |
|----------|--|-----------|
| 1.40 | Parenthèses | 14 |
| 1.41 | Taille de la police | 14 |
| 1.42 | Hydra | 15 |
| 1.43 | Modes git | 15 |
| 1.44 | FlySpell | 17 |
| 1.45 | Compilation | 17 |
| 1.46 | CMake | 18 |
| 1.47 | Markdown | 18 |
| 1.48 | Dumb jump | 18 |
| 1.49 | Origami | 18 |
| 1.50 | IBuffer | 19 |
| 1.51 | WGrep | 19 |
| 1.52 | PDF tools | 19 |
| 1.53 | AutoYASnippet | 20 |
| 1.54 | Divers | 20 |
| 1.55 | Keyfreq | 20 |
| 1.56 | Dictionnaire et césures | 20 |
| 1.57 | Mode pugs | 20 |
| 2 | Annexes | 21 |
| 2.1 | Génération de toute la documentation | 21 |

1 Choix d'interface

1.1 Pas d'écran de bienvenue

```
(setq inhibit-startup-message t)
```

1.2 Désactive des raccourcis

```
(global-set-key (kbd "C-x C-z") nil)
```

1.3 Augmente la mémoire pour le *garbage collector* ⇒ meilleures performances

```
(setq gc-cons-threshold (* 100 1024 1024)
      read-process-output-max (* 1024 1024))
```

1.4 *toolbar*

Elle est désactivée par défaut

```
(tool-bar-mode -1)
```

Elle est néanmoins lancée lorsque *gud* (l'interface à *gdb* dans Emacs) est actif.

```
(load-file "~/emacs.d/extra/tool-bar+.el")
(require 'tool-bar+)
(add-hook 'gud-mode-hook (lambda () (tool-bar-here-mode 1)))
```

1.5 Réponses par y ou n

```
(fset 'yes-or-no-p 'y-or-n-p)
```

1.6 Commentaires

| Raccourci | Description |
|-----------|-----------------------------------|
| C-c ; | Commente ou décommente une région |

```
(global-set-key (kbd "C-c ;") 'comment-or-uncomment-region)
```

1.7 Désactive l'auto-save

```
(setq auto-save-default nil)
```

1.8 Auto-fill

Utilise le mode mineur `auto-fill` (des retours à la ligne sont automatiquement ajoutés quand les lignes sont trop longues).

```
(turn-on-auto-fill)
```

... sauf quand on code

```
(add-hook 'prog-mode-hook (lambda () (auto-fill-mode -1)))
```

1.9 Suppression des blancs inutiles

On retire les blancs qui traînent en fin de ligne à la sauvegarde d'un *buffer*

```
(add-hook 'before-save-hook
  (lambda ()
    (when (not (derived-mode-p 'ein:notebook-multilang-mode))
      (delete-trailing-whitespace))))
```

1.10 Numéros de lignes

Affiche les numéros des lignes en marge de gauche.

```
(when (version<= "26.0.50" emacs-version)
  (progn
    (global-display-line-numbers-mode 'visual)
    (add-hook 'pdf-view-mode-hook (lambda () (display-line-numbers-mode -1)))
    (add-hook 'compilation-mode-hook (lambda () (display-line-numbers-mode -1)))
    (add-hook 'magit-mode-hook (lambda () (display-line-numbers-mode -1)))
    (add-hook 'ediff-display-help-hook (lambda () (display-line-numbers-mode -1)))
    (add-hook 'gud-mode-hook (lambda () (display-line-numbers-mode -1)))
    (add-hook 'speedbar-before-popup-hook (lambda () (display-line-numbers-mode -1)))))
```

1.11 Try

Permet d'essayer des paquets (sans les installer de manière permanente)

```
(use-package try
  :ensure t)
```

1.12 Posframe

Affiche des boîtes de dialogue

```
(use-package posframe
  :ensure t)
```

1.13 Which key

Aide en ligne pour les raccourcis (*quelle touche ?*)

```
(use-package which-key
  :ensure t
  :config
  (which-key-mode))
```

1.14 Gnuplot

Ajout du mode `gnuplot`, en particulier pour les interactions avec `Org-mode`

```
(use-package gnuplot
  :ensure t)
```

