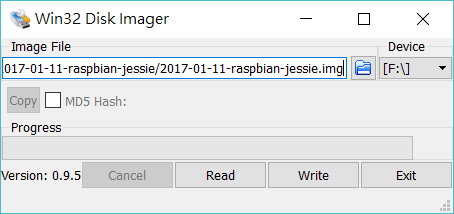
# Format SD card

* Run Win32Discimager.exe
* Select image file : 2017-01-11-raspbian-jessie.img
* Select “Device”
* Select “Write”

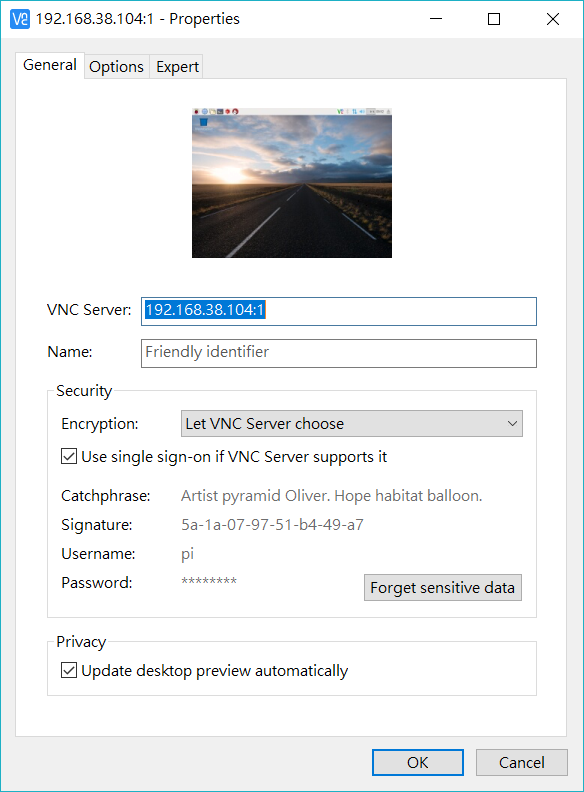


# VNC viewer Setup

1. Install tightvncserver 🡪 sudo apt-get install tightvncserver

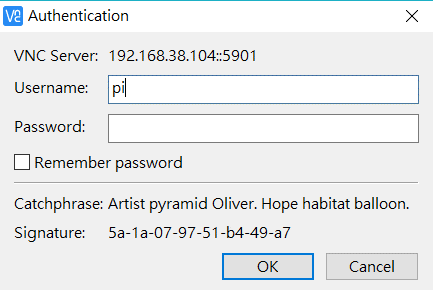
* sudo apt-get install x11vnc

1. Set VNC Server IP : 192.168.38.104:1
2. Name : Any



Username : pi

Password: raspberry



====================================================================

* **如何在開機時就啟動 VNC 伺服器？**
  + sudo nano /etc/init.d/tightvncserver

內容如下，重要的是在 start) 部份設定啟動的參數。

#!/bin/bash

### BEGIN INIT INFO

# Provides: tightvncserver

# Required-Start: $syslog

# Required-Stop: $syslog

# Default-Start: 2 3 4 5

# Default-Stop: 0 1 6

# Short-Description: vnc server

# Description:

### END INIT INFO

export USER='pi'

eval cd ~$USER

# Check the state of the command - this'll either be start or stop

case "$1" in

start)

# if it's start, then start vncserver using the details below

su $USER -c '/usr/bin/vncserver :1 -geometry 800x600 -depth 16 -pixelformat rgb565'

echo "Starting vncserver for $USER "

;;

stop)

# if it's stop, then just kill the process

pkill Xtightvnc

echo "vncserver stopped"

;;

\*)

echo "Usage: /etc/init.d/tightvncserver {start|stop}"

exit 1

;;

esac

exit 0

修改腳本權限。

pi@raspberrypi ~ $ sudo chmod 755 /etc/init.d/tightvncserver

將該腳本加入預設的啟動程序。

pi@raspberrypi ~ $ sudo update-rc.d tightvncserver defaults

重新啟動 Pi，再執行 netstat -nutlp 或是 ps aux | grep vnc，可以看到 VNC 伺服器已經跑起來了。

* 我們可以用 vncpasswd 這個指令改 VNC 的連線密碼

# Raspberry PI(RASPBIAN) ftp伺服器安裝

1. **sudo apt-get install vsftpd  
   2. sudo nano /etc/vsftpd.conf  
   3. 刪除前面的 “#"==> at line local enable =yes  
    刪除前面的 “#"==>at line write enable =yes  
    local\_enable=YES, write\_enable=YES  
    sudo service vsftpd restart 服務重新啟動**

