

Instalación Mosquitto en Raspberry Pi

Para instalar MQTT en una Raspberry Pi voy a utilizar un servidor ampliamente conocido como es [Eclipse Mosquitto](#).



Mosquitto es un mediador de mensajes que incluye el protocolo MQTT. Además es de código abierto lo que supone una ventaja para los Makers ya que lo podemos utilizar sin ningún tipo de problema.

Para que el protocolo MQTT esté en constante disponibilidad, es recomendable instalar el broker en un servidor que esté siempre encendido. Tenemos diferentes opciones pero la más interesante es Raspberry Pi por su bajo coste y consumo.

Vamos a ver los pasos que deben seguir para tener operativo nuestro servidor MQTT.

Debes tener instalado Raspbian y recomendable la versión Jessie. En la página web de la [Fundación Raspberry Pi](#) tienes un tutorial paso a paso.

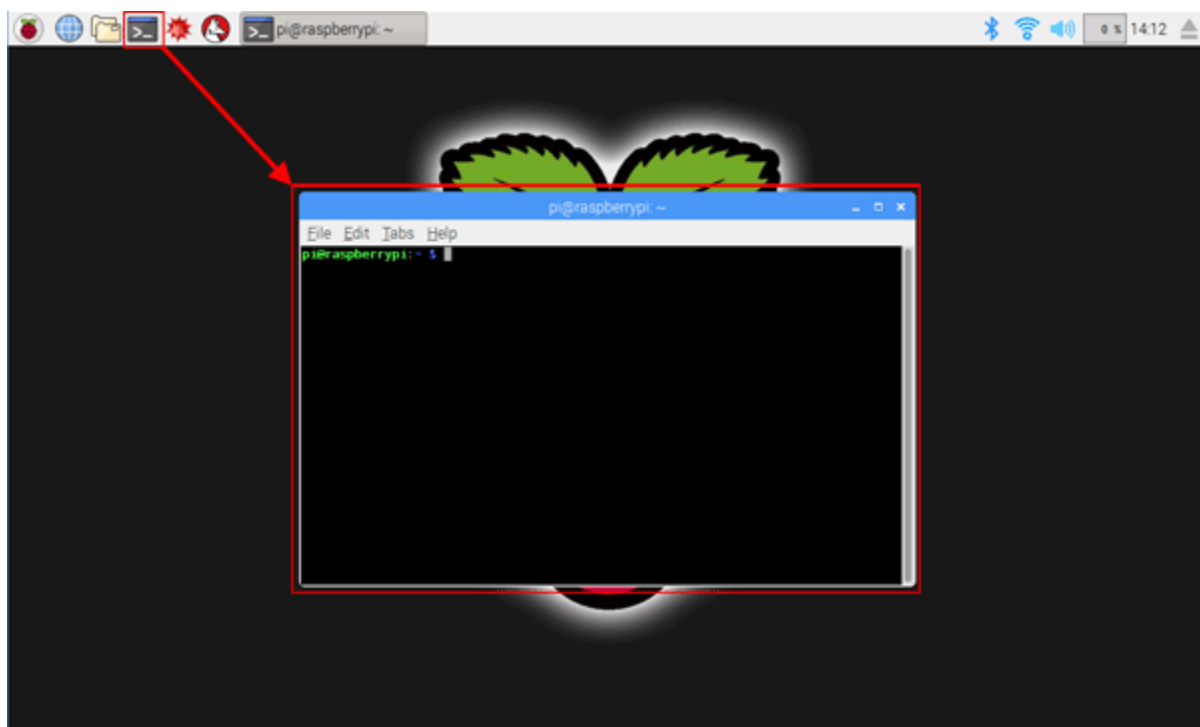
Paso 1: Instalar Broker MQTT

Si ya tienes experiencia con Raspberry Pi conocerás *apt-get*. Se trata de una herramienta para gestionar paquetes instalables a través de la línea de comandos. Desafortunadamente no podemos utilizar sólo esta herramienta para instalar Mosquitto MQTT.

Esto es debido a que si lo hacemos, no nos instalará la última versión del software y nos generará errores de compatibilidad de software.

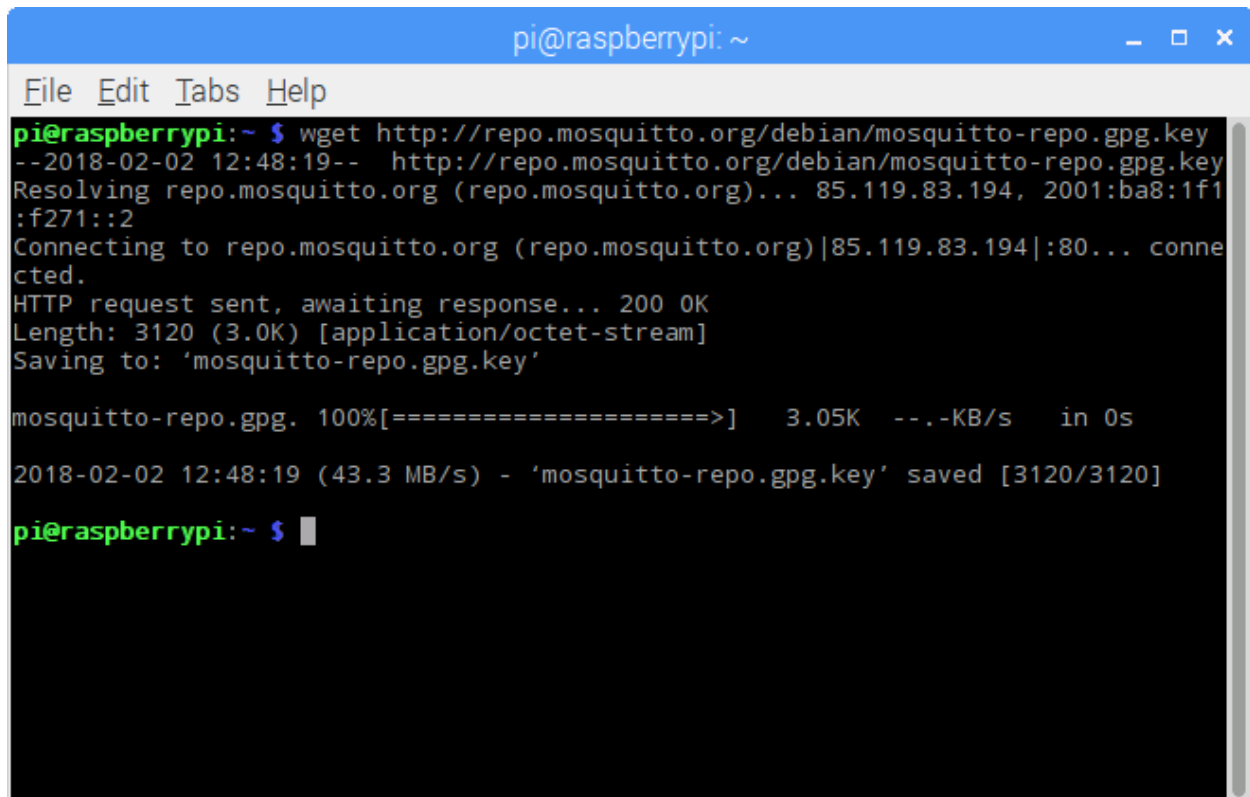
La Raspberry Pi tiene que estar conectada a Internet para poder instalar el Broker MQTT Mosquitto.

Abre una terminal en tu Raspberry Pi.



