Practical session 2: Packaging GNUjump

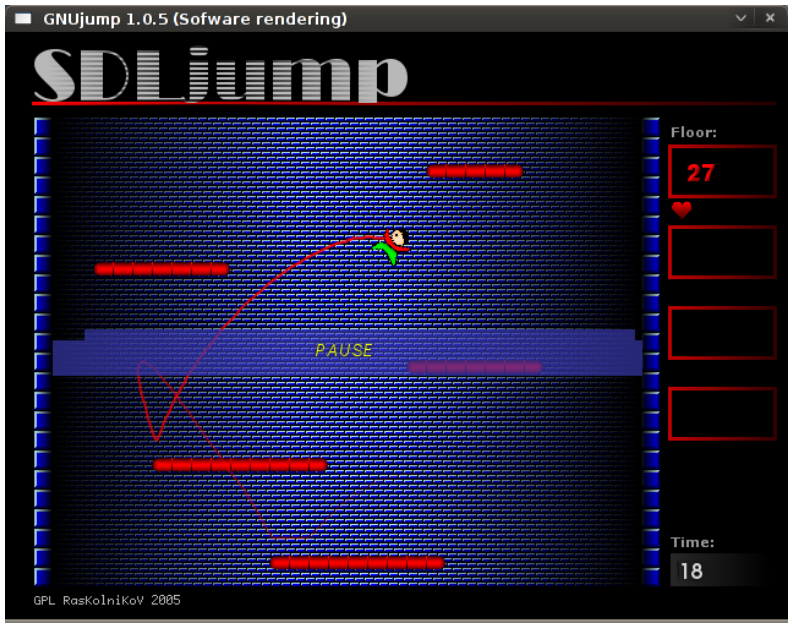
Lucas Nussbaum
lucas@debian.org

Practical session 2: packaging GNUjump

1. Download GNUjump 1.0.6 from
   `http://ftp.gnu.org/gnu/gnujump/1.0.6/gnujump-1.0.6.tar.gz`

2. Create a Debian package for it
   - Install build-dependencies so that you can build the package
   - Get a basic working package
   - Finish filling `debian/control` and other files

3. Enjoy



Step by step...

- `wget http://ftp.gnu.org/gnu/gnujump/1.0.6/gnujump-1.0.6.tar.gz`
- `mv gnujump-1.0.6.tar.gz gnujump_1.0.6.orig.tar.gz`
- `tar xf gnujump_1.0.6.orig.tar.gz`
- `cd gnujump-1.0.6/`
- `dh_make`
  - Type of package: single binary (for now)

```
gnujump-1.0.6$ ls debian/
changelog              gnujump.default.ex   preinst.ex
compat                 gnujump.doc-base.EX  prerm.ex
control                init.d.ex            README.Debian
copyright              manpage.1.ex         README.source
docs                   manpage.sgml.ex      rules
emacsen-install.ex     manpage.xml.ex       source
emacsen-remove.ex      menu.ex              watch.ex
emacsen-startup.ex     postinst.ex
gnujump.cron.d.ex      postrm.ex
```

Step by step . . . (2)

- Look at `debian/changelog`, `debian/rules`, `debian/control` (auto-filled by **dh_make**)
- In `debian/control`:
  `Build-Depends: debhelper (>= 7.0.50 ), autotools-dev`
  Lists the *build-dependencies* = packages needed to build the package
- Try to build the package as-is (thanks to **dh** magic)
  - And add build-dependencies, until it builds
  - Hint: use `apt-cache search` and `apt-file` to find the packages
  - Example:

```
checking for sdl-config... no
checking for SDL - version >= 1.2.0... no
[...]
configure: error: *** SDL version 1.2.0 not found!
```

    → Add **libsdl1.2-dev** to Build-Depends and install it.

  - Better: use **pbuilder** to build in a clean environment

Step by step . . . (3)

- After installing `libsdl1.2-dev`, `libsdl-image1.2-dev`, `libsdl-mixer1.2-dev`, the package builds fine.
- Use `debc` to list the content of the generated package.
- Use `debi` to install it and test it.
- Fill in `debian/control` using `http://www.debian.org/doc/debian-policy/ch-controlfields.html`
- Test the package with `lintian`
- Remove the files that you don't need in `debian/`
- Compare your package with the one already packaged in Debian:
  - It splits the data files to a second package, that is the same across all architectures ($\rightarrow$ saves space in the Debian archive)
  - It installs a .desktop file (for the GNOME/KDE menus) and also integrates into the Debian menu
  - It fixes a few minor problems using patches