HOWTO Débuter sous Emacs

Jeremy D. Zawodny, *jzawodn@wcnet.org* v1.7, 14 Octobre 1999

Ce document est une aide aux débutants sous l'éditeur Emacs. Il prend pour acquis la manipulation de `vi` ou d'un éditeur similaire. La dernière version de ce document est aussi disponible sur *http://www.wcnet.org/jzawodn/emacs/*

Contents

1 Introduction

1.1 Copyright

1.2 Public et Dessein

Ce document est destiné aux utilisateurs de Linux intéressés d'apprendre à utiliser Emacs et à le mettre en oeuvre. Ce document commença comme l'illustration d'un bref tutorial que j'ai été donné de présenter aux rencontres des utilisateurs de Linux de Tolède *http://www.talug.org/* . Il a quelque peu grossi depuis, suite aux retours que j'ai eu grâce aux membres de la communauté. Voyez la section remerciement pour les détails.

Dans ce document, rien n'est spécifique à Linux. Il s'applique à toutes les versions d'Emacs, même celle fonctionnant sous Microsoft Windows. Mais étant donné que ce document fait parti du projet de documentation Linux, je précise le fait qu'il a été développé pour les utilisateurs de Linux, car c'est pour eux qu'il a été écrit.

Enfin, pour ceux d'entre vous qui préfère le nom GNU/Linux au simple Linux, (allez lire *http://www.gnu.org/gnu/linux-and-gnu.html* pour voir de quoi il retourne) vous êtes les bienvenus à substituer mentalement GNU/Linux pour toutes les occurences de Linux dans ce document. Bien que je ne désaprouve pas le raisonnement et l'esprit contenu derrière cette idée, je ne me sens pas forcé d'écrire GNU/Linux.

1.3 Qu'est ce qu'Emacs?

Emacs possède différentes utilitées suivant les personnes qui l'utilisent. Suivant la personne à qui vous posez la question, vous pourrez obtenir l'une des réponses suivantes:

- Un éditeur de texte
- Un client de Messagerie
- Un lecteur de News
- Un traitement de texte
- Une religion / un sacerdoce
- Un environnement de développement intégré
- Tout ce que vous voulez qu'il soit!

Mais pour notre démonstration, prétendons seulement qu'il s'agit d'un éditeur de texte, un éditeur de texte étonnamment flexible. Nous approfondirons la question un peu plus tard. Emacs a été écrit par Richard Stallman (fondateur de la Free Software Fondation : *http://www.fsf.org/* et du projet GNU *http://www.gnu.org/*) et qui le maintient encore aujourd'hui.

Emacs est l'un des outils d'éditions de textes les plus populaires utilisés sur Linux (et Unix). C'est le second en popularité après **vi**. Il est aussi connu pour ses incommensurables possibilitées, sa capacité à être customizé, et son manque de bug (son absence de bug)! Ses larges possibilitées et sa capacité à être réellement configuré sont le résultat de la conception et de l'implémentation d'Emacs. Sans aller dans les détails, nous dirons qu'Emacs n'est simplement pas **juste un éditeur**. C'est un éditeur écrit principalement en **Lisp**. Dans le noyau d'Emacs il y a un interpréteur contenant toutes les subtilitées de Lisp écris en C. Seulement les parties les plus basiques et les pièces de bas-niveau d'Emacs sont écrites en C. La majorité de l'éditeur est en réalité écrit en Lisp. Donc, dans un sens, Emacs possède un langage de programmation entier incorporé, que vous pouvez utilisez pour customizer, pour étendre, et pour changer son comportement.

Emacs est aussi l'un des plus vieux éditeurs. Le fait est qu'il a été utilisé par des milliers de programmeurs durant les 20 dernières années signifie qu'il existe énormément de modules externes. Ces modules vous permettent de faire avec Emacs ce que Stallman n'a probablement jamais révé être possible de faire lorsqu'il a commencé à travailler sur Emacs. Nous parlerons de cela dans une prochaine section.

Il y a beaucoup d'autre sites Web et de documents qui donnent un meilleur apercu d'Emacs, de son histoire, et des sujets associés. Plutot que de tenter de reproduire tout cela ici, je suggère que vous voyez par vous même quelques endroits listés dans la section 6 (Autres Ressources) de ce document.

1.3.1 Portages et Versions

Cela vaut la peine de d'indiquer qu'il existe aujourd'hui deux éditeurs différents: GNU Emacs, et XEmacs. Les deux proviennent du même héritage et partageant quasiment les mêmes caractéristiques. Ce document se base sur GNU Emacs(plus précisement la version 20.3) mais la plupart de ce qui sera dit ici s'applique aussi à XEmacs et aux versions précedentes de GNU Emacs. Au cours de ce document, je ferais simplement référence à Emacs. Lorsque je le ferais, pensez à ce que je viens de vous dire.

1.3.2 Obtenir Emacs

Obtenir Emacs est simple. Si vous utilisez une distribution de Linux assez populaire, comme la debian, la RedHat ou la Slackware, ou n'importe quelles autres (NdT: comme la Mandrake), Emacs est un package optionnel que vous pouvez installer à partir du médium de distribution (NdT: CD-Rom, Disquette, réseau). Sinon, vous pouvez toujours obtenir les codes sources d'Emacs et les compiler vous mêmes. Allez sur le site web de GNU pour l'endroit précis où récupérer Emacs: *http://www.gnu.org/software/emacs/emacs.html*

2 Exécuter Emacs

2.1 Lancer & quitter Emacs

En tant que nouvel utilisateur, vous allez probablement vouloir lancer Emacs pour vous amuser un peu et l'essayer. Une fois dans Emacs, l'une des première choses à savoir, c'est: comment sortir, vous ne serez peut-être pas capable de savoir le faire. Donc, si vous n'avez jamais utilisé Emacs avant, lancer vous maintenant. Sur le prompt du shell, tapez `emacs` et appuyez sur entrée. Emacs devrais se lancer. Sinon c'est qu'il n'est soit pas installé, soit absent de votre chemin(PATH).

Une fois que vous voyez Emacs, vous devez savoir comment sortir. Les commandes pour quitter Emacs sont `C-x C-c`. La notation `C-x` signifie que vous devez appuyez sur la touche`Ctrl` et sur la touche`x`, en même temps. Dans ce cas ci, vous devrez ensuite appuyez sur les touches `Ctrl` et `c` toujours en même temps, pour quitter Emacs.

Les commandes au clavier utilisées par Emacs semblent être vraiment étranges, bizarre, et peut être même inconfortable pour vous de prime abord, spécialement si vous étes un utilisateur de `vi`. A la différence de `vi`, Emacs n'a pas de mode séparé pour éditer le texte, et executer des commandes.

