

Hands on Lab: Connecting an ESP8266 (with temperature sensor) to Azure IoT Hub, Stream Analytics and PowerBI

Overview:

This hands on lab demonstrates how to connect, code and configure an ESP8266 device with a temperature sensor to an Azure IoT hub. Once connected to the hub, the data will be streamed to PowerBI using Azure Stream Analytics. Once in Power BI, we will analyze and present the data for user consumption.

For this demonstration, we will use an inexpensive DHT11 sensor, which has a variance of $\pm 2^{\circ}\text{C}$ accuracy.

Time for lab: 60 minutes

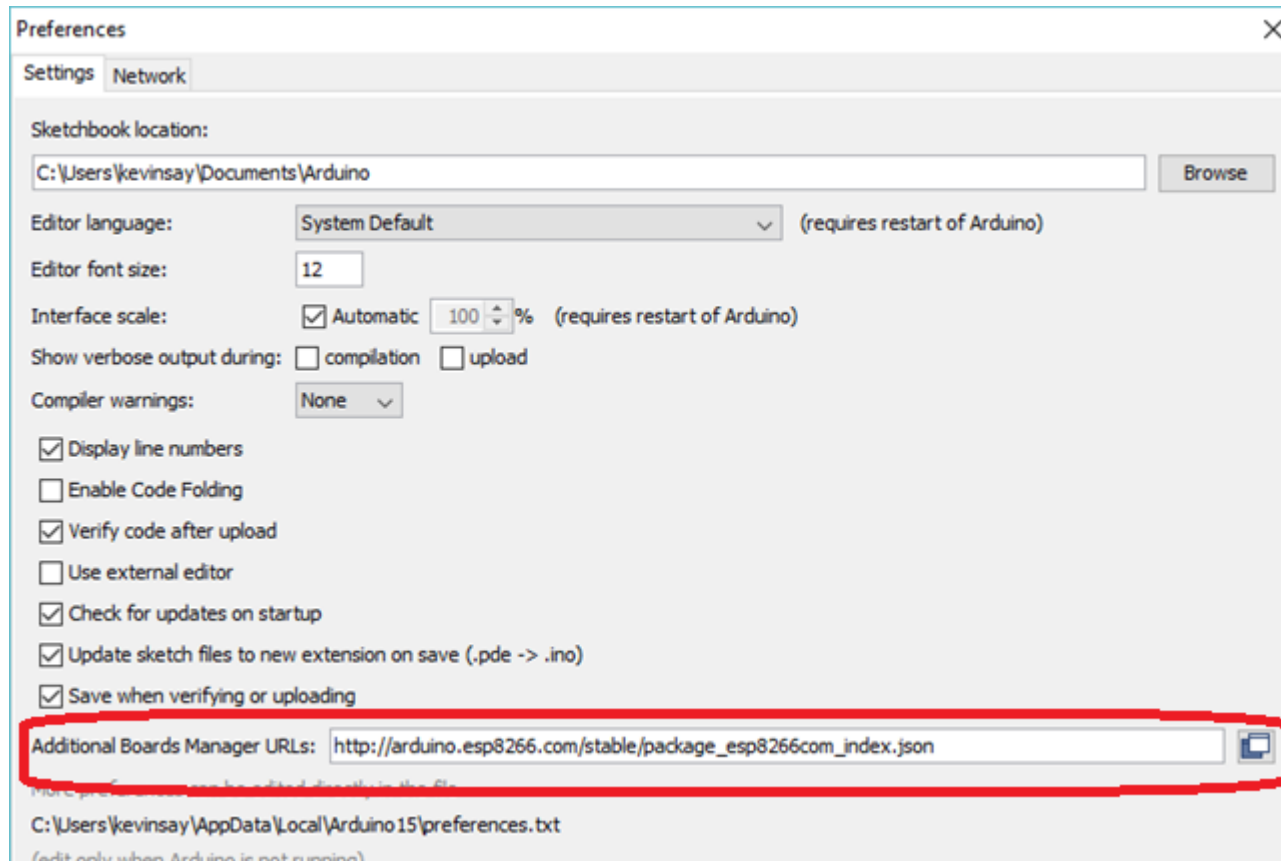
Required Materials:

This lab requires each student have:

