Arduino Manual para IoT

(openFramework)

Arduino+Mosquitto+NodeREd +Freeboard = Visualizacion Data en tiempo real



freeboard

arduino

Smartphonene

Mqtt Broker

arduino

# Guía de instalación de un framework abierto para IoT framework/dashboard con NodeRed, Mosquitto y Freeboard.io dashboard

Basado en el tutorial de primalcortex (febrero 25, 2015)

Conocimientos básicos Requeridos:

Arduino IDE

Programación básica

Programación con sensores.

Programación con librería #include <ArduinoJson.h> // <https://github.com/bblanchon/ArduinoJson/releases/tag/v5.0.7>

Haber realizado los ejemplos por defecto para Comunicaciones por medio de clientes TCP (wifi/GSM/ETH)

Haber realizado los ejemplos de las librerías de #include <PubSubClient.h> // <https://github.com/knolleary/pubsubclient/releases/tag/v2.3>

Conocimiento básico de Linux

Conocimiento básico de Git

Conocimientos básicos de nodejs

Conocimiento básico de Apache2

Symlimks

Debido a su bajo costo se utilizara un chip esp8266 para el desarrollo del tutorial (cualquiera de las versiones de hardware disponibles esta bien)

Diagrama 1.0 (Modulo de ESP8266, NodeMCU v3) ($6 USD)

