# ESP8266, MQTT, & openHab Show & Tell

Pete Keefe

March 10, 2016

# Physical Parts

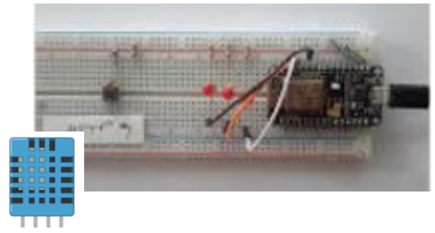
Club Ethernet

TP-Link WR841N Wireless Router
(running open source Gargoyle software)
ap: pkTest   pass: testpassword

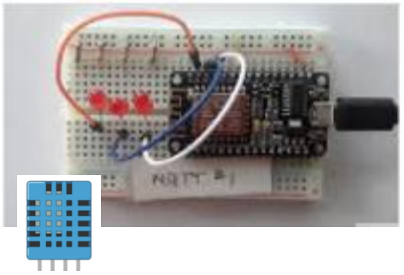
Raspberry Pi 2
Linux
openHab
MQTT broker
web server

Win7 Laptop
Browser,
Remote
Control
of Pi

ESP8266-12 NodeMcu Development
Model 2



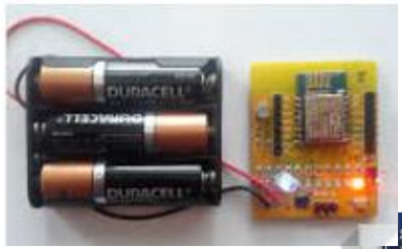
Web Server



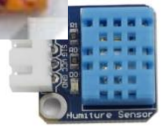
MQTT Client
ESP8266-1

ESP8266-12 NodeMcu Development Model 1

MQTT Client
ESP8266-2



MQTT Client
ESP8266-3



DHT11

## Raspberry Pi 2

- Quad-code 32bit 1 GHz processor with 1 GB memory, microSD card (32 GB in this case)
- Running Debian Linux (Jessie release downloaded from raspberrypi.org)
- Packages installed:
  - Apache Web Server
  - MySQL database
  - openHAB web server
  - OpenLibre (office suite)
  - Mosquitto MQTT Broker
  - Mosquitto Test Clients (_sub, _pub)

# NodeMcu Development Kits

- NodeMcu kits are open source / design
  - Normally firmware is Lua programming language
    - Lots of sample open source code available
  - Firmware for C language also available (although I have not used)
- Model 1
  - Includes battery pack & voltage regulator (3.3v)
  - Includes test lights, photo-resistor
  - Need extra usb to Serial interface board for program / debugging
- Model 2
  - Includes self contained usb connector for power and data
  - Includes voltage regulator (5v usb to 3.3v ESP8266)
  - Works well with standard breadboards
- Model 3 (un-official)
  - Caution:  does not work with standard breadboards

# ESP8266 Web Server

- Need to know IP address of the ESP8266 web server to call from browser:
  - http://192.168.0.168/Lights (get status)
  - http://192.168.0.168/Lights?1=ON (set light)
  - http://192.168.0.168/Temp (get temp)
  - http://192.168.0.168/Humidity (get)
- Server always waiting for request – can't tell anyone of problems / success unless asked

# ESP8266 HTTP Request & Response

- http://192.168.0.168/Lights
  - {"Lights": [{"Light":"5", "status":"OFF"},{"Light":"4", "status":"OFF"},{"Light":"0", "status":"OFF"}]}
- http://192.168.0.168/Lights?light5=1&light4=0
  - {"Lights": [{"Light":"5", "status":"ON"},{"Light":"4", "status":"OFF"},{"Light":"0", "status":"OFF"}]}
- http://192.168.0.168/Temp
  - {"Temp": "75.2" }
- http://192.168.0.168/Humidity
  - {"Humidity": "56" }

# MQTT

- Broker software running on Pi waits for clients
  - to connect and either send (publish) topic + data
  - Or to connect and subscribe (listen) to topic(s)
- Clients can both subscribe and publish
- Allows for multiple subscribers to same topic
- Topics used today:
  - ESP8266-1/IN/… messages being sent to ESP #1)
  - ESP9266-1/OUT/… messages from ESP #1

# MQTT ESP8266 Clients

- Listen for messages with topic - subscribes to ESP8266-1/IN/#
  - SetLight Data: 1=ON or 1=OFF
    - No response
  - GetLight Data: Light #
    - Response: ESP8266-1/OUT/Light Data: 1=ON
  - GetTemp
    - Response ESP8266-1/OUT/Temp Data: 99
  - GetHumidity
    - Response ESP8266-1/OUT/Humidity Data: 11
- Automatically sends out Temp, Humidity, and LightLevel messages on change in values
- At reboot – sends message
  - ESP8266-1/OUT/Restart
  - [Rule runs in openHAB to reset the status of lights to OFF.]
- Clients must know address of broker

# MQTT Test Clients

- MQTT Broker runs on Pi
  - Linux package called mosquitto
- Browser (sample html / Javascript copied from web site and heavily amended)
  - [http://raspberrypi/test.mosquitto.html](http://raspberrypi/test.mosquitto.html)
- On Pi
  - mosquitto_sub –v –t ESP8266-1/#
    - # is a wildcard to match multiple sub-levels
  - mosquitto_pub –t ESP8266-1/OUT/Lights –m ‘’

# Cost / Benefits – using mqtt

- Cost:
  - mqtt requires broker software to be running
- Benefits:
  - Notification immediate – no waiting for subscribers to poll each remote unit for updates
  - Client can send and receive messages
  - mqtt broker can store all or last message for each topic (Quality of Service & Message Retention options)
  - multiple programs can see same message (very helpful in debugging)

# openHAB

- Open source project for home automation
  - openHAB.org
  - Written in Java with many extensions (“bindings”) for connecting to products (i.e., Philips Hue, Wemo, etc)
  - Running as a web server on Pi at port 8080
  - Use browser plus apps for Android / Apple
- Configuration files
  - Sitemap defines layout
  - Items defines items and how connected / controlled
  - Rules define actions to take on data updates/changes
  - Transformations to extract or reform data
  - openhab.cfg defines services run automatically
    - i.e., use to collect weather data each 5 minutes

## openHAB

- Steep learning curve
  - Took over two weeks before started to make real sense
  - Learn to look to demo / examples configurations
  - Documentation apparently written by German program authors
- Some very good YouTube videos
- Demo