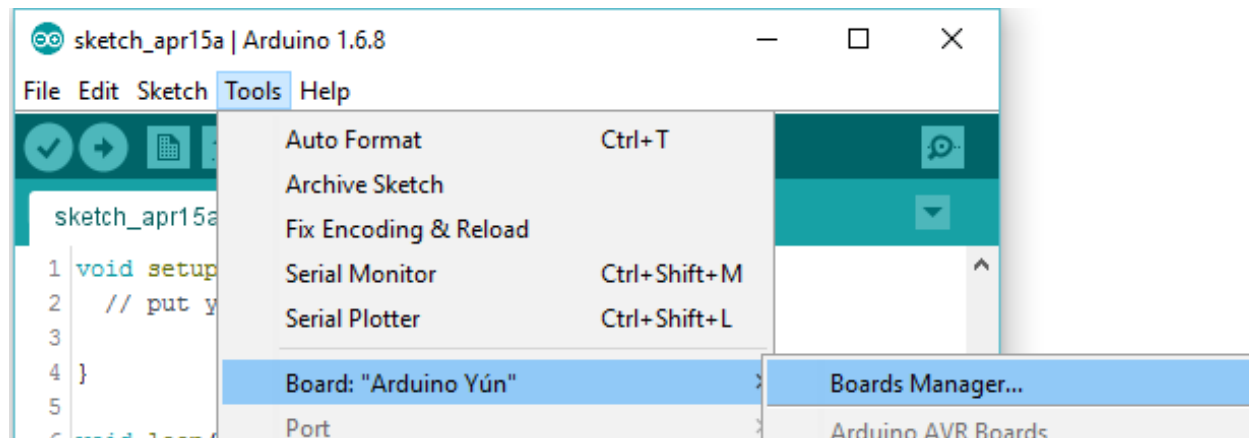
1. NodeMCU ESP8266 hardware (http://nodemcu.com/index_en.html), ~\$5.00 on eBay
2. DHT11 temperature sensor (<https://www.adafruit.com/products/386>), ~\$1.00 on eBay
3. 3 female to female jumper cables
4. USB cable with Micro connection
5. Windows PC (MAC can be used)
6. Arduino IDE (<https://www.arduino.cc/en/Main/Software>)
7. Azure IoT Device Explorer: (<https://github.com/Azure/azure-iot-sdks/releases>)
8. Azure Subscription: (<http://portal.azure.com>)

Get Started:

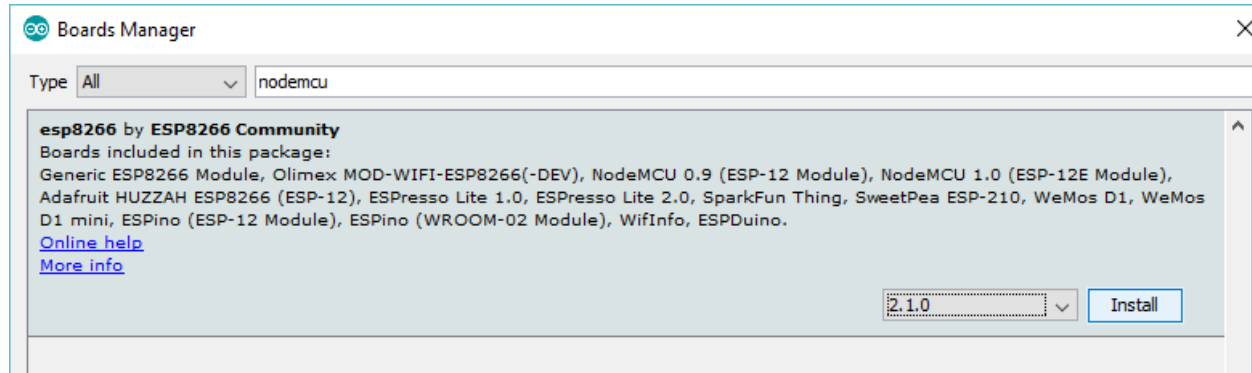
- Step 1. On your PC, install Arduino IDE from: <https://www.arduino.cc/en/Main/Software>.
- Step 2. Start Arduino, and under **FILE -> PREFERENCES** add http://arduino.esp8266.com/stable/package_esp8266com_index.json to the Board Manager URL



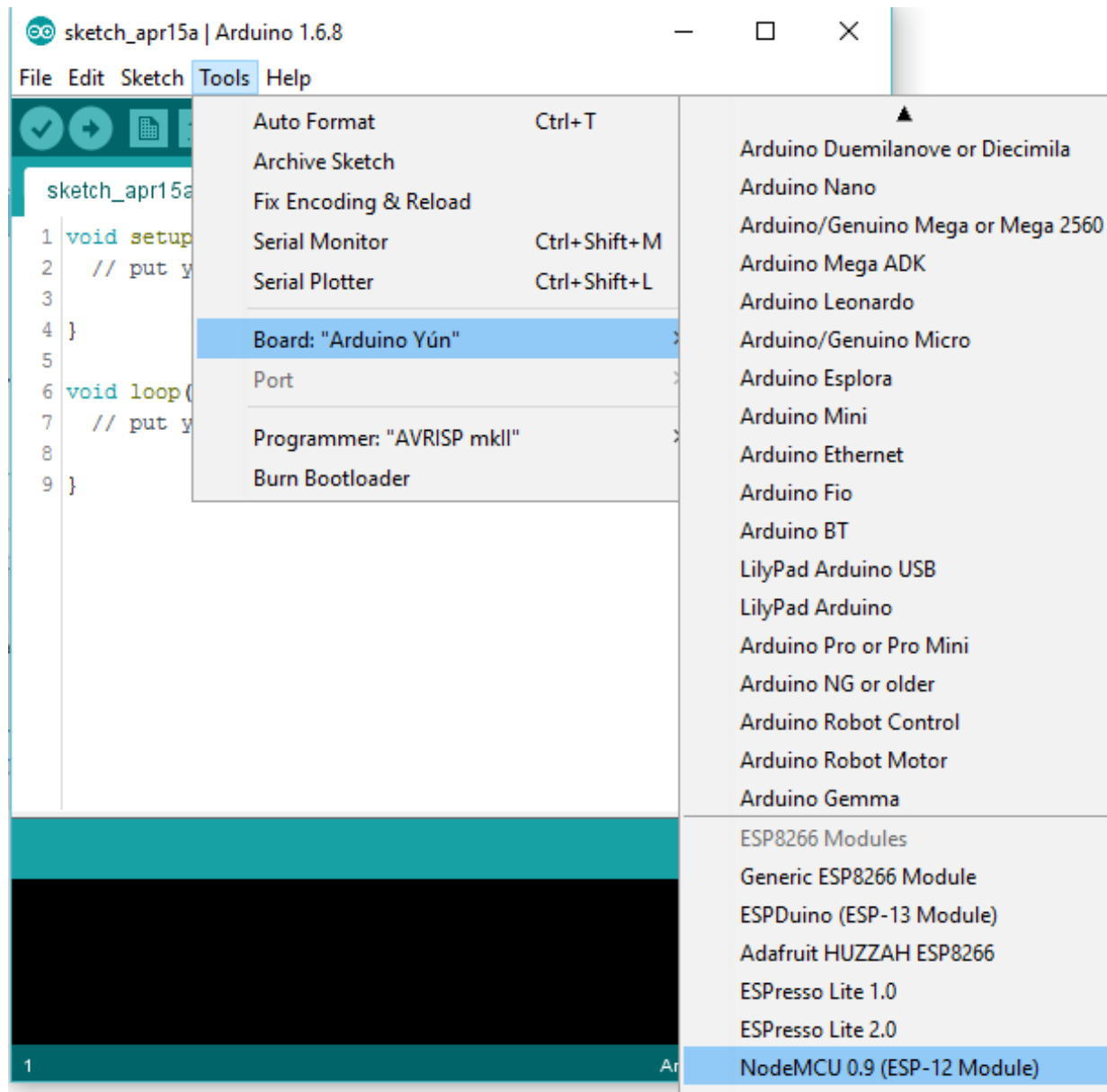
Step 3. In Arduino, **TOOLS -> BOARD:** select **Board Manager...**



