Debian Post-install

Sergio Alvariño salvari@gmail.com

julio-2016

Resumen

Instalación de Debian
Solo para referencia rápida y personal.

Índice general

1 Introducción

Mi portátil es un ordenador Acer 5755G con las siguientes características:

- Core i5 2430M 2.4GHz
- NVIDIA Geforce GT 540M
- 8Gb RAM
- 750Gb HD

La gráfica es una Nvidia Optimus, es decir una tarjeta híbrida que funciona perfectamente en Ubuntu 14.04 usando Bumblebee.

Para hacer la actualización del sistema opté por desinstalar el dvd y montar en su lugar un disco SSD en un Caddie para Acer. La instalación fué muy fácil, y aunque el portátil arranca perfectamente de cualquiera de los dos discos opté por instalar el SSD en la bahía del HD original y pasar el HD al caddie.

Comentar los problemas con calentamiento en Ubuntu

Comentar la creación de usb bootable

Lo primero fue la instalación del Bumblebee

firmware-linux-nonfree Bumblebee-nvidia primus

2 Gestión de paquetes

Instalamos *aptitude*, *synaptic* y *gdebi*

```
sudo apt-get install aptitude
sudo apt-get install synaptic
sudo apt-get install gdebi
```

Cambiamos las opciones de *aptitude* para que **no instale** los paquetes recomendados.

2.1 Quitamos el cdrom de los sources.list

Editamos el fichero */etc/apt/sources.list* y comentamos las lineas del cdrom.

2.2 Habilitamos los backports y multimedia

2.2.1 Backports:

```
sudo cat > /etc/apt/sources.list.d/backports.list << EOF
# backports
deb http://ftp.debian.org/debian/ jessie-backports main contrib non-free
EOF
```

2.2.2 Multimedia:

```
sudo cat >> /etc/apt/sources.list.d/multimedia.list << EOF
```

```
# multimedia
deb http://www.deb-multimedia.org/ jessie main non-free
EOF

sudo apt-get -y --allow-unauthenticated install --reinstall deb-multimedia-keyring
```

Y actualizamos

```
sudo aptitude update
```

3 Instalación de varios paquetes sueltos

3.1 Programas de utilidad y uso frecuente

3.1.1 Menulibre

Un editor de menús para Gnome, nos permite generar los archivos desktop para cualquier aplicación. Mucho más completo que *alacarte* la otra alternativa.

```
sudo apt-get install menulibre
```

3.1.2 Terminator

Terminator es un emulador de terminal muy completo y muy flexible. Los instalamos desde *aptitude*

```
sudo aptitude install terminator python-keybinder
```

3.1.3 Keepass2

Instalado *keepass2* desde Debian

```
sudo aptitude install keepass2
```

3.1.4 gksu

Un *sudo* en modo gráfico:

```
sudo aptitude install gksu
```

3.1.5 Diskmanager

Para gestionar discos portátiles

```
sudo apt-get install ntfs-3g disk-manager
```

3.1.6 Gnucash

Finanzas en linux

```
sudo apt-get -t jessie-backports install gnucash
```

3.1.7 Herramientas *sync*

No sin mis *backups*

```
sudo apt-get install rsync grsync
```

3.1.8 Dropbox

Bajado el paquete Debian desde la página web de Dropbox, instalado el paquete con *packageinstall*, es decir, simplemente pinchando desde el gestor de ficheros.

3.1.9 Compresores et al

```
sudo apt-get install rar unrar zip unzip unace bzip2 lzop p7zip p7zip-full p7zip-rar
```

3.2 Internet

3.2.1 Chrome

Instalado chrome añadiendo fuentes a aptitude. No recuerdo como las añadí, en el fichero */etc/apt/sources.list.d/google-chrome.list*, tengo los siguientes contenidos:

```
###
###
###
###
### THIS FILE IS AUTOMATICALLY CONFIGURED ###
```

```
# You may comment out this entry, but any other modifications may be lost.
deb [arch=amd64] http://dl.google.com/linux/chrome/deb/ stable main
```

