We’re building the GNU system!

Ludovic Courtès
ludo@gnu.org

GNU Hackers Meeting
15 August 2014, München
Howdy!

(guile)
Howdy!

(\texttt{guile})

❄️ NixOS
Howdy!
the "GNU system",
aka. "a GNU distribution"
the “GNU system”, aka. “a GNU distribution”

- protect & enhance computing freedom
- improve integration of GNU software, consistency
- improve workflow among GNU hackers & users
Dependable.
per-user, transactional package installation etc.

alice@foo$ guix package --install=gcc
alice@foo$ guix gc --references ‘which gcc’
    /gnu/store/...-glibc-2.19
    /gnu/store/...-gcc-4.9.1
...
demo!

bob@foo$ guix package --install=gcc-4.7.3
bob@foo$ guix gc --references ‘which gcc’
    /gnu/store/...-glibc-2.17
    /gnu/store/...-gcc-4.7.3
...
alice@foo$ guix package --install=emacs
The following package will be installed:
  emacs-24.3 out /gnu/store/...-emacs-24.3

The following files will be downloaded:
  /gnu/store/...-emacs-24.3
  /gnu/store/...-libxpm-3.5.10
  /gnu/store/...-libxext-1.3.1
  /gnu/store/...-libxaw-1.0.11
alice@foo$ guix package --install=emacs
The following package will be installed:
  emacs-24.3 out /gnu/store/...-emacs-24.3

The following files will be downloaded:
  /gnu/store/...-libxext-1.3.1
  /gnu/store/...-libxaw-1.0.11
The following derivations will be built:
  /gnu/store/...-emacs-24.3.drv
  /gnu/store/...-libxpm-3.5.10.drv
(define foo (package ...))
(define foo (package ...))

user

(test)

guix build foo
/gnu/store/...-foo-1.0
```scheme
(define foo (package ...))
```

**test**

```
guix build foo
/gnu/store/...-foo-1.0
```

**user**

```
git push
git.sv.gnu.org
```
(define foo (package ...))

guix build foo
/gnu/store/...-foo-1.0

git push

git.sv.gnu.org

user

hydra.gnu.org

build farm

pull

hydra.gnu.org

pull

git.sv.gnu.org
(define foo (package ...))

test

guix build foo

/gitnu/store/...-foo-1.0

user

get binary

build farm

git.push

get binary

pull

git.sv.gnu.org

git.sv.gnu.org

pull
(define foo (package ...))

guix build foo
/gnu/store/...-foo-1.0

git push
git.sv.gnu.org

workflow

user

test

pull
(define foo (package ...))

user

no "maintainer uploads"

git push
git.sv.gnu.org

no single point of trust

guix build foo
/gnu/store/...-foo-1.0

test
does this binary correspond to that source?
towards deterministic builds

$ guix build guile

isolated build: chroot, separate name spaces, etc.
towards deterministic builds

$ guix build guile
/gnu/store/ h2g4sc09h4... -guile-2.0.9

hash of **all** the dependencies
towards deterministic builds

$ guix build guile
/gnu/store/ h2g4sc09h4...-guile-2.0.9

$ guix gc --references /gnu/store/...-guile-2.0.9
/gnu/store/4jl83jgzaac...-glibc-2.17
/gnu/store/iplay43cg58...-libunistring-0.9.3
/gnu/store/47p47v92cj9...-libffi-3.0.9
/gnu/store/drkwck2j965...-gmp-5.0.5
...
towards deterministic builds

$ guix build guile
/gnu/store/ h2g4sc09h4... -guile-2.0.9

$ guix gc --references /gnu/store/...-guile-2.0.9
/gnu/store/4jl83jgzaac...-glibc-2.17
/gnu/store/iplay43cg58...-libunistring-0.9.3
/gnu/store/47p47v92cj9...-libffi-3.0.9
/gnu/store/drkwck2j9...-gmp-5.0.5
...

