Titorial de Empaquetado en Debian

Lucas Nussbaum
packaging-tutorial@packages.debian.org

versión 0.30 – 2024-03-16

Sobre este titorial

- Obxectivo: **contarche o que debes saber sobre os paquetes en Debian**
  - Modificar paquetes existentes
  - Crear os teus paquetes propios
  - Interactuar coa comunidade de Debian
  - Converterse nun usuario avanzado
- Trata os temas máis importantes, pero non está completo
  - Terás que ler máis documentación
- A maioría dos contidos tamén se aplican ás distribucións derivadas de Debian
  - Eso inclúe Ubuntu

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Debian

- **Distribución GNU/Linux**
- A primeira distribución importante «centrada nas ideas de GNU»
- **Non comercial**, construído en común por milleiros de voluntarios
- 3 características principais:
  - **Calidade** – comunidade con experiencia técnica
    Sairá cando teña que saír
  - **LIbertade** – desenvolvedores e usuarios obrigados polo *Contrato Social*
    Promovendo a cultura do Software Libre dende 1993
  - **Independencia** – non hai unha compañía única que controle Debian
    E un sistema de toma de decisións aberto (*laborocracia* + *democracia*)
- **Afeccionados** no bo sentido: traballar por amor ao arte

Paquetes de Debian

- Ficheiros **.deb** (paquetes binarios)
- Unha forma potente e cómoda de distribuír os programas aos usuarios
- Un dos dous formatos de paquetes máis usados, xunto con RPM
- Universal:
  - Hai 30 000 paquetes binarios en Debian
    → a maioría dos programas libres están empaquetados en Debian!
  - Dispoñibles en 12 arquitecturas, incluíndo 2 non Linux (Hurd, kFreeBSD)
  - Tamén os usan 120 distribucións derivadas de Debian

O formato de empaquetado Deb

- ficheiro `.deb`: un arquivo `ar`

```
$ ar tv wget_1.12-2.1_i386.deb
rw-r--r-- 0/0      4 Sep  5 15:43 2010 debian-binary
rw-r--r-- 0/0   2403 Sep  5 15:43 2010 control.tar.gz
rw-r--r-- 0/0 751613 Sep  5 15:43 2010 data.tar.gz
```

  - `debian-binary`: versión do formato do ficheiro deb, `"2.0\n"`
  - `control.tar.gz`: metainformación sobre o paquete
    `control, md5sums, (pre|post)(rm|inst), triggers, shlibs,` ...
  - `data.tar.gz`: ficheiros coa información do paquete

- Pódense crear os ficheiros `.deb` manualmente
  `http://tldp.org/HOWTO/html_single/Debian-Binary-Package-Building-HOWTO/`

- Pero a maioría da xente non o fai así

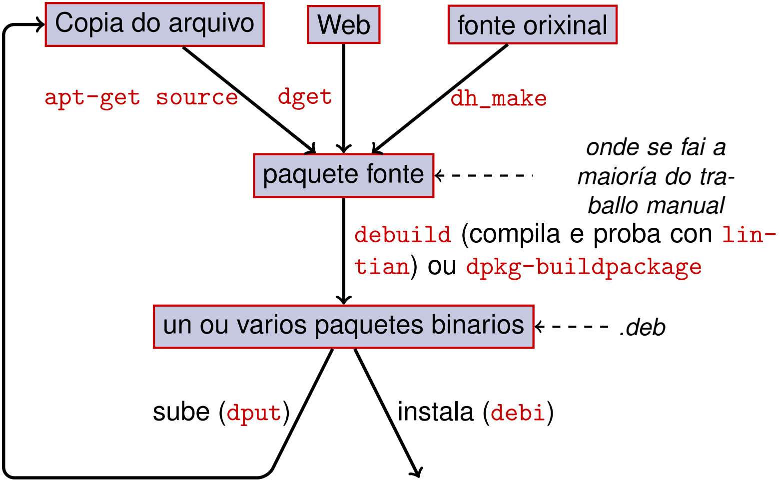
Neste titorial: crear paquetes Debian, á maneira Debian

Ferramentas necesarias

- Un sistema Debian (ou Ubuntu) con acceso de superusuario
- Algúns paquetes:
  - **build-essential**: depende de paquetes que se supoñen instalados na máquina do desenvolvedor (non se necesitan indicar no campo de control `Build-Depends:` do teu paquete)
    - inclúe unha dependencia a **dpkg-dev**, que contén ferramentas básicas para crear paquetes específicas para Debian
  - **devscripts**: conten moitos programas útiles para os mantedores de Debian

Máis tarde mencionaranse outras ferramentas, coma **debhelper**, **cdbs**, **quilt**, **pbuilder**, **sbuild**, **lintian**, **svn-buildpackage**, **git-buildpackage**, ...
Instáleas cando as precise.

Fluxo de traballo xeral



Exemplo: recompilando dash

1. Instale os paquetes necesarios para compilar dash, e devscripts
   `sudo apt-get build-dep dash`
   (require as liñas `deb-src` en `/etc/apt/sources.list`)
   `sudo apt-get install --no-install-recommends devscripts fakeroot`
2. Cree un cartafol de traballo, e entre nel:
   `mkdir /tmp/debian-tutorial ; cd /tmp/debian-tutorial`
3. Obteña o paquete fonte de `dash`
   `apt-get source dash`
   (Isto require que teña as liñas `deb-src` no seu `/etc/apt/sources.list`)
4. Compila o paquete
   `cd dash-*`
   `debuild -us -uc` (`-us -uc` desactiva a firma do paquete con GPG)
5. Comprobe se funcionou
   - Hai algúns ficheiros `.deb` novos no cartafol superior
6. Fíxese no directorio `debian/`
   - É aí onde se realiza o traballo de empaquetado.

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Paquete fonte

- Un paquete fonte pode xerar varios paquetes binarios
  p.e. a fonte `libtar` xera os paquetes binarios `libtar0` e `libtar-dev`
- Dous tipos de paquetes: (se ten dúbidas, escolla non nativo)
  - Paquetes nativos: normalmente programas específicos para Debian (*dpkg*, *apt*)
  - Paquetes non nativos: programas desenvoltos fora de Debian
- Ficheiro principal: `.dsc` (meta-información)
- Outros ficheiros dependendo da versión do formato fonte
  - 1.0 ou 3.0 (nativo): `paquete_version.tar.gz`
  - 1.0 (non nativo):
    - `pkg_ver.orig.tar.gz`: fonte orixinal
    - `pkg_debver.diff.gz`: parche para engadir cambios específicos para Debian
  - 3.0 (quilt):
    - `pkg_ver.orig.tar.gz`: fonte orixinal
    - `pkg_debver.debian.tar.gz`: arquivo tar cos cambios de Debian

(ver `dpkg-source(1)` para máis información)