Ejecutamos:

```
sudo aptitude install google-chrome-stable
sudo aptitude install chromium
```

3.2.2 Tor

Bajado el comprimido desde la web y descomprimido en *~/apps* copiado el fichero desktop a *~/.local/share/applications*

3.2.3 Deluge

Instalamos desde aptitude

```
sudo aptitude install deluge
xdg-mime default deluge.desktop x-scheme-handler/magnet
```

3.2.4 TiddlyDesktop

Tiddly es una wiki auto-contenida y muy flexible, tiene un sinfin de versiones adaptadas para diferentes usos. Hace años que la uso como cuaderno de bitácora personal, pero no había seguido su evolución.

Me he descargado:

- Tiddlywiki y le he instalado los plugins de *FontAwesome* y *WikiMap*, este será mi nuevo cuaderno de bitácora.
- GSD5 un *TiddlyWiki* adaptado a *GTD*

A mayores me he instalado la aplicación TiddlyDesktop, basada en *node webkit* que simplifica el tema de backups (en teoría).

Como siempre la instalamos en *~/apps* y creamos un lanzador con *MenuLibre*.

3.3 Gráficos

3.3.1 Inkscape

```
apt-cache policy inkscape
apt-get -t jessie-backports install inkscape
aptitude install ink-generator
```

3.3.2 LibreCAD y FreeCAD

Instalado desde repos con aptitude

```
apt-get install librecad
```

```
apt-get -t jessie-backports install freecad
```

3.3.3 Gimp

Gimp ya estaba instalado, adicionalmente instalado el gimp data-extra

```
sudo aptitude install gimp-plugin-registry gimp-texturize gimp-data-extras gimp-gap
```

3.4 Fotografía

3.4.1 Rawtherapee y Darktable: Tratamiento de imágenes fotográficas

```
sudo aptitude install icc-profiles icc-profiles-free
sudo aptitude install rawtherapee darktable
```

3.4.2 Stopmotion

```
sudo aptitude install stopmotion vgrabbj dvgrab
```

TODO: Probar qStopmotion

3.5 Audio y video

3.5.1 Codecs

Instalamos los codecs

```
sudo apt-get install libav-tools

sudo apt-get install faad gstreamer0.10-ffmpeg gstreamer0.10-x \
gstreamer0.10-fluendo-mp3 gstreamer0.10-plugins-base \
gstreamer0.10-plugins-good gstreamer0.10-plugins-bad \
gstreamer0.10-plugins-ugly ffmpeg lame twolame vorbis-tools \
libquicktime2 libfaac0 libmp3lame0 libxine2-all-plugins libdvdread4 \
libdvdnav4 libmad0 sox libxvidcore4 libstdc++5

sudo apt-get install w64codecs
```

3.5.2 Reproductores de música

Instalamos *Clementine*, *decibel*, *audacity*, *soundconverter*:

```
sudo aptitude install clementine gstreamer0.10-plugins-bad
sudo aptitude install decibel-audio-player audacity soundconverter
```

3.5.3 Gpodder

Instalamos *gpodder* para gestionar nuestros podcast, aunque *Clementine* también nos vale.

```
sudo aptitude install gpodder
```

3.5.4 Spotify

Cliente de *Spotify*

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys BBEBDCB318AD50EC6865090613B00F1
echo deb http://repository.spotify.com stable non-free | sudo tee /etc/apt/sources.list.d/spotify.list
sudo aptitude update
sudo aptitude install spotify-client
```

3.5.5 Video

Instalamos también utilidades de video:

```
sudo aptitude install vlc browser-plugin-vlc
sudo aptitude install recordmydesktop gtk-recordmydesktop
sudo aptitude install handbrake handbrake-cli handbrake-gtk
```

4 Documentos

4.1 Calibre

Ejecutamos lo que manda la página web:

```
sudo -v && wget -nv -O- https://raw.githubusercontent.com/kovidgoyal/calibre/master/setup/linux-instal
| sudo python -c "import sys; main=lambda:sys.stderr.write('Download failed\n'); exec(sys.stdin.read()
```

Para usar el calibre con el Kobo Glo:

- Desactivamos todos los plugin de Kobo menos el *Kobo Touch Extended*
- Creamos una columna *MyShelves* con identificativo *#myshelves*
- En las opciones del plugin:
  - En la opción *Collection columns* añadimos las columnas *series,#myshelves*
  - Marcamos las opciones *Create collections* y *Delete empy collections*
  - *Update metadata on device* y *Set series information*

Algunos enlaces útiles:

- https://github.com/jgoguen/calibre-kobo-driver
- http://www.lectoreselectronicos.com/foro/showthread.php?15116-Manual-de-instalaci%C3%B3n-y-uso-del-plugin-Kobo-Touch-Extended-para-Calibre
- http://www.redelijkheid.com/blog/2013/7/25/kobo-glo-ebook-library-management-with-calibre
- https://www.netogram.com/kobo.htm

4.2 Pandoc

Instalado el Pandoc descargando paquete *deb* desde la página web del Pandoc.

Descargamos las plantillas desde el repo ejecutando los siguientes comandos:

```
cd ~/.pandoc
git clone https://github.com/jgm/pandoc-templates templates
```

4.3 Vanilla LaTeX

El LaTeX de Debian está un poquillo anticuado, si se quiere usar una versión reciente hay que aplicar este truco.

```
cd ~
mkdir tmp
cd tmp
wget http://mirror.ctan.org/systems/texlive/tlnet/install-tl-unx.tar.gz
tar xzf install-tl-unx.tar.gz
cd install-tl-xxxxxx
```

La parte xxxxxx varía en función del estado de la última versión de LaTeX disponible.

```
sudo ./install-tl
```

Una vez lanzada la instalación podemos desmarcar las opciones que instalan la documentación y las fuentes. Eso nos obligará a consultar la documentación *on line* pero ahorrará practicamente el 50% del espacio necesario. En mi caso sin *doc* ni *src* ocupa 2,3Gb

```
mkdir -p /opt
sudo ln -s /usr/local/texlive/2016/bin/* /opt/texbin
```

Por último para acabar la instalación añadimos **/opt/texbin** al *path*.

4.3.1 Falsificando paquetes

Ya tenemos el **texlive** instalado, ahora necesitamos que el gestor de paquetes sepa que ya lo tenemos instalado.

```
sudo apt-get install equivs --no-install-recommends
mkdir -p /tmp/tl-equivs && cd /tmp/tl-equivs
equivs-control texlive-local
```

Para hacerlo más fácil podemos descargarnos un fichero ya preparado, ejecutando:

```
wget http://www.tug.org/texlive/files/debian-equivs-2015-ex.txt
/bin/cp -f debian-equivs-2015-ex.txt texlive-local
```

Editamos la versión y

```
equivs-build texlive-local
sudo dpkg -i texlive-local_2015-1_all.deb
```

Todo listo, ahora podemos instalar cualquier paquete que dependa de texlive

4.3.2 Fuentes

Para dejar disponibles las fuentes opentype y truetype que vienen con texlive para el resto de aplicaciones:

```
sudo cp $(kpsewhich -var-value TEXMFSYSVAR)/fonts/conf/texlive-fontconfig.conf /etc/fonts/conf.d/09-te
gksudo gedit /etc/fonts/conf.d/09-texlive.conf
```

Borramos la linea:

```
<dir>/usr/local/texlive/2016/texmf-dist/fonts/type1</dir>
```

Y ejecutamos:

```
sudo fc-cache -fsv
```

4.3.3 Actualizaciones

Para actualizar nuestro latex a la última versión de todos los paquetes:

```
sudo /opt/texbin/tlmgr update --self
sudo /opt/texbin/tlmgr update --all
```

También podemos lanzar el instalador gráfico con:

```
sudo /opt/texbin/tlmgr --gui
```

Para usar el instalador gráfico hay que instalar previamente:

```
sudo apt-get install perl-tk --no-install-recommends
```

4.3.4 Lanzador para el actualizador de texlive

