Compiler le code de Minefield sous Linux.

Ou comment avoir un aperçu du futur Firefox 3 sous Linux... :)

Table des matières

Avant propos :

Voici donc un petit tutoriel rapide pour compiler le code de développement de Firefox (et Thunderbird et SeaMonkey, les différences étant expliquées en fin de document) sous Linux. Tutoriel aussi bien valable pour un linux x86 que x86_64, voire même PowerPC ;)

Ce qui permettra aussi aux personnes ayant des machines autres que le bon vieux processeur 32 bits ou qui utilisent une distribution 64 bits de profiter pleinement des versions de développement, même si celles-ci ne devraient pas être utiliser pour effectuer des rapports de bogues sur Bugzilla ;)

Je prends comme base une Debian Etch pour i386, instructions qui s'appliqueront sans trop de problème sur une Ubuntu 6.06.1 LTS, 6.10, 7.04, etc...

Pour des distributions basées sur des RPMs, comme Fedora ou OpenSuSE, il vous suffira de trouver les paquets équivalents à ceux listé plus bas.

NB : Les fichiers .mozconfig utilisés se basent sur les options *officielles* de compilation.

I. Configurer l'environnement de compilation.

Pour compiler le code de développement de Firefox, voici les pré-requis, si l'on en croit les pages http://developer.mozilla.org/fr/docs/Documentation_sur_la_compilation et http://developer.mozilla.org/fr/docs/Pr%C3%A9alables_%C3%A0_la_compilation_sous_Linux :

Sur le plan matériel :

- 512 Mo de ram conseillé pour être tranquille.
- 2 Go de place libre sur le disque dur.

Sur le plan logiciel :

- Une distribution linux récente, avec au minimum un noyau 2.2.14 (avec glibc 2.3.2, XFree86-3.3.6, gtk+2.0, fontconfig/xft et libstdc++5). Autant dire, toutes les distributions depuis début 2005.
- gcc 3.2 ou plus récent
- perl 5.6 ou suivant
- gnu make 3.79.1 ou suivant
- cvs 1.11 ou suivant
- GTK2 récent.
- libIDL 0.6.3 ou plus récent
- freetype 2.1.0 ou suivant
- zip 2.3 ou suivant
- fontconfig
- pkgconfig 0.9.0

Donc sur une debian 4.0 (alias Etch) fraîchement installée et mise à jour, il faut rentrer sur la ligne de commande – ou ajouter dans Synaptic :

En mode root :

```
# aptitude install build-essential libgtk2.0-dev libidl-dev xorg-dev cvs libgnomevfs2-dev
```

Et en version *ubuntu* :

```
$ sudo aptitude install build-essential libgtk2.0-dev libidl-dev xorg-dev cvs libgnomevfs2-dev
```

