Lintian

Internals and Thoughts about further Development

Frank Lichtenheld <djpig@debian.org>

Debian Project

09/11/2005 / Debian QA Meeting 2005

Outline

Outline

History of Lintian development

- Development started 1998 by Christian Schwarz and Richard Braakman
- General design unchanged since then
- A lot of people have contributed since then and the maintainership changed several times
  - 1999 to Darren Benham
  - 2000 to Sean 'Shaleh' Perry
  - 2002 to Josip Rodin
  - 2004 to *Debian Lintian Maintainers* (Jeroen van Wolffelaar, Frank Lichtenheld, Marc 'HE' Brockschmidt)

  (People that have contributed significantly without ever assuming maintainership include Joey Hess, Colin Watson, and Denis Barbier).
- Currently maintained in a private subversion repository on Jeroen's server, but we plan to move that to Alioth.
- If you want to have an account, you can request one at `http://svn.wolffelaar.nl/`

Outline

Overview over Lintian's workflow

- `lintian` – Frontend
  1. `unpack/` – Scripts to unpack packages to certain degrees
  2. `collection/` – Scripts to collect files and meta data from the unpacked packages
  3. `checks/` – Modules to search for errors with the collected information
  4. `lintian-info` – Script to postprocess the output of the checks

Unpacking

binary packages

Level 1	unpacks the `control.tar` from the package and generates an index of the files in the `data.tar`
Level 2	unpacks the `data.tar`

source packages

Level 1	extracts information from the `.dsc`
Level 2	unpacks the source package

Data Collection

Scripts (mostly Perl, one Shell). Each script has a corresponding description file (*scriptname*.desc):

collection/objdump-info.desc

```
Collector-Script: objdump-info
Author: Christian Schwarz <schwarz@debian.org>
Info: This script runs ‘objdump’ over all binaries
 and object files of a binary package.
Type: binary, udeb
Unpack-Level: 2
Output: objdump-info
Order: 2
Needs-Info: file-info
```

Checking

Also formerly scripts but we converted them to Perl modules that are directly included instead if run with `exec`. They also have a description file:

checks/etcfiles.desc

```
Check-Script: etcfiles
Author: Sean 'Shaleh' Perry <shaleh@debian.org>
Abbrev: etc
Standards-Version: 3.5.0.0
Type: binary
Needs-Info: etcfiles
Unpack-Level: 1

Tag: file-in-etc-not-marked-as-conffile
Type: error
Ref: policy 10.7
Info: Files in <tt>/etc</tt> must be marked conffiles if they
 are included in a package.  Otherwise they should be created
 by maintainer scripts.
```

A simple checking script

```perl
# [copyright stuff]
package Lintian::control_file;
use Util; use Tags;

sub run {
my $pkg = shift;
my $type = shift;
# Check that each field is only used once:
my $seen_fields = {};
open (CONTROL, "debfiles/control") or fail "Couldn't␣read␣control:␣$!";
while (<CONTROL>) {
        #Reset seen_fields if we enter a new section:
        $seen_fields = {} if /^$/;
        if (/^(\S+):/) {
                my $field = lc ($1);
                if ($seen_fields->{$field}) {
                        tag "debian-control-with-duplicate-fields",
                            "$field:␣$$seen_fields{$field},␣$.";
                }
                $seen_fields->{$field} = $.;
        }
}
1;
```

`lintian-info` (Postprocessing)

- Reads the `Info:` fields from the check description files
- strips out HTML tags and expands some special entities
- wraps the text to the screen width

Outline

Code quality, Refactoring

- A lot of code could be rewritten in a much cleaner and shorter way due to the new Perl features of the last seven years (e.g. the whole `fork-and-exec` stuff)
- By converting more of the scripts to Perl modules we can make the whole process more flexible (and probably faster by saving quite some forks and maybe even file reads).
- By moving more code from the frontend to separate Perl modules we can make the frontend code more maintainable.

Outline

1. Internals
    - History
    - Workflow
2. Further Development
    - Code quality, Refactoring
    - Backend, Data Storage and Managment
    - User Interaction and Reporting
    - Integration with other tools
3. Helping with Lintian

Backend, Data Storage and Managment

- The current lab code is *teh suck*.
- The lab should cache more things, like the result of the last run (lintian.debian.org uses a single log file for that right now which is a really ugly hack) and probably even some of the results of the collection scripts
- Lintian tries to use information from the source package while processing the binary package but the implementation for that is a hack too

Outline

User Interaction and Reporting

- The current system is not very accurate and mixes information about severity and quality of the check (e.g. we downgraded some errors to warnings because they produced too many false positives)
- The plan is to introduce two dimensions "Severity" and "Significance" where each of them has 3-4 possible values
- There will be one human-readable output format that will just add a qualifier to the severity value (e.g. `E?` or `E!` instead of just E)) and one machine-readable output with numerical values (don't know which format to use yet, tough, colon-separated? XML? RFC822?)