Exemplo de paquete fonte (wget_1.12-2.1.dsc)

```
Format: 3.0 (quilt)
Source: wget
Binary: wget
Architecture: any
Version: 1.12-2.1
Maintainer: Noel Kothe <noel@debian.org>
Homepage: http://www.gnu.org/software/wget/
Standards-Version: 3.8.4
Build-Depends: debhelper (>> 5.0.0), gettext, texinfo,
 libssl-dev (>= 0.9.8), dpatch, info2man
Checksums-Sha1:
 50d4ed2441e67[..]1ee0e94248 2464747 wget_1.12.orig.tar.gz
 d4c1c8bbe431d[..]dd7cef3611 48308 wget_1.12-2.1.debian.tar.gz
Checksums-Sha256:
 7578ed0974e12[..]dcba65b572 2464747 wget_1.12.orig.tar.gz
 1e9b0c4c00eae[..]89c402ad78 48308 wget_1.12-2.1.debian.tar.gz
Files:
 141461b9c04e4[..]9d1f2abf83 2464747 wget_1.12.orig.tar.gz
 e93123c934e3c[..]2f380278c2 48308 wget_1.12-2.1.debian.tar.gz
```

Obtendo un paquete fonte existente

- no arquivo Debian:
  - `apt-get source` *package*
  - `apt-get source` *package=version*
  - `apt-get source` *package/release*

  (Necesita as liñas `deb-src` no `sources.list`)

- Do Internet:
  - `dget` *url-to.dsc*
  - `dget http://snapshot.debian.org/archive/debian-archive/20090802T004153Z/debian/dists/bo/main/source/web/wget_1.4.4-6.dsc`
    (`snapshot.d.o` encárgase de todos os paquetes Debian dende o 2006)

- Dende o sistema (declarado) de control de versións:
  - `debcheckout` *package*

- Cando se descargue, descomprímao con `dpkg-source -x` *file.dsc*

Crear un paquete fonte básico

- Descarga a fonte orixinal
(*fonte orixinal* = a que provén os desenvolvedores orixinais do programa)
- Renoméeo a `<fonte_paquete>_<orixinal_versión>.orig.tar.gz`
(por exemplo: `simgrid_3.6.orig.tar.gz`)
- Descomprímeo
- Renomea o cartafol a `<fonte_paquete>-<orixinal_versión>`
(por exemplo: `simgrid-3.6`)
- `cd <fonte_paquete>-<orixinal_versión> && dh_make`
(do paquete **dh-make**)
- Hai algunhas alternativas a `dh_make` para conxuntos específicos de paquetes: **dh-make-perl**, **dh-make-php**, . . .
- Creado cartafol `debian/`, con moitos ficheiros nel

Ficheiros en debian/

Todo o traballo de empaquetar débese facer editando os ficheiros en `debian/`

- Ficheiros principais:
  - **control** – metainformación sobre o paquete (dependencias, etc)
  - **rules** – especifica como compilar o paquete
  - **copyright** – información do paquete sobre dereitos de autor
  - **changelog** – historial do paquete de Debian
- Outros ficheiros:
  - compat
  - watch
  - obxectivos dh_install*
    *.dirs, *.docs, *.manpages, ...
  - programiños dos mantedores
    *.postinst, *.prerm, ...
  - source/format
  - patches/ – se os necesitas para modificar a fonte orixinal
- Algúns ficheiros usan un formato baseado no RFC 882 (cabeceiras do correo-e)

debian/changelog

- Rexistra os cambios no empaquetado de Debian
- Garda a versión actual do paquete

  1.2.1.1-5

  1.2.1.1 = versión orixinal; 5 = revisión Debian

- Editar manualmente ou con `dch`
  - Crear unha entrada no rexistro de cambios para unha nova versión: `dch -i`
- Formato especial para pechar automaticamente fallos en Debian ou Ubuntu
  Debian: `Closes: #595268`; Ubuntu: `LP: #616929`
- Instalado coma `/usr/share/doc/`*`paquete`*`/changelog.Debian.gz`

```
mpich2 (1.2.1.1-5) unstable; urgency=low

  * Use /usr/bin/python instead of /usr/bin/python2.5. Allow
    to drop dependency on python2.5.  Closes: #595268
  * Make /usr/bin/mpdroot setuid. This is the default after
    the installation of mpich2 from source, too. LP: #616929
    + Add corresponding lintian override.

 -- Lucas Nussbaum <lucas@debian.org>  Wed, 15 Sep 2010 18:13:44 +0200
```

debian/control

- Metainformación do paquete
  - Do paquete fonte en si
  - Para cada paquete binario compilado dende esta fonte
- Nome do paquete, sección, prioridade, mantedor, subidores, dependencias de compilación, dependencias, descrición, páxina, ...
- Documentación: Políticas de Debian, capítulo 5
  `https://www.debian.org/doc/debian-policy/ch-controlfields`

```
Source: wget
Section: web
Priority: important
Maintainer: Noel Kothe <noel@debian.org>
Build-Depends: debhelper (>> 5.0.0), gettext, texinfo,
 libssl-dev (>= 0.9.8), dpatch, info2man
Standards-Version: 3.8.4
Homepage: http://www.gnu.org/software/wget/

Package: wget
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: retrieves files from the web
 Wget is a network utility to retrieve files from the Web
```

Arquitectura: todas ou calquera

Dous tipos de paquetes binarios:

- Paquetes con contidos diferentes dependendo da arquitectura Debian
  - Exemplo: Programa C
  - `Architecture: any` en `debian/control`
    - Ou, se só funciona nun certo número de arquitecturas: `Architecture: amd64 i386 ia64 hurd-i386`
  - buildd.debian.org: compila as outras arquitecturas por ti ao o subires
  - Nomeado *paquete_versión_arquitectura*.deb

- Paquetes co mesmo contido en todas as arquitecturas
  - Exemplo: biblioteca Perl
  - `Architecture: all` en `debian/control`
  - Nomeado *paquete_versión_all*.deb

Un paquete fonte pode xerar unha mestura de paquetes binarios `Architecture: any` e `Architecture: all`

debian/rules

- Makefile
- Interface usada para compilar paquetes Debian
- Explicado na Política de Debian, capítulo 4.8
  `https://www.debian.org/doc/debian-policy/ch-source#s-debianrules`
- Obxectivos requiridos:
  - `build`, `build-arch`, `build-indep`: encargaranse de toda a configuración e compilación
  - `binary`, `binary-arch`, `binary-indep`: compilan os paquetes binarios
    - `dpkg-buildpackage` chamará a `binary` para compilar todos os paquetes, ou a `binary-arch` para compilar soamente os paquetes `Architecture: any`
  - `clean`: limpa o cartafol fonte

Axudas no empaquetado – debhelper

- Podes escribir código da liña de ordes directamente en `debian/rules`
- Boas prácticas (usado na maioría dos paquetes): usa un *Axudante no empaquetado*
- Máis popular: **debhelper** (usado polo 98% dos paquetes)
- Obxectivos:
  - Organizar as tarefas comúns nas ferramentas normativizadas usadas por todos os paquetes
  - Arranxar algúns fallos de empaquetado en todos os paquetes

  dh_installdirs, dh_installchangelogs, dh_installdocs, dh_install, dh_installdebconf, dh_installinit, dh_link, dh_strip, dh_compress, dh_fixperms, dh_perl, dh_makeshlibs, dh_installdeb, dh_shlibdeps, dh_gencontrol, dh_md5sums, dh_builddeb, ...
  - Chamado dende `debian/rules`
  - Pódese configurar usando opcións na liña de ordes ou ficheiros en `debian/`

  package`.docs`, *package*`.examples`, *package*`.install`, *package*`.manpages`, ...
- Axudas de terceiros para conxuntos de paquetes: **python-support**, **dh_ocaml**, ...
- `debian/compat`: Versión compatible de debhelper
  - Define o comportamento preciso de dh_*

debian/rules usando debhelper (1/2)

