โครงการจัดอบรมพัฒนาความรู้ทักษะด้าน

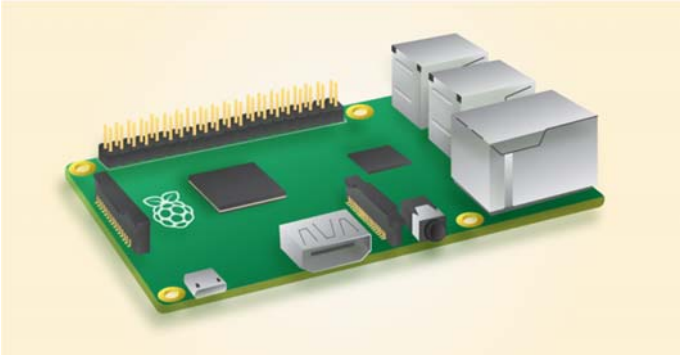
Internet of Things

ได้รับการสนับสนุนจาก งบประมาณแผ่นดิน ปี 2560
ตามแผนบูรณาการการพัฒนาคนตลอดช่วงวัย

Raspberry Pi

What is Raspberry Pi

- The Raspberry Pi is a low cost, **credit-card sized computer** that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.



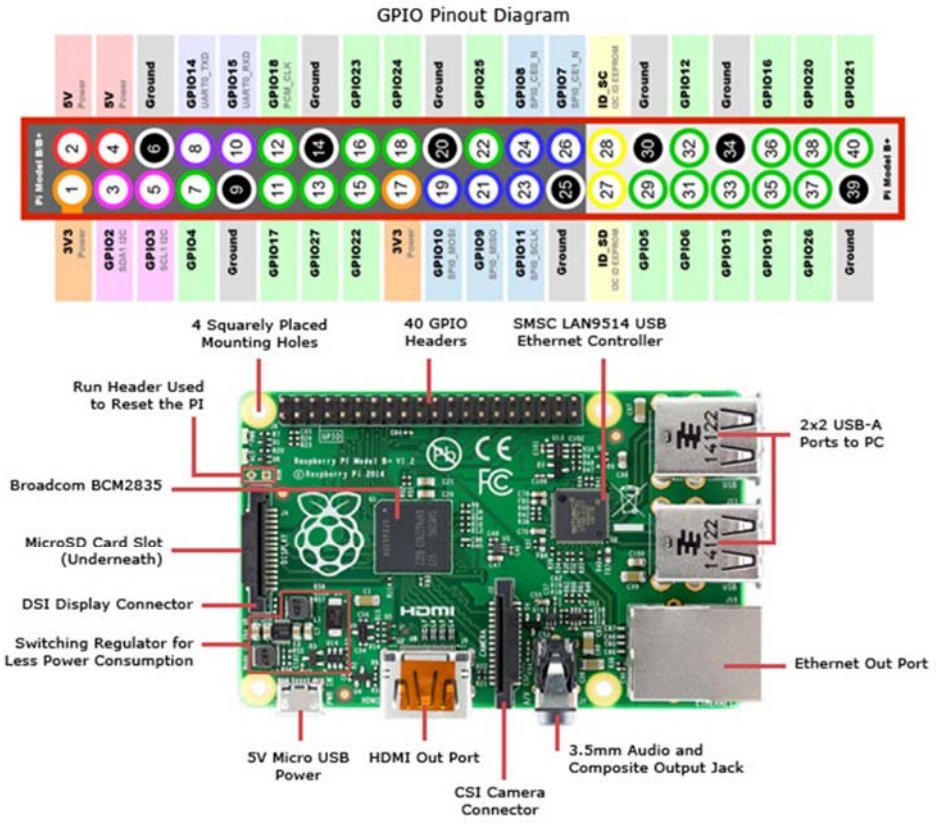
Raspberry Pi 2 Model B

The Raspberry Pi 2 Model B is the second generation Raspberry Pi. It replaced the original Raspberry Pi 1 Model B+ in **February 2015**. Compared to the Raspberry Pi 1 it has:

A 900MHz quad-core ARM Cortex-A7 CPU
1GB RAM
Like the (Pi 1) Model B+, it also has:
4 USB ports
40 GPIO pins
Full HDMI port
Ethernet port
Combined 3.5mm audio jack and composite video
Camera interface (CSI)
Display interface (DSI)
Micro SD card slot
VideoCore IV 3D graphics core

Because it has an ARMv7 processor, it can run the full range of ARM GNU/Linux distributions, including Snappy Ubuntu Core, as well as Microsoft Windows 10

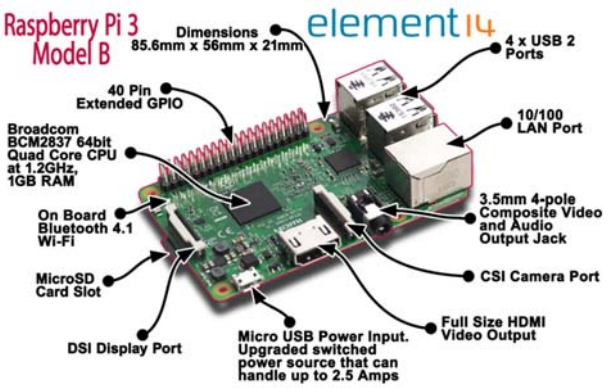
Raspberry Pi 2 Model B



Raspberry Pi 3 Model B

The Raspberry Pi 3 is the third generation Raspberry Pi. It replaced the Raspberry Pi 2 Model B in **February 2016**. Compared to the Raspberry Pi 2 it has:

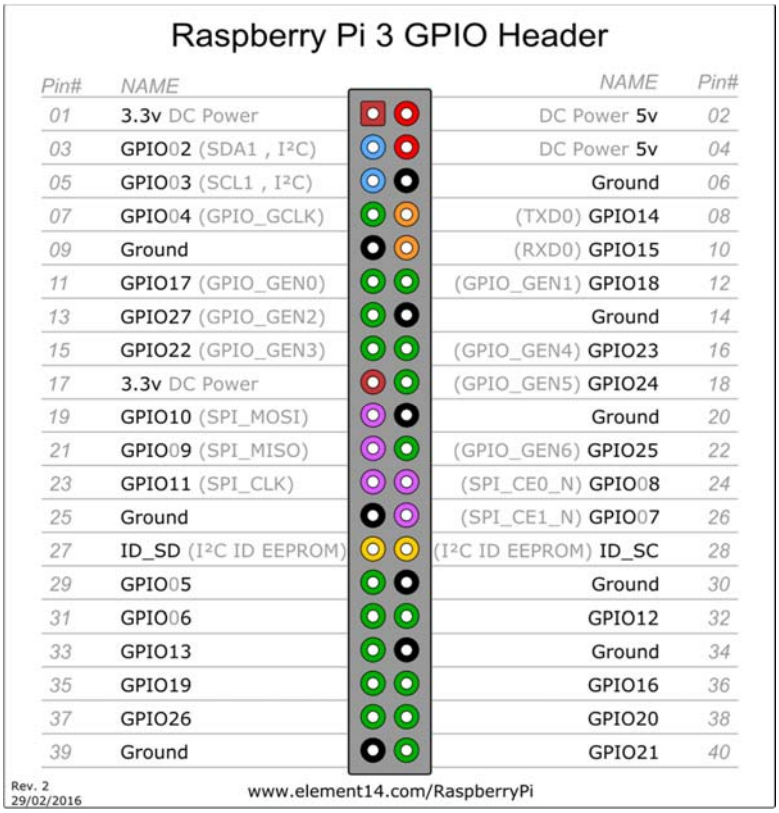
A 1.2GHz 64-bit quad-core ARMv8 CPU
802.11n Wireless LAN
Bluetooth 4.1
Bluetooth Low Energy (BLE)