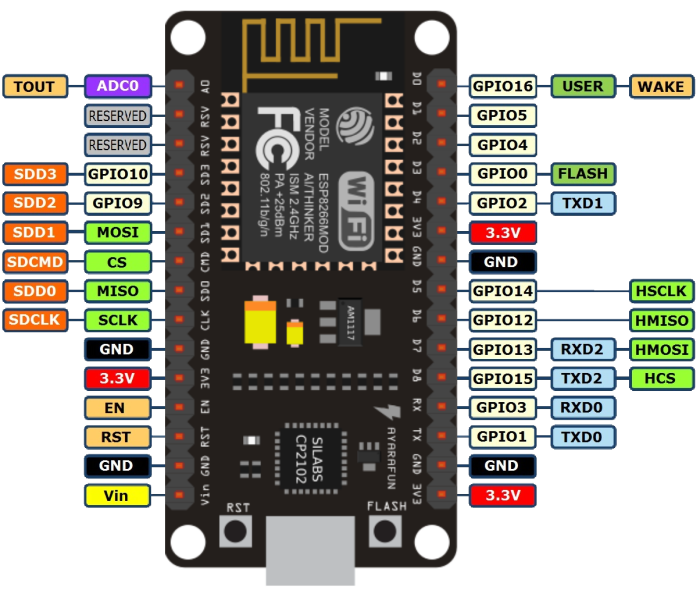
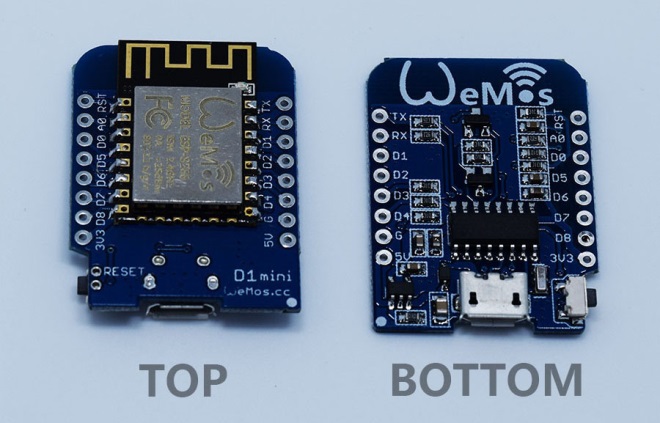
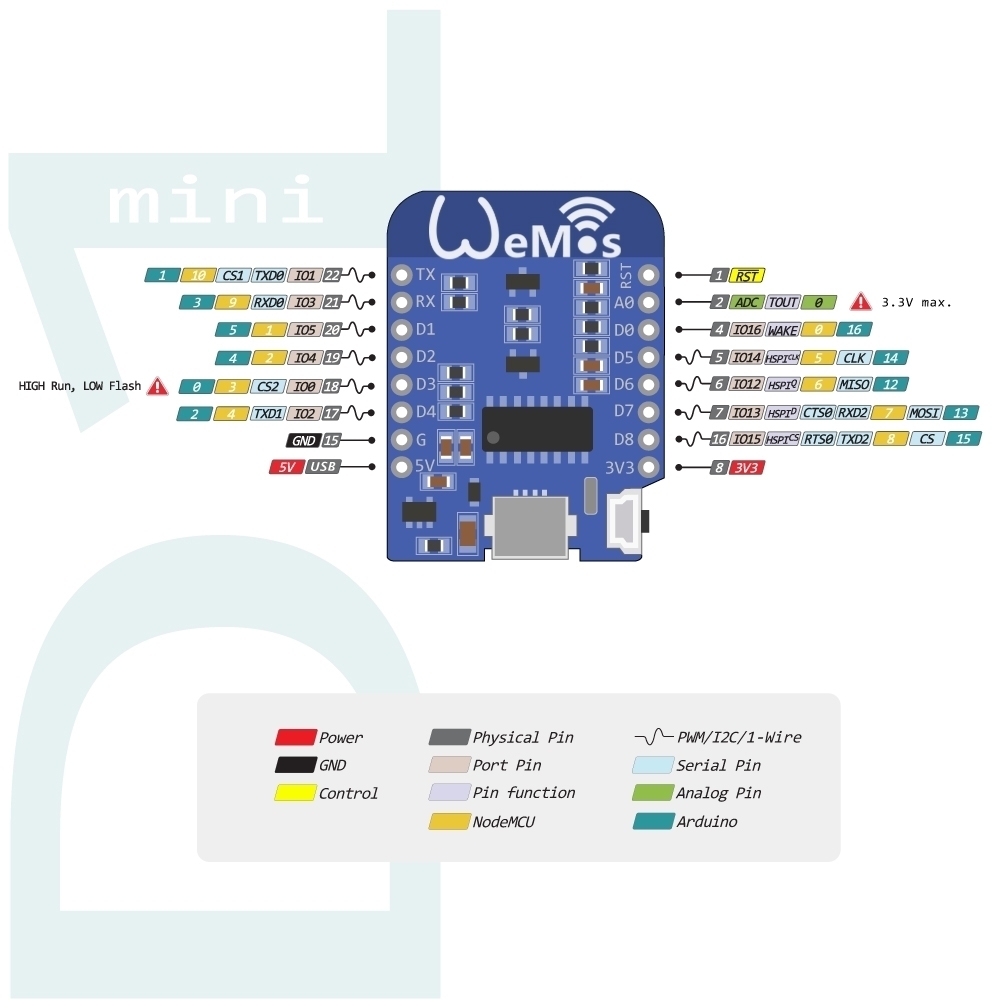


Diagrama 1.1 (Modulo de ESP8266, wemos D1 mini) ($4 USD)





Para este tutorial se recomienda una máquina virtual (local o en la nube) con los siguientes prerrequisitos:

Ubuntu14.04-64 Minimal for VSI:

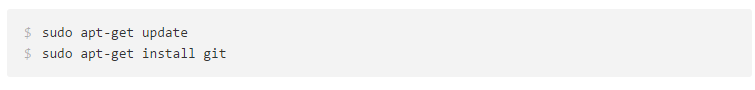
Hardware:

1x2.66 GHz 2GB RAM 25GBHDD 100Mps 250MB Bandwidth

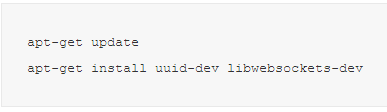
SSH enabled

IPV 4 IP address

Software:

Git:Configurado con tu cuenta de Git.

libwebsockets-dev:



Node.js:







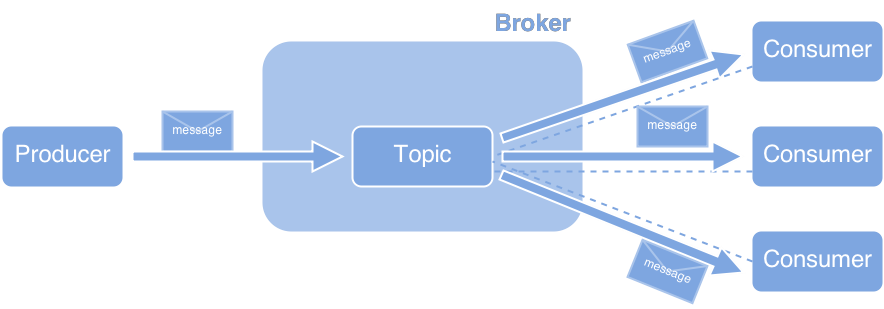
En la muchos de los casos también deberás instalar npm:



HTTPD - Apache2 Web Server:



Que es un open framwork para IoT, es un conjunto de programas con licencias de código abierto los cuales nos ayudaran a capturar, estructurar y presentar los datos que transmitamos al internet desde nuestros módulos de Arduino. Existen diversos protocolos de IoT como AMQP, MQTT, STOMP, para este caso en particular estaremos utilizando MQTT para conocer a profundidad sobre el mismo pueden leer más en : <http://mqtt.org/> este es un simple servicio de publicar suscripción:



El software que estaremos usando dentro de Linux será:

**IBM NodeRed:** (<http://nodered.org/>) Node-RED nos provee de una interfaz basada en un navegador web que nos permite crear flujos de eventos e interconectarlos todos ellos a través de un ligero entorno de trabajo y desarrollo. Está construido en node.js, lo que le permite funcionar justo al borde de la red o en la nube, dotándole de una notable flexibidad. El ecosistema del gestor de paquetes de node (npm) puede ser usado para extender de forma sencilla la paleta de nodos disponibles, permitiendo conexiones entre nuevos dispositivos y servicios. Node-RED ha sido desarrollado como un proyecto open-source en GitHub y está bajo una licencia Apache 2

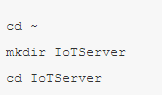
.**MQTT Broker Mosquitto:** (http://mosquitto.org/) Mosquitto es un corridor de Mqtt para linux el cual es liviano y facil de instalar

**Freeboard.io Dashboard:** (<http://freeboard.io/>) es un motor dinámico de tableros opensource project GitHub page (<https://github.com/Freeboard/freeboard>). El cual nos permite diseñar y administrar los datos capturados por el corredor de MQTT por medio de WEbSockets

**Paso 1: instalando NodeRed:**

Primero verificamos que se encuentre instlado node.js y npm con el commandos **node -v** and **npm -v** .

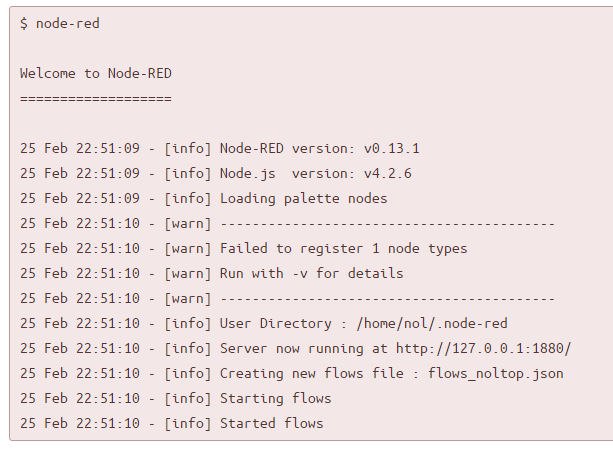
Luego creamos nuestra estructura de archivos llamado IOTServer en el cual mantendremos todas nuestras configuraciones de software.



