

# DEB packages

CC-BY-SA 3.0 ©2017 [jan.celis@kdg.be](mailto:jan.celis@kdg.be)



# Inhoudsopgave

1	INLEIDING.....	3
2	POOR MAN'S DEB.....	4
2.1	Bestanden.....	4
2.1.1	Executable.....	4
2.1.2	Manual Page.....	4
2.1.3	Copyright.....	5
2.1.4	Changelog.....	6
2.1.5	Ikoontje.....	6
2.1.6	desktop bestand .....	6
2.1.7	menu.....	7
2.2	CONTROL FILE.....	7
2.3	LINTIAN.....	8
2.3.1	fakeroot.....	8
2.3.2	Onnodig executable.....	8
2.4	COMPILATIESCRIPT.....	9
2.5	POSTINST.....	10
3	RICH MAN'S DEB.....	11
3.1	Opzetten email en username.....	11
3.2	Maken van de executable.....	12
3.3	dh_make.....	12
3.4	control .....	13
3.4.1	Section.....	13
3.4.2	Dependencies.....	13
3.4.3	Description.....	13
3.4.4	Architecture.....	14
3.5	changelog.....	14
3.6	manpage.....	15
3.7	rules.....	16
3.8	menu.....	16
3.9	pakket.desktop.....	16
3.10	debuild.....	18
3.10.1	install.....	18
3.11	Build fouten.....	18
4	SIGNED DEBS.....	20
4.1	Aanmaak van een sleutelpaar.....	20
4.2	Handtekenen van een bestaande deb.....	21
4.3	Handtekenen van een deb met debbuild.....	21
5	OEFENING.....	22
5.1	Oefening 1 Volg hoofdstuk 3.....	22
5.2	Oefening 2 Opstarten met een .desktop bestand .....	22
5.3	Oefening 3 Pakket voor python .....	22
6	REFERENTIES RTFM.....	27

## 1 INLEIDING

Een DEB bestand is een soort ZIP structuur waarin binaire bestanden (programma's) worden geplaatst. Verder kan je definiëren welke andere programma's of bibliotheken je programma nodig heeft. Je kan ook ikoontjes voorzien en een menu item waarmee het programma kan opgestart worden.

## 2 POOR MAN'S DEB

In dit hoofdstuk genereren we een deb bestand dat voldoet aan de policies zonder hulptools zoals debhelper. Op Linux probeer je deze manier te vermijden, omdat deze voor veel fouten kan zorgen. Op Mac OSX is het mogelijk om op deze manier deb pakketten te genereren voor Linux machines (via macports).

Met het maken van een poor man's deb leer je wel hoe debhelper werkt.

### 2.1 Bestanden

#### 2.1.1 Executable

Bestand:        /usr/bin/pakket  
Rechten:        chmod 755 /usr/bin/pakket  
Lintian:        Lintian kijkt de 755 rechten na en het juiste begin  
                       bv #!/bin/sh

Voorbeeld /usr/bin/pakket  
 #!/bin/sh  
 zenity --warning --text=""`exec uname -a`"



#### Opmerking:

Wanneer je pakket gebruik maakt van bibliotheken of gecompileerde executables, staan deze in /usr/lib/pakket/. Andere onderdelen van je pakket komen in principe in de directory /usr/share/pakket/

Java en mono (.NET) programma's staan hier ook.

Voor bv het examentool staat er /usr/lib/examentool/examentool.exe

In /usr/bin/examentool staat er het volgende:

```
#!/bin/sh
# $@ vangt eventuele argumenten op die iemand meegeeft (bv bestandsnaam)
exec /usr/bin/mono /usr/lib/examentool/examentool.exe "$@"
```

#### 2.1.2 Manual Page

Bestand:        /usr/share/man/man1/pakket  
Rechten:        chmod 644 /usr/share/man/man1/pakket  
Lintian:        Een executable MOET een man page hebben.  
                       Lintian kijkt vooral de eerste lijn na van de manual page  
                       .TH pakket 1 "January 12, 2016" "" ""  
                       De 1 als sectie moet overeenkomen met de directory man1

/usr/share/pixmaps/pakket.xpm  
 Voor elke executable is er minstens een manual page  
 Deze komt in /usr/share/man/man1/pakket.1.gz

sudo apt-get install gmanedit

Er bestaat ook `txt2man`, voor als je enkel in tekstmodus werkt (`sudo apt-get install txt2man`) of `help2man` voor als je de `--help` output wil omzetten naar een manpage. De manual page moet minstens een NAME en een SYNOPSIS hebben,

Maak met `gmanedit` via de menu File, Wizard een eenvoudige manpage voor pakket.1 (met minstens NAME en SYNOPSIS):

```
.\"Created with GNOME Manpages Editor Wizard
.\"http://sourceforge.net/projects/gmanedit2
.TH pakket 1 "December 17, 2012" "" "pakket"

.SH NAME
pakket \- program that shows a graphical uname \-a

.SH SYNOPSIS
.B pakket
.RI [ options ]
.br

.SH DESCRIPTION
This manual page explains the
.B pakket
program. This program uses zenity to show a warning message with
uname \-a info
.PP
\fBpakket\fP is showing uname \-a info in X11

.SH "SEE ALSO"
    zenity(1), uname(1)
AUTHOR
    Jan Celis <jan.celis@kdg.be>
```

```
mkdir -p ./debian/usr/share/man/man1
cp pakket.1 ./debian/usr/share/man/man1/
gzip --best ./debian/usr/share/man/man1/pakket.1
```

