Comment avoir plus de paquets pour Guix ?

Simon Tournier

INSERM US 53 - CNRS UAR 2030
simon.tournier@u-paris.fr

7 julliet 2022

https://hpc.guix.info

CaféGuix

Université Paris Cité

Avant de commencer : annonce publicitaire

Journée Guix, 16–18 septembre 2022, IRILL, Paris

`https://10years.guix.gnu.org/`

Vendredi 16 Sept. : « Reproducible deployment for reproducible research »

Nous allons discuter de...

```
$ guix search julia csv | wc -l
0
```

Que faire ?

Qu'est-ce qu'un canal (*channel*) ?

https://guix.gnu.org/manual/devel/en/guix.html#Channels

un canal = un dépôt Git

Pour les curieux :

```
$ ls $HOME/.cache/guix/checkouts/
last-expiry-cleanup
pjmkglp4t7znuugeurpurzikxq3tnlaywmisyr27shj7apsnalwq

$ git -C $HOME/.cache/guix/checkouts/pjmkg.../ remote -v
origin https://git.savannah.gnu.org/git/guix.git (fetch)
origin https://git.savannah.gnu.org/git/guix.git (push)
```

Qu'est-ce qu'un canal (*channel*) ? (2)

Ce que fait `guix pull`, c'est :

- mise à jour du dépôt Git (`git pull`)
- « compilation » pour créer le nouveau Guix

Cette compilation, qui semble parfois longue, est ce qui permet
le voyage dans le temps (`guix time-machine`).

https://guix.gnu.org/en/blog/2018/multi-dimensional-transactions-and-rollbacks-oh-my/
https://guix.gnu.org/en/blog/2021/outreachy-guix-git-log-internship-wrap-up/

Qu'est-ce qu'un canal (*channel*) ? (digression)

- le Guix courant est construit à partir de quel *commit* ?

```
$ guix describe
Generation 82 Jul 07 2022 01:05:31 (current)
  guix 06493e7
    repository URL: https://git.savannah.gnu.org/git/guix.git
    branch: master
    commit: 06493e738825598447f5b45d7100ca7eff8b669d
```

Qu'est-ce qu'un canal (*channel*) ? (digression)

- le Guix courant est construit à partir de quel *commit* ?

```
$ guix describe
Generation 82 Jul 07 2022 01:05:31 (current)
  guix 06493e7
    repository URL: https://git.savannah.gnu.org/git/guix.git
    branch: master
    commit: 06493e738825598447f5b45d7100ca7eff8b669d
```

- `guix time-machine` fait comme `guix pull` temporairement

```
$ guix time-machine --commit=c2ce62b6db -- describe | grep commit
    commit: c2ce62b6db8020931f7eb7fa29f09e70f3e25f8c
$ git -C $HOME/.cache/guix/checkouts/pjmkg.../ log -1 --format="%h"
3f70501532
```

Qu'est-ce qu'un canal (*channel*) ? (digression)

- le Guix courant est construit à partir de quel *commit* ?

```
$ guix describe
Generation 82 Jul 07 2022 01:05:31 (current)
  guix 06493e7
    repository URL: https://git.savannah.gnu.org/git/guix.git
    branch: master
    commit: 06493e738825598447f5b45d7100ca7eff8b669d
```

- `guix time-machine` fait comme `guix pull` temporairement

```
$ guix time-machine --commit=c2ce62b6db -- describe | grep commit
    commit: c2ce62b6db8020931f7eb7fa29f09e70f3e25f8c
$ git -C $HOME/.cache/guix/checkouts/pjmkg.../ log -1 --format="%h"
3f70501532
```

⇒ l'état interne du dépôt Git n'est pas toujours le même état

Qu'est-ce qu'un canal (*channel*) ? (digression)

- le Guix courant est construit à partir de quel *commit* ?

```
$ guix describe
Generation 82 Jul 07 2022 01:05:31 (current)
  guix 06493e7
    repository URL: https://git.savannah.gnu.org/git/guix.git
    branch: master
    commit: 06493e738825598447f5b45d7100ca7eff8b669d
```

- `guix time-machine` fait comme `guix pull` temporairement

```
$ guix time-machine --commit=c2ce62b6db -- describe | grep commit
    commit: c2ce62b6db8020931f7eb7fa29f09e70f3e25f8c
$ git -C $HOME/.cache/guix/checkouts/pjmkg.../ log -1 --format="%h"
3f70501532
```

Tout est transparent pour l'utilisateur et il ne faut pas s'en soucier !