Like the Pi 2, it also has:

- 1GB RAM
- 4 USB ports
- 40 GPIO pins
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display interface (DSI)
- Micro SD card slot (now push-pull rather than push-push)
- VideoCore IV 3D graphics core

Raspberry Pi 3 Model B



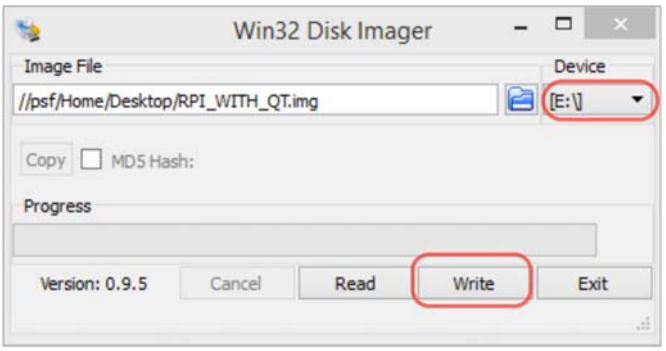
การทดลองที่ 1 ติดตั้งระบบปฏิบัติการ Raspbian OS

- ระบบปฏิบัติการที่เป็น Official Support จาก Raspberry Pi ชื่อว่า Raspbian OS สามารถ Download ได้ที่

  https://www.raspberrypi.org/downloads/

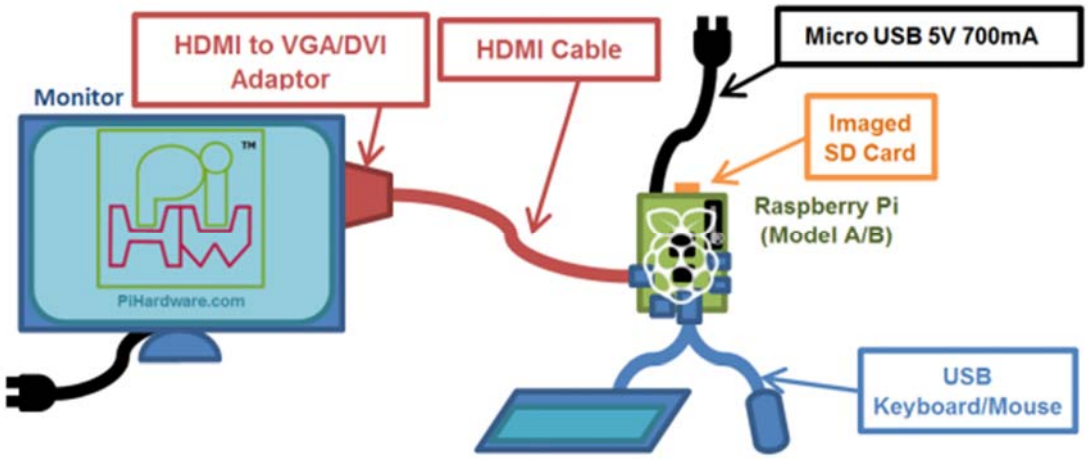
- ติดตั้งโปรแกรม Win32 Disk Imager

1. เลือก Image ไฟล์
2. เลือก Device ที่ต้องการจะลง OS
3. กด Write



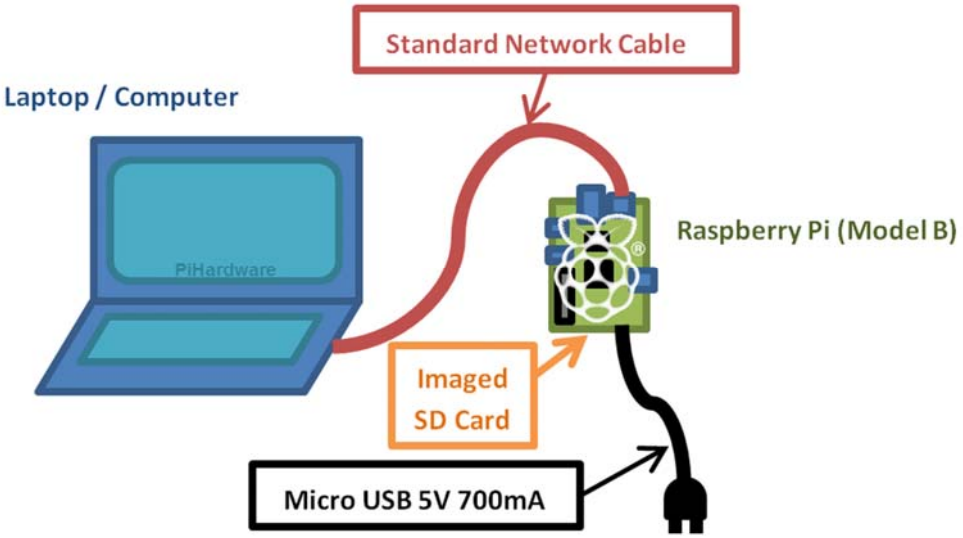
Raspberry Pi Connect

- connect via a DVI connection, commonly available on newer monitors and digital TVs.



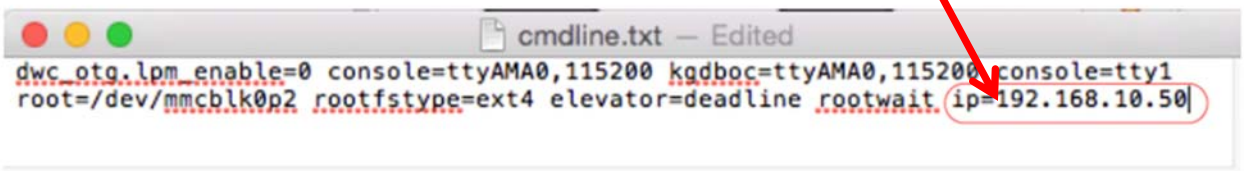
Raspberry Pi Connect

- Raspberry Pi Remote Connections



Setting the Raspberry Pi's IP address

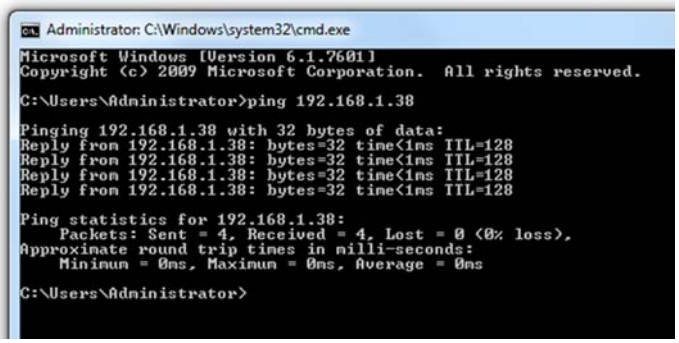
- **Quick Setup / Setting up when you don't have a monitor**
  - Ensure the Raspberry Pi is powered off, and remove the SD-Card.
  - Insert the SD-Card into a card reader and plug it into your laptop.
  - Make a copy of cmdline.txt and rename it cmdline.normal
  - Edit cmdline.txt and add the IP address at the end (be sure you don't add any extra lines). **10.30.4.1XX XX =** เบอร์ฝากล่อง



ข้อควรระวัง บน **Windows** ไม่ควรใช้ **Notepad** ในการแก้ เพราะมีปัญหาเรื่องอักขระพิเศษและการเว้นบรรทัด ควรใช้ **Editor** ตัวอื่นๆ เช่น **Notepad++**