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

# Descomenta para activar o modo detallado.
#export DH_VERBOSE=1

build:
	$(MAKE)
	#docbook-to-man debian/nomepaquete.sgml > nomepaquete.1

clean:
	dh_testdir
	dh_testroot
	rm -f build-stamp configure-stamp
	$(MAKE) clean
	dh_clean

install: build
	dh_testdir
	dh_testroot
	dh_clean -k
	dh_installdirs
	# Engade aquí comandos para instalar o paquete en debian/nomep
	$(MAKE) DESTDIR=$(CURDIR)/debian/nomepaquete install
```

debian/rules usando debhelper (2/2)

```
# Compilar ficheiros comúns a todas as arquitecturas aquí.
binary-indep: build install

# Compilar ficheiros específicos a cada arquitectura aquí.
binary-arch: build install
        dh_testdir
        dh_testroot
        dh_installchangelogs
        dh_installdocs
        dh_installexamples
        dh_install
        dh_installman
        dh_link
        dh_strip
        dh_compress
        dh_fixperms
        dh_installdeb
        dh_shlibdeps
        dh_gencontrol
        dh_md5sums
        dh_builddeb

binary: binary-indep binary-arch
.PHONY: build clean binary-indep binary-arch binary install configure
```

CDBS

- Con debhelper, inda queda moita redundancia entre paquetes
- Axudas de segundo nivel que organizan funcións comúns
  - Por exemplo compilando con `./configure && make && make install` ou CMake
- CDBS:
  - Introducido en 2005, baseado en maxia avanzada do *GNU make*
  - Documentación: `/usr/share/doc/cdbs/`
  - É compatible con Perl, Pyton, Ruby, GNOME, KDE, Java, Haskell, ...
  - Pero algunhas persoas o odian:
    - As veces é difícil personalizar as compilacións dos paquetes: «*labirinto arrevesado de makefiles e variables do entorno*»
    - Máis lento que usar só o debhelper (moitas chamadas inútiles a `dh_*`)

```
#!/usr/bin/make -f
include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/class/autotools.mk

# Engadir unha acción despois de compilar
build/meupaquete::
    /bin/bash debian/scripts/foo.sh
```

Dh (tamén chamado Debhelper 7, ou dh7)

- Introducido en 2008 como o *asasino do CDBS*
- comando **dh** que chama `dh_*`
- Sinxelo *debian/rules*, só describindo as ordes substituídas
- Máis sinxelo de personalizar que o CDBS
- Documentación: páxinas do manual (`debhelper(7)`, `dh(1)`) + presentacións da conferencia DebConf9
  `http://kitenet.net/~joey/talks/debhelper/debhelper-slides.pdf`

```
#!/usr/bin/make -f
%:
    dh $@

override_dh_auto_configure:
     dh_auto_configure -- --with-kitchen-sink

override_dh_auto_build:
     make world
```

debhelper clásico contra CDBS contra dh

- Porcentaxes de popularidade:
  debhelper clásico: 15%   CDBS: 15%   dh: 68%
- Cales debería coñecer?
  - Probablemente todos
  - Necesitas coñecer debhelper para usar dh e CDBS
  - Ao mellor tes que modificar paquetes CDBS
- Cal debería usar para un novo paquete?
  - **dh** (o único cuxa popularidade crece)
  - Véxase `https://trends.debian.net/#build-systems`

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Compilando paquetes

- `apt-get build-dep mypackage`
  Instala as dependencias de compilación *build-dependencies* (para paquetes xa existentes en Debian)
  Ou `mk-build-deps -ir` (para paquetes non existentes)

- `debuild`: Compila e proba con `lintian`, firma con GPG

- Tamén pode chamar a `dpkg-buildpackage` directamente
  - Normalmente con `dpkg-buildpackage -us -uc`

- É mellor compilar os paquetes nun ambiente limpo e co mínimo necesario
  - `pbuilder` – Axuda para compilar paquetes nun *chroot*
    Boa documentación: `https://wiki.ubuntu.com/PbuilderHowto`
    (optimización: `cowbuilder` `ccache` `distcc`)
  - `schroot` e `sbuild`: usados nos daemons
    (non é tan sinxelo coma `pbuilder`, pero permite usar imaxes LVM
    véxase: `https://help.ubuntu.com/community/SbuildLVMHowto`)

- Xera ficheiros `.deb` e un ficheiro `.changes`

Instalando e probando paquetes

- Instalar o paquete de forma local: `debi` (usa `.changes` para saber que ten que instalar)
- Amosar os contidos do paquete: `debc ../mypackage<TAB>.changes`
- Comparar o paquete cunha versión anterior:
  `debdiff ../mypackage_1_*.changes ../mypackage_2_*.changes`
  ou comparar as fontes:
  `debdiff ../mypackage_1_*.dsc ../mypackage_2_*.dsc`
- Comprobar o paquete co analizador estático `lintian`:
  `lintian ../mypackage<TAB>.changes`
  `lintian -i`: da máis información sobre os erros
  `lintian -EviIL +pedantic`: amosa máis erros
- Subir o paquete a Debian (`dput`) (previa configuración)
- Xestionar un arquivo privado de Debian con `reprepro` ou `aptly`
  Documentación:
  `https://wiki.debian.org/HowToSetupADebianRepository`

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Sesión práctica 1: modificando o paquete grep

1. Vaia a `http://ftp.debian.org/debian/pool/main/g/grep/` e descargue a versión 2.12-2 do paquete
   - Se o paquete fonte non se descomprime automáticamente, descomprímao con `dpkg-source -x grep_*.dsc`
2. Fixese nos ficheiros en `debian/`.
   - Cantos paquetes binarios son xerados por este paquete fonte?
   - Que programa de empaquetado usa este paquete?
3. Compile o paquete
4. Agora imos modificar o paquete. Engada unha entrada no rexistro de cambios e aumente o número de versión.
5. Agora desactiea a compatibilidade con perl-regexp (é unha opción en exttt./configure)
6. Recompile o paquete
7. Compare o orixinal e o novo paquete con debdiff
8. Instale o novo paquete recentemente compilado

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

debian/copyright

- Información das licencias e dos dereitos de autor da fonte e do paquete
- Tradicionalmente un ficheiro de texto
- Novos formatos lexibles polos ordenadores:
  https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/

```
Format: https://www.debian.org/doc/packaging-manuals/copyright-format/1.0/
Upstream-Name: X Solitaire
Source: ftp://ftp.example.com/pub/games

Files: *
Copyright: Copyright 1998 John Doe <jdoe@example.com>
License: GPL-2+
 This program is free software; you can redistribute it
 [...]
 .
 On Debian systems, the full text of the GNU General Public
 License version 2 can be found in the file
 ‘/usr/share/common-licenses/GPL-2’.