安裝完成後，我們可以來查看是否安裝成功，可用下列指令。  
netstat -tul | grep ftp

**4. 使用ftp客端軟體去連線  
登入者請用pi  
PS:**

1. **port :21  
   sudo nano /etc/ftpusers  
   在root前邊加#,註釋掉它  
   設定root密碼sudo passwd root  
   解除root鎖定sudo passwd –unlock root  
   重開機  
   這樣就可以用root用戶登錄了**

# Install Netbaens

$ mv ~/Downloads/netbeans-8.0.2-201411181905-javase.zip ~/  
$ unzip netbeans-8.0.2-201411181905-javase.zip  
  
然後執行，  
$ netbeans/bin/netbeans

# Install USB serial devices

* **Lsusb to list all USB device**
* **dmesg** prints kernel debug/log information:
* shows me details for a device:

udevadm info -a -n /dev/ttyACM0

* use Minicom Terminal Program

sudo apt-get install minicom

minicom -D /dev/ttyACM0 -b 115200

* use [udev rules](http://www.reactivated.net/writing_udev_rules.html)
  + In /etc/udev/rules.d I create a new file named: 99-usb-serial.rules

(The number 99 is because rules >=90 should run last)

* + To create a dedicated device for the Segger J-Link I can add the following line to the **99-usb-serial.rules**:

SUBSYSTEM=="tty", ATTRS{idVendor}=="04d8", ATTRS{product}=="000a",

ATTRS{manufacturer}==" Microchip Technology Inc.",SYMLINK+="ttyS1"

* + That means that the device will be still used as \dev\ttyACMx, but a symbolic link /dev/ttyUSB50 or /dev/ttyUSB51 will be created.
  + use the following command to test the rules:

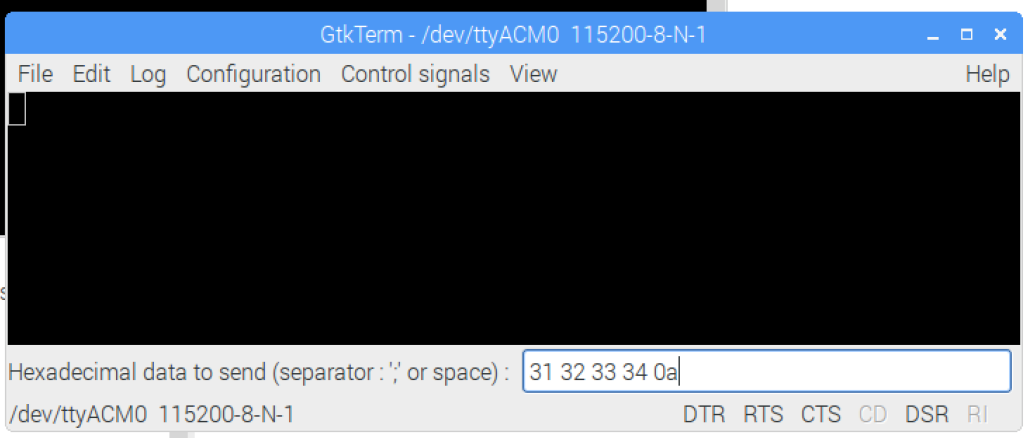
udevadm test /dev/ttyACM0

* + To test my new rules, I have to disconnect and re-connect the USB devices. or “ udevadm trigger”, “ls -l /dev”
  + I should see that a link has been created:

lrwxrwxrwx 1 root root           7 Dec 26 19:23 ttyUSB50 -> ttyACM0

lrwxrwxrwx 1 root root           7 Dec 26 19:30 ttyUSB51 -> ttyACM1

* Test : install GtkTerm 🡪 sudo apt-get install gtkterm
  + Use gtkterm to send data to ttyACM0