### 2.1.3 Copyright

Bestand:     /usr/share/doc/pakket/copyright  
Rechten:     chmod 644 /usr/share/doc/pakket/copyright  
Lintian:     Een executable MOET een 644 copyright hebben

```
mkdir -p ./debian/usr/share/doc/pakket
echo 'Author: Jan Celis <jan.celis@kdg.be> ' > ./debian/usr/share/doc/pakket/copyright
echo 'Copyright: (c) 2016 Jan Celis' >> ./debian/usr/share/doc/pakket/copyright
chmod 644 ./debian/usr/share/doc/pakket/copyright
```

### 2.1.4 Changelog

Bestand: /usr/share/doc/pakket/changelog.gz  
Rechten: chmod 644 /usr/share/doc/pakket/changelog.gz  
Lintian: Een executable MOET een 644 changelog hebben

```
echo 'pakket-0.1 (1) xenial; urgency=low'
>./debian/usr/share/doc/pakket/changelog
echo ' * Made Debian package of pakket.'
>>./debian/usr/share/doc/pakket/changelog
echo ' -- Jan Celis <jan.celis@kdg.be> 2016-01-12' >>
./debian/usr/share/doc/pakket/changelog
cd ./debian/usr/share/doc/pakket/changelog
gzip --best ./debian/usr/share/doc/pakket/changelog
```

### 2.1.5 Ikoontje

Bestand: /usr/share/pixmaps/pakket.xpm  
Rechten: chmod 644 /usr/share/pixmaps/pakket.xpm  
Lintian: Nodig met rechten 644 als je dit in een menu of applicatie gebruikt

Standaard ikoontje van de applicatie komen in /usr/share/pixmaps  
Het ikoontje is rechthoekig in pixels (bv 128x128) en staat in xpm of png formaat.  
Voor ons pakket is dit bijvoorbeeld:  
/usr/share/pixmaps/pakket.png  
Soms worden ook verschillende formaten voorzien in de bestandsnaam bv  
pakket\_16x16.xpm  
pakket\_32x32.xpm

### 2.1.6 desktop bestand

Bestand: /usr/share/applications/pakket.desktop  
Rechten: chmod 644 /usr/share/applications/pakket.desktop  
Lintian: Kijkt na WAT je opstart binnen het bestand .desktop  
Kijkt na of de Categories bestaan in de Debian Policy

Applicaties komen in de directory /usr/share/applications. Je defineert ze in een tekstbestand met de extensie .desktop. Opgelet, de extensie .desktop wordt door de window manager geïnterpreteerd!

Voorbeeld van de inhoud van /usr/share/applications/pakket.desktop:

```
[Desktop Entry]
Version=1.0
Name=Pakket
Exec=gksu /usr/bin/pakket
Icon=pakket
Terminal=false
Type=Application
```

```
Categories=Application;System;
StartupNotify=false
GenericName[en_US]= Pakket
```

Opgelet: Bij Icon geef je GEEN extensie mee. Automatisch wordt er hier gezocht naar bestanden met de extensie png of xpm

### 2.1.7 menu

Bestand: /usr/share/menu/pakket  
Rechten: chmod 644 /usr/share/menu/pakket  
Lintian: Kijkt na WAT je opstart binnen het menubestand  
 Kijkt na of de section bestaat in de Debian Policy

De officiële manier bij debian om een menu item aan te maken en ook het meest compatible bij alle linux systemen is een bestand aanmaken in /usr/share/menu. De meeste pakketten maken zowel een application als een menu aan

Voorbeeld met /usr/share/menu/pakket

In tegenstelling tot een .desktop bestand moet je hier wel het volledige pad opgeven naar het juiste xpm bestand.

```
?package(pakket): \
    command="/usr/bin/pakket" \
    needs="X11" \
    icon="/usr/share/pixmaps/pakket.xpm" \
    section="Applications/Education" \
    hints="Opstarten Pakket" \
    title="Pakket" \
    longtitle="Opstarten Pakket"
```

## 2.2 CONTROL FILE

Bestand: DEBIAN/control  
Rechten: chmod 644 DEBIAN/control  
Lintian: Kijkt alle velden na

```
Package: pakket
Priority: optional
Section: misc
Maintainer: Jan Celis <jan.celis@kdg.be>
Size: 120
Installed-Size: 764
Architecture: all
Depends: zenity (>= 3), menu
Description: Grafische output van uname
  Dit programma gebruikt zenity voor het weergeven van de output
  van uname -a in een grafische warning
Version: 11.12-21
```

## 2.3 LINTIAN

Lintian kijkt na of een pakket voldoet aan de debian policy  
apt-get install lintian

Meest voorkomende fouten:

### 2.3.1 fakeroot

De owner en de group moeten root zijn, en je mag je pakket niet als root maken.

