

Installation de GNAT FSF 15 pour macOS 13

Une seule distribution du compilateur GNAT FSF existe pour macOS :

- le compilateur GNAT FSF 15.1 sur Alire (mai 2025) pour les langages Ada, C et C++

Sommaire

1.	Installation du compilateur GNAT FSF 15 depuis Alire	2
2.	Utilisation avec le Terminal	5
3.	Les commandes utiles avec le Terminal	6

1. Installation du compilateur GNAT FSF 15 depuis Alire

a) Le compilateur

Installez auparavant l'utilitaire *alr* pour avoir accès à la bibliothèque Alire, voir sur [Blady](#).

Lancer le Terminal dans une session administrateur et taper les commandes suivantes :

```
% alr toolchain --select
```

```
Welcome to the toolchain selection assistant
```

```
In this assistant you can set up the default toolchain to be used with any crate  
that does not specify its own top-level dependency on a version of gnat or  
gprbuild.
```

```
If you choose "None", Alire will use whatever version is found in the  
environment.
```

```
① Currently configured: gnat_native=xx.xx
```

```
Please select the gnat version for use with this configuration
```

```
1. gnat_native=15.1.2
```

```
2. None
```

```
3. gnat_external=15.1.0 [Detected at (...)e/alire/toolchains/gnat_native_15.1.2_2166c311/  
bin/gnat]
```

```
4. gnat_arm_elf=15.1.2
```

```
5. gnat_avr_elf=15.1.2
```

```
6. gnat_riscv64_elf=15.1.2
```

```
7. gnat_xtensa_esp32_elf=15.1.2
```

```
8. gnat_arm_elf=14.2.1
```

```
9. gnat_avr_elf=14.2.1
```

```
0. gnat_native=14.2.1
```

```
a. (See more choices...)
```

```
Enter your choice index (first is default):
```

```
> 1
```

```
① Selected tool version gnat_native=15.1.2
```

```
① Choices for the following tool are narrowed down to releases compatible with just selected  
gnat_native=15.1.2
```

```
① Currently configured: gprbuild=xx.xx
```

```
Please select the gprbuild version for use with this configuration
```

```
1. gprbuild=25.0.1
```

```
2. None
```

```
3. gprbuild=22.0.1
```

```
4. gprbuild=21.0.2
```

```
5. gprbuild=21.0.1
```

```
Enter your choice index (first is default):
```

```
> 1
```

```
① Selected tool version gprbuild=25.0.1
```

Le compilateur GNAT et le constructeur de projets GPRbuild s'installent dans le dossier :
`$HOME/.local/share/alire/toolchains`.

b) L'environnement de développement intégré

L'éditeur intégré GNAT Studio ne fait pas partie de cette installation, cependant celui de la livraison de 2019 est toujours utilisable, à installer dans un autre dossier (voir sur Blady) puis déclarer l'alias suivant avec la commande suivante, par exemple :

```
% alias gps=/usr/local/adacore/2019/bin/gps
```

c) Le débogueur

D'autre part, le debugger gdb n'est pas encore fonctionnel, il est bloqué par le système de surveillance de macOS et provoque cette erreur à l'exécution du programme à débbugger :
Unable to find Mach task port for process-id 6633: (os/kern) failure (0x5).
(please check gdb is codesigned - see taskgated(8))

Nous allons suivre la procédure décrite sur le blog de Simon Wright pour qu'il fonctionne avec macOS (forward-in-code.blogspot.com/2018/11/mojave-vs-gdb.html). Lancer le Terminal dans une session administrateur et taper les commandes suivantes :

i) Ouvrir l'application *Trousseau d'accès* dans le dossier *Applications -> Utilitaires*. Sélectionner le menu *Trousseau d'accès -> Assistant de certification -> Créer un certificat...*

Dans la fenêtre qui apparaît :

- . donner le nom *gdb-cert*,
- . le type d'identité à *Racine auto-signée*,
- . le type de certificat à *Signature de code*,
- . cocher le case *Me laisser ignorer les réglages pas défaut*.

