Premiers pas avec ALIRE

Ada LIbrary REpository (Alire - alire.ada.dev) est un répertoire de programmes Ada accompagné d'un utilitaire "alr" pour trouver, récupérer, construire et publier ses programmes. Cette initiative récente (début 2018) comble un manque dans le monde Ada. Encore maintenant, pour trouver une bibliothèque Ada, la recherche porte sur plusieurs sites Internet sans relation directe les uns avec les autres. L'ambition d'Alire est de fédérer les acteurs Ada pour qu'ils alimentent ce répertoire de leur production. Alire contient à ce jour plus de 150 projets (alire.ada.dev/crates.html).

Pour nos premiers pas avec Alire nous allons voir comment installer Alire pour publier nos programmes.

Un compte GitHub est nécessaire pour publier ses créations.

Configuration : macOS 12.4, Xcode 13.4, GNAT Community 2021.

Sommaire

1. Installer Alire

La version installée est v1.2.0.
Télécharger le fichier suivant sur le bureau du Mac : github.com/alire-project/alire/releases/download/v1.2.0/alr-1.2.0-bin-x86_64-macos.zip.

Saisir les commandes suivantes dans le Terminal :

```
% cd ~/Desktop
% unzip alr-1.2.0-bin-x86_64-macos
% cd alr-1.2.0-bin-x86_64-macos/bin
% xattr -r -d com.apple.quarantine alr
% mkdir $HOME/bin  # si pas déjà créé
% cp -p alr-1.2.0-bin-x86_64-macos/bin/alr $HOME/bin
```

L'utilitaire alr est installé dans le dossier $HOME/bin.

Pour une utilisation courante, nous positionnons la variable d'environnement PATH pour une utilisation en ligne de commande (si pas déjà fait):

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

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

```
% PATH=$HOME/bin:$PATH
```

Le compilateur Ada GNAT et les outils GPR doivent être aussi présents dans le PATH.

2. Première utilisation

a) Liste des commandes

La liste des commandes est obtenue de la façon suivante :

% alr

```
Ada Library Repository manager
USAGE:
alr [global options] command [command options] [arguments]
alr help [<command>|<topic>]

GLOBAL OPTIONS
  -f, --force          Keep going after a recoverable troublesome situation
  -h, --help           Display general or command-specific help
  --version            Displays version and exits
  -q                   Limit output to errors
  -v                   Be more verbose (use twice for extra detail)
  -d, --debug[ARG]     Enable debug-specific log messages
```

```
COMMANDS
  General
  help          Shows help on the given command/topic
  config        List, Get, Set or Unset configuration options
  printenv      Print the build environment variables
  toolchain     Manage Alire-provided toolchains
  version       Shows detailed version, configuration, and environment information

  Index
  get           Fetches a crate release
  index         Manage indexes used by current configuration
  init          Creates a new working release with alire metadata, or generate metadata
  pin           Pin dependencies to exact versions
  search        Search a string in release names and properties
  show          See information about a release
  update        Updates alire catalog and working release dependencies
  with          Manage release dependencies

  Build
  action        List or manually trigger action hooks
  build         GPRbuild current working release
  clean         GPRclean working release and manage cached releases
  dev           Developer helpers
  edit          Start GNATstudio with Alire build environment setup
  run           Launch an executable built by the release
  test          Tests the compilation of all or some releases
  exec          Run the given command in the alire project context

  Publish
  publish       Help with the publication of a new release

TOPICS
  aliases       User defined command aliases
  identifiers   Naming rules for crate and index names
  toolchains    Configuration and use of toolchains

ALIASES
  gnatcov       exec -P2 -- gnatcov
  gnatprove     exec -P1 -- gnatprove
```

b) Liste des programmes référencés sur Alire

La liste des programmes du catalogue Alire est obtenue de la façon suivante :
(Index local : $HOME/.config/alire/indexes/communauty)

% alr -q search --list

```
NAME   STATUS      VERSION     DESCRIPTION       NOTES
aaa                0.2.1       Alex's Ada Assortment (of miscellaneous utilities)
ada_fuse       X   1.0.1       Ada bindings for Fuse (Filesystem in Userspace)
ada_lua        X   0.1.0       An Ada binding for lua
...
zipada             56.0.1      Manage Zip Archives and raw LZMA streams
zipdcf             2.0.2       Tools that can (un)zip document container files
zlib_ada       X   1.3.0       ZLib for Ada thick binding
```

STATUS
E: the release is externally provided.
S: the release is available through a system package.
U: the release is not available in the current platform.
X: the release has dependencies that cannot be resolved.