Install Pi4J tool package

* Direct download from internate

curl -s get.pi4j.com | sudo bash

* Offline Manual

1. Download from : <http://get.pi4j.com/download/pi4j-1.1.deb>
2. transfer the download installer package over to your RaspberryPi
3. sudo dpkg -i pi4j-1.1.deb

* upgrade :

sudo apt-get install pi4j or pi4j --update

* Installed Location / Example Files

/opt/pi4j/lib   
/opt/pi4j/examples

* compile a Java program using the Pi4J libraries,

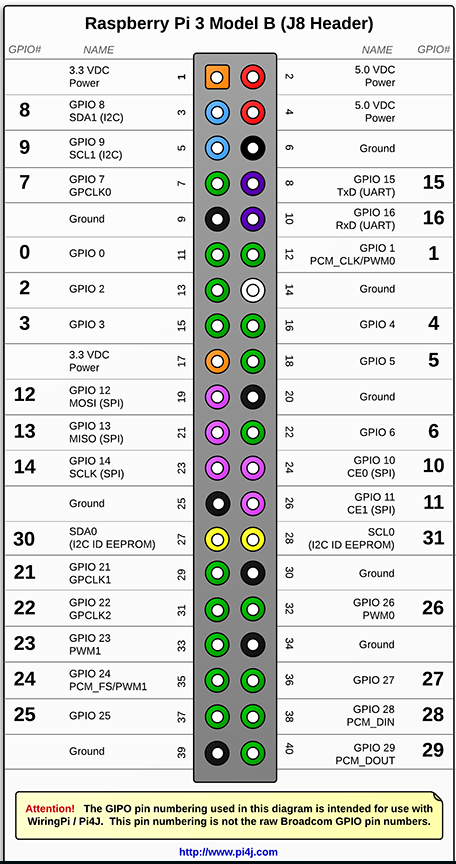
javac -classpath .:classes:/opt/pi4j/lib/'\*' –d . SerialExample.java

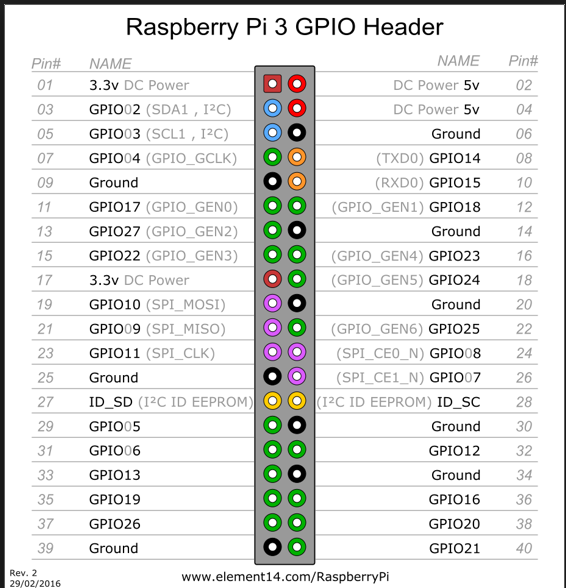
* to start a Java program using the Pi4J libraries

sudo java -classpath .:classes:/opt/pi4j/lib/'\*' SerialExample

* pi4j Web resource :