```
mkdir -p ~/.local/share/applications
/bin/rm ~/.local/share/applications/tlmgr.desktop
cat > ~/.local/share/applications/tlmgr.desktop << EOF
[Desktop Entry]
Version=1.0
Name=TeX Live Manager
```

```
Comment=Manage TeX Live packages
GenericName=Package Manager
Exec=gksu -d -S -D "TeX Live Manager" '/opt/texbin/tlmgr -gui'
Terminal=false
Type=Application
Icon=system-software-update
EOF
```

Ojo que hay que dejar instalado el gksu (aunque debería estar de antes si sigues este doc)

```
sudo aptitude install gksu
```

4.4 Emacs

Instalado emacs desde los repos:

```
sudo aptitude install emacs
```

Instalamos los paquetes *markdown-mode*, *mardown-plus* y *pandoc-mode* desde el menú de gestión de paquetes de **emacs**.

También instalamos *d-mde* y *flymake-d*. Hay una sección de configuración en el fichero *.emacs*.

Configuramos el fichero *.emacs* definimos algunas preferencias, algunas funciones útiles y añadimos orígenes extra de paquetes.

```
(custom-set-variables
 ;; custom-set-variables was added by Custom.
 ;; If you edit it by hand, you could mess it up, so be careful.
 ;; Your init file should contain only one such instance.
 ;; If there is more than one, they won't work right.
 '(column-number-mode t)
 '(show-paren-mode t))
(custom-set-faces
 ;; custom-set-faces was added by Custom.
 ;; If you edit it by hand, you could mess it up, so be careful.
 ;; Your init file should contain only one such instance.
 ;; If there is more than one, they won't work right.
 '(default ((t (:family "Mensch" :foundry "bitstream" :slant normal :weight normal :height 128 :width
;;------------------------------------------------------------
;; Some settings
(setq inhibit-startup-message t) ; Eliminate FSF startup msg
(setq frame-title-format "%b")   ; Put filename in titlebar
```

```
;(setq visible-bell t)            ; Flash instead of beep
(set-scroll-bar-mode 'right)     ; Scrollbar placement
(show-paren-mode t)              ; Blinking cursor shows matching parentheses
(setq column-number-mode t)  ; Show column number of current cursor location
(mouse-wheel-mode t)         ; wheel-mouse support

(setq fill-column 78)
(setq auto-fill-mode t)          ; Set line width to 78 columns...

(setq-default indent-tabs-mode nil)       ; Insert spaces instead of tabs
(global-set-key "\r" 'newline-and-indent) ; turn autoindenting on
;(set-default 'truncate-lines t)           ; Truncate lines for all buffers
(require 'iso-transl)

;;-------------------------------------------------------------
;; Some useful key definitions
(define-key global-map [M-S-down-mouse-3] 'imenu)
(global-set-key [C-tab] 'hippie-expand)                   ; expand
(global-set-key [C-kp-subtract] 'undo)                    ; [Undo]
(global-set-key [C-kp-multiply] 'goto-line)               ; goto line
(global-set-key [C-kp-add] 'toggle-truncate-lines)        ; goto line
(global-set-key [C-kp-divide] 'delete-trailing-whitespace) ; delete trailing whitespace
(global-set-key [C-kp-decimal] 'completion-at-point)      ; complete at point
(global-set-key [C-M-prior] 'next-buffer)                 ; next-buffer
(global-set-key [C-M-next] 'previous-buffer)              ; previous-buffer

;;-------------------------------------------------------------
;; Set encoding
(prefer-coding-system 'utf-8)
(setq coding-system-for-read 'utf-8)
(setq coding-system-for-write 'utf-8)

;;-------------------------------------------------------------
;; Maximum colors
(cond ((fboundp 'global-font-lock-mode)  ; Turn on font-lock (syntax highlighting)
       (global-font-lock-mode t)              ; in all modes that support it
       (setq font-lock-maximum-decoration t))) ; Maximum colors

;;-------------------------------------------------------------
;; Use % to match various kinds of brackets...
```