Puis cliquer sur le bouton *Continuer* plusieurs fois jusqu'à ce qu'apparaisse le panneau *Indiquez l'emplacement du certificat*.

ii) Sélectionner alors *Système*, puis sur le bouton *Créer*. Une fenêtre d'autorisation de modification du trousseau apparaît, entrer le mot de passe puis cliquer sur le bouton *Modifier le trousseau*. Le certificat est créé, cliquer alors sur le bouton *Terminer*.

iii) Dans la fenêtre des trousseaux, sélectionner le trousseau *Système* et double-cliquer sur le certificat *gdb-cert*. Dans la fenêtre qui apparaît, déployer le triangle *Se fier* et en face de *Lors de l'utilisation de ce certificat* sélectionner *Toujours approuver* puis fermer la fenêtre en cliquant sur sa bulle rouge de fermeture. Une fenêtre d'autorisation de modification des réglages apparaît, entrer le mot de passe puis cliquer sur le bouton *Mettre à jour les réglages*.

Quitter l'application *Trousseau*, ça été long et ce n'est pas encore tout à fait fini. Il nous faut alors redémarrer le Mac (c'est malheureusement nécessaire) puis télécharger le fichier de description *gdb.xml* à partir de Blady :

blady.pagesperso-orange.fr/telechargements/gnat/gdb.xml

Nous pouvons alors enfin signer *GDB* :

```
$ cd <gnat path>/bin
$ codesign --force --sign gdb-cert --entitlements ~/Downloads/gdb.xml gdb
Password:
```

Une fenêtre s'ouvre *macOS souhaite effectuer des modifications*, saisir nom administrateur et mot de passe puis cliquer sur le bouton *Autoriser* pour autoriser la signature.

Si l'erreur *errSecInternalComponent* s'affiche alors vérifiez que vous êtes bien dans une session administrateur.

d) Les bibliothèques *C* / *C++* standards

Apple donne la possibilité de construire des applications non seulement pour macOS mais aussi pour iOS, watchOS, etc, en fournissant un environnement de développement complet pour chaque. Les emplacements *include* et *lib* des bibliothèques *C* standard ne sont plus localisées dans les emplacements standards */usr/include* et */usr/lib* notamment en utilisant GNAT avec le langage *C++*, une erreur se produit :

```
/Users/me/.local/share/alire/toolchains/gnat_native_15.1.2_2166c311/lib/gcc/x86_64-apple-
darwin22.6.0/15.1.0/include/stdint.h:11:16: fatal error:stdint.h: No such file or directory
ou
ld: library not found for -lSystem
```

Vous devez auparavant récupérer l'emplacement de l'environnement de développement courant avec la commande :

```
% xcrun --show-sdk-path
/Applications/Xcode.app/Contents/Developer/Platforms/MacOSX.platform/Developer/SDKs/
MacOSX.sdk
```

Ensuite ajouter cet emplacement à vos compilations, en ligne :

```
% gnatmake hello1 -larges -isysroot /Applications/Xcode.app/Contents/Developer/Platforms/
MacOSX.platform/Developer/SDKs/MacOSX.sdk
```

Ou dans les projets GPR :

```
package Compiler is
  -- ...
  for Default_Switches ("C") use ("-isysroot", "/Applications/Xcode.app/Contents/
Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX.sdk");
  for Default_Switches ("C++") use ("-isysroot", "/Applications/Xcode.app/Contents/
Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX.sdk");
end Compiler;
```

```
package Linker is
  -- ...
  for Default_Switches ("Ada") use ("-isysroot", "/Applications/Xcode.app/Contents/
Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX.sdk");
  for Default_Switches ("C") use ("-isysroot", "/Applications/Xcode.app/Contents/
Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX.sdk");
  for Default_Switches ("C++") use ("-isysroot", "/Applications/Xcode.app/Contents/
Developer/Platforms/MacOSX.platform/Developer/SDKs/MacOSX.sdk");
end Linker;
```

2. Utilisation avec le Terminal

La commande "gnatmake" seule, sans paramètre, donne justement la liste des paramètres possibles. Néanmoins, la simple commande suivante donnera de bons résultats :

```
$ gnatmake hello.adb
```