Lo primero es descargar la signing key o clave de firma utilizando el comando *wget*. Este comando descarga el fichero indicado como parámetro en el directorio en el que te encuentras.

sudo wget <http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key>



```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key  
--2018-02-02 12:48:19-- http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key  
Resolving repo.mosquitto.org (repo.mosquitto.org)... 85.119.83.194, 2001:ba8:1f1:f271::2  
Connecting to repo.mosquitto.org (repo.mosquitto.org)|85.119.83.194|:80... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 3120 (3.0K) [application/octet-stream]  
Saving to: 'mosquitto-repo.gpg.key'  
  
mosquitto-repo.gpg. 100%[=====>] 3.05K ---KB/s in 0s  
2018-02-02 12:48:19 (43.3 MB/s) - 'mosquitto-repo.gpg.key' saved [3120/3120]  
  
pi@raspberrypi:~ $
```

Añadimos la clave para a una lista para autenticar el paquete que vamos a descargar más tarde.

```
sudo apt-key add mosquitto-repo.gpg.key
```

```
pi@raspberrypi: ~  
File Edit Tabs Help  
pi@raspberrypi:~ $ wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key  
--2018-02-02 12:48:19-- http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key  
Resolving repo.mosquitto.org (repo.mosquitto.org)... 85.119.83.194, 2001:ba8:1f1:f271::2  
Connecting to repo.mosquitto.org (repo.mosquitto.org)|85.119.83.194|:80... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 3120 (3.0K) [application/octet-stream]  
Saving to: 'mosquitto-repo.gpg.key'  
  
mosquitto-repo.gpg. 100%[=====>] 3.05K --.-KB/s in 0s  
2018-02-02 12:48:19 (43.3 MB/s) - 'mosquitto-repo.gpg.key' saved [3120/3120]  
  
pi@raspberrypi:~ $ sudo apt-key add mosquitto-repo.gpg.key  
OK  
pi@raspberrypi:~ $
```

Luego tienes que ir a la siguiente carpeta utilizando el comando `cd`.

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

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
pi@raspberrypi:~ $ wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
--2018-02-02 12:48:19-- http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
Resolving repo.mosquitto.org (repo.mosquitto.org)... 85.119.83.194, 2001:ba8:1f1:f271::2
Connecting to repo.mosquitto.org (repo.mosquitto.org)|85.119.83.194|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3120 (3.0K) [application/octet-stream]
Saving to: 'mosquitto-repo.gpg.key'

mosquitto-repo.gpg. 100%[=====] 3.05K --.-KB/s in 0s
2018-02-02 12:48:19 (43.3 MB/s) - 'mosquitto-repo.gpg.key' saved [3120/3120]

pi@raspberrypi:~ $ sudo apt-key add mosquitto-repo.gpg.key
OK
pi@raspberrypi:~ $ cd /etc/apt/sources.list.d/
pi@raspberrypi:/etc/apt/sources.list.d $
```

Después descargamos la lista de repositorios de Mosquitto con *wget*.

Si tu versión de Raspbian es Jessie, ejecuta el siguiente comando.

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

Si tu versión de Raspbian es Stretch, ejecuta el siguiente comando.

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

Si utilizas la última versión, Buster, utiliza el siguiente comando.

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

Como ves, si utilizas una versión anterior o más reciente, lo único que tienes que hacer es cambiar el nombre la versión.

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help

mosquitto-repo.gpg. 100%[=====>] 3.05K ---KB/s in 0s
2018-02-02 12:48:19 (43.3 MB/s) - 'mosquitto-repo.gpg.key' saved [3120/3120]

pi@raspberrypi:~ $ sudo apt-key add mosquitto-repo.gpg.key
OK
pi@raspberrypi:~ $ cd /etc/apt/sources.list.d/
pi@raspberrypi:/etc/apt/sources.list.d $ sudo wget http://repo.mosquitto.org/debian/mosquitto-wheezy.list
--2018-02-02 12:50:19-- http://repo.mosquitto.org/debian/mosquitto-wheezy.list
Resolving repo.mosquitto.org (repo.mosquitto.org)... 85.119.83.194, 2001:ba8:1f1:f271::2
Connecting to repo.mosquitto.org (repo.mosquitto.org)|85.119.83.194|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 49 [application/octet-stream]
Saving to: 'mosquitto-wheezy.list'

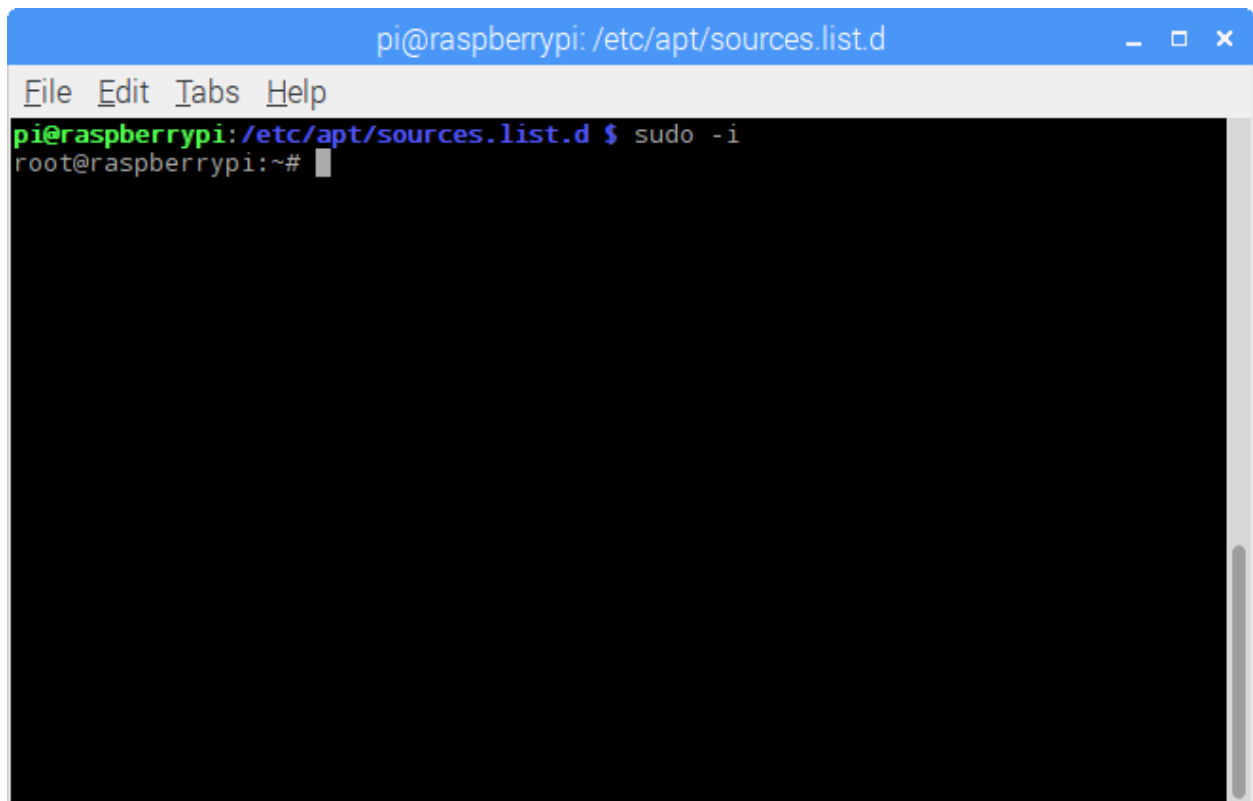
mosquitto-wheezy.li 100%[=====>] 49 ---KB/s in 0s
2018-02-02 12:50:19 (1.10 MB/s) - 'mosquitto-wheezy.list' saved [49/49]

pi@raspberrypi:/etc/apt/sources.list.d $
```

Ahora para no tener que estar constantemente poniendo la palabra *sudo*, escribimos lo siguiente en la terminal para ser usuario *root*.

```
sudo -i
```

Este paso no es obligatorio, si no lo haces tendrás que poner *sudo* en los comandos que van a continuación.