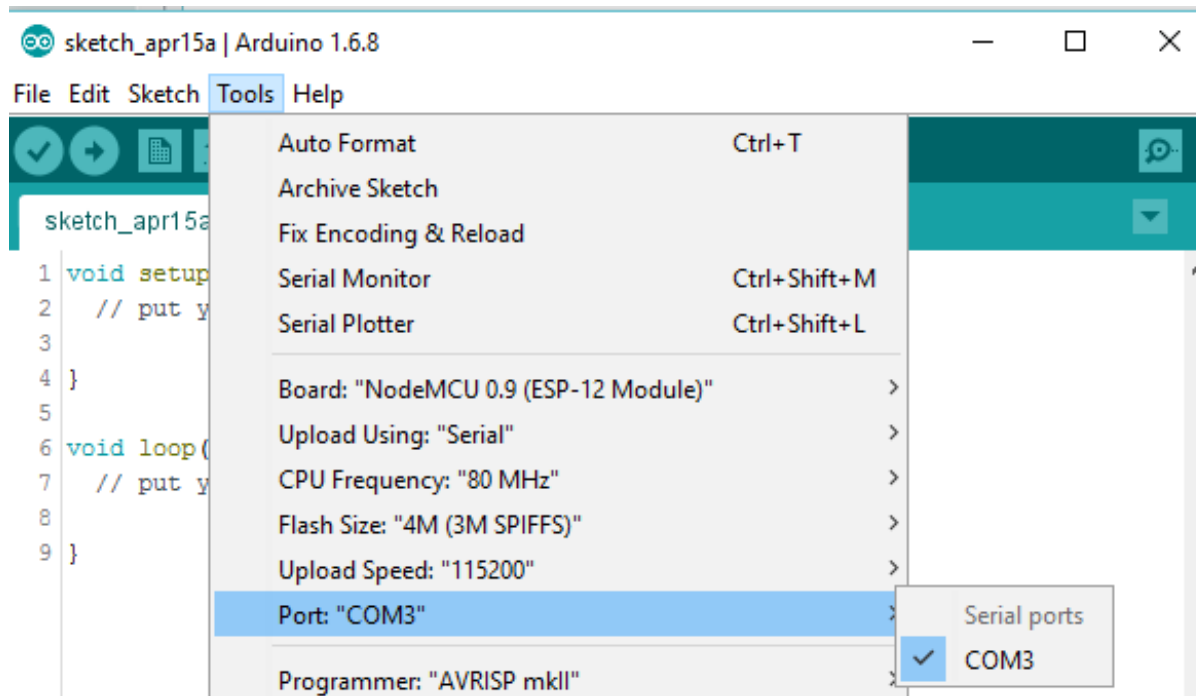
Step 4. Search for “*NodeMCU*” and click **Install**. If you can’t find the board, check your Additional Board Manager URL.



