

**ESCENARIO IOT CON SENSOR DE TEMPERATURA Y HUMEDAD DHT11
EN RASPBERRY PI**



Universidad
del Cauca

PLATAFORMAS UBICUAS

PRESENTADO A:

PhD. GUSTAVO ADOLFO RAMIREZ

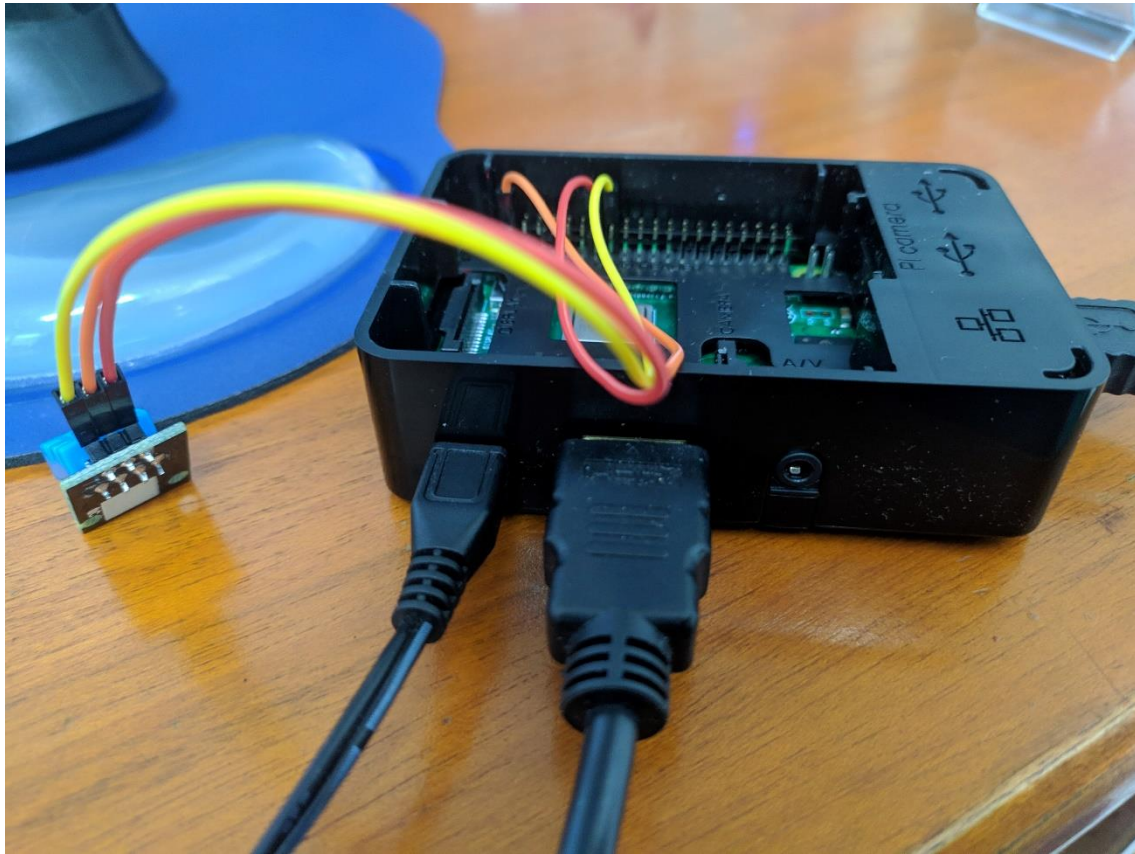
PRESENTADO POR:

ING. JUAN FELIPE VALENCIA MOSQUERA

UNIVERSIDAD DEL CAUCA

MAESTRÍA EN INGENIERÍA TELEMÁTICA

En el presente documento encontraremos la guía de pasos con el cual se llevo a cabo la instalación y creación del escenario que consta de un sensor de temperatura con referencia DHT11 y una raspberry pi.

