IoT With ESP8266

Mr.Prarinya Ekapho

Agenda for 2 days training

Day1

- IoT Concept and Applications
- Workshop#1 (Start with NodeMCU ESP8266)
- Workshop#2 (Use platform Thinkspeak)

Day2

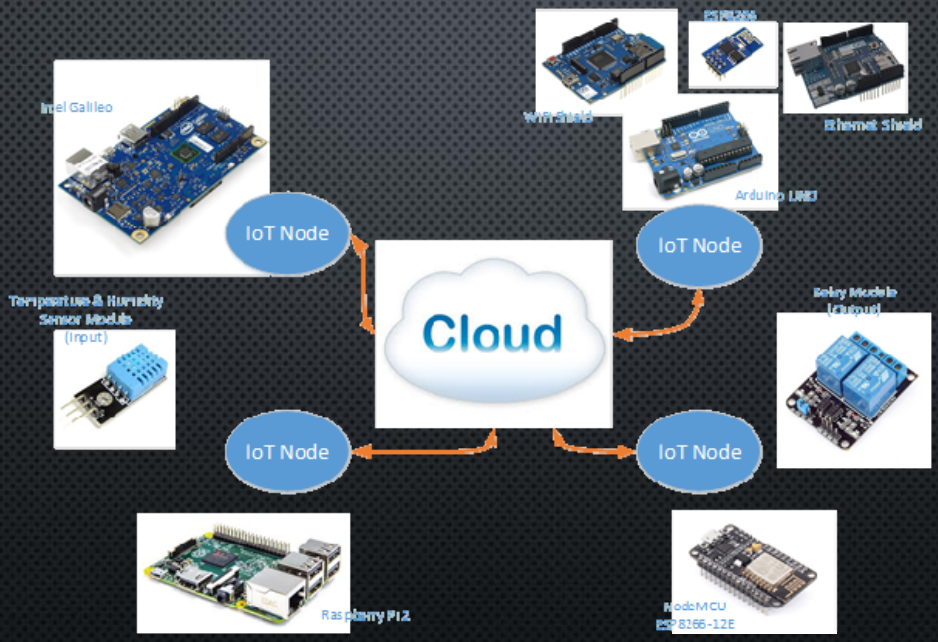
- Workshop#3 (Use platform NETPIE)
- Workshop#4 (Use others application to control IoT)
- Conclude and guide to development

IoT Concept

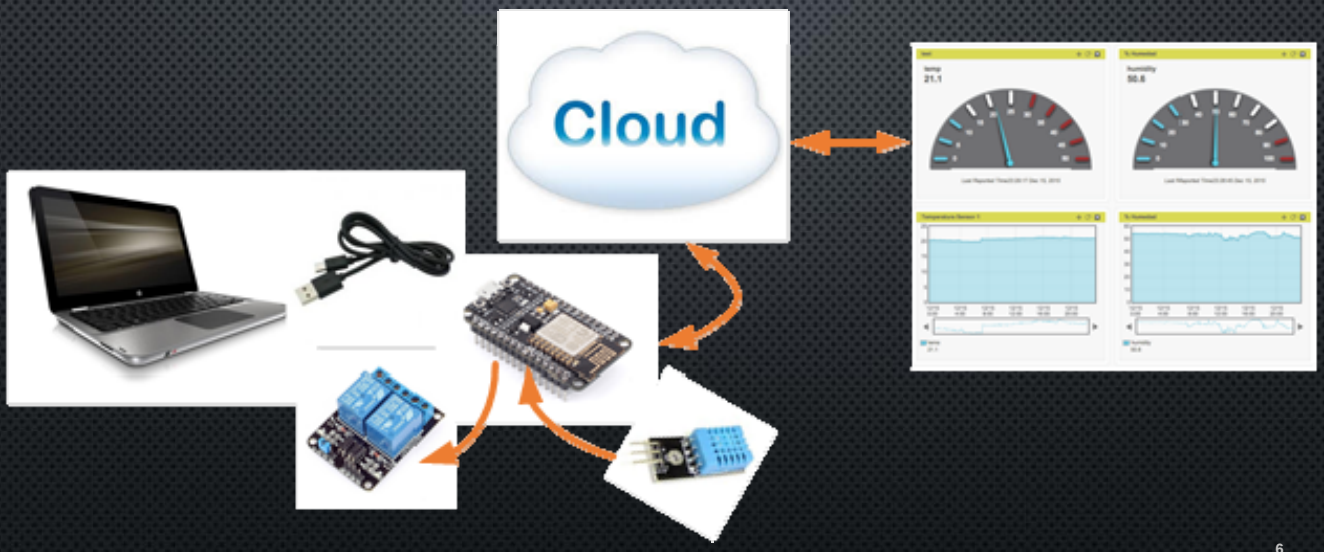
Internet of Things (IoT) Components

- Things
- Sensors & Controls I/O
- Internet connect devices
- Data
- Cloud Service

IoT Development guide



System Design concept



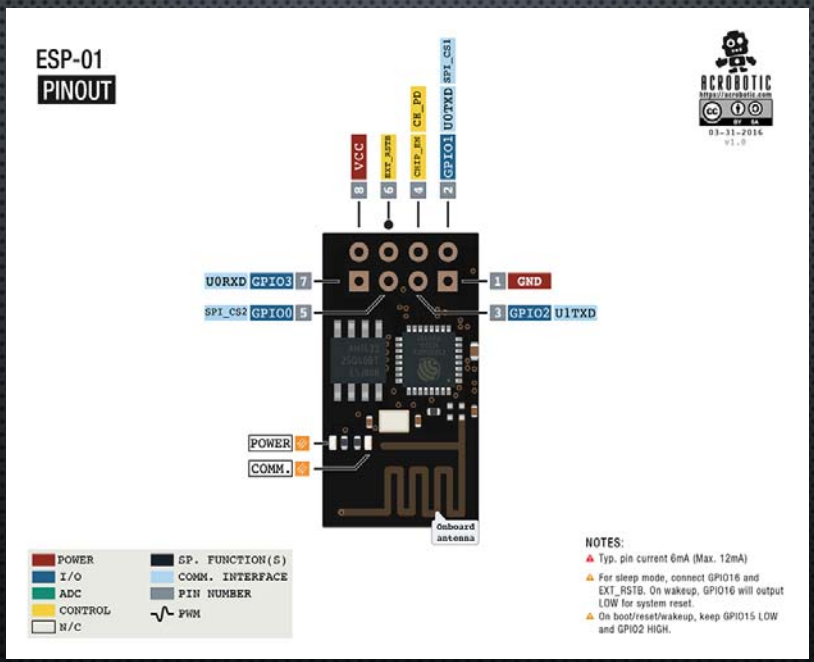
About NodeMCU ESP8266

ESP8266(ESP-01)

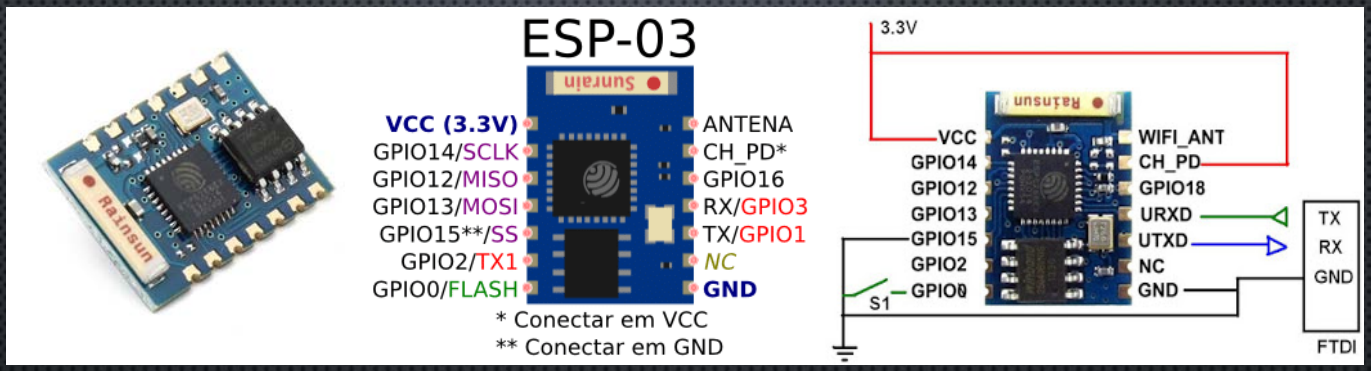
NodeMCU Devkit 0.9 (ESP-12) Version 1

NodeMCU Devkit 1.0 (ESP-12E) Version 2

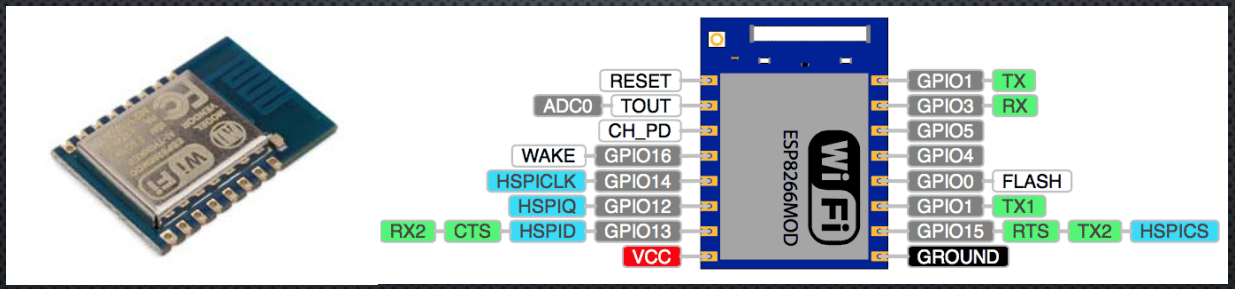
ESP8266 (ESP-01)