Pour récapituler: `emacs` lance Emacs. `C-x C-c` termine Emacs.

2.1.1 Ce que vous allez voir

Quand Emacs se lance il prend la place entière de l'écran(en mode texte), ou bien d'une fenêtre. Vous verez alors un menu en haut, un peu de texte au milieu, et 2 lignes en bas de l'écran.

Cela ressemblera à cette ébauche en ASCII:

```
+----------------------------------------------------------------------+
|Buffers Files Tools Edit Search Mule Help                             |
|                                                                      |
|Welcome to GNU Emacs, one component of a Linux-based GNU system.      |
|                                                                      |
|                                                                      |
|                                                                      |
| ...                                                                  |
|                                                                      |
|---1:---F1  *scratch*         (Lisp Interaction)--L1--All-------------|
|For information about the GNU Project and its goals, type C-h C-p.    |
+----------------------------------------------------------------------+
```

NOTE: Emacs utilisera normalement la totalité de l'écran/fenêtre. J'ai réduit l'exemple ci-dessus dans un soucis de place. Vous aurez aussi un message de bienvenue dans Emacs lorsque vous le lancerez pour la première fois. Je l'ai aussi omis et l'ai remplacé par "`...`" à la place. Le message de bienvenue donne simplement la version courante d'Emacs que vous utilisez, ainsi que des liens vers la documentation en-ligne et des articles s'y rapportant.

La Barre de menu La ligne la plus en haut de l'interface d'Emacs est un menu. Si vous étes sous X, vous la reconnaitrez comme étant un menu traditionnel que vous pouvez utilisez en vous servant de votre pointeur. Sinon , vous devrez utiliser les raccourcis claviers(non expliqués ici) pour acceder aux menus.

La Barre de status & et le Mini-buffer Sur les deux dernières lignes de l'interface d'Emacs, la plus haute et principalement une barre de status. Elle contient des informations a propos du buffer sur lequel vous travaillez, dans quel mode Emacs se trouve, et d'autres informations variées sur d'autres choses. Pour l'instant, regardez juste ce qu'elle contient.

La plus basse des lignes est appelé **mini-buffer**. Elle est séparée du buffer principal par la barre de status dont nous venons juste de parler. Vous pouvez penser que le mini-buffer d'Emacs est une ligne de "commande". C'est ici que les commandes que vous donnez à Emacs apparaissent, et c'est aussi ici que les messages d'états sont imprimés en réponses aux actions que vous effectuez.

Vous trouverez que ce que je nomme la barre de status est courament nommé `mode line` dans la documentation d'Emacs. C'est ici qu'Emacs affiche les informations à propos des modes courants que vous utilisez, ainsi que des choses tels que l'heure et la date courantes, le numero de ligne, la taille du fichier, ainsi que tous ce que vous voudriez y voir afficher.

2.2 Quelques Terminologies

Cette section couvre les terminologies les plus simple d'Emacs que vous pourrez rencontrer quand lors de l'utilisation et des lectures à propos d'Emacs.

2.2.1 Tampons & Fichiers

A la différence d'autres éditeurs, quand vous ouvrez un fichier dans Emacs il ne reste pas ouvert tout le temps que vous travaillez dessus. A la place, Emacs lit celui-ci et le place dans un **tampon** en mémoire. Tant que vous éditerez le tampon et travaillerez avec les données rien ne sera changé sur le disque. Simplement, lorsque vous sauvegarderez le tampon que le fichier sur le disque sera modifié. Cette méthode possède des avantages et des inconvénients mais cela est important que vous compreniez qu'il fonctionne de cette manière.

Comme conséquence, vous verez le terme "buffer" (NdT : Tampon) utilisé dans la documentation Emacs , dans les differents modes, dans les packages, etc... Il faut juste réaliser que buffer signifie "une copie du fichier qui se situe en mémoire". Il faut aussi signaler que le buffer ne pointe pas toujours sur un fichier spécifique. Souvent Emacs crée des buffers comme résultat d'une commande que vous avez éxécuté, une liste de choix à effectuer, et ainsi de suite.

2.2.2 Points & Regions

Dans la terminologie d'Emacs, vous entendrez ou lirez souvent des références au sujet des **points**. En des termes simple le point signifie le curseur. La différence entre point et curseur n'est pas importante lorsque vous commencer avec Emacs. Mais si vous etes curieux, pensez-y de cette manière: le curseur est la réprésentation visuelle du point. Le curseur est toujours sur une position d'un caractère précis du buffer courant. Le point, d'un autre coté, évolue dans l'espace *entre les caractères* du buffer. Donc vous pouvez dire que si le curseur est sur la lettre 'e' dans le mot "les" alors le point est entre le 'l' et le 's'.

Comme la plupart des éditeurs modernes, Emacs permet d'éffectuer des opérations sur une section du buffer courant (identation, correction orthographique, reformatage, couper, coller, copier, ...). Vous pouvez marquer un bloc de texte en utilisant le clavier ou la souris et effectuer alors des opérations uniquement sur la zone séléctionnée. Dans Emacs, ces blocs des textes sont nommés **régions**.

2.2.3 Fenêtres

Cela sera peut-être un peu confus pour n'importe qui ayant utilisé une interface graphique avant. Il faut juste se rappeler qu'Emacs a été développé longtemps avant que les interfaces graphiques et les gestionnaires de fenêtres soit populaires.

Une **fenêtre** dans Emacs est une zone de l'écran dans laquelle un buffer est affiché. Quand Emacs est lancé, vous avez seulement une fenêtre affichée. Quelques fonctions d'Emacs, comme la documentation et l'aide, ouvrent souvent (temporairement) une fenêtre supplémentaire dans votre écran d'Emacs.

Les fenêtres d'Emacs n'ont rien en commun avec celle de X-Window en terme d'interface graphique. Vous pouvez ouvrir d'autres fenêtre X-Window pour afficher des buffers d'Emacs, par exemple pour comparer deux fichiers côte à côte. Ces nouvelles fenêtres X-Window sont référencées comme des **frames** dans les termes d'Emacs.

2.2.4 Frames

Dans Emacs, une **frame** est une fenêtre X-Window séparée dans laquelle un bufer d'Emacs est affiché. Mais les deux sont une partie de la même session d'Emacs. L'attitude est un peu la même que celle d'ouvrir plusieurs fenêtres d'Emacs lorsque vous tapez Alt+N.