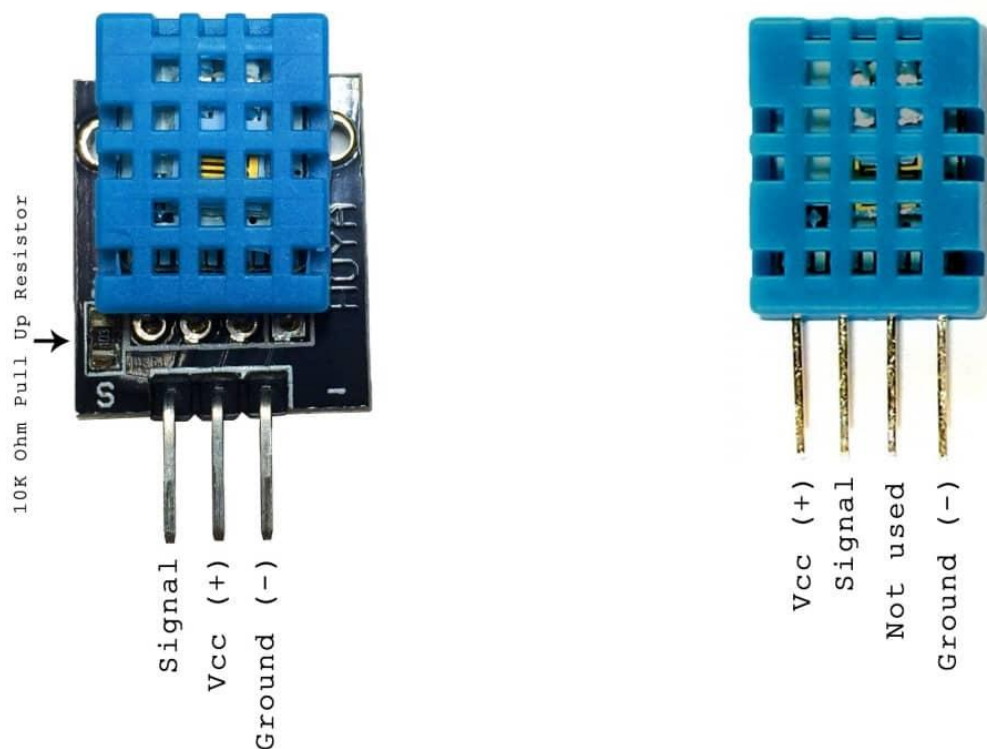


MATERIALES:

- Raspberry pi
- Sensor de temperatura DHT11
- Jumpers de conexión (Cableado)

CONSIDERACIONES PREVIAS RESPECTO AL SENSOR

Actualmente existen dos modelos de sensor DHT11. Podríamos decir que uno es más moderno que otro. En función del que tengamos las conexiones en los pines son diferentes. En la siguiente imagen vemos la comparativa de ambos sensores.



El sensor de la izquierda es el nuevo sensor DHT11 en el cual lleva integrada en su placa una resistencia de 10K necesaria para su funcionamiento. Mientras que el sensor del lado derecho es un poco mas antiguo necesita que se añada una resistencia externa en la protoboard con el objetivo de cuidar el sensor y no sea susceptible a daños por variaciones en la energía. Para nuestro escenario contamos con el sensor de la izquierda el cual nos evita el uso de protoboard.

CONFIGURACIÓN DE LA RASPBERRY PI

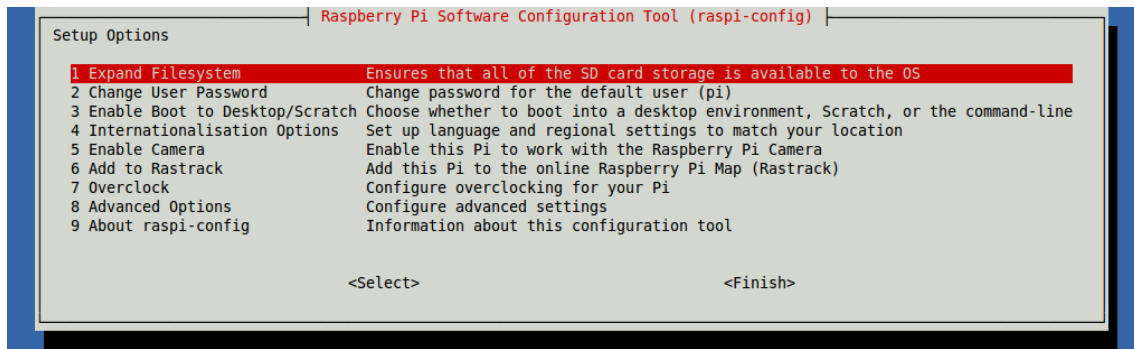
Inicialmente se debe instalar el sistema operativo de la raspberry utilizando una microSD. Para ello se descarga desde la pagina <https://www.raspberrypi.org/downloads/>

