Installer ou construire GTKAda Quartz sur MAC OS X avec la bibliothèque XNAdaLib

GTKAda est la boite à outil graphique en Ada 95 basée sur GTK+ pour construire des applications portables sur la plupart des plateformes.

GTK+ est une bibliothèque graphique Linux conçue à l'origine pour fonctionner sur les systèmes Unix avec X-Windows... mais les développeurs GTK-OSX ont inclus une interface de Mac OS X utilisant le moteur graphique quartz. Un programme peut donc être conçu pour tourner sur OSX sans activer le sous-système X11.

Site http://www.gtk.org/download/macos.php.

La livraison GPL 2016 ne propose pas GTKAda prêt à l'emploi pour MacOS. Nous pouvons soit le construire à partir des sources (voir §2 et suivants) soit le prendre prêt à l'emploi sur Source Forge (voir §1).

GTKAda est inclue dans la bibliothèque XNAdaLib qui comprend également Glade, Gate3, GNATColl, Florist, Simple Components, AICWL, AdaCurses, Zanyblue, PragmARC, Gnoga, AdaControl, AdaDep, AdaSubst.

Sommaire

1.	Installer GTKAda Quartz avec la bibliothèque XNAdaLib	2
2.	Construire GTK+ Quartz avec la bibliothèque XNAdaLib	3
3.	Construire GTKAda (sans OpenGL)	6
4.	Construire GLADE	8
5.	Construire Florist	8
6.	Construire Gate3	9
7.	Construire AdaCurses	10
8.	Construire Simple Components et AICWL	11
9.	Construire GNATColl	15
10.	Construire Zanyblue	16
11.	Construire PragmARC	17
12.	Construire Gnoga	18
13.	Construire les utilitaires Adalog	19

1. Installer GTKAda Quartz avec la bibliothèque XNAdaLib

Télécharger le fichier suivant sur le bureau du Mac :
xnadalib-gpl-2016-quartz-x86_64-apple-darwin14.5.0-bin.tgz,
depuis le site de Source Forge "http://sourceforge.net/projects/gnuada/files/GNAT_GPL%20Mac%20OS%20X/2016-el-capitan".

Lancer le Terminal dans un compte administrateur et taper les commandes suivantes :

```
$ instbase=/usr/local # ou tout autre répertoire
$ mkdir -p $instbase
$ cd $instbase
$ tar xzf ~/Desktop/xnadalib-gpl-2016-quartz-x86_64-apple-darwin14.5.0-bin.tgz
```

GTK+, GTKAda, Glade, Gate3, GNATColl, Florist, Simple Components, AICWL, AdaCurses, Zanyblue, PragmARC, Gnoga, AdaControl, AdaDep, AdaSubst s'installent à partir du répertoire : $instbase/xnadalib-2016.

Pour une utilisation courante, saisir aussi les commandes suivantes :

```
$ instxada=$instbase/xnadalib-2016
$ echo 'PATH=$instxada/bin:$PATH' >> ~/.profile
$ echo 'PATH=$instxada/bin:$PATH' >> ~/.bashrc
$ echo 'export MANPATH=$instxada/man:$MANPATH' >> ~/.profile
$ echo 'export MANPATH=$instxada/man:$MANPATH' >> ~/.bashrc
$ echo 'export MANPATH=$instxada/share/man:$MANPATH' >> ~/.profile
$ echo 'export MANPATH=$instxada/share/man:$MANPATH' >> ~/.bashrc
$ echo 'export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:
$GPR_PROJECT_PATH' >> ~/.profile
$ echo 'export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:
$GPR_PROJECT_PATH' >> ~/.bashrc
$ echo 'export XDG_DATA_DIRS=$instxada/share' >> ~/.profile
$ echo 'export XDG_DATA_DIRS=$instxada/share' >> ~/.bashrc
```

Pour une utilisation temporaire, utiliser à chaque fois les commandes suivantes :

```
$ instxada=$instbase/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export MANPATH=$instxada/man:$MANPATH
$ export MANPATH=$instxada/share/man:$MANPATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:$GPR_PROJECT_PATH
$ export XDG_DATA_DIRS=$instxada/share
```

Des exemples de programme Ada sont disponibles dans le répertoire $instxada/share/examples/gtkada.
Une documentation au format HTML est disponible dans les répertoires
$instxada/share/doc/gtkada et $instxada/share/gtk-doc/html :

```
$ open $instxada/share/doc/gtkada/gtkada_rm/index.html
$ open $instxada/share/doc/gtkada/gtkada_ug/index.html
$ open $instxada/share/gtk-doc/html/gtk3/index.html
```

Voir l'utilisation de GTKAda avec des exemples sur Blady :
http://blady.pagesperso-orange.fr/a_savoir.html#gtkada

ainsi que Gnoga :
http://blady.pagesperso-orange.fr/a_savoir.html#gnoga

2. Construire GTK+ Quartz avec la bibliothèque XNAdaLib

Gtk+ est une bibliothèque graphique en C pour X-Window et Win32. Elle fut développée initialement pour Gimp. Nous allons construire la version comportant le rendu natif MacOS directement avec Quartz.