Quels canaux ?

https://hpc.guix.info/channels

N'hésitez pas à demander l'ajout de votre canal !

Trois façons :

- `guix pull` **puis** `guix search julia csv`
  avec la liste des canaux sauvegarder là `$HOME/.config/guix/channels.scm`

Oui, il est encore difficile de chercher dans les canaux. :-)

Quels canaux ?

https://hpc.guix.info/channels

N'hésitez pas à demander l'ajout de votre canal !

Trois façons :

- `guix pull` **puis** `guix search julia csv`
  avec la liste des canaux sauvegarder là `$HOME/.config/guix/channels.scm`
- `guix pull -C liste-canaux.scm` **puis** `guix search julia csv`

Oui, il est encore difficile de chercher dans les canaux. :-)

Quels canaux ?

https://hpc.guix.info/channels

N'hésitez pas à demander l'ajout de votre canal !

Trois façons :

- `guix pull` **puis** `guix search julia csv`
  avec la liste des canaux sauvegarder là `$HOME/.config/guix/channels.scm`
- `guix pull -C liste-canaux.scm` **puis** `guix search julia csv`
- `guix time-machine -C liste-canaux.scm -- search julia csv`

Oui, il est encore difficile de chercher dans les canaux. :-)

Ma façon de faire *(peut-être pas la bonne)*

```
guix pull
```

```
$ cat $HOME/.config/guix/channels.scm
(use-modules (guix ci))

(list (channel-with-substitutes-available
       %default-guix-channel
       "https://ci.guix.gnu.org"))
```

(Je ne fais très peu `guix pull` mais presque exclusivement `guix time-machine`, surtout avec l'efficacité des mécanismes de *cache*.)

Ma façon de faire (2) *(peut-être pas la bonne)*

```
guix time-machine -C $HOME/.config/guix/extra-channels.scm -- search
```

```
$ cat $HOME/.config/guix/extra-channels.scm
(list
 (channel                ; .guix.channel depends on channel:
  (name 'bimsb-nonfree) ; github.com/BIMSBbioinfo/guix-bimsb.git
  (url "https://github.com/BIMSBbioinfo/guix-bimsb-nonfree.git")
  (branch "master"))
 (channel
  (name 'past)
  (url "https://gitlab.inria.fr/guix-hpc/guix-past.git")
  (branch "master"))
 %default-channels)
```

Quand le paquet n'est pas disponible ?

```
$ guix time-machine -C all-channels-on-Earth.scm \
       -- search julia csv | wc -l
0
```

Il faut donc empaqueter !

Empaqueter est un vrai *artisanat*

L'artisanat est la transformation de produits ou la mise en œuvre de services grâce à un savoir-faire particulier et hors contexte industriel de masse : l'artisan assure en général tous les stades de sa transformation [...]
https://fr.wikipedia.org/wiki/Artisanat

Points clés

- Il n'y a pas de secret, empaqueter prend du temps
- La difficulté est très variable, et dépend aussi l'écosystème

  p. ex. R est plus facile que Python
- Il y a deux situations :
  - un paquet totalement nouveau
  - un paquet variant

  p. ex. une autre version
- L'option `--load-path`/`-L` facilite le cycle de tests/erreurs

Exemple (0)

```
$ cat path/to/more-packages/custom.scm
(define-module (custom)
  #:use-module (guix packages)
  #:use-module (guix git-download)
  #:use-module (guix build-system julia)
  #:use-module ((guix licenses) #:prefix license:)
  #:use-module (gnu packages julia-xyz)
[...]
```

guix *cmd* -L path/to/more-packages

cmd = `show` ou `build` ou `lint` ou autres

```
(define-public julia-parsers-variant
  (package
    (name "julia-parsers")
    (version "2.2.4")
    (source (origin
              (method git-fetch)
              (uri (git-reference
                    (url "https://github.com/JuliaData/Parsers.jl")
                    (commit (string-append "v" version))))
              (file-name (git-file-name name version))
              (sha256
               (base32 "09v2x9yd1wdp74hzsf6218dpamlf2hb5nkmixqb4bc53ll8hpw4i"))
    (build-system julia-build-system)
    (home-page "https://github.com/JuliaData/Parsers.jl")
    (synopsis "Fast parsing machinery for basic types in Julia")
    (description "@code{Parsers.jl} is a collection of type parsers and
utilities for Julia.")
    (license license:expat)))
```

Exemple (1)

```
$ guix show -L more-packages/ julia-parsers | recsel -p name,version
name: julia-parsers
version: 2.2.4

name: julia-parsers
version: 1.1.0
```