c) Mise à jour de l'index local

% alr index --update-all

d) Configuration globale

La liste des options globales est obtenue de la façon suivante :
(Configuration : $HOME/.config/alire/config.toml)

% alr config --list --global
user.email=moi@me.fr
user.github_login=moi
user.name=moi

Ou indépendamment pour chaque option :

% alr config --global --get user.name
moi

L'éditeur est par défaut GNAT Studio. Si vous êtes comme moi restés à GPS, voici comment le positionner par défaut :

% alr config --global --set editor.cmd "/usr/local/adacore/2019/bin/gps"

e) Informations générales

L'ensemble des informations générales est obtenue de la façon suivante :

% alr version
APPLICATION
alr version: 1.2.0
libalire version: 1.2.0
compilation date: 2022-05-18 13:18:06
compiler version: Community 2020 (20200818-84)

CONFIGURATION
config folder: /Users/opt/.config/alire
force flag: FALSE
non-interactive flag: FALSE
community index branch: stable-1.2
compatible index versions: ^1.1 & <=1.2
indexes folder: /Users/opt/.config/alire/indexes
indexes metadata: OK
index #1: (community) git+https://github.com/alire-project/alire-index#stable-1.2
toolchain assistant: disabled
tool #1 gnat: gnat_external=2021.0.0
tool #2 gprbuild: gprbuild=2021.0.0

WORKSPACE
root status: OUTSIDE
root release: N/A
root load error: N/A
root folder: N/A
current folder: /Users/opt/.config/alire/indexes/community/repo

SYSTEM
distribution: DISTRO_UNKNOWN
host-arch: X86_64
os: MACOS
target: NATIVE
toolchain: USER
word-size: BITS_64

La version de alr est juste donnée par :

% alr --version
alr 1.2.0

La chaine de compilation Ada est obtenue de la façon suivante :

% alr toolchain
CRATE VERSION STATUS NOTES
gprbuild 2021.0.0 Default Detected at /opt/gnat-ce-2021/bin/gprbuild
gnat_external 2021.0.0 Default Detected at /opt/gnat-ce-2021/bin/gnat

3. Récupérer, construire et exécuter un programme Alire

Le petit programme Hello est suffisamment simple pour illustrer le processus :

- Rechercher le programme avec la commande search
- Afficher la description du programme avec la commande show
- Récupérer le programme avec la commande get
- Construire le programme avec la commande build
- Exécuter le programme avec la commande run

Ce programme a une dépendance avec la bibliothèque libhello.

Saisir les commandes suivantes dans le Terminal :

% cd <dossier de test>
% alr search hello

```
NAME      STATUS  VERSION  DESCRIPTION                                    NOTES
hello             1.0.1    "Hello, world!" demonstration project
libhello          1.0.0    "Hello, world!" demonstration project support library
```

% alr show hello

```
hello=1.0.1: "Hello, world!" demonstration project
Origin: source archive v1.0.1.tar.gz at https://github.com/alire-project/hello/archive/v1.0.1.tar.gz with hash
sha512:dcc36a2f6f483fab49a73db30247cfd6b53338adb45d728827573a84894456cb65ac29e64d276f38c8c73a7574f871d1d384ef4b9a2c65d3f433501068b95ad9
Properties:
  Description: "Hello, world!" demonstration project
  Maintainer: alejandro@mosteo.com
  Maintainers_Logins: mosteo
  Name: hello
  Version: 1.0.1
Dependencies (direct):
  libhello^1.0
```

% alr get hello

```
...
hello=1.0.1 successfully retrieved.
Dependencies were solved as follows:
  + libhello 1.0.0 (new)
```

% cd hello_1.0.1_dcc36a2f
% alr build

```
Setup
  [mkdir]       object directory for project Libhello
  [mkdir]       library directory for project Libhello
  [mkdir]       object directory for project Hello
Compile
  [Ada]         hello.adb
  [Ada]         libhello.adb
Build Libraries
  [gprlib]      hello.lexch
  [archive]     libhello.a
  [index]       libhello.a
Bind
  [gprbind]     hello.bexch
  [Ada]         hello.ali
Link
  [link]        hello.adb
```

% alr run
gprbuild: "hello" up to date
Hello, world!

Et voilà c'est tout. Alire cache la complexité dans un dossier "alire" où on retrouve les dépendances. La description du projet est dans le fichier catalogue "alire.toml". Le contenu des fichiers catalogue est détaillé sur cette page : alire.ada.dev/docs/#catalog-format-specification.