A terminal window titled 'pi@raspberrypi: /etc/apt/sources.list.d' with standard window controls. The menu bar shows 'File', 'Edit', 'Tabs', and 'Help'. The terminal text shows the user 'pi' at the prompt 'pi@raspberrypi:/etc/apt/sources.list.d' typing 'sudo -i'. The prompt changes to 'root@raspberrypi:~#' indicating a successful switch to root privileges. The rest of the terminal area is black with a white cursor.

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
pi@raspberrypi:/etc/apt/sources.list.d $ sudo -i
root@raspberrypi:~#
```

Como ves, la shell cambia de aspecto. A partir de ahora ya no hace falta poner la palabra *sudo* antes.

Actualizamos la lista de paquetes disponibles y sus versiones.

```
apt-get update
```

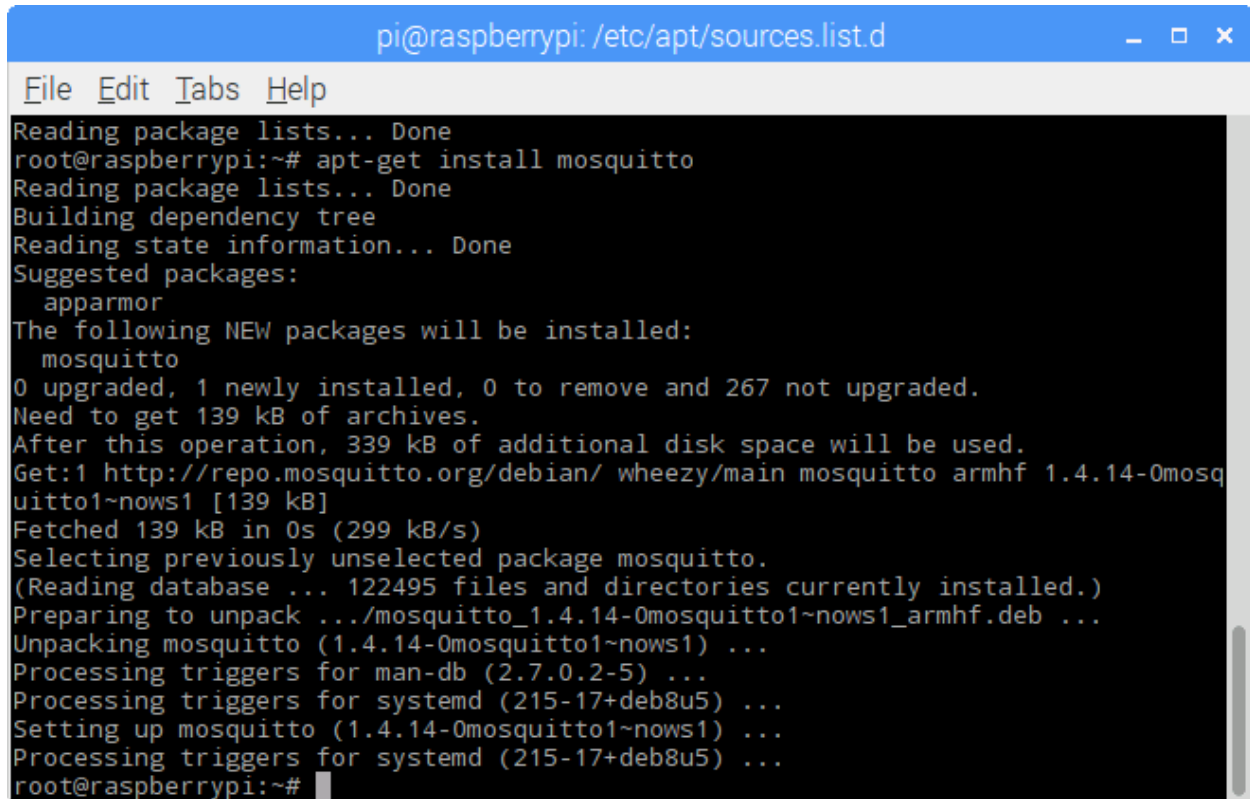
Esto puede tardar un rato así que ten paciencia.

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
pi@raspberrypi:/etc/apt/sources.list.d $ sudo -i
root@raspberrypi:~# apt-get update
Get:1 http://mirrordirector.raspbian.org jessie InRelease [14.9 kB]
Get:2 http://archive.raspberrypi.org jessie InRelease [22.9 kB]
Get:3 http://repo.mosquitto.org wheezy InRelease [3,893 B]
Get:4 http://mirrordirector.raspbian.org jessie/main armhf Packages [9,536 kB]
Get:5 http://repo.mosquitto.org wheezy/main armhf Packages [2,457 B]
Get:6 http://archive.raspberrypi.org jessie/main armhf Packages [171 kB]
Ign http://repo.mosquitto.org wheezy/main Translation-en_GB
Ign http://repo.mosquitto.org wheezy/main Translation-en
Hit http://archive.raspberrypi.org jessie/ui armhf Packages
Get:7 http://mirrordirector.raspbian.org jessie/contrib armhf Packages [43.3 kB]
Ign http://archive.raspberrypi.org jessie/main Translation-en_GB
Get:8 http://mirrordirector.raspbian.org jessie/non-free armhf Packages [84.2 kB]
Ign http://archive.raspberrypi.org jessie/main Translation-en
Ign http://archive.raspberrypi.org jessie/ui Translation-en_GB
Get:9 http://mirrordirector.raspbian.org jessie/rpi armhf Packages [1,356 B]
Ign http://archive.raspberrypi.org jessie/ui Translation-en
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en
Ign http://mirrordirector.raspbian.org jessie/main Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/main Translation-en
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en_GB
```