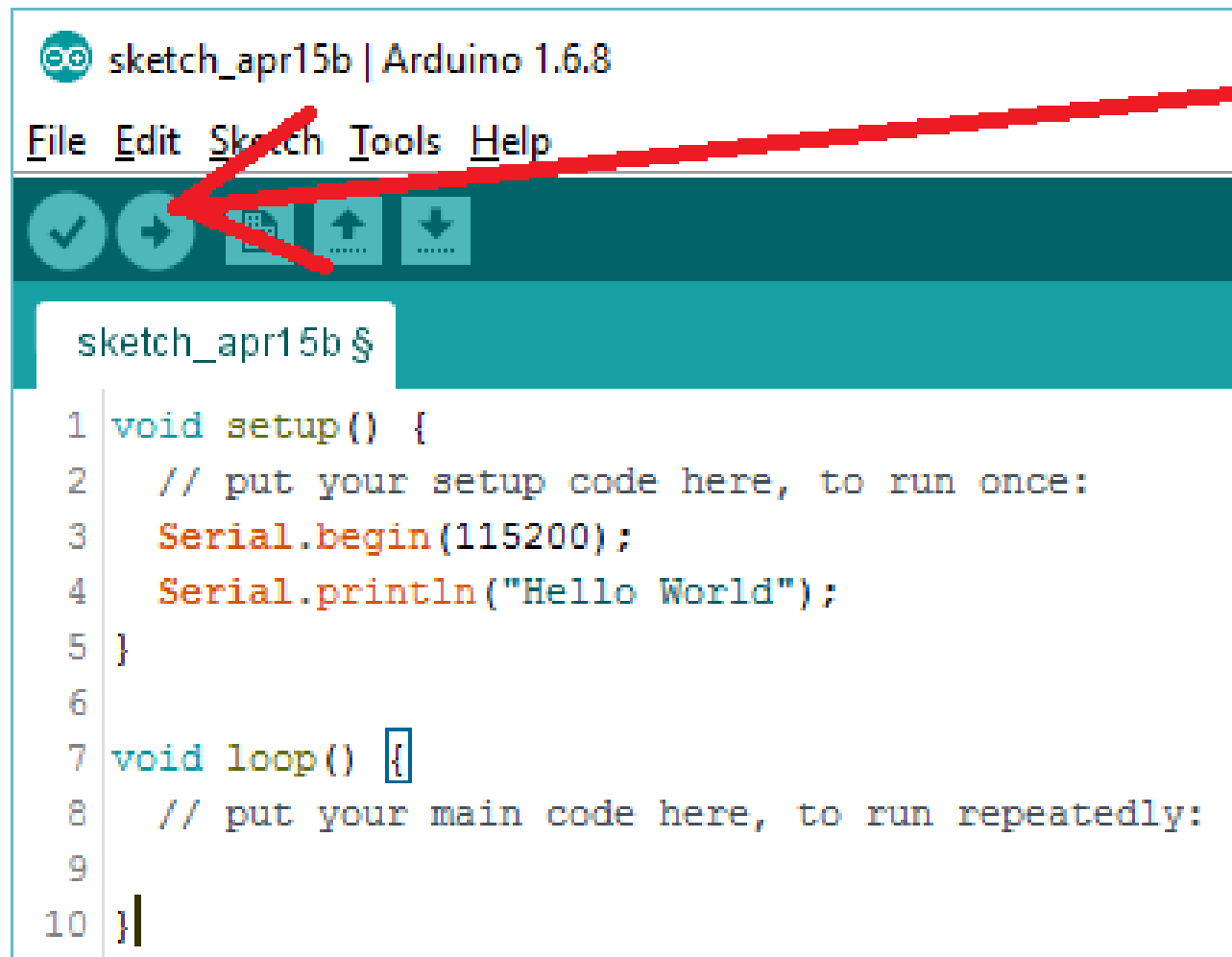
Step 5. Close the board manager window. Select **TOOLS -> BOARD:** and select the **NodeMCU 0.9** board.



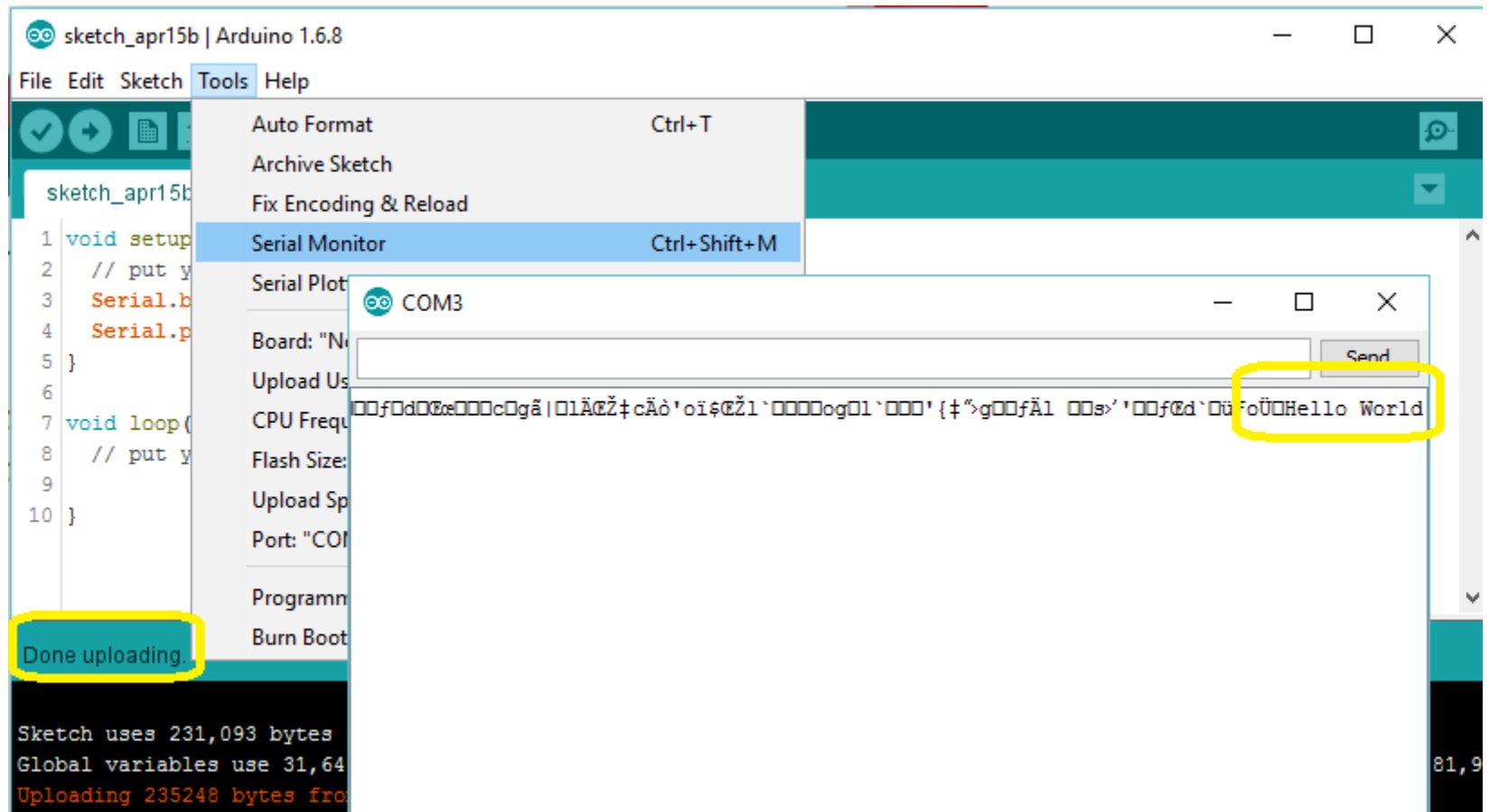
Step 6. Plug your ESP8266 into the USB cable. Your PC should detect this device and add a COM port that Arduino will detect. To verify you see the device, click **TOOLS** -> **PORT**: to see the new com port.