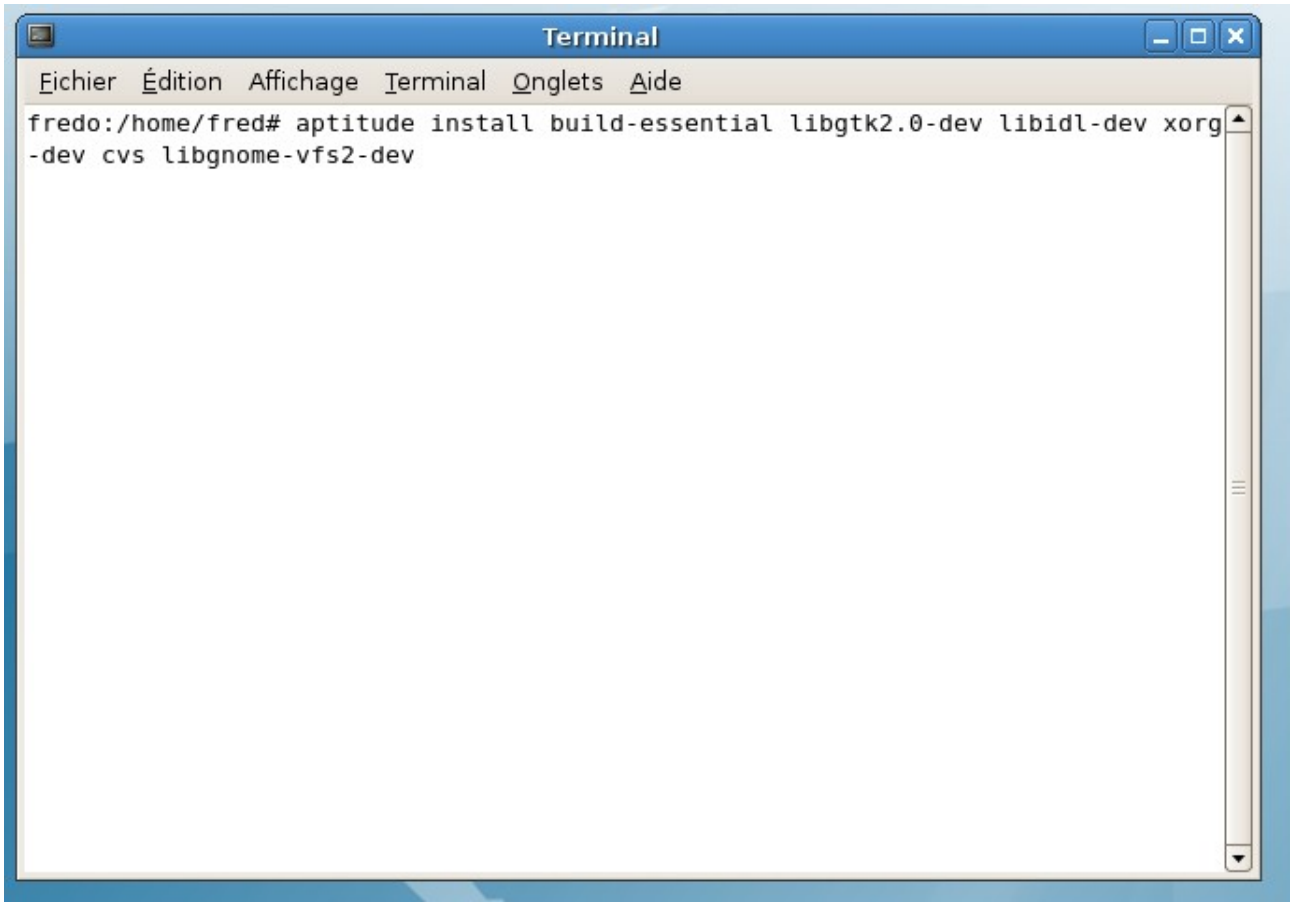

Illustration 1: Les dépendances sous Debian Etch en console.

On va maintenant modifier le .bashrc – en supposant que vous utilisez bash - en y rajoutant les constantes suivantes. Voici ce qu'il faut rentrer, dans un terminal.

```
$ cd
$ nano .bashrc
```

Et on ajoute le texte suivant en fin du fichier :

```
export MOZILLA_OFFICIAL=1
export BUILD_OFFICIAL=1
export CVSROOT=:pserver:anonymous@cvs-mirror.mozilla.org:/cvsroot
```

Bien entendu, si vous préférez gedit ou un autre éditeur, libre à vous ;)