1.15 Org mode

Org bullets pour un plus bel affichage des sections

```
(use-package org
  :ensure t
  :pin org)

(setenv "BROWSER" "firefox")

(use-package org-bullets
  :ensure t
  :config
  (add-hook 'org-mode-hook (lambda () (org-bullets-mode 1)))
  (add-hook 'org-babel-after-execute-hook
    '(lambda () (org-redisplay-inline-images)))
  (add-to-list 'org-latex-packages-alist '("" "listingsutf8"))
  (add-to-list 'org-latex-packages-alist '("" "minted")))

(custom-set-variables
 ' (org-default-notes-file (concat org-directory "/notes.org"))
 ' (org-export-html-postamble nil)
 ' (org-hide-leading-stars t)
 ' (org-startup-folded (quote overview))
 ' (org-startup-indented t)
 ' (org-confirm-babel-evaluate nil)
 ' (org-src-fontify-natively t)
 ' (org-html-htmlize-output-type 'css)
 ' (org-latex-listings 'minted)
 ' (org-hide-emphasis-markers t))

(setq org-startup-with-inline-images t)

(setq org-latex-listings 'minted
  org-latex-packages-alist '("" "minted"))
  org-latex-pdf-process
  ' ("pdflatex -shell-escape -interaction nonstopmode -output-directory %o %f"
    "pdflatex -shell-escape -interaction nonstopmode -output-directory %o %f"))

(setq org-file-apps
  (append ' (
    ("\\.x?html?\\.\\'" . "/usr/bin/firefox %s")
  ) org-file-apps ))

(global-set-key "\C-ca" 'org-agenda)
(setq org-agenda-start-on-weekday nil)
(setq org-agenda-custom-commands
```

```

'(("c" "Simple agenda view"
  ((agenda "")
    (alltodo ""))))))

(global-set-key (kbd "C-c c") 'org-capture)

(defun make-capture-frame ()
  "Create a new frame and run org-capture."
  (interactive)
  (make-frame '((name . "capture"))))
  (select-frame-by-name "capture")
  (delete-other-windows)
  (noflet ((switch-to-buffer-other-window (buf) (switch-to-buffer buf)))
    (org-capture)))

(define-key org-mode-map (kbd "C-c >")
  (lambda () (interactive (org-time-stamp-inactive))))

(use-package htmlize :ensure t)

(setq org-ditaa-jar-path "/usr/share/ditaa/ditaa.jar")

```

1.16 Ace window

Permet de changer facilement de fenêtre. S'il y a plus de deux *buffers* leur numéros sont affichés et il suffit de taper le numéro de la fenêtre choisie pour s'y rendre.

| Raccourci | Description |
|-----------|--|
| C-x o | Met le curseur dans une autre fenêtre. |

```

(use-package ace-window
  :ensure t
  :init
  (progn
    (setq aw-scope 'global) ;; was frame
    (global-set-key (kbd "C-x O") 'other-frame)
    (global-set-key [remap other-window] 'ace-window)
    (custom-set-faces
     '(aw-leading-char-face
      ((t (:inherit ace-jump-face-foreground :height 3.0)))))
    ))

```

1.17 Window-jump

Permet de changer intuitivement de fenêtre en utilisant les flèches du clavier. S'il y a plusieurs possibilités, la fenêtre choisie est celle alignée avec le curseur.

| Raccourci | Description |
|-----------|--|
| C-M-right | Déplace le curseur dans une fenêtre à droite |
| C-M-left | Déplace le curseur dans une fenêtre à gauche |
| C-M-up | Déplace le curseur dans une fenêtre au dessus |
| C-M-down | Déplace le curseur dans une fenêtre en dessous |

```

(use-package window-jump
  :ensure t
  :bind (("C-M-<up>" . 'window-jump-up)
        ("C-M-<down>" . 'window-jump-down))

```

```
("C-M-<right>" . 'window-jump-right)
("C-M-<left>" . 'window-jump-left)))
```

1.18 Ivy / Counsel

Interface de complétion

```
(use-package counsel
  :ensure t
  :bind
  (("M-y" . counsel-yank-pop)
   :map ivy-minibuffer-map
   ("M-y" . ivy-next-line)))

(use-package ivy
  :ensure t
  :demand t
  :diminish (ivy-mode)
  :bind (("C-x b" . ivy-switch-buffer))
  :config
  (ivy-mode 1)
  (setq ivy-use-virtual-buffers t)
  (setq ivy-count-format "%d/%d ")
  (setq ivy-display-style 'fancy)
  (setq ivy-wrap t)
  ;; (custom-set-faces
  ;; '(ivy-minibuffer-match-face-1 ((t (:background "#74b5727c5654" :foreground "black"))))
  ;; '(ivy-minibuffer-match-face-2 ((t (:background "#ce8bf1006cba" :foreground "black"))))
  ;; '(ivy-minibuffer-match-face-3 ((t (:background "#fefdc4fe5941" :foreground "black"))))
  ;; '(ivy-minibuffer-match-face-4 ((t (:background "#fefdc40f85a" :foreground "black"))))
  ;; )
  )
```

1.19 Swiper

Améliore la recherche incrémentale. Les raccourcis sont les raccourcis classiques. Deux différences notables.

- SPC joue le rôle de joker : **r rc** correspond au mot *recherche* par exemple
- Si on souhaite créer un nouveau fichier avec **C-x C-f** dont le nom est une sous-chaîne d'un fichier existant, il ne faut pas utiliser **RET** pour valider, mais **C-M-j**. Utiliser **RET** valide la recherche, **C-M-j** valide la chaîne saisie. Il est également possible de sélectionner la zone de saisie (avec les flèches) et de simplement valider la chaîne avec **RET**.