`julia-parsers@2.2.4` ne dépend pas de `julia-parsers@1.1.0`

```
(define-public julia-weakrefstrings-variant
  (package
    (inherit julia-weakrefstrings)
    (name "julia-weakrefstrings")
    (version "1.4.0")
    (source
      (origin
        (method git-fetch)
        (uri (git-reference
               (url "https://github.com/JuliaData/WeakRefStrings.jl")
               (commit (string-append "v" version))))
        (file-name (git-file-name name version))
        (sha256
         (base32 "1ca94bpsjqrap2y9wlixspnisfkcms7aax0kpv7yn0v2vs9481wk"))))
    (propagated-inputs
     (modify-inputs (package-propagated-inputs julia-weakrefstrings)
       (append julia-inlinestrings)
       (replace "julia-parsers" julia-parsers-variant)))))
```

Exemple (2)

```
$ guix show -L more-packages/ julia-weakrefstrings | recsel -p name,version
name: julia-weakrefstrings
version: 1.4.0

name: julia-weakrefstrings
version: 1.1.0
```

- `julia-weakrefstrings@1.4.0` dépend de `julia-weakrefstrings@1.1.0` (inherit)
- `julia-weakrefstrings@1.4.0` dépend de `julia-parsers@2.2.4` (replace)

Peut-être pas judicieux ?

```
  #:use-module ((gnu packages julia-xyz) #:hide (julia-parsers)))
...

(define-public julia-parsers
...

(define-public julia-weakrefstrings-variant
  (package (inherit julia-weakrefstrings)
...
    (propagated-inputs
     (modify-inputs (package-propagated-inputs julia-weakrefstrings)
       (append julia-inlinestrings)
       (replace "julia-parsers" julia-parsers)))))
```

Après du travail...

```
$ ag --scheme define-public more-packages/ | wc -l
5
```

```
$ guix search -L more-packages/ julia csv | recsel -p name,version
name: julia-csv
version: 0.10.4
$ guix shell -L more-packages/ julia julia-csv \
       -- julia -e 'using CSV; print(methods(CSV.File))'
# 6 methods for type constructor:
[1] CSV.File(ctx::CSV.Context) in CSV at /gnu/store/...-profile/share/julia/loa
[2] CSV.File(ctx::CSV.Context, chunking::Bool) in CSV at /gnu/store/...-profile
...
```

Coupure publicitaire

Journée Guix, 16–18 septembre 2022, IRILL, Paris

`https://10years.guix.gnu.org/`

Vendredi 16 Sept. : « Reproducible deployment for reproducible research »

On souhaite partager

Le plus souhaitable est de contribuer à un canal déjà existant

un canal = un dépôt Git

```
cd path/to/more-packages
git init
git add *.scm
git commit -m 'Message utile'
```

Et voilà !

https://guix.gnu.org/manual/devel/en/guix.html#Creating-a-Channel
https://guix.gnu.org/manual/devel/en/guix.html#Channel-Authentication
https://doi.org/10.22152/programming-journal.org/2023/7/1

Tester son canal en local

```
$ cat mon-canal.scm
(list (channel
        (name 'guix)
        (url "https://git.savannah.gnu.org/git/guix.git")
        (branch "master")
        (commit
         "06493e738825598447f5b45d7100ca7eff8b669d"))
      (channel
        (name 'mine)
        (url "file:///path/to/more-packages")
        (branch "master")))
```

```
guix time-machine -C mon-canal.scm -- show julia-csv
```

Reproductibilité

Il faut fixer la révision (état) de tous les canaux

```
alice@laptop$ guix describe -f channels > my-state.scm
```

```
carole@cluster$ guix time-machine -C my-state.scm -- shell -m my-packages.scm
```

Les canaux sont des dépôts Git qui peuvent donc être archivés dans Software Heritage (SWH) et Guix va automatiquement y piocher si le serveur du canal a disparu.