(nearly) bit-identical for everyone
The Hacking Process

1. (R)econnaissance
2. (I)nfection
3. (C)ommand And Control
4. (E)xfiltration
do not trust a single binary provider
Deterministic Builds: Integrity through Decentralization

– Mike Perry
cool features for GNU maintainers
$ guix package -i gettext
looking for latest release of GNU gettext...
$ guix package -i gettext
note: using 0.18.3 but 0.19.2 is available upstream
$ guix build emacs
$ guix build emacs \n  --with-source=ftp://alpha.gnu..../emacs-24.3.92.tar.xz
$ guix build emacs \
--with-source=ftp://alpha.gnu..../emacs-24.3.92.tar.xz

pre-release testing in a pristine environment
Hello,

As some of you might know, I'll be taking care of the library for now (co-maintainers are welcome). Before applying big changes such as updating to the latest Unicode standard, I thought it would be a good idea to make a new release from the current git master.

So, I've generated a pre-release tarball:
http://alpha.gnu.org/gnu/libunistring/libunistring-0.9.4-rc1.tar.gz

I applied a couple of minor build fixes, but if you find anything else which should be included in the release, please report to the list.

Regards,

--
Daiki Ueno
Hello,

As some of you might know, I'll be taking care of the library for now (co-maintainers are welcome). Before applying bug changes such as updating to the latest Unicode edition, I thought it would be a good idea to make a new release from the current git master.

So, I've generated a pre-release tarball:
http://alpha.gnu.org/gnu/libunistring/libunistring-0.9.4-rc1.tar.gz

I applied a couple of minor build fixes, but if you find anything else which should be included in the release, please report to the list.

Regards,

--
Daiki Ueno
(use-modules (guix) (gnu))

(package (inherit
  (car (find-packages-by-name "libunistring")))
  (version "0.9.4rc1")
  (source (origin
    (method url-fetch
      (uri "http://.../libunistring-0.9.4-rc1.tar.gz")
      (sha256 (base32 "14pi90..."))))))
(use-modules (guix) (gnu))

(package (inherit
  (car (find-packages-by-name "libunistring")))
  (version "0.9.4rc1")
  (source (origin
    (method url-fetch)
    (uri "http://.../libunistring-0.9.4-rc1.tar.gz")
    (sha256 (base32 "14pi90...")))
))

$ guix build -e '(load "libunistring-rc.scm")'
The following derivations will be built:
  /gnu/store/...-libunistring-0.9.4rc1.drv
  /gnu/store/...-libunistring-0.9.4-rc1.tar.gz.drv
(use-modules (guix) (gnu))

(package (inherit
  (car (find-packages-by-name "libunistring")))
  (version "0.9.4rc1")
  (source (origin
    (method url-fetch)
    (uri "http://.../libunistring-0.9.4-rc1.tar.gz")
    (sha256 (base32 "14pi90..."))))

but does, say, Guile work with the RC?

$ guix build -e '(load "libunistring-rc.scm")'

The following derivations will be built:
  /gnu/store/...-libunistring-0.9.4rc1.drv
  /gnu/store/...-libunistring-0.9.4-rc1.tar.gz.drv
(use-modules (gnu packages guile) (guix)
    (srfi srfi-1))

(package (inherit guile-2.0)
  ;; Replace the stable libunistring with the RC
  ;; in Guile’s dependencies.
  (inputs `(("libunistring"
              ,(primitive-load "libunistring-rc.scm"))
              ,(alist-delete "libunistring"
                (package-inputs guile-2.0))))
(use-modules (gnu packages guile) (guix)
  (srfi srfi-1))

(package (inherit guile-2.0)
  ;; Replace the stable libunistring with the RC
  ;; in Guile’s dependencies.
  (inputs '(("libunistring"
        ,(primitive-load "libunistring-rc.scm"))
        ,(alist-delete "libunistring"
        (package-inputs guile-2.0))))

$ guix build -e '(load "guile-libunistring.scm")'
The following derivations will be built:
 /gnu/store/...-guile-2.0.11.drv
 /gnu/store/...-libunistring-0.9.4-rc1.tar.gz.drv
 /gnu/store/...-libunistring-0.9.4rc1.drv
(use-modules (gnu packages guile) (guix)
   (srfi srfi-1))