Site web : http://www.gtk.org.
La version installée est 3.20.3.
Récupérer sur le bureau le script d'installation gtk-osx-build-setup.sh à l'adresse :
https://git.gnome.org/browse/gtk-osx/plain/gtk-osx-build-setup.sh
Source https://wiki.gnome.org/action/show//Projects/GTK+/OSX/Building?action=show&redirect=GTK%2B%2FOSX%2FBuilding.

Et aussi xnadalib-2016-diff.tgz sur http://blady.pagesperso-orange.fr/telechargements/gtkada/xnadalib-2016-diff.tgz.

Saisir la commande suivante dans le Terminal tout en étant connecté à Internet :

```
$ sh ~/Desktop/gtk-osx-build-setup.sh ...
PATH does not contain /Users/blady/.local/bin, it is recommended that you add that.
```

Cette commande installe jhbuild dans ~/Source et créé ~/.local/bin/jhbuild. Il installe aussi ~/.jhbuildrc et ~/.jhbuildrc-custom et copie les modules gtk-osx courants dans ~/Source/jhbuild/modulesets. (Si ces fichiers sont déjà présents, je recommande de les supprimer avant de lancer la commande ci-dessus.)

Comme suggérer nous allons ajouter l'accès demandé.

```
$ echo 'PATH=$HOME/.local/bin:$PATH' >> ~/.profile
$ echo 'PATH=$HOME/.local/bin:$PATH' >> ~/.bashrc
```

Pour une utilisation temporaire, utiliser à chaque fois la commande :

```
$ PATH=$HOME/.local/bin:$PATH
```

Pour toute la construction de GTK+ nous utiliserons le compilateur natif du Mac, adapter la variable PATH en conséquence :

```
$ echo $PATH
/Users/blady/.local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin:/usr/local/git/bin:/usr/X11/bin
$ which gcc
/usr/bin/gcc
```

Saisir les commandes suivantes tout en étant connecté à Internet :

```
# Appliquer les correctifs apportés sur Blady :
$ cd ~/Desktop
$ tar xzf ~/Desktop/xnadalib-2016-diff.tgz
$ cd
$ patch -p0 < ~/Desktop/xnadalib-2016-diff/jhbuildrc-custom.diff
$ cd /usr/local
$ sudo mkdir xnadalib-2016
$ sudo chown $LOGNAME xnadalib-2016
$ sudo mkdir src-gtk-osx
$ sudo chown $LOGNAME src-gtk-osx
$ cd src-gtk-osx
$ export CMAKE_OSX_DEPLOYMENT_TARGET="10.11"
$ export CMAKE_OSX_SYSROOT=/Applications/Xcode.app/Contents/Developer/Platforms/
MacOSX.platform/Developer/SDKs/MacOSX10.11.sdk
$ jhbuild bootstrap
...
# Python fourni avec MacOS 10.9 n'est pas installé dans le SDK
$ jhbuild build python
...
$ jhbuild build meta-gtk-osx-bootstrap
...
$ jhbuild build meta-gtk-osx-gtk3
...
```

Liste des modules installés :

- adwaita-icon-theme-3.16.0
- atk-2.20.0
- autoconf-2.69
- automake-1.10.3
- automake-1.11.6
- automake-1.12.6
- automake-1.13.4
- automake-1.14.1
- automake-1.15
- bash-4.3.30
- bison-3.0.2
- cairo-1.14.6
- cmake-3.2.1
- expat-2.1.0
- flex-2.5.37
- fontconfig-2.11.1
- freetype-2.5.3
- gdk-pixbuf-2.34.0
- glib-2.48.0
- gnome-common-3.14.0
- gnome-themes-standard-3.16.0
- gobject-introspection-1.48.0
- gtk-doc-1.21
- gtk-mac-integration-2.0.8
- gtk+-3.20.3
- harfbuzz-0.9.40
- hicolor-icon-theme-0.13
- icon-naming-utils-0.8.90
- icu4c-55_1
- intltool-0.51.0
- itstool-2.0.2
- jpeg-9a
- libcroco-0.6.8
- libepoxy-v1.3.1
- libffi-3.2.1
- libpng-1.6.17
- librsvg-2.40.5
- libtool-2.4.6
- libxml2-2.9.2
- libxslt-1.1.28
- pango-1.38.1
- pixman-0.34.0
- pkg-config-0.28
- Python-2.7.11
- ragel-6.9
- readline-6.3
- tiff-4.0.3