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
Get:5 http://repo.mosquitto.org wheezy/main armhf Packages [2,457 B]
Get:6 http://archive.raspberrypi.org jessie/main armhf Packages [171 kB]
Ign http://repo.mosquitto.org wheezy/main Translation-en_GB
Ign http://repo.mosquitto.org wheezy/main Translation-en
Hit http://archive.raspberrypi.org jessie/ui armhf Packages
Get:7 http://mirrordirector.raspbian.org jessie/contrib armhf Packages [43.3 kB]
Ign http://archive.raspberrypi.org jessie/main Translation-en_GB
Get:8 http://mirrordirector.raspbian.org jessie/non-free armhf Packages [84.2 kB]
Ign http://archive.raspberrypi.org jessie/main Translation-en
Ign http://archive.raspberrypi.org jessie/ui Translation-en_GB
Get:9 http://mirrordirector.raspbian.org jessie/rpi armhf Packages [1,356 B]
Ign http://archive.raspberrypi.org jessie/ui Translation-en
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en
Ign http://mirrordirector.raspbian.org jessie/main Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/main Translation-en
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en
Ign http://mirrordirector.raspbian.org jessie/rpi Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/rpi Translation-en
Fetched 9,880 kB in 31s (314 kB/s)
Reading package lists... Done
root@raspberrypi:~#
```

Ejecuta el siguiente comando para instalar el Broker Mosquitto.

apt-get install mosquitto

A screenshot of a terminal window titled 'pi@raspberrypi: /etc/apt/sources.list.d'. The terminal shows the command 'apt-get install mosquitto' being executed. The output indicates that the package lists were read, a dependency tree was built, and the state information was read. It suggests installing 'apparmor' and lists 'mosquitto' as a new package to be installed. It shows that 139 kB of archives are needed and that 339 kB of additional disk space will be used. The package is then fetched from the repository 'http://repo.mosquitto.org/debian/ wheezy/main mosquitto armhf 1.4.14-0mosquitto1~nows1 [139 kB]'. Finally, the package is unpacked and triggers are processed for 'man-db' and 'systemd'. The terminal ends with the prompt 'root@raspberrypi:~#'.

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
Reading package lists... Done
root@raspberrypi:~# apt-get install mosquitto
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  apparmor
The following NEW packages will be installed:
  mosquitto
0 upgraded, 1 newly installed, 0 to remove and 267 not upgraded.
Need to get 139 kB of archives.
After this operation, 339 kB of additional disk space will be used.
Get:1 http://repo.mosquitto.org/debian/ wheezy/main mosquitto armhf 1.4.14-0mosq
uito1~nows1 [139 kB]
Fetched 139 kB in 0s (299 kB/s)
Selecting previously unselected package mosquitto.
(Reading database ... 122495 files and directories currently installed.)
Preparing to unpack .../mosquitto_1.4.14-0mosquitto1~nows1_armhf.deb ...
Unpacking mosquitto (1.4.14-0mosquitto1~nows1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Processing triggers for systemd (215-17+deb8u5) ...
Setting up mosquitto (1.4.14-0mosquitto1~nows1) ...
Processing triggers for systemd (215-17+deb8u5) ...
root@raspberrypi:~#
```

Con esto ya tendríamos el Broker Mosquitto instalado en nuestra Raspberry Pi. Ahora toca instalar el cliente para hacer las pruebas.

Paso 2: Instalar cliente MQTT en Raspberry Pi

Este paso va a ser muy sencillo. Si todavía no has cerrado la terminal anterior sólo tendrás que escribir el siguiente comando.

apt-get install mosquitto-clients

Recuerda que si no estás logueado como *root* tendrás que poner antes del comando *sudo*.

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
After this operation, 339 kB of additional disk space will be used.
Get:1 http://repo.mosquitto.org/debian/ wheezy/main mosquitto armhf 1.4.14-0mosq
uitto1~nows1 [139 kB]
Fetched 139 kB in 0s (299 kB/s)
Selecting previously unselected package mosquitto.
(Reading database ... 122495 files and directories currently installed.)
Preparing to unpack .../mosquitto_1.4.14-0mosquitto1~nows1_armhf.deb ...
Unpacking mosquitto (1.4.14-0mosquitto1~nows1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Processing triggers for systemd (215-17+deb8u5) ...
Setting up mosquitto (1.4.14-0mosquitto1~nows1) ...
Processing triggers for systemd (215-17+deb8u5) ...
root@raspberrypi:~# apt-get install mosquitto-clients
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libmosquitto1
The following NEW packages will be installed:
  libmosquitto1 mosquitto-clients
0 upgraded, 2 newly installed, 0 to remove and 267 not upgraded.
Need to get 110 kB of archives.
After this operation, 276 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

En un momento dado te hará la siguiente pregunta «Do you want to continue? (¿Quieres continuar?)». Tienes que dar a la tecla «y» y al *Enter*.

```
pi@raspberrypi: /etc/apt/sources.list.d
File Edit Tabs Help
libmosquitto1
The following NEW packages will be installed:
libmosquitto1 mosquitto-clients
0 upgraded, 2 newly installed, 0 to remove and 267 not upgraded.
Need to get 110 kB of archives.
After this operation, 276 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://repo.mosquitto.org/debian/ wheezy/main libmosquitto1 armhf 1.4.14-0
mosquitto1~nows1 [52.7 kB]
Get:2 http://repo.mosquitto.org/debian/ wheezy/main mosquitto-clients armhf 1.4.
14-0mosquitto1~nows1 [57.4 kB]
Fetched 110 kB in 0s (227 kB/s)
Selecting previously unselected package libmosquitto1:armhf.
(Reading database ... 122526 files and directories currently installed.)
Preparing to unpack .../libmosquitto1_1.4.14-0mosquitto1~nows1_armhf.deb ...
Unpacking libmosquitto1:armhf (1.4.14-0mosquitto1~nows1) ...
Selecting previously unselected package mosquitto-clients.
Preparing to unpack .../mosquitto-clients_1.4.14-0mosquitto1~nows1_armhf.deb ...
Unpacking mosquitto-clients (1.4.14-0mosquitto1~nows1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Setting up libmosquitto1:armhf (1.4.14-0mosquitto1~nows1) ...
Setting up mosquitto-clients (1.4.14-0mosquitto1~nows1) ...
Processing triggers for libc-bin (2.19-18+deb8u6) ...
root@raspberrypi:~#
```

Con esto ya tendríamos todo lo necesario para hacer nuestro primer ejemplo.

Paso 3: Ejemplo con Mosquitto MQTT

Ahora mismo si tu Raspberry Pi está conectada a la red de tu casa, puedes crear un topic accesible desde cualquier cliente MQTT que esté conectado al Broker.

De momento, lo único que queremos es probar que todo está bien instalado así que lo haremos sobre la misma Raspberry Pi.

Pero antes de continuar vamos a ver los dos comandos que vamos a utilizar del cliente MQTT Mosquitto, *mosquitto_sub* y *mosquitto_pub*.

Comando mosquitto_sub

Este comando nos permite suscribirnos a un topic escuchando y mostrando por pantalla los mensajes enviados a este topic. Puede [tomar muchos parámetros](#) pero nosotros nos vamos a centrar en 4.

```
mosquitto_sub -h BROKER -t TOPIC
```

Donde:

- -h: indica que lo que viene después es el host. El host se refiere a la dirección IP o nombre de la máquina en la red del Broker Mosquitto. Si no pasamos este parámetro cogerá por defecto localhost.
- BROKER: dirección IP o nombre de la máquina en la red del Broker Mosquitto.
- -t: indica que lo que viene después es el topic al que queremos suscribirnos.
- TOPIC: nombre del topic al que nos vamos a suscribir.

Comando mosquitto_pub

Con este comando podemos publicar mensajes muy simples. Como ocurre con *mosquitto_sub* puede tomar muchos parámetros, este comando pero a nosotros nos importan sólo 6.

```
mosquitto_pub -h BROKER -t TOPIC -m MENSAJE
```