Files: debian/*
Copyright: Copyright 1998 Jane Smith <jsmith@example.net>
License:
 [TEXTO DA LICENCIA]
```

Modificando a fonte orixinal

As veces hai que:

- Arranxar fallos ou engadir personalizacións específicas de Debian
- Copiar parches dunha versión máis actual na fonte orixinal

Algunhas formas de facelo:

- Modificando directamente os ficheiros
  - Sinxelo
  - Pero non temos forma de rexistrar e documentar os cambios
- Usando o sistema de parches
  - Facilita a contribución dos teus parches á fonte orixinal
  - Axuda a compartir os parches cos derivativos
  - Dalle máis publicidade aos cambios
    `http://patch-tracker.debian.org/` (por agora non funciona)

Sistemas de parcheado

- Principios: os cambios almacénanse coma parches en `debian/patches/`
- Aplicados e quitados durante a compilación
- Antes: varias implementacións – *simple-patchsys* (*cdbs*), *dpatch*, ***quilt***
  - Cada unha permite dous obxectivos de `debian/rules`:
    - `debian/rules patch`: aplica todos os parches
    - `debian/rules unpatch`: quita todos os parches
  - Máis información: `https://wiki.debian.org/debian/patches`
- **Novo formato do paquete fonte cun sistema de parches incorporado: 3.0 (quilt)**
  - Solución recomendada
  - Terá que coñecer *quilt*
    `https://perl-team.pages.debian.net/howto/quilt.html`
  - Ferramenta válida para todos os sistemas de parches en `devscripts`: `edit-patch`

Documentar os parches

- Cabeceiras normativizadas ao comenzo do parche
- Documentado en DEP-3 - Normas para o Etiquetado de Parches
  http://dep.debian.net/deps/dep3/

```
Description: Arranxar a velocidade ao fedellar nos trebellos
 Fedellar nos trebellos moi r{\'a}pido causaba explosi{\'o}ns.
Forwarded: http://lists.example.com/2010/03/1234.html
Author: John Doe <johndoe-guest@users.alioth.debian.org>
Applied-Upstream: 1.2, http://bzr.foo.com/frobnicator/revision/123
Last-Update: 2010-03-29

--- a/src/widgets.c
+++ b/src/widgets.c
@@ -101,9 +101,6 @@ struct {
```

Facendo cousas durante a instalación e o borrado

- As veces non é suficiente descomprimir o paquete
- Crear/borrar usuarios do sistema, arrancar/parar servizos, xestionar *alternativas*
- Iso faise nos *programas de mantemento* `preinst`, `postinst`, `prerm`, `postrm`
  - Pódense xerar exemplos para accións comúns con debhelper
- Documentación:
  - Manual das Políticas de Debian, capítulo 6 `https://www.debian.org/doc/debian-policy/ch-maintainerscripts`
  - Informe dos Desenvolvedores de Debian, capítulo 6.4 `https://www.debian.org/doc/developers-reference/best-pkging-practices.html`
  - `https://people.debian.org/~srivasta/MaintainerScripts.html`
- Preguntarlle ao usuario
  - Débese facer con **debconf**
  - Documentación: `debconf-devel(7)` (paquete `debconf-doc`)

Vixiando as versións da fonte orixinal

- Indica onde mirar en `debian/watch` (véxase `uscan(1)`)

```
version=3

http://tmrc.mit.edu/mirror/twisted/Twisted/(\d\.\d)/ \
  Twisted-([\d\.]*)\.tar\.bz2
```

- Hai rastrexadores de novas versións nas fontes orixinais, que lle avisan automaticamente ao mantedor mediante diferentes paneis de control, como `https://tracker.debian.org/` e `https://udd.debian.org/dmd/`
- `uscan`: executar unha comprobación manual
- `uupdate`: tenta actualizar o teu paquete á última versión da fonte orixinal

Empaquetando cun Sistema de Control de Versións

- Varias ferramentas para axudarche a xestionar as pólas e etiquetas no teu empaquetado:
  `svn-buildpackage`, `git-buildpackage`
- Exemplo: `git-buildpackage`
  - A póla `upstream` para seguir a fonte orixinal coa etiqueta `upstream/`*`version`*
  - A póla `master` segue o paquete de Debian
  - Para cada subída as etiquetas `debian/`*`version`*
  - A póla `pristine-tar` para reconstruír o arquivo tar orixinal

  Documentación: `http://honk.sigxcpu.org/projects/git-buildpackage/manual-html/gbp.html`
- Para atopar o repositorio use os campos `Vcs-*` en `debian/control`
  - `https://wiki.debian.org/Salsa`

```
Navegador SCV: https://salsa.debian.org/debian/devscripts
SCV Git: https://salsa.debian.org/debian/devscripts.git