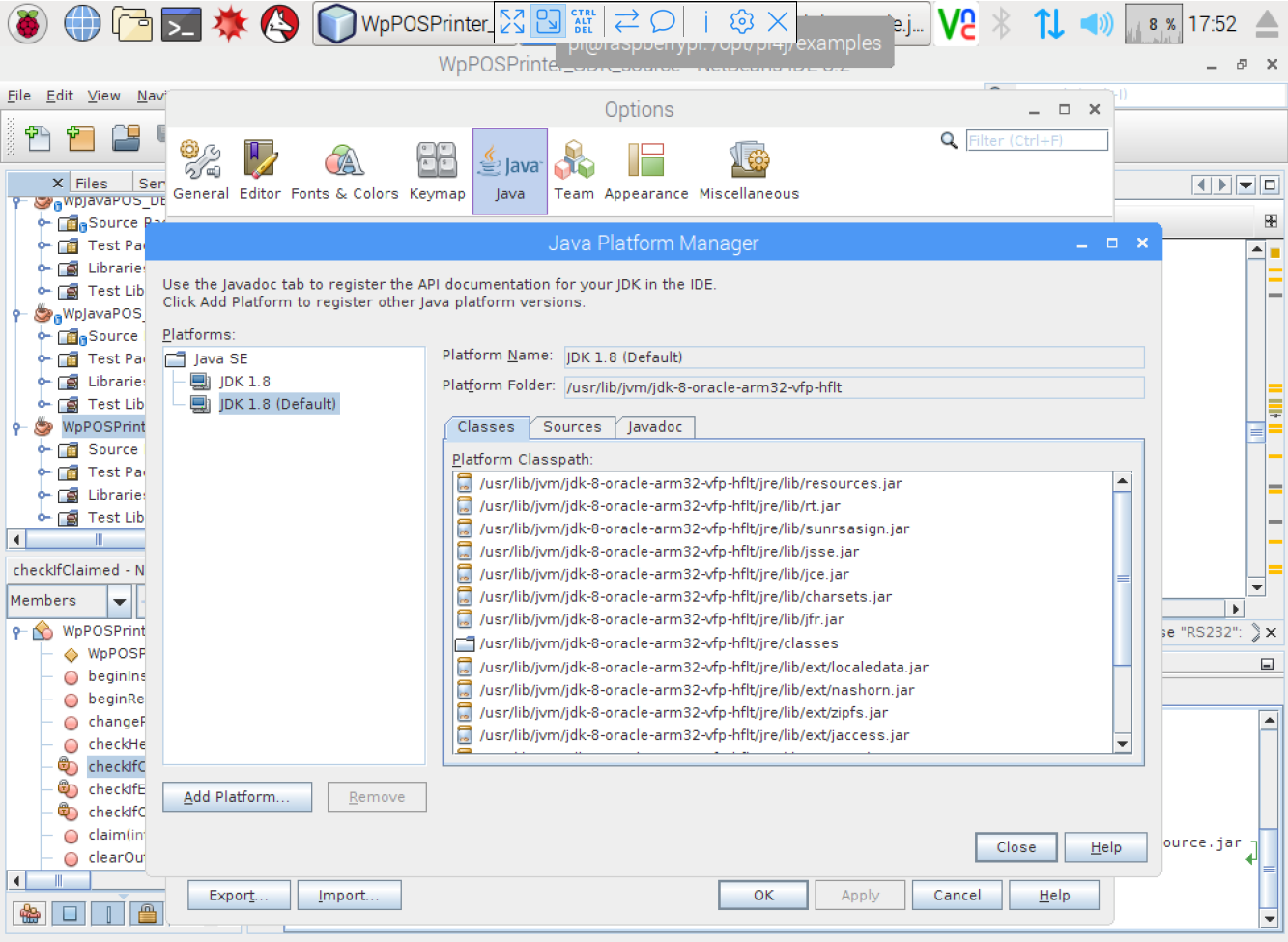
<http://pi4j.com/example/serial.html>



15#

NetBean IDE setup

Options 🡪 Java 🡪 Java Platform Manager 🡪 your java install version



* Auto startup setting 1 : startup from terminal
  + cd /home/pi
  + Sudo nano ~/.bashrc
  + Add “bash myjar.sh” in the last line

**PS: SSH disabled by default; can be enabled by creating a file with name "ssh" in boot partition**

# Run GUI as root

sudo nano /etc/rc.local

Before the "exit 0" line in it add the following line:

#Auto run the GUI as root

sudo startx

Save it by pressing Ctrl+X, " Y", ENTER

Re-boot your RPi and it should automatically run the GUI as the root user.

* **Setting An Application / Your Application To Automatically Run In The GUI**
  + For the standard Pi user (if not using "sudo startx"):

sudo nano /home/pi/.config/lxsession/LXDE-pi/autostart

* For the root Pi user (if running the GUI with "sudo startx"):

sudo nano /root/.config/lxsession/LXDE-pi/autostart

* Add @lxterminal in the end of lines
* **Free SD card space**

df -h shows you your disk usage

sudo apt-get purge wolfram-engine

sudo apt-get clean  
sudo apt-get autoremove

…which showed me I could free up 658 MB

sudo apt-get remove --purge libreoffice\*

sudo apt-get clean  
sudo apt-get autoremove

…which showed another large space saving of 253MB

rem

First thing after afresh install I purge off the SD card are:

wolfram-engine

scratch

minecraft-pi

sonic-pi

dillo gpicview penguinspuzzle oracle-java8-jdk openjdk-7-jre oracle-java7-jdk openjdk-8-jre

Nearly 2 GB of things I do not need!

