debplate

A template system for Debian packages

Ben Hutchings <benh@debian.org>

MiniDebConf Online — May 2020

Why use a template system?

- Source package builds multiple binary packages that should be consistent:
  - Optional plugins split into multiple separate packages
  - Server software linked with alternate TLS libraries
  - Compiler built for different targets
  - Multiple build configurations needed for different systems
- ABI changes require changing package names in multiple places
- Don't Repeat Yourself (DRY) principle

This must have been done before...

- There *are* source packages with their own template systems:
  - The kernel team has one; it's simple but needs a *lot* of custom logic besides templates and variables
  - gcc & gcc-defaults have one; it's even more limited and I found it hard to work with
  - ...probably others?
- I didn't find a general tool for this, so started writing one
- **debplate** is intended to be usable by *any* source package that needs a template system, so that those packages will be less unusual and easier to work on

What can **debplate** generate?

- `debian/control` and `debian/tests/control`
- A makefile to be *included* in `debian/rules`
- Any per-package configuration files, like:
  - *package-name*`.dirs`
  - *package-name*`.metainfo.xml`
  - *package-name*`.postinst`
  - etc.

Using **debplate** — overview of files

debian/rules:

```
#!/usr/bin/make -f

include /usr/share/debplate/dh.mk

%:
	dh $@
```

debian/debplate-config.yml:

```
version: 0
groups:
  foo:
    templates: binary
    variables:
```

debian/templates/:

```
binary.control
binary.copyright
binary.dirs
binary.install
binary.links
binary.metainfo.xml
binary.postinst
binary.preinst
binary.rules
binary.templates
meta.control
source.control
```

Using **debplate** — configuration file

`debian/debplate-config.yml`:

```
version: 0
groups:
  foo:
    templates: binary
    variables:
      depends: foo
      recommends: foo-full
      description_short: foo framework
      description_long: >-
        This version is configured to work with the equally awesome
        foo framework.
  bar:
    templates: binary
```

Using **debplate** — template file

`debian/templates/binary.control`:

```
Package: simple-{{group}}
Build-Profiles: <!pkg.simple.no{{group}}>
Architecture: any
Depends: {{ depends }}, ${misc:Depends}, ${shlibs:Depends}
Recommends: {{ recommends | default() }}
Suggests: {{ suggests | default() }}
Description: simple package for {{description_short}}
 A simple package that's really useful.
 .
 {{ description_long | wordwrap(width=72) }}
{% if description_long_pre is defined %} .
 {{ description_long_pre }}
{% endif %}
```

More features

- Groups can have child groups, to any depth
  - Variables are inherited, but can be overridden or extended by child groups
  - Templates can be specified to be used in *all* descendant groups at a certain level
- Groups can have architecture restrictions
- Groups can be disabled (for derivatives or local builds)
- Binary package .control templates can add build-deps, or make tests depend on them
- Lots of extra Jinja filters taken from Ansible

So it's ready to use?

- Not quite yet:
  - It can't do everything the kernel team's templates and scripts can
  - Totally untested with other source packages
  - Configuration schema is not stable
- Hoping to make a usable 1.0 release this year, with stable versioned schema for then on
- Give it a try if you have a package that could benefit:
  - `git clone https://salsa.debian.org/benh/debplate.git`
  - Open issues
  - Open merge requests

Credits & License

- Content by Ben Hutchings
  https://www.decadent.org.uk/ben/talks/
  License: GPL-2+
- Original OpenOffice.org template by Raphaël Hertzog
  http://raphaelhertzog.com/go/ooo-template
  License: GPL-2+
- Background based on “Serenity” theme by Edward Padilla
  https://wiki.debian.org/DebianArt/Themes/serenity
  License: GPL-2