| Raccourci | Description |
|----------------|---|
| C-s | Recherche incrémentale vers le bas |
| C-r | Recherche incrémentale vers le haut |
| M-x | Recherche incrémentale d'une commande |
| C-x C-f | Recherche incrémentale ou création d'un fichier |

```
(use-package swiper
  :ensure t
  :bind (("C-s" . swiper-isearch)
          ("C-r" . swiper-isearch)
          ("C-c C-r" . ivy-resume)
          ("M-x" . counsel-M-x)
          ("C-x C-f" . counsel-find-file))
  :config
  (progn
   (ivy-mode 1)
```

```

(setq ivy-use-virtual-buffers t)
(setq ivy-display-style 'fancy)
(setq ivy-use-selectable-prompt t)
(define-key read-expression-map (kbd "C-r") 'counsel-expression-history)
(define-key ivy-minibuffer-map (kbd "C-w") 'ivy-yank-word)
(custom-set-faces
 '(swiper-match-face-1 ((t (:background "#74b5727c5654" :foreground "black"))))
 '(swiper-match-face-2 ((t (:background "#ce8bf1006cba" :foreground "black"))))
 '(swiper-match-face-3 ((t (:background "#fefdc4fe5941" :foreground "black"))))
 '(swiper-match-face-4 ((t (:background "#fefd9c40f85a" :foreground "black"))))
)
)
)
)

```

1.20 Avy

Saute très rapidement vers la zone de texte contenant une lettre.

| Raccourci | Description |
|-----------|-------------------------------------|
| M-s | Demande une lettre du mot recherché |

```

(use-package avy
  :ensure t
  :bind ("M-s" . avy-goto-word-1))

```

1.21 Company

Boîtes de dialogue pour la complétion

```

(use-package company
  :ensure t
  :config
  (setq company-idle-delay 0)
  (setq company-minimum-prefix-length 2)

  (global-company-mode t))

(use-package company-box
  :ensure t
  :hook (company-mode . company-box-mode))

(defun my/python-mode-hook ()
  (add-to-list 'company-backends 'company-jedi))

(add-hook 'python-mode-hook 'my/python-mode-hook)
(use-package company-jedi
  :ensure t
  :config
  (add-hook 'python-mode-hook 'jedi:setup)
)

(defun my/python-mode-hook ()
  (add-to-list 'company-backends 'company-jedi))

(add-hook 'python-mode-hook 'my/python-mode-hook)

```

1.22 Treemacs

```
(use-package treemacs
  :ensure t)
```

1.23 C++

Transforme Emacs en un véritable IDE en utilisant `lsp`.

De nombreuses fonctionnalités sont accessibles au clic droit de la souris. On peut accéder à de nombreuses fonctionnalités en utilisant les raccourcis contextuels proposés par `hydra` en faisant `C-1`. En passant la souris sur le texte, un *popup* décrivant l'élément est affiché.

On rappelle quelques raccourcis utiles :

| Raccourci | Description |
|-------------------|---|
| M-. ou C-<mouse1> | Se rend à la déclaration de l'élément sous le curseur |
| M-? | Affiche les références à l'élément sous le curseur |
| C-1 | Début l'interaction <code>hydra</code> |
| C-<mouse3> | Menu contextuel |

```
(unless (package-installed-p 'use-package)
  (package-refresh-contents)
  (package-install 'use-package))
(eval-when-compile
  (require 'use-package))
(require 'bind-key)
(setq use-package-always-ensure t)

;; config basique de lsp
(use-package lsp-mode
  :init
  (setq lsp-keymap-prefix "C-c l")
  ;; on vire les recommandations de codage : un peu lourd
  (setq lsp-ui-sideline-show-code-actions nil)
  (setq lsp-modeline-code-actions-enable nil)
  ;; on vire l'affichage de la doc sous le curseur : trop invasif
  (setq lsp-ui-doc-show-with-cursor nil)
  ;; on vire les ajouts de code automatiques à la complétion : insupportable
  (setq lsp-completion-enable-additional-text-edit nil)
  ;; barre de localisation
  (setq lsp-headerline-breadcrumb-enable t)
  :hook ((c++-mode . lsp)
         (lsp-mode . lsp-enable-which-key-integration))
  :custom-face
  (lsp-face-highlight-textual ((t (:inherit highlight :underline t))))
  :commands lsp)

;; on utilise lsp-ui et ivy
(use-package lsp-ui :commands lsp-ui-mode)
(use-package lsp-ivy :commands lsp-ivy-workspace-symbol)

(use-package treemacs
  :ensure t)

(use-package lsp-treemacs :commands lsp-treemacs-errors-list)

;; autres paquets de base
(setq package-selected-packages '(yasnippet tramp company-lsp
  projectile hydra flycheck company avy which-key ivy-xref))

(when (cl-find-if-not #'package-installed-p package-selected-packages)
```



```
(package-refresh-contents)
(mapc #'package-install package-selected-packages))

(add-hook 'c-mode-hook 'lsp)
(add-hook 'c++-mode-hook 'lsp)
(setq lsp-keymap-prefix "C-c l")

; réactivité de lsp
(setq lsp-idle-delay 0.1)

(with-eval-after-load 'lsp-mode
  (add-hook 'lsp-mode-hook #'lsp-enable-which-key-integration)
  ;; (lsp-register-client
  ;; ;; *** Mettre le chemin complet vers clangd ***
  ;; (make-lsp-client :new-connection (lsp-tramp-connection "/home/test-emacs/bin/clangd")
  ;;
  ;; :major-modes '(c++-mode)
  ;; :remote? t
  ;; :server-id 'clangd-remote))
  )
```