4. Créer, construire et exécuter un programme Alire

Cette fois nous allons créer notre propre programme de bienvenue avec un message coloré et clignotant en utilisant les séquences ANSI disponible dans Alire.

Le petit programme Hello est suffisamment simple pour illustrer le processus :

- Créer un nouveau projet Alire avec la commande init
- Rechercher la bibliothèque ANSI avec la commande search
- Afficher la description de la bibliothèque avec la commande show
- Ajouter la dépendance de la bibliothèque avec la commande with
- Éditer notre programme avec la commande edit
- Construire et exécuter le programme avec la commande run

Lors de sa première exécution, la commande "init" demande la saisie de renseignements sur un identifiant de compte GitHub, un nom et une adresse mel.

Saisir les commandes suivantes dans le Terminal :

% cd <dossier de test>
% alr init --bin bonjour
Alire needs some user information to initialize the crate author and
maintainer, for eventual submission to the Alire community index. This
information will be interactively requested now.
You can edit this information at any time with 'alr config'
Please enter your GitHub login: (default: 'github-username')
> **Blady-Com**
Please enter your full name: (default: 'Your Name')
> **Pascal Pignard**
Please enter your email address: (default: 'example@example.com')
> **blady.net@orange.fr**
✓ bonjour initialized successfully.
% cd bonjour
% alr search ansi
NAME STATUS VERSION DESCRIPTION NOTES
ansiada 0.1.0 Comprehensive ANSI control sequences for terminal output
% alr show ansiada
Origin: commit 27a89150c1f5481a821722601a3b6d4a5368596c from https://github.com/mosteo/ansi-ada
Properties:
Description: Comprehensive ANSI control sequences for terminal output
Executable: ansi-demo
GPR Scenario: ANSIADA_BUILD_MODE := On_Demand | Static_Lib | Shared_Lib
GPR External: ANSIADA_BUILD_MODE := On_Demand
License: MIT
Long_Description: # ANSI-Ada

...
ANSI control sequences for the Ada language.
This library consists of a single pure package for the generation of escape
sequences that allow to control, in ANSI-enabled TTYs:
* Text color and styles
* Cursor position
* Clearing of parts of the terminal
...
  Maintainer: alejandro@mosteo.com
  Maintainers_Logins: mosteo
  Name: ansiada
  Project_File: ansi.gpr
  Tag: console
  Tag: terminal
  Tag: tty
  Version: 0.1.0
% alr with ansiada
Requested changes:
  ✓ ansiada ~0.1.0 (add)
Changes to dependency solution:
  + ansiada 0.1.0 (new)
Do you want to proceed?
[Y] Yes [N] No (default is Yes) **<CR>**
Using default: Yes
Cloning into '/Users/opt/Essais/bonjour/alire/cache/dependencies/ansiada_0.1.0_27a89150'...
remote: Enumerating objects: 45, done.
remote: Counting objects: 100% (45/45), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 45 (delta 15), reused 32 (delta 10), pack-reused 0
Unpacking objects: 100% (45/45), done.
Do you want Alire to automatically update your project file with the new dependency solution?
[Y] Yes [N] No (default is Yes) **<CR>**
Using default: Yes
Do you want Alire to remember this choice?
[Y] Yes [N] No (default is No) **<CR>**
Using default: No
% alr edit &
Editing crate with: ['/usr/local/adacore/2019/bin/gps' '']
Modifier le fichier bonjour.adb en le remplaçant par :

```
with Ada.Text_IO;
with ANSI; use ANSI;
procedure Bonjour is
begin
  Ada.Text_IO.Put_Line
    (Wrap
      (Text       => "Bonjour avec Alire !",
       Style => Blink,
       Foreground => Foreground (Red),
       Background => Background (Cyan)));
end Bonjour;
```

% alr run
Compile
  [Ada]   bonjour.adb
  [Ada]   ansi.ads
Bind
  [gprbind]   bonjour.bexch

```
   [Ada]          bonjour.ali
Link
   [link]         bonjour.adb
Bonjour avec Alire !
```

5. Construire et publier une bibliothèque existante (GitHub)

Nous allons publier une bibliothèque de notre crue déjà existante sur notre GitHub.
Nous prendrons la bibliothèque UXStrings suffisamment simple pour illustrer le processus :

- Créer un nouveau projet Alire pour notre bibliothèque existante avec la commande init
- Compléter la description du catalogue
- Construire la bibliothèque avec la commande build
- Vérifier la cohérence du catalogue avec la commande version
- Publier la bibliothèque avec la commande publish
- Demander le référencement de la bibliothèque dans Alire

Tout d'abord, il nous faut définir un nom qui puisse être référencé dans le catalogue d'Alire. Ce nom doit se conformer aux règles édictées par Alire :

- Ne pas déjà être un nom existant
- Avoir une longueur comprise entre 3 et 64 caractères
- Être composé de caractères alpha-numérique

Nous choisissons ici le nom "uxstrings". Notre projet pourra alors devenir un "crate", nom donné par Alire aux projets indexés.