(package (inherit guile-2.0)
   ;; Replace the stable libunistring with the RC
   ;; in Guile’s dependencies.
   (inputs '(("libunistring"
         ,(primitive-load "libunistring-rc.scm"))
         @(alist-delete "libunistring"
         ,(package-inputs guile-2.0)))))

$ guix build -e '(load "guile-libunistring.scm")'
The following derivations will be built:
   /gnu/store/...-guile-2.0.11.drv
   /gnu/store/...-libunistring-0.9.4-rc1.tar.gz.drv
   /gnu/store/...-libunistring-0.9.4rc1.drv
even better

1. **iterate** over the packages with fold-packages
2. identify those **depending on libunistring**
3. compute **package variants** that use the RC
4. **build them**
even better
(left as an exercise to the audience)

1. **iterate** over the packages with fold-packages
2. identify those **depending on libunistring**
3. compute **package variants** that use the RC
4. **build them**
Hackable.
The truth is that Lisp is not the right language for any particular problem. Rather, Lisp encourages one to attack a new problem by implementing new languages tailored to that problem.

– Abelson & Sussman, 1987
(define hello
  (package
    (name "hello")
    (version "2.8")
    (source (origin
              (method url-fetch)
              (uri (string-append
                     "mirror://gnu/.../hello-" version
                     ".tar.gz"))
              (sha256 (base32 "0wqd...dz6"))))
    (build-system gnu-build-system)
    (synopsis "Hello, GNU world: An example GNU package")
    (description "Produce a friendly greeting.")
    (home-page "http://www.gnu.org/software/hello/")
    (license gpl3+)))
build processes
chroot, separate UIDs

build daemon

Guile
(guix packages)
(guix store)
build processes
chroot, separate UIDs

Guile
(guix packages)
(guix store)

build daemon

RPCs
build processes
chroot, separate UIDs

Guile, make, etc.

Guile, make, etc.

Guile, make, etc.

build daemon

Guile

(guix packages)

(guix store)

RPCs
(use-modules (guix) (gnu packages base))

,enter-store-monad

(interned-file "README")
=> "/gnu/store/rwmzi9jlj77a7bq5kyiy2abdqznr3f02-README"

(gexp->derivation "list-files"
  ~(symlink (string-append #$coreutils " /bin/ls")
     #$output)
=> #<derivation "/gnu/store/xyz...-list-files.drv" ...>

(package->derivation hello)
=> #<derivation "/gnu/store/xyz...-hello-2.8.drv" ...>

(built-derivations (list drv))
... daemon builds/downloads package on our behalf...
=> #t
copy fields from hello except for version and source

(package (inherit hello)
  (version "2.7")
  (source
   (origin
    (method url-fetch)
    (uri "mirror://gnu/hello/hello-2.7.tar.gz")
    (sha256 (base32 "7dqw3..."))))
(define (static-package p)
  ;; Return a statically-linked variant of P.
  (package (inherit p)
    (arguments
      '(
        #:configure-flags '("--disable-shared"
                            "LDFLAGS=-static")
        ,(package-arguments p))))))
and now, the operating system
(define my-config
 (operating-system
  (host-name "gnubox")
  (timezone "Europe/Paris")
  (locale "en_US.UTF-8")
  (bootloader (grub-configuration (device "/dev/sdX")))))

(file-systems (cons (file-system
      (mount-point "/") ...)
        %base-file-systems))
(users (list (user-account
      (name "ludo") (group "users")
      (comment "Hello, this is me!")
      (home-directory "/home/ludo")))
(packages (cons* iotop jnettop %base-packages)))
(define my-config
  (operating-system
    (host-name "gnubox")
    (timezone "Europe/Paris")
    (locale "en_US.UTF-8")
    (bootloader (grub-configuration (device="/dev/sdX")))
  )
  (initrd (cut base-initrd <>
    #:extra-modules '("usb-storage.ko")))
  (file-systems (cons (file-system
                  (mount-point "/") ...)
                  %base-file-systems))
  (users (list (user-account
                  (name "ludo") (group "users")
                  (comment "Hello, this is me!")
                  (home-directory "/home/ludo")))
  )
  (packages (cons* iotop jnettop %base-packages)))
$ guix system build os-config.scm
/gnu/store/lq5kbp84kl9q9ncbk578wa7x5x01rl3f-system