Dit probleem kan je met het pakket fakeroot oplossen

apt-get install fakeroot

Nu kan je het pakket als gewone gebruiker compileren met:

fakeroot dpkg-deb --build debian

### 2.3.2 Onnodig executable

Warning:

W: pakket: executable-not-elf-or-script ./usr/share/pixmaps/pakket.xpm

De rechten van dat bestand staan op executable, maar het is niet uitvoerbaar. Geef het bestand de rechten 644 met chmod

E: kdguntu-metacity-theme: extended-description-is-empty

extended description betekent dat er na de lijn Description (met een korte omschrijving van je pakket) een langere beschrijving volgt.

Belangrijk is wel dat je de extended description begint met een spatie. De extended description moet verstaanbaar zijn voor mensen die niet thuis zijn in Linux (bv je moeder)



## 2.4 COMPILATIESCRIPT

```
#!/bin/bash
PACKET="pakket"
#VERSIE="12.01.01" #datum is versie om steeds een nieuwe versie te krijgen
VERSIE="`date +%y.%m-%d`" #datum is versie => steeds nieuwe versie
ARCHITECTURE="all"
DEB="${PACKET}-${VERSIE}_${ARCHITECTURE}.deb"
MD5SUMS="./DEBIAN/md5sums" # alleen /usr files (geen /etc, /var...)
CONTROL="./debian/DEBIAN/control"

# Grote schoonmaak
find . -iname "*~" -type f -exec rm -rf {} \; # alle tilde bestanden weg
rm -f ./DEBIAN/$MD5SUMS # Anders tellen we bij de size de md5sum mee!

# Update Installed-Size in control
ISIZE="Installed-Size: `du -sk .|cut -f 1`"
sed -i "s/Installed-Size:./$ISIZE/" $CONTROL

# Update Version in control
NEWVERSION="Version: $VERSIE"
sed -i "s/Version:./$NEWVERSION/" $CONTROL

# Update md5sums van alle bestanden in /usr
cd ./debian
find usr -type f -print0 | xargs -0 md5sum > $MD5SUMS

# standaard rechten instellen
cd ..
chmod -R 755 ./debian
chmod -R 755 ./debian/*
chmod 644 ./debian/usr/share/man/man1/*
chmod 644 ./debian/usr/share/menu/*
chmod 644 ./debian/usr/share/doc/$PACKET/*
chmod 755 ./debian/DEBIAN/p*
chmod 644 ./debian/DEBIAN/c*
chmod 644 ./debian/DEBIAN/md5sums

chmod -R 644 ./debian/usr/share/applications/*
#chmod -R 644 ./debian/usr/share/lintian/overrides/*
chmod -R 644 ./debian/usr/share/$PACKET/*
chmod -R 644 ./debian/usr/share/pixmaps/*

# pakket aanmaken en nakijken op compatibiliteit
fakeroot dpkg-deb --build debian
mv debian.deb "$PACKET.deb"
echo "Lintian rapport van $PACKET.deb:"
lintian -i --color always "$NEWNAME.deb"
```

## 2.5 POSTINST

Een debian postinst script wordt uitgevoerd nadat het programma geïnstalleerd is op de schijf. Bij wijze van voorbeeld: Dit postinst script maakt op de desktop van elke gebruiker in /home een shortcut aan naar het programma. Helemaal niet volgens de Debian policy, maar het gaat hier om een voorbeeld.



**Hou er mee rekening dat deze installatiescripts steeds als "root" gebruiker gebeuren!**

```
#!/bin/sh
set -e

#DEBHELPER#
case "$1" in
    configure)
        # copy icon on desktop
        for DESKTOPDIR in `find /home/* -maxdepth 1 -iname "Desktop"`
        do
            cp /usr/share/applications/pakket.desktop $DESKTOPDIR
            chmod 777 $DESKTOPDIR/pakket.desktop
        done
        # Zorgen dat het ikoontje in de cache komt
        if [ -x /usr/bin/update-icon-caches ]; then
            update-icon-caches /usr/share/pixmaps
        fi
        # Zorgen dat het .desktop bestand bekend is in het systeem
        if [ -x /usr/bin/update-desktop-database ]; then
            update-desktop-database /usr/share/applications
        fi
        # Update van de menu structuur
        if [ "$1" = "configure" ] && [ -x "`which update-menus 2>/dev/null`" ];
    then
        update-menus
    fi
    ;;
esac
```

### 3 RICH MAN'S DEB

Uitgaande van een eenvoudig bash programma, bouwen we hier een eerste echte deb (vanuit "source"). Er zijn ruwweg twee manieren waarop je deb pakketten kan genereren vanuit source. De eerste manier is met *cdb*s en de tweede manier met *debhelper*. Je kan ook beiden combineren. Mogelijk kom je ook informatie tegen over *quilt*. Quilt is een systeem van patches, waardoor debian packages worden aangepast aan ubuntu.

Hier gebruiken we debhelper. Als je inspiratie nodig hebt om pakketten te genereren dan kan je ook de source binnenhalen van een bestaand deb pakket. Dat kan bv voor het pakket xubuntu-wallpapers met volgend commando:

```
apt-get source xubuntu-wallpapers
```

Als je pakketten wilt handtekenen, dan maak je best een account aan op Launchpad. De procedure hier werkt zonder handtekening.

