

GNU Guix Reference Card

for version 1.3.0-19751-ge640da33f3
<https://guix.gnu.org/>

Getting Started

To read the on-line documentation run `info guix` or visit <https://guix.gnu.org/manual/>. See <https://emacs-guix.gitlab.io/website/> for an Emacs interface to Guix.

Specifying Packages

Most commands take a “package specification” denoted `spec` in the sequel. Here are some examples:

```
emacs  
gcc-toolchain@7  
gcc-toolchain:debug  
Emacs package, latest version  
GCC toolchain, version 7.x  
latest GCC toolchain, debugging symbols
```

Managing Packages

```
guix search regexp ...  
guix show spec  
guix install spec...  
guix upgrade [regexp]  
guix remove name...  
guix package -m file  
guix package --export-manifest  
manifest  
roll back  
list installed packages  
list profile generations  
display search paths  
guix package -p profile ...  
use a different profile
```

Manifests

`guix package -m` and other commands take a “manifest” file listing packages of interest, along these lines:

```
(specifications->manifest  
, ("gcc-toolchain@7" "gcc-toolchain@7:debug"  
 "openmpi"))
```

One-Off Environments

```
guix shell spec...  
environment containing spec...  
guix shell -D python  
environment to develop Python itself  
guix shell python -C -- python3  
run Python in a container  
guix shell --check  
check if the shell clobbers environment variables  
guix shell -m file  
create an environment for the packages in manifest file
```

Updating Guix

```
guix describe  
guix describe -f channels  
guix pull  
guix pull -l  
guix pull --commit=commit  
guix pull --branch=branch  
guix pull -C file  
update the given channels  
describe current Guix  
produce a channel spec  
update Guix  
view history  
update to commit  
update to branch  
update the given channels
```

Channel Specifications

Channels specify Git repositories where `guix pull` looks for updates to Guix and external package repositories. By default `guix pull` reads `~/ .config/guix/channels.scm`; with `-C` it can take channel specifications from a user-supplied file that looks like this:

```
(cons (channel  
      (name 'guix-hpc)  
      (url "https://gitlab.inria.fr/guix-hpc/guix-hpc.git")  
      (branch "master"))  
      %default-channels)
```

Using a Different Version of Guix

The `guix time-machine` command provides access to other revisions of Guix, for example to install older versions of packages, or to reproduce a computation in an identical environment.

```
guix time-machine -C file -- commands...  
Run commands in a version of Guix specified by the given channels in file
```

Customizing Packages

```
guix command name --with-source=name=source  
build name with a different source URL  
guix command spec --with-input=spec1=spec2  
replace spec1 with spec2 in the dependency graph of spec  
guix command spec --with-graft=spec1=spec2  
graft spec2 in lieu of spec1 in spec  
guix command --with-git-url=spec=URL  
build spec from the given Git URL  
guix command spec --with-branch=package=branch  
build spec from the given Git branch of package  
guix command spec --with-commit=package=commit  
build spec from the given Git commit of package  
guix command spec --with-patch=package=file  
build spec after applying the given patch file to package  
guix command spec --with-latest=package  
build spec using the latest upstream release for package  
guix command spec --with-c-toolchain=package=toolchain  
build spec using toolchain for package  
guix command spec --without-tests=package  
build spec without running the tests for package
```

Developing Packages

```
guix edit spec  
guix build spec ...  
guix build --log-file spec  
guix build -K spec ...  
trees on failure  
guix build -S spec  
obtain the source of spec  
guix build --check spec  
rebuild a package  
guix build --target=triplet ...  
cross-compile to triplet-e-g,  
arm-linux-gnueabihf  
guix download URL  
download from URL and print  
its SHA256 hash  
guix hash file  
print the hash of file  
guix graph spec | dot -Tpdf ...  
view dependencies  
guix refresh spec  
update package definition  
guix import repo name  
import name from repo
```