```
;; See: http://www.lifl.fr/~hodique/uploads/Perso/patches.el

(global-set-key "%" 'match-paren)               ; % key match parents
(defun match-paren (arg)
  "Go to the matching paren if on a paren; otherwise insert %."
  (interactive "p")
  (let ((prev-char (char-to-string (preceding-char)))
        (next-char (char-to-string (following-char))))
    (cond ((string-match "[[{(<]" next-char) (forward-sexp 1))
          ((string-match "[\]})>]" prev-char) (backward-sexp 1))
          (t (self-insert-command (or arg 1))))))

;;------------------------------------------------------------
;; The wonderful bubble-buffer
(defvar LIMIT 1)
(defvar time 0)
(defvar mylist nil)

(defun time-now ()
   (car (cdr (current-time))))

(defun bubble-buffer ()
   (interactive)
   (if (or (> (- (time-now) time) LIMIT) (null mylist))
       (progn (setq mylist (copy-alist (buffer-list)))
          (delq (get-buffer " *Minibuf-0*") mylist)
          (delq (get-buffer " *Minibuf-1*") mylist)))
   (bury-buffer (car mylist))
   (setq mylist (cdr mylist))
   (setq newtop (car mylist))
   (switch-to-buffer (car mylist))
   (setq rest (cdr (copy-alist mylist)))
   (while rest
     (bury-buffer (car rest))
     (setq rest (cdr rest)))
   (setq time (time-now)))

(global-set-key [f8] 'bubble-buffer)    ; win-tab switch the buffer

(defun geosoft-kill-buffer ()
```

```
   ;; Kill default buffer without the extra emacs questions
   (interactive)
   (kill-buffer (buffer-name))
   (set-name))
(global-set-key [C-delete] 'geosoft-kill-buffer)

;;-------------------------------------------------------------------
;; MELPA and others
(when (>= emacs-major-version 24)
  (require 'package)
  (package-initialize)
  (add-to-list 'package-archives '("melpa" . "http://melpa.org/packages/") t)
  (add-to-list 'package-archives '("gnu" . "http://elpa.gnu.org/packages/") t)
  (add-to-list 'package-archives '("marmalade" . "https://marmalade-repo.org/packages/") t)
  )

;;-------------------------------------------------------------------
;; flymake installed from package

(require 'flymake)
(global-set-key (kbd "C-c d") 'flymake-display-err-menu-for-current-line)
(global-set-key (kbd "C-c n") 'flymake-goto-next-error)
(global-set-key (kbd "C-c p") 'flymake-goto-prev-error)

;; Activate flymake for D
(add-hook 'd-mode-hook 'flymake-d-load)
```

4.5 Scribus

Instalado con aptitude

```
sudo aptitude install scribus
```

4.6 Comix

Instalado con aptitude

```
sudo aptitude install comix
```

5 Desarrollo sw

5.1 Git

Instalado git desde aptitude

```
sudo aptitude install git
```

Configuración básica de **git**

```
git config --global user.name "Sergio Alvariño"
git config --global user.email "salvari@gmail.com"
git config --global core.editor emacs
git config --global color.ui true
git config --global credential.helper cache
git config --global credential.helper 'cache --timeout=7200'
git config --global push.default simple
git config --global alias.sla 'log --oneline --decorate --graph --all'
git config --global alias.car 'commit --amend --no-edit'
git config --global alias.unstage reset
git config --global alias.st status
git config --global alias.last  'log -1 HEAD'
git config --global alias.ca 'commit -a'
```

5.2 Paquetes esenciales

```
sudo apt-get install build-essential checkinstall make automake cmake autoconf git git-core dpkg wget
```

5.3 Open Java

```
apt-get install openjdk-7-jre icedtea-7-plugin
```

5.4 D-apt e instalación de programas

configurado d-apt, instalados todos los programas incluidos

```
sudo wget http://master.dl.sourceforge.net/project/d-apt/files/d-apt.list -O /etc/apt/sources.list.d/d
sudo apt-get update && sudo apt-get -y --allow-unauthenticated install --reinstall d-apt-keyring && su
```

Instalamos todos los programas asociados.

5.5 DCD

Configuración de DCD

5.6 Emacs para editar D

Instalados los siguientes paquetes desde *marmalade*

- *d-mode*
- *flymake-d*

Se configura en el fichero **~/.emacs**:

```
(require 'flymake)
(global-set-key (kbd "C-c d") 'flymake-display-err-menu-for-current-line)
(global-set-key (kbd "C-c n") 'flymake-goto-next-error)
(global-set-key (kbd "C-c p") 'flymake-goto-prev-error)

;; Activate flymake for D
(add-hook 'd-mode-hook 'flymake-d-load)
```

5.7 Processing

Bajamos los paquetes de las respectivas páginas web, descomprimimimos en *~/apps/* y creamos los desktop file con **Menulibre**

5.8 Openframeworks

Bajamos el paquete comprimido de la página web del proyecto.

Descomprimimos en *~/apps*

Bajamos al directorio de la aplicación y ejecutamos:

```
sudo  scripts/linux/debian/install_dependencies.sh
sudo  scripts/linux/debian/install_codecs.sh

cd scripts/linux
./compileOF.sh -j2

cd OF/examples/graphics/polygonExample
```

```
make
make Run

cd OF/scripts/linux
./compilePG.sh
```

Va a instalar un montón de dependencias, hay que tomarlo con calma.

Al final también va a añadir una linea al fichero *~/.profile*

```
export PG_OF_PATH=/home/salvari/apps/of/of_v0.9.3_linux64_release
```

5.9 Python

De partida tenemos instalado dos versiones: *python* y *python3*

```
python -V
Python 2.7.9

python3 -V
Python 3.4.2
```

Instalado python-pip y python-virtualenv desde aptitude.

```
sudo aptitude install python-pip python-virtualenv
```

Instalamos a mayores *Ananconda*, es la forma fácil de poder usar *ipython notebook*. De hecho me he instalado dos versiones la que incluye el python2 y la que incluye el python3.

Las instalaciones de *Anaconda* son redundantes, basta con instalar uno de ellos. En cualquier caso para realizar la instalación basta con descargar los scripts de instalación desde la página web de Anaconda.

```
bash Anaconda3-4.2.0-Linux-x86_64.sh
bash Anaconda2-4.2.0-Linux-x86_64.sh
```

Los he dejado instalados en *~/apps/anaconda2* y *~/apps/anaconda3*

Cada una de estas instalaciones incorpora su propia versión de Python. Para usarlas tenemos que cambiar nuestro PATH para que el Python deseado sea el primero que se selecciona.

Por ejemplo para activar anaconda3 en bash:

```
export PATH="~/apps/anaconda3/bin:$PATH"
```

Para hacer lo mismo en fish:

```
set -x PATH ~/apps/anaconda3/bin $PATH
```

5.9.1 iPython y GraphLab

Creamos un entorno conda con Python 2.7.x

```
conda create -n gl-env python=2.7 anaconda
```

Activamos el nuevo entorno (todo esto lo hice en bash, en fish hay un problemilla con el entorno conda mas info)

```
bash
source activate gl-env
```

En el futuro esto es todo lo que tendremos que hacer activar el entorno conda donde estamos instalando el iPython.

Nos aseguramos de tener *pip* al dia:

```
conda update pip
```

Instalamos la biblioteca GraphLab Create. Esta biblioteca se supone que es fácil de usar pero está sujeta a licencia. [1]

Una vez registrado en la página web te pasan un número de registro que tienes que usar para instalar la biblioteca.

```
pip install --upgrade --no-cache-dir https://get.graphlab.com/GraphLab-Create/2.1/your registered emai
```

Y para terminar instalamos iPython [2]:

```
conda install ipython-notebook
```

Desde ahora basta con activar el entorno que hemos creado para tener acceso al iPython.