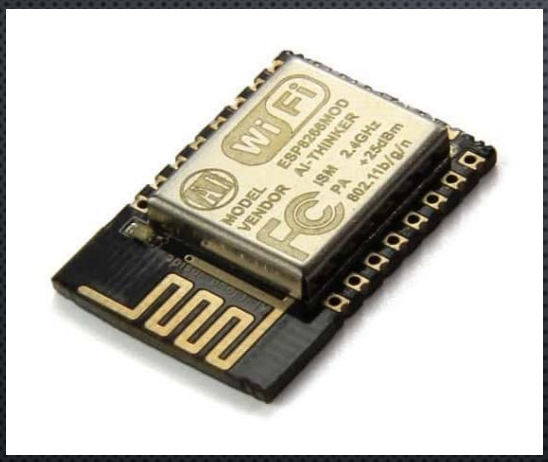
ESP8266 (ESP-03)

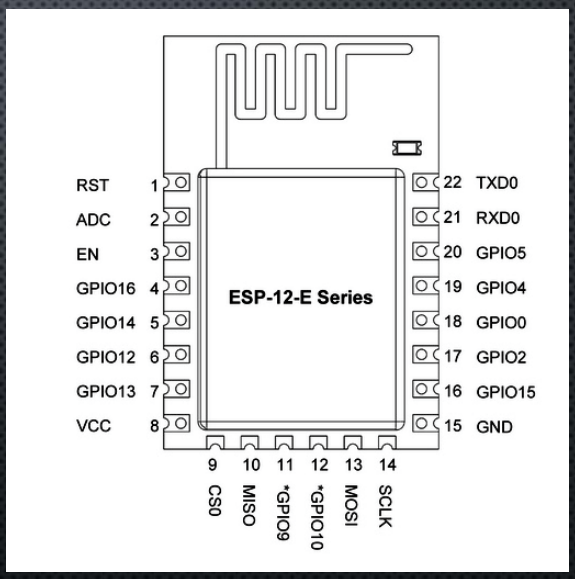


ESP8266 (ESP-07)

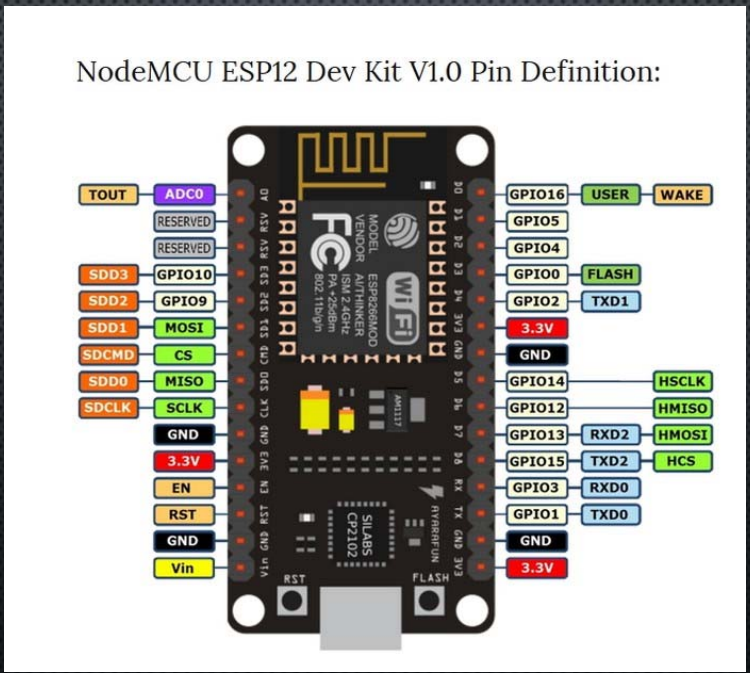


ESP8266 (ESP-12E)