Step 7. Create the mandatory "Hello World!" demo. In the **void setup()** section, type the following 2 Serial commands shown below (line #3 and #4). Then click the "Upload button".



Step 8. When uploading is complete, click **TOOLS -> SERIAL MONITOR** and verify that Hello World was sent to the serial connection.



Step 9. Connect the ESP8266 to the **WIFI network**. Add line #1 and #7 - #10 below. Upload the code and monitor it via Serial Monitor.



sketch_apr15b \$

```

1 #include <ESP8266WiFi.h>
2
3 void setup() {
4     // put your setup code here, to run once:
5     Serial.begin(115200);
6     Serial.println("Hello World");
7     WiFi.begin("saye.org", "{password}");
8     WiFi.waitForConnectResult();
9     Serial.print("My IP is: ");
10    Serial.print(WiFi.localIP());
11 }
12
13 void loop() {

```

Done uploading.

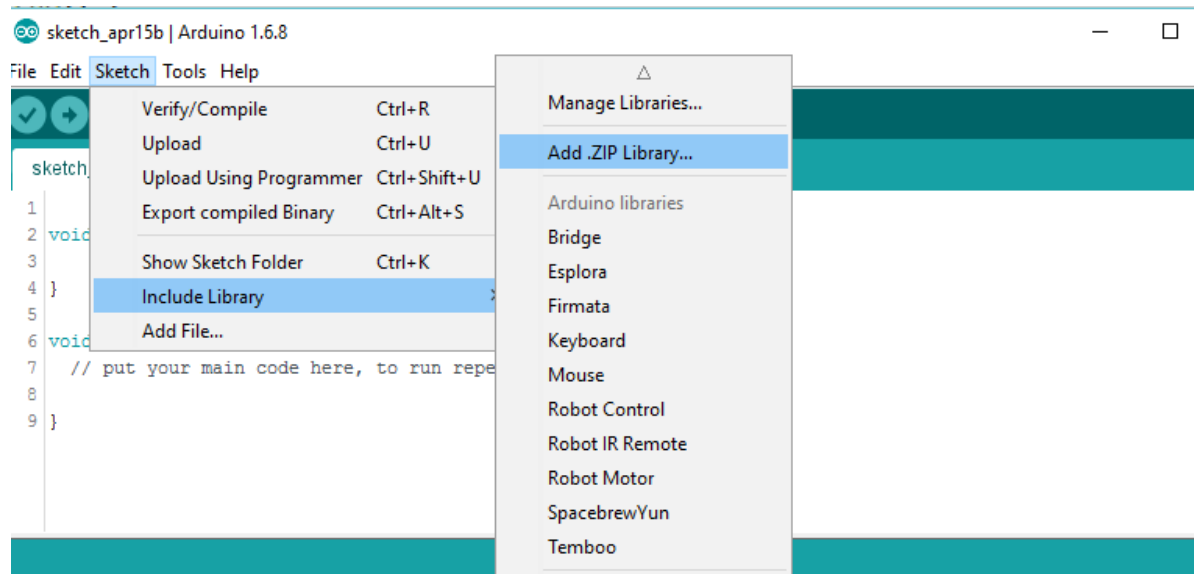
Sketch uses 232,899 bytes (22%) of program storage space. Maximum is 1024K bytes.
Global variables use 31,912 bytes (38%) of dynamic memory, leaving 50,088 bytes for the global variable stack.
Uploading 237056 bytes from USB to flash at 0x00000000
.....