* **Resizing a partition within an image file**

1. rpi-clone is a shell script that will back up (clone using dd and rsync) a running Raspberry Pi file system to a destination SD card 'sdN' plugged into a Pi USB port (via a USB card reader).
2. rpi-clone works on Raspberry Pi distributions which have a VFAT boot partition 1 and a Linux root partition 2.
3. rpi-clone does not work with NOOBS.
4. rpi-clone can clone the running system to a new SD card or can incrementally rsync to existing backup Raspberry Pi SD cards.
5. During the clone to new SD cards, rpi-clone gives you the opportunity to give a label name to the partition 2 .so you can keep track of which SD cards have been backed up.
6. Just stick a correspondingly named sticky label on each SD card you have and you can look up the last clone date for that cardin the rpi-clone log file /var/log/rpi-clone.
7. If the destination SD card has an existing partition 1 and partition 2 matching the running partition types, rpi-clone assumes (unless using the -f option) that the SD card is an existing backup with the partitions properly sized and set up for a Raspberry Pi.
8. This is done by a partial 'dd' from the running booted device /dev/mmcblk0 to the destination SD card /dev/sdN followed by a fdisk resize and mkfs.ext4 of /dev/sdN partition 2.
9. This creates a completed partition 1 containing all boot files and an empty but properly sized partition 2 rootfs. The SD card partitions are then mounted on /mnt/clone and rsynced to the running system.
10. This is so you can go look at the clone results or make any custom final adjustments if needed. For example, I have a couple of Raspberry Pis and I use one as a master. When I clone for the benefit of the second Pi, I do a "cd /mnt/clone/etc" and fix the files needed to customize for the second Pi (well, actually I do that with a script that takes my desired Pi hostname as an argument).
11. Either way, you typically might need to change at least these files:

/etc/hostname # I have one of rpi0, rpi0, ...

/etc/hosts # The localhost line should probably be changed

/etc/network/interfaces # If you need to set up a static IP or alias

1. If you cd into the /mnt/clone/tree to make some of these customizations or just to look around, don't forget to cd out of the /mnt/clone tree before telling rpi-clone to unmount.

* Backup Raspberry SD by using rpi-clone
* rpi-clone is on github, to get it and install it to /usr/local/sbin:

Go to https://github.com/billw2/rpi-clone and download the zip file:

$ unzip rpi-clone-master.zip

$ cd rpi-clone-master

$ cp rpi-clone /usr/local/sbin

* or, use git to clone the repository:

$ git clone https://github.com/billw2/rpi-clone.git

$ cd rpi-clone

$ sudo cp rpi-clone /usr/local/sbin

$ cd /usr/local/sbin

$ sudo rpi-clone sda –f -v

* or operation from GUI

sudo chmod +x rpi-clone to make the file executable

sudo mv rpi-clone /usr/local/sbin Open a terminal and move the rpi-clone script to the **usr/bin** folder

Now create a **.desktop** file at the appropriate location by typing...

cd /usr/share/raspi-ui-overrides/applications  
sudo nano rpi-clone.desktop

...then enter the following...

[Desktop Entry]  
Name=RPI-Clone  
Comment=Clones the OS SD card to a USB-connected SD card of greater capacity than the total occupied file space  
Icon=/usr/share/pixmaps/rpi-clone.jpg  
Exec= lxterminal -e 'sudo /usr/local/sbin/rpi-clone sda -f -v'  
Type=Application  
Terminal=false  
Categories=Utility

*Note: The "Comment=" line should all be on one line.*

You can change the jpgfile by yours in : /usr/share/pixmaps/rpi-clone.jpg

* Now look in **Menu/Accessories** and rpi-clone should be there ready for launch at the click of a mouse button whenever you need it.
* **Load MyPi Server on boot:**