2.3 Les commandes de bases au clavier

Cette section couvre les bases de la frappe au clavier pour Emacs. Comme tous les éditeurs puissant, tout ce qui peut être fait avec Emacs n'est qu'a quelques touches plus loin.

Si vous etes un utilisateur de `vi`, la notion d'utilisez les touches`k,j,l,h` pour remonter ou descendre d'une ligne, avancer ou reculer sur celle-ci vous a surement pris un peu de temps. En fait cela vous a surement pris quelques heures pour y être habitué, à pouvoir vous déplacer normalement dans un fichier en utilisant les différentes combinaisons de touches disponibles dans `vi`.

Emacs est différent. Il existe différentes combinaisons et commandes à apprendre. Comme `vi`, vous aurez uniquement à maitriser les commandes de bases pour pouvoir effectuer quasiment tous les travaux. Alors, au fil du temps, vous pourrez continer tranquillement l'apprentissage et trouver de meilleures voies pour arriver à vos fins.

2.3.1 Les touches de commandes(Meta, Esc, Control, and Alt)

Comme vous allez bientot l'apprendre, Emacs fait une énorme utilisation des combinaisons des touches multiples. Etant donné que ce n'est pas un éditeur possédant différents modes comme `vi`, vous n'aurez pas a penser en terme de mode de commande ou de mode d'édition avant de pouvoir bouger le curseur ou d'éxécuter une commande. A la place, vous aurez juste à presser la bonne combinaison de touches et Emacs fera ce que vous lui aura demandé(normalement :-)).

Les touches qu'Emacs utilise les plus sont le plus souvent les touches Control, référencée dans la documentation commme `C` et la touche Meta `M`. Comme peu d'ordinateur possède un clavier possédant cette touche, il faudra substituer à cette touche la touche Escape `Esc` ou la touche Alternate `Alt`. Dans la plupart des configurations les touches Esc et Alt font la même chose.

Donc lorsque vous verez une référence dans n'importe quelles documentation sur Emacs à `C-x f` cela signifie "appuyez sur control-x et ensuite f". Et si vous voyez un référence à quelque chose de la forme `M-x shell` cela signifie "appuyez sur alt-x et tapez le mot shell".

Une commande trés utile pour les débutants est la commande `M-x apropos` ou la commande `C-h a`. apropos cherchera dans la documentation Emacs en ligne de toutes les fonctions comportant l'expression régulière que vous tapé. C'est la manière la plus simple de découvrir toutes les commandes relatives aux frames. Tapez simplement `C-h a` et puis `frame`

2.3.2 Se déplacer dans un Buffer

Maintenant que vous savez ce que signifie toutes ces abbréviations fantaisistes signifient, voici la liste des commandes les plus populaires pour se déplacer dans un buffer.

```
Commandes
Clavier     Action
-----------------------------------
C-p         Monter d'une ligne
C-n         Descendre d'une ligne
C-f         Avancer d'un caractére
C-b         Reculer d'un caractère
C-a         Début de ligne
C-e         Fin de ligne
C-v         Descendre d'une page
M-v         Monter d'une page
```

```
M-f         Avancer d'un mot
M-b         Reculer d'un mot
M-<         Debut du buffer
M->         Fin du buffer
C-g         Finir l'opération courante
-----------------------------------
```

Et, comme vous pouviez vous y attendre, les flèches de direction fonctionnent comme vous vous l'esperiez. Votre touche `Backspace` peut-être pas. Mais c'est une autre histoire.

2.3.3 Commandes Principales q

Bon, maintenant que vous savez vous déplacer dans un buffer, qu'en est-il d'ouvrir et de fermer des fichiers, d'effectuer des recherches. Voici les commandes de base.

Avant de passer à ces commandes, je ferais le brievement le point sur son fonctionnement.

Toutes ces "commandes claviers" dans Emacs (celles avec `M-x qqle chose` ou `C-qqle chose`) ne sont juste que des raccourcis vers des fonctions internes a Emacs. Vous pouvez les appeler aussi en tapant `M-x nomdelafonction` et en appuyant sur `Entrée`. Vous pouvez aussi utiliser le raccourci clavier de cette fonction (si elle en a un).

Par exemple, la fonction Emacs qui sauvegarde le buffer vers le disque est appelée `save-buffer`. Par défaut, elle est aussi reliée à `C-x C-s`. Vous pouvez donc soit tapez `M-x save-buffer` et obtenir le même résultat.

Toutes les fonctions le plus communes possèdent un raccourci clavier par défaut. En voici quelques exemples.

```
Commandes
Clavier     Fonction             Description
-------------------------------------------------------------------
C-x C-s     save-buffer          Sauvegarde le buffer courant sur le disque
C-x u       undo                 Annule la commande précédente
C-c C-f     find-file            Ouvre un fichier
C-s         isearch-forward      Recherche en avancant une chaine de caractère
C-r         isearch-backward     Recherche en reculant une chaine de caractère
            replace-string       Recherche et remplace une chaine de caractère
            replace-regexp       Recherche et remplace une chaine de caractère en utilisant regexp
C-h t       help-with-tutorial Utilisation du tutorial
C-h f       describe-function    Affiche l'aide pour une fonction
C-h v       describe-variable    Affiche l'aide pour une variable
C-h x       describe-key         Affiche le comportement d'une combinaison de touches
C-h a       apropos              Recherche l'aide pour string/regexp
C-h F       view-emacs-FAQ       Affiche la FAQ d'Emacs
C-h i       info                 Affiche la documentation d'Emacs
C-x r m     bookmark-set         Configure un signet. Utile dans les recherches.
C-x r b     bookmark-jump        Déplacement vers un signet.
-------------------------------------------------------------------
```

Comme vous utiliserez beaucoup de ces fonctions, vous remarquerer que beaucoup vous demanderons quelque chose. Cela aura toujours rapport avec le mini-buffer. C'est similaire d'utiliser la commande `:` dans `vi` ou dans la plupart des commandes que vous utilisez avec votre shell favori.

Emacs possède des centaines de fonctions de base. La liste au dessus n'est qu'un trés petit exemple de ce que j'utilise régulièrement. Voyez la documentation en ligne pour une liste plus complète des fonctions disponible et une documentation complète sur ce qui est mentionné au dessus.

2.3.4 La Complétion par la touche Tab

Comme la plupart des shells unix (bash, csh, tsch, ...) Emacs offre la possibilité d'utiliser la complétion grace à la touche `Tab`. En fait, la commande de complétion du `bash` a été modifié après celle d'Emacs, donc si vous utilisez cette possibilité dans `bash` vous serez chez vous.