$ guix system vm os-config
/gnu/store/yan79p9k83dwk5yd6jgmhpjh0nxalfn6-run-vm.sh

# guix system init os-config /mnt
...

# guix system reconfigure os-config
...
# deco status dmd
Started: (term-tty1 term-tty2 nsqd syslog)
Stopped: ()

# deco stop nsqd
Service nsqd has been stopped
GNU dmd in a nutshell

- born in 2003, revived in 2013 :-)
- dependency-based service manager
GNU dmd in a nutshell

- born in **2003**, revived in 2013 :-)  
- dependency-based service manager  
- `<dmd>` is PID 1, `<deco>` is a client
GNU dmd in a nutshell

- born in 2003, revived in 2013 :-)  
- dependency-based service manager  
- dmd is PID 1, deco is a client  
- written in Guile Scheme  
- dynamic, extensible, etc.
extra coolness
step #1: cross-compiling to GNU/Hurd
step #1: cross-compiling to GNU/Hurd
packaged Mach, MiG, the Hurd, and...
... libc
step #1: cross-compiling to GNU/Hurd
packaged Mach, MiG, the Hurd, and...
... libc
but! GNU libc **fails to build** on GNU/Hurd
GNU/Hurd port—the true GNU!
WIP by Manolis Ragkousis

- step #1: cross-compiling to GNU/Hurd
- packaged Mach, MiG, the Hurd, and...
- ... libc
- but! GNU libc **fails to build** on GNU/Hurd
- (Debian’s version does build...)

GNU/Hurd port—the true GNU!
WIP by Manolis Ragkousis

- step #1: cross-compiling to GNU/Hurd
- packaged Mach, MiG, the Hurd, and...
- ... libc
- but! GNU libc fails to build on GNU/Hurd
- (Debian’s version does build...)

help the Hurd & libc regain harmony!
<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Synopsis</th>
<th>Home Page</th>
<th>License</th>
</tr>
</thead>
<tbody>
<tr>
<td>bigloo</td>
<td>4.1a</td>
<td>Bigloo, an efficient Scheme compiler</td>
<td><a href="http://www-sop.inria.fr/index/fp/Bigloo/">http://www-sop.inria.fr/index/fp/Bigloo/</a></td>
<td>GPL 2+</td>
</tr>
<tr>
<td>chicken</td>
<td>4.8.0.3</td>
<td>R5RS Scheme implementation that compiles native code via C</td>
<td><a href="http://www.call-cc.org/">http://www.call-cc.org/</a></td>
<td>Modified BSD</td>
</tr>
<tr>
<td>guile</td>
<td>1.8.8</td>
<td>Scheme implementation intended especially for extensions</td>
<td><a href="http://www.gnu.org/software/guile/">http://www.gnu.org/software/guile/</a></td>
<td>LGPL 2.0+</td>
</tr>
<tr>
<td>guile</td>
<td>2.0.9</td>
<td>Scheme implementation intended especially for</td>
<td><a href="http://www.gnu.org/software/guile/">http://www.gnu.org/software/guile/</a></td>
<td>LGPL 3+</td>
</tr>
</tbody>
</table>
### Packages