Saisir les commandes suivantes dans le Terminal :

```
% alr init --in-place --no-skel --lib uxstrings
✓ uxstrings initialized successfully.
# Modifier le fichier alire.toml comme l'exemple proposé ci-dessous
% cat alire.toml
        name = "uxstrings"
        description = "Unicode Extended Strings utilities"
        version = "0.1.2+alpha-20210226"
        tags = ["unicode", "dynamic", "string"]
        authors = ["Pascal Pignard"]
        maintainers = ["Pascal Pignard <blady.net@orange.fr>"]
        maintainers-logins = ["Blady-Com"]
        website = "https://github.com/Blady-Com/UXStrings"
        licenses = "CECILL-2.1"
        project-files = ["lib_uxstrings1.gpr"]
        [[depends-on]]
        gnat = ">=2020"
```

La description du projet est dans le fichier catalogue "alire.toml". Le contenu des fichiers catalogue est détaillé sur cette page : alire.ada.dev/docs/#catalog-format-specification. Nous avons modifié les attributs :

- Name (inchangé) : le nom du crate
- Description (à modifier) : une phrase décrivant notre bibliothèque
- Version (à modifier) : un triplet ou plus indiquant la version (en conformité avec la grammaire semver.org)

- Tags (à ajouter) : une liste de mots associés à notre bibliothèque
- Authors (inchangé) : une liste des auteurs
- Maintainers (inchangé) : une liste des propriétaires du compte GitHub
- Maintainers-login (inchangé) : l'identifiant du compte GitHub
- Website (à ajouter) : un lien sur le site de la bibliothèque
- Licenses : un identifiant de la licence utilisée (liste sur le site SPDX spdx.org/licenses)
- Project-files (à ajouter) : la liste des projets GNAT de la bibliothèque (par défaut <nom du crate>.gpr)
- Depends-on (à ajouter) : les dépendances à des ressources externes

Ne pas définir dans ce fichier l'attribut "origin", il sera ajouter automatiquement au moment de la publication.

% alr build

```
Compile
  [Ada]          uxstrings1.adb
   [Ada]         uxstrings-text_io1.adb
...
Build Libraries
   [gprlib]      uxstrings.lexch
   [archive]     libuxstrings.a
   [index]       libuxstrings.a
```

La publication sera réalisée dans le projet "alire-index" sur GitHub (github.com/alire-project/alire-index). Nous devons d'abord créer un fork de ce projet dans notre GitHub puis le cloner.

Lancer la commande de publication avec le projet Github et le commit correspondant :