Pour exemple, essayer `M-x search` et appuyez sur `Tab`. Emacs ajoutera un trait d'union pour indiquer qu'il existe plusieurs possibilitées mais qu'elles possèdent toutes un trait d'union comme prochain caractère. Tapez la touche `Tab` une fois de plus pour qu'Emacs affiche les suites possibles dans lesquelles vous pourrez choisir. Prenez note du fait qu'il fait cela dans une nouvelle fenêtre: une contient le buffer courant que vous éditer en ce moment, l'autre contient la selection des complétions possibles pour "search-". Vous pourrez alors appuyer sur `C-g` pour sortir du processus de selection et fermer cette nouvelle fenêtre.

2.4 Tutorials, Aides, & Informations

Emacs est installé avec un tutorial en ligne qui vous aidera à utiliser les fonctiones d'édition de base et les fonctions que tout le monde doit connaitre. Il explique aussi comment utiliser les autres aides d'Emacs.

Je vous recommande vraiment de passer un peu de temps a travers ce tutorial si vous pensez à faire un serieux effort pour apprendre Emacs. Comme indiqué dans la table plus haut, vous pouvez entrer dans le tutorial par `C-h t`. Le tutorial est dirigé est ciblé pour les personnes qui commencent tout juste avec Emacs.

Si vous lancer Emacs sous X, Vous verez que le menu le plus en haut à droite est labelé Help (Aide). Lorsque vous explorerez le menu, vous verez que certaines rubriques possèdent des raccourcis et ceux ci sont listés dans la partie droite du menu.

Enfin, pour voir le volume de documentation disponible avec Emacs, vous pouvez essayer `M-x info` ou `C-h`> `i` qui lancera Info, le navigateur dans la documentation d'Emacs.

3 Les modes d'Emacs

Les modes d'Emacs proposent différentes caractéristiques et possibilités qui peuvent être activée(ou configurée, bien sur) pour être utilisée dans différentes circonstances. Ces modes aident a transformer l'éditeur (Emacs) pour en faire un outil utilisable autant pour écrire une documentation, programmer dans différents langages (C, C++, Perl, Java, et bien d'autres), créer une page Web, envoyer des emails, lire des news, garder une traces des ?appointements??, et même jouer.

Les modes d'Emacs sont simplement des librairies de code Lisp qui étendent, modifient, améliorent Emacs d'une certaine manière.

3.1 Modes majeur vs. modes mineur

Il existe deux types de modes différents disponible: le mode majeur et le mode mineur. La distinction n'est pas facile à voir sans avoir utilisé certains de ces modes, mais tracons en les grands traits.

Seul un mode majeur peut être actif à un instant donné. Les modes majeur sont des modes spécifiques à un langage, ou à une tache, tandis que les modes mineurs sont des petits utilitaires moins spécifique qui peuvent être utilisée avec nombre de taches différentes.

Cela ressemble à un résumé, donc voici un exemple. Il existe un que j'utilise assez souvent lorsque j'écris en format texte brut. Ce mode est appelé `text-mode`. Ce mode a été crée pour écrire du texte brut comme par exemple un fichier README. Il reconnait les mots et les paragraphes et normalement fait ce que je veux lorsque j'utilise les touches de direction standard

Quand j'écris un texte qui sera utilisé par d'autres personnes, je veux courament qu'il ait un aspect correct. Il doit être correctement cassé, etc... Pour activer le cassage du texte, je met en place le mode mineur `auto-fill` Ce mode essaye de faire les bons choix quand je tape et que je valide à la fin d'une ligne. Le fait est qu'étant donné que c'est un mode mineur, on peut l'utiliser avec beaucoup de mode majeur différents. Mes notions de bon choix à effectuer quand j'atteind la fin d'une ligne sont différentes que je soit dans le cas où je suis en mode texte `text-mode` du cas où je suis en java `java-mode` par exemple.Je ne veux pas que mon code java soit cassé comme un texte anglais. Mais je *veux* que les commentaires de mon code Java soit cassé. Le mode `autofill` est assez bon pour deviner quoi faire.

Les auteurs des différents mode d'emacs ont fait du bon travail en implémentant les caractéristiques mineures devant être dans des modes mineurs dans ces mêmes modes mineurs!

Si vous regardez la composition d'un écran d'Emacs, vous remarquerez que le mode ligne identifie le mode dans lequel se situe Emacs. Dans ce cas c'est le mode qui se nomme Lisp Interaction, c'est à dire le mode par défault. Il est réelement utile seulement lors de la frappe de code Lisp. (Mais puisque Emacs est écris en Lisp, pourquoi pas ?)

3.2 Les Modes de Programmations

D'abord et avant tout, Emacs a été concu par des programmeurs, pour des programmeurs (NdT: et des traducteurs...). Il existe des modes de trés grande qualité disponible pour chaqu'un des langages de programmation auquel vous pouvez pensez (et même d'autres moins populaires). Je ne peux que décrire brievement quelqu'uns d'entre eux ici. La plupart des modes de programmations partagent certaines caractéristiques. Communément, Il font la plupart des choses suivantes:

- Procure le surlignement par la couleur pour le langage.
- Procure une identation automatique et le formattage du code pour le langage.
- procure une aide contextuelle (langage) sensitive.
- Une interface automatique avec votre debogger.
- Ajoute un menu spécifique à votre langage sur la barre de menu.

En plus, il existe des modes non spécifiques à un langage qui aide à la réalisation de tâches commune à plusieurs langages. Ces tâches sont par exemple l'interfacage avec un contrôleur de version (RCS,CVS) , l'ajout automatique de commentaires, la création de makefile, la mise à jour des fichiers de changement(ChangeLog), etc...

Quand vous ajouter tout ces modes ensembles, et considérer la maturité et la stabilité du code d'Emacs, vous pouvez le comparer d'une manière plaisante à des applications IDE(Intergrated Developpement Environnement) pour des langages comme C++ et Java. Et, en plus, il est gratuit/libre.

3.2.1 C/C++/Java

A cause de leur syntaxte, le C, le C++ et le java sont assez simiaires, il existe donc un seul mode majeur pour gérer les trois langages (comme l'objective-C et l'IDL). C'est un package trés mature et complet et il est inclu dans la distribution d'Emacs. Ce mode est nommé `cc-mode` ou `CC Mode`

Pour plus de détails, ou pour télécharger une nouvelle version, aller à *http://www.python.org/emacs/* .