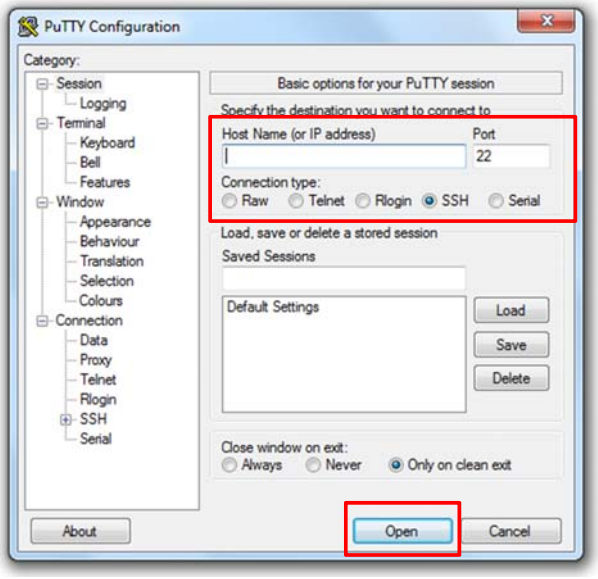
การทดลองที่ 2 เริ่มต้นใช้งาน Raspberry Pi

- นำ SD Card ที่ติดตั้ง OS เสร็จแล้ว และแก้ไข IP Address เรียบร้อยแล้ว ไปติดตั้งที่ Rasp Pi
- เสียบสาย LAN เข้าที่ Rasp Pi
- เสียบ Adapter เพื่อจ่ายกำลังงานไฟฟ้า ให้กับ Rasp Pi
- ทดลองใช้คำสั่ง ping ที่เครื่อง PC ไปยัง IP Address ที่กำหนดไว้ตอนต้น ว่าสามารถติดต่อกับ Rasp Pi ได้หรือไม่



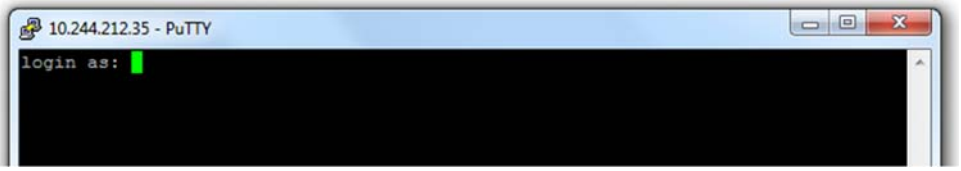
การทดลองที่ 2 เริ่มต้นใช้งาน Raspberry Pi

- ทำการติดตั้งโปรแกรม Putty เพื่อทำการเชื่อมต่อกับ Rasp Pi ผ่าน Secure Shell V2 (SSH)
- ระบุ IP Address ของ Rasp Pi
- เลือก SSH
- กดปุ่ม Open

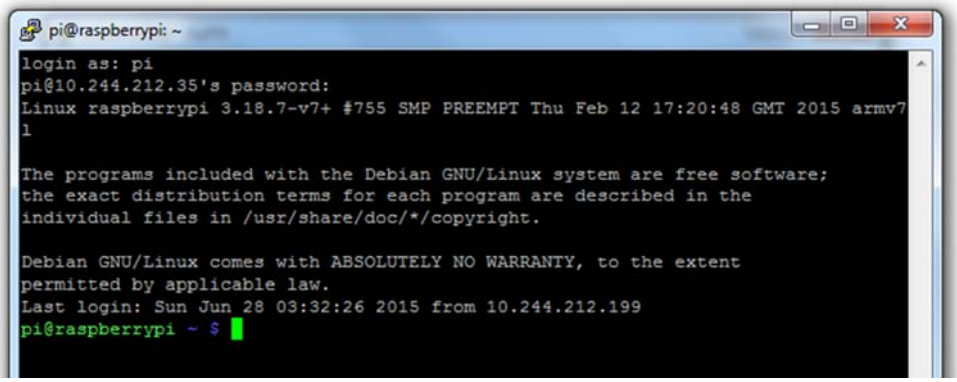


การทดลองที่ 2 เริ่มต้นใช้งาน Raspberry Pi

- จะปรากฏหน้าจอ Console ของ Rasp Pi
- ใส่ username : pi และ Password คือ raspberry



- เมื่อ Login เสร็จ จะปรากฏหน้าจอดังรูป



Basic Linux command

คำสั่ง	หน้าที่	รายละเอียด
ls	List file	แสดงรายชื่อไฟล์และไดเรคทอรี่
cp	copy file	สำเนาไฟล์
mv	Move file	เหมือน Ctrl + X แล้ว Ctrl+V
rm	Delete files	ลบไฟล์
cd	Change directory	ย้ายไปไดเรคทอรี่ที่ต้องการ
pwd	Print current directory name	แสดงชื่อไดเรคทอรี่ปัจจุบัน
mkdir	Create directory	สร้างไดเรคทอรี่
rmdir	Delete directory	ลบไดเรคทอรี่
cat	view file	ดูเนื้อหาของ text file
scp	SSH transfer protocol	ทรานเฟอร์ข้อมูล ผ่าน SSH
tar	read/write type archives	จัดเก็บไฟล์แบบ compress หรือแตกไฟล์ออกมา
sudo	Super user do	super user ทำ
chmod	Change file protections	เปลี่ยนระดับ permission ของไฟล์
vi	vi	editor ชื่อ VI
nano	nano	editor ชื่อ nano

การทดลองที่ 2 เริ่มต้นใช้งาน Raspberry Pi

- ทดลองใช้คำสั่ง basic command ของ Linux เช่น
  - ls ในการแสดง รายชื่อไฟล์
  - cd ในการเปลี่ยน directory เช่น cd / , cd ..
  - df –h ในการดูขนาดของ file system

- ทำการ expand file system เพื่อขยายพื้นที่ให้เต็มความจุของ disk โดยใช้คำสั่ง sudo raspi-config จากนั้นเลือกหัวข้อ Expand FileSystem จากนั้นเลื่อนไปที่ <Finish> แล้ว Rasp Pi จะทำการ Reboot ใหม่

การทดลองที่ 2 เริ่มต้นใช้งาน Raspberry Pi

- ทำการ Config Static IP Address ของ Raspi บน file system
- ใช้คำสั่ง

  sudo nano /etc/dhcpcd.conf

- พิมพ์ข้อมูลต่อไปนี้ลงไป ต่อท้ายบรรทัดล่างสุด

  interface eth0
  static ip_address=10.30.4.1XX/24
  static routers=10.30.4.1
  static domain_name_servers=8.8.8.8

- ออกจากโปรแกรม nano โดยการกด Ctrl+x และ save file โดยการกด yes

การทดลองที่ 2 เริ่มต้นใช้งาน Raspberry Pi

- จากนั้น ลบค่า IP Address ใน cmdline.txt ออก โดยใช้คำสั่ง

  sudo nano /boot/cmdline.txt

- ทำการ Reboot และ ทดลองเชื่อมต่อ Rasp pi ด้วย IP Address ใหม่ ที่กำหนด

  sudo shutdown –r now

Python เป็นภาษาระดับสูงภาษาหนึ่ง ที่มีความสามารถสูงถูกสร้างขึ้นในปี 1989 โดย Guido van Rossum ซึ่งถูกพัฒนาขึ้นมาโดยไม่ยึดติดกับแพลตฟอร์ม

เป็นภาษาที่ Raspberry PI แนะนำให้ใช้ สังเกตุได้ว่ามีไลบรารี่ในการเชื่อมต่อฮาร์ดแวร์ออกมาเป็นจำนวนมาก และตัวอย่างโปรแกรมของ Raspberry PI ก็มักจะเป็นภาษา Python