% alr publish github.com/Blady-Com/UXStrings.git 194041529e9b3b36a286748f3ae3baf4a6247c61

```
✓ Local repository is clean.
✓ Revision exists in local repository (194041529e9b3b36a286748f3ae3baf4a6247c61).
Publishing assistant: step 1 of 6: Verify origin URL
✓ Origin is of supported kind: GIT
✓ Origin is hosted on trusted site: github.com
Publishing assistant: step 2 of 6: Verify GitHub infrastructure
✓ User has a GitHub account: Blady-Com
✓ User has forked the community repository
✓ User's fork contains base branch: stable-1.0
Publishing assistant: step 3 of 6: Deploy sources
Cloning into 'alr-ychd.tmp'...
remote: Enumerating objects: 176, done.
remote: Counting objects: 100% (176/176), done.
remote: Compressing objects: 100% (100/100), done.
remote: Total 176 (delta 103), reused 145 (delta 72), pack-reused 0
Receiving objects: 100% (176/176), 110.86 KiB | 692.00 KiB/s, done.
Resolving deltas: 100% (103/103), done.
Publishing assistant: step 4 of 6: Build release
No dependency solution found, updating workspace...
Dependencies automatically updated as follows:
  + gnat 2020.0.0 (new)
Setup
   [mkdir]       object directory for project Lib_UXStrings1
```

```
   [mkdir]       library directory for project Lib_UXStrings1
Compile
...
Build Libraries
   [gprlib]      uxstrings.lexch
   [archive]     libuxstrings.a
   [index]       libuxstrings.a
✓ Build succeeded.
Publishing assistant: step 5 of 6: User review
The release to be published contains this information:
uxstrings=0.1.2+alpha-20210226: Unicode Extended Strings utilities
Origin: commit 194041529e9b3b36a286748f3ae3baf4a6247c61 from https://github.com/Blady-
Com/UXStrings.git
Properties:
   Author: Pascal Pignard
   Description: Unicode Extended Strings utilities
   License: CECILL-2.1
   Maintainer: Pascal Pignard <blady.net@orange.fr>
   Maintainers_Logins: Blady-Com
   Name: uxstrings
   Project_File: lib_uxstrings1.gpr
   Tag: unicode
   Tag: dynamic
   Tag: string
   Version: 0.1.2+alpha-20210226
   Website: https://github.com/Blady-Com/UXStrings
Dependencies (direct):
   gnat>=2020
Do you want to proceed with this information?
[Y] Yes  [N] No  (default is Yes) <CR>
Using default: Yes
Publishing assistant: step 6 of 6: Generate index manifest
✓ Your index manifest file has been generated at /Users/opt/Essais/UXStrings/alire/releases/
uxstrings-0.1.2+alpha-20210226.toml
ⓘ Please upload this file to https://github.com/Blady-Com/alire-index/upload/stable-1.0/index/ux/
uxstrings to create a pull request against the community index.
% cat alire/releases/uxstrings-0.1.2+alpha-20210226.toml
name = "uxstrings"
description = "Unicode Extended Strings utilities"
version = "0.1.2+alpha-20210226"
tags = ["unicode", "dynamic", "string"]
authors = ["Pascal Pignard"]
maintainers = ["Pascal Pignard <blady.net@orange.fr>"]
maintainers-logins = ["Blady-Com"]
website = "https://github.com/Blady-Com/UXStrings"
licenses = "CECILL-2.1"
project-files = ["lib_uxstrings1.gpr"]
[[depends-on]]
gnat = ">=2020"
[origin]
commit = "194041529e9b3b36a286748f3ae3baf4a6247c61"
url = "git+https://github.com/Blady-Com/UXStrings.git"
```

L'attribut origin a été ajouté avec nos indications de publication dans le fichier généré uxstrings-0.1.2+alpha-20210226.toml.

Depuis le clone d'alire-index, avec GIT, nous créons une nouvelle branche du nom du fichier TOML : "uxstrings-0.1.2" pour s'y positionner.

Comme indiqué, nous copions ce fichier à l'emplacement indiqué (il y a une erreur dans le chemin : ne pas prendre en compte "upload") de notre clone d'alire-index.

Puis nous commitons et poussons cette modification sur notre GitHub.

Sur notre GitHub, nous demandons une pull request pour le projet alire-index.

Des vérifications automatiques vont se lancer puis la pull request sera en attente de prise en compte par Alire.

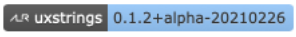
Une fois la pull request acceptée, notre programme fait partie de la bibliothèque Alire. Il est accessible en ligne dans la liste des Crates : alire.ada.dev/crates.html, par exemple pour UXStrings : alire.ada.dev/crates/uxstrings.

Un badge référençant la page de notre programme sur Alire peut être ajouté dans le texte de présentation du programme sur GitHub. Si notre texte est en Markdown, il suffit d'insérer les lignes suivantes, par exemple pour UXStrings :

[![Alire](https://img.shields.io/endpoint?url=https://alire.ada.dev/badges/uxstrings.json)](https://alire.ada.dev/crates/uxstrings.html)

On obtient :

Unicode Extended Strings (UXStrings)



Erreur possible :

% alr publish https://github.com/Blady-Com/j2ada.git 7e5314b4f738f6a3ab20592662a3c27ac1280dc7

Publishing assistant: step 1 of 6: Verify origin URL
✓ Origin is of supported kind: GIT
✓ Origin is hosted on trusted site: github.com

Publishing assistant: step 2 of 6: Verify GitHub infrastructure
✓ User has a GitHub account: Blady-Com
✓ User has forked the community repository
error: Could not complete the publishing assistant:
error: Your index fork is missing the current base branch (stable-1.2) for pull requests to the community repository:
error: Please synchronize this branch and try again:
error: Your fork URL is: https://github.com/Blady-Com/alire-index

Cela indique que la notre birfucation (fork) de l'index n'est pas à jour.

Pascal Pignard, mars 2021, juin 2022.