COM3

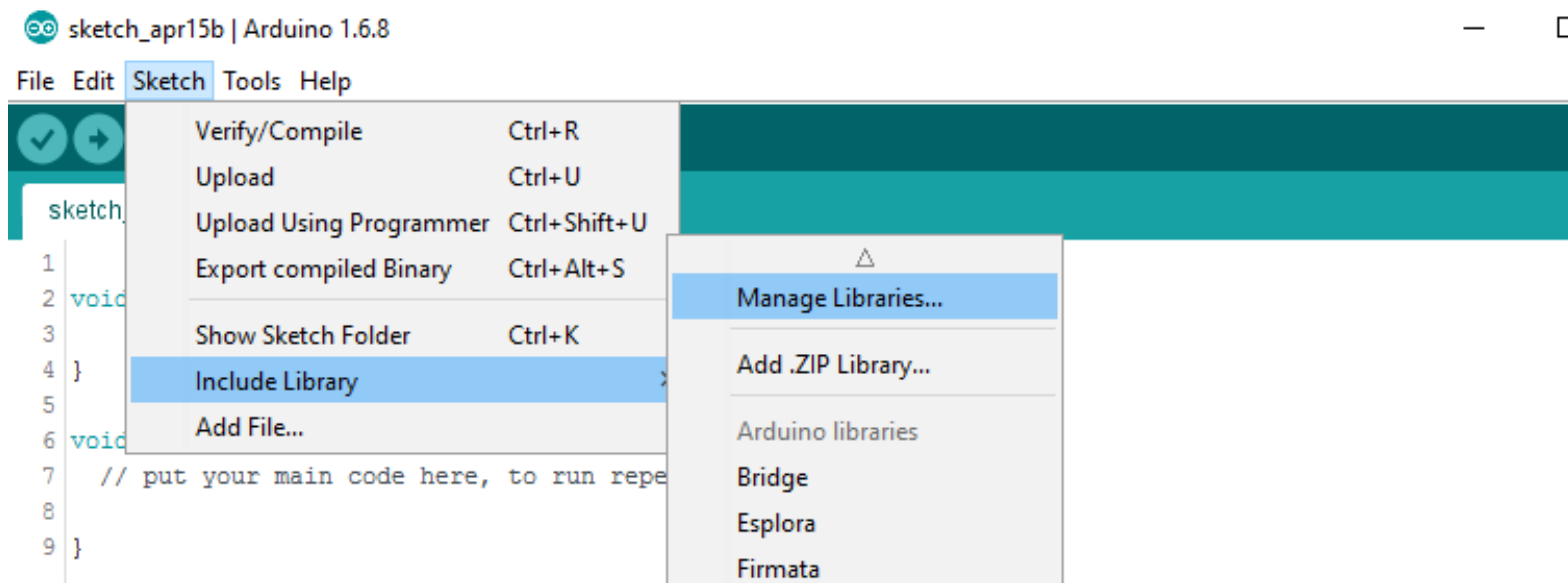
My IP is: 192.168.15.134

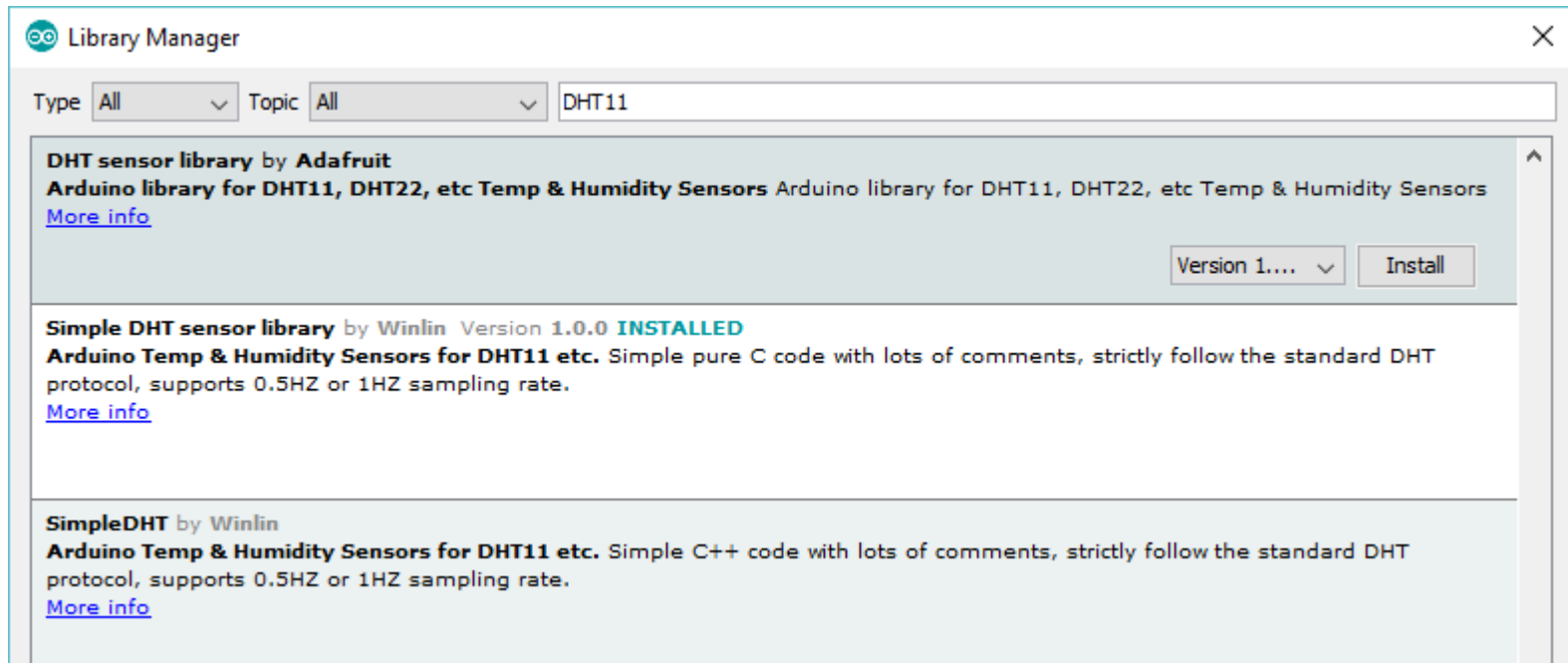
Step 10. Download the MQTT library that supports TLS from: <https://github.com/Imroy/pubsubclient>. Click the “Download ZIP” button and save it to your desktop.