สรุป ภาษา python แบบรวบรัด

- การแสดงค่า ใช้คำสั่ง print เช่น print “Hello”
- การประกาศตัวแปร ไม่ต้องระบุประเภทตัวแปร เช่น varint = 10
- คำสั่งทางคณิตศาสตร์ เช่น + - * /
- คำสั่งในการเปรียบเทียบ เช่น == , >= , <=, > , <
- คำสั่งการตัดสินใจ if ..else เช่น

```
if x > 10:
        print  “x is large”
else:
        print  “x is small”
```

- คำสั่งวน loop เช่น

```
x = 0
while x < 10
        print “Hello”
        x = x+1
```

ขั้นตอนการเขียนโปรแกรม python

- เปิดไฟล์เพื่อเขียนโปรแกรม ใช้คำสั่ง nano เช่น

nano helloworld.py

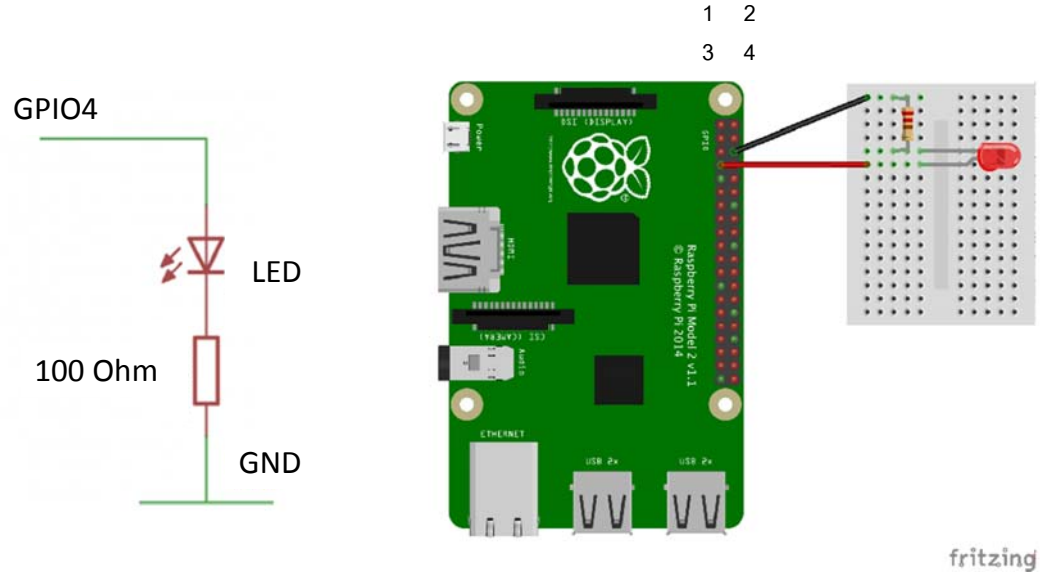
- Run ไฟล์โปรแกรมที่เขียน ด้วย python interpreter โดยใช้คำสั่ง python เช่น

python helloworld.py หรือ

sudo python helloworld.py

การทดลองที่ 3 GPIO Blink

- ต่อขา GPIO4 เข้ากับ LED และ ตัวต้านทาน 100 โอห์ม ดังรูป



การทดลองที่ 3 GPIO Blink

- พิพม์คำสั่ง nano led.py
- เขียนโปรแกรม python ดังนี้

```
import RPi.GPIO as GPIO
import time

GPIO.setmode (GPIO.BCM)
LIGHT = 4

GPIO.setup(LIGHT,GPIO.OUT)

while True:
        GPIO.output(LIGHT,True)
        time.sleep(0.5)
        GPIO.output(LIGHT,False)
        time.sleep(0.5)
```

- Run โปรแกรมโดยใช้คำสั่ง sudo python led.py

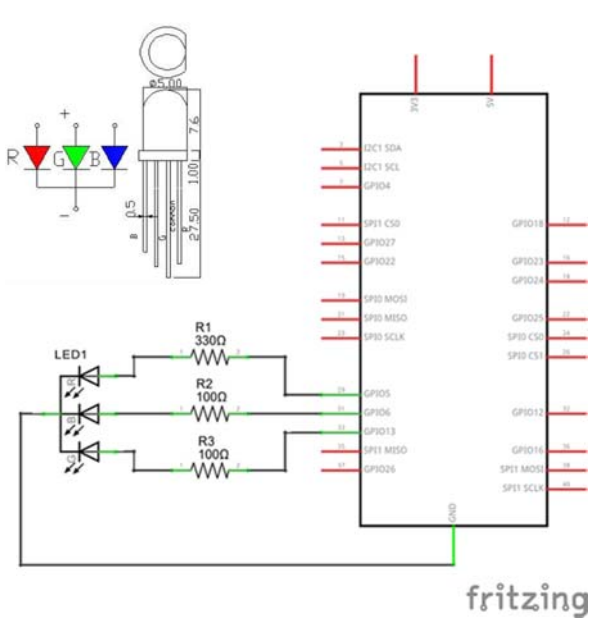
การทดลองที่ 3 GPIO Blink

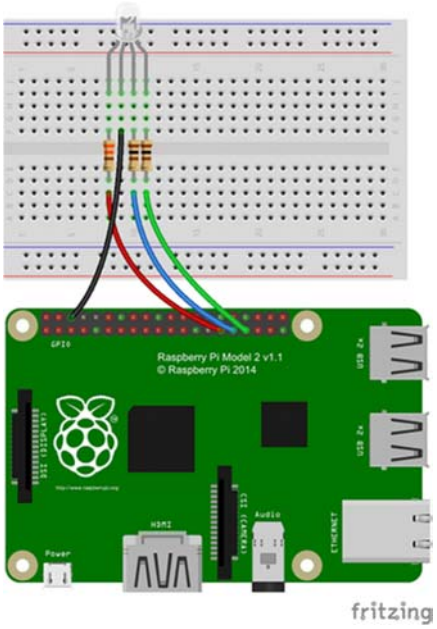
อธิบายโปรแกรม

```
import RPi.GPIO as GPIO          # เรียก Library สำหรับควบคุม GPIO Pin
import time                      # เรียก Library สำหรับสร้าง time delay

GPIO.setmode (GPIO.BCM)          # Set the pin designation type to GPIO.BCM
                                   to use BCM numbering convention
                                   ( ใช้การกำหนดเลข Pin ตาม chip ของ BCM)
LIGHT = 4                        # กำหนด GPIO4 เท่ากับตัวแปร Light

GPIO.setup(LIGHT,GPIO.OUT)       # กำหนด GPIO4 เป็นขา Output

while True:
        GPIO.output(LIGHT,True)      # กำหนด GPIO4 มีค่า High
        time.sleep(0.5)              # delay 0.5 วินาที
        GPIO.output(LIGHT,False)     # กำหนด GPIO4 มีค่า Low
        time.sleep(0.5)              # delay 0.5 วินาที
```

แบบฝึกหัด

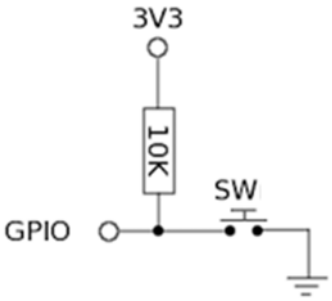
จงเขียนโปรแกรมแสดงสี 8 สี โดยเว้นช่วงสีละ 1 วินาที
กำหนดสีเอง ตามใจชอบ