#### 3.1 Opzetten email en username

OEF

Run volgend script. Het script zet in je homedirectory in .bashrc de variabelen die door volgende scripts zullen gebruikt worden:

```
#!/bin/bash
cat >> ~/.bashrc <<EOF
DEBEMAIL="jan.celis@kdg.be"
DEBFULLNAME="Jan Celis"
export DEBEMAIL DEBFULLNAME
EOF
```

Log uit je terminal en log terug in zodat .bashrc wordt uitgevoerd. Nakijken of dat is gelukt kan met het commando:

```
env | grep "DEB"
DEBEMAIL=jan.celis@kdg.be
DEBFULLNAME=Jan Celis
```

## 3.2 Maken van de executable

OEF

Installeer volgende pakketten:

apt-get install dh-make debhelper fakeroot devscripts zenity menu

Draai het volgende script in je homedirectory als gewone gebruiker:

```
#!/bin/bash
# Create my first debian package
package="pakket"
version="1"
mkdir -p "${package}/${package}-${version}"
cd "${package}/${package}-${version}"
cat > ${package} <<EOF
#!/bin/sh
# (C) 2011 Uname Zenity, GPL3+
zenity --warning --text="\`exec uname -a\`"
EOF
chmod 755 $package
cd ..
tar -cvzf ${package}-${version}.tar.gz ${package}-${version}
```

## 3.3 dh\_make

We gaan nu de structuur laten opbouwen met alle informatie die nodig is om het pakket te compileren. Dat doen we met `dh_make`. Het tooltje `dh_make` heeft standaard een aantal templates voor software pakketten (single binary, indep binary, multiple binary, library, kernel module, kernel patch). Voor ons volstaat nu een single binary.

OEF

```
cd pakket/pakket-1
pakket-1$ dh_make --native

Type of package: single binary, indep binary, multiple binary, library, kernel
module, kernel patch?
[s/i/m/l/k/n] s

Maintainer name   : Jan Celis
Email-Address      : jan.celis@kdg.be
Date              : Mon, 27 Mar 2017 17:28:05 +0200
Package Name      : pakket
Version           : 1
License           : gpl3
Type of Package    : single
Hit <enter> to confirm:
Currently there is no top level Makefile. This may require additional tuning.
Done. Please edit the files in the debian/ subdirectory now. You should also
check that the pakket-0.1 Makefiles install into $DESTDIR and not in / .
```

Maak je niet ongerust over de Makefile foutmelding. Standaard gaat debhelper er van uit dat we een c/c++ programma gaan compileren, maar het is geen probleem als we dat niet zullen doen.

Het programma `dh_make` maakte een debian directory aan.

### 3.4 control

Het debian/control bestand is één van de belangrijkste bestanden voor het genereren van een pakket. Opgelet! Verwijder niet de lege lijn in dit bestand.

De lijn met Homepage mag je verwijderen

#### 3.4.1 Section

Geldige Sections zijn:

admin, cli-mono, comm, database, debug, devel, doc, editors, education, electronics, embedded, fonts, games, gnome, gnu-r, gnustep, graphics, hamradio, haskell, httpd, interpreters, introspection, java, kde, kernel, libdevel, libs, lisp, localization, mail, math, metapackages, misc, net, news, ocaml, oldlibs, otherosfs, perl, php, python, ruby, science, shells, sound, tasks, tex, text, utils, vcs, video, web, x11, xfce, zope

**OEF** Kies hier bv Section: misc

#### 3.4.2 Dependencies

Dependencies worden gescheiden door een komma. Ze kunnen ook tussen haakjes een versie of minimum versie bevatten. bv

```
Build-Depends: debhelper (>= 9), python (>=2.4)
```

##### Build-Depends

De build dependencies zijn programma's die je nodig hebt om het programma te compileren.

Je mag bij Build-Depends `${shlibs:Depends}` verwijderen.

Typische invullingen voor Build-Depends zijn bv python, java, mono of perl afhankelijk van het programma dat nodig is om alles te compileren

##### Depends

De Depends zijn programma's die nodig zijn om jou gecompileerde programma te draaien  
Voeg hier zenity en menu toe

#### 3.4.3 Description

De Description moet je invullen. De eerste lijn geeft KORT weer wat je programma doet. Bij Ubuntu Software Center wordt deze lijn getoond, wanneer je de deb wilt installeren.

**OEF** De volgende lijnen zijn een Extended description (uitgebreide omschrijving). Deze beginnen met een spatie. Je moet zelf een regeleinde voorzien (er is geen automatische lijn wrap)

Als je een volgende paragraaf wil starten schrijf je gewoon spatie punt.

### 3.4.4 Architecture

Meest gebruikte architectures zijn i386 en amd64. Amd64 is ook voor x86\_64 bit intel.