Step 11. In Arduino click **SKETCH -> INCLUDE LIBRARY -> ADD.ZIP LIBRARY...** Locate the “pubsubclient-master.zip” you just downloaded and click open.

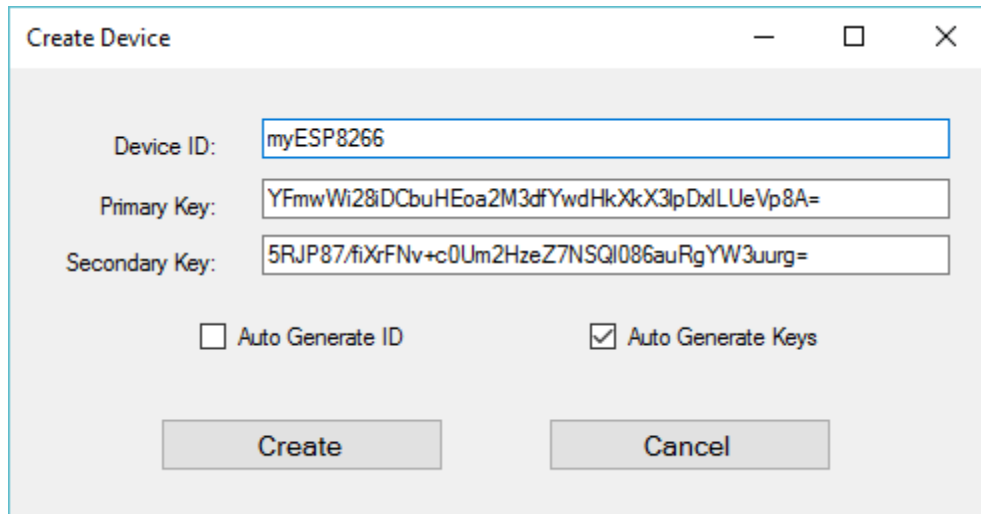


Step 12. In Arduino click **SKETCH -> INCLUDE LIBRARY -> MANAGE LIBRARIES...** Search for DHT11 and install the “DHT sensor library by Adafruit”.





- Step 13. Install Device Explorer from <https://github.com/Azure/azure-iot-sdks/releases>.
- Step 14. In Azure, create an **Azure IoT Hub** using the Free SKU.
- Step 15. In the IoT Hub, in **SHARED ACCESS POLICIES -> IOTHUBBROWSER**, copy the **CONNECTION STRING – PRIMARY KEY**
- Step 16. In Device Explorer, past the Connection String in **THE IOT CONNECTION HUB CONNECTION STRING** setting and click update.
- Step 17. In Device Explorer, on the Management Tab, click create to create a device. Write down the **name of the device**:



Create Device

Device ID: myESP8266

Primary Key: YFmwWi28DCbuHEoa2M3dfYwdHkXkX3lpDxlLUeVp8A=

Secondary Key: 5RJP87/fiXrFNv+c0Um2HzeZ7NSQl086auRgYW3uurg=

☐ Auto Generate ID ☒ Auto Generate Keys

Create Cancel

Step 18. In Device Explorer, on the Management Tab, click SAS Token, select the device just created and copy the **SharedAccessSignature**, save this for later. Also note the **hostname**, which ends with .azure-devices.net.

SASTokenForm

DeviceID myESP8266

DeviceKeys PtTAloTcORnyl+kyLRdnpZE8mTENkdq31LYx9vAevGg=

TTL (Days) 365

HostName=kevinsay.azure-devices.net;DeviceId=myESP8266;SharedAccessSignature=
sr=kevinsay.azure-devices.net%2fdevices%2fmyESP8266
&sig=oUgKxUxG6J7aprK49A436Aem%2fATwAFj%2fYBZSGRt8%3d&se=
1492291543