Creating Application Bundles

```
guix pack spec ...
guix pack -f docker spec ...
guix pack -f squashfs spec ...
guix pack -f deb spec ...

guix pack -RR spec ...
guix pack -S /bin=bin spec ...

guix pack -m file
```

Managing Storage Space

```
guix gc
guix gc -C nG
guix gc -F nG
guix gc -d duration

guix size spec ...
guix gc -R /gnu/store/...
guix graph -t references spec ...
```

create a tarball
create a Docker image
create a Singularity image
create a Debian package
archive
create a relocatable tarball
make /bin a symlink to the packages' bin directory
bundle the packages from the manifest in file

collect all garbage
collect n GB of garbage
ensure n GB are available
delete generations older than duration—e.g., 1m for one month
view package size
list run-time dependencies
view run-time dependencies

Managing the Home Environment

```
guix home
```

guix home takes a configuration file that declares dotfiles, packages, and user services.

```
guix home search regexp
guix home configure file
guix home list-generations [pattern]
guix home delete-generations matching pattern
guix home delete-generations matching pattern
guix home roll-back
guix home build file
```

search for services matching regexp
reconfigure the home according to the configuration in file
list home generations matching pattern
delete generations matching pattern
delete generations matching pattern
roll back to the previous generation
build the home environment declared in file

Declaring an Operating System

guix system takes a configuration file that declares the complete configuration of an operating system, along these lines:

```
(use-modules (gnu))
(use-service-modules networking ssh)
(use-package-modules certs screen)

(operating-system
 (host-name "gnu")
 (timezone "Europe/Berlin")
 (locale "en_US.utf8")
 (keyboard-layout (keyboard-layout "us" "altgr-intl")))

(bootloader (bootloader-configuration
 (bootloader grub-efi-bootloader)
 (target (list "/boot/efi"))
 (keyboard-layout keyboard-layout)))

(file-systems (cons (file-system
 (device (file-system-label "my-root"))
 (mount-point "/")
 (type "ext4"))
 %base-file-systems))

(users (cons (user-account
 (name "charlie")
 (comment "Charlie Smith")
 (group "users")
 (supplementary-groups ('("wheel"
 "audio" "video"))))
 %base-user-accounts))

;; Globally installed packages.
(packages (append (list screen nss-certs)
 %base-packages))

;; System services: add sshd and DHCP to the base services.
(services (append (list (service dhcp-client-service-type)
 (service openssh-service-type)
 (openssh-configuration
 (port-number 2222))))
 %base-services)))
```

Building Operating Systems

```
guix system image file
guix system image --image-type=iso9660 file
guix system image --image-type=qcow2 file
guix system vm file
```

create a raw disk image for the OS declared in file
create an ISO CD/DVD image for the OS declared in file
produce a QCOW2 image of the OS in file
produce a script that runs the OS declared in file in a VM

Managing the Operating System

```
guix system search regexp
guix system reconfigure file
guix system list-generations [pattern]
guix system roll-back
guix system delete-generations pattern
guix system build file
guix system build file
```

search for services matching regexp
reconfigure the OS according to the configuration in file
list OS generations matching pattern—e.g., 1m for one month
roll back to the previous system generation
delete generations matching pattern
build the OS declared in file

Building and Running Containers

```
guix system container file
guix system docker-image file
```

produce a script that runs the OS declared in file in a container
build a Docker image of the OS declared in file

Inspecting an Operating System

```
guix system extension-graph file
guix system shepherd-graph file
```

show the graph of services extensions for the OS in file
show the dependency graph of Shepherd services for file

Copyright © 2018, 2019, 2020 Ludovic Courtiès <ludo@gnu.org>
Copyright © 2022 Ricardo Wurmus <rekado@lephylite>
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is available at <https://gnu.org/licenses/gfdl.html>.
The source of this document is available from <https://git.sv.gnu.org/git/guix/maintenance.git>.