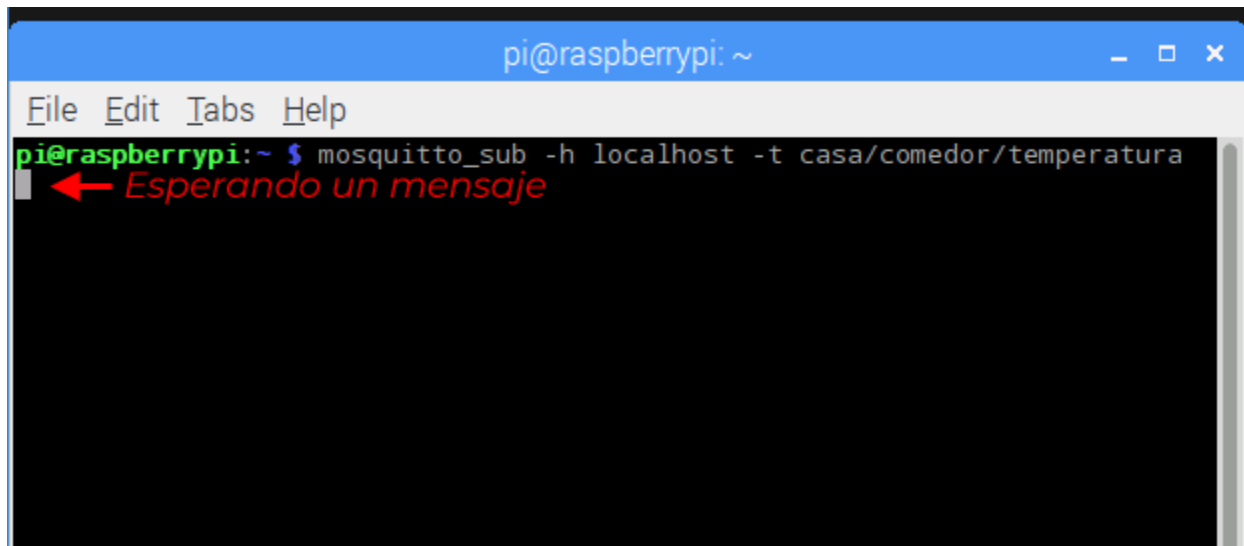
Donde:

- -h: indica que lo que viene después es el host. El host se refiere a la dirección IP o nombre de la máquina en la red del Broker Mosquitto. Si no pasamos este parámetro cogerá por defecto localhost.
- BROKER: dirección IP o nombre de la máquina en la red del Broker Mosquitto.
- -t: indica que lo que viene después es el topic al que queremos enviar el mensaje.
- TOPIC: nombre del topic al que vamos a enviar el mensaje.
- -m: indica que lo que viene después es el mensaje.
- MENSAJE: el mensaje que queremos enviar. Tiene que ir entre comillas dobles («). Por ejemplo «Temperatura = 25°C».

Ahora sólo nos queda probar así que abre dos terminales en la Raspberry Pi.

Uno va a ser el que publique y otro el que esté escuchando. Elige el que quieras y escribe el siguiente comando.

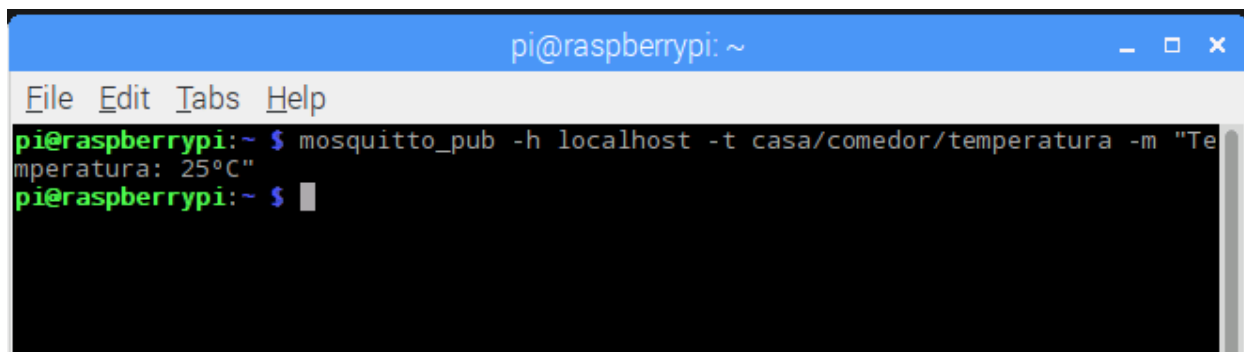
```
mosquitto_sub -h localhost -t casa/comedor/temperatura
```

A terminal window titled 'pi@raspberrypi: ~' with a menu bar containing 'File', 'Edit', 'Tabs', and 'Help'. The command 'mosquitto_sub -h localhost -t casa/comedor/temperatura' has been entered. Below the command, a red arrow points to the text 'Esperando un mensaje'.

La terminal se queda esperando a recibir algún mensaje.

En el otro terminal escribe el siguiente comando que envía un mensaje al topic casa/comedor/temperatura.

```
mosquitto_pub -h localhost -t casa/comedor/temperatura -m "Temperatura: 25°C"
```

A terminal window titled 'pi@raspberrypi: ~' with a menu bar containing 'File', 'Edit', 'Tabs', and 'Help'. The command 'mosquitto_pub -h localhost -t casa/comedor/temperatura -m "Temperatura: 25°C"' has been entered. Below the command, the output 'Temperatura: 25°C' is displayed.

Ahora si te fijas en la terminal donde estaba esperando un mensaje aparece precisamente el que hemos enviado.

A screenshot of a terminal window titled 'pi@raspberrypi: ~'. The window has a menu bar with 'File', 'Edit', 'Tabs', and 'Help'. The terminal shows a command prompt 'pi@raspberrypi:~ \$' followed by the command 'mosquitto_sub -h localhost -t casa/comedor/temperatura'. Below the command, the message 'Temperatura: 25°C' is displayed. A red arrow points from the text 'Mensaje recibido' to the message 'Temperatura: 25°C'.

```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ mosquitto_sub -h localhost -t casa/comedor/temperatura
Temperatura: 25°C

```

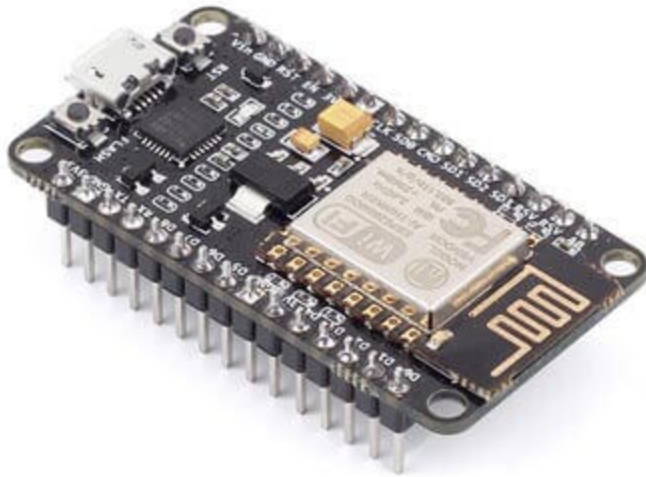
Puedes seguir jugando con los comodines que hemos visto anteriormente. Utiliza los símbolos + y #.

Por último es aconsejable que nuestra Raspberry Pi tenga una IP estática o fija. Esto quiere decir que aunque se desconecte de la red, siempre tendrá la misma IP. Esto lo puedes hacer siguiendo el tutorial de [ModMyPi](#).

Y así de fácil hemos conseguido crear una red con el protocolo MQTT. Ahora lo que haremos es poder encender un LED conectado a un [NodeMCU](#) desde la Raspberry Pi.

Enviando y recibiendo mensajes MQTT con NodeMCU

Lo primero de todo es que tienes que tener una placa con un ESP8266 o ESP32.



Ahora sí, ya podemos empezar a trabajar con NodeMCU y MQTT. Lo primero será instalar la librería.