3.2.2 Perl

Il existe véritablement deux modes pour éditer du code Perl dans Emacs. Le premier est nommé `perl-mode` (comme vous pouviez l'éspèrer, et le second se nomme `cperl-mode`. Je n'ai pas de bonnes raisons pour lesquelles il existe deux modes (la documentation ne le dit pas, mais il semblerais que `perl-mode` soit le mode original pour éditer du code Perl dans Emacs. il semble avoir moins de possibilitées que `cperl-mode` et la reconnaisance de certaines possibilitées étranges du langage manque.

Personnalement, j'utilise et je recommande `cperl-mode` qui semble être mieux maintenu et possèdent toutes les caractéristiques que je demande. Vous pouvez trouver la dernière version ici: *ftp://ftp.math.ohio-state.edu/pub/users/ilya/emacs* .

Mais ne prenez pas mes mots comme paroles d'évangiles: essayez les deux et prenez celui qui vous convient le mieux.

3.2.3 Python

Python(un autre langage de script trés populaire possède aussi un mode Emacs. Pour autant que je puisse dire, il n'est pas distribué avec GNU Emacs, mais est distribué avec XEmacs. Il fonctionne malgré tout assez bien avec les deux éditeurs.

Vous pouvez obtenir le `python-mode` à partir du site web officiel de Python *http://www.python.org/emacs/python-mode/* .

3.2.4 Les autres

Il existe beaucoup d'autres modes d'éditions disponibles pour aider les programmeurs. De tel mode vous aideront à executer des tâches comme :

- Shell Scripts (Bash, sh, ksh, csh, ...)
- Awk, Sed, Tcl, ...
- Makefiles
- Change Logs
- Documentation
- Deboggage

Et bien plus. Voyez la dernière section de ce document pour plus d'informations pour trouver d'autres modes et des ajouts.

3.3 Authoring

Les modes d'Emacs `ne` sont `pas` limité aux seuls développeurs. Les personnes écrivant de la documentation peuvent bénéficier d'un vaste ensemble de modes Emacs.

3.3.1 Correction Orthographique (mode `ispell`)

Les auteurs de beaucoup de type de document on toujours besoin de correction orthographique. Si vous avez **GNU ispell** , vous pouvez taper `M-x ispell` et corriger le buffer courant. Si ispell trouve des mots qu'il ne connait pas, it affichera la liste des remplacements possible et vous laissera en choisir un (ou aucun) de ceux-ci. Il est équivalent à bien des outils de correction orthographique de logiciel non-libre.

3.3.2 HTML (mode `html-helper`)

Si vous vous retrouver à écrire des fichiers en HTML une fois (et même beaucoup) vous voudrez peut-être tester `html-helper-mode`. Il est disponible à l'adresse *http://www.santafe.edu/˜nelson/tools/* avec une documentation.

Comme son nom le suggère, `html-helper-mode` offre beaucoup de fonctions pour aider ceux qui écrivent du code HTML à la main, à l'ancienne mode.

3.3.3 TeX (`tex-mode`)

Lorsque vous écrivez des documents en TeX, il est souvent utile d'obtenir Emacs certains surlignage ou coloration pour les `/`, les accolades, et d'autres caractères. `tex-mode` s'occupera de cela pour vous.

Bien que je n'écrivent plus en TeX directement maintenant, lorsque je le faisais ce mode a été assez utile en transformant mon source Tex en un source un peu plus lisible.

3.3.4 SGML (`sgml-mode`)

Le document que vous êtes en train de lire a été écris en SGML (et probablement converti vers le format dans lesquel vous êtes en train de lire). `sgml-mode` procure toute les bases pour un document SGML: validation, surlignage, étiquetage avant/arrière, et bien d'autre choses. C'est une des parties standard d'Emacs.

3.4 Autres Modes

Bien sur, il existe des tas d'autres modes pratique pour rendre la vie facile. Voici juste un panel des plus populaire.

3.4.1 Le Contrôle de Version (mode `vc`)

Le mode `vc` s'interface avec les plus populaires contrôleurs de version(RCS, SCCS, CVS) pour simplifier le contrôle des fichiers, contrôler les release, etc... C'est un mode standard d'Emacs, documenté dans la doc d'Emacs.

3.4.2 Le Mode Shell

Pourquoi passer vers une autre fenêtre X, ou une console virtuelle, juste pour lancer quelques commandes? Faites le a partir d'Emacs, et garder vous des problèmes. `:-)`

`M-x shell` lancera un shell à l'interieur d'un buffer Emacs. Vous pourrez effectuer la plupart des tâches que vous effectuer sur un shell classique (a par des programmes tournant en plein écran tel que `vi` ou `pine` car Emacs dialogue avec votre shell réel dernière vous...

Cela fait aussi partie des standard d'Emacs, donc vous pourrez le trouver documenté dans la doc d'Emacs.

3.4.3 Telnet et FTP

Pourquoi passer vers une autre fenêtre ou une console virtuelle juste pour lancer telnet ou ftp? Faites le avec Emacs, et gagner du temps. (Vous avez remarqué la ressemblance ?)

Comme vous pouvez lancer un shell avec Emacs, vous pouvez aussi lancer telnet et ftp. Essayez `M-x telnet` ou `M-x ftp` pour en faire l'experience. voyez la documentation pour tous les détails.

3.4.4 Pages Man

Pourquoi passer vers une autre fenêtre ou une console virtuelle juste pour lire une page man? Faites le avec Emacs, et gagner du temps. (Promis, c'est fini.)

Comme vous pouvez lancer un shell avec Emacs, vous pouvez aussi lire les pages de manuel. Essayez `M-x man` pour en faire l'experience. voyez la documentation pour tous les détails.

3.4.5 Ange-FTP

Pour citer la documentation d'`ange-ftp`

> Ce package tente de rendre accessible des fichiers et des répertoires en utilisant FTP à l'intérieur d'Emacs aussi transparent que possible. Un sous ensemble du système de gestion de fichier a été étendu pour utiliser FTP.

Cela signifie que vous pouvez traiter des fichiers de machines distantes comme si elle étaient situé sur votre ordinateur. Donc si vous devez éditer un fichier sur différents ordinateurs, demander à Emacs de l'ouvrir (en utilisant une syntaxte différente pour le chemin) et vous occuper des détails de connection et de récuperer le fichier. Vous pourrez alors sauver le fichier via `C-x C-s`, et `ange-ftp` interceptera la sauvegarde et écrira le fichier sur la machine distante.

La syntaxte différente pour le chemin se présente ainsi. Un fichier nommé "monfichier", dans un répertoire "utilisateur", sur une machine nommée :"ma.machine.org" peut-être obtenu en ouvrant le fichier (`C-x f`)

```
/utilisateur@ma.machine.org:~utilisateur/monfichier
```

Etant un package standard de la distribution, vous pouvez aussi consulter la documentation pour tous les détails.