- util-macros-1.19.0
- XML-Parser-2.41
- XML-Simple-2.22
- xz-5.2.1
- yelp-tools-3.16.1
- yelp-xsl-3.16.1
- zlib-1.2.8

Voilà c'est un peu long mais c'est fait. Nous pouvons alors découvrir les exemples :

```
$ jhbuild shell
$ cd /usr/local/src-gtk-osx/gtk+-3.16.0/examples
$ ./drawing
...
```

Une documentation sous forme d'un manuel de référence est disponible dans le répertoire / usr/local/xnadalib-2016/share/gtk-doc/html, par exemple :

```
$ open /usr/local/xnadalib-2016/share/gtk-doc/html/gtk3/index.html
```

Et surtout, les programme de test et démo qui donnent plein de situations d'emploi des objets GTK avec leur description, le code source C correspondant et une démonstration du résultat :

```
$ jhbuild shell # si pas déjà fait
$ gtk3-demo
$ cd /usr/local/src-gtk-osx/gtk+-3.16.0/tests
$ ./testgtk
...
```

3. Construire GTKAda (sans OpenGL)

GTKAda est la boite à outil graphique en Ada 95 basée sur GTK+ pour construire des applications portables sur la plupart des plateformes.

Site web : http://libre.adacore.com/libre/tools/GtkAda.

Télécharger sur le bureau l'archive "gtkada-gpl-2016-src.tar.gz" à partir du site Libre d'AdaCore "https://libre.adacore.com" à la page "Download GNAT GPL" puis cocher "Free Software or Academic Development" et cliquer sur "Build your download package", sélectionner plateforme "x86-64 GNU Linux (64 bits)" "GNAT GPL 2016" (même si GtkAda n'a pas été intégré pour MacOS / Darwin, les sources pour Linux sont utilisables) puis "GtkAda GPL 2016" et "Sources" :

- gtkada-gpl-2016-src.tar.gz 11.9 MB May 16, 2016

Saisir les commandes suivantes dans le Terminal :

```
# Activation du compilateur Ada GNAT
$ PATH=/usr/local/gnat/bin:$PATH
# Activation de la bibliothèque GTK-OSX
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ cd /usr/local/src-gtk-osx
```

```
$ tar xzf ~/Desktop/gtkada-gpl-2016-src.tar.gz
$ cd gtkada-gpl-2016-src
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xnadalib-2016-diff/gtkada-gpl-2016.diff
# Ajout des flags de compilation pour la prise en compte de INTL :
$ CPPFLAGS=-I$instxada/include LDFLAGS=-L$instxada/lib ./configure --prefix=$instxada --
enable-build=Debug --disable-shared
$ make
$ make install
$ cp -p po/build_skeleton.pl $instxada/bin
```

La bibliothèque GTKAda s'est installée dans le répertoire $instxada.
Pour une utilisation courante, nous positionnons les variables d'environnement PATH pour une utilisation en ligne de commande et GPR_PROJECT_PATH pour une utilisation avec un projet GPS :

```
$ echo 'PATH=$instxada/bin:$PATH' >> ~/.profile
$ echo 'PATH=$instxada/bin:$PATH' >> ~/.bashrc
$ echo 'export MANPATH=$instxada/man:$MANPATH' >> ~/.profile
$ echo 'export MANPATH=$instxada/man:$MANPATH' >> ~/.bashrc
$ echo 'export MANPATH=$instxada/share/man:$MANPATH' >> ~/.profile
$ echo 'export MANPATH=$instxada/share/man:$MANPATH' >> ~/.bashrc
$ echo 'export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:
$GPR_PROJECT_PATH' >> ~/.profile
$ echo 'export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:
$GPR_PROJECT_PATH' >> ~/.bashrc
```

Pour une utilisation temporaire, utiliser à chaque fois les commandes suivantes :

```
$ PATH=$instxada/bin:$PATH
$ export MANPATH=$instxada/man:$MANPATH
$ export MANPATH=$instxada/share/man:$MANPATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:$GPR_PROJECT_PATH
```

Une démo tout en Ada est présente dans le répertoires "$instxada/share/examples/gtkada" :

```
$ export instxada
$ ~/.local/bin/jhbuild shell # si pas déjà fait
$ cd $instxada/share/examples/gtkada/testgtk
$ ./testgtk
```

Une documentation sous forme d'un manuel utilisateur et d'un manuel de référence est disponible dans le répertoire "$instxada/share/doc/gtkada".

```
$ open $instxada/share/doc/gtkada/gtkada_ug/index.html
$ open $instxada/share/doc/gtkada/gtkada_rm/index.html
```

Voir l'utilisation de GTKAda avec des exemples sur Blady :
http://blady.pagesperso-orange.fr/a_savoir.html#gtkada

4. Construire GLADE

Glade est un outil graphique de développement d'interfaces utilisateurs pour la bibliothèque GTK. Les fichiers XML produits par Glade peuvent être utilisés par de nombreux langage de programmation comme C, C++, C#, Vala, Java, Perl, Python et ... Ada ;-).
Site web : https://glade.gnome.org.
La version installée est 3.18.3.