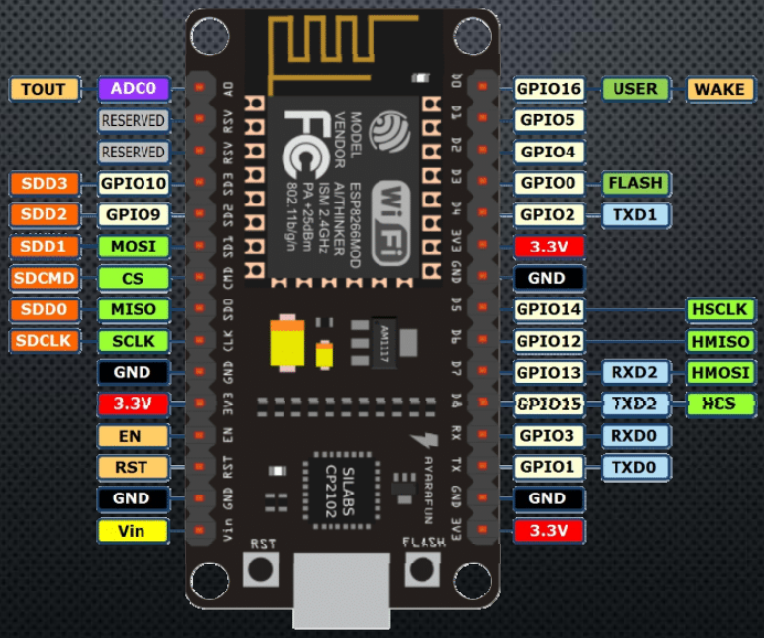




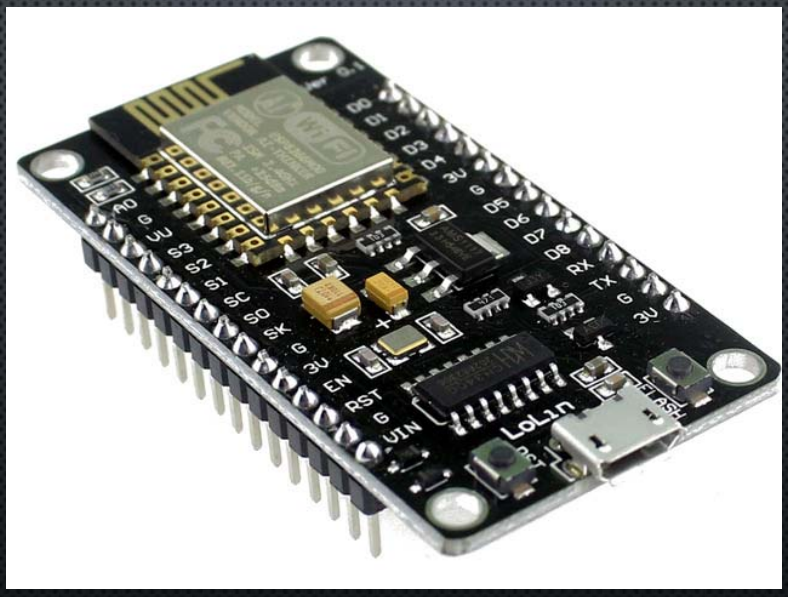
ESP8266 NodeMCU V1 Pin layout



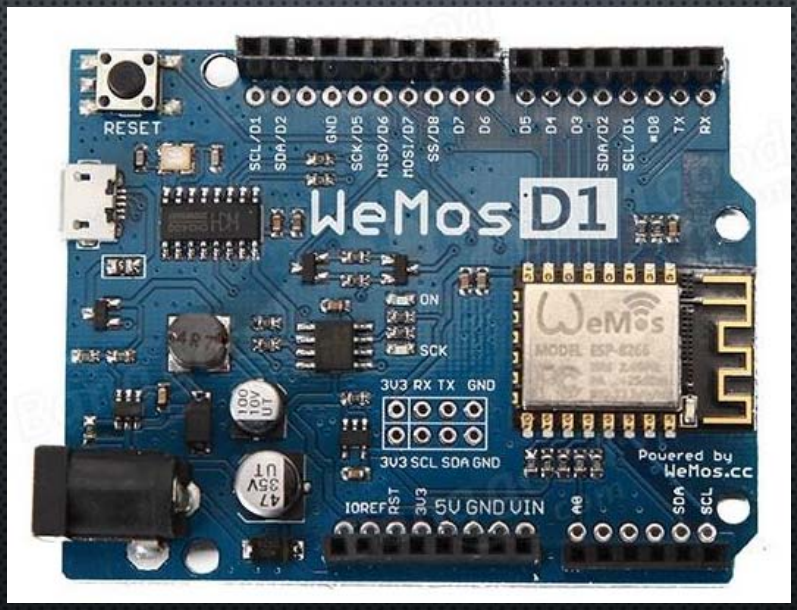
ESP8266 NodeMCU V2 Pin layout



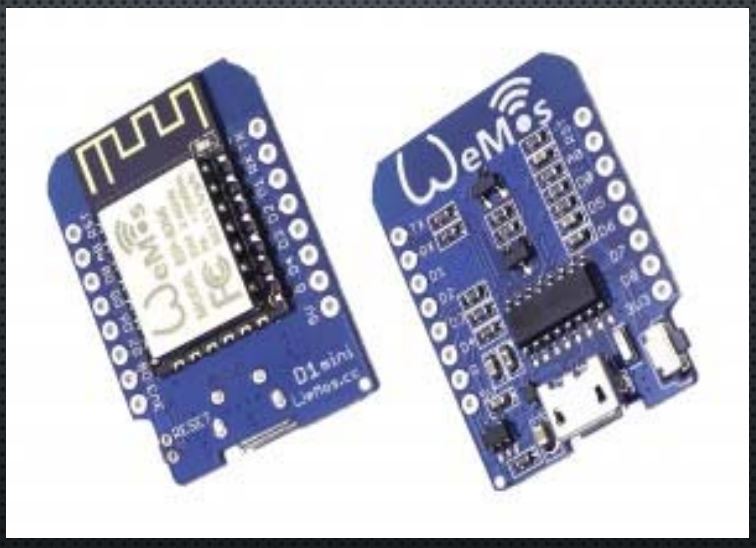
ESP8266 NodeMCU V3 Pin layout



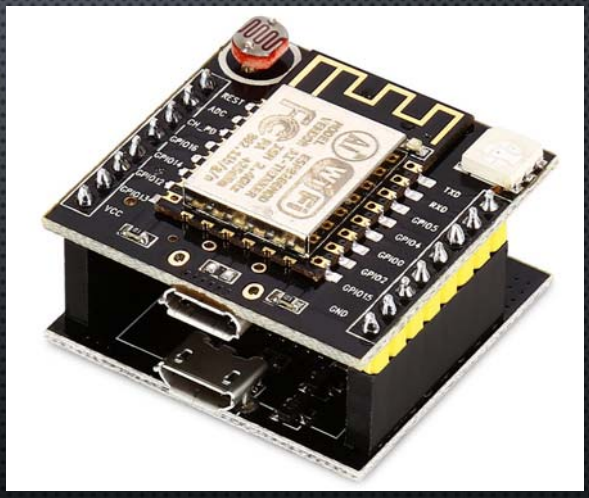
WeMos D1



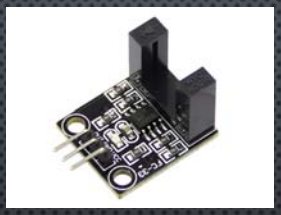
WeMos D1 Mini

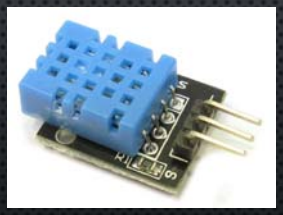


Witty cloud Mini NodeMCU



Sensor & Control modules



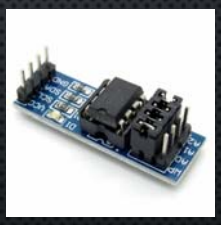


Sensor & Control modules



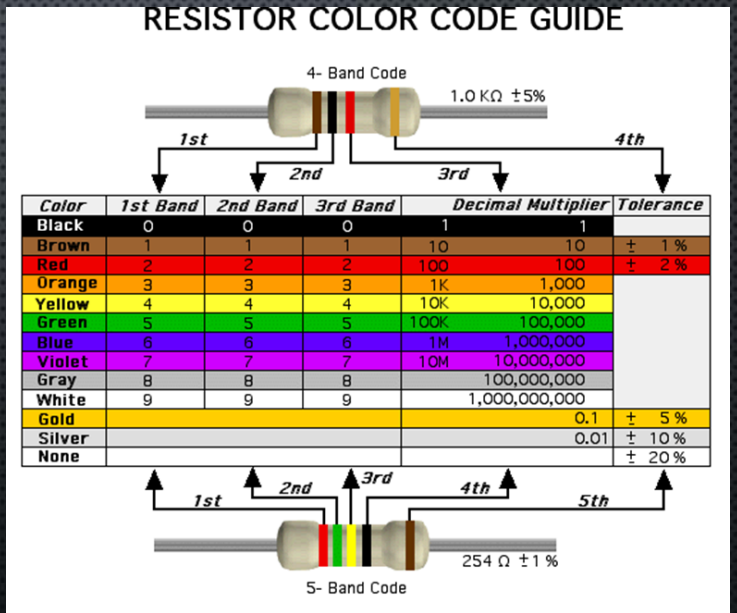




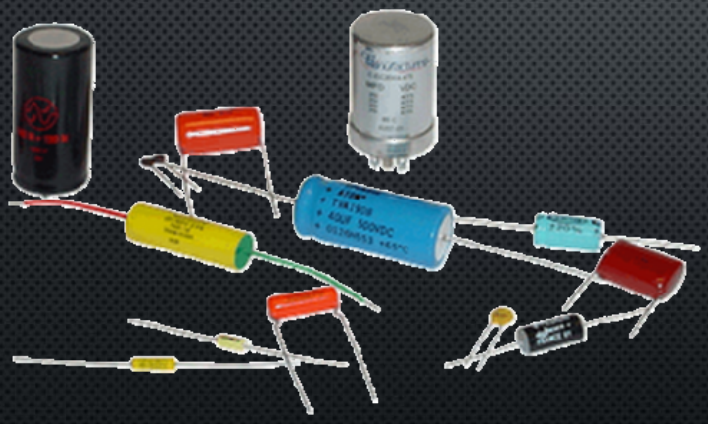
Basic electronic devices

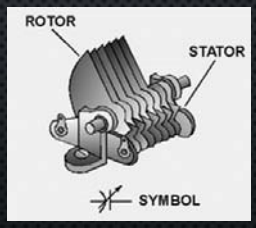
Resistor

Basic electronic devices

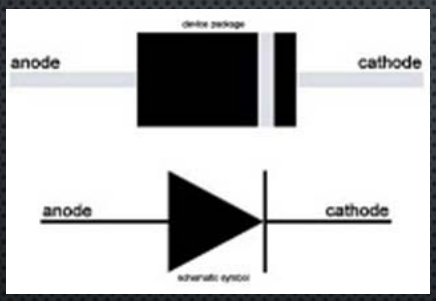
Capacitor



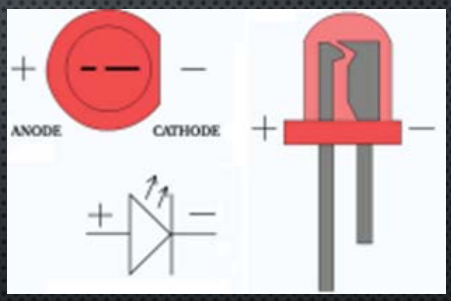


Basic electronic devices

Diode



LED



Tools for Develop

1. Arduino IDE and SDK for ESP8266
2. NETPIE Microgear library for ESP8266
3. CP2012 USB to UART driver
4. Web Browser: Chrome or Firefox