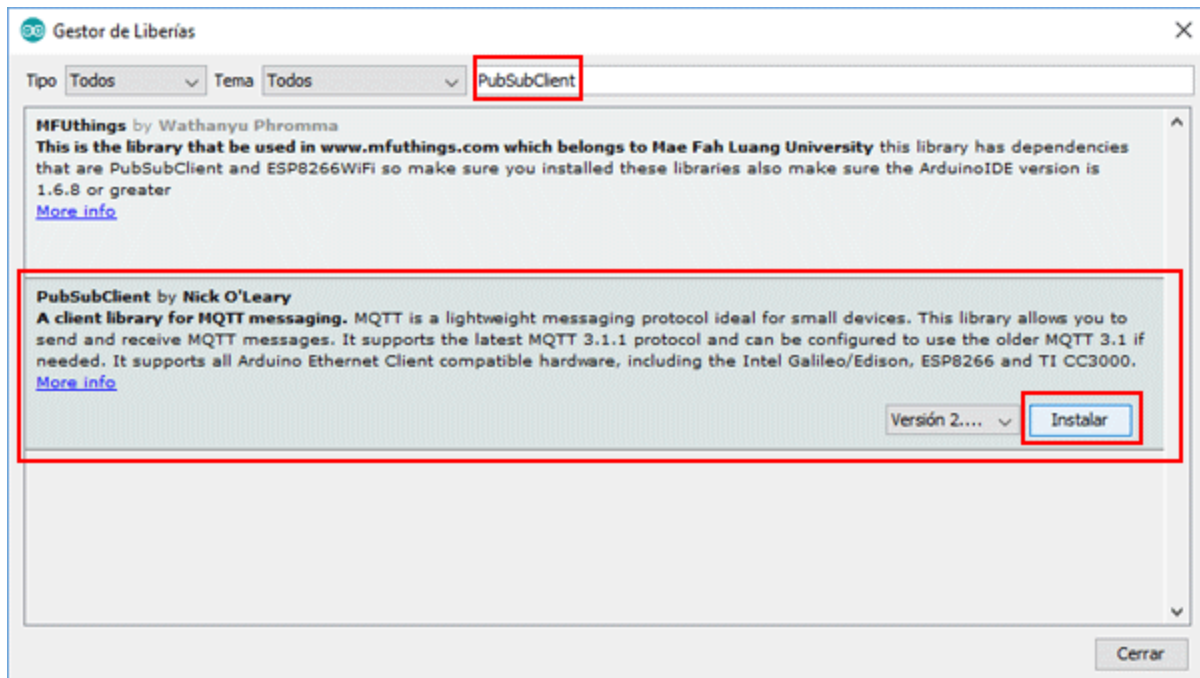
Instalando la librería PubSubClient

Una de las grandes ventajas que tenemos a la hora de prototipar con Arduino o ESP8266, es que tenemos un montón de librerías que nos hacen la vida mucho más fácil. Si no tienes experiencia con librerías, te aconsejo que eches un vistazo al tutorial donde explico cómo [instalar librerías en el IDE de Arduino](#).

PubSubClient es una librería compatible con Arduino y ESP8266. Básicamente hace que nuestra placa se comporte como un cliente MQTT es decir, que podamos publicar mensajes y suscribirnos a un topic o varios para recibir mensajes.

Da lo mismo si utilizas un Arduino o un ESP8266, el código es prácticamente el mismo. La diferencia reside en cómo nos conectamos a la red WiFi, cada placa utiliza su propia librería.

Lo primero es instalar la librería siguiendo. Abre el la opción del menú *Programa>Incluir Librería>Gestor de librerías* y busca PubSubClient



Luego le das a instalar y listo, ya tienes la librería instalada en tu ordenador y preparada para ser utilizada.

Enviando un mensaje a través del protocolo MQTT con NodeMCU

Vamos a partir de uno de los ejemplos que vienen dentro de la librería. Elige el adecuado dependiendo de la tarjeta a utilizar. Lo encontrarás en *Archivo > Ejemplos > PubSubClient > mqtt_esp8266*. Esta opción te abre el siguiente código.

Vamos a ir viendo las modificaciones que tenemos que hacer para enviar un mensaje con el protocolo MQTT desde NodeMCU.

Paso 1: poner ssid, password y mqtt_server

Sustituye por los valores de tu red WiFi y de tu servidor o Broker MQTT. Como lo tenemos instalado en la Raspberry Pi, lo único que tienes que hacer es entrar en una línea de comandos y poner lo siguiente

```
ifconfig
```

Esto te mostrará algo parecido a lo siguiente.


```
pi@raspberrypi: ~  
File Edit Tabs Help  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:1000  
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)  
  
lo    Link encap:Local Loopback  
      inet addr:127.0.0.1  Mask:255.0.0.0  
      inet6 addr: ::1/128 Scope:Host  
      UP LOOPBACK RUNNING MTU:65536 Metric:1  
      RX packets:32578 errors:0 dropped:0 overruns:0 frame:0  
      TX packets:32578 errors:0 dropped:0 overruns:0 carrier:0  
      collisions:0 txqueuelen:1  
      RX bytes:15523893 (14.8 MiB) TX bytes:15523893 (14.8 MiB)  
  
wlan0 Link encap:Ethernet  Hwaddr b8:27:eb:e7:64:af  
      inet addr: 192.168.0.167  Bcast:192.168.0.255  Mask:255.255.255.0  
      inet6 addr: fe80::9a16:373e:ba2b:7af2/64 Scope:Link  
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
      RX packets:193968 errors:0 dropped:175578 overruns:0 frame:0  
      TX packets:13386 errors:0 dropped:0 overruns:0 carrier:0  
      collisions:0 txqueuelen:1000  
      RX bytes:70424946 (67.1 MiB) TX bytes:5563696 (5.3 MiB)  
  
pi@raspberrypi:~ $
```

En mi caso, la IP de mi Raspberry Pi es la 192.168.0.167

Por lo tanto el código que debes cambiar quedaría algo parecido a esto.

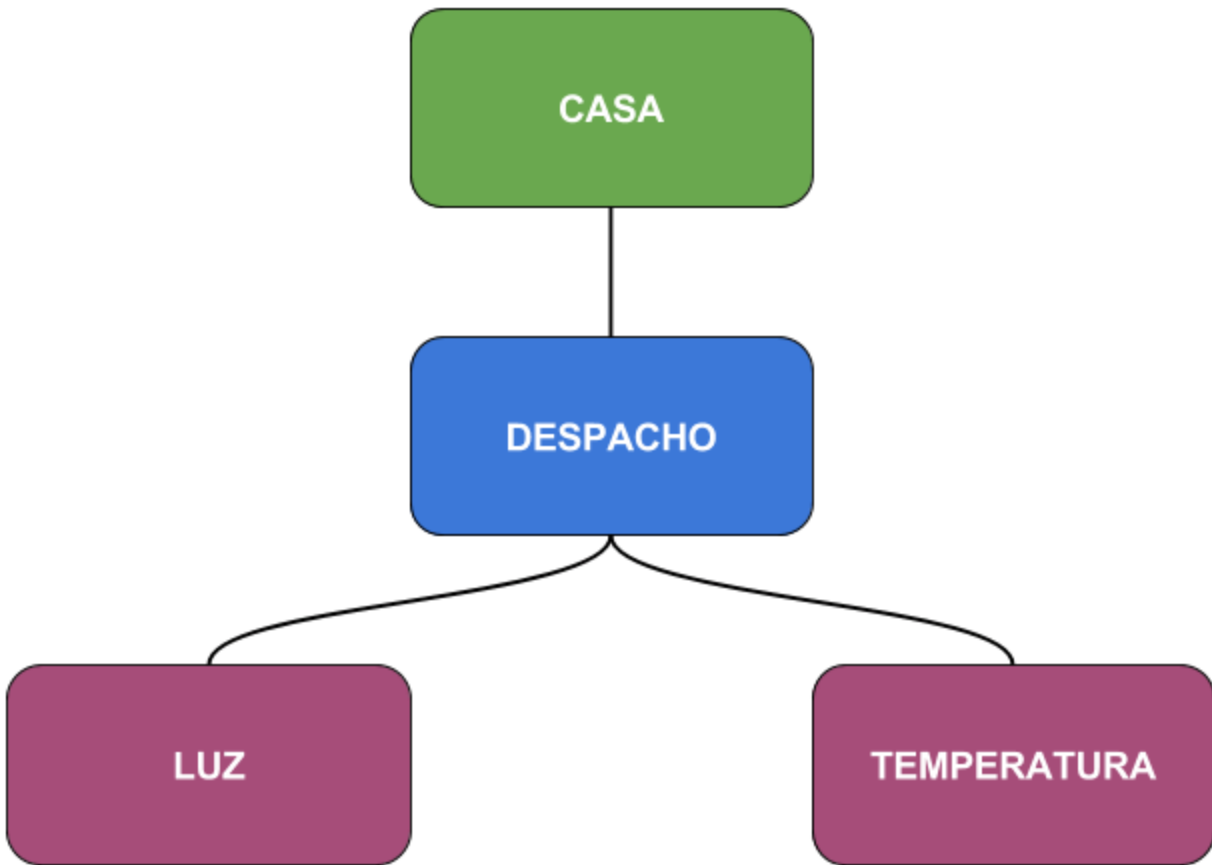
```
const char* ssid = "WIFILUIS";
```

```
const char* password = "123456";
```

```
const char* mqtt_server = "192.168.0.167";
```