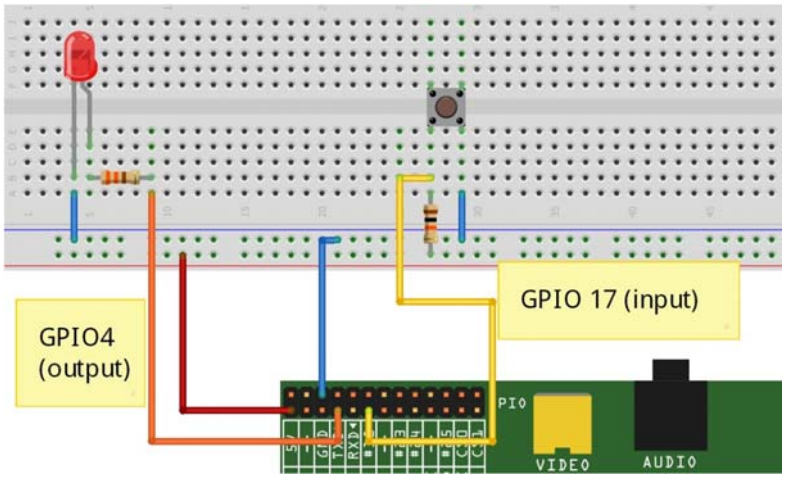




การทดลองที่ 4 GPIO Input

- ต่อขา GPIO17 เข้ากับ Switch และ ตัวต้านทาน 10 กิโลโอห์ม ดังรูป





การทดลองที่ 4 GPIO Input

การกำหนดค่าให้ GPIO17 เป็น Input ใช้คำสั่ง

```
GPIO.setup(17,GPIO.IN)        # กำหนด GPIO17 เป็นขา Input
```

และการอ่านค่าจาก GPIO17 ใช้คำสั่ง

```
val = GPIO.input(17)          # ค่าที่อ่านได้ มาเก็บในตัวแปร val โดย
                                จะได้ค่าเป็น 0 หรือ 1
```

แบบฝึกหัด

เมื่อกดปุ่ม ให้ LED ติด และเมื่อปล่อยให้ LED ดับ

การทดลองที่ 5 ติดตั้ง MQTT Broker

ในที่นี้จะใช้ MQTT Broker ของ Mosquitto

Mosquitto
An Open Source MQTT v3.1/v3.1.1 Broker

ในการติดตั้ง MQTT Broker จะติดตั้ง package 3 ตัวดังนี้

- mosquitto is the MQTT broker (i.e. server)
- mosquitto-clients are the command-line clients
- python-mosquitto are the Python Library

Repository

- โดยปกติ เวลาเราติดตั้งโปรแกรมบน Windows คือเราจะต้องโหลดโปรแกรมมาติดตั้งเอง จากเว็บไหนก็ได้ ซึ่งบางทีอาจได้ของแถมมาด้วย แต่การติดตั้งแอพพลิเคชั่นบน Ubuntu หรือ OS ตระกูล Linux จะต่างกัน โดย Ubuntu จะใช้การติดตั้งซึ่งดึงไฟล์แอพพลิเคชั่นจากเว็บไซต์โดยตรง ซึ่งเราเรียก เว็บที่เก็บโปรแกรมพวกนั้นว่า ***repository*** ซึ่งเวลามีนักพัฒนาแอพพลิเคชั่นส่งขึ้นไปบนเว็บเหล่านี้จะมีการตรวจสอบก่อนที่จะปล่อยให้ผู้ใช้ทั่วไปติดตั้ง ซึ่งโดยปกติแอพพลิเคชั่นที่ถูกส่งไปบน ***repository*** มักจะเป็นโปรแกรมที่สมบูรณ์ ไม่ใช่เวอร์ชั่น Alpha,Beta และเป็นโปรแกรมที่ถูกรวมเข้าไปใน ***default repository***

- *ที่มา http://www.ubuntuthailand.com/*

การทดลองที่ 5 ติดตั้ง MQTT Broker

1. add key repository ของ mosquitto โดยใช้คำสั่ง

   wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
   sudo apt-key add mosquitto-repo.gpg.key

2. เข้าโฟลเดอร์ source.list.d เพื่อเตรียมติดตั้ง repository ใหม่ โดยใช้คำสั่ง

   cd /etc/apt/sources.list.d/

3. add repository ของ mosquitto โดยใช้คำสั่ง

   sudo wget http://repo.mosquitto.org/debian/mosquitto-wheezy.list

การทดลองที่ 5 ติดตั้ง MQTT Broker

4. สั่ง update apt information โดยใช้คำสั่ง

   sudo apt-get update

5. ติดตั้ง package ทั้งสาม โดยใช้คำสั่ง

   sudo apt-get install mosquitto mosquitto-clients python-mosquitto

6. ติดตั้ง Library สำหรับ paho.mqtt เพื่อเขียน Python เชื่อมต่อ MQTTโดยใช้คำสั่ง (ออกมา home directory ก่อนติดตั้ง)

   git clone https://github.com/eclipse/paho.mqtt.python.git
   cd paho.mqtt.python
   sudo python setup.py install

การทดลองที่ 6 ทดสอบการทำงานของ MQTT Broker

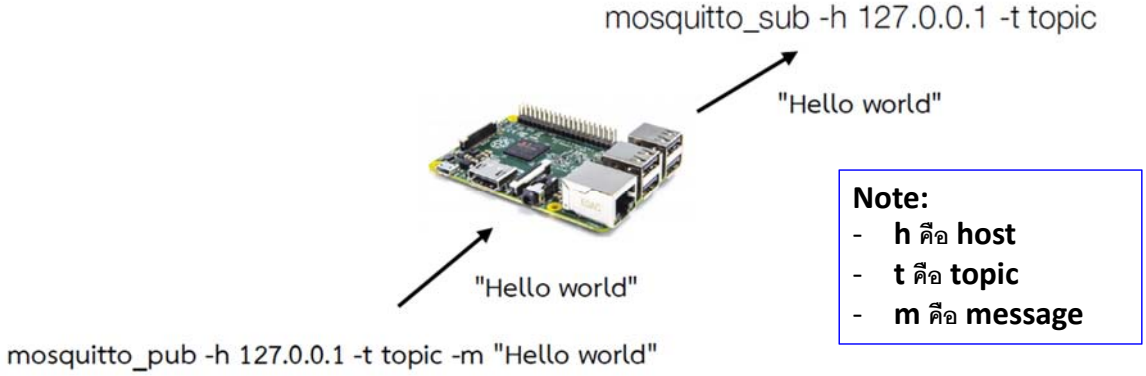
- เปิด Terminal หนึ่งเพื่อทำการ Subscribe โดยใช้คำสั่ง

  mosquitto_sub -h 127.0.0.1 -t topic

- และเปิดอีก Terminal หนึ่ง เพื่อทำการ Publish

  mosquitto_pub -h 127.0.0.1 -t topic -m "Hello world“

- เมื่อทำการ Publish แล้ว ให้สังเกตผลลัพธ์ที่หน้าจอ Subscribe



Note:
- **h คือ host**
- **t คือ topic**
- **m คือ message**

การเชื่อมต่อ Wi-Fi ของ Raspberry Pi

- ทำการ Scan หา SSID โดยใช้คำสั่ง sudo iwlist wlan0 scan
- ทำการ Config file system โดยใช้คำสั่ง

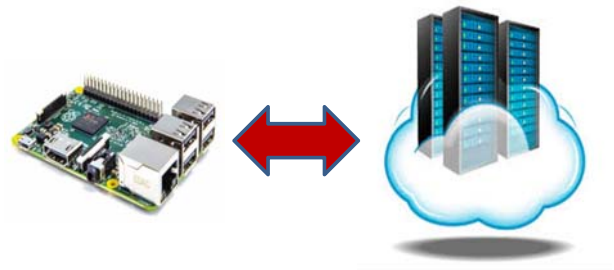
  sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
- พิมพ์ข้อมูลต่อไปนี้ลงไป

```
network={
        ssid="The_SSID_from_earlier"
        psk="Your_wifi_password"
}
```

- ออกจากโปรแกรม nano โดยการกด Ctrl+x และ save file โดยการกด yes
- หากต้องการดู IP Adress ใช้คำสั่ง ifconfig