Merci à Etienne Grossmann (*etienne@anonimo.isr.ist.utl.pt*) pour l'exemple ci-dessus.

4 Customiser Emacs

Toutes les customisations d'Emacs peuvent virtuellement être effectué par du code Lisp. Vous pouvez modifier des variables qui influence la manière d'agir d'Emacs, ou vous pouvez ajouter de nouvelles fonctions à Emacs (ou effacer des fonction existantes, en les remplacant par les vôtres).

4.1 Customisation Temporaire

Lorsque vous vous habituerez à customiser Emacs, vous voudrez probablement le faire d'une manière temporaire. Si vous faites quelque chose de vraiment mauvais, vous pourrez sortir avec `C-x C-c` et le relancer. Une fois que vous aurez trouvé les changements qui vous plaisent, vous voudrez les rendre permanent, vous pourrez alors les ajouter à votre fichier `.emacs` pour qu'ils soient effectif à chaque lancement d'Emacs. Nous en parlerons dans la prochaine section.

4.1.1 Assignements de Variable

La plus simple des customisations à faire est de changer la valeur d'une variable d'Emacs. Le code pour effectuer cette action ressemble à celui-ci:

```
(setq nom-variable nouvelle-valeur)
```

Où `nom-variable` est le nom de la variable et `nouvelle-valeur`est la valeur que vous voulez affecter. (En langage Lisp, vous assignez une variable à une valeur.) La fonction `setq` est analogue à un opérateur d'assignement(courament =) dans d'autres langages de programmations.

NOTE: Je passe sur beaucoup de détail dans un soucis de simplicité. Vous me verez moi, ou d'autres utiliser la fonction Lisp `set` et même `setq-default`. Si vous êtes vraiment curieux, aller voir de quoi il retourne dans la réference Lisp d'Emacs.

Regarder cette ligne de mon fichier `.emacs`.

```
(setq-default transient-mark-mode t)
```

La Variable `transient-mark-mode` contrôle si une region doit être surligné ou lorsqu'on la marque. Dans beaucoup d'application graphique, lorsque vous cliquez et déplacez le pointeur pour selectionner une partie du texte, elle devient surlignée en mode inverse, ou dans une autre couleur. Emacs effectuera de même lorsque la variable `transient-mark-mode` aura une valeur non nulle d'affectée.

QUELLE valeur?

D'accord. Petite digression. Beaucoup de langage de programmation ont des notions de valeurs vrai/fausse (true/false). En C/C++ une valeur est considéré vrai (true) si elle possède une valuer non nulle. En Perl, une valeur non nulle ou différentes de zéro est vrai (true). En Lisp, la même idée s'applique, mais avec des noms et des symboles différents.

Vrai (True) est courament écris comme ceci `t` et faux (False) (ou Null) est écrit comme `nil`. Comme d'autres language, bien sur, toute valeur non nulle est considéré comme vrai.

Pour donner la description complète de ce que fait `transient-mark-mode`, vous pouvez utilisez l'aide en-ligne. Tapez `C-h v` ou `M-x describe-variable` puis `transient-mark-mode`. Si vous êtes fainéant comme moi, vous pourrez tirez avantage de la complétion des noms de variables en utilisant la touche `Tab` Tapez juste les premières lettres et appuyez sur la touche `Tab`. Si vous avez tapé assez de lettre pour qu'Emacs puisse l'identifier, vous verez apparaitre le mot entier complété pour vous.

Une autre variable que l'on utilise souvent est `fill-column`. Si vous dites à Emacs la largeur de l'écran, il effectura la casse des lignes (et `auto-fill-mode` respecte cette valeur). Pour affecter une valeur absurde vous pouvez tapez:

```
(setq fill-column 20)
```

Mais cela ne fait rien maintenant. Vous devrez demander à Emacs d'**évaluer** l'expression que vous avez tapé. Pour l'évaluer, placer le point (le curseur) à la fin de l'expression et tapez `C-x C-e`, qui appelera la fonction `eval-last-sexp`. Lorsque vous ferez cela, notez que 20 (ou ce que vous avez tapé) est renvoyé dans la mini-buffer en bas de l'écran. C'est la valeur de retour de l'expression que vous avez évalué.

Juste pour montrer que cela marche, taper une phrase ou deux. Si vous avez le mode `auto-fill-mode` activé (probablement pas) Vous remarquerez que le texte est cassé à la colonne 20. Autrement après avoir tapé quelque chose, tapez `M-q` qui appelera la fonction `fill-paragraph`. Il effectura alors la casse du paragraphe.

4.1.2 Associations de fichiers

Vous pouvez configurer Emacs pour qu'il effectue des tâches automatiques lorsque vous ouvrez des fichiers d'un certain type (comme certaines interfaces graphiques lance une application lorsque l'on clique sur un fichier précis). Par exemple, Je veux qu'Emacs passe en mode `text-mode` chaque fois que j'ouvre un fichier avec l'extension `.txt`. Bon, il le fait déjà. `:-)` Donc faisont le passer en mode `text-mode` lorsqu'il ouvre un fichier "README".

```
(setq auto-mode-alist (cons '("README" . text-mode) auto-mode-alist))
```

Ouch?

Sans plonger dans des flots de programmation Lisp que vous n'avez pas besoin de connaitre (mais cela ne vous fera pas de mal d'en apprendre plus), laisser juste vous expliquer que la variable `auto-mode-alist` contien une liste de paires . Chaque paires contient une expression régulière et un nom de mode Emacs. Si dans un fichier que vous ouvrez correspond à l'une des expressions régulières (dans ce cas, la chaine `README`) Emacs lance automatiquement le mode que vous avez spécifié.

La syntaxte amusante ;-) au dessus est du au fait que nous rajoutons une paire à ce mode. Vous ne voulez surement pas juste assigner à `auto-mode-alist` une valeur sans vous assurer que les valeurs contenues ne sont pas perdue.

Et si vous voulez qu'Emacs passe dans le mode `html-helper-mode` chaque fois que j'ouvre un fichier que se termine par `.html` ou `.htm`, je n'ai qu'à ajouter à mon fichier .emacs:

```
(setq auto-mode-alist (cons '("\\.html$" . html-helper-mode) auto-mode-alist))
(setq auto-mode-alist (cons '("\\.htm$"  . html-helper-mode) auto-mode-alist))
```

Les possibilités sont vraiment sans fin.