Navegador SCV: https://salsa.debian.org/perl-team/modules/packages/libwww-perl
SCV Git: https://salsa.debian.org/perl-team/modules/packages/libwww-perl.git
```

- Interface SCV independente: `debcheckout`, `debcommit`, `debrelease`

Modernizar paquetes

- Obxectivo: usar unha versión nova do paquete nun sistema vello
  p.e. usar *mutt* de Debian *inestable* en Debian *estable*

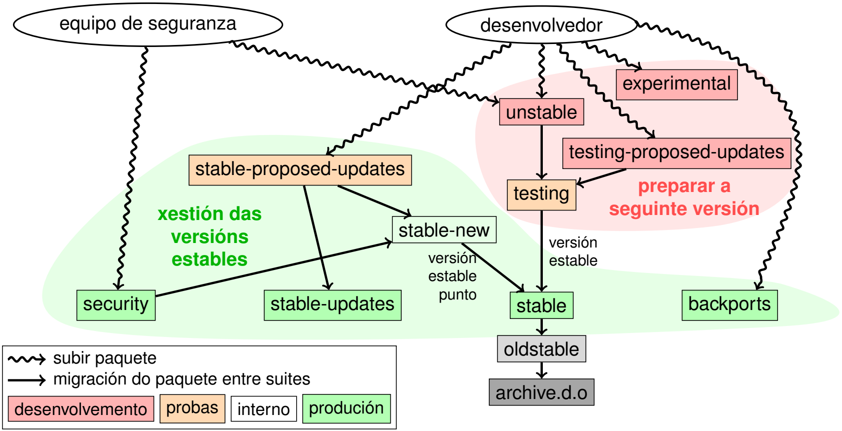
- Idea xeral:
  - Coller o paquete fonte de Debian inestable
  - Modificalo para que compile e funcione ben en Debian estable
    - Ás veces banal (non necesita cambios)
    - Ás veces dificil
    - Ás veces imposible (hai moitas dependencias que non se poden conseguir)

- Algunhas modernizacións («backports» en inglés) está feitas e aprobadas polo proxecto Debian
  `http://backports.debian.org/`

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Arquivo e suites de Debian



Baseado nu gráfico de Antoine Beaupré. `https://salsa.debian.org/debian/package-cycle`

Suites para o desenvolvemento

- As novas versións dos paquetes sóbense á suite inestable **unstable** (**sid**)
- Os paquetes migran dende **unstable** a **testing** dependendo de varios criterios (coma que leve en unstable 10 días, e que non tivera regresións)
- Os novos paquetes tamén se poden subir a:
  - **experimental** (para os paquetes máis *experimentais*, por exemplo cando a nova versión inda non está lista para substituír á que xa existe en inestable)
  - **testing-proposed-updates**, para actualizar a versión de probas en **testing** sen pasar por **unstable** (non se adoita facer)

Conxelando e soltando unha nova versión

- Chegado o momento no ciclo de versións, o equipo de versións decide *conxelar* as probas: páranse as migracións automáticas dende **unstable** a **testing**, e substitúense por unha aprobación manual
- Cando o equipo de versións considera que **testing** está listo para saír:
  - A suite de probas **testing** convértese na nova suite **stable**
  - Da mesma forma, o anterior **stable** convértese en **oldstable**
  - As versións que quedaron sen mantemento móvense a `archive.debian.org`
- Véxase `https://release.debian.org/`

Xestión e suites das versións estables

- Úsanse varias suites para fornecer a versión estable dos paquetes:
  - **stable**: a suite principal
  - A suite de actualizacións **security** aloxada en `security.debian.org`, usada polo equipo de seguridade. As actualizacións anúncianse na lista de correo `debian-security-announce`
  - **stable-updates**: actualizacións estables que non están relacionadas coa seguridade, pero que deberían ser instaladas con urxencia (sen esperar á seguinte versión punto): bases de datos dos antivirus, paquetes relacionados coas franxas horarias, etc. Anúncianse na lista de correo `debian-stable-announce`
  - **backports**: Modernizacións baseadas na versión en **testing**
- A suite estable **stable** actualízase cada par de meses por *versións punto estables* (que só inclúen arranxar fallos)
  - Os paquetes pensando na seguinte versión punto estable sóbense á suite de propostas de actualización **stable-proposed-updates** e son revisadas polo equipo de versións
- A seguinte versión estable anterior **oldstable** ten o mesmo conxunto de suites

Varias maneiras de contribuír a Debian

- As **peores** maneiras de contribuír:
  1. Empaquetar a túa propia aplicación
  2. Metela en Debian
  3. E desaparecer
- As **mellores** maneiras de contribuír:
  - Implicarse nos equipos de empaquetado
    - Moitos equipos céntranse nun conxunto de paquetes, e necesitan axuda
    - Equipos dispoñibles en `https://wiki.debian.org/Teams`
    - Unha maneira excelente de aprender dos contribuíntes máis experimentados
  - Adoptar paquetes xa existentes pero sen mantedores (*paquetes orfos*)
  - Traer novos programas a Debian
    - Pero só se son interesantes e/ou útiles, por favor
    - Hai alternativas xa empaquetadas en Debian?

Adoptar paquetes orfos

- Moitos paquetes sen mantedores en Debian
- Lista completa e proceso: `https://www.debian.org/devel/wnpp/`
- Instalado na túa máquina: `wnpp-alert`
  Ou mellor: `how-can-i-help`
- Diferentes estados:
  - **O**rfos: o paquete non ten mantedores
    Adopta sen medo
  - **RFA**: Mantedor buscando adopción [**R**equest **F**or **A**dopter]
    O mantedor está buscando adopción, pero segue a traballar nel entrementres
    Adóptao sen medo. É de boa educación avisar ao mantedor
  - **ITA**: Intención De Adoptar [**I**ntent **T**o **A**dopt]
    Alguén quere adoptar o paquete
    Podes porte coma candidato!
  - **RFH**: Requírese Axuda [**R**equest **F**or **H**elp]
    O mantedor está buscando axuda
- Algúns paquetes sen mantedores non descubertos → inda non son orfos

Adoptando un paquete: exemplo

```
De: Ti <ti@teudominio>
A: 640454@bugs.debian.org, control@bugs.debian.org
Cc: Francois Marier <francois@debian.org>
Asunto: ITA: verbiste -- conxugador franc{\'e}s

retitle 640454 ITA: verbiste -- conxugador franc{\'e}s
owner 640454 !
thanks

Ola,

Estou usando verbiste e estou disposto a coidar do paquete.