Outline

Integration with other tools

- We probably should have better unpacking scripts that can handle things like a unpacked source package or even an installed package. The latter would also open the possibility for a set of new checks that can't be done on the package file itself
- Currently many maintainers run a lot of tools to test their packages (lintian, linda, debdiff, piuparts, etc.). It would be certainly useful to have some kind of integration between these tools, even if it would be only a frontend to run them all on a given package in a senseful way. But also things like running lintian on the installed package within piuparts sound interesting

How to help making lintian better

- Provide complete patches (let’s fix a bug to demonstrate what that means, #324944 looks simple enough)
- If you can’t provide patches, provide an algorithm (and invest some thoughts how good it avoids false positives)
- Subscribe to `debian-lint-maint@lists.debian.org` and contribute to discussions and development

Bug #324944

```
From: Otavio Salvador <otavio@debian.org>
To: Debian Bug Tracking System <submit@bugs.debian.org>
Subject: lintian: False positive to extra license file
Date: Wed, 24 Aug 2005 21:26:47 -0300

Package: lintian
Version: 1.23.11
Severity: minor

  W: gnome-icon-theme-gartoon binary: extra-license-file \
     usr/share/[...]/gnome-mime-text-x-copying.svg

[...]
```

Let’s search the code

```
lintian/trunk$ grep extra-license-file checks/*
checks/files:   tag "extra-license-file", "$file";
checks/files.desc:Tag: extra-license-file
lintian/trunk$ grep extra-license-file testset/tags.*
lintian/trunk$
```

Found it

```
# ———————————— license files
if ($file =~ m,(copying|licen[cs]e)(\.[^/]+)?$,i
    # ignore some common extensions; there was at least one file
    # named "license.el".  These are probably license-displaying
    # code, not license files.
    # Another exception is made for .html and .php because preserving
    # working links is more important than saving some bytes, and
    # because a package had a HTML form for licenses called like that.
    # Another exception is made for various picture formats since
    # those are likely to just be simply pictures.
    and not $file =~ m/\.(el|c|h|py|cc|pl|pm|html|php|xpm|png)$/
    and not defined $link) {
    tag "extra-license-file", "$file";
}
```

Changed it

```
# ------------------------ license files
if ($file =~ m,(copying|licen[cs]e)(\.[^/]+)?$,i
    # ignore some common extensions; there was at least one file
    # named "license.el".  These are probably license-displaying
    # code, not license files.
    # Another exception is made for .html and .php because preserving
    # working links is more important than saving some bytes, and
    # because a package had a HTML form for licenses called like that.
    # Another exception is made for various picture formats since
    # those are likely to just be simply pictures.
    and not $file =~ m/\.(el|c|h|py|cc|pl|pm|html|php|xpm|png|jpe?g|gif|sv
    and not defined $link) {
    tag "extra-license-file", "$file";
}
```

Changelog entry:

```
  * debian/files:
    + [FL] Add some formats to the exception list for
      extra-license-file (jpe?g, gif and svg)
      (Closes: #324944)
```

Add a testset

In `testset/filenames/debian/rules` we add:

```
install -d debian/tmp/usr/share/pixmaps
install -d debian/tmp/usr/share/pixmaps/foo
touch debian/tmp/usr/share/pixmaps/license.jpeg
touch debian/tmp/usr/share/pixmaps/licence.jpg
touch debian/tmp/usr/share/pixmaps/copying.xpm
touch debian/tmp/usr/share/pixmaps/foo/COPYING.svg
touch debian/tmp/usr/share/pixmaps/foo/copying.png
touch debian/tmp/usr/share/pixmaps/license.txt
touch debian/tmp/usr/share/pixmaps/license.foo
touch debian/tmp/usr/share/pixmaps/COPYING
```

Add a testset 2

Let's run the testset:

```
lintian/trunk$ ./debian/rules runtests onlyrun=filenames
.... running tests ....
[ -d debian/tests ] || mkdir debian/tests
LINTIAN_ROOT="" /usr/bin/perl testset/runtests -k testset \
  debian/tests filenames
Checking for missing info tags ... done.
Running static lab test ... create ...  renew ...  remove ... rmdir ...dor
Running test on filenames 12-0.1: copying... building... testing... FAILED
--- testset/tags.filenames      2005-09-09 07:12:43.000000000 +0200
+++ debian/tests/tags.filenames 2005-09-09 08:03:58.073754528 +0200
@@ -38,6 +38,9 @@
 W: filenames: binary-without-manpage testxbin
 W: filenames: binary-without-manpage testxbin2
 W: filenames: cvsignore-file-in-package files/.cvsignore
+W: filenames: extra-license-file usr/share/pixmaps/COPYING
+W: filenames: extra-license-file usr/share/pixmaps/license.foo
+W: filenames: extra-license-file usr/share/pixmaps/license.txt
 W: filenames: file-in-unusual-dir files/ .tif
 W: filenames: file-in-unusual-dir files/".tif
 W: filenames: file-in-unusual-dir files/'\\
Checking whether all tags are tested and tags have description ...
done.
touch runtests
```

Add a testset 3

In `testset/tags.filenames` we add the expected results

```
[...]
W: filenames: binary-without-manpage testxbin
W: filenames: binary-without-manpage testxbin2
W: filenames: cvsignore-file-in-package files/.cvsignore
W: filenames: extra-license-file usr/share/pixmaps/COPYING
W: filenames: extra-license-file usr/share/pixmaps/license.foo
W: filenames: extra-license-file usr/share/pixmaps/license.txt
W: filenames: file-in-unusual-dir files/ .tif
W: filenames: file-in-unusual-dir files/".tif
W: filenames: file-in-unusual-dir files/’\\
[...]
```

Add a testset 4

Now it works...

```
lintian/trunk$ ./debian/rules runtests onlyrun=filenames
.... running tests ....
[ -d debian/tests ] || mkdir debian/tests
LINTIAN_ROOT="" /usr/bin/perl testset/runtests -k testset \
  debian/tests filenames
Checking for missing info tags ... done.
Running static lab test ... create ...  renew ...  remove ... rmdir ...dor
Running test on filenames 12-0.1: copying... building... testing... done.
Checking whether all tags are tested and tags have description ...
done.
touch runtests
```