4.2 Utiliser un fichier `.emacs`

Après avoir passé quelque temps avec Emacs et avoir une idée de base de ce que l'on peut customiser, vous voudrez probablement changer quelques petites options de manière permanente. (ou au moins jusqu'à ce que vous ayez changé d'avis. Si vous utiliser Emacs tout les jours, vous trouverez que votre fichier .emacs grossis de plus en plus. C'est bon puisque cela signifie que vous avez compris comment faire fonctionner Emacs de la manière dont **vous** voulez. C'est triste de voir que peu de programmes permettent cette approche.

Dans le cas ou vous n'auriez pas encore deviné, chaque fois que vous lancez Emacs, il cherche dans le fichier `.emacs` dans votre répertoire home. Votre fichier `.emacs` est le fichier où tout le code que vous voulez lancer automatiquement et toutes sortes de customisations que nous avons vu précedement.

Un autre exemple de mon fichier `.emacs`:

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

La varialbe `inhibit-startup-message` contrôle si Emacs doit afficher le message de bienvenu ou non. Après un certain temps, je devenais malade en le voyant (car je savais où trouver l'aide et autres trucs) donc j'ai cherché une manière de le désactiver.

Comme exercice, essayer de créer un fichier `.emacs` par vous même et en ajoutant cette ligne. Sortez alors d'Emacs, et relancer le. Vous ne devriez plus voir le message de bienvenue.

Souvent, lorsque vous lisez quelque chose à propos d'un mode Emacs(ou d'un package), la documentation suggèrera d'ajouter des lignes à votre fichier `.emacs` pour pouvoir faire fonctionner le package ou le mode d'une manière précise.

La FAQ GNU Emacs (`C-h F`) contient quelques unes des options concernant le fichier `.emacs` que vous trouverez peut-être utile.

4.3 Le Package Customisé.

Au fur et à mesure du développement d'Emacs en popularité et son évolution, quelques un ont dis "il y doit y avoie une meilleure manière de permettre aux novices de customiser Emacs." Et `customize` était né.

Customize procure aux utilisateurs une manière plus intuitive de configurer des parties d'Emacs. Pour l'essayez, visitez le sous-menu `Customize` de votre menu `Help` , ou tapez `M-x customize`.

Customize groupe la configuration dans des groupes logiques, tels que “Editing”, “Programming”, “Files”, et ainsi de suite Quelques groupes contiennent des sous-groupes.

Si vous effectuer des changements en utilisant l'interface `Customize`, Emacs sauvegardera vos changements dans votre fichier `.emacs` C'est assez pratique, car cle code ajouté peut-être facilement inspecté (et changé)

Je n'utilise pas l'interface `Customize`, je ne peut donc pas vous en dire plus..

4.4 L'affichage X Windows

Comme n'importe quelle application X, Emacs respecte vos ressources X. Cela signifie que vous pouvez contrôler les couleurs initiales, (geometrie), et d'autres choses spécifiques que vous pouvez faire avec un `xterm` ou `nxterm`, ou autre.

Voici une partie utile de mon fichier `~/.Xdefaults`:

```
emacs*Background: DarkSlateGray
emacs*Foreground: Wheat
emacs*pointerColor: Orchid
emacs*cursorColor: Orchid
emacs*bitmapIcon: on
emacs*font: fixed
emacs.geometry: 80x25
```

Voyez les pages man `X`pour plus de détails à propos des ressources X.

Chris Gray (*cgray4@po-box.mcgill.ca*) dis aussi:

> Dans la debian, le fichier `~/.Xdefaults` ne semble pas être utilisé. Néanmoins, les utilisateurs de la debian peuvent mettre ce que vous avez mis plus haut dans le fichier `/etc/X11/Xresources/emacs` et ils auront toutes les belles couleurs comme si ils utilisaient une RedHat.

5 Packages Populaires

En ajout de tout les différents modes disponibles d'Emacs, il existe aussi beaucoup de package que l'on peut ajouter. Je les appelle package car ce sont plus que des nouveaux modes. Ils proposent souvent des possiblitées étendue, ou sont trop gros pour être nommé modes, car cela ne leur rendrais pas justice. Dans d'autres cas, ce sont des logiciels qui étendent ou integrent d'autres modes d'Emacs et des packages. La distinction n'est pas totalement claire, mais c'est assez.

5.1 VM (Mail)

Pour citer la VM FAQ:

> VM (View Mail) est un sous-système d'Emacs qui permet de lire des mails et de les manipuler à l'intérieur d'Emacs. Des commandes existent pour effectuer toutes les actiosn classiques d'un client Mail, comme la génération de réponse, la sauvegarde dans un répertoire, l'effacement de message, etc ... Il existe aussi d'autres commandes avancées qui effectue des tâches comme la création ou l'éclatement de sommaire/résumé, la retransmission (Forward), et l'organisation de la présentation des messages suivant différents critères.

Quand J'ai commencé à utilisez Emacs, J'ai essayé VM pendant un certain temps. J'ai trouvé que c'était le grand remplacant pour Pone, Elm, ou d'autres client Mail. Mais je n'avais pas envie d'utiliser de programme distinct pour consulter Mail et News. VM est activement développé et assez bien supporté aujourd'hui.

Il est disponible ici: *http://www.wonderworks.com/vm/* .

5.2 Gnus (Mail et News)

Pour citer le manuel GNUS:

> Gnus est un laboratoire de lecture des messages. Il vous laissera voir n'importe quoi comme étant un groupe de discussion. Vous pouvez lire des mails, naviguer dans vos répertoires, faire du ftp, vous pouvez même lire des messages des groupes de discussion!
>
> Gnus essaye de permettre aux utilisateurs de lire des news de la manière dont Emacs permet aux utilisateurs d'éditer un texte. Gnus ne fixe aucune limite à ce que les utilisateur peuvent être capable de faire. Les utilisateurs sont encouragés à étendre Gnus pour qu'il se comporte comme l'utilisateur le veut. Un programme `ne` doit `pas` contrôler les utilisateurs; les utilisateurs doivent permttre de faire se qu'ils veulent en l'utilisant (ou en abusant) le programme.

GNUS est le programme que j'utilise en ce moment pour mes mails et mes news (comme je l'ai dis plus tôt). GNUS est activement développé et trés bien supporté aujourd'hui..

Il est disponible ici: *http://www.gnus.org/* .

5.3 BBDB (A rollodex)

BBDB est une insidieuse base de donnée de type `Big Brother` un programme comme rolldex pour Emacs qui fonctionne avec la plupart des packages populaires de mails d'Emacs (Vm et GNUS inclus).

Il est disponible ici: *http://pweb.netcom.com/˜simmonmt/bbdb/index.html* .