1.23.1 Formatage automatique : clang-format

Création d'un raccourci spécial pour formater une zone du code.

| Raccourci | Description |
|-----------|--|
| C-c C-f | Indente la région comme définie par clang-format |

```
(use-package clang-format
  :ensure t
  :bind
  (("C-c C-f" . clang-format-region)))
```

Surcharge l'indentation classique d'Emacs par clang-format en C~/~C++. Si clang-format n'est pas installé ou désactivé, l'indentation classique fonctionne encore.

| Raccourci | Description |
|-----------|--|
| <TAB> | Indente la région comme définie par clang-format |

```
(defun clang-format-c-mode-common-hook ()
  (define-key c-mode-base-map (kbd "<tab>")
    (lambda ()
      (interactive)
      (if (use-region-p)
          (progn
            (c-indent-line-or-region (region-beginning) (region-end))
            (clang-format-region (region-beginning) (region-end))
          )
          (progn
            (c-indent-line-or-region)
            (clang-format-region (point) (point)))))))
(add-hook 'c-mode-common-hook 'clang-format-c-mode-common-hook)
```

Force le formatage du fichier quand il est sauvegardé

```
(defun clang-format-buffer-on-save ()
  "Add auto-save hook for clang-format-buffer-smart."
  (add-hook 'before-save-hook 'clang-format-buffer nil t))
(add-hook 'c-mode-common-hook 'clang-format-buffer-on-save)
```

1.23.2 Coloration syntaxique (C++ moderne)

```
(use-package modern-cpp-font-lock
  :ensure t
  :init
  (eval-when-compile
    ;; Silence missing function warnings
    (declare-function modern-c++-font-lock-global-mode
      "modern-cpp-font-lock.el"))
  :config (modern-c++-font-lock-global-mode t))
```

1.24 Thèmes

Chargement de quelques thèmes

```
(use-package cloud-theme :ensure t)
(use-package zenburn-theme :ensure t)
(use-package sourcerer-theme :ensure t)
(use-package monokai-theme :ensure t)
(use-package gruvbox-theme :ensure t)
(use-package bubbleberry-theme :ensure t)
(use-package solarized-theme :ensure t)
```

1.25 Modeline

Augmente la taille de la police

```
(defvar my-font-size 140)
```

Taille de la Modeline

```
(set-face-attribute 'mode-line nil :height my-font-size)
```

Taille du titre

```
(set-face-attribute 'header-line nil :height my-font-size)
```

Taille des attributs

```
(set-face-attribute 'default nil :height my-font-size)
```

Taille de la fenêtre et position

```
(setq default-frame-alist
  '((top . 0) (left . 0) ;; position
    (width . 110) (height . 90) ;; size
  ))
```

1.26 Parenthèses arc-en-ciel

Améliore le visuel des parenthèses

```
(use-package rainbow-delimiters
  :ensure t
  :init
  (eval-when-compile
    ;; Silence missing function warnings
    (declare-function rainbow-delimiters-mode "rainbow-delimiters.el"))
  (add-hook 'prog-mode-hook #'rainbow-delimiters-mode))
```

1.27 FlyCheck

On charge FlyCheck

```
(use-package flycheck
  :ensure t
  :init
  (global-flycheck-mode t))
```

Active FlyCheck globalement

```
(add-hook 'after-init-hook #'global-flycheck-mode)
```

On utilise posframe pour afficher les erreurs en ligne

```
(use-package flycheck-posframe
  :ensure t
  :init
  (flycheck-posframe-configure-pretty-defaults)
  :config
  (add-hook 'flycheck-mode-hook #'flycheck-posframe-mode)
  (setq flycheck-posframe-position 'window-bottom-left-corner))
```

1.28 Python

Attention pour que la configuration fonctionne, il faut installer `virtualenv`. Pour les systèmes Debian

```
apt install virtualenv
```

```
(setq py-python-command "python3")
(setq python-shell-interpreter "python3")

(use-package elpy
  :ensure t
  :custom (elpy-rpc-backend "jedi")
  :config
  (elpy-enable))

(use-package virtualenvwrapper
  :ensure t
  :config
  (venv-initialize-interactive-shells)
  (venv-initialize-eshell))
```

1.29 YASnippet

Le système de templates pour Emacs

```
(use-package yasnippet
  :ensure t
  :init
  (yas-global-mode 1)
  (unbind-key "<tab>" yas-minor-mode-map)
  :bind ("C-<return>" . yas-expand))

(use-package yasnippet-snippets
  :ensure t)
```

1.30 Divers paquets

1.30.1 Highlight line

Mise en valeur de la ligne courante

```
(global-hl-line-mode t)
```

1.30.2 Beacon

Flash de la ligne aux changements de page ou de buffer

```
(use-package beacon
  :ensure t
  :config
  (beacon-mode 1))
```