Luego corremos desde el bash:



Ahora podemos correr el servidor de nodered con el comando: **node red.js -v.  utilizamos el** -v al final para saber si existe algún error o si algún modulo se encuentra mal instalado:



Podremos verificar el funcionamiento del nuestro sitio de nodered en la dirección :

[http://localhost:1880](http://localhost:1880/)

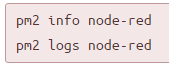
Podemos encontrar más información sobre como configurar NodeRed en <http://nodered.org/docs/configuration.html>  ahora podemos empezar a usar nodered con: **node red.js -v -s settings.js**

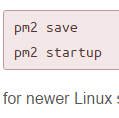
En node red podemos diseñar el flujo de información para la data del ESP8266 recibida por el MQTT e incluso elaborar un servicio simple REST basado en HTTP protocolo, procesar la data y almacenarla.

Para hacer que el servicio corra desde el boot:







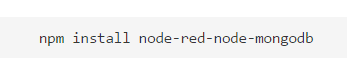


Reboot!!!

Para almacenar la data en una base de datos podemos usar MongoDB instalando los módulos

Que hacen falta de la siguiente manera:

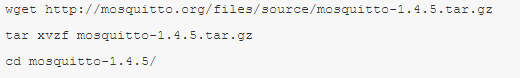
En el directorio ~/ .node-red



**Paso2: Inatalar Mosquitto MQTT con websocket**

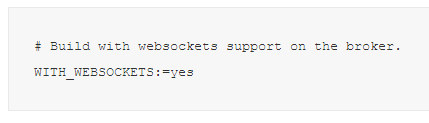
Bajar la última imagen del repositorio oficial:



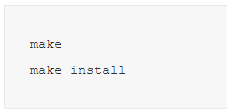


**NOTA Cambiar wget al ultimo reléase del mosquitto actualmente mosquitto-1.4.8.tar.gz**

Editar el archivo config.mk para habilitar el soporte a websockets:



Y completar la instalación:

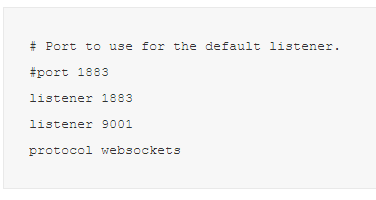


Al terminar deberán crear y modificar el archivo de configuración por defecto **mosquitto.conf**

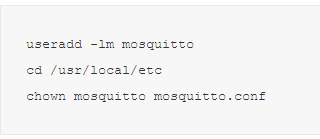
**Desde ~/IoTServer/mosquitto-1.4.8/**



Y cambien la configuración por defecto a:



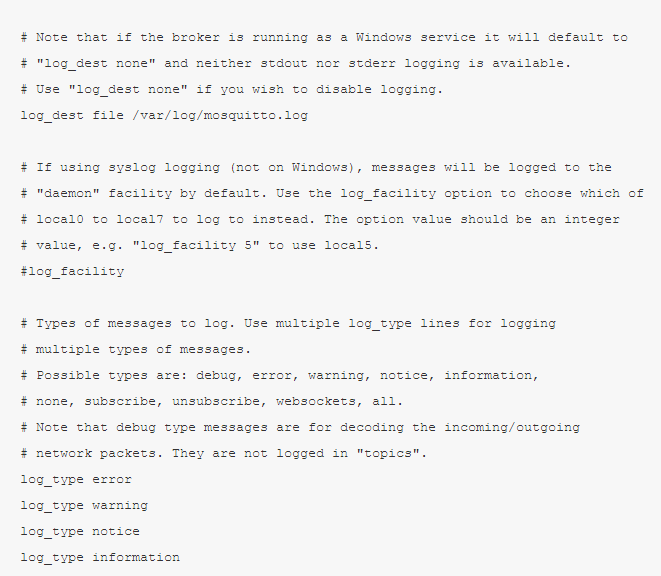
NOTA Importante: después de la versión 1.4.5 el usuario por defecto para la ejecución del servicio es **mosquitto. Asi que deberá asignarle permisos para que ejecute el servicio:**