Alle mogelijke architectures zijn:

[alpha] [amd64] [arm] [armel] [armhf] [avr32] [hppa] [hurd-i386] [i386] [ia64] [kfreebsd-i386] [kfreebsd-amd64] [m68k] [mips] [mipsel] [powerpc] [powerpcspe] [s390] [s390x] [sh4] [sparc] [sparc64]

Opmerking: Architecture **any** betekent compileren voor elk platform en **all** betekent één pakket voor alle platformen



Verander bij Architecture: all

## 3.5 changelog

De changelog houdt alle wijzigingen bij. Sommige IDE's voegen zelf changelog items toe. Opgelet de distribution "unstable" is niet toegelaten op een Ubuntu systeem.

**package:** Naam van je pakket

**version:** Versienummer (update van je pakket krijgt hogere nummer)

**distribution:** Eén van de opties uit dit bestand:

/usr/share/lintian/vendors/ubuntu/main/data/changes-file/known-dists

**urgency:** low, medium, high of critical

Algemene vorm van een changelog:

```
package (version) distribution; urgency=urgency

* change details
- more change details
* even more change details

-- maintainer name <email address>[two spaces]  date
```

Per keer dat je een wijziging aan de code doorvoert, gebruik je een sterretje \* en een streepje voor subveranderingen

Ubuntu en Debian gebruiken lichtjes afwijkende versiesystemen om te vermijden dat pakketten vanuit dezelfde source conflicten kunnen geven.

Wanneer een Debian pakket wordt aangepast in Ubuntu, krijgt het ubuntuX als achtervoegsel bij de Debian versie (X is de Ubuntu revisie nummer)

Voorbeeld:

Een voor Ubuntu aangepast Debian pakket "hello\_2.6-1" wordt in ubuntu:

- hello\_2.6-1ubuntu1
- de Debian versie en revisie blijft behouden

Een nieuw pakket bij Ubuntu, zonder dat er een Debian versie voor bestaat wordt:

- pakket\_2.6-0ubuntu1
- de Debian revisie staat dan op 0

Voorbeeld changelog:

```
pakket (1) xenial; urgency=medium

* Initial Release.

-- Jan Celis <jan.celis@kdg.be> Thu, 15 Dec 2016 16:23:44 +0100
```

OEF

Via de commandline kan je het programma "dch" gebruiken. Voorbeeld:

```
dch --create
dch -a "Icon toegevoegd /usr/share/pixmap/pakket.xpm"
```

## 3.6 manpage

mv manpage.1.ex pakket.1

OEF

Maak met gmanedit een nieuwe manpage aan. Bewaar deze als pakket.1  
Voorbeeld pakket.1 manpage:

```
.\ "Created with GNOME Manpages Editor Wizard
.\ "http://sourceforge.net/projects/gmanedit2
.TH pakket 1 "December 17, 2016" "" "pakket"

.SH NAME
pakket \- program that shows a graphical uname \-a

.SH SYNOPSIS
.B pakket
.RI [ options ]
.br

.SH DESCRIPTION
This manual page explains the
.B pakket
program. This program uses zenity to show a warning message with uname \-a info
.PP
\fBpakket\fP is showing uname \-a info in X11

.SH "SEE ALSO"
zenity(1), uname(1)
AUTHOR
Jan Celis <jan.celis@kdg.be>
```



Maak een nieuw tekstbestand met de naam "manpages". Hierin schrijf je volgende lijn:  
debian/pakket.1

Mogelijke man page secties:

Section	Description	Notes
1	User command	Executable commands or scripts
2	System calls	Functions provided by the kernel
3	Library calls	Functions within system libraries
4	Special files	Usually found in /dev
5	File formats	E.g. /etc/passwd's format
6	Games	Games or other frivolous programs
7	Macro packages	Such as man macros
8	System administration	Programs typically only run by root
9	Kernel routines	Non-standard calls and internals

### 3.7 rules

In de rules file kan je bestaande stappen overrulen.

### 3.8 menu

Je programma MOET opstarten in de bestaande menu structuur.

De hoofdcategoriën zijn:

- Applications
- Games
- Screen (bv voor screen lock)
- Window Managers

Veel gebruikte subcategoriën zijn:

- Applications/Graphics
- Applications/Network/Web Browsing
- Applications/Office
- Applications/Programming
- Applications/Sound
- Applications/System

Je bent verplicht om een subcategory (en soms 2) te kiezen.

Kijk op <http://www.debian.org/doc/packaging-manuals/menu-policy/ch2.html> welke menu secties er mogelijk zijn.

Ofwel kies je voor een menu, ofwel kies je voor een desktop bestand.

Voorbeeld menu instelling:



`mv menu.ex menu`

Pas de inhoud van menu aan:

```
?package(pakket):needs="X11" section="Applications/Education"\
title="pakket" command="/usr/bin/pakket"
```

Hoewel dit de standaard debian manier is om een menu aan te maken, zal je merken dat dit op een Ubuntu systeem geen effect heeft. Ubuntu gebruikt een .desktop bestand.



### 3.9 pakket.desktop

Een .desktop bestand is de manier waarop je applicatie in de menustructuur (of dash bij unity op Ubuntu) verschijnt.

Ze gelden onder andere ook voor Gnome, KDE, Cinnamon en XFCE.