Arduino IDE and SDK for ESP8266

Download and Install Arduino IDE

- https://www.arduino.cc/en/Main/Software

Additional Board Manager URLs

- http://arduino.esp8266.com/stable/package_esp8266com_index.json

Check Install ESP8266 in Boards Manager... Menu

- esp8266 by ESP8266 Community version 2.3.0 INSTALLED

NETPIE Microgear library for ESP8266

Download from GitHub

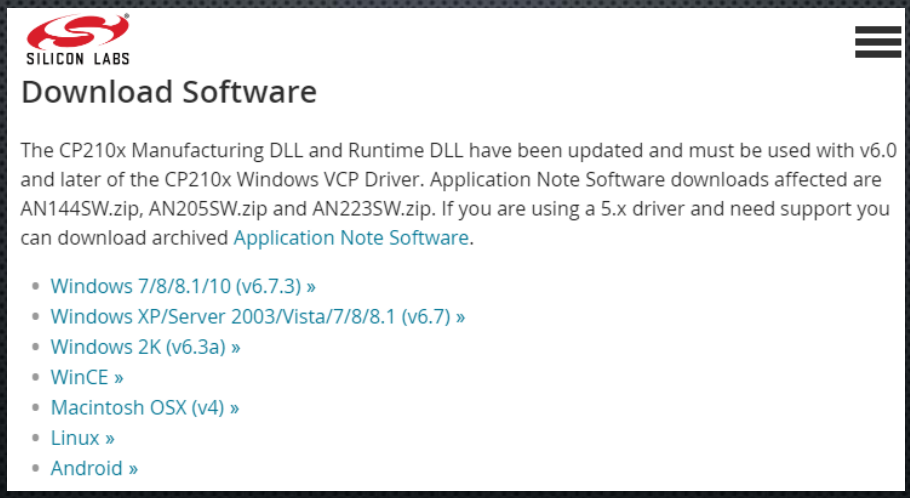
- https://github.com/netpieio/microgear-esp8266-arduino

From Manage Libraries... Menu

- Sketch --> Include Library --> Manage Libraries...
- Search “Microgear” click Choose

CP2012 USB to UART driver

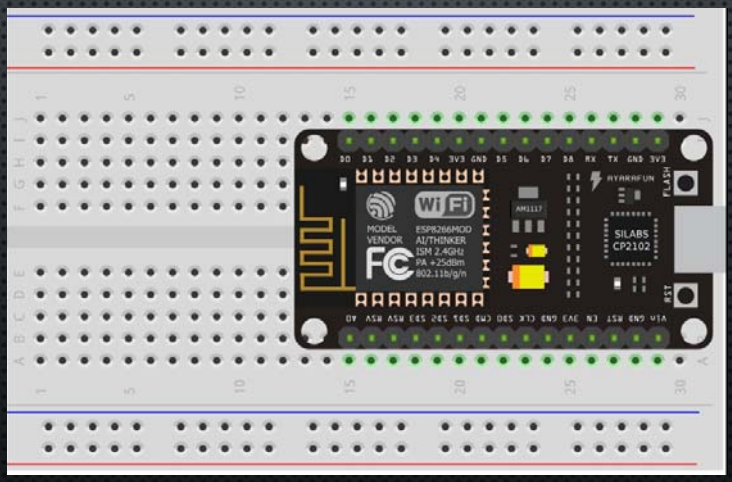
https://www.silabs.com/products/mcu/Pages/USBtoUARTBridgeVCPDrivers.aspx



Workshop#1
Start learning ESP8266 with Arduino IDE

- BLINK
- TOGGLE SWITCH
- WIFI SCAN
- WEB SERVER

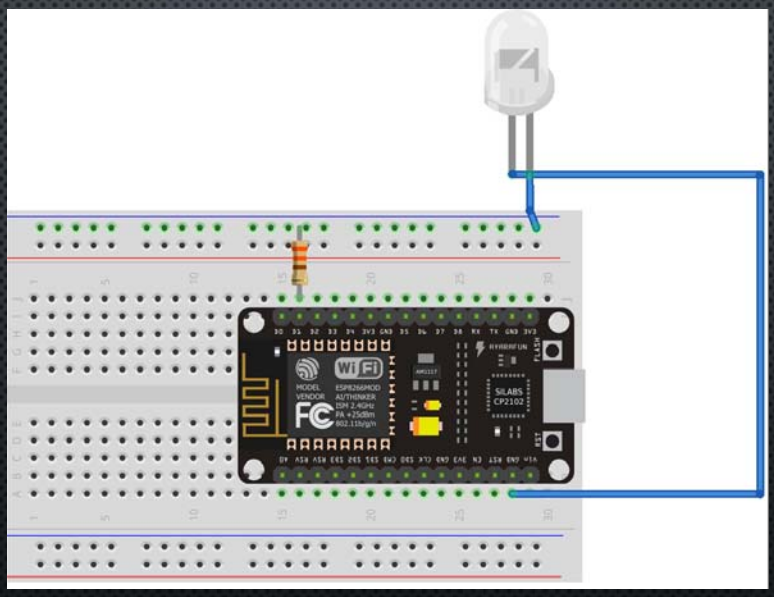
Workshop#1
Prepare hardware snap in breadboard



Workshop#1 [test board with blink Built in LED]

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);     // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, LOW);   // Turn the LED on (Note that LOW is the voltage level
                                    // but actually the LED is on; this is because
                                    // it is acive low on the ESP-01)
  delay(1000);                      // Wait for a second
  digitalWrite(LED_BUILTIN, HIGH);  // Turn the LED off by making the voltage HIGH
  delay(2000);                      // Wait for two seconds (to demonstrate the active low LED)
}
```

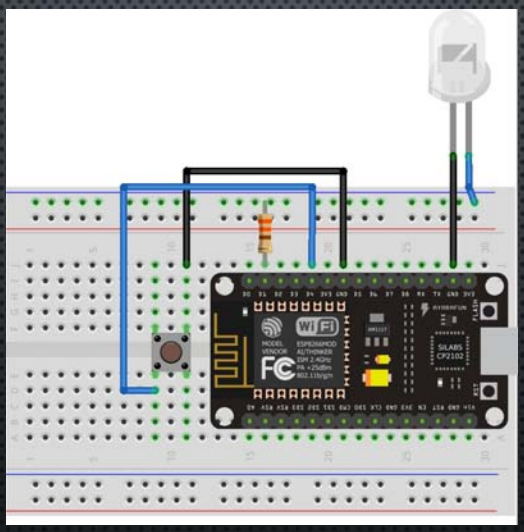
Workshop#1 [test board with blink GPIO5]



Workshop#1 [test board with blink GPIO5]

```
#define ledPin1 D1 //GPIO5
void setup() {
  pinMode(ledPin1, OUTPUT);     // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(ledPin1, HIGH);   // Turn the LED on (Note that LOW is the voltage level
                                 // but actually the LED is on; this is because
                                 // it is acive low on the ESP-01)
  delay(1000);                   // Wait for a second
  digitalWrite(ledPin1, LOW);  // Turn the LED off by making the voltage HIGH
  delay(2000);                   // Wait for two seconds (to demonstrate the active low LED)
}
```

Workshop#1 [test Toggle Switch]



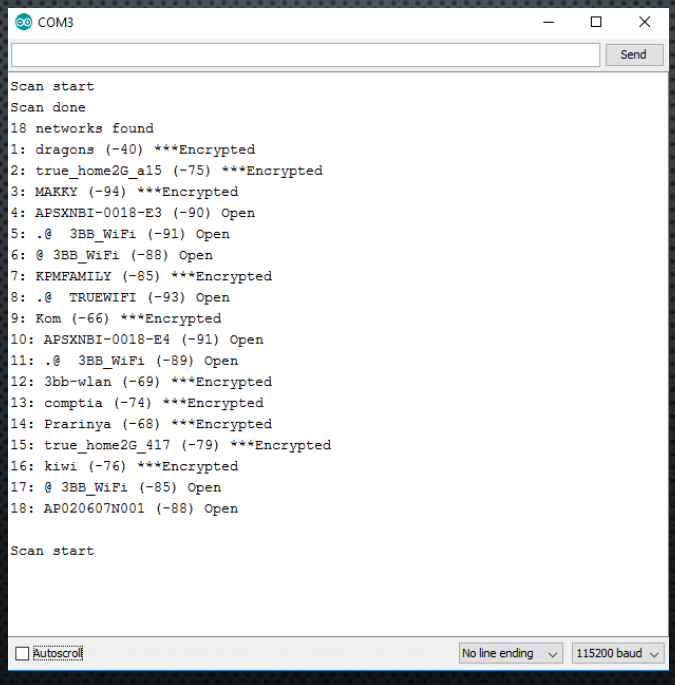
Workshop#1 [test Toggle Switch]

```
#define ledPin D1 // GPIO5
#define sw1 D4 // GPIO2
int st_sw = 0;
int st = 0;
int last_st_sw = 1;
void setup() {
  pinMode(ledPin, OUTPUT); // Set pin mode
  pinMode(sw, INPUT_PULLUP);
}

void loop() {
  st_sw = digitalRead(sw); // Read input port1
  if ((st_sw == 0) && (last_st_sw == 1)) // Check current status
  {
    st = ~st; // Toggle
    digitalWrite(ledPin, st); // Drive LED
    delay(250);
  }
  last_st_sw = st_sw; // Update current status
}
```

Workshop#1 [Wifi scan] - sketch

```
#include "ESP8266WiFi.h"

void setup() {
  Serial.begin(115200);

  // Set WiFi to station mode and disconnect from an AP if it
was previously connected
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  delay(100);