1. Stop MyPi Server by pressing CTRL C twice.
2. Edit rc.local file by typing:

**sudo nano /etc/rc.local**

1. Scroll down and type the following command BEFORE the exit 0 line:

**python3 /home/pi/mypi\_server.pyc &**

https://www.raspberrypi.org/forums/viewtopic.php?f=29&t=137693&p=914109#p914109

* Raspberry Pi 解鎖 root 帳號
  + sudo passwd root
  + 會要求你輸入兩次新密碼，這時你可以輸入你想設的密碼。
  + sudo passwd -u root
  + 再輸入這個， root 帳號就解鎖了！

# How to run MyPi TCP server

1. Donwnload & install MyPi TCP Server

$Cd /home/pi

$wget [www.ioslinks.com/mypi/server.zip](http://www.ioslinks.com/mypi/server.zip)

$unzip server.zip

There will hav mypy\_server.pyc & mypi.cfg two file

1. Run & Test MyPi Server
   * + Start Mypi server by typing :

$python3 mypi\_server.pyc

* + - Then you can run IOS app
    - In mypi.cfg :
      * TCP port :5000
      * Connection Password : 123456
      * Output pin : 7,11,13,15,16,18,19,21
      * Input pin: 29,31,32,33,36,37,23,24
* Send data directerly to Printer
  + $ stty -F /dev/ttyACM0 115200
  + $echo -e "This is a test.\n" > /dev/ttyACM0
* install CUPS library

$ sudo apt-get update

$ sudo apt-get install libcups2-dev libcupsimage2-dev git build-essential cups system-config-printer

* install zj-58

$git clone <https://github.com/klirichek/zj-58.git>

$cd zj-58/

$make

$sudo ./install

* + lpt command
    - $lpstat –P ask printer status
    - $echo "This is a test." | lpr
    - $lpr -o fit-to-page /usr/share/raspberrypi-artwork/raspberry-pi-logo.png
* List the devices

# lpinfo -v #

$ /usr/lib/cups/backend/snmp *ip\_address* # Use SNMP to find a URI

* List the drivers

$ lpinfo -m

* Add a new queue

# lpadmin -p *queue\_name* -E -v *uri* -m *model*

* The *queue\_name* is up to you. Example:

# lpadmin -p HP\_DESKJET\_940C -E -v "usb://HP/DESKJET%20940C?serial=CN16E6C364BH" -m drv:///HP/hp-deskjet\_940c.ppd.gz

# lpadmin -p SHARED\_PRINTER -m raw # Raw queue; no PPD or filter

* Set the default printer

$ lpoptions -d *queue\_name*

* Change the options

$ lpoptions -p *queue\_name* -l # List the options

$ lpoptions -p *queue\_name* -o *option*=*value* # Set an option

* Example:

$ lpoptions -p HP\_DESKJET\_940C -o PageSize=A4

* Check the status

$ lpstat -s

$ lpstat -p *queue\_name*

* Deactivate a printer

# cupsdisable *queue\_name*

* Activate a printer

# cupsenable *queue\_name*

* Set the printer to accept jobs

# cupsaccept *queue\_name*

* Remove a printer
* First set it to reject all incoming entries:

# cupsreject *queue\_name*

* Then disable it.

# cupsdisable *queue\_name*

* Finally remove it.

# lpadmin -x *queue\_name*

* Print a file

$ lpr *file*

$ lpr -# 17 *file* # print the file 17 times

$ echo "Hello, world!" | lpr -p # print the result of a command. The -p switch adds a header.

* Check the queue

$ lpq

$ lpq -a # on all queues

* Clear the queue

# lprm # remove last entry only

# lprm - # remove all entries

* Build CAUPS UBUNTU16 printer driver
  + - Modify WptCUPS4Ubuntu20170310.sh
      * Mark line 15 : “rm -f /tmp/wptcups.tgz”
        + $sudo ./QuickConfig.sh

🡺 will generator the wptcups.tgz

* + - * + Then you can modify/add files in the
      * Encode the tar file
        + $uuencode –m /tmp/wptcups.tgz wptcups.tgz > wptcups.tgz.base64
        + Copy the contain of the wptcups.tgz.base64 into the WptCUPS4Ubuntu20170310.sh
* Remove cups :
  + Sudo apt-get autoremove
  + $sudo clean
* Install cups
  + sudo apt-get install cups
  + sudo usermod -a -G lpadmin pi
  + sudo nano /etc/cups/cupsd.conf
  + modify the following :