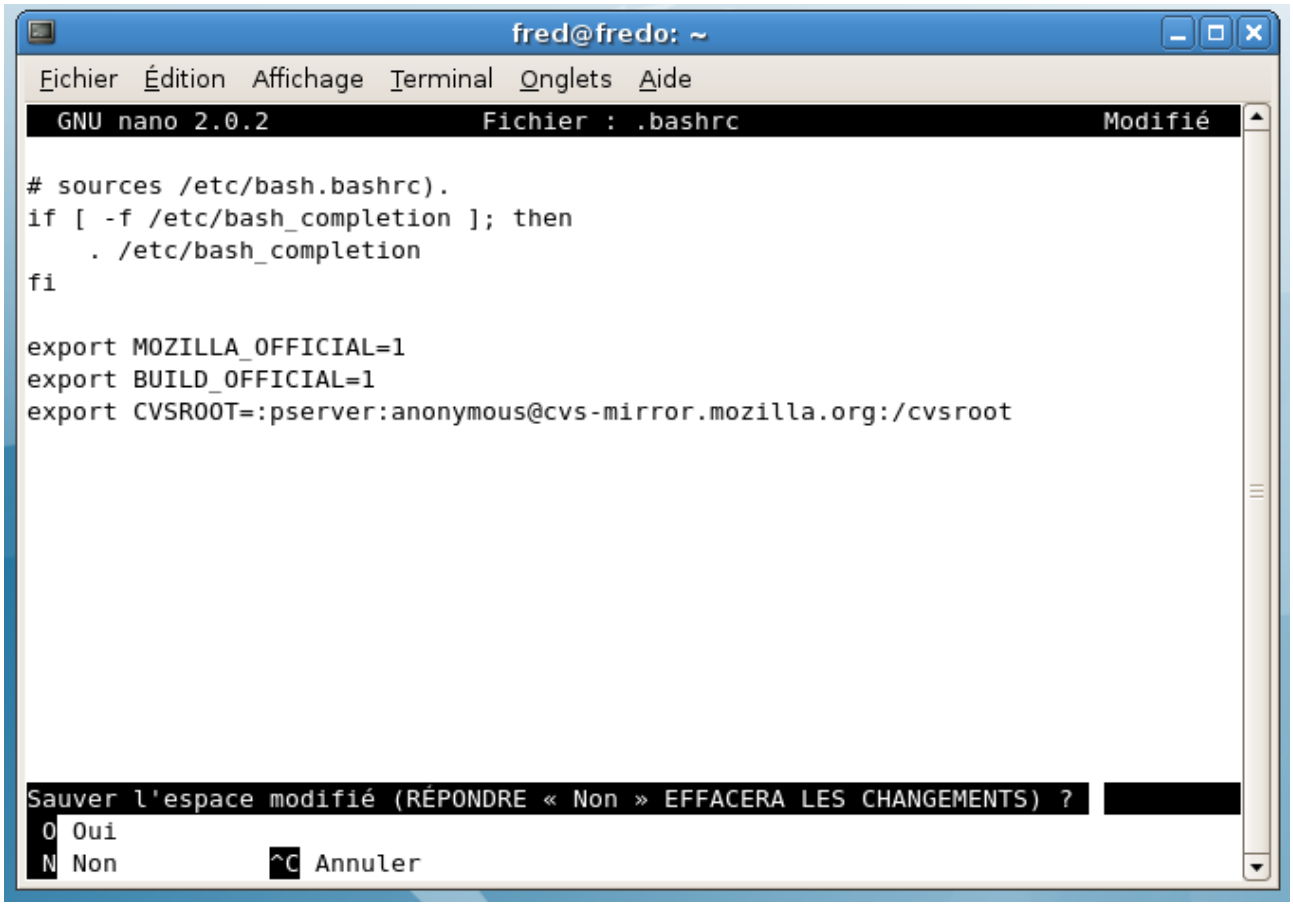

Illustration 2: Le .bashrc modifié.

Faisons ctrl +x pour sauver et quitter nano. Relançons maintenant le terminal. Entrons le mot de passe pour le serveur cvs de mozilla.org

```
$ cvs login
```

Il suffit de presser sur entrée, et de répéter l'opération. Maintenant, on crée un répertoire fox puis on y entre :

```
$ mkdir fox
$ cd fox
```

On va commencer à récupérer le code source :

```
$ cvs co mozilla/client.mk mozilla/browser/config
$ cd mozilla
```

Maintenant, on crée le fichier .mozconfig qui va reprendre les options de compilations.

```
$ gedit .mozconfig &
```

On peut remplacer gedit par son éditeur préféré, bien entendu :)