Sa{\'u}dos,

Ti
```

- É de boas maneiras contactar o mantedor previo (sobre todo se o paquete estaba RFA, non orfo)
- É unha moi boa idea contactar o proxecto orixinal

Levando o teu paquete a Debian

- Non necesitas ter ningún título para poder meter o teu paquete en Debian
  1. Envía unha petición **ITP** (Intención De Empaquetado, ou en inglés **I**ntent **T**o **P**ackage) usando `reportbug wnpp`
  2. Preparar un paquete fonte
  3. Atopar un desenvolvedor de Debian que patrocine o teu paquete
- Estado oficial (cando sexa un mantedor de paquetes experimentado):
  - **Mantedor Debian (DM):**
    Permiso para subir os teus propios paquetes
    Véxase `https://wiki.debian.org/DebianMaintainer`
  - **Desenvolvedor Debian (DD):**
    Membro do proxecto Debian; pode votar e subir calquera paquete

Cousas a comprobar antes de pedir padroádego

- Debian **céntrase moito na calidade**
- Normalmente, **os patróns son difíciles de atopar e están moi ocupados**
  - Asegúrate de que o teu paquete está listo antes de preguntar por un padroádego
- A comprobar:
  - Que teña todas as dependencias de compilación: asegúrate de que o teu paquete compila ben nun *chroot sid* limpo
    - Recoméndase usar `pbuilder`
  - Execute `lintian -EviIL +pedantic` no seu paquete
    - Os erros débense arranxar, os outros problemas son máis opcionais
  - Por suposto, faille moitas probas ao teu paquete
- Se tes dúbidas, pide axuda

Onde atopar axuda?

Axuda necesaria:

- Consellos e respostas as túas preguntas, revisións do código
- Padroádegos para as túas subidas, cando o teu paquete estea listo

Podes conseguir axuda de:

- **Outros membros dun equipo de empaquetado**
  - Lista de equipos: `https://wiki.debian.org/Teams`
- O **grupo de Mentores de Debian** (se o teu paquete non encaixa con ningún equipo)
  - `https://wiki.debian.org/DebianMentorsFaq`
  - Lista de correo: `debian-mentors@lists.debian.org` (tamén unha boa forma de aprender dos erros)
  - IRC: `#debian-mentors` en `irc.debian.org`
  - `http://mentors.debian.net/`
  - Documentación: `http://mentors.debian.net/intro-maintainers`
- **Listas de correo sobre a localización** (obter axuda na túa lingua)
  - `debian-devel-{french,italian,portuguese,spanish}@lists.d.o`
  - Lista completa: `https://lists.debian.org/devel.html`
  - Ou lista de usuarios: `https://lists.debian.org/users.html`

Máis documentación

- Curruncho dos Desenvolvedores de Debian
  `https://www.debian.org/devel/`
  Ligazóns a moitos recursos sobre desenvolver en Debian
- Guía para os Mantedores de Debian
  `https://www.debian.org/doc/manuals/debmake-doc/`
- Referencia dos Desenvolvedores de Debian
  `https://www.debian.org/doc/developers-reference/`
  A meirande parte é sobre os procesos de Debian, pero tamén inclúe boas prácticas no empaquetado (parte 6)
- Política de Debian
  `https://www.debian.org/doc/debian-policy/`
  - Todos os requirimentos que todo paquete ten que satisfacer
  - Políticas específicas para Perl, Java, Python, ...
- Guía de Empaquetado Ubuntu
  `https://packaging.ubuntu.com/html/`

Paneis de control de Debian para mantedores

- **Orientado aos paquetes fonte**: `https://tracker.debian.org/dpkg`
- **Orientado aos equipos/mantedores**: Vista Xeral dos Paquetes dos Desenvolvedores (DDPO) `https://qa.debian.org/developer.php?login=pkg-ruby-extras-maintainers@lists.alioth.debian.org`
- **Orientado ás listas PORFACER(TODO)**: Panel de Control dos Mantedores de Debian (DMD) `https://udd.debian.org/dmd/`

Usando o Sistema de Seguimento de Fallos de Debia

- Unha maneira bastante única para xestionar fallos
  - Interface na rede para consultar fallos
  - Interface por correo electrónico para facer cambios nos fallos
- Engadir información sobre os fallos:
  - Escribe a `123456@bugs.debian.org` (sen incluír o remitente, necesitas engadir `123456-submitter@bugs.debian.org`)
- Cambiar a categoría do problema:
  - Envía ordes a `control@bugs.debian.org`
  - Interface por liña de ordes: comando `bts` en `devscripts`
  - Documentación: `https://www.debian.org/Bugs/server-control`
- Avisando de fallos: use `reportbug`
  - Normalmente usado cun servidor local de correo: instale `ssmtp` ou `nullmailer`
  - Ou use `reportbug --template`, e envíeo (manualmente) a `submit@bugs.debian.org`

Usando o BTS: exemplos

- Enviándolle un correo ao problema e o remitente:
  https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=680822#10
- Etiquetando e cambiando a importancia:
  https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=680227#10
- Resignando, cambiando a importancia, cambiando o título, ...:
  https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=680822#93
  - `notfound`(non atopado), `found`(atopado), `notfixed`(sen arranxar), `fixed`(arranxado) son para **version-tracking**(seguimento de versións)
    Véxase
    https://wiki.debian.org/HowtoUseBTS#Version_tracking
- Usando etiquetas de usuario: https://bugs.debian.org/cgi-bin/bugreport.cgi?msg=42;bug=642267
  Véxase https://wiki.debian.org/bugs.debian.org/usertags
- Documentación BTS:
  - https://www.debian.org/Bugs/
  - https://wiki.debian.org/HowtoUseBTS

Estás máis interesado en Ubuntu?

- Ubuntu normalmente céntrase en xestionar as diferenzas con Debian
- Non se centran en paquetes específicos
  En troques, colaboran cos equipos de Debian
- Normalmente recomendan subir os novos paquetes a Debian primeiro
  `https://wiki.ubuntu.com/UbuntuDevelopment/NewPackages`
- Unha mellor idea:
  - Involúcrate nun equipo de Debian e fai de ponte con Ubuntu
  - Axuda a reducir as diverxencias e os fallos de triaxe no Launchpad
  - Moitas ferramentas en Debian:
    - A columna de Ubuntu no resumo do desenvolvedor dos paquetes
    - A caixa de Ubuntu no Sistema de Seguimento de Paquetes (PTS)
    - Recibir correo dos fallos do Launchpad polo PTS

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Conclusión

- Xa ten unha vista completa do sistema de empaquetado en Debian
- Pero vai necesitar ler máis documentación
- O que se consideran boas prácticas vai cambiando co tempo
  - Se ten dúbidas, use a axuda de empaquetado **dh**, e o formato **3.0 (quilt)**

Correccións: **packaging-tutorial@packages.debian.org**

Asuntos legais

Copyright ©2011–2019 Lucas Nussbaum – lucas@debian.org

Este documento é software libre: podes redistribuílo e/ou modificalo baixo (escolla segundo as súas preferencias):

- Os termos da Licencia Pública Xeral GNU tal coma a publica a Free Software Foundation, xa for versión 3 da Licenza, ou (se o prefire) unha versión posterior.
  `http://www.gnu.org/licenses/gpl.html`
- Os termos da licenza Creative Commons Attribution-ShareAlike 3.0 Unported License.
  `http://creativecommons.org/licenses/by-sa/3.0/`

Contribuír ao tutorial

- Contribúa:
  - `apt-get source packaging-tutorial`
  - `debcheckout packaging-tutorial`
  - `git clone https://salsa.debian.org/debian/packaging-tutorial.git`
  - `https://salsa.debian.org/debian/packaging-tutorial`
  - Fallos sen resolver: `bugs.debian.org/src:packaging-tutorial`
- Enviar correccións:
  - `mailto:packaging-tutorial@packages.debian.org`
    - Que deberíamos engadirlle ao tutorial?
    - Que podemos mellorar?
  - `reportbug packaging-tutorial`

Contidos

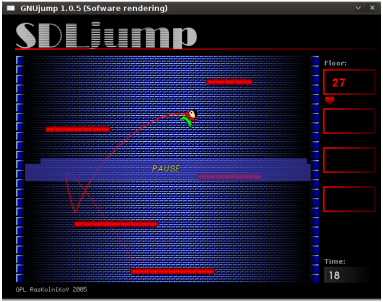
1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Sesión práctica 2: empaquetar GNUjump

1. Descargue GNUjump 1.0.8 dende
   `http://ftp.gnu.org/gnu/gnujump/gnujump-1.0.8.tar.gz`

2. Cree un paquete Debian para el
   - Instale as dependencias de compilación para poder compilar o paquete
   - Arranxe os fallos
   - Cree un paquete funcional básico
   - Remate de encher `debian/control` e outros ficheiros

3. Disfrute



Sesión práctica 2: empaquetando GNUjump (consel

- Para obter un paquete funcional básico use `dh_make`
- Primeiro, crear un paquete fonte *1.0* é máis sinxelo que un *3.0 (quilt)* (pode cambialo en `debian/source/format`)
- Para buscar as dependencias de compilación necesarias, busque o ficheiro que falta e use `apt-file` para atopar o paquete necesario
- Se atopa este erro:

```
/usr/bin/ld: SDL_rotozoom.o: undefined reference to symbol 'ceil@@GLIBC_2.2.5'
//lib/x86_64-linux-gnu/libm.so.6: error adding symbols: DSO missing from command line
collect2: error: ld returned 1 exit status
Makefile:376: recipe for target 'gnujump' failed
```

Ten que engadir `-lm` á liña de ordes do ligante:
Modifique `src/Makefile.am` e substitúa