1.30.3 hungry-delete

Suppression de tous les blancs quand appuie sur backspace ou delete

```
(use-package hungry-delete
  :ensure t
  :config
  (global-hungry-delete-mode))
```

1.30.4 Expand-region

Sélectionne une zone de manière incrémentale. mot, phrase, paragraphe, etc de manière intelligente.

```
(use-package expand-region
  :ensure t
  :config
  (global-set-key (kbd "C-=") 'er/expand-region))
```

1.30.5 Meilleure gestion du kill-ring

```
(setq save-interprogram-paste-before-kill t)
```

1.30.6 Gestion de la restauration des buffers

```
(global-auto-revert-mode 1)
(setq auto-revert-verbose nil)
(setq revert-without-query (quote ("*.pdf")))
(global-set-key (kbd "<f5>") 'revert-buffer)
```

1.31 Powerline

Un bel affichage pour la barre d'état (*modeline*)

```
(use-package powerline
  :ensure t
  :config
  (powerline-default-theme)
  (setq powerline-default-separator 'utf-8)
  (setq powerline-gui-use-vcs-glyph 1))
```

Remplace le nom du mode majeur par une icône si possible.

```
(use-package mode-icons
  :ensure t
  :config (mode-icons-mode 1))
```

N'affiche pas tous les modes mineurs, les place dans le menu de la *modeline* désigné par un *smiley*.

```
(use-package minions
  :ensure t
  :config (minions-mode 1)
  (defpowerline powerline-major-mode "")
  (defpowerline powerline-process "")
  (defpowerline powerline-minor-modes minions-mode-line-modes))
```

1.32 iedit

Modifie les copies d'une zone sélectionnée simultanément

| Raccourci | Description |
|-----------|----------------------------|
| C-h C-; | démarrer les modifications |
| M-ESC ESC | sortir du mode |

```
(use-package iedit
  :ensure t)
```

1.33 Narrow/widen dwim

Réduit/agrandit une zone d'édition de manière intelligente

```
(defun narrow-or-widen-dwim (p)
  "If the buffer is narrowed, it widens. Otherwise, it narrows intelligently.
Intelligently means: region, org-src-block, org-subtree, or defun,
whichever applies first.
Narrowing to org-src-block actually calls `org-edit-src-code'."

  With prefix P, don't widen, just narrow even if buffer is already
  narrowed."
  (interactive "P")
  (declare (interactive-only))
  (cond ((and (buffer-narrowed-p) (not p)) (widen))
        ((region-active-p)
         (narrow-to-region (region-beginning) (region-end)))
        ((derived-mode-p 'org-mode)
         ;; `org-edit-src-code' is not a real narrowing command.
         ;; Remove this first conditional if you don't want it.
         (cond ((ignore-errors (org-edit-src-code))
                  (delete-other-windows))
               ((org-at-block-p)
                  (org-narrow-to-block))
               (t (org-narrow-to-subtree))))
        (t (narrow-to-defun))))
```

Remplace la fonction Emacs standard par `dwim`

```
(define-key ctl-x-map "n" #'narrow-or-widen-dwim)
(add-hook 'LaTeX-mode-hook
  (lambda ()
    (define-key LaTeX-mode-map "\C-xn"
      nil)))
```

1.34 Web Mode

Mode avancé pour l'édition de pages HTML, CSS,...

```
(use-package web-mode
  :ensure t
  :config
  (add-to-list 'auto-mode-alist '("\\.html?\\\\" . web-mode))
  (add-to-list 'auto-mode-alist '("\\.vue?\\\\" . web-mode))
  (setq web-mode-engines-alist
    '(("django" . "\\html\\\\")))
  (setq web-mode-ac-sources-alist
    '(("css" . (ac-source-css-property))
      ("vue" . (ac-source-words-in-buffer ac-source-abbrev))
      ("html" . (ac-source-words-in-buffer ac-source-abbrev))))
  (setq web-mode-enable-auto-closing t))
(setq web-mode-enable-auto-quoting t) ; this fixes the quote problem I mentioned
```

1.35 Emmet mode

Ensemble de fonctions permettant l'édition rapide de *markup languages* (HTML, SGML,...)

```
(use-package emmet-mode
  :ensure t
  :config
  ;; Auto-start on any markup modes
  (add-hook 'sgml-mode-hook 'emmet-mode)
  ;; Auto-start on any markup modes
  (add-hook 'web-mode-hook 'emmet-mode)
  ;; enable Emmet's css abbreviation
  (add-hook 'css-mode-hook 'emmet-mode))
```

1.36 Dired-dwim

Permet d'utiliser réduire/agrandir les affichages de répertoires

```
(setq dired-dwim-target t)

(use-package dired-narrow
  :ensure t
  :config
  (bind-key "C-c C-n" #'dired-narrow)
  (bind-key "C-c C-f" #'dired-narrow-fuzzy)
  (bind-key "C-x C-N" #'dired-narrow-regexp))

(use-package dired-subtree :ensure t
  :after dired
  :config
  (bind-key "<tab>" #'dired-subtree-toggle dired-mode-map)
  (bind-key "<backtab>" #'dired-subtree-cycle dired-mode-map))
```