Et on entre le code suivant dans le .mozconfig :

```
#
# See http://www.mozilla.org/build/ for build instructions.
#

. $topsrcdir/browser/config/mozconfig

# Options for 'configure' (same as command-line options).
ac_add_options --enable-optimize="-Os -freorder-blocks -fno-reorder-functions"
ac_add_options --disable-debug
ac_add_options --disable-tests

ac_add_options --enable-canvas
ac_add_options --enable-svg
ac_add_options --enable-pango
ac_add_options --enable-default-toolkit=cairo-gtk2
ac_add_options --enable-libxul
ac_add_options --enable-places-bookmarks
ac_add_options –enable-strip
ac_add_options --disable-airbag
```

Note : l'option –disable-airbag est indispensable si vous ne faites pas une version avec le code de débogage, comme c'est le cas pour ce document.

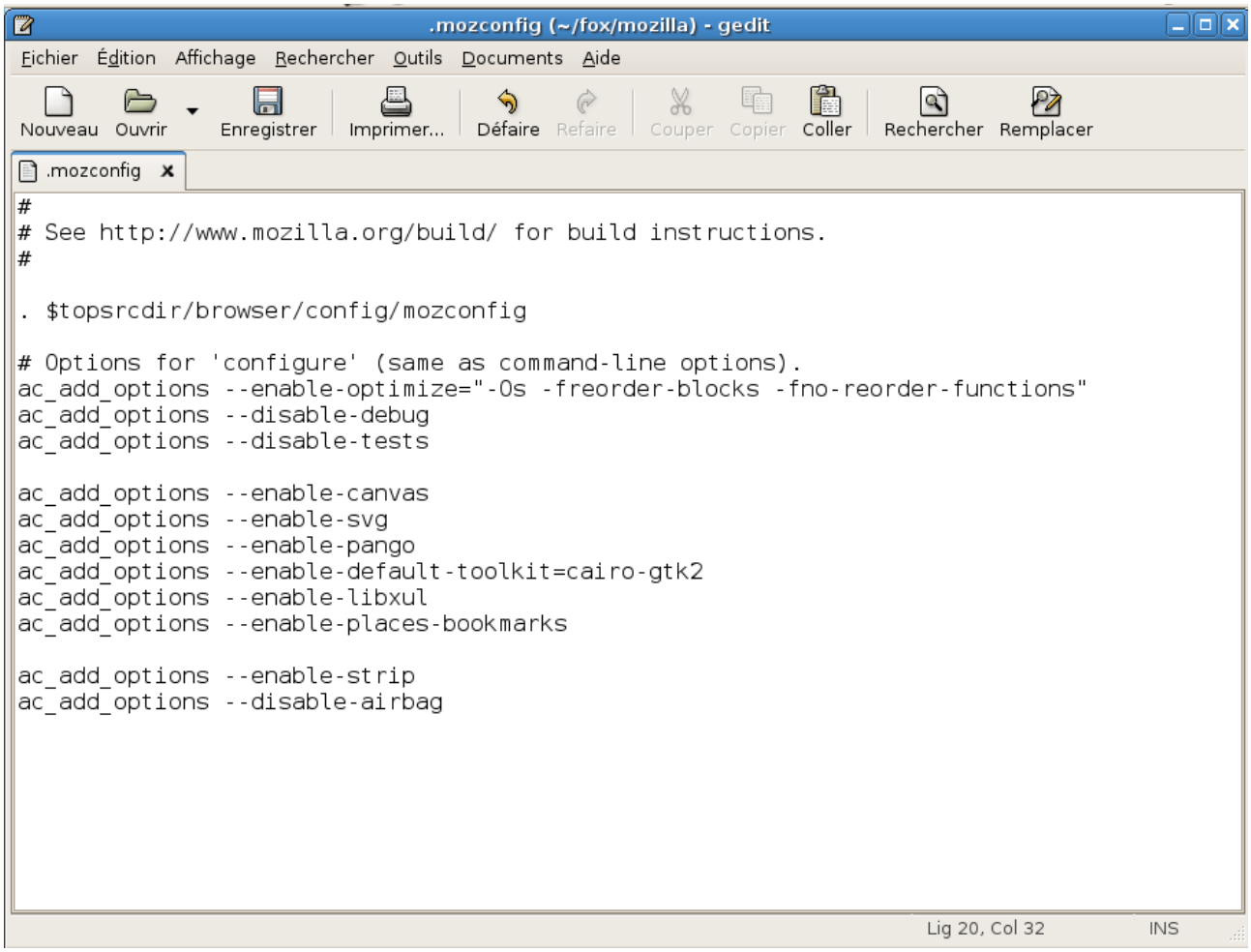

Illustration 3: Le .mozconfig que nous allons utiliser pour compiler Minefield.

On lance la récupération du code :

```
$ make -f client.mk checkout
```

Puis, une fois le code récupéré, on l'archive, histoire de pouvoir le réutiliser par la suite ;)

```
$ cd ..
$ tar cvfj moz-fox.tar.bz2 mozilla/
```

On retourne dans le répertoire du code source.

II. La compilation proprement dite.

Maintenant, la compilation va se passer en trois temps.

a) Etape 1 : les dépendances

```
$ make -f client.mk checkout
```

b) Etape 2 : la compilation

```
$ make -f client.mk build
```

Il ne faut pas être pressé, il faut compter *environ* **1 heure** sur un AMD Sempron 3100+ avec 1 Go de ram et une Ubuntu linux 7.10 x86_64...

c) Etape 3 : archivage du logiciel

```
$ make -C browser/installer
```

L'archive du logiciel se trouve dans le répertoire mozilla/dist/bin sous la forme d'une archive tar.bz2.

Illustration 4: Minefield 3.0 pré-alpha 7 fraichement recompilé :)

III. Pour Thunderbird et SeaMonkey ?