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Synopsis</th>
<th>Home Page</th>
<th>License</th>
</tr>
</thead>
<tbody>
<tr>
<td>bigloo</td>
<td>4.1a</td>
<td>Bigloo, an efficient Scheme compiler</td>
<td><a href="http://www-sop.inria.fr/INDEX/~p/Bigloo/">http://www-sop.inria.fr/INDEX/~p/Bigloo/</a></td>
<td>GPL 2+</td>
</tr>
<tr>
<td>chicken</td>
<td>4.8.0.3</td>
<td>R5RS Scheme implementation that compiles native code via C</td>
<td><a href="http://www.call-cc.org/">http://www.call-cc.org/</a></td>
<td>Modified BSD</td>
</tr>
<tr>
<td>guile</td>
<td>1.8.8</td>
<td>Scheme implementation intended especially for extensions</td>
<td><a href="http://www.gnu.org/software/guile/">http://www.gnu.org/software/guile/</a></td>
<td>LGPL 2.0+</td>
</tr>
<tr>
<td>guile</td>
<td>2.0.9</td>
<td>Scheme implementation intended especially for</td>
<td><a href="http://www.gnu.org/software/guile/">http://www.gnu.org/software/guile/</a></td>
<td>LGPL 3+</td>
</tr>
<tr>
<td>Name</td>
<td>Version</td>
<td>Outputs</td>
<td>Installed</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>---------</td>
<td>-------------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>bigloo</td>
<td>4.1a</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>chicken</td>
<td>4.8.0.3</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>geiser</td>
<td>0.6</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile</td>
<td>1.8.8</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile</td>
<td>2.0.11</td>
<td>out, debug</td>
<td>out</td>
<td></td>
</tr>
<tr>
<td>guile</td>
<td>2.0.11</td>
<td>out, debug</td>
<td>out</td>
<td></td>
</tr>
<tr>
<td>guile-cairo</td>
<td>1.4.1</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile-lib</td>
<td>0.2.2</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile-reader-for---</td>
<td>0.6</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile-reader-for---</td>
<td>0.6</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile-static-stri</td>
<td>2.0.11</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile-static-stri</td>
<td>2.0.11</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>guile-wm</td>
<td>1.0</td>
<td>out</td>
<td>out</td>
<td></td>
</tr>
<tr>
<td>guile-xcb</td>
<td>1.3</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mcron</td>
<td>1.0.8</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mit-scheme</td>
<td>9.1.1</td>
<td>out</td>
<td>out</td>
<td></td>
</tr>
<tr>
<td>racket</td>
<td>5.3.4</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>scheme48</td>
<td>1.9</td>
<td>out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>skribilo</td>
<td>0.9.2</td>
<td>out</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Name: guile
Version: 2.0.11
Synopsis: Scheme implementation intended especially for extensions
Outputs:
  out
Installed path:
  /gnu/store/h8d3hinaiwmxvbk9g9b67s0bw7v8d3r-guile-2.0.11

Dependencies:
  /gnu/store/bda98x7qxd90qz4kr4rd03gbqsd038x-libunistring-0.9.3
  /gnu/store/j2ainmhcqcb5hgcqanywdzci84dns5lf-libtool-2.4.2
  /gnu/store/lfgzbha6s711hynxdcj0xcd4l17582sy-libgc-7.4.0
  /gnu/store/84xqp874r7a65alwvl11g6znyk90812n-gmp-6.0.0a
debug

Location: gnu/packages/guile.scm:110:2

U:%*-  *Guix Package Info*  Top (1,0)  (Guix-Info) 16:44
Quit
Name: guile
Version: 2.0.11
Synopsis: Scheme implementation intended especially for extensions
Outputs: out
Installed path: `/gnu/store/h8d3hinaimvxbhk9g9b67s0bw7v8d3r-guile-2.0.11`
WIP by Alex Kost
https://github.com/alezost/guix.el
(merged soon)

Dependencies:
- `/gnu/store/bda98x7qxmd90qz4kr4rd03gbqsd038x-libunistring-0.9.3`
- `/gnu/store/j2ainmhcqcb5hgcqanywdzci84dns5lf-libtool-2.4.2`
- `/gnu/store/lfgzbha6s711hynxdcj0xcd4l17582sy-libgc-7.4.0`
- `/gnu/store/84xqp874r7a65alwvl11g6znyk90812n-gmp-6.0.0a`

debug Install

location: gnu/packages/guile.scm:110:2
U:%*- *Guix Package Info* Top (1,0) (Guix-Info) 16:44