Wat je nodig hebt voor een deb bestand is een vierkant (bv 32x32) ikoontje in /usr/share/pixmap in xpm of png formaat en een .desktop bestand in /usr/share/applications.



Opgelet! Editeer nooit via je file manager (bv nautilus) een .desktop bestand. De file manager interpreteert namelijk het bestand (voor hem eigenlijk een applicatie) terwijl je er in aan het werken bent

Je programma MOET opstarten in de bestaande menu structuur.

Veel gebruikte Categories zijn:

- Applications;Graphics;
- Applications;Network;Web Browsing;
- Applications;Office;
- Applications;Programming;
- Applications;Sound;
- Applications;System;

Voorbeeld van het bestand pakket.desktop:

```
[Desktop Entry]
Name=Pakket
Comment=Pakket met zenity
GenericName=Pakket
X-GNOME-FullName=Pakket
Exec=pakket
Icon=pakket
StartupNotify=false
Terminal=false
Type=Application
Categories=Application;Education;
```

Je ziet dat de Icon geen extensie heeft. Deze icon met extensie xpm of png wordt automatisch gezocht in /usr/share/pixmaps.

Je ziet ook dat er geen pad voor de applicatie staat. Standaard moet elke applicatie beschikbaar zijn in `/usr/bin` of `/usr/sbin` zonder dat iemand een pad moet ingeven.

Je kan bv het bestand `/usr/share/applications/pakket.desktop` uittesten door hier op te dubbelklikken in de filemanager nautilus. Opgelet. De naam van het bestand wordt afgeleid uit het Name veld!

### 3.10 debuild

**OEF**

Verwijder eerst de bestanden die we niet gebruiken:

```
rm *.ex *.EX
```

Compileer het deb pakket met volgend commando:

```
debuild -us -uc
```

- -us betekent unsigned source
- -uc betekent unsigned changes (cfr. changelog)

Normaal kan (moet!) je als developer alles laten signen met jou key (hiervoor moet je een launchpad account hebben, wat overigens gratis is, maar dat skippen we hier even)

#### 3.10.1 install

Het pakket installeert GEEN executable. Om dit te doen maak je het bestand install aan. In install schrijf je welke bestanden je naar waar wil copieren (bron doel). Opgelet

**OEF**

In het bestand install voeg je volgende lijn toe (maak het aan wanneer het nog niet bestaat):

```
pakket usr/bin
```

### 3.11 Build fouten

Alle begin is moeilijk. Lintian ZAL vloeken op sommige bestanden die niet voldoen aan de debian standaard. Na een update van lintian is het ook mogelijk dat bij een bestaand pakket dat in orde was, er toch nieuwe lintian "fouten" opduiken.

Er zijn twee soorten meldingen Errors en Warnings (E: en W:). Errors zorgen ervoor dat het pakket niet kan gecompileerd worden, dus die moet je zeker vermijden.

Een pakket met teveel warnings wordt niet geaccepteerd door de software center.

Je kan het meestal wel installeren via de commandline (dpkg -i pakketnaam.deb).

Vind je dat lintian ongelijk heeft, en dat jou pakket een uitzondering is op de debian policy, dan kan je dat aangeven met een lintian.override bestand.

Voorbeeld:

Wanneer je zelf gebruik wil maken van het programma gksu (dat in het pakket "gksu" staat), dan detecteert lintian dat je een exe gebruikt, die NIET opgenomen is in jou deb.

Je kan een override geven op dat lintian bericht als je zelf een dependency oplegt met het pakket gksu.

**Enkele mogelijke fouten:**

E: pakket-0.1: helper-templates-in-copyright

OPLOSSING: Bestand copyright: De eerste invullijnen verwijderen

E: pakket-0.1: copyright-contains-dh\_make-todo-boilerplate

OPLOSSING: Bestand copyright: verwijder #Please also... lijnen onderaan

W: pakket-0.1 source: out-of-date-standards-version 3.9.6 (current is 3.9.7)

OPLOSSING: Bestand control: Pas de versie aan.

VOORBEELD: Standards-Version: **3.9.7**

W: pakket-0.1: readme-debian-contains-debmake-template

OPLOSSING: **rm README.Debian** # niet nodig dus je mag deze verwijderen  
je mag ook README.source verwijderen

W: pakket-0.1: binary-without-manpage usr/bin/pakket

OPLOSSING: Bestand manpages aanmaken en manpage voor pakket toevoegen

VOORBEELD: In manpages deze lijn toevoegen:  
**debian/pakket.1**

E: Missing Dependency cdb

Je hebt gebruik gemaakt van cdb en dit niet aangegeven bij Build-Depends.

OPLOSSING: Bestand control voeg cdb toe

VOORBEELD: Build-Depends: debhelper (>= 9.0.0), **cdb**

E: pakket-0.1 changes: bad-distribution-in-changes-file unstable

OPLOSSING: Bestand changelog: Pas de distributie aan. Kijk in:  
/usr/share/lintian/vendors/ubuntu/main/data/changes-file/known-dists

VOORBEELD: pakket-0.1 (1) **xenial**; urgency=high

## 4 SIGNED DEBS

Debs die een handtekening dragen van de maker, geven de zekerheid dat de deb "echt" is en dus niet aangepast door iemand anders.