Saisir les commandes suivantes dans le Terminal :

```
$ PATH=$HOME/.local/bin:$PATH
$ instxada=/usr/local/xnadalib-2016
# Le fichier $instxada/etc/xml/catalog est vide (?)
# prenons celui de $instxada/share/xml/
$ cp -p $instxada/share/xml/catalog $instxada/etc/xml
$ jhbuild build glade
...
# Le fichier de traduction n'est pas installé là où Glade l'attend
$ cp -p $instxada/lib/locale/fr/LC_MESSAGES/glade.mo $instxada/share/locale/fr/LC_MESSAGES
```

Ne pas oublier de configurer LANG (pour l'affichage français) et XDG_DATA_DIRS avant de lancer Glade :

```
$ export XDG_DATA_DIRS=$instxada/share
$ export LANG=fr_FR.UTF-8
$ $instxada/bin/glade
```

Voir sur Blady pour les premiers pas avec un exemple :
http://blady.pagesperso-orange.fr/a_savoir.html#gtkada

5. Construire Florist

Florist contient les composants conforment aux standards Posix Ada : IEEE Standards 1003.5: 1992, IEEE STD 1003.5b: 1996 et en partie IEEE STD 1003.5c: 1998.
Récupérer l'archive suivante depuis le site Libre d'AdaCore "http://libre.adacore.com" à la page "Download GNAT GPL" puis cocher "Free Software or Academic Development" et cliquer sur "Build your download package", sélectionner plateforme "x86-64 GNU Linux (64 bits)" "GNAT GPL 2016" puis "Florist GPL 2016" :

- florist-gpl-2016-src.tar.gz 292 KB May 16, 2016

Avant de démarrer la compilation, GNAT et le répertoire d'installation doivent être activés, ajouter leur emplacement comme par exemple (si pas déjà fait) :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ tar xzf ~/Desktop/florist-gpl-2016-src.tar.gz $ cd florist-gpl-2016-src/
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xnadalib-2016-diff/florist-gpl-2016-src.diff
$ ./configure CC='gcc -g -O0' --prefix=$instxada
$ Build=Debug make
$ make install
```

Une documentation au format HTML est disponible dans le répertoire $instxada/share/doc/florist/florist_rm.

6. Construire Gate3

Gate3 est un utilitaire qui produit du code Ada à partir d'un fichier Glade.

Gate3 a été développé par Francois Fabien sous licence MIT (http://sourceforge.net/projects/lorenz).

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple (si pas déjà fait) :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

a) Construire Template-Parser

Il s'agit d'un composant qui remplace des zones de textes balisées dans des modèles.

Nous allons d'abord installer Template-Parser (https://forge.open-do.org/projects/template-parser) :

```
$ cd /usr/local/src-gtk-osx
$ git clone http://forge.open-do.org/anonscm/git/template-parser/template-parser.git
$ cd template-parser
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xadalib-2016-diff/template-parser.diff
$ make DEBUG=true prefix=$instxada
# Utilisation de latexpdf
$ PATH=/usr/local/texlive/2014/bin/x86_64-darwin:$PATH
$ make build-doc
$ make install
```

b) Construire Gate3

Récupérer ensuite sur le bureau le fichier gate3_05b.zip à partir du site Blady.

```
$ cd /usr/local/src-gtk-osx
$ unzip ~/Desktop/xadalib-2016-diff/gate3_05b.zip
$ cd gate3_05
$ make
$ make PREFIX=$instxada install
```

Des exemples sont construits avec :