5.4 AucTeX (another TeX mode)

AucTeX est un autre mode pour éditer des fichiers TeX.

Pour siter le site internet d'AucTeX:

> AUC TeX est un package extensible qui supporte l'écriture et le formatage des fichiers TeX pour la plupart des variantes d'Emacs. Beaucoup de package de macro sont disponible, tel que AMS TeX, LaTeX, and TeXinfo.

Il est disponible ici: *http://sunsite.auc.dk/auctex/* .

6 Autres Resources

Cette section concerne les livres, les sites Web, les groupes de discussion, les mailings listes et d'autres lieux ou vous pourrez trouver des informations à propos d'Emacs.

6.1 Livres

Il existe quelques bon livres disponibles pour apprendre Emacs. En plus de cela, vous remarquerez que dans beaucoup de livres sur Linux ou Unix vous avez un chapitre ou deux au sujet d'Emacs (et `vi`)..

6.1.1 Apprendre GNU Emacs

Auteurs: Debra Cameron, Bill Rosenblatt, Eric S. Raymond

Editeur: O'Reilly & Associates - *http://www.ora.com/*

Vous pouvez l'acheter par Amazon.com via leur programme d'association: *http://www.amazon.com/exec/obidos/ASIN/1565921526/*

Commentaire: C'est probablement le meilleur livre pour commencer. Après avoir lu le HOWTO,et et regardé la FAQ, ce livre vous servira comme tutorial compréhensible et approchable.

6.1.2 Ecrire des extensiosn GNU Emacs

Auteur: Bob Glickstein

Editeur: O'Reilly & Associates - *http://www.ora.com/*

Vous pouvez l'acheter par Amazon.com via leur programme d'association: *http://www.amazon.com/exec/obidos/ASIN/1565921526/*

Commentaire: Après avoir utilisé Emacs durant quelques temps, et avoir décidé que vous aimeriez écrire votre propre mode, ou peut-être essayer quelques customisations avancée, ce livre est fait pour vous. Il ne prétend pas vous apprenne le Lisp, mais contient quand même une courte introduction.

6.1.3 Programmer avec le Lisp d'Emacs : Une Introduction

Auteur: Robert J. Chassell

Extrait du fichier README

> C'est une introduction elementaire pour programmer en Lisp Emacs, pour les non programmeurs, et qui ne sont pas intéressé par la programmation, mais qui veulent customiser ou étendre leur environement de travail.

Vous pouvez récuperer le manual dans sa totalité via FTP anonyme à partir du serveur GNU FTP: *ftp://prep.ai.mit.edu/gnu/emacs/* .

Vous pouvez acheter une version imprimé par Amazon.com via leur programme d'association: *http://www.amazon.com/exec/obidos/ASIN/1882114418/jeremydzawodny/* .

Commentaire: C'est un bon manuel d'introduction au Lisp d'Emacs, même si vous n'êtes pas un programmeur confirmé.

6.1.4 Le manuel de référence GNU Emacs Lisp

Auteur: Richard Stallman

Editeur: The Free Software Foundation - *http://www.fsf.org/*

Vous pouvez récuperer le manual dans sa totalité via FTP anonyme à partir du serveur GNU FTP: *ftp://prep.ai.mit.edu/gnu/emacs/* .

Commentaire:C'est le guide ultime pour le langage de programmation Lisp d'Emacs

6.2 Sites Internet

6.2.1 EMACSulation

EMACSulation est une colonne écrite par Eric Marsden qui apparait dans le magazine en-ligne Linux Gazette situé à *http://www.linuxgazette.com/* . La colonne écrite la plus récente à ce jour est situé à *http://www.linuxgazette.com/issue39/marsden.html* . Regarder en bas de l'article pour obtenir les colonnes précédentes.

6.3 Groupes de discussions

Rechercher dans votre fournisseur de news local les groupes de discussion qui contiennent la chaine `emacs` et vous en trouverez probablement quelques uns ... Voici ceux que mon serveur supporte:

- comp.emacs
- comp.emacs.sources
- gnu.emacs
- gnu.emacs.bug
- gnu.emacs.help
- gnu.emacs.sources

6.4 Listes de diffusions

La seule liste de diffusion dédiée à Emacs que je connaisse à l'heure actuelle et la liste NT-Emacs. C'est une liste pour les utilisateurs d'Emacs sur Micro$oft Windows. Voyez la FAQ NT-Emacs *http://www.cs.washington.edu/homes/voelker/ntemacs.html* pour plus d'informations.

6.5 Les archives Lisp Emacs

Extrait du README des archives Lisp Emacs:

> Les archives Lisp Emacs sur ftp.cis.ohio-state.edu contiennent différentes portions et packages du code de Lisp Emacs. Lisp Emacs est le langage utilisé pour l'éditeur GNU Emacs publié par la FSF. Bien que le code de Lisp Emacs soit inclut dans la distribution d'Emacs, bon nombre de personne ont écrit des packages pour l'interfacer avec d'autres systèmes, pour avoir un meilleur support de l'édition de leur langage de programmation, pour ajouter des options, ou pour changer le comportement par défault d'Emacs. La plupart du contenu de cette archive a été écrit par des

personnes(individuals) et distribué sur internet par les listes de diffusions info-emacs, info-gnu-emacs, ou des groupes de discussion tel que comp.emacs, gnu.emacs ou gnu.emacs.sources.

Les archives sont disponibles via Ftp anonyme à l'adresse *ftp://ftp.cis.ohio-state.edu/pub/emacs-lisp/* .

NOTE: Pour autant que je puisse dire, les archives Lisp d'Emacs commencent à être périmées. J'ai vu seulement quelques nouveaux package ou des mises à jour apparaitre, quoique je sais qu'il en existe. Ils `sont` postés sur le groupe de discussion `comp.emacs.sources`. (Corrigez moi si je me trompe.)

7 Crédits

Les personnes suivantes ont contribué au succés de ce document.

- Robert Vollmert *rvollmer@gmx.net*
- Larry Brasfield *larrybr@seanet.com*
- Etienne Grossmann *etienne@anonimo.isr.ist.utl.pt*
- Thomas Weinell *kf6mli@amsat.org*
- Adam C. Finnefrock *adam@bigbro.biophys.cornell.edu*
- Chris Gray *cgray4@po-box.mcgill.ca*
- Robert J. Chassell *bob@rattlesnake.com*
- Isaac To *kkto@csis.hku.hk*
- Matteo Valsasna *valsasna@elet.polimi.it*
- Tijs van Bakel *smoke@casema.net*