# Only listen for connections from the local machine  
# Listen localhost:631  
Port 631

* install EPSON TM88 printer



* IOT printer (https://www.dfrobot.com/blog-407.html)
  + Install Neopixel
    - "sudo apt-get install python-pip"
    - "sudo pip install imapclient"
    - "sudo apt-get install python-serial python-imaging python-unidecode"
    - "sudo apt-get install git"
    - "git clone <https://github.com/DAFRELECTRONICS/IoTprinter>"
    - "cd IoTprinter"
    - "sudo python printertest.py"
  + **Configuring Gmail account for the printer**
  + **Configuring the script in the Raspberry**
    - **Install neopixel**
      * **sudo apt-get install build-essential python-dev git scons swig**
      * git clone https://github.com/jgarff/rpi\_ws281x.git
      * cd rpi\_ws281x
      * scons
      * **cd python**
      * **sudo python setup.py install**
    - “sudo nano mailverifier.py”
      * Need to preinstall “sudo *pip install cryptography”*
    - “sudo python mailverifier.py”
  + Need Auto run ?
    - sudo nano /etc/rc.local 🡪 add the following

cd /home/pi/IOTPrinter  
python mailverifier.py &

* UBUNTU install java ( for PC)

$ sudo apt-get install default-jdk

$ sudo add-apt-repository ppa:webupd8team/java

$ sudo apt-get update

$ sudo apt-get install oracle-java8-installer

$ sudo update-alternatives --config java

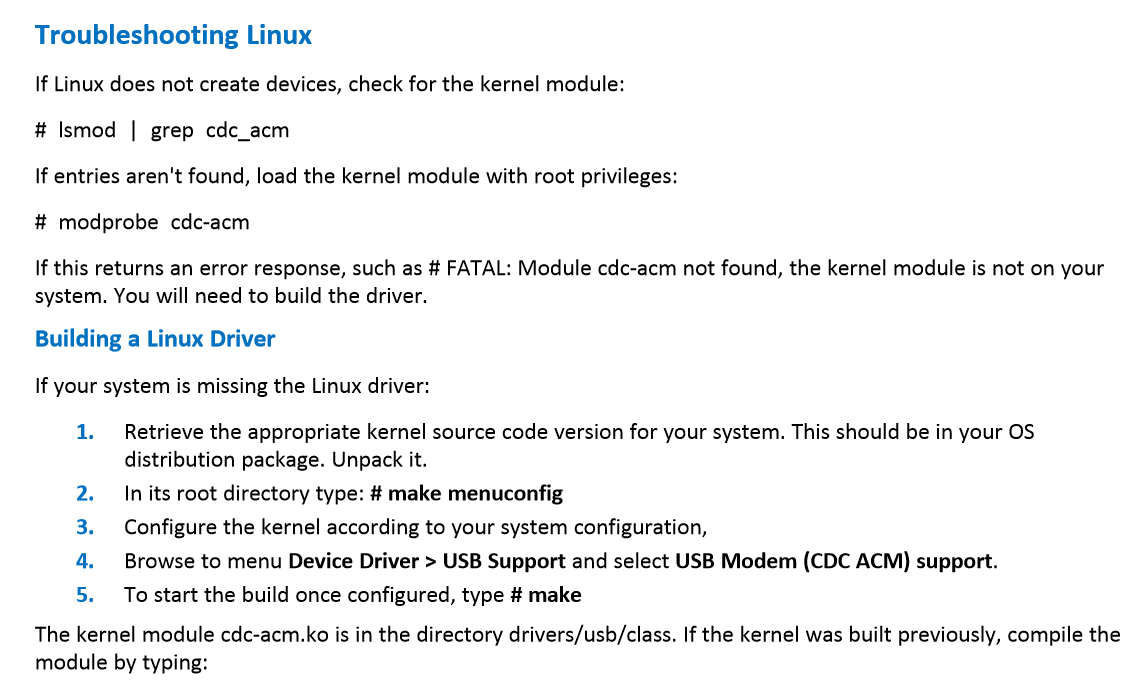
$ sudo nano /etc/environment

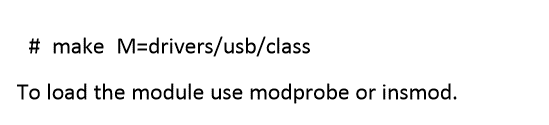
Add 🡪 JAVA\_HOME="/usr/lib/jvm/java-8-oracle"