Paso 2: cambiar el topic donde enviamos el mensaje

Vamos a comenzar cambiando los topic que vienen por defecto. El esquema de nuestra red MQTT será el siguiente:



Antes de continuar déjame que te explique este esquema. El nivel superior es la casa. De aquí partirán el resto de niveles. En este caso sólo tenemos uno que es el despacho pero podría haber más niveles como habitación, cocina, salón, etc...

Del despacho salen dos subniveles: luz y temperatura. Lo lógico es suscribirse a dos topic:

- casa/despacho/luz
- casa/despacho/temperatura

Como hemos visto antes, existe una gran variedad de topic si utilizamos los comodines + y #. Pero en este ejemplo no los vamos a utilizar.

Por otro lado tenemos que tener claro cual es la función de cada uno de estos topic. Por ejemplo, el topic *casa/despacho/luz* debería ser un actuador y por lo

tanto, el dispositivo que se encargue de controlar la luz debería estar suscrito y escuchando los mensajes que llegan a ese topic.

Puede ser un [Arduino MKR1000](#), un [ESP8266](#) o cualquier otra placa con conexión a la red. En esta placa podemos conectar un relé que controle una luz. Al recibir un mensaje en el topic *casa/despacho/luz*, actuará en consecuencia.

Si es un 1 activará el [relé](#) y si es un 0 lo apagará. Así de sencillo.

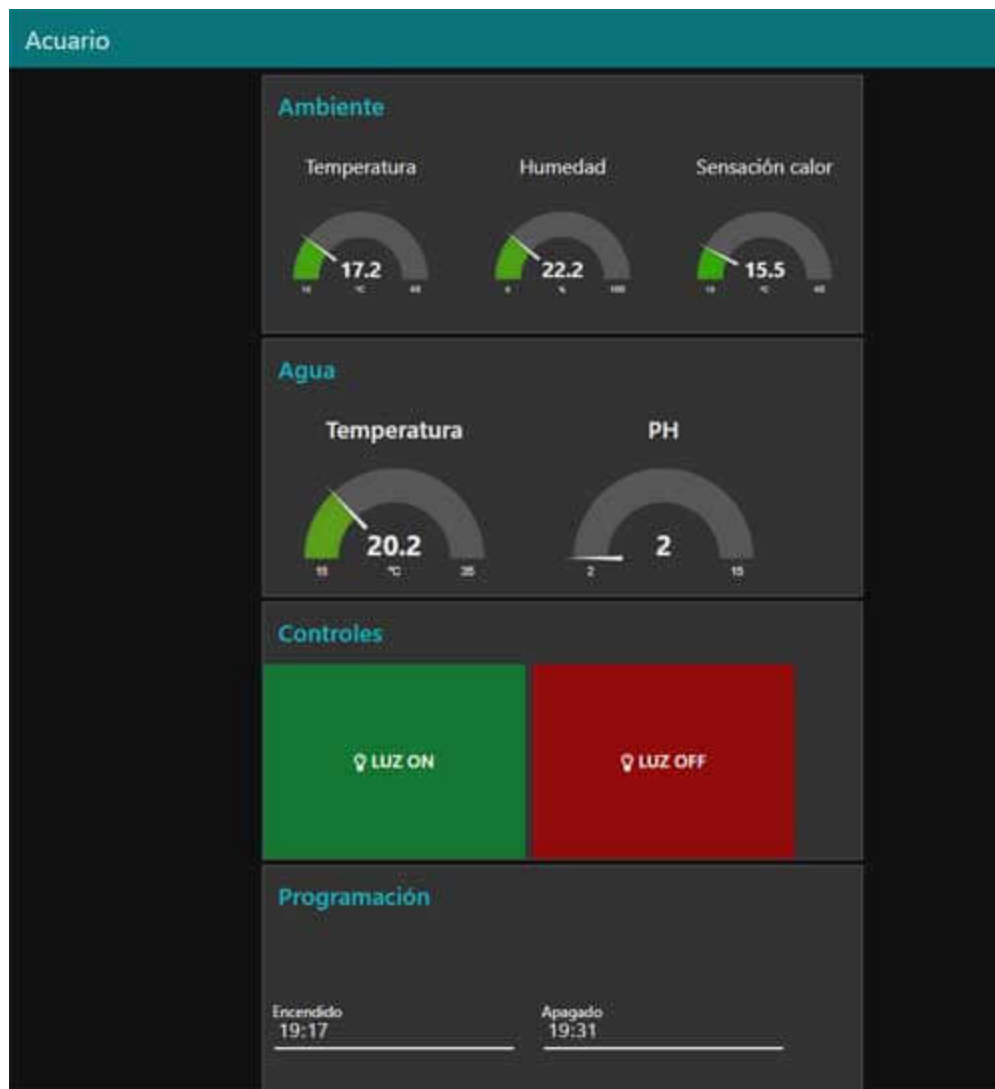
A la vez, el mismo dispositivo puede publicar la temperatura en el topic *casa/despacho/temperatura*. Ahora sólo nos falta saber quién publicará en el topic de la luz y quién recibirá los mensajes de temperatura.

Ese otro dispositivo puede ser cualquier cosa. Imagínate una Raspberry Pi que sea capaz de trabajar como cliente MQTT. Se pueden hacer aplicaciones con muchos lenguajes que tienen librerías o frameworks para conectarnos vía MQTT.

Como hemos visto al configurar Mosquitto con Raspberry Pi, hemos instalado un cliente MQTT. Por lo tanto, Raspberry Pi se comportará como Broker y como cliente.

Seremos capaces de enviar un mensaje a NodeMCU para que encienda el LED. Por otro lado, nos suscribiremos al topic de temperatura para leer la información que envíe NodeMCU.

Todo esto lo podemos hacer fácilmente con Node-RED aunque esto se sale fuera de este tutorial. Si quieres aprender cómo hacerlo, no te pierdas esta [introducción a Node-RED](#).



Panel de control Acuario con Node-RED, Wemos Mini D1 y protocolo MQTT

La idea final es que un topic puede lanzar un evento como actuador o como sensor. Lo típico es que los actuadores se suscriban y escuchen mensajes. Por otro lado los sensores publicarán en un topic los datos que van adquiriendo.