```
source activate gl-env

ipython notebook

source deactivate gl-env
```

5.9.1.1 Instalación alternativa con virtualenv

[1] TODO: Pasarme a *scikit-learn*
[2] TODO: conda install jupyter

```
# Create a virtual environment named e.g. gl-env
virtualenv gl-env

# Activate the virtual environment
source gl-env/bin/activate

# Make sure pip is up to date
pip install --upgrade pip

# Install IPython Notebook (optional)
pip install "ipython[notebook]"

# Install Jupyter Notebook (optional)
pip install "jupyter"


# Install your licensed copy of GraphLab Create
pip install --upgrade --no-cache-dir https://get.graphlab.com/GraphLab-Create/2.1/your registered emai
```

5.9.2 Usar Emacs para editar Python

Instalamos *elpy* desde el gestor de paquetes de Emacs, concretamente desde el repo *marmalade*

Hay que habilitar *elpy* en el fichero **~/.emacs** para ello añadimos la linea

```
(elpy enable)
```

5.9.2.1 TODO

Estudiar esto con calma https://elpy.readthedocs.io/en/latest

6 Desarrollo hardware

6.1 Arduino IDE

Bajamos los paquetes de la página web , descomprimimimos en *~/apps/arduino*.

Creamos un link al directorio del software que hemos descargado:

```
cd ~/apps/arduino
ln -s arduino-x.y.z current
```

La primera ves que instalamos será necesario crear el desktop file con **Menulibre** con las actulizaciones no será necesario, siempre y cuando apunte a *~/apps/arduino/current*

6.2 Pinguino IDE

Tenemos el paquete de instalación disponible en su página web

Ejecutamos el programa de instalación. El programa descargará los paquetes Debian necesarios para dejar el IDE y los compiladores instalados.

Al acabar la instalación he tenido que crear el directorio *~/Pinguino/v11*, parece que hay algún problema con el programa de instalación y no lo crea automáticamente.

El programa queda correctamente instalado en */opt* y arranca correctamente, habrá que probarlo con los micros.

6.3 KiCAD

Instalamos desde *backports*:

```
sudo aptitude install -t jessie-backports kicad
```

Vamos a instalar a mayores algunas librerias de KiCAD, para poder crear Shields de Arduino.

- Freetronics una libreria que no solo incluye Shield para Arduino sino una completa colección de componentes que nos permitirá hacer proyectos completos. Freetronics es una especie de BricoGeek australiano, publica tutoriales, vende componentes, y al parecer mantiene una biblioteca para KiCAD. La biblioteca de Freetronics se mantiene en un repo de github. Lo suyo es incorporarla a cada proyecto, por que si la actualizas se pueden romper los proyectos que estes haciendo.
- eklablog Esta biblioteca de componentes está incluida en el github de KiCAD, así que teoricamente no habría que instalarla en nuestro disco duro.

7 Docker

```
apt-get install apt-transport-https ca-certificates
apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys 58118E89F3A912897C070ADBF7622
edit docker.list with
```

```
deb https://apt.dockerproject.org/repo debian-jessie main

apt-cache policy docker-engine   -- comprobamos que todo está bien.


sudo apt-get install docker-engine   -- da un error en makedev por udev activo


sudo service docker start

sudo docker run hello-world   - todo bien

sudo gpasswd -a salvari docker
```

8 Shells alternativos: zsh y fish

Los dos son muy interesantes. He usado zsh casi un año, ahora voy a probar **fish**.

8.1 fish

Instalamos **fish** desde aptitude con:

```
sudo aptitude install fish
```

Instalamos oh-my-fish

```
curl -L https://github.com/oh-my-fish/oh-my-fish/raw/master/bin/install > install
fish install
rm install

chsh -s `which fish`
```

8.2 zsh

Igualmente instalamos **zsh**:

```
sudo aptitude install zsh
```

Vamos a usar antigen así que nos lo clonamos en _~/apps/

```
cd ~/apps
git clone https://github.com/zsh-users/antigen
```

Y editamos el fichero *~/.zshrc* para que contenga:

```
source ~/apps/antigen/antigen.zsh

# Load the oh-my-zsh's library.
antigen use oh-my-zsh

# Bundles from the default repo (robbyrussell's oh-my-zsh).
antigen bundle git
antigen bundle command-not-found
antigen bundle autojump
antigen bundle extract
# antigen bundle heroku
# antigen bundle pip
# antigen bundle lein