  Serial.println("Setup done");
}
void loop() {
  Serial.println("Scan start");

  // WiFi.scanNetworks will return the number of networks
found
  int n = WiFi.scanNetworks();
  Serial.println("Scan done");
if (n == 0)
    Serial.println("No networks found");
  else
  {
    Serial.print(n);
    Serial.println(" networks found");
    for (int i = 0; i < n; ++i)
    {
      // Print SSID and RSSI for each network found
      Serial.print(i + 1);
      Serial.print(": ");
      Serial.print(WiFi.SSID(i));
      Serial.print(" (");
      Serial.print(WiFi.RSSI(i));
      Serial.print(")");
      Serial.println((WiFi.encryptionType(i) ==
ENC_TYPE_NONE)?" Open":" ***Encrypted");
      delay(10);
    }
  }
  Serial.println("");

  // Wait a bit before scanning again
  delay(5000);
}
```


Workshop#1 [Wifi scan] – serial monitor



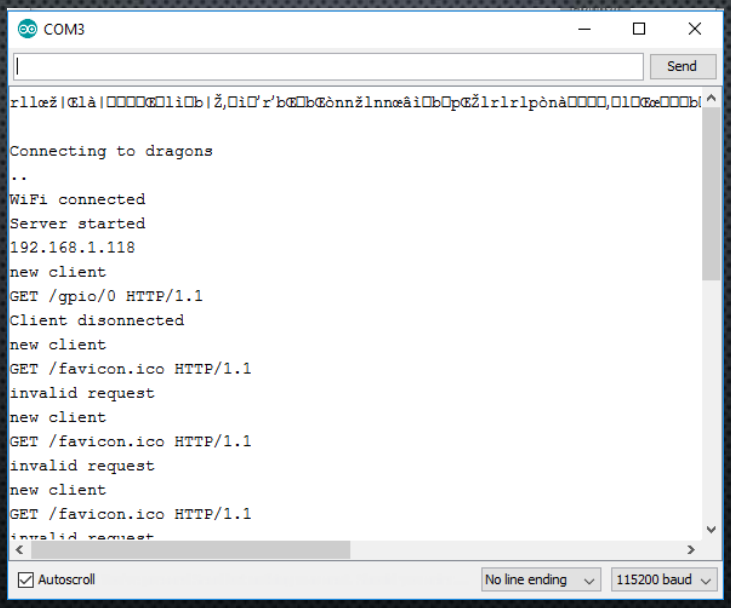
Workshop#1 [WifiWeb server] – sketch GPIO5

```
#include <ESP8266WiFi.h>
#define ledPin D1 //GPIO5

const char* ssid = “your-ssid";
const char* password = “your-password";

// Create an instance of the server
// specify the port to listen on as an argument
WiFiServer server(80);

void setup() {
 Serial.begin(115200);
 delay(10);

 // prepare GPIO5
 pinMode(ledPin, OUTPUT);
 digitalWrite(ledPin, 0);

 // Connect to WiFi network
 Serial.println();
 Serial.println();
 Serial.print("Connecting to ");
 Serial.println(ssid);
 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
 }
 Serial.println("");
 Serial.println("WiFi connected");

 // Start the server
 server.begin();
 Serial.println("Server started");

 // Print the IP address
 Serial.println(WiFi.localIP());
}
```

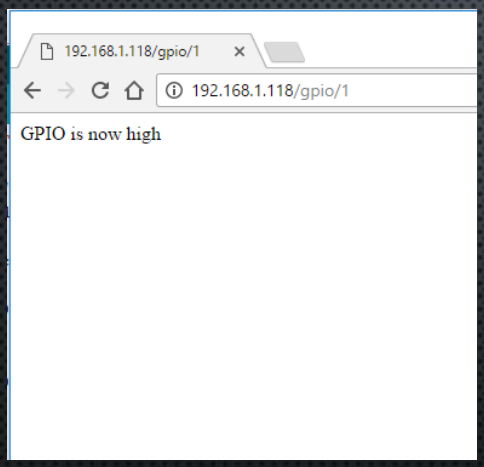

Workshop#1 [WifiWeb server] – sketch GPIO5

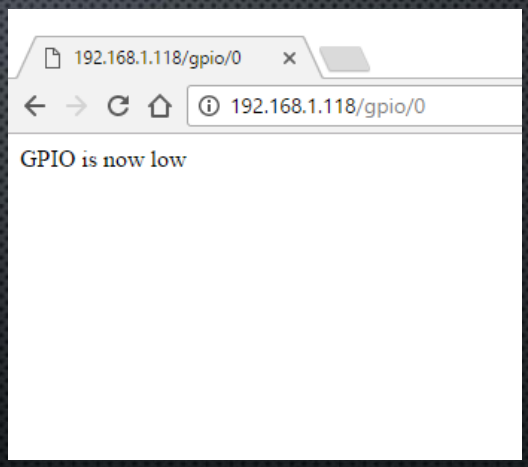
```
void loop() {
 // Check if a client has connected
 WiFiClient client = server.available();
 if (!client) {
  return;
 }

 // Wait until the client sends some data
 Serial.println("new client");
 while(!client.available()){
  delay(1);
 }

 // Read the first line of the request
 String req = client.readStringUntil('\r');
 Serial.println(req);
 client.flush();

 // Match the request
 int val;
 if (req.indexOf("/gpio/0") != -1)
  val = 0;
 else if (req.indexOf("/gpio/1") != -1)
  val = 1;
 else {
  Serial.println("invalid request");
  client.stop();
  return;
 }

 // Set GPIO5 according to the request
 digitalWrite(ledPin, val);

 client.flush();

 // Prepare the response
 String s = "HTTP/1.1 200 OK\r\nContent-Type:
text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\nGPIO is now ";
 s += (val)?"high":"low";
 s += "</html>\n";

 // Send the response to the client
 client.print(s);
 delay(1);
 Serial.println("Client disonnected");

