Howto update a Guix package

Updating a package in Guix is straightforward, but needs some steps to set up, so I decided to write this step by step guide.

Contents

1 Prerequisite

- You already run GNU Guix SD
- You are in a shell (bash), i.e. in `konsole` or `xfce4-terminal`.

2 Clone and prepare a local package repository

- Clone the repository with packages from savannah
- Setup a shell with a clean development environment (`-D guix`) — pure ensures you **only** have guix.
- Add the tools that may be needed during built (expected from the base system)

- Configure and build all the package definitions

```
git clone https://git.savannah.gnu.org/git/guix.git && \
cd guix && \
guix shell --pure -D guix guix -- \
  guix shell gperf texinfo gettext perl -- \
  bash -c 'make clean; autoreconf -ik;
    ./configure --localstatedir=/var; ./bootstrap;
    make -j8'
```

This example uses double-shell invocation. You can also combine these in a single call:

```
guix shell --pure -D guix guix gperf texinfo gettext perl -- \
  guix --version
```

3 Locate the package

3.1 Automatic lookup with guix edit

Guix edit opens the package it would install in your editor, so you need to run it in the pre-inst-env to get the local package:

```
./pre-inst-env guix edit {{package name}}
```

3.2 Manual lookup

```
grep -R "define-public {{package name}}" gnu/packages
```

That gives you a file like gnuzilla.scm and a line number. You can also use the Emacs projectile package to search and then jump to the right package with a simple click.

4 Change the version and adjust the hash

4.1 Automatic update with guix refresh

For many packages you can simply use guix refresh:

```
guix refresh -u {{package name}}
```

4.2 Manual update

If that does not work, just look for `(version {{whatever}})`. Then **change that version**. You'll recognize special cases — I'll ignore those here. They are similar, but you need to Scheme a bit more.

Now build the package. Let's assume, it is wine64:

```
guix shell -D guix -- \
  guix shell gperf texinfo gettext perl -- \
  ./pre-inst-env guix build wine64
```

You will get an error that includes something like

```
expected hash: 1zgkqflqgl2y3a90f2nvcc1vhzr9ni0lps276553j8zgbqvnd0hn
actual hash:   1ni2sk1gj99fsnc1dbm7060b1ilydbrfyy6cmisn9jjbg1hsx3zd
```

Look for a pattern in the package like

```
(sha256
  (base32
    "1zwpgis7py1bf8p88pz3mpai6a02qrdb8ww2fa9kxxdl9b8r2k81"))
```

Just put the actual hash in there (if you're sure that it's correct). Now build again:

```
guix shell -D guix -- \
  guix shell gperf texinfo gettext perl -- \
  ./pre-inst-env guix build wine64
```

For most packages this just works.

5 Do a smoke test

```
guix shell -D guix -- guix shell gperf texinfo gettext perl -- \
  ./pre-inst-env guix shell wine64 -- \
  wine64
```

6 Commit the change and send the patch

If that works, look into `git log -v` to see how people write their commit messages. Adapt one of those for your update to commit, then create the patch.

```
git commit -a
# (then type in the message and save)
git format-patch -1
```

Now send the created patch file via email to `guix-patches@gnu.org`