แบบฝึกหัด

ให้ Raspberry Pi ทำการ Subscribe และ publish ไปที่ Broker คณะ



neutron.it.kmitl.ac.th, 1883
หรือ
161.246.38.194, 1883

ตัวอย่าง MQTT Publish โดยภาษา python

- พิมพ์คำสั่ง nano pub.py
- เขียนโปรแกรม python ดังนี้

```
import  paho.mqtt.client as mqtt
import  time

mqttc = mqtt.Client()
mqttc.connect("127.0.0.1", 1883)

while True:
        mqttc.publish("test/pub", "Hello")
        time.sleep(2)
```

- Run โปรแกรมโดยใช้คำสั่ง sudo python pub.py

Note: อาจจะใช้ mqttc.connect("neutron.it.kmitl.ac.th", 1883) สำหรับ publish ลง Broker คณะ หรือ mqttc.connect("161.246.38.194", 1883)

ตัวอย่าง MQTT Subscribe โดยภาษา python

- พิพม์คำสั่ง nano sub.py
- เขียนโปรแกรม python ดังนี้

```
import  paho.mqtt.client as mqtt

def on_connect(client, userdata, rc):
        print("Connected with result code "+str(rc))
        client.subscribe("test/sub")

def on_message(client, userdata, msg):
        print(msg.topic+" "+str(msg.payload))

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect("127.0.0.1", 1883)
client.loop_forever()
```

- Run โปรแกรมโดยใช้คำสั่ง sudo python sub.py

แบบฝึกหัด

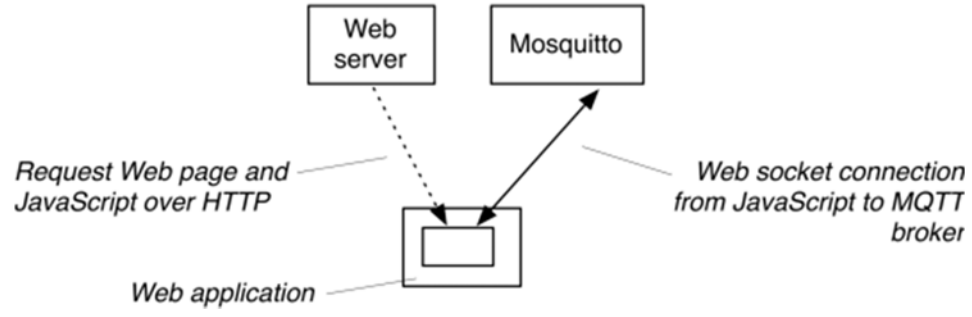
แบบฝึกหัดที่ 1.1 Raspberry Pi Publish ไปที่ Broker คณะ

- อ่านค่าจาก ปุ่มกด
- ถ้ากดปุ่ม ให้ Publish ไป Broker คณะ ว่า ON

แบบฝึกหัดที่ 1.2 Raspberry Pi ทำการ Subscribe จาก Broker คณะ

- รอรับค่าจาก Broker คณะ
- ถ้ารับได้คำว่า ON ให้ LED ติด
- ถ้ารับได้คำว่า OFF ให้ LED ดับ

MQTT broker gets Websockets support



มีองค์ประกอบคือ

- Web Server ทำหน้าที่เก็บไฟล์ html ที่มีโค้ด javascript
- Web Browser หรือ Web application ทำหน้าที่ เรียกไฟล์ html จาก Web Server มาแสดงผล
- MQTT Broker ที่เปิด Web Socket เพื่อให้ Javascript ไปดึงข้อมูล มาแสดงผล

MQTT broker gets Websockets support

ติดตั้ง Websocket บน Raspberry Pi

Download wget http://www.it.kmitl.ac.th/~panwit/mosquitto-ws.zip

ทำการ unzip ไฟล์

unzip mosquitto-ws.zip

เข้าไปยัง folder ที่ unzip ไฟล์

cd mosquitto-1_4-ws

ทำการติดตั้ง

chmod +x install
sudo service mosquitto stop
sudo ./install
sudo service mosquitto start

การทดลองที่ 7 Real Time graph

Simple example of plotting live numbered data from a subscribed mqtt/websockets topic and plotting it using highcharts. http://www.highcharts.com/

สิ่งที่ต้องมี คือ

- Web Server หรือ Web hosting
- ไฟล์ html ที่เขียนโค้ด Javascript

ไฟล์ที่เตรียมไว้ให้ คือ

livemqttchart.html	จะ plot กราฟ เมื่อมีการ Publish ค่า
livemqttchart1.html	จะ plot กราฟ แบบ real time
mqttws31.js	

ไฟล์ตัวอย่างที่ให้นี้ คือ ทำหน้าที่ Subscribe ตาม Topic ที่กำหนด และ รับค่ามาแสดงผล

การทดลองที่ 7 Real Time graph

1. แก้ไขไฟล์ livemqttchart.html ดังนี้

```
var MQTTbroker = 'IP ของ Raspi';      // Raspi broker
var MQTTport = 8888;                  // port ของ Webseocket ที่เปิดไว้
var MQTTsubTopic = 'temp';            //แก้ชื่อ topic ได้ตามใจชอบ
```

2. ทำการ Upload ไฟล์ทั้ง 2 ไปยัง web hosting
3. ทดสอบการแสดงผล โดยการ publish ค่าข้อมูล ไปยัง topic ที่กำหนด
   - การ Publish ค่า จะ Publish โดยการเปิดอีก Terminal หนึ่ง เพื่อทำการ Publish

```
mosquitto_pub -h 10.X.X.X -t temp -m "Hello world“
```

Note:
Raspberry เป็น Broker ที่มีค่า Default ของ Websocket คือพอร์ต 8888
สามารถแก้ไขได้ โดยไปแก้ที่ไฟล์ mosquitto.conf โดยใช้คำสั่ง sudo nano /etc/mosquitto/mosquitto.conf แล้วหาบรรทัด

```
# listener port-number [ip address/host name]
listener 8888
```

การทดลองที่ 8 Control via Websocket

ไฟล์ที่เตรียมไว้ให้ คือ

control.html

ไฟล์ตัวอย่างที่ให้นี้ คือ ทำหน้าที่ Publish ไปยัง Topic ที่กำหนด
เมื่อกดปุ่ม ON จะ publish ค่า on ไปยัง Raspi Broker
เมื่อกดปุ่ม OFF จะ publish ค่า off ไปยัง Raspi Broker