```
$ make editor
$ make calculator
$ make lady
$ make lorenz
```

Leur exécution :

$ ./editor $ ./calculator $ ./lady $ ./lorenz

Les sources Ada sont construits par Gate3 à partir d'un fichier Glade avec gate3.sh. Voir le tutoriel Factoriel sur Blady :

http://blady.pagesperso-orange.fr/a_savoir.html#gtkada

Les fichiers modèles peuvent être modifiés en incluant notamment la licence dans les modèles des paquetages spécification et corp :

- gate3_license.txtlicence MIT par défaut, à changer à votre convenance
- gate3_header.tmplten-tête de la procédure principale, inclut gate3_license.txt
- gate3_main.tmplt modèle de la procédure principale
- gate3_spec.tmplt modèle du paquetage spécification des callbacks
- gate3_body.tmplt modèle du paquetage corps des callbacks

7. Construire AdaCurses

AdaCurses est une bibliothèque Ada 95 basée sur NCurses. La correspondance avec les fonctions de NCurses n'est pas directe mais a été construite dans l'esprit Ada de privilégier la lisibilité.

(http://invisible-island.net/ncurses/ncurses-Ada95.html)

Avant de démarrer la compilation, GNAT et le répertoire d'installation doivent être activés, ajouter leur emplacement comme par exemple :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ PATH=/usr/local/gnat/bin:$PATH
```

Récupérer ensuite sur le bureau le fichier AdaCurses.tar.gz à partir du site :
http://invisible-island.net/datafiles/release/AdaCurses.tar.gz

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ tar xzf ~/Desktop/AdaCurses.tar.gz $ cd AdaCurses-20110404/
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xadalib-2016-diff/AdaCurses-20110404.diff
$ ./configure CC='gcc -g -O0' --prefix=$instxada
$ make
$ make rm-docs
$ make gpr-install
$ mv $instxada/share/gpr/library.gpr $instxada/share/gpr/adacurses.gpr
```

Des exemples de programme sont disponibles dans le répertoire $instxada/share/AdaCurses.
Une documentation au format HTML est disponible dans le répertoire $instxada/share/doc/AdaCurses.

8. Construire Simple Components et AICWL

Simple Components et AICWL proposés par Dmitry Kazakov contiennent les composants :

- Simple components : graphes, ensembles, piles, vecteurs, analyseurs d'expressions, primitives de synchronisation, nombre pseudo-aléatoires...
- Strings edit : mise à l'échelle des axes, nombre romains, entiers et réels, codage UTF-8 et Unicode, recherche avec jokers...
- Tables : container de données avec recherche par chaines de caractères,
- AICWL (Ada industrial control widget library) : collection de widgets de visualisation type compteur de vitesse et vue mètre, d'horloges, d'oscillogramme, éditeur de widget...
- GtkAda contributions : multi-tâche, vue arborescente, navigation de fichiers, image en code source Ada, fichier de ressources graphiques, modèle de couleur HSL, des boutons, exécution de processus asynchrones...

Site web : http://www.dmitry-kazakov.de.

a) Construire GTKSourceView

Il s'agit d'un widget qui étend GtkTextView avec les fonctionnalités d'un éditeur de code source comme par exemple la coloration syntaxique.
Site web : https://wiki.gnome.org/Projects/GtkSourceView.
La version installée est 3.14.3.

Saisir les commandes suivantes dans le Terminal :

```
$ instxada=/usr/local/xnadalib-2016
$ ~/.local/bin/jhbuild build gtksourceview3
# Le fichier de traduction n'est pas installé là où gtksourceview l'attend
$ cp -p $instxada/lib/locale/fr/LC_MESSAGES/gtksourceview-3.0.mo $instxada/share/locale/fr/LC_MESSAGES/0
```

b) Construire GNUTLS

Il s'agit d'une bibliothèque qui contient des API pour les protocoles SSL, TLS et DTLS.
Site web : http://www.gnutls.org.
La version installée est 3.3.12.

Saisir la commande suivante dans le Terminal :