$ reload 🡪 $ source /etc/environment

* UBUNTU fix Broken sudo
  + $ mount -o rw,remount /
  + $ adduser username sudo
* UBUNTU install CUPS
  + $ sudo apt install cups
  + $ sudo nano /etc/cups/cupsd.conf
    - * + **Browsing On**
  + **$ sudo systemctl restart cups.service**
  + **$ sudo usermod –aG lpadmin winpos**
* UBUNTU /dev/ttyACM0 access denined solution
  + /etc/apparmor.d/usr.sbin.cupsdA 🡪 add “/dev/ttyACM\* rw,

# Install Linux CDC ACM driver





Install JAVA JDK8

1. 下載[JDK 8 for ARM](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-arm-downloads-2187472.html)
2. $ sudo mkdir -v -v /opt/java  
   $ sudo tar zxvf ~/jdk-8u33-linux-arm-vfp-hflt.tar.gz -C /opt/java
3. $ sudo update-alternatives --install "/usr/bin/java" "java" "/opt/java/jdk1.8.0\_33/bin/java" 1  
   $ sudo update-alternatives --set java /opt/java/jdk1.8.0\_33/bin/java
4. 另外，有些軟體需要環境變數JAVA\_HOME，請編輯~/.bashrc，加入：  
   export JAVA\_HOME="/opt/java/jdk1.8.0\_33"  
   export PATH=$PATH:$JAVA\_HOME/bin

* **Install and Setup Samba Server**

<https://eltechs.com/raspberry-pi-nas-guide/?utm_source=facebook.com&utm_medium=smm_post&utm_campaign=raspberry_pi_nas_guide>

$ sudo apt-get update

$ sudo apt-get upgrade

$ sudo apt-get install samba samba-common-bin

open the */etc/samba/smb.conf and add the following:*

[INTERNAL]

comment = internal Files

browseable = yes

path = /home/pi/INTERNAL

writeable = Yes

create mask = 0777

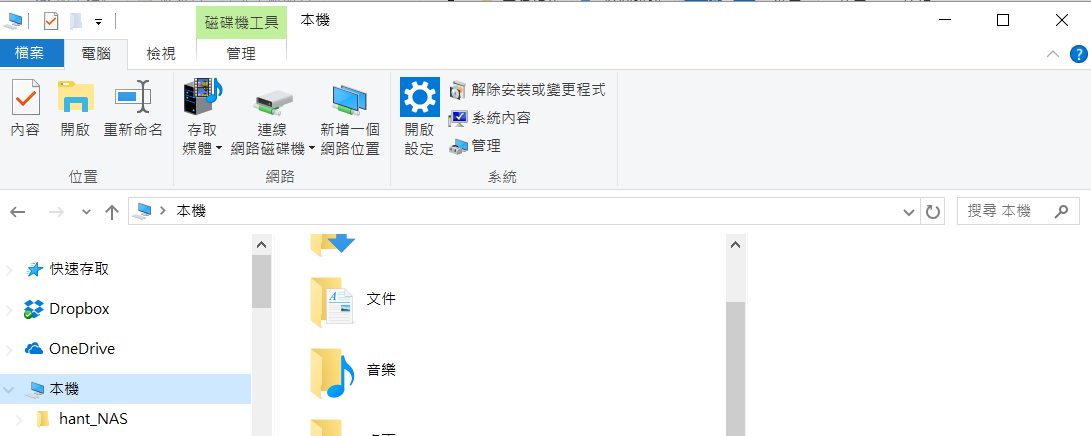
directory mask = 0777

browseable = Yes

public = yes

$ sudo smbpasswd -a pi to change the smba password

$ sudo /etc/init.d/samba restart

On Windows open the Explorer and choose “Map network drive” from the Menu 

選連線網路磁碟