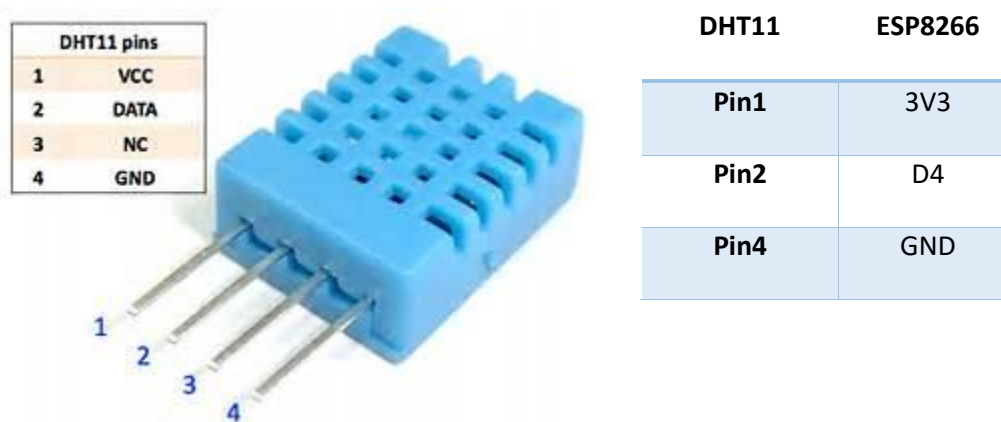
Generate Done

- Step 19. Download the file “NodeMCU_ESP8266_AzureIoT.ino” from <https://github.com/ksaye/NodeMCU-to-Azure-IoT>.
- Step 20. Open the file “NodeMCU_ESP8266_AzureIoT.ino” in Arduino and modify the following sections with the data captured above:

```
NodeMCU_ESP8266_AzureIoT | Arduino 1.6.8
File Edit Sketch Tools Help

NodeMCU_ESP8266_AzureIoT $
4
5 /* Begin of Constants */
6 /* WIFI Settings */
7 const char *ssid = "My SSID";
8 const char *password = "My Password";
9
10 /* Azure IOT Hub Settings */
11 const char* mqtt_server = "kevinsay.azure-devices.net";
12 const char* devicename = "one";
13 // Example "SharedAccessSignature sr=kevinsay.azure-devices.net%2fdevices%2fone&sig=UAWH
14 const char* devicesas = "{ Generate this from Device Explorer }";
15 long interval = 60000; //(ms) - 60 seconds between reports
16 /* End of Constants */
```

Step 21. Wire the DHT11 to the ESP8266 using the jumper cables following the schema below:



Step 22. In the Arduino IDE, upload the code to the ESP8266 and watch both in the Serial Monitor and in Device Explorer on the Data tab.

Step 23. In Azure, create a new Stream Analytics job.

Step 24. On **Inputs**, add a data stream of type **IoT Hub**, provide an Alias, select the **IoT Hub** you created, select the access policy “iothubowner” and set the serialization to JSON.

ADD A DATA STREAM

Add a data stream to your job

- ☐ Event Hub [?]
- ☐ Blob storage [?]
- ☒ IoT Hub ^{PREVIEW} [?]

Missing an input source? [Let us know!](#) (Powered by [UserVoice](#) - [Privacy Policy](#))

1



3

1

2

ADD AN IOT HUB

IoT Hub settings

INPUT ALIAS [?]

MyIoTHub

SUBSCRIPTION

Use IoT Hub from Current Subscription

CHOOSE AN IOT HUB [?]

kevinsay (eastus)

IOT HUB SHARED ACCESS POLICY NAME [?]

iothubowner

IOT HUB CONSUMER GROUP [?]

\$Default



4

123

ADD A BLOB STORAGE

Serialization settings

EVENT SERIALIZATION FORMAT ?
JSON

ENCODING ?
UTF8

←

✓

Step 25. On Outputs, add **PowerBI** as your target and **authorize** the connection.

ADD AN OUTPUT

Add an output to your job

☐ SQL Database ?

☐ Blob storage ?

☐ Event Hub ?

☒ Power BI ?

☐ Table storage ?

☐ Service Bus Queue ?

☐ Service Bus Topic ?

☐ DocumentDB ?

☐ Data Lake Store **PREVIEW** ?

Missing an output sink? [Let us know!](#) (Powered by [UserVoice](#) - [Privacy Policy](#))

→

2

1

ADD A MICROSOFT POWER BI OUTPUT

Authorize Connection

Existing Microsoft Power BI User

Authorize Stream Analytics to access your organizational Microsoft Power BI subscription to create a live dashboard.

[Authorize Now](#) →

Note: You are granting this output permanent access to your Power BI dashboard. Should you need to revoke this access in the future you can do one of the following:

1. Change the user account password.
2. Delete this output.
3. Delete this job.

New User

Don't have a Microsoft Power BI account yet? [Sign up now.](#)

←

→

3

Step 26. On Query, add the following query and click **Save**, then click **Start**

esp8266toazureiot

[DASHBOARD](#)[MONITOR](#)[INPUTS](#)[FUNCTIONS](#) **PREVIEW**[QUERY](#)[OUTPUTS](#)[SCALE](#)[CONFIGURE](#)

Need help with your query? Check out some of the most common Stream Analytics query patterns [here](#).

query

```
1 SELECT
2   *
3 INTO
4   MyPowerBI
5 FROM
6   MyIoTHub
```

Parts of your query could be used in error messages that are written to logs in a potentially different geography.

Missing some language constructs? [Let us know!](#) (Powered by [UserVoice](#) - [Privacy Policy](#))

[Test](#)[Rerun](#)

Step 27. In Azure, click Dashboard to monitor Stream Analytics move data from the Hub to PowerBI.

☒ DATA CONVERSION ERRORS ☒ RUNTIME ERRORS ☒ INPUT EVENT BYTES ☒ INPUT EVENTS [7 MORE](#) RELATIVE 1 HOUR ↺ ↻