```
$ ~/.local/bin/jhbuild build gnutls
```

Si une erreur survient, il faut alors recommencer la construction puis ignorer les erreurs 2 fois (réponses en gras ci-dessous) :

...

/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin/ranlib: file: ../../libcrypto.a(v3_addr.o) has no symbols
*** Error during phase build of openssl: ########## Error running make -j 5 *** [2/6]

[1] Rerun phase build
[2] Ignore error and continue to install
[3] Give up on module
[4] Start shell
[5] Reload configuration
[6] Go to phase "wipe directory and start over"
[7] Go to phase "configure"
[8] Go to phase "clean"
[9] Go to phase "distclean"
choice: **1<ret>**
...
making all in crypto/pqueue...
make[2]: Nothing to be done for `all'.
*** Error during phase build of openssl: ########## Error running make -j 5 *** [2/6]

[1] Rerun phase build
[2] Ignore error and continue to install
[3] Give up on module
[4] Start shell
[5] Reload configuration
[6] Go to phase "wipe directory and start over"
[7] Go to phase "configure"
[8] Go to phase "clean"
[9] Go to phase "distclean"
choice: **2<ret>**

making install in engines...
installing 4758cca
cp: lib4758cca.so: No such file or directory
make[1]: *** [install] Error 1
make: *** [install_sw] Error 1
*** Error during phase install of openssl: ########## Error running make DESTDIR=/usr/local/xnadalib-2016/_jhbuild/root-openssl install_sw *** [2/6]

[1] Rerun phase install
[2] Ignore error and continue to next module
[3] Give up on module
[4] Start shell
[5] Reload configuration
choice: **2<ret>**
...
*** Installing gnutls *** [6/6]
...
*** success *** [6/6]

c) Construire Simple Components

Site web : http://www.dmitry-kazakov.de/ada/components.htm.
Version installée : 4.15.
Récupérer l'archive suivante sur le bureau :
http://www.dmitry-kazakov.de/ada/components_4_15.tgz

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple (si pas déjà fait) :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ mkdir components_4_15
$ cd components_4_15
$ tar xzf ~/Desktop/components_4_15.tgz
# Récupérer les projets GPR apportés sur Blady
$ unzip ~/Desktop/xadalib-2016-diff/components_gpr.zip
$ gprbuild -p -P lib_components.gpr
# Certains sources provoquant une erreur avec gnatdoc sont ignorés :
$ gnatdoc --enable-build --ignore-files="gnat-sockets-connection_state_machine-big_endian-generic_double_precision_ieee_754.ads gnat-sockets-connection_state_machine-big_endian-generic_single_precision_ieee_754.ads gnat-sockets-connection_state_machine-chain_code-generic_integer.ads gnat-sockets-connection_state_machine-chain_code-generic_unsigned.ads gnat-sockets-connection_state_machine-little_endian-generic_double_precision_ieee_754.ads gnat-sockets-connection_state_machine-little_endian-generic_single_precision_ieee_754.ads gnat-sockets-connection_state_machine-http_server-sqlite_browser.ads gnat-sockets-connection_state_machine-http_server-sqlite_browser.ads gnat-sockets-connection_state_machine-modbus_client-synchronous.ads gnat-sockets-connection_state_machine-modbus_client-synchronous.ads" -P lib_components.gpr
$ gprinstall -f -p --prefix=$instxada -P lib_components.gpr
```

Une documentation au format HTML est disponible dans le répertoire $instxada/share/doc/components :

```
$ open $instxada/share/doc/components/strings_edit.htm
$ open $instxada/share/doc/components/tables.htm
$ open $instxada/share/doc/components/components_rm/index.html
```

Des exemples de programme sont disponibles dans le répertoire $instxada/share/examples/components.

d) Construire AICWL

Site web : http://www.dmitry-kazakov.de/ada/aicwl.htm.
Version installée : 3.15.
Récupérer l'archive suivante sur le bureau :
http://www.dmitry-kazakov.de/ada/aicwl_3_15.tgz

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple (si pas déjà fait) :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ mkdir aicwl_3_15
$ cd aicwl_3_15
$ tar xzf ~/Desktop/aicwl_3_15.tgz
# Récupérer les projets GPR apportés sur Blady
$ unzip ~/Desktop/xadalib-2016-diff/aicwl_gpr.zip
$ gprbuild -p -P lib_aicwl.gpr
$ gnatdoc --no-subprojects -P lib_aicwl.gpr
$ gprinstall -f -p --prefix=$instxada -P lib_aicwl.gpr
$ gprbuild -p -P xpm2gtkada/build_xpm2gtkada.gpr
$ gprinstall -f -p --prefix=$instxada -P xpm2gtkada/build_xpm2gtkada.gpr
```

Une documentation au format HTML est disponible dans le répertoire $instxada/share/doc/aicwl :

```
$ open $instxada/share/doc/aicwl/aicwl.htm
$ open $instxada/share/doc/aicwl/gtkada_contributions.htm
$ open $instxada/share/doc/aicwl/aicwl_rm/index.html
```

Des exemples de programme sont disponibles dans le répertoire $instxada/share/examples/aicwl.

Ne pas oublier de configurer XDG_DATA_DIRS avant de lancer les exemples :

```
$ cd $instxada/share/examples/aicwl
$ gprbuild -p -P build_examples.gpr
$ export XDG_DATA_DIRS=$instxada/share
$ ./bin/oscilloscope_plotter
...
```

9. Construire GNATColl

GNAT Component Collection (GNATColl) est une bibliothèque d'usage générale utilisée pour les outils d'AdaCore comme GPS. Elle inclue une vingtaine de composants dont les traces, la mémoire, les chaînes de caractères, les e-mail, la logique trois états, JSON, SQL, ReadLine...

Site web : http://libre.adacore.com/tools/gnat-component-collection.

Récupérer l'archive suivante depuis le site Libre d'AdaCore "http://libre.adacore.com" à la page "Download GNAT GPL" puis cocher "Free Software or Academic Development" et cliquer sur "Build your download package", sélectionner plateforme "x86-64 GNU Linux (64 bits)" "GNAT GPL 2016" puis "GNATcoll GPL 2016" :

- gnatcoll-gpl-2016-src.tar.gz 5.69 MB May 16, 2016

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ tar xzf ~/Desktop/gnatcoll-gpl-2016-src.tar.gz
$ cd gnatcoll-gpl-2016-src/
# Utilise GMP installé auparavant avec GNUTLS
$ ./configure CC='gcc -g -O0' --prefix=$instxada --enable-build=Debug --enable-gpl --disable-shared
--with-postgresql=/usr --with-gmp=$instxada
$ make
$ make examples
$ make test
$ make install
```