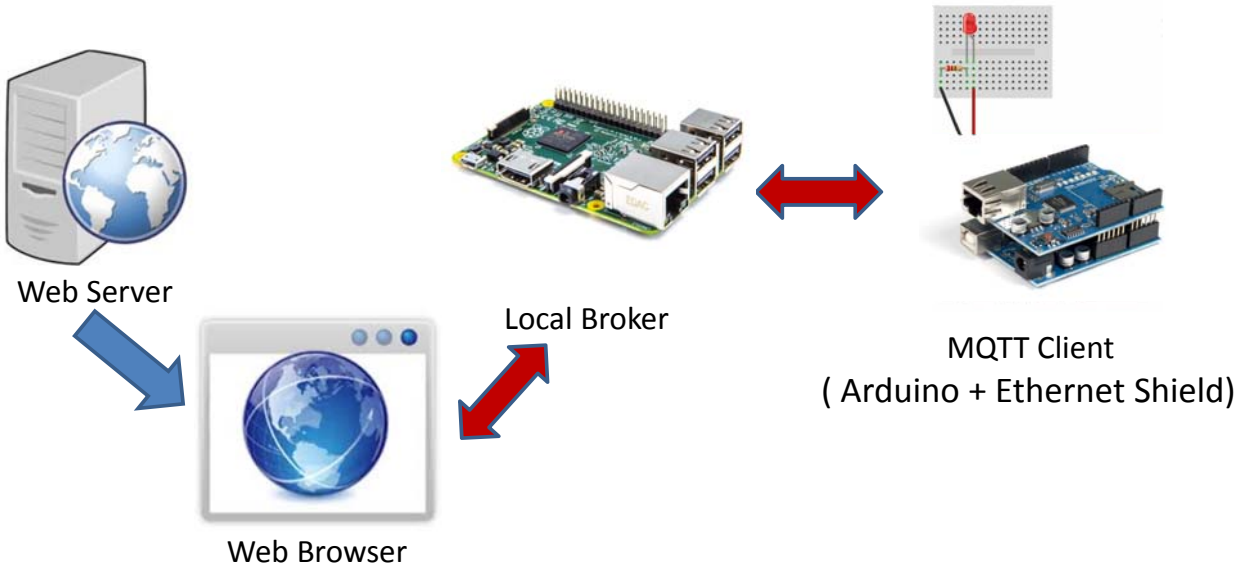
การทดลองที่ 8 Control via Websocket

1. แก้ไขไฟล์ control.html ดังนี้

```
var MQTTbroker = ' IP ของ Raspi ';          // broker ของคณะ
var MQTTport = 8888;                  // port ของ Webseocket ที่เปิดไว้
var MQTTsubTopic = 'temp';                  //แก้ชื่อ topic ได้ตามใจชอบ
```

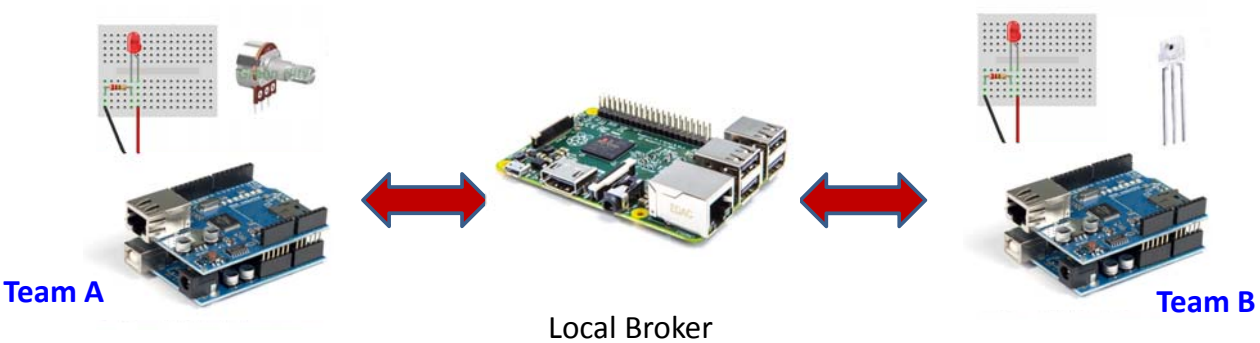
2. ทำการ Upload ไฟล์ ไปยัง web hosting
3. ทดสอบการแสดงผล โดยการ ใช้ Raspi Broker ทำการ Subscribe topic ที่กำหนด เมื่อกดปุ่มต่างๆ จะปรากฏค่า on, off และค่าตัวเลข ที่ Broker

แบบฝึกหัด



ณ แบบฝึกหัดนี้ ทุกท่านจะมีโครงสร้างระบบเป็นดังรูป
จงเขียนโปรแกรมให้ สามารถสั่ง ปิด-เปิด LED และหรี่ไฟ LED จาก website ได้
Note แก้ website ที่ไฟล์ control.html ด้วยนะครับ

แบบฝึกหัด : หลอด LED & R ปรับค่าได้



MQTT Client
(Arduino + Ethernet Shield)

Local Broker

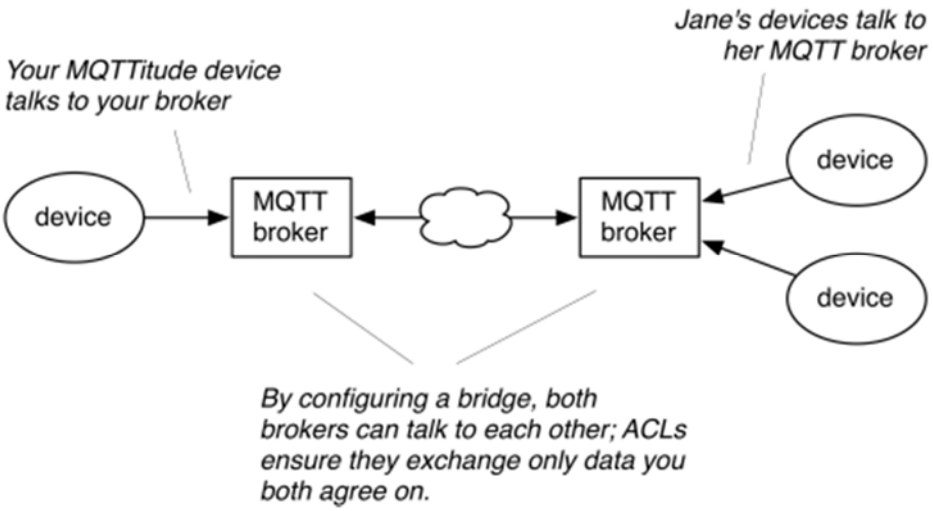
MQTT Client
(Arduino + Ethernet Shield)

วัตถุประสงค์ : ต้องการให้ Arduino ทั้งสองคุยกันผ่าน Internet

- ให้จับคู่ ทำด้วยกัน

1. ปรับตัวต้านทาน ที่ฝั่ง Team A แล้วให้ไป หรี่ไฟ ที่ฝั่ง Team B
2. ให้ฝั่ง Team B หนึ่งใช้ sensor แสง วัดแสง แล้วสั่งไปยัง Team A โดยที่
   - -เมื่อแสงน้อย ให้หลอดไฟ LED ติด
   - -เมื่อแสงมาก ให้หลอดไฟ LED ดับ

Bridge Broker



Configuring Bridges

- connection *name*

  This variable marks the start of a new bridge connection. It is also used to give the bridge a name which is used as the client id on the remote broker.

- address *address[:port] [address[:port]]*, addresses *address[:port] [address[:port]]*
  - Specify the address and optionally the port of the bridge to connect to. This must be given for each bridge connection. If the port is not specified, the default of 1883 is used. Multiple host addresses can be specified on the address config.

Configuring Bridges

- topic *pattern* [[[out | in | both] qos-level] local-prefix remote-prefix]
  - Define a topic pattern to be shared between the two brokers. Any topics matching the pattern (which may include wildcards) are shared. The second parameter defines the direction that the messages will be shared in, so it is possible to import messages from a remote broker using *in*, export messages to a remote broker using *out* or share messages in both directions. If this parameter is not defined, the default of ***out*** is used. The QoS level defines the publish/subscribe QoS level used for this topic and defaults to **0**.
  - The *local-prefix* and *remote-prefix* options allow topics to be remapped when publishing to and receiving from remote brokers. This allows a topic tree from the local broker to be inserted into the topic tree of the remote broker at an appropriate place.
  - For incoming topics, the bridge will prepend the pattern with the remote prefix and subscribe to the resulting topic on the remote broker. When a matching incoming message is received, the remote prefix will be removed from the topic and then the local prefix added.
  - For outgoing topics, the bridge will prepend the pattern with the local prefix and subscribe to the resulting topic on the local broker. When an outgoing message is processed, the local prefix will be removed from the topic then the remote prefix added.