a) Pour Thunderbird :

Pour le code source, il faut remplacer la première commande par un :

```
$ cvs co mozilla/client.mk mozilla/mail/config
$ cd mozilla
```

Bien entendu, il faudra prendre un répertoire du doux nom de mail ;)

Et pour le .mozconfig :

```
#
# See http://www.mozilla.org/build/ for build instructions.
#

. $topsrcdir/mail/config/mozconfig

# Options for 'configure' (same as command-line options).
ac_add_options --enable-optimize="-Os"
ac_add_options --disable-debug
ac_add_options --disable-tests
ac_add_options --disable-tests
ac_add_options --disable-shared
ac_add_options --enable-static
ac_add_options --enable-pango
ac_add_options --enable-default-toolkit=cairo-gtk2
ac_add_options --disable-airbag
```

Enfin, la commande d'empaquetage :

```
$ make -C mail/installer
```

b) Pour SeaMonkey :

Pour le code source, il faut remplacer la première commande par un :

```
$ cvs co mozilla/client.mk
$ cd mozilla
```

Bien entendu, il faudra prendre un répertoire du doux nom de suite ;)

Et pour le .mozconfig :

```
#
# See http://www.mozilla.org/build/ for build instructions.
#

mk_add_options MOZ_CO_PROJECT=suite

# Options for 'configure' (same as command-line options).
ac_add_options --enable-optimize="-Os"
ac_add_options --disable-debug
ac_add_options --disable-tests
ac_add_options --disable-shared
ac_add_options --enable-static
ac_add_options –enable-pango
ac_add_options --enable-canvas
ac_add_options --enable-default-toolkit=cairo-gtk2
ac_add_options --disable-airbag
```

Enfin, la commande d'empaquetage :

```
$ make -C suite/installer
```

Bonne compilation de votre logiciel préféré... ;)

Document créé avec VMWare Server, avec une Debian Linux 4.0r0 pour i386 et OpenOffice.org 2.2.0 sous une Ubuntu linux 7.04 x86_64 par Frédéric Béziès – fredbezies@gmail.com