Des exemples de programme sont disponibles dans le répertoire $instxada/share/examples/gnatcoll.

Une documentation au format PDF et HTML est disponible dans le répertoire $instxada/share/doc/gnatcoll.

(Attention la documentation est pré-construite dans l'archive initiale, si "make clean" est exécuté il faudra alors la reconstruire avec les outils sphinx-build et pdflatex)

Un plug-in pour GPS est disponible dans le répertoire $instxada/share/gps/support.

10.Construire Zanyblue

Zanyblue est une bibliothèque native en Ada pour l'internalisation d'un logiciel avec l'affichage de textes traduits dans différentes langues.

Site web : http://zanyblue.sourceforge.net.

Version installée : 1.3.0b.

Récupérer l'archive suivante sur le bureau :

https://sourceforge.net/projects/zanyblue/files/zanyblue-1.3.0b-r3059.tar.gz

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple (si pas déjà fait) :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ tar xzf ~/Desktop/zanyblue-1.3.0b-r3059.tar.gz
$ cd zanyblue-1.3.0b
$ cd src
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xadalib-2016-diff/zanyblue.diff
$ make BUILD=Debug
$ make INSTALL_DIR=$instxada install
$ rm -f $instxada/bin/zbx_gjenkins
$ cd ../doc
$ mkdir -p $instxada/share/doc/zanyblue
$ tar -cf - . l tar -C $instxada/share/doc/zanyblue -xf -
```

Une documentation au format PDF et HTML est disponible dans le répertoire $instxada/share/doc/zanyblue :

```
$ open $instxada/share/doc/zanyblue/index.html
$ open $instxada/share/doc/zanyblue/ref/index.html
```

11.Construire PragmARC

PragmAda Reusable Components (PragmARCs) est une collection de composants mathématiques, de dates, de listes proposée par Jeff Carter.

Site web : http://pragmada.x10hosting.com/pragmarc.htm.

Version installée : 07-2016-09.

Récupérer l'archive suivante sur le bureau :

http://pragmada.x10hosting.com/pragmarc-07-2016-08.zip

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple (si pas déjà fait) :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ mkdir pragmarc-07-2016-09
$ cd pragmarc-07-2016-09
$ unzip ~/Desktop/pragmarc-07-2016-09.zip
# Récupérer le projet GPR apporté sur Blady :
$ cp -p ~/Desktop/xnadalib-2016-diff/lib_pragmarc.gpr .
$ gprbuild -p -P lib_pragmarc.gpr
$ gnatdoc -P lib_pragmarc.gpr
$ gprinstall -f -p --prefix=$instxada lib_pragmarc.gpr
```

Une documentation au format HTML est disponible dans le répertoire $instxada/share/doc/pragmarc :

```
$ open $instxada/share/doc/pragmarc/pragmarc_rm/index.html
```

12.Construire Gnoga

Gnoga est une bibliothèque graphique créée nativement en Ada. Ce n'est pas une surcouche Ada à une bibliothèque existante en C ou C++. Sa particularité est de permettre de construire des applications graphiques orientées Web indépendantes de la plateforme. Double indépendance garantie d'une part de fait du langage Ada lui même, Ada assure qu'un code source aura un comportement identique quelque soit la plate-forme d'exécution de part son compilateur (s'il accepte la compilation), d'autre part avec l'utilisation du HTML et Javascript pour le rendu graphique dans un navigateur Web.

Source : http://www.gnoga.com.