 // The client will actually be disconnected
 // when the function returns and 'client' object is detroyed
}
```

Workshop#1 [WifiWeb server] – serial monitor

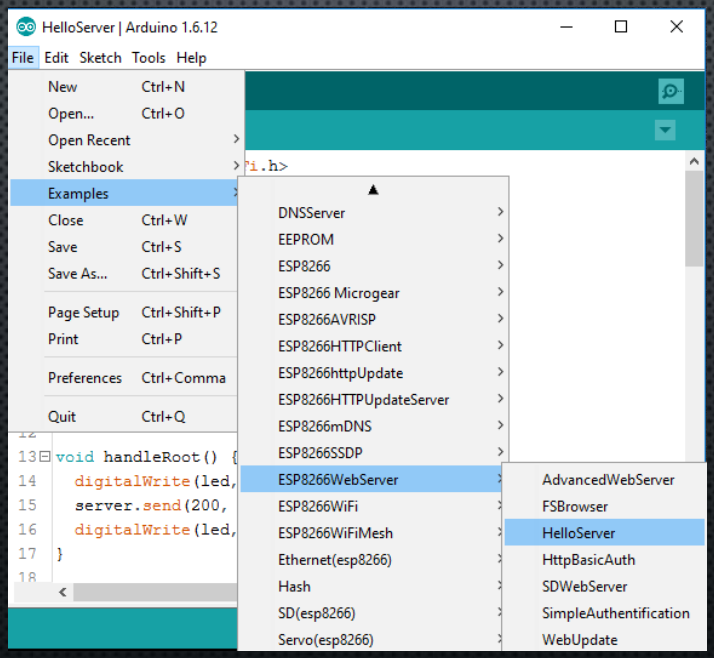


Workshop#1 [WifiWeb server] – web browser

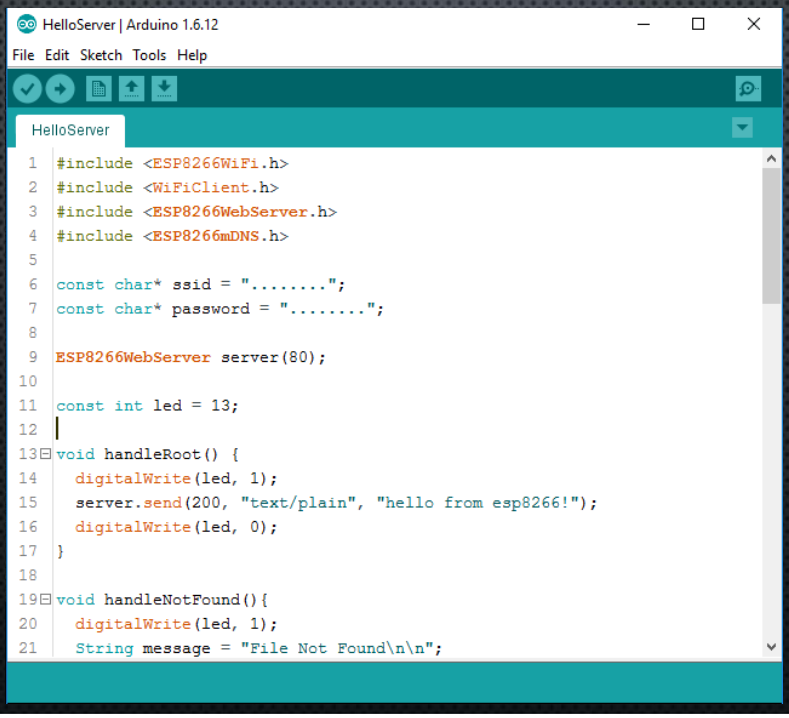




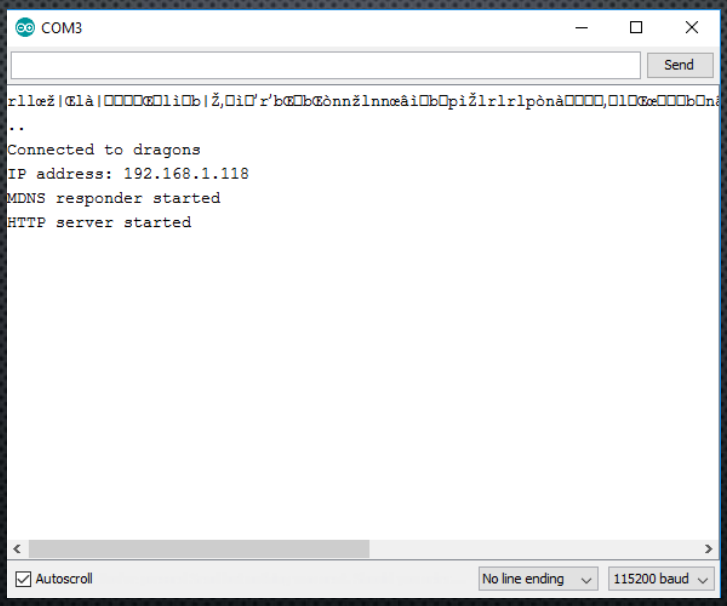
Workshop#1 [HelloServer]



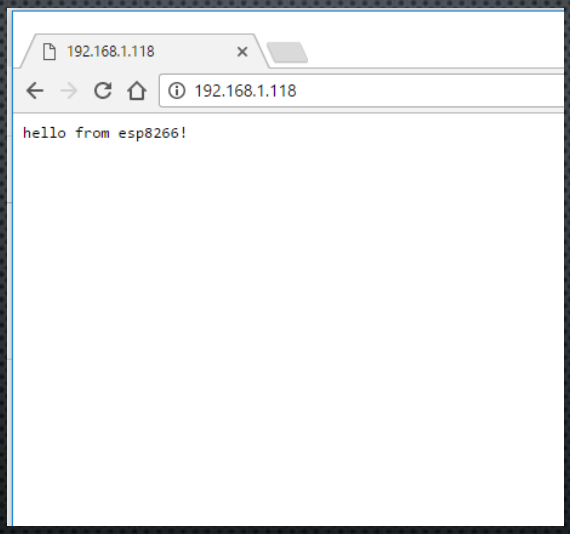
Workshop#1 [HelloServer]



Workshop#1 [HelloServer]-serial monitor



Workshop#1 [HelloServer]-web browser



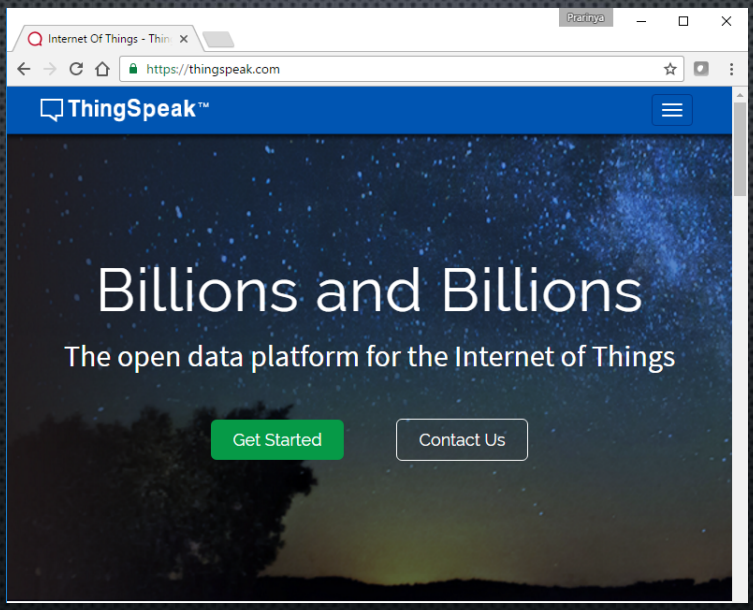
Workshop#2
up data to cloud (Thinkspeak)

- Apply Thinkspeak
- Test send data
- Customize visual board

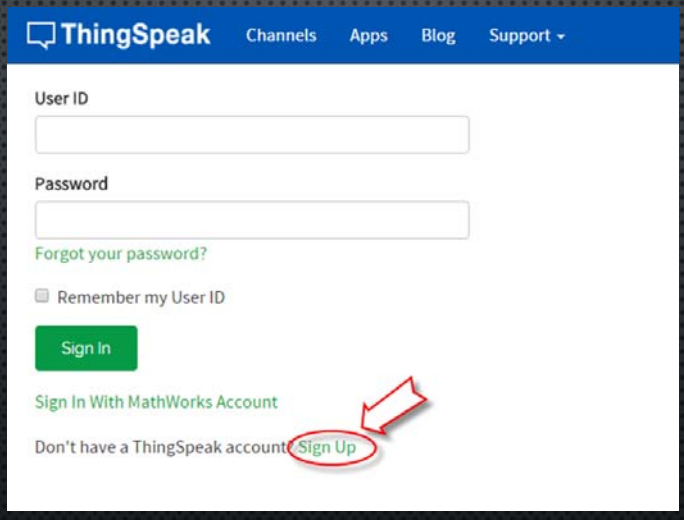
Workshop#2 Apply Thinkspeak

1. Register Thingspeak
2. Create New Channel
3. Receive API Key
4. Use API Key send data to server

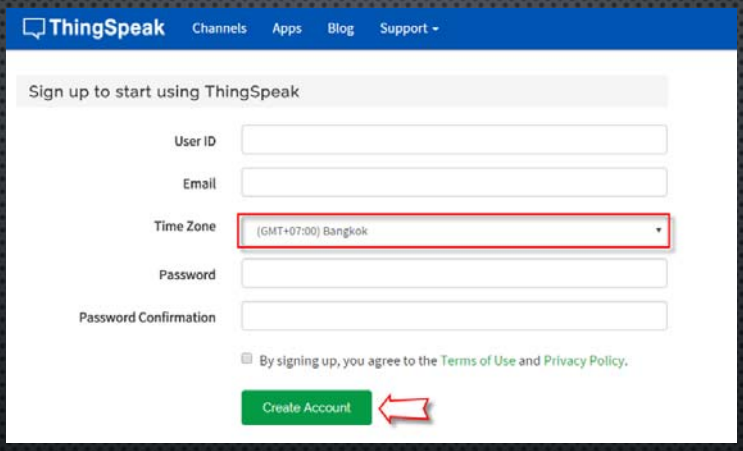
Workshop#2 Apply Thinkspeak



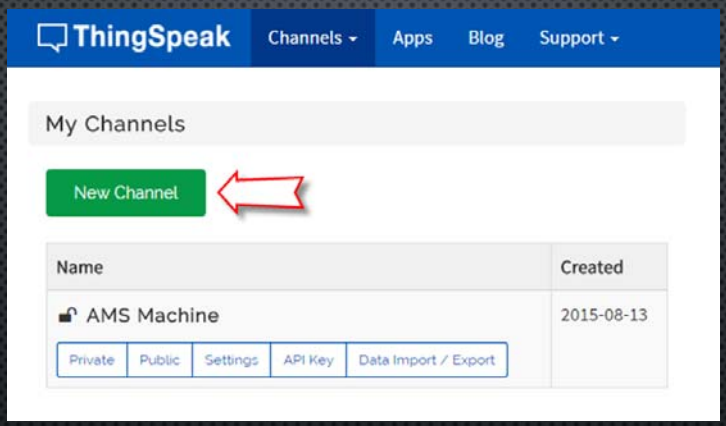
Workshop#2 Apply Thinkspeak



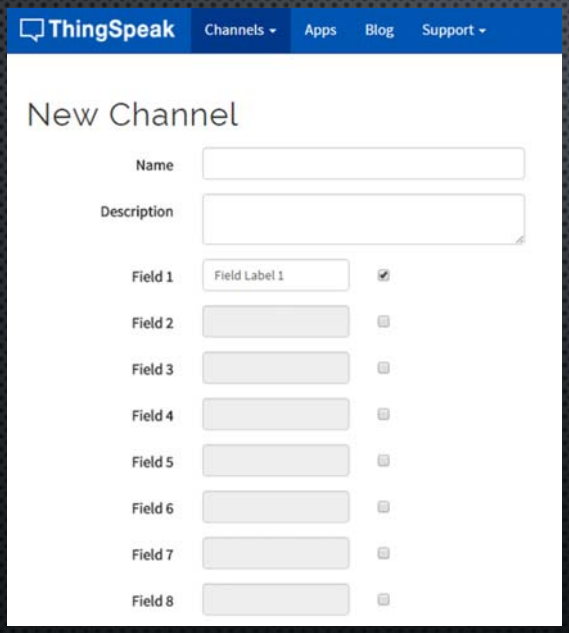
Workshop#2 Apply Thinkspeak

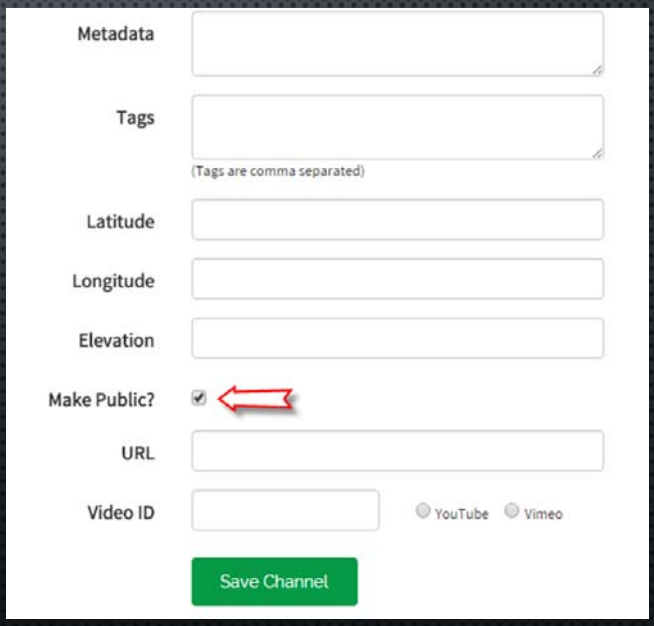


Workshop#2 Create New Channel

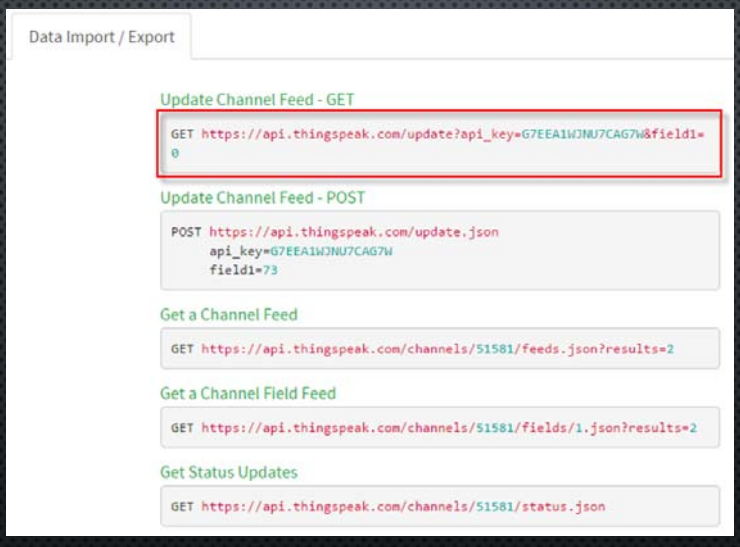


Workshop#2 Create New Channel

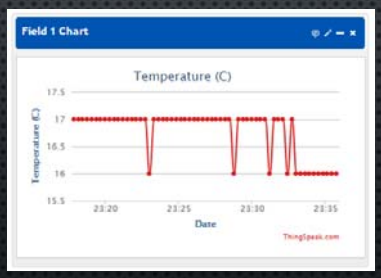




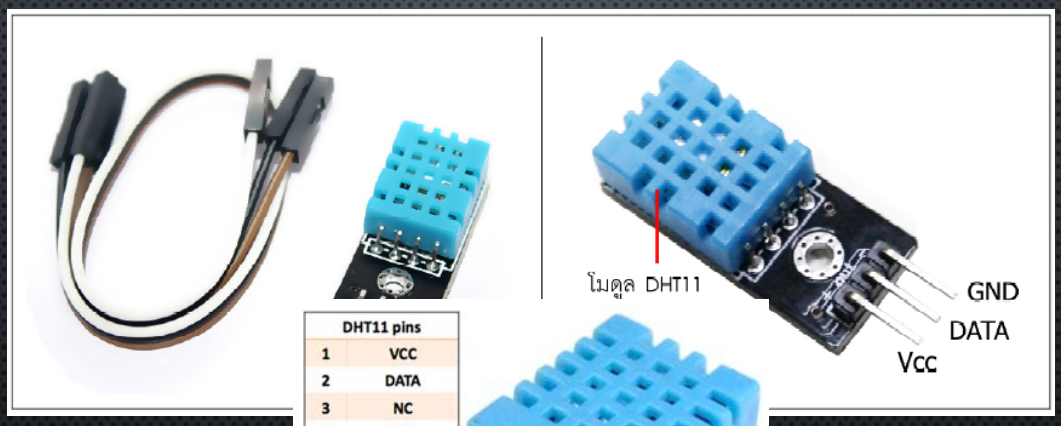
Workshop#2 Create New Channel