Quit
.plan working as expected
plan working as expected

embed DSLs!
Lively!
timeline

- July 2012 — GHM, Düsseldorf
- Nov. 2012 — dubbed GNU
- Jan. 2013 — 0.1
- June 2013 — European Lisp Symposium
- 27 Sep. 2013 — 0.4, with VM image
- Dec. 2013 — 0.5, system config, mips64
- Apr. 2014 — 0.6, signed binaries, guix system
timeline

- July 2012 — GHM, Düsseldorf
- Nov. 2012 — dubbed GNU
- Jan. 2013 — 0.1
- June 2013 — European Lisp Symposium
- 27 Sep. 2013 — 0.4, with VM image
- Dec. 2013 — 0.5, system config, mips64
- Apr. 2014 — 0.6, signed binaries, guix system
- July 2014 — 0.7, **installable operating system!**
status
status

- full-featured package manager
- 900 packages, 3 platforms
- **standalone distro!** (alpha)
- binaries built & served at http://hydra.gnu.org
- tooling: auto-update, sync descriptions with GNU, etc.
- l10n: 7 languages!
In a Nutshell, GNU Guix...

... has had 3,403 commits made by 25 contributors representing 55,086 lines of code

... is mostly written in Scheme with a very well-commented source code

... has a young, but established codebase maintained by a large development team with stable Y-O-Y commits

... took an estimated 14 years of effort (COCOMO model) starting with its first commit in April, 2012 ending with its most recent commit 27 days ago
thanks for the code, reports, ideas, and translations!

Eric Bavier, Taylan Ulrich Bayirli/Kammer, John Darrington, Eelco Dolstra & the Nix crew, Andreas Enge, Alírio Eyng, Guy Grant, Raimon Grau, Nikita Karetnikov, Aljosha Papsch, Manolis Ragkousis, Cyril Roelandt, Alex Sassmannshausen, Cyrill Schenkel, Jason Self, Sree Harsha Totakura, David Thompson, Mark H. Weaver

Lluís Batlle i Rossell, Marek Benc, Daniel Clark, Alexandru Cojocaru, Aleix Conchillo Flaqué, Rafael Ferreira, Christian Grothoff, Jeffrin Jose, Kete, Alex Kost, Matthew Lien, Niels Möller, Yutaka Niibe, Adam Pribyl, Benno Schulenberg, Alen Skondro, Matthias Wachs, Zerwas

Pavel Fric, Mario Blättermann, Felipe Castro, Balázs Úr, Rafael Ferreira, Miroslav Nikolic, Tran Ngoc Quân
the road to 1.0

0. official FSDG compliance
1. **OS features**: dm-crypt, LVM, etc.
2. more service definitions
the road to 1.0

0. official FSDG compliance
1. OS features: dm-crypt, LVM, etc.
2. more service definitions
3. improved guix system reconfigure
the road to 1.0

0. official FSDG compliance
1. **OS features**: dm-crypt, LVM, etc.
2. **more service definitions**
3. improved **guix system reconfigure**
4. **authenticated** **guix pull** (signed commits?)
the road to 1.0

0. official FSDG compliance
1. **OS features**: dm-crypt, LVM, etc.
2. more service definitions
3. improved **guix system reconfigure**
4. authenticated **guix pull** (signed commits?)
5. **user interfaces**: Emacs, web, curses(?)
the road to 1.0

0. official FSDG compliance
1. OS features: dm-crypt, LVM, etc.
2. more service definitions
3. improved guix system reconfigure
4. authenticated guix pull (signed commits?)
5. user interfaces: Emacs, web, curses(?)
6. larger, more robust build farm...
7. less dog food...
8. more packages...
the road to 1.0

0. official FSDG compliance
1. OS features: dm-crypt, LVM, etc.
2. more service definitions
3. improved guix system reconfigure
4. authenticated guix pull (signed commits?)
5. user interfaces: Emacs, web, curses(?)
6. larger, more robust build farm...
7. less dog food...
8. more packages...
9. your idea here
install the distribution

use it, report bugs, add packages

help with the infrastructure + admin

share your ideas!
credits


► commit stats & project summary, http://www.ohloh.net/p/gnuguix