# Syntax highlighting bundle.
antigen bundle zsh-users/zsh-syntax-highlighting

# git
antigen bundle arialdomartini/oh-my-git
antigen theme arialdomartini/oh-my-git-themes oppa-lana-style

# autosuggestions
antigen bundle tarruda/zsh-autosuggestions

#antigen theme agnoster

# Tell antigen that you're done.
antigen apply

# append to path
path+=('/home/salvari/apps/julia/current/bin/')
# prepend
# path=('/home/salvari/bin/' $path)
# export PATH
```

Antigen ya se encarga de descargar todo lo que queramos utilizar en zsh.

Nos queda arreglar las fuentes para que funcione correctamente la linea de estado en los repos de git. Necesitamos una fuente *Awesome*

8.3 Instalación de fuentes adicionales

Nos bajamos unas cuantas fuentes que soporten los iconos *Awesome*.

```
cd ~/tmp
git clone https://github.com/abertsch/Menlo-for-Powerline
git clone https://github.com/powerline/fonts

mkdir ~/.fonts
cp someFontFile ~/.fonts/
fc-cache -vf ~/.fonts/
```

9 Cambiar las opciones de idioma

Ejecutamos:

```
sudo dpkg-reconfigure locales
```

Y después solo tenemos que cambiar la selección del idioma en la configuración de Gnome.

Nos pedirá rearrancar Gnome y renombrará todos los directorios de sistema.

10 Reprap

10.1 Sl1c3r

Descargamos el paquete binario desde la página web.

- Cambiar permisos en directorio */lib/vrt/*
- Instalado *lib-canberra-module* desde aptitude
- Es necesario instalar *freeglut*

10.2 OpenScad

Instalado desde aptitude.

10.3 Printrun

Descargamos desde github

10.4 Cura

Descargamos desde la pagina web

```
sudo aptitude install python3-pyqt5
sudo dpkg -i Cura-2.1.3-Linux.deb
```

```
sudo apt-get install python-serial python-wxgtk2.8 python-pyglet python-numpy \
cython python-libxml2 python-gobject python-dbus python-psutil python-cairosvg git

python setup.py build_ext --inplace
```

11 Bases de datos

11.1 MySQL

Instalamos desde aptitude *mysql-server.5.6*

Opcionalmente (y muy recomendable)

```
mysql_secure_instalallation
```

11.1.1 Actualización

Cambiamos el fichero *mysql.conf.d/mysqld.cnf*

```
# max_allowed_packet      = 16M
max_allowed_packet  = 500M
```

Reiniciamos el servicio:

```
/etc/init.d/mysql restart
```

11.2 Cliente SQL SQuirreL SQL

Descargamos el paquete desde la página web y lo descomprimimos en *~/apps*, también tendremos que descargar el conector de mysql para java, por ejemplo desde aquí

Una vez instalado, creamos el desktop-file con *MenuLibre* y configuramos el driver *MySQL* añadiendo el path a donde hayamos dejado el conector java.

12 Cuentas online abiertas

- google
- pocket (plugin de chrome)

13 TODO

- cinelerra
- zotero
- playonlinux
- darktable
- rawtherapee
- krita
- mypaint
- qStopmotion

Inkscape https://elizsarobhasa.makes.org/thimble/MTMwNDIzMjE5Mg==/3d-printing-from-a-2d-drawing Instalar tb jessyink

chibios * http://wiki.chibios.org/dokuwiki/doku.php?id=chibios:community:setup:openocd_chibios * http://www.josho.org/blog/blog/2014/11/30/nucleo-gcc/ * http://www.stevebate.net/chibios-rpi/GettingStarted.html

rclone [https://syncthing.net/]

vmware

sudo aptitude install chromium

14 Links

- Systemd
- Gnome shortcuts
- Gnome optimizaciones
- Instalación Debian
- zsh
- zsh
- https://www.roaringpenguin.com/products/remind
- http://taskwarrior.org/