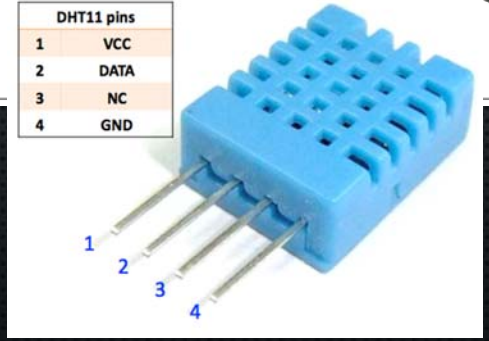
Workshop#2 Create New Channel



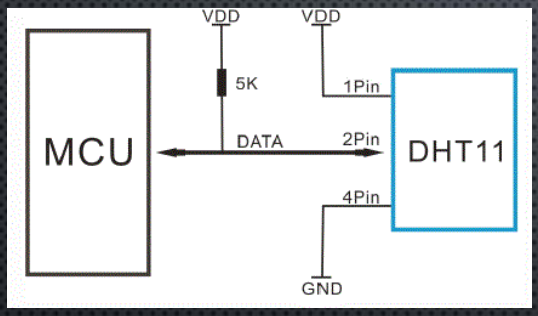
Workshop#2 DHT xx Temperature and Humidity Sensor

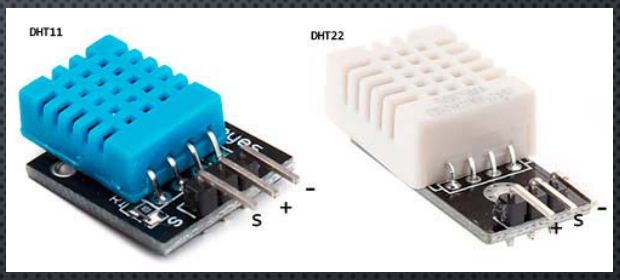


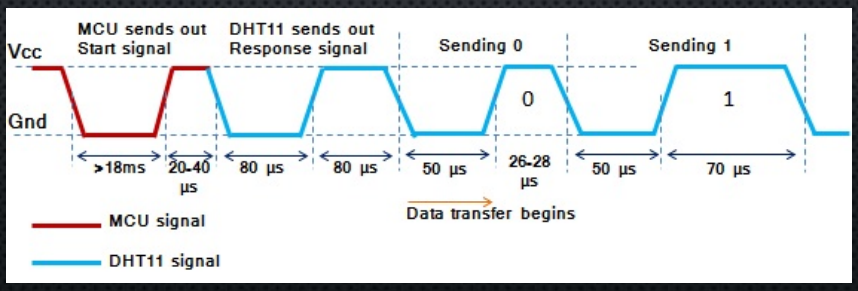
DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



Workshop#2 DHT xx Temperature and Humidity Sensor







Workshop#2 DHT xx Temperature and Humidity Sensor

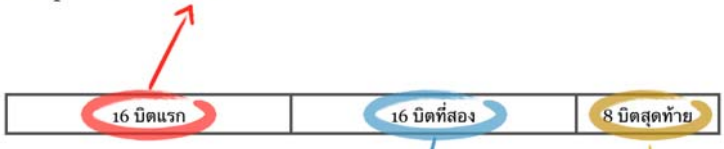
DHT 11

Temperature = xx . x °C

Humidity = xx

DHT 22

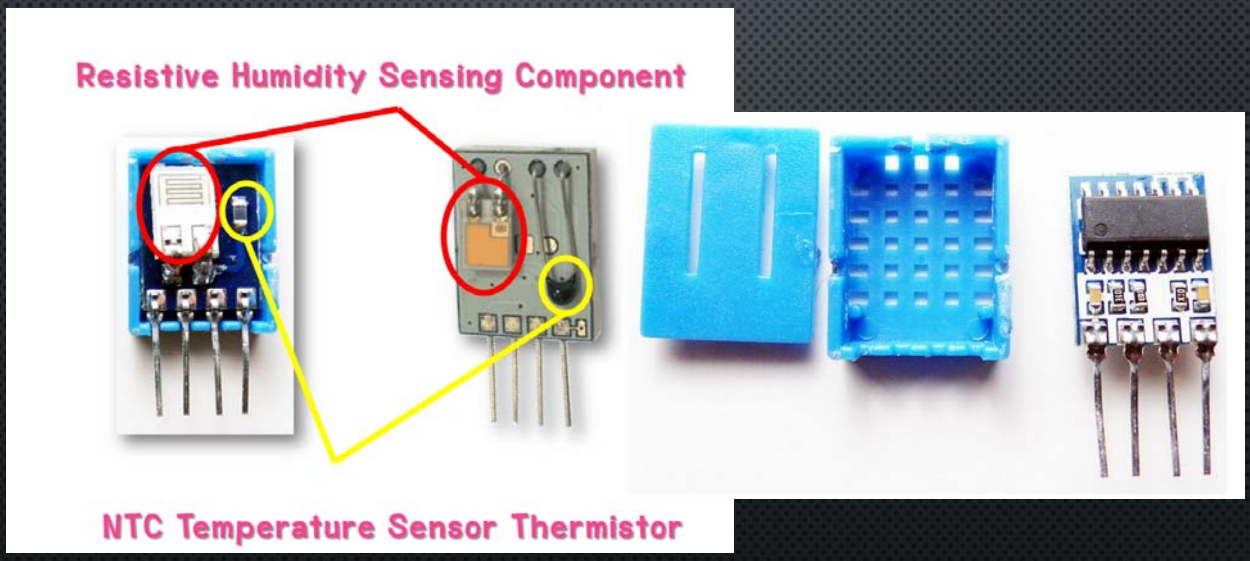
Temperature = xx.x °C



Humidity = xx.x %

Check Error

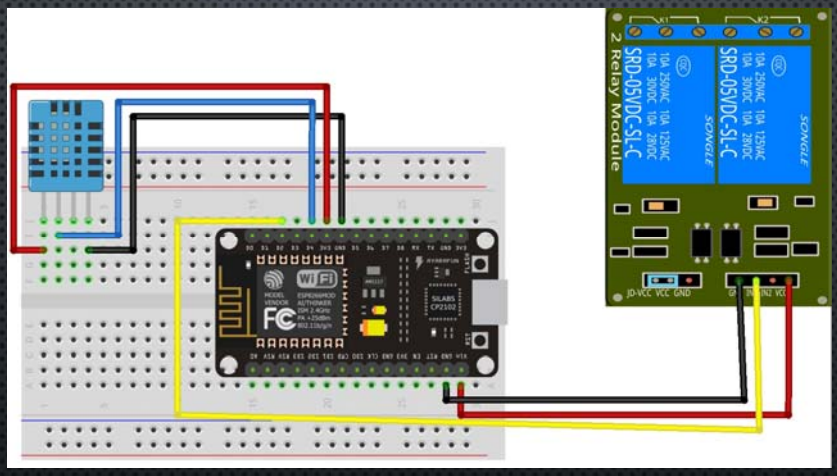
Workshop#2 DHT xx Temperature and Humidity Sensor



Workshop#2 DHT xx Temperature and Humidity Sensor

ความแตกต่าง	DHT11	DHT22
ขอบเขตและความเคลื่อนในการวัดความชื้นสัมพัทธ์	ความชื้น : 20 - 80 % ความคลาดเคลื่อน : 5 %	ความชื้น : 0 - 100 % ความคลาดเคลื่อน : 5 %
ขอบเขตและความเคลื่อนในการวัดอุณหภูมิ	อุณหภูมิ : 0 - 50 °C ความคลานเคลื่อน : ± 2 °C	อุณหภูมิ : -40 - 125 °C ความคลานเคลื่อน : ± 0.5 °C
ความว่องไวในการประมวลผล	1 Hz (1 ครั้งต่อวินาที)	0.5 Hz (2 ครั้งต่อวินาที)
ขนาด	กว้าง x ยาว x สูง : 5.5 mm x 12 mm x 15.5 mm	กว้าง x ยาว x สูง : 7.7 mm x 15.1 mm x 25 mm
ราคา	ประมาณ 100 บาท	ประมาณ 200 บาท

Workshop#2 connect hardware



Workshop#2 sketch for test send data