Avant de démarrer la compilation, GNAT doit être activé, ajouter son emplacement comme par exemple :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ git clone git://git.code.sf.net/p/gnoga/code gnoga-code
$ cd gnoga-code
$ git checkout V1.2-beta
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xadalib-2016-diff/gnoga.diff
$ make gnoga
$ make html-docs
$ make rm-docs
$ make PREFIX=$instxada install_debug
$ tar -cf - demo tutorial | tar -C $instxada/share/gnoga -xf -
```

Pour une utilisation courante, nous positionnons les variables d'environnement PATH pour une utilisation en ligne de commande et GPR_PROJECT_PATH pour une utilisation avec un projet GPS :

```
$ echo 'PATH=$instxada/bin:$PATH' >> ~/.profile
$ echo 'PATH=$instxada/bin:$PATH' >> ~/.bashrc
$ echo 'export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:
$GPR_PROJECT_PATH' >> ~/.profile
$ echo 'export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:
$GPR_PROJECT_PATH' >> ~/.bashrc
```

Pour une utilisation temporaire, utiliser à chaque fois les commandes suivantes :

```
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr:$GPR_PROJECT_PATH
```

Des démos et tutoriels sont présents respectivement dans les répertoires "$instxada/share/gnoga/demo" et "$instxada/share/gnoga/tutorial".

Une documentation sous forme de manuels utilisateur et d'un manuel de référence est disponible dans le répertoire "$instxada/share/gnoga/html".

```
$ open $instxada/share/gnoga/html/user_guide.html
$ open $instxada/share/gnoga/html/api_summary.html
$ open $instxada/share/gnoga/html/gnoga_rm/index.html
```

Voir l'utilisation de Gnoga avec des exemples sur Blady :
http://blady.pagesperso-orange.fr/a_savoir.html#gnoga

13.Construire les utilitaires Adalog

Adalog offre quelques utilitaires basés sur la bibliothèque ASIS que nous allons installer en premier.

Avant de démarrer la compilation, GNAT et le répertoire d'installation de XNAdaLib doivent être activés, ajouter leur emplacement comme par exemple :

```
$ instxada=/usr/local/xnadalib-2016
$ PATH=$instxada/bin:$PATH
$ export GPR_PROJECT_PATH=$instxada/lib/gnat:$instxada/share/gpr
$ PATH=/usr/local/gnat/bin:$PATH
```

a) ASIS

Ada Semantic Interface Specification (ASIS) est un standard (ISO/IEC 15291:1995) conçu pour être indépendant du compilateur. ASIS pour GNAT est une bibliothèque qui permet de manipuler la structure syntaxique et sémantique d'une unité de compilation Ada pour l'environnement GNAT.

Site web : http://www.adacore.com/asis.

Récupérer l'archive suivante depuis le site Libre d'AdaCore "http://libre.adacore.com" à la page "Download GNAT GPL" puis cocher "Free Software or Academic Development" et cliquer sur "Build your download package", sélectionner plateforme "x86-64 GNU Linux (64 bits)" "GNAT GPL 2016" puis "GNAT Ada 2016" et "Sources" :

asis-gpl-2016-src.tar.gz 5.24 MB May 16, 2016

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ tar xzf ~/Desktop/asis-gpl-2016-src.tar.gz
$ cd asis-gpl-2016-src
$ make
$ make prefix=$instxada install
```

b) AdaControl

Le but de cet utilitaire est de contrôler l'application de règles de style ou de programmation comme la présence de certaines entités, déclarations ou instructions voire la vérification du respect de véritables patrons de conception.

Site web : http://adalog.fr/fr/adacontrol.html.

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ mkdir adalog
$ cd adalog
$ git clone git://git.code.sf.net/p/adacontrol/code adacontrol-code
$ git clone git://git.code.sf.net/p/adacontrol/adalog-comps adacontrol-adalog-comps
$ git clone git://git.code.sf.net/p/adacontrol/adalog-asiscomps adacontrol-adalog-asiscomps
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xadalib-2016-diff/adacontrol.diff
$ make build
# Utilisation de latexpdf
$ PATH=/usr/local/texlive/2014/bin/x86_64-darwin:$PATH
$ make doc
$ make PREFIX=$instxada install
```

Une documentation PDF, HTML, TXT, INFO sous forme d'un manuel utilisateur et d'un manuel de programmation est disponible dans le répertoire "$instxada/share/adacontrol".

c) AdaDep

Le but de cet utilitaire est de donner pour une unité de compilation les entités réellement utilisées d'une bibliothèque.

Site web : http://adalog.fr/fr/composants.html.

Saisir les commandes suivantes dans le Terminal :

```
$ cd /usr/local/src-gtk-osx
$ mkdir adalog
$ cd adalog
$ tar xzf ~/Desktop/adadep-src-1.3r3.zip
$ cd adadep-1.3r3/
# Appliquer les correctifs apportés sur Blady :
$ patch -p0 < ~/Desktop/xadalib-2016-diff/adadep.diff
$ make build
$ make PREFIX=$instxada install
```

Une documentation PDF, HTML, TXT, INFO sous forme d'un manuel utilisateur et d'un manuel de programmation est disponible dans le répertoire "$instxada/share/adadep".

Pascal Pignard, avril-septembre 2011, août 2014, juillet 2015, août-septembre 2016.
http://blady.pagesperso-orange.fr