การทดลองที่ 9 Bridge Broker Configuration

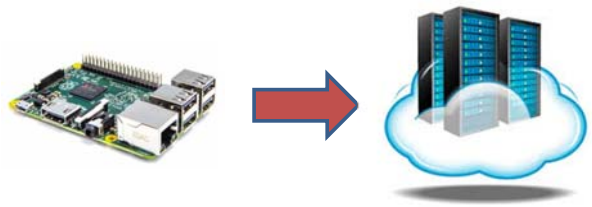
เป็นการทดลองเชื่อม Rasp pi เข้ากับ Broker ของคณะ แบบ Single Direction

- เปิดไฟล์ mosquitto.conf โดยใช้คำสั่ง

  sudo nano /etc/mosquitto/mosquitto.conf

  – แก้ไขไฟล์ดังนี้

  connection bridge_test
  address 161.246.38.194:1883
  topic local



- ทำการ Restart โดยใช้คำสั่ง

  sudo /etc/init.d/mosquitto stop
  sudo /etc/init.d/mosquitto start

Note:
ชื่อ Connection ต้องไม่ซ้ำกัน
Address คือ address ของปลายทางที่ไปเชื่อมต่อ

- ทดสอบโดยการ
  – Subscribe ที่ Broker คณะ ด้วย Topic local
  – Publish ไปที่ Raspberry Pi Broker ด้วย Topic local

การทดลองที่ 9 Bridge Broker Configuration

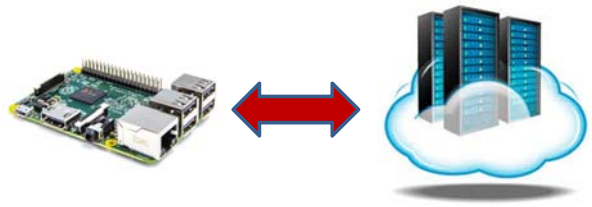
เป็นการทดลองเชื่อม Rasp pi เข้ากับ Broker ของคณะ แบบ Dual Direction

- เปิดไฟล์ mosquitto.conf โดยใช้คำสั่ง

  sudo nano /etc/mosquitto/mosquitto.conf

  – แก้ไขไฟล์ดังนี้

  connection bridge_test
  address 161.246.38.194:1883
  topic # both 2 local/topic/ remote/topic/



local remote

map an entire topic tree

- ทำการ Restart โดยใช้คำสั่ง

  sudo /etc/init.d/mosquitto stop
  sudo /etc/init.d/mosquitto start

- ทดสอบโดยการ
  – Subscribe ที่ Broker คณะ ด้วย Topic remote/topic
  – Publish ที่ Raspberry Pi ด้วย Topic local/topic

  – Subscribe ที่ Raspberry Pi ด้วย Topic local/topic
  – Publish ที่ Broker คณะ ด้วย Topic remote/topic

การทดลองที่ 9 Bridge Broker Configuration

เป็นการทดลองเชื่อม **Rasp pi** เข้ากับ **Broker** ของคณะ แบบ **Dual Direction**

- เปิดไฟล์ mosquitto.conf โดยใช้คำสั่ง

  sudo nano /etc/mosquitto/mosquitto.conf

  – แก้ไขไฟล์ดังนี้

  connection bridge_test
  address 161.246.38.29:1883
  topic # both 2 local/topic/ remote/topic/

local remote

map an entire topic tree

- ทำการ Restart โดยใช้คำสั่ง

  sudo /etc/init.d/mosquitto stop
  sudo /etc/init.d/mosquitto start

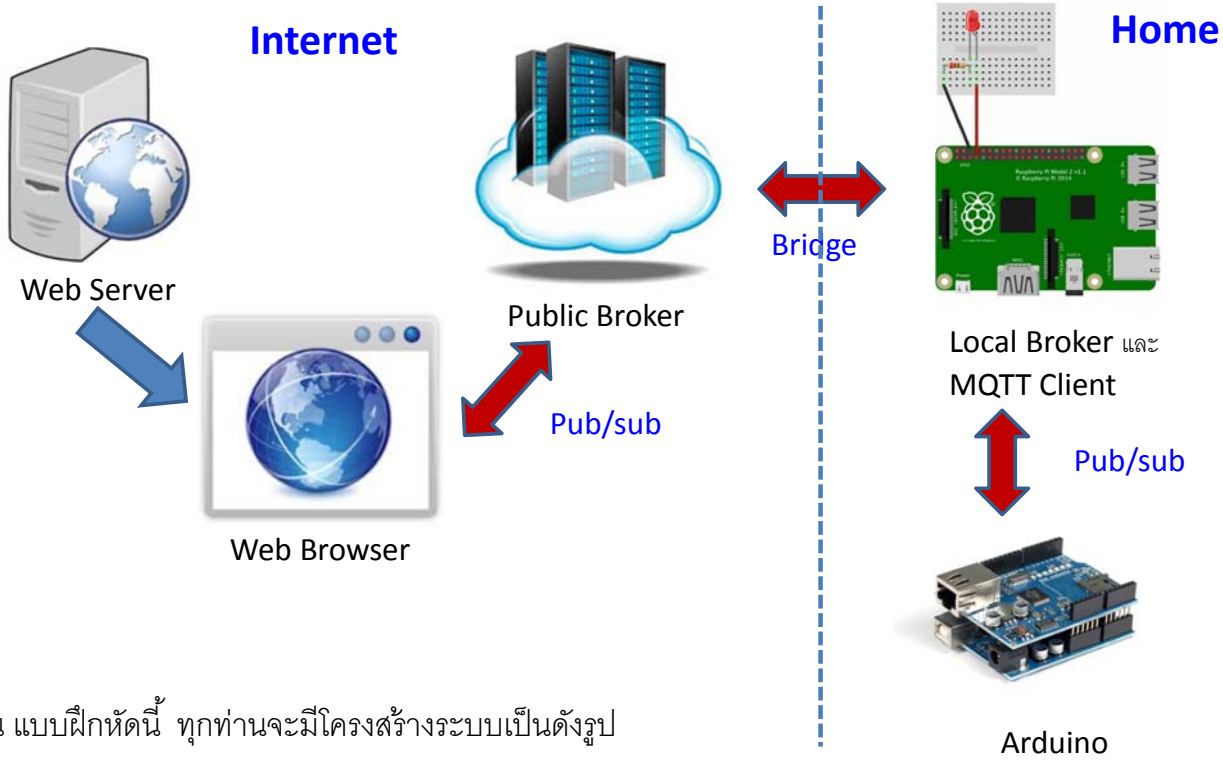
- ทดสอบโดยการ

  – Subscribe ที่ Broker คณะ ด้วย Topic remote/topic/test

  – Publish ที่ Raspberry Pi ด้วย Topic local/topic/test ด้วยคำสั่ง

  mosquitto_pub -h 127.0.0.1 -t local/topic/test -m "hello"

แบบฝึกหัด



ณ แบบฝึกหัดนี้ ทุกท่านจะมีโครงสร้างระบบเป็นดังรูป

แบบฝึกหัด

ณ แบบฝึกหัดนี้ ทุกท่านจะมีโครงสร้างระบบเป็นดังรูป

- จงเขียนโปรแกรมให้ Arduino คุยกับ Raspberry Pi โดยที่
  กดปุ่มที่ Raspberry Pi แล้วให้ LED ที่ Arduino ติด
  เมื่อปล่อยปุ่มกด ที่ Raspberry Pi ให้ LED ที่ Arduino ดับ

- จงเขียนโปรแกรมให้ Arduino ทำการ Publish ค่า Analog ที่อ่านได้
  ไปแสดงผลออกกราฟ real time graph ที่ Website (ทดสอบการ Bridge)

-จงเขียนโปรแกรมให้ Website มาควบคุมการปิด เปิด หรี่ไฟ ที่ Arduino ได้
(ทดสอบการ Bridge)