1.37 L^AT_EX

Configuration L^AT_EX basique.

```
(use-package tex
  :ensure auctex
  :init
  (add-hook 'LaTeX-mode-hook 'turn-on-reftex)
  (setq TeX-parse-self t))
```

```
:config
(add-hook 'reftex-mode-hook
  '(lambda ()
    (define-key reftex-mode-map (kbd "C-c [")
      (lambda ()
        (interactive)
        (let ((ivy-count-format "'=' for all | %d/%d "))
          (reftex-citation)
          )))))
```

1.38 Gestion de projets

```
(use-package projectile
  :ensure t
  :bind ("C-c p" . projectile-command-map)
  :config
  (projectile-mode)
  (setq projectile-completion-system 'ivy))
```

1.39 Org-mode (langages suportés)

Charge les langages disponibles pour org-mode

```
(org-babel-do-load-languages
 'org-babel-load-languages
 '((python . t)
  (emacs-lisp . t)
  (shell . t)
  (C . t)
  (gnuplot . t)
  (js . t)
  (ditaa . t)
  (dot . t)
  (org . t)
  (latex . t )))
(setq org-babel-python-command "python3")
(add-hook 'org-babel-after-execute-hook
  'org-redisplay-inline-images)
```

1.40 Parenthèses

Gestion intelligente des parenthèses.

Attention les guillemets sont automatiquement échappés à l'intérieur de chaînes de caractères, ce qui peut être gênant pour l'utilisateur non averti. Pour couper une chaîne en deux morceaux, utiliser alors le raccourci suivant.

| Raccourci | Description |
|-----------|---------------------------------------|
| C-" | Coupe la chaîne de caractères en deux |

```
(use-package smartparens
  :ensure t
  :hook (prog-mode . smartparens-mode)
  :custom
  (sp-escape-quotes-after-insert nil)
  :config
  (require 'smartparens-config)
  (global-set-key (kbd "C-\\") 'sp-split-sexp)
)
```

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

1.41 Taille de la police

Change la taille de la police dynamiquement (temporaire)

| Raccourci | Description |
|-----------|---------------------------------|
| C-M-= | Augmente la taille de la police |
| C-M-- | Réduit la taille de la police |

```
(use-package default-text-scale
  :ensure t
  :config
  (global-set-key (kbd "C-M-=") 'default-text-scale-increase)
  (global-set-key (kbd "C-M--") 'default-text-scale-decrease))
```

1.42 Hydra

Outil de simplification des raccourcis. Un *popup* contextuel apparaît.

| Raccourci | Description |
|-----------|---------------------------------------|
| C-x t | Active désactive certains utilitaires |
| C-c t | Gestion du timer |

```
(use-package hydra
  :ensure hydra
  :init
  (global-set-key
   (kbd "C-x t")
   (defhydra toggle (:color blue :hint nil)
     "
     Toggle
     -----
     [_a_] Abbrev      [_s_] FlySpell      [_c_] FlyCheck
     [_d_] Debug       [_f_] Auto-Fill      [_t_] Truncate lines
     [_l_] Line num.   [_w_] Whitespace    [_q_] Quit"
     ("a" abbrev-mode)
     ("s" flyspell-mode)
     ("c" flycheck-mode)
     ("d" toggle-debug-on-error)
     ("f" auto-fill-mode)
     ("t" toggle-truncate-lines)
     ("l" display-line-numbers-mode)
     ("w" whitespace-mode)
     ("q" nil)))
  (global-set-key
   (kbd "C-c t")
   (defhydra hydra-global-org (:color blue)
     "Org"
     ("t" org-timer-start "Start Timer")
     ("s" org-timer-stop "Stop Timer")
     ("r" org-timer-set-timer "Set Timer") ; This one requires you be in an orgmode doc, as it sets the timer
     ("p" org-timer "Print Timer") ; output timer value to buffer
     ("w" (org-clock-in '(4)) "Clock-In") ; used with (org-clock-persistence-insinuate) (setq org-clock-persistence-insinuate t)
     ("o" org-clock-out "Clock-Out") ; you might also want (setq org-log-note-clock-out t)
     ("j" org-clock-goto "Clock Goto") ; global visit the clocked task
     ("c" org-capture "Capture") ; Don't forget to define the captures you want http://orgmode.org/manual-capturing.html
     ("l" (or )rg-capture-goto-last-stored "Last Capture"))
```


))

1.43 Modes git

Le sublime Magit.

| Raccourci | Description |
|-----------|---|
| C-x g | Démarre magit (M-x magit-status) |

```
(use-package magit
  :ensure t
  :init
  (progn
    (bind-key "C-x g" 'magit-status))
  :config
  (setq magit-display-buffer-function 'magit-display-buffer-same-window-except-diff-v1))
```

Affiche l'état git dans la marge?

```
(setq magit-status-margin
  '(nil "%Y-%m-%d %H:%M " magit-log-margin-width t 18))
```