Le fichier hello.adb étant :

```
with Ada.Text_IO; use Ada.Text_IO;
procedure Hello is
begin
  Put_Line ("Hello again, avec Ada.");
end Hello;
```

Et les résultats ne se font pas attendre :

```
$ gnatmake hello.adb
gcc -c hello.adb
gnatbind -x hello.ali
gnatlink hello.ali
$ ./hello
Hello again, avec Ada.
```

3. Les commandes utiles avec le Terminal

La liste des commandes est obtenue de la façon suivante :

```
$ gnat
GNAT 15.1.0
Copyright 1996-2025, Free Software Foundation, Inc.
List of available commands
GNAT BIND      gnatbind      réalise l'édition des liens des unités Ada compilées
GNAT CHOP      gnat chop     découpe un fichier en unités pour satisfaire les
conventions Gnat
GNAT CLEAN     gnatclean    nettoie les fichiers générés par gnat
GNAT COMPILE   gnatmake -f -u -c compile une entité Ada
GNAT CHECK     gnatcheck    vérifie le code source suivant des règles définies (non
présent avec gnat-osx)
GNAT ELIM      gnatelim     détecte et élimine les sous-programmes inutilisés
GNAT FIND      gnatfind     liste toutes les utilisations d'une entité Ada
GNAT KRUNCH    gnatkr       réduit les noms de fichiers au nombre maximal de lettres
spécifié
GNAT LINK      gnatlink     réalise l'édition des liens de l'exécutable
GNAT LIST      gnatls       liste le contenu des objets générés
GNAT MAKE      gnatmake     utilitaire optimisé de compilation multi-unités
GNAT METRIC    gnatmetric   statistiques sur le code Ada
GNAT NAME      gnatname     réalise la correspondance entre les unités Ada et les
noms des fichiers lorsque ceux-ci ne sont pas au standard GNAT
GNAT PREPROCESS gnatprep     pré-processeur externe
GNAT PRETTY    gnatpp       reformate le source Ada
GNAT STACK     gnatstack    calcul la taille de pile mémoire maximale théorique (non
présent avec gnat-gpl)
GNAT STUB      gnatstub     crée le squelette d'un corps d'une spécification
GNAT TEST      gnat test    crée ou exécute la suite de test unitaire
GNAT XREF      gnatxref     utilitaire d'édition des références croisées
```

De même chacune des commandes ci-dessus exécutée sans argument affichera justement la liste des arguments possibles.

```
$ gnatmake (extrait)
Usage: gnatmake opts name {[ -cargs opts] [-bargs opts] [-largs opts]}
name is a file name from which you can omit the .adb or .ads suffix
gnatmake switches:
--version  Display version and exit
--help     Display usage and exit
-c         Compile only
-f         Force recompilations of non predefined units
-k         Keep going after compilation errors
-m         Minimal recompilation
-n         Check objects up to date, output next file to compile if not
-o name    Choose an alternate executable name
-p         Create missing obj, lib and exec dirs
-Pproj     Use GNAT Project File proj
-s         Recompile if compiler switches have changed
-v         Display reasons for all (re)compilations
```

To pass an arbitrary switch to the Compiler, Binder or Linker:

- cargs opts opts are passed to the compiler
- bargs opts opts are passed to the binder
- largs opts opts are passed to the linker
- margs opts opts are passed to gnatmake

Compiler switches (passed to the compiler by gnatmake):

- ldir Specify source files search path
- gnat2012 Ada 2012 mode (default)
- gnat2022 Ada 2022 mode

Et aussi avec gcc :

\$ gcc --help

...

Pour en savoir plus : voir l'utilisation avancée des outils GNAT, l'environnement de développement GPS ainsi que du dévermineur GDB et des exceptions Ada sur la page à savoir de Blady.

Pascal Pignard, juin 2018, mai 2019, janvier 2020, février 2020, octobre 2020, juillet 2021, juillet-décembre 2022, mai 2023, septembre 2024, février-juillet 2025.