Una vez descargado el sistema operativo se añade el archivo descargado como imagen ISO a la tarjeta SD, posterior a ello se coloca la SD a la raspberry para poder que cuando la raspberry se encienda se ejecute el sistema operativo.

CONFIGURACIÓN DE SSH PARA LA RASPBERRY

Para poder realizar conexiones remotas con la raspberry Pi desde nuestro PC. Escribiremos en un terminal el comando

\$ sudo raspi-config



Posterior a ejecutar el comando desde la terminal se muestra la imagen anterior otorgando las opciones, para encender el SSH se realizan los siguientes pasos:

1. Seleccionamos la opción **8 Advanced Options**
2. Elegimos la opción de menú **SSH**
3. Seleccionamos **Yes**
4. Presionamos **Ok**
5. Elegimos la opción **Finish**

Una vez realizados estos pasos, la raspberry tiene habilitado la opción de generar conexiones de escritorio remotas (SSH).

PREPARACIÓN DEL SOFTWARE DHT11

Para iniciar la configuración del escenario principalmente actualizaremos la lista de paquetes en nuestra Raspberry Pi con el siguiente comando:

\$ sudo apt-get update && sudo apt-get upgrade

Posterior a esto, cabe resaltar que la tecnología que utilizaremos será NodeJS para la conexión del escenario. Por lo tanto, descargamos la última versión de nodejs directamente desde la web oficial bajo el siguiente comando desde el terminal:

\$ curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -

Una vez descargado el paquete realizamos la instalación en la Raspberry Pi con el comando:

\$ sudo apt-get install nodejs

Adicional a lo anterior, se debe instalar ciertas librerías necesarias para el sensor DHT11, de igual manera necesitamos instalar el gestor de paquetes npm bajo el comando

\$ sudo npm install

ESQUEMA DE CONEXIÓN DEL SENSOR A LA RASPBERRY PI

Para la conexión física del sensor DHT11 usaremos los pines del GPIO de la raspberry 1 (3v3),14 (Ground),16 (GPIO23) situados en la posición que se indica en la siguiente figura:

Mientras tanto el sensor DHT11 tiene 3 pines que son signal, Vcc (+) y Ground (-):

El esquema de conexión entre el sensor DHT11 y la raspberry PI quedaría entonces de la siguiente manera:

- Pin 3 del DHT11 (Ground (-)) conectado al Pin 14 (Ground) del GPIO de la Raspberry Pi

INSTALACION DE LOS COMPONENTES SOFTWARE PARA EL SENSOR DHT11

El sensor DHT11 necesita dos componentes software principales para funcionar.

1. Librería bcm2835
2. Paquete node-dht-sensor

Para la instalación del primero ejecutamos los siguientes comandos:

```
$ wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.46.tar.gz
```

```
$ tar zxvf bcm2835-1.46.tar.gz
```

```
$ cd bcm2835-1.46
```

```
$ ./configure
```

```
$ make
```

```
$ sudo make check
```

```
$ sudo make install
```

```
$ cd ..
```

Una vez instalado la librería bcm2835 se instala el paquete npm:

```
$ sudo npm install -g --unsafe-perm node-dht-sensor
```

Ahora se procede a conectar físicamente el sensor DHT11 a los pines del GPIO de la raspberry pi, posterior a esto creamos un archivo "sensor-text" con extensión .js:

```
//Funcion que permite la toma de datos del sensor DHT11
```

```
setInterval(function() {
```

```
    //Requerimos las librerias para funcionamiento del sensor
```

```
    var sensor = require('/usr/lib/node_modules/node-dht-sensor');
```

```
    //Utilizamos la funcion read de la librería del sensor pasandole como parametro 11 por el tipo de sensor "DHT11", y 23
```

```
    //que es el valor del GPIO023 el pin 16 de la raspberry
```

```
    sensor.read(11, 23, function(err, temperature, humidity) {
```

```
        if (!err) {
```