Utilise **git-gutter** avec **hydra**. Permet de voir rapidement les modifications, de les valider (**git add -p**) ou de les annuler (**git checkout -p**)

| Raccourci | Description |
|-----------|--|
| M-g M-g | Lance le menu Hydra pour l'interaction git-gutter |

```
(use-package git-gutter
  :ensure t
  :init
  (global-git-gutter-mode +1)
  (add-hook 'magit-post-refresh-hook
    #'git-gutter:update-all-windows))

(global-set-key (kbd "M-g M-g") 'hydra-git-gutter/body)

(use-package git-timemachine
  :ensure t)
(defhydra hydra-git-gutter (:body-pre (git-gutter-mode 1)
  :hint nil)
  "
  Git gutter:
  _j_: next hunk      _s_tage hunk      _q_uit
  _k_: previous hunk  _r_evert hunk    _Q_uit and deactivate git-gutter
  ^ ^                _p_opup hunk
  _h_: first hunk
  _l_: last hunk      set start _R_evision
  "
  ("j" git-gutter:next-hunk)
  ("k" git-gutter:previous-hunk)
  ("h" (progn (goto-char (point-min))
    (git-gutter:next-hunk 1)))
  ("l" (progn (goto-char (point-min))
    (git-gutter:previous-hunk 1)))
  ("s" git-gutter:stage-hunk)
  ("r" git-gutter:revert-hunk)
  ("p" git-gutter:popup-hunk)
  ("R" git-gutter:set-start-revision)
  ("q" nil :color blue))
```

```

("Q" (progn (git-gutter-mode -1)
            ;; git-gutter-fringe doesn't seem to
            ;; clear the markup right away
            (sit-for 0.1)
            (git-gutter-mode))
:color blue))

```

1.44 FlySpell

Correcteur orthographique à la volée.

```

(add-hook 'LaTeX-mode-hook 'turn-on-flyspell)
(add-hook 'git-commit-setup-hook 'git-commit-turn-on-flyspell)
(add-hook 'c++-mode-hook 'flyspell-prog-mode)
(add-hook 'c-mode-hook 'flyspell-prog-mode)
(add-hook 'python-mode-hook 'flyspell-prog-mode)
(add-hook 'LaTeX-mode-hook 'turn-on-flyspell)
(add-hook 'org-mode-hook 'turn-on-flyspell)
(add-hook 'org-mode-hook 'turn-on-auto-fill)
(add-hook 'mu4e-compose-mode-hook 'turn-on-flyspell)
(add-hook 'mu4e-compose-mode-hook 'turn-on-auto-fill)
(setq flyspell-issue-message-flag nil)

```

1.45 Compilation

Active le rendu des couleurs ANSI dans le *buffer* de compilation

```

(defun endless/colorize-compilation ()
  "Colorize from `compilation-filter-start' to `point'."
  (let ((inhibit-read-only t))
    (ansi-color-apply-on-region
     compilation-filter-start (point))))
(add-hook 'compilation-filter-hook
  #'endless/colorize-compilation)

(add-to-list 'compilation-environment
  "LC_ALL=C")
(add-to-list 'compilation-environment
  "TERM=xterm-256color")

```

Force la création du *buffer* de compilation en dessous

```

(defun display-buffer-by-splitting-largest (buffer force-other-window)
  "Display buffer BUFFER by splitting the largest buffer vertically, except if
  there is already a window for it."
  (or (get-buffer-window buffer)
      (let ((new-win
              (with-selected-window (get-largest-window)
                (split-window-vertically))))
        (set-window-buffer new-win buffer)
        new-win)))
(defun my-compile ()
  "Ad-hoc display of compilation buffer."
  (interactive)
  (let ((display-buffer-function 'display-buffer-by-splitting-largest))
    (call-interactively 'compile)))

```

Quelques réglages supplémentaires.

```
(setq-default
 compilation-read-command t
 compilation-scroll-output 'first-error
 compilation-ask-about-save nil
 compilation-window-height 15)
```

Définit C-c C-c comme raccourci pour invoquer make.

```
(global-set-key (kbd "C-c C-c") 'compile)
(global-set-key (kbd "M-<up>") 'previous-error)
(global-set-key (kbd "M-<down>") 'next-error)
(defun compilation-c-mode-common-hook ()
  (define-key c-mode-base-map (kbd "C-c C-c") 'my-compile)
  (setq compilation-scroll-output 'first-error))
(add-hook 'c-mode-common-hook 'compilation-c-mode-common-hook)
```

1.46 CMake

```
(use-package cmake-mode
 :ensure t
 :mode ("CMakeLists.txt" ".cmake")
 :hook (cmake-mode . (lambda ()
                      (add-to-list 'company-backends 'company-cmake)))

:config
(use-package cmake-font-lock
 :ensure t
 :defer t
 :commands (cmake-font-lock-activate)
 :hook (cmake-mode . (lambda ()
                      (cmake-font-lock-activate)
                      (font-lock-add-keywords
                       nil '(((("\\<\\(FIXME\\|TODO\\|BUG\\|DONE\\)"
                               1 font-lock-warning-face t)))))))
```

1.47 Markdown

```
(use-package markdown-mode
 :ensure t
 :mode (".md" ".markdown"))
```

1.48 Dumb jump

Permet de se déplacer très rapidement dans un texte ou de retrouver une définition.