```
gnujump_LDFLAGS = $(all_libraries)
```

por

```
gnujump_LDFLAGS = -Wl,--as-needed
gnujump_LDADD = $(all_libraries) -lm
```

Despois execute `autoreconf -i`

Sesión práctica 3: empaquetando unha biblioteca de

1. Bótelle unha ollada a algunha documentación sobre como empaquetar en Java:
   - `https://wiki.debian.org/Java`
   - `https://wiki.debian.org/Java/Packaging`
   - `https://www.debian.org/doc/packaging-manuals/java-policy/`
   - `/usr/share/doc/javahelper/tutorial.txt.gz`
2. Descargue IRClib dende `http://moepii.sourceforge.net/`
3. Empaquéteo

Sesión práctica 4: empaquetando unha xema de Rul

1. Bótelle unha ollada á documentación sobre como empaquetar en Ruby:
   - https://wiki.debian.org/Ruby
   - https://wiki.debian.org/Teams/Ruby
   - https://wiki.debian.org/Teams/Ruby/Packaging
   - `gem2deb(1)`, `dh_ruby(1)` (no paquete `gem2deb`)
2. Cree un paquete fonte Debian básico a partir da xema `peach`:
   `gem2deb peach`
3. Mellóreo para que se poida converter nun verdadeiro paquete Debian

Sesión práctica 5: empaquetar un módulo Perl

1. Bótelle unha ollada á documentación sobre como empaquetar en Perl:
   - `https://perl-team.pages.debian.net`
   - `https://wiki.debian.org/Teams/DebianPerlGroup`
   - `dh-make-perl(1)`, `dpt(1)` (no paquete `pkg-perl-tools`)
2. Cree un paquete fonte Debian básico a partir da distribución CPAN `Acme`:
   `dh-make-perl --cpan Acme`
3. Mellóreo para que se poida converter nun verdadeiro paquete Debian

Contidos

1. Introdución
2. Creando paquetes fonte
3. Compilando e probando paquetes
4. Sesión práctica 1: modificando o paquete grep
5. Temas de empaquetado avanzados
6. Mantendo paquetes en Debian
7. Conclusión
8. Sesións prácticas adicionais
9. Respostas ás sesións prácticas

Respostas as sesións prácticas

Sesión práctica 1: modificando o paquete grep

1. Vaia a `http://ftp.debian.org/debian/pool/main/g/grep/` e descargue a versión 2.12-2 do paquete
2. Fixese nos ficheiros en `debian/`.
   - Cantos paquetes binarios son xerados por este paquete fonte?
   - Que programa de empaquetado usa este paquete?
3. Compile o paquete
4. Agora imos modificar o paquete. Engada unha entrada no rexistro de cambios e aumente o número de versión.
5. Agora desactiea a compatibilidade con perl-regexp (é unha opción en exttt./configure)
6. Recompile o paquete
7. Compare o orixinal e o novo paquete con debdiff
8. Instale o novo paquete recentemente compilado

Obtendo a fonte

1. Vaia a `http://ftp.debian.org/debian/pool/main/g/grep/` e descargue a versión 2.12-2 do paquete

- Use dget para descargar o ficheiro `.dsc`:
  `dget http://cdn.debian.net/debian/pool/main/g/grep/grep_2.12-2.dsc`
- Se ten `deb-src` nunha versión de Debian con `grep` versión 2.12-2 (consúlteo en `https://tracker.debian.org/grep`), pode usar `apt-get source grep=2.12-2`
  ou `apt-get source grep/release` (p.e. `grep/stable`)
  ou, se quere tentar á sorte: `apt-get source grep`
- O paquete fonte `grep` ten 3 ficheiros:
  - `grep_2.12-2.dsc`
  - `grep_2.12-2.debian.tar.bz2`
  - `grep_2.12.orig.tar.bz2`

  Esto é típico do formato "3.0 (quilt)".
- Se é necesario, descomprima a fonte con
  `dpkg-source -x grep_2.12-2.dsc`

Botando unha ollada e compilando o paquete

2. Mire os ficheiros en `debian/`
   - Cantos paquetes binarios son xerados por este paquete fonte?
   - Que programa de empaquetado usa este paquete?

- Segundo `debian/control`, este paquete só xera un paquete binario, nomeado `grep`.
- Segundo `debian/rules`, este paquete é típico do empaquetado debhelper *clásico*, sen usar *CDBS* ou *dh*. Pódense ver as diferentes chamadas ás ordes `dh_*` en `debian/rules`.

3. Compile o paquete

- Use `apt-get build-dep grep` para obter as dependencias de compilación
- Entón use `debuild` ou `dpkg-buildpackage -us -uc` (Tarda aprox. 1 min)

Modificar o rexistro de cambios

4. Agora imos modificar o paquete. Engada unha entrada no rexistro de cambios e aumente o número de versión.

- `debian/changelog` é un ficheiro de texto. Pode modificalo e engadir novas entradas manualmente.
- Ou pode usar `dch -i`, que engade unha entrada e abre o procesador de textos
- O nome e correo electrónico pódense definir usando as variables de entorno `DEBFULLNAME` e `DEBEMAIL`
- Despois diso, recompile o paquete: compilouse unha nova versión do paquete
- O sistema de versións dos paquetes explícase na sección 5.6.12 da Política de Debian
  `https://www.debian.org/doc/debian-policy/ch-controlfields`

Desactivar a compatibilidade con expresións regula

5. Agora desactiea a compatibilidade con perl-regexp (é unha opción en exttt./configure)
6. Recompile o paquete

- Comprobeo con `./configure --help`: a opción para desactivar as expresións regulares Perl é `--disable-perl-regexp`
- Modifique `debian/rules` e atope a liña `./configure`
- Engada `--disable-perl-regexp`
- Recompile con `debuild` ou `dpkg-buildpackage -us -uc`

Comparar e probar os paquetes

7. Compare o orixinal e o novo paquete con debdiff
8. Instale o novo paquete recentemente compilado

- Compara os paquetes binarios: `debdiff ../*changes`
- Comparar os paquetes fonte: `debdiff ../*dsc`
- Instalar o paquete acabado de compilar: `debi`
  Ou `dpkg -i ../grep_<TAB>`
- `grep -P foo` xa non funcionará!

Reinstalar a anterior versión do paquete:

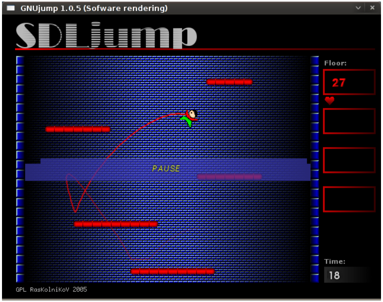
- `apt-get install --reinstall grep=2.6.3-3` *(= versión anterior)*

Sesión práctica 2: empaquetar GNUjump

1. Descargue GNUjump 1.0.8 dende
   `http://ftp.gnu.org/gnu/gnujump/gnujump-1.0.8.tar.gz`

2. Cree un paquete Debian para el
   - Instale as dependencias de compilación para poder compilar o paquete
   - Cree un paquete funcional básico
   - Remate de encher `debian/control` e outros ficheiros

3. Disfrute



Paso a paso...

- `wget http://ftp.gnu.org/gnu/gnujump/gnujump-1.0.8.tar.gz`
- `mv gnujump-1.0.8.tar.gz gnujump_1.0.8.orig.tar.gz`
- `tar xf gnujump_1.0.8.orig.tar.gz`
- `cd gnujump-1.0.8/`
- `dh_make -f ../gnujump-1.0.8.tar.gz`
  - Tipo de paquete: único binario (por agora)

```
gnujump-1.0.8$ 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
```

Paso a paso. . . (2)

- Mire `debian/changelog`, `debian/rules`, `debian/control` (xa cuberto por **dh_make**)
- En `debian/control`:
  `Build-Depends: debhelper (>= 7.0.50 ), autotools-dev`
  Amosa as *build-dependencies* = paquetes necesarios para compilar o paquete
- Intente compilar o paquete tal como está con `debuild` (grazas á máxia do **dh**)
  - E engada dependencias de compilación ata que compile
  - Pista: use `apt-cache search` e `apt-file` para atopar os paquetes
  - Por exemplo:

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

    → Engádalle **libsdl1.2-dev** a Build-Depends e instáleo.
  - Ou mellor: use **pbuilder** para compilar nun ambiente limpo

Paso a paso. . . (3)

- As dependencias de compilación necesarias son `libsdl1.2-dev`, `libsdl-image1.2-dev`, `libsdl-mixer1.2-dev`
- Despois, probablemente se atope con outro erro:

```
/usr/bin/ld: SDL_rotozoom.o: undefined reference to symbol 'ceil@@GLIBC_2.2.5'
//lib/x86_64-linux-gnu/libm.so.6: error adding symbols: DSO missing from command line
collect2: error: ld returned 1 exit status
Makefile:376: recipe for target 'gnujump' failed
```

- Este problema é causado por corrupción do binario: gnujump non foi axustado despois de cambios na ligadura.
- Se está a usar a versión **1.0** do formato fonte, pode cambiar directamente as fontes orixinais.
  - Modifique `src/Makefile.am` e substitúa

    ```
    gnujump_LDFLAGS = $(all_libraries)
    ```

    por

    ```
    gnujump_LDFLAGS = -Wl,--as-needed
    gnujump_LDADD = $(all_libraries) -lm
    ```

  - Despois execute `autoreconf -i`

Paso a paso…(4)

- Se estas a usar a versión **3.0 (quilt)** do formato fonte, usa `quilt` para preparar un parche. (Véxase `https://wiki.debian.org/UsingQuilt`)
  - `export QUILT_PATCHES=debian/patches`
  - `mkdir debian/patches`
    `quilt new linker-fixes.patch`
    `quilt add src/Makefile.am`
  - Modifique `src/Makefile.am` e substitúa

    ```
    gnujump_LDFLAGS = $(all_libraries)
    ```

    por

    ```
    gnujump_LDFLAGS = -Wl,--as-needed
    gnujump_LDADD = $(all_libraries) -lm
    ```

  - `quilt refresh`
  - Como o `src/Makefile.am` foi modificado, autoreconf debe ser chamado durante a compilación. Para que `dh` o faga automáticamente, cambie a chamada a `dh` en `debian/rules` de: `dh $ --with autotools-dev`
    a: `dh $ --with autotools-dev --with autoreconf`

Paso a paso…(5)

- Agora o paquete debería compilar sen problemas.
- Use `debc` para listar os contidos do paquete xerado, e `debi` para instalalo e probalo.
- Comprobe o paquete con `lintian`
  - Aínda que non é obrigatorio, recoméndase que os paquetes que se suban a Debian estean *sen erros de lintian*
  - Pódense listar os problemas con `lintian -EviIL +pedantic`
  - Algúns consellos:
    - Borre os ficheiros innecesarios en `debian/`
    - Encha `debian/control`
    - Sobrescriba `dh_auto_configure` para instalar o executable en `/usr/games`
    - Use as opcións de endurecemento do compilador para aumentar a seguridade.
      Véxase `https://wiki.debian.org/Hardening`

Paso a paso. . . (6)

- Compare o teu paquete co que xa existe en Debian:
  - O paquete en Debian separa os ficheiros de datos nun paquete secundario, que é o mesmo en todas as arquitecturas ($\rightarrow$ afórrase espazo no arquivo de Debian)
  - Tamén instala un ficheiro .desktop (para os menús de GNOME/KDE) e tamén se integra no menú Debian
  - E arranxa algúns problemas pequenos con parches

Sesión práctica 3: empaquetando unha biblioteca de

1. Bótelle unha ollada a algunha documentación sobre como empaquetar en Java:
   - `https://wiki.debian.org/Java`
   - `https://wiki.debian.org/Java/Packaging`
   - `https://www.debian.org/doc/packaging-manuals/java-policy/`
   - `/usr/share/doc/javahelper/tutorial.txt.gz`
2. Descargue IRClib dende `http://moepii.sourceforge.net/`
3. Empaquéteo

Paso a paso…

- `apt-get install javahelper`
- Cree un paquete fonte básico: `jh_makepkg`
  - Biblioteca
  - Ningún
  - Compilador e sistema execución en tempo real libre por omisión
- Observe e corrixa `debian/*`
- `dpkg-buildpackage -us -uc` ou `debuild`
- `lintian`, `debc`, etc.
- Compare os seus resultados co paquete fonte `libirclib-java`

Sesión práctica 4: empaquetando unha xema de Rul

1. Bótelle unha ollada á documentación sobre como empaquetar en Ruby:
   - `https://wiki.debian.org/Ruby`
   - `https://wiki.debian.org/Teams/Ruby`
   - `https://wiki.debian.org/Teams/Ruby/Packaging`
   - `gem2deb(1)`, `dh_ruby(1)` (no paquete `gem2deb`)
2. Cree un paquete fonte Debian básico a partir da xema `peach`:
   `gem2deb peach`
3. Mellóreo para que se poida converter nun verdadeiro paquete Debian

Paso a paso...

`gem2deb peach`:

- A xema descárgase dende rubygems.org
- Cree un arquivo .orig.tar.gz axeitado, e descomprímao
- Inicialice un paquete fonte Debian alicerzado nos metadatos da xema
    - Chamado `ruby-`*`nomedaxema`*
- Intente compilar o paquete Debian binario (pode fallar)

`dh_ruby` (incluído en *gem2deb*) fai as operacións específicas de Ruby:

- Compila as extensións C para cada versión de Ruby
- Copia os ficheiros ao cartafol de destino
- Actualiza os camiños dos executables nos programas interpretados
- Executa as probas definidas en `debian/ruby-tests.rb`, `debian/ruby-tests.rake`, ou `debian/ruby-test-files.yaml`; xunto con outras comprobacións varias

Paso a paso. . . (2)

Mellorar o paquete xerado:

- Execute `debclean` para limpar o código fonte. Mire en `debian/`
- `changelog` e `compat` deberían estar correctos
- Modifique `debian/control`: mellore a `Description`
- Escriba un ficheiro `copyright` axeitado coa licenza alicerzada nos ficheiros orixinais
- Compile o paquete
- Compare o seu paquete co paquete `ruby-peach` no arquivo de Debian

Sesión práctica 5: empaquetar un módulo Perl

1. Bótelle unha ollada á documentación sobre como empaquetar en Perl:
   - `https://perl-team.pages.debian.net`
   - `https://wiki.debian.org/Teams/DebianPerlGroup`
   - `dh-make-perl(1)`, `dpt(1)` (no paquete `pkg-perl-tools`)
2. Cree un paquete fonte Debian básico a partir da distribución CPAN `Acme`:
   `dh-make-perl --cpan Acme`
3. Mellóreo para que se poida converter nun verdadeiro paquete Debian

Paso a paso…

`dh-make-perl --cpan Acme`:

- Descarga os arquivos tar da CPAN
- Crea un .orig.tar.gz axeitado, e o descomprime
- Inicializa un paquete fonte Debian alicerzado nos metadatos da distribución
    - Chámao `lib`*`nomedistribución`*`-perl`

Paso a paso. . . (2)

Mellorar o paquete xerado:

- Non debería ter que tocar `debian/changelog`, `debian/compat`, `debian/libacme-perl.docs` nin `debian/watch`
- Modifique `debian/control`: mellore a descrición `Description`, e borre o ciscallo no fondo
- Modifique `debian/copyright`: borre o parágrafo sobrante do principio, e engádalle os anos da licenza na liña `Files: *`