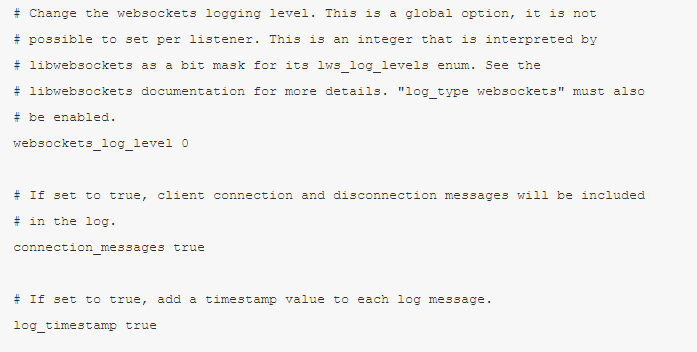
****

Adicionalmente se recomienda agregar las siguientes líneas de configuración a modo de obtener la mayor visibilidad sobre el servicio:

En el archivo cd /usr/local/etc/mosquitto.conf

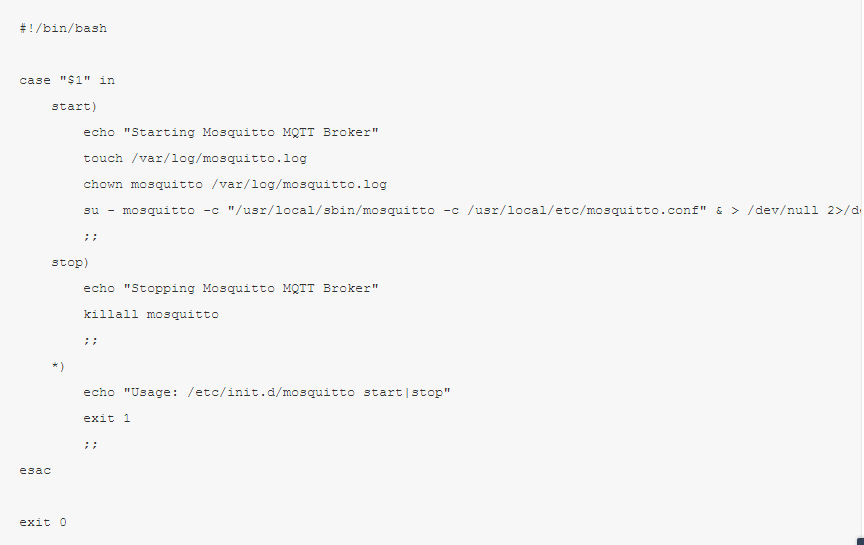
Ver imagen abajo.





**Configurando el inicio automático del servicio.**

La forma más rápida es agregando un archivo al directorio init.d, entonces creamos un archivo llamado mosquitto en /etc/init.d



su - mosquitto -c "/usr/local/sbin/mosquitto -c /usr/local/etc/mosquitto.conf" & > /dev/null 2>/dev/null

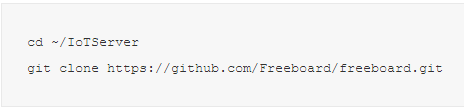
Encuentra el nivel de permisos con # runlevel

Crea el link automático de inicio con:

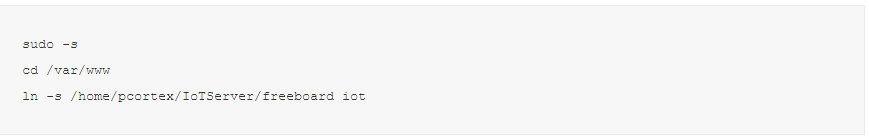


**Paso 3: instala freeboard.io dashboard**

Ahora solo necesitamos un medio de ver la data en la web… ya casi terminamos..