//Si no hay errores en lectura de libreria y ejecución de la función del sensor entonces imprime la temperatura "temp"

//ambiente y el valor de la humedad "humidity"

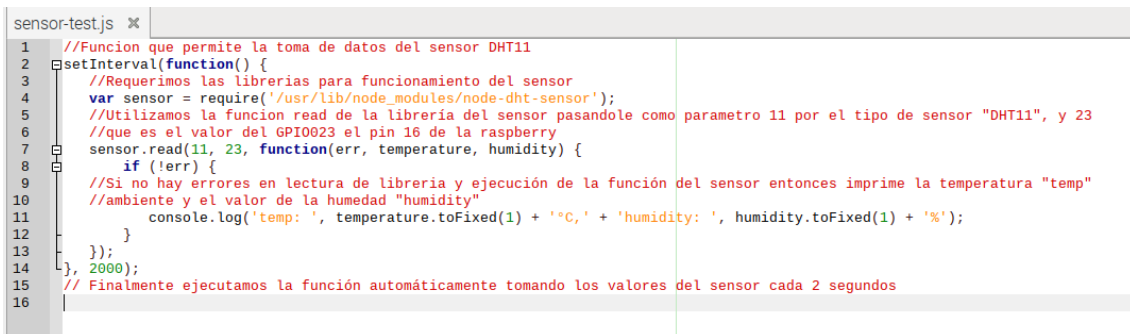
console.log('temp: ', temperature.toFixed(1) + '°C,' + 'humidity: ', humidity.toFixed(1) + '%');

}

});

}, 2000);

// Finalmente ejecutamos la función automáticamente tomando los valores del sensor cada 2 segundos.

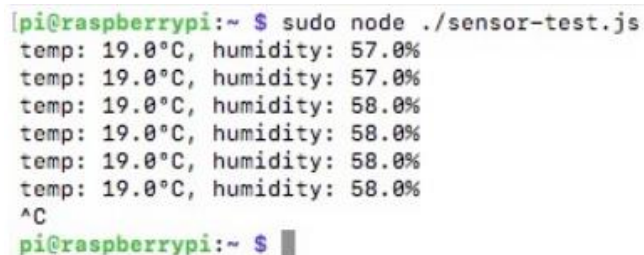


```
1 //Funcion que permite la toma de datos del sensor DHT11
2 setInterval(function() {
3   //Requerimos las librerias para funcionamiento del sensor
4   var sensor = require('/usr/lib/node_modules/node-dht-sensor');
5   //Utilizamos la funcion read de la libreria del sensor pasandole como parametro 11 por el tipo de sensor "DHT11", y 23
6   //que es el valor del GPIO023 el pin 16 de la raspberry
7   sensor.read(11, 23, function(err, temperature, humidity) {
8     if (!err) {
9       //Si no hay errores en lectura de libreria y ejecución de la función del sensor entonces imprime la temperatura "temp"
10      //ambiente y el valor de la humedad "humidity"
11      console.log('temp: ', temperature.toFixed(1) + '°C,' + 'humidity: ', humidity.toFixed(1) + '%');
12    }
13  });
14 }, 2000);
15 // Finalmente ejecutamos la función automáticamente tomando los valores del sensor cada 2 segundos
16
```

Lo importante del código es la función Read y los parámetros que se le pasan, como se observa se le ha pasado 11 por el tipo de sensor que estamos utilizando y 23 que es el valor del GPIO con pin 16 en la raspberry.

Finalmente, desde la terminal ejecutamos el comando:

\$ sudo node ./sensor-test.js



```
pi@raspberrypi:~ $ sudo node ./sensor-test.js
temp: 19.0°C, humidity: 57.0%
temp: 19.0°C, humidity: 57.0%
temp: 19.0°C, humidity: 58.0%
temp: 19.0°C, humidity: 58.0%
temp: 19.0°C, humidity: 58.0%
temp: 19.0°C, humidity: 58.0%
^C
pi@raspberrypi:~ $
```