En este ejemplo, NodeMCU se suscribirá al *topic casa/despacho/luz* y publicará en *casa/despacho/temperatura* los datos de temperatura. Raspberry Pi se suscribirá al *topic casa/despacho/temperatura* y enviará mensajes a *casa/despacho/luz*.

Ahora sólo nos falta cambiar los topic. Ves a la línea 98 y sustituye esto

```
client.publish("outTopic", "hello world");
```

Por esto

```
client.publish("casa/despacho/temperatura", "Enviando el primer mensaje");
```

En esta línea estamos diciendo que publique un mensaje (*Enviando el primer mensaje*) en el topic *casa/despacho/temperatura*.

Luego en la línea 100 sustituye esto

```
client.subscribe("inTopic");
```

Por esto

```
client.subscribe("casa/despacho/luz");
```

En esta línea estamos diciendo que se suscriba al topic *casa/despacho/luz*.

Sólo nos falta cambiar la línea 125. Sustituye esto

```
client.publish("outTopic", msg);
```

Por esto

```
client.publish("casa/despacho/temperatura", msg);
```

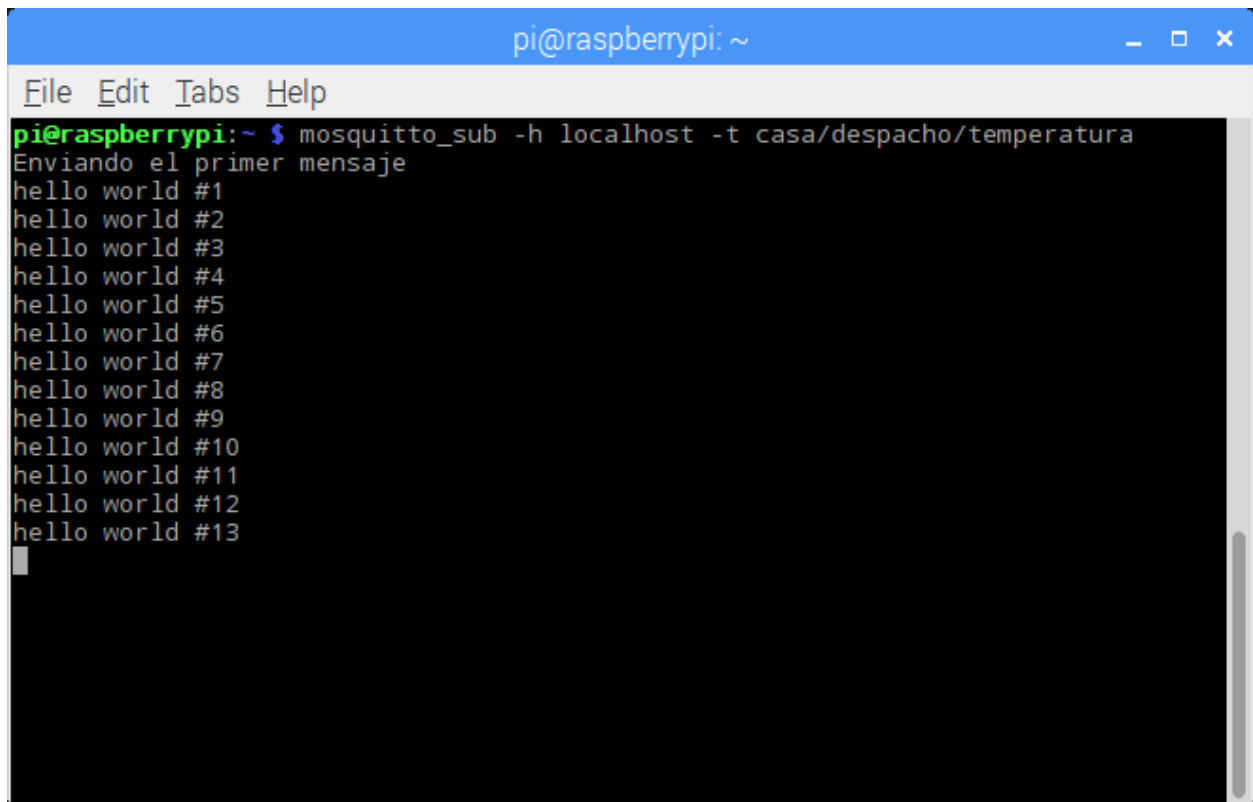
Probando la aplicación MQTT con ESP8266 y Raspberry Pi

Por último, nos queda probar todo el sistema. No te olvides de cargar el código en la placa con las modificaciones hechas.

Abre una terminal en la Raspberry Pi y escribe el siguiente comando.

```
mosquitto_sub -h localhost -t casa/despacho/temperatura
```

Si ya has cargado el código comprobarás cómo estás recibiendo mensajes desde NodeMCU. Tendrás algo parecido a esto.

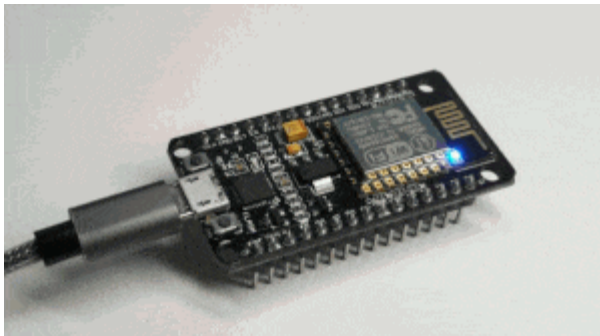
A screenshot of a terminal window titled 'pi@raspberrypi: ~'. The window has a menu bar with 'File', 'Edit', 'Tabs', and 'Help'. The terminal shows the command 'mosquitto_sub -h localhost -t casa/despacho/temperatura' being executed. Below the command, it says 'Enviando el primer mensaje' followed by a list of 13 'hello world' messages, each with a sequence number from #1 to #13.

```
pi@raspberrypi:~ $ mosquitto_sub -h localhost -t casa/despacho/temperatura
Enviando el primer mensaje
hello world #1
hello world #2
hello world #3
hello world #4
hello world #5
hello world #6
hello world #7
hello world #8
hello world #9
hello world #10
hello world #11
hello world #12
hello world #13
```

Abre otro terminal y escribe el siguiente comando.

```
mosquitto_pub -h localhost -t casa/despacho/luz -m 0
```

Si el LED que vienen incorporado en la placa estaba encendido, se apagará.



Si escribes este otro comando

```
mosquitto_pub -h localhost -t casa/despacho/luz -m 1
```

Encenderás el LED. Así de sencillo.

Conclusiones proyecto MQTT

Los objetivos de MQTT son optimizar el ancho de banda, minimizar los recursos a nivel de hardware y hacer que las comunicaciones sean fiables.

La típica arquitectura de un sistema MQTT se basa en una red con topología de estrella donde hay un nodo central llamado Broker por el que pasan todas las comunicaciones. Los clientes se conectan al Broker para recibir y enviar mensajes.

Un topic es el tema al que se suscriben los clientes y el tema al que se pueden enviar mensajes.

Por último, hemos puesto en práctica lo aprendido montando el Broker MQTT Mosquitto en una Raspberry Pi. Utilizando como cliente tanto la Raspberry Pi como un NodeMCU hemos visto cómo comunicar estos dos dispositivos y enviar mensajes entre ellos.

Se trata de un protocolo que tiene muchas posibilidades en el movimiento Maker y dentro del IoT. Te aconsejo que sigas practicando y nos cuentes, en los comentarios, qué te parece MQTT.