De handtekening van een deb gebeurt door het genereren van een publiek/ privaat sleutelpaar. Door de publieke sleutel ter beschikking te stellen van gebruikers, kunnen deze nakijken of het pakket niet aangepast werd.

In de praktijk zijn alle standaard Ubuntu en Debian repositories gesigned. Je krijgt een waarschuwing wanneer je een gehandtekend pakket probeert te installeren, waarvan je de publieke sleutel niet kent.

### 4.1 Aanmaak van een sleutelpaar

De eerste stap bij het handtekenen van een deb is het genereren van je eigen sleutelpaar. Dat kan met het programma gpg. Belangrijk is dat je dit doet met dezelfde gebruiker als waarmee je de pakketten genereert (dus niet root!)

```
jancelis@kdguntu:~$ gpg --gen-key
gpg (GnuPG) 1.4.20
Copyright (C) 2015 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory `/home/jancelis/.gnupg' created
gpg: new configuration file `/home/jancelis/.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/jancelis/.gnupg/gpg.conf' are not yet active
during this run
gpg: keyring `/home/jancelis/.gnupg/secring.gpg' created
gpg: keyring `/home/jancelis/.gnupg/pubring.gpg' created
Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
Please specify how long the key should be valid.
    0 = key does not expire
    <n> = key expires in n days
    <n>w = key expires in n weeks
    <n>m = key expires in n months
    <n>y = key expires in n years
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y

You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Jan Celis
Email address: kdguntu@gmail.com
Comment: KdGuntu
You selected this USER-ID:
```

```
"Jan Celis (KdGuntu) <kdguntu@gmail.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.
supersecretdeesradenooitonsbommaeetnesjokkotof
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

Not enough random bytes available. Please do some other work to give
the OS a chance to collect more entropy! (Need 279 more bytes)
.....+++++
.....+++++
gpg: /home/jancelis/.gnupg/trustdb.gpg: trustdb created
gpg: key ABCBDA36 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u
pub 2048R/ABCBDA36 2016-08-25
    Key fingerprint = 8F7B 6F55 E5BA EC0B 61CD C573 5D11 1B03 ABCB DA36
uid                               Jan Celis (KdGuntu) <kdguntu@gmail.com>
sub 2048R/C0AD17AC 2016-08-25
```

Op de volgende manier kan je een "leesbare" ASCII versie maken van de publieke sleutel. Deze mag je online ter beschikking stellen van je gebruikers

```
jancelis@kdguntu:~$ gpg --armor --export kdguntu@gmail.com > jan_public.key
```

## 4.2 Handtekenen van een bestaande deb

Voorwaarde voor het handtekenen van een deb is dat je eigen naam aangegeven is in de changelog. Met het pakket dpkg-sig kan je bestaande debs handtekenen. Installeren kan als volgt.

```
jancelis@kdguntu:~$ sudo apt-get install dpkg-sig
```

Daarna kan je volgend commando gebruiken om een deb te handtekenen. Hierbij pas je uiteraard de naam van de deb aan. De optie "builder" gebruikt jou handtekening.

```
jancelis@kdguntu:~$ dpkg-sig --sign builder mijnpakket_0.1.-all.deb
```


## 4.3 Handtekenen van een deb met debuild

Standaard zal debuild (zie hoofdstuk 3) zonder opties, het pakket handtekenen met je gpg sleutel.

```
debuild
```

## 5 OEFENING

### 5.1 Oefening 1 Volg hoofdstuk 3

Voer eerst in het hoofdstuk hiervoor alle paragrafen uit die gemarkeerd worden door het volgende ikoon: 

Als het commando `debuild` lukt zonder foutmeldingen, installeer dan het pakket en test uit of het werkt. Hiervoor start je een terminal op en geef je het commando pakket

### 5.2 Oefening 2 Opstarten met een .desktop bestand

Maak een desktop bestand en zorg ervoor dat je het via de menu of dash kan opstarten met je eigen ikoon.

Je zal in het bestand install zowel je ikoon als je .desktop bestand moeten toevoegen.

### 5.3 Oefening 3 Pakket voor python

Maak vanuit source een deb pakket waarmee je onderstaand python script via de menu kan opstarten.

Het desktop bestand ziet er ongeveer als volgt uit:

```
[Desktop Entry]
Name=ServiceMenu
GenericName=ServiceMenu
Comment=Beheertool voor servers
Type=Application
Exec=gksu "python /usr/lib/servicemenu/service_menu.py"
Icon=service
```