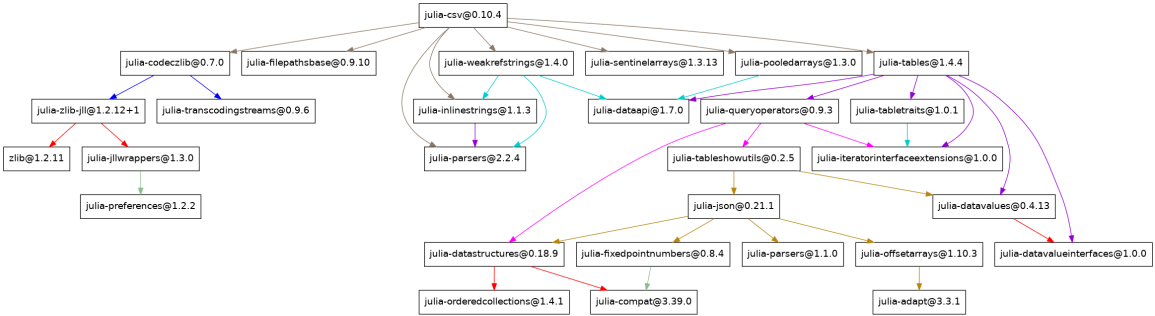
- Ajout automatique du code `source` p. ex. `guix lint -c archival julia-csv`
- Ajout manuel du dépôt Git correspondant au canal dans SWH

Pour (re)jouer

https://archive.softwareheritage.org/swh:1:rev:74811b87a1f7c4fba146ba011e96cdf3017c0b07 (lien)

```
$ cat replica.scm
(list (channel (name 'mon-canal)
               (url "https://serveur-canal-disparu.org")
               (branch "main")
               (commit "1648b9d39a46fb19678d0c269c38bf93d496bdd4"))
      (channel (name 'guix)
               (url "https://git.savannah.gnu.org/git/guix.git")
               (branch "master")
               (commit "06493e738825598447f5b45d7100ca7eff8b669d")))
$ guix time-machine -C replica.scm -- build --source julia-csv
```

- Contenu `https://serveur-canal-disparu.org` automatiquement récupéré depuis SWH
- Contenu `https://serveur-source-disparu.org` idem, depuis SWH


julia-csv@0.10.4
julia-codeczlib@0.7.0
julia-filepathsbase@0.9.10
julia-weakrefstrings@1.4.0
julia-sentinelarrays@1.3.13
julia-pooledarrays@1.3.0
julia-tables@1.4.4
julia-zlib-jll@1.2.12+1
julia-transcodingstreams@0.9.6
julia-inlinestrings@1.1.3
julia-dataapi@1.7.0
julia-queryoperators@0.9.3
julia-tabletraits@1.0.1
zlib@1.2.11
julia-jllwrappers@1.3.0
julia-parsers@2.2.4
julia-tableshowutils@0.2.5
julia-iteratorinterfaceextensions@1.0.0
julia-preferences@1.2.2
julia-json@0.21.1
julia-datavalues@0.4.13
julia-datastructures@0.18.9
julia-fixedpointnumbers@0.8.4
julia-parsers@1.1.0
julia-offsetarrays@1.10.3
julia-datavalueinterfaces@1.0.0
julia-orderedcollections@1.4.1
julia-compat@3.39.0
julia-adapt@3.3.1

Multi-canaux et graphe de dépendances

Le graphe de dépendances mélange les nœuds venant de plusieurs canaux différents

Les nœuds peuvent cacher d'autres graphes

(`julia-weakrefstrings@1.4.0` dépend de `julia-weakrefstrings@1.1.0` qui n'apparaît pas explicitement dans le graphe)

Pour reproduire : il faut décrire l'état (révision) de chaque canal

```
guix describe -f channels
```

Maintenance

Au quotidien, il faut s'assurer que les changements dans Guix ne cassent pas le canal

p. ex. reécente suppression de Python 2 ⇒ canal `guix-past` cassé

Si vos canaux tiers dépendent d'autres canaux, n'hésitez pas à collaborer

Empaqueter est un artisanat

En bref

- Une commande pour la reproductibilité

  ```
  guix time-machine -C channels.scm -- shell -m manifest.scm
  ```

  et deux fichiers :
  - `channels.scm` qui décrit les états (révisions) des canaux
  - `manifest.scm` qui décrit la liste des paquets
- Si c'est dans Guix, des efforts (collectif et infrastructure) pour la maintenance
- Si c'est dans un canal mutualisé, des efforts (coll. et infra.) pour la maintenance

  attention, l'état du canal peut être lié à l'état de Guix
- Si c'est dans votre canal, c'est du boulot à maintenir
- Ma façon de contribuer est :
  - via `--load-path`
  - via un canal mutalisé
  - `guix shell -D guix` puis `./pre-inst-env guix`

(voir patch#56426 (lien))

Rappel

Journée Guix, 16–18 septembre 2022, IRILL, Paris

`https://10years.guix.gnu.org/`

Vendredi 16 Sept. : « Reproducible deployment for reproducible research »

Des questions ?

guix-science@gnu.org

https://hpc.guix.info/events/2022/café-guix/