| Raccourci | Description |
|-----------|---|
| M-g j | Saute à la définition de l'objet sous le curseur |
| M-g o | Saute à la définition de l'objet sous le curseur dans une autre fenêtre |

```
(use-package dumb-jump
 :bind ((("M-g o" . dumb-jump-go-other-window)
         ("M-g j" . dumb-jump-go)
         ("M-g x" . dumb-jump-go-prefer-external)
         ("M-g z" . dumb-jump-go-prefer-external-other-window))

:init
(dumb-jump-mode)
:ensure t)
```

1.49 Origami

Mode permettant le pliage (*folding*) de régions

```
(use-package origami
  :ensure t)
```

1.50 IBuffer

Un meilleur gestionnaire de *buffers*.

| Raccourci | Description |
|-----------|---|
| C-x C-b | Ouvre le gestionnaire de <i>buffers</i> |
| C-x b | Change de <i>buffer</i> |

```
(global-set-key (kbd "C-x C-b") 'ibuffer)
(setq ibuffer-saved-filter-groups
  (quote (("default"
    ("dired" (mode . dired-mode))
    ("org" (name . "^.*org$"))
    ("magit" (mode . magit-mode))
    ("IRC" (or (mode . circe-channel-mode)
      (mode . circe-server-mode)))
    ("web" (or (mode . web-mode) (mode . js2-mode)))
    ("shell" (or (mode . eshell-mode) (mode . shell-mode)))
    ("mu4e" (or (mode . mu4e-compose-mode)
      (name . "\\*mu4e\\*")))
    ("programming" (or (mode . clojure-mode)
      (mode . clojurescript-mode)
      (mode . python-mode)
      (mode . cplusplus-mode)))
    ("emacs" (or (name . "\\*scratch\\*$")
      (name . "\\*Messages\\*$")))))
  )))
(add-hook 'ibuffer-mode-hook
  (lambda ()
    (ibuffer-auto-mode 1)
    (ibuffer-switch-to-saved-filter-groups "default")))

;; Don't show filter groups if there are no buffers in that group
(setq ibuffer-show-empty-filter-groups nil)

;; Don't ask for confirmation to delete marked buffers
(setq ibuffer-expert t)
```

1.51 WGrep

Permet de modifier le résultat d'un **grep** (donc simultanément dans plusieurs fichiers par exemple).

| Raccourci | Description |
|-----------|---|
| C-c C-p | passé en mode écriture dans le buffer résultat de grep |
| C-c C-e | sauve les modifications |
| C-x C-q | quitte le mode |

voir **wgrep** pour plus d'infos.

```
(use-package wgrep
  :ensure t)
(use-package wgrep-ag
```

```
:ensure t)
(require 'wgrep-ag)
```

1.52 PDF tools

Outils d'édition de PDF dans Emacs

```
(use-package pdf-tools
  :ensure t
  :config
  (when (eq system-type 'gnu/linux)
    (pdf-tools-install)
    (setq TeX-view-program-selection '((output-pdf "pdf-tools")))
    (setq TeX-view-program-list '(("pdf-tools" "TeX-pdf-tools-sync-view"))))
  :init
  (add-hook 'LaTeX-mode-hook
    '(lambda () (local-set-key (kbd "C-c C-g")
      'pdf-sync-forward-search))))

(use-package org-pdftools
  :ensure t)
```

1.53 AutoYASnippet

Outil de création rapide de snippets. Voir la documentation en ligne <https://github.com/abo-abo/auto-yasnippet>

```
(use-package auto-yasnippet
  :ensure t)
```

1.54 Divers

Quelques réglages utilitaires

```
(setq browse-url-browser-function 'browse-url-generic
      browse-url-generic-program "firefox")
(setq auto-window-vscroll nil)
(blink-cursor-mode t)
(setq default-frame-alist
      '((cursor-color . "DarkGrey")))
```

1.55 Keyfreq

Enregistre la fréquence d'utilisation de commandes. Pour obtenir les statistiques, utiliser la commande M-x keyfreq-show

```
(use-package keyfreq
  :ensure t
  :config
  (require 'keyfreq)
  (keyfreq-mode 1)
  (keyfreq-autosave-mode 1))
```

1.56 Dictionnaire et césures

```
(use-package dictionary
  :ensure t)
(use-package synosaurus
  :ensure t)
```

1.57 Mode pugs

```
(load-library "~/emacs.d/extra/pugs.el")
(add-to-list 'auto-mode-alist '("\\.pugs?\\'" . pugs-mode))
(require 'pugs-mode)
```

2 Annexes

2.1 Génération de toute la documentation

On génère la documentation HTML et PDF de cette configuration à partir de tous les fichier `.org` du répertoire de configuration Emacs.

```
(defun config-org-to-export()
  "Regenerate all .emacs.d docs"
  (interactive)
  (let ((files (file-expand-wildcards "~/emacs.d/*.org")))
    (mapc
     (lambda (f) .
       ((with-current-buffer
          (find-file-noselect f)
          (org-latex-export-to-pdf)
          (org-html-export-to-html))))
     files)))
```