Het bestand `service_menu.py`:

```
#!/usr/bin/env python

from Tkinter import *
from time import *
import subprocess
import tkMessageBox
import os

class ServiceMenu:
    # Daemon Dictionary
    # User to set the daemons that are used for each service.
    dd = {'FTP':'vsftpd',
          'WEB':'apache2'}

    # Create the GUI and check if script is run as root.
    def __init__(self, root):
        self.parent = root
        self.parent.resizable(0, 0)

    # Container
    self.container = Frame(self.parent, padx = 5, pady = 5)
    self.container.grid()
```

```

# Row 1
self.lblFtp = Label(self.container, text = "FTP server:", anchor = W, width = 12)
self.lblFtp.grid(row = 0)
self.lblFtpStatus = Label(self.container, text = "Checking...", width = 10)
self.lblFtpStatus.grid(row = 0, column = 1)
self.btnFtp = Button(self.container, text = "Wait", state = DISABLED, width = 8,
command = lambda: self.setService("FTP"))
self.btnFtp.grid(row = 0, column = 2, padx = 5)

#Row 2
self.lblWeb = Label(self.container, text = "Web server:", anchor = W, width = 12)
self.lblWeb.grid(row = 1)
self.lblWebStatus = Label(self.container, text = "Checking...", width = 10)
self.lblWebStatus.grid(row = 1, column = 1)
self.btnWeb = Button(self.container, text = "Wait", state = DISABLED, width = 8,
command = lambda: self.setService("WEB"))
self.btnWeb.grid(row = 1, column = 2, padx = 5, pady = 3)

# Start/Stop
self.frmAll = Frame(self.container)
self.frmAll.grid(columnspan = 3, row = 2)
self.btnStartAll = Button(self.frmAll, text = "Start All", width = 10, command =
lambda: self.allStartStop("START"), state = DISABLED)
self.btnStartAll.grid(row = 0)
self.btnStopAll = Button(self.frmAll, text = "Stop All", width = 10, command = lambda:
self.allStartStop("STOP"), state = DISABLED)
self.btnStopAll.grid(row = 0, column = 1, pady = 3)

# Initial refresh
self.refreshAll()
self.btnStartAll.config(state = NORMAL)
self.btnStopAll.config(state = NORMAL)

# Check if $USER is root
if os.getenv('USER') != 'root':
    tkMessageBox.showerror('Error', 'Script must run as root', parent = self.parent)
    root.destroy()

# Used to start a stopped service or stop a running service.
def setService(self, service):
    if service == 'FTP':
        status = self.getServiceStatus(service)
        # If running...
        if status:
            self.startStopService(service, 'STOP')
        # If stopped...
        else:
            self.startStopService(service, 'START')

    elif service == 'WEB':
        status = self.getServiceStatus(service)
        # If running...
        if status:
            self.startStopService(service, 'STOP')
        # If stopped...
        else:
            self.startStopService(service, 'START')

# Updates the interface according to the service status.
def checkServiceStatus(self, service):
    if service == 'FTP':
        status = self.getServiceStatus(service)
        # If running...
        if status:
            self.lblFtpStatus.config(text = "Running", bg = "green")
            self.btnFtp.config(state = NORMAL, text = "Stop")
        # If stopped...
        else:
            self.lblFtpStatus.config(text = "Stopped", bg = "red")
            self.btnFtp.config(state = NORMAL, text = "Start")

    elif service == 'WEB':

```



```

        status = self.getServiceStatus(service)
        # If running...
        if status:
            self.lblWebStatus.config(text = "Running", bg = "green")
            self.btnWeb.config(state = NORMAL, text = "Stop")
        # If stopped...
        else:
            self.lblWebStatus.config(text = "Stopped", bg = "red")
            self.btnWeb.config(state = NORMAL, text = "Start")

    # Returns True if a service is running or False if it is not.
    def getServiceStatus(self, service):
        output = subprocess.Popen('ps aux |grep "/" + self.dd[service] + "' |grep -v grep',
        shell = True, stdout = subprocess.PIPE).communicate()[0]
        if output != '':
            status = True
        else:
            status = False
        return status

    # Start or stop a specified service.
    def startStopService(self, service, action):
        output = subprocess.Popen(['service', self.dd[service], action.lower()], shell =
        False, stdout = subprocess.PIPE).communicate()[0]
        sleep(1)
        self.checkServiceStatus(service)
        self.parent.update_idletasks()

    # Start or stop all services.
    def allStartStop(self, action):
        if action == 'START':
            if not self.getServiceStatus('FTP'):
                self.startStopService('FTP', 'START')
            if not self.getServiceStatus('WEB'):
                self.startStopService('WEB', 'START')
        else:
            if self.getServiceStatus('FTP'):
                self.startStopService('FTP', 'STOP')
            if self.getServiceStatus('WEB'):
                self.startStopService('WEB', 'STOP')

    # Update the interface for all services.
    def refreshAll(self):
        self.checkServiceStatus("FTP")
        self.checkServiceStatus("WEB")

root = Tk()
root.title('Service Menu')
app = ServiceMenu(root)
root.mainloop()

```

## 6 REFERENCES RTFM

1. Debian Policy (de bijbel)  
<http://www.debian.org/doc/debian-policy/>
2. Debian Packaging tutorial  
<https://www.debian.org/doc/manuals/packaging-tutorial/packaging-tutorial.en.pdf>
3. Howto package files for your PPA  
<http://ubuntuforums.org/showthread.php?p=5862233>
4. Intro Debian Packaging - Debian wiki,  
<https://wiki.debian.org/Packaging/>
5. Ubuntu Packaging Guide  
<http://packaging.ubuntu.com/html/>

License: <http://creativecommons.org/licenses/by-nc-sa/3.0/> jan.celis@kdg.be