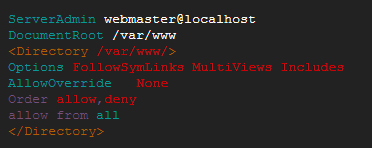
Se crea el symlink de apache apuntado hacia donde se instala el freeboard.



Antes de encontrar el link en <http://myserveraddres/iot>.

Debes otrorgar permisos editando el archivo en /etc/apache2/sites-enabled/000-default.conf

Y cambiar la configuración por:



Y reinciamos el apache!

sudo service apache2 restart

Primero bajamos la versión de desarrollo de Mqtt.js

<http://git.eclipse.org/c/paho/org.eclipse.paho.mqtt.javascript.git/plain/src/mqttws31.js?h=develop>)

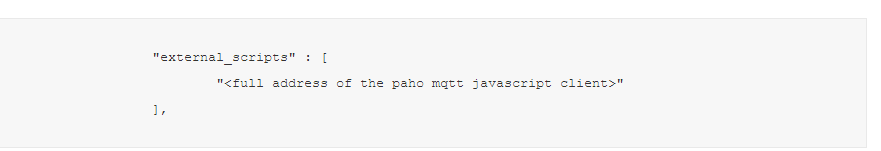


Y luego bajamos el plug in par freeboard de MQTT:

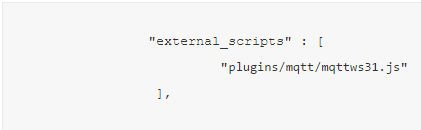


Ahora solo editamos el archivo paho.mqtt.plugin.js y cambiamos la línea:

De:



A:



Por ultimo modificamos el archive  **index.html en el directorio donde instalamos freeboard de:**



A:



Por ultimo comprobamos que mosquito este corriendo los WEBSOCKETS con el commando

**nohup mosquitto -c /etc/mosquitto/mosquitto.conf &**

**y verificamos que el Protocolo este corriendo:**



Con eso habremos logrado ya configurar nuestro servidor : así que solo queda programar nuestro Arduino y probar el envió de datos.

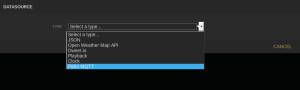
**Paso 4. Cargar el código al arduino.**

Bajar el Zip con el Codigo para arduino de: <https://github.com/EdwinKestler/IoTarduinoDayGT/>

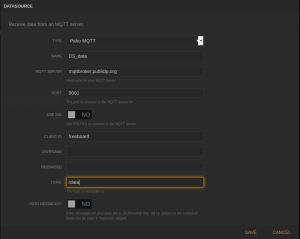
Compilarlo y cargarlo al ESP desde el IDE de Arduino.

Paso 5. Configurar nuestros panles de Freeboard.

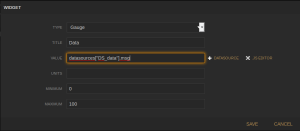
Seleccione la Fuente de datos (PahoMQTT)

[](https://primalcortex.files.wordpress.com/2015/02/selection_055.png)

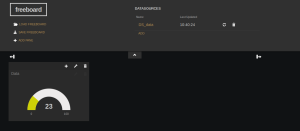
Configurar el servidor y la fuente de dato:

[](https://primalcortex.files.wordpress.com/2015/02/selection_056.png)

Crear los paneles.

[](https://primalcortex.files.wordpress.com/2015/02/selection_057.png)

Y VEr La Data Fluir…..

[](https://primalcortex.files.wordpress.com/2015/02/selection_058.png)

**Felicidades han creado su primer openframework de IoT**

**IOT Arduino Day Guatemala 2016!**