GNU Guix Reference Card

for version 0.16.0
https://gnu.org/software/guix/

Getting Started

To read the on-line documentation run `info guix` or visit *https://gnu.org/s/guix/manual/en/html_node*. 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`	Emacs package, latest version
`gcc-toolchain@7`	GCC toolchain, version 7.x
`gcc-toolchain:debug`	latest GCC toolchain, debugging symbols

Managing Packages

`guix package -s` *regexp ...*	search for packages
`guix package --show=`*spec*	show package info
`guix package -i` *spec...*	install packages
`guix package -u` [*regexp*]	upgrade packages
`guix package -r` *name...*	remove packages
`guix package -m` *file*	instantiate from manifest
`guix package --roll-back`	roll back
`guix package -l`	list profile generations
`guix package --search-paths`	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 environment --ad-hoc` *spec...*
: environment containing *spec...*

`guix environment python`
: environment to develop Python itself

`guix environment --ad-hoc python -C -- python3`
: run Python in a container

`guix environment -m` *file*
: create an environment for the packages in manifest *file*

Updating Guix

`guix describe`	describe current Guix
`guix pull`	update Guix
`guix pull -l`	view history
`guix pull --commit=`*commit*	update to *commit*
`guix pull --branch=`*branch*	update to *branch*
`guix pull -C` *file*	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)
```

Managing Storage Space

`guix gc`	collect all garbage
`guix gc -C` *n*`G`	collect *n* GB of garbage
`guix gc -F` *n*`G`	ensure *n* GB are available
`guix package -d` *duration*	delete generations older than *duration*—e.g., `1m` for one month
`guix size` *spec ...*	view package size
`guix gc -R /gnu/store/...`	list run-time dependencies
`guix graph -t references` *spec ...*	view run-time dependencies

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*

Developing Packages

`guix edit` *spec*	view the definition
`guix build` *spec ...*	build packages
`guix build --log-file` *spec*	view the build log
`guix build -K` *spec ...*	build packages, keep build 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 -l` *spec*	count dependent packages
`guix refresh` *spec*	update package definition
`guix import json` *file*	import JSON package metadata from *file*
`guix import` *repo name*	import *name* from *repo*
`guix lint` *spec ...*	"lint" packages

Creating Application Bundles

`guix pack` *spec ...*	create a tarball
`guix pack -f docker` *spec ...*	create a Docker image
`guix pack -f squashfs` *spec ...*	create a Singularity image
`guix pack --relocatable` *spec ...*	create a relocatable tarball
`guix pack -S /bin=bin` *spec ...*	make `/bin` a symlink to the packages' `bin` directory
`guix pack -m` *file*	bundle the packages from the manifest in *file*