```
#include "DHT.h"
#include <ESP8266WiFi.h>

#define PUMP_RLY 14    // output drive relay for pump GPIO14 (D5)
#define DHTPIN 2     // what pin we're connected to GPIO2 (D4)
#define DHTTYPE DHT11   // DHT 11

#define DEBUG

#define DEBUG_PRINTER Serial

#ifdef DEBUG
#define DEBUG_PRINT(...) { DEBUG_PRINTER.print(__VA_ARGS__); }
#define DEBUG_PRINTLN(...) { DEBUG_PRINTER.println(__VA_ARGS__); }
#else
#define DEBUG_PRINT(...) {}
#define DEBUG_PRINTLN(...) {}
#endif

const char* ssid     = "SSID"; //Use own ssid
const char* password = "PASSWORD"; //use password of AP

DHT *dht;

void connectWifi();
void reconnectWifiIfLinkDown();
void initDht(DHT **dht, uint8_t pin, uint8_t dht_type);
void readDht(DHT *dht, float *temp, float *humid);
void uploadThingsSpeak(float t, float h);

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(PUMP_RLY, OUTPUT); // Initialize the PUMP_RLY(4) pin as
an output
  digitalWrite(PUMP_RLY, HIGH);// Make sure relay is normal off

  connectWifi();

  initDht(&dht, DHTPIN, DHTTYPE);
}

void loop() {
  static float t_dht;
  static float h_dht;

  readDht(dht, &t_dht, &h_dht);
  if(h_dht > 50 || t_dht > 26) // condition for make relay on
  {
    digitalWrite(PUMP_RLY, LOW); //If condition true do this!
  } else
  {
    digitalWrite(PUMP_RLY, HIGH);
  }
  uploadThingsSpeak(t_dht, h_dht);

  // Wait a few seconds between measurements.
  delay(10 * 1000);
  reconnectWifiIfLinkDown();
}
```

Workshop#2 sketch for test send data

```
void reconnectWifiIfLinkDown() {
  if (WiFi.status() != WL_CONNECTED) {
    DEBUG_PRINTLN("WIFI DISCONNECTED");
    connectWifi();
  }
}

void connectWifi() {
  DEBUG_PRINTLN();
  DEBUG_PRINTLN();
  DEBUG_PRINT("Connecting to ");
  DEBUG_PRINTLN(ssid);

  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    DEBUG_PRINT(".");
  }

  DEBUG_PRINTLN("");
  DEBUG_PRINTLN("WiFi connected");
  DEBUG_PRINTLN("IP address: ");
  DEBUG_PRINTLN(WiFi.localIP());
}

void initDht(DHT **dht, uint8_t pin, uint8_t dht_type) {
  // Connect pin 1 (on the left) of the sensor to +5V
  // NOTE: If using a board with 3.3V logic like an Arduino Due
connect pin 1
  // to 3.3V instead of 5V!
  // Connect pin 2 of the sensor to whatever your DHTPIN is
  // Connect pin 4 (on the right) of the sensor to GROUND
  // Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the
sensor

  // Initialize DHT sensor for normal 16mhz Arduino
  // NOTE: For working with a faster chip, like an Arduino Due or
Teensy, you
  // might need to increase the threshold for cycle counts
considered a 1 or 0.
  // You can do this by passing a 3rd parameter for this threshold.
It's a bit
  // of fiddling to find the right value, but in general the faster the
CPU the
  // higher the value.  The default for a 16mhz AVR is a value of 6.
For an
  // Arduino Due that runs at 84mhz a value of 30 works.
  // Example to initialize DHT sensor for Arduino Due:
  //DHT dht(DHTPIN, DHTTYPE, 30);

  *dht = new DHT(pin, dht_type, 30);
  (*dht)->begin();
  DEBUG_PRINTLN(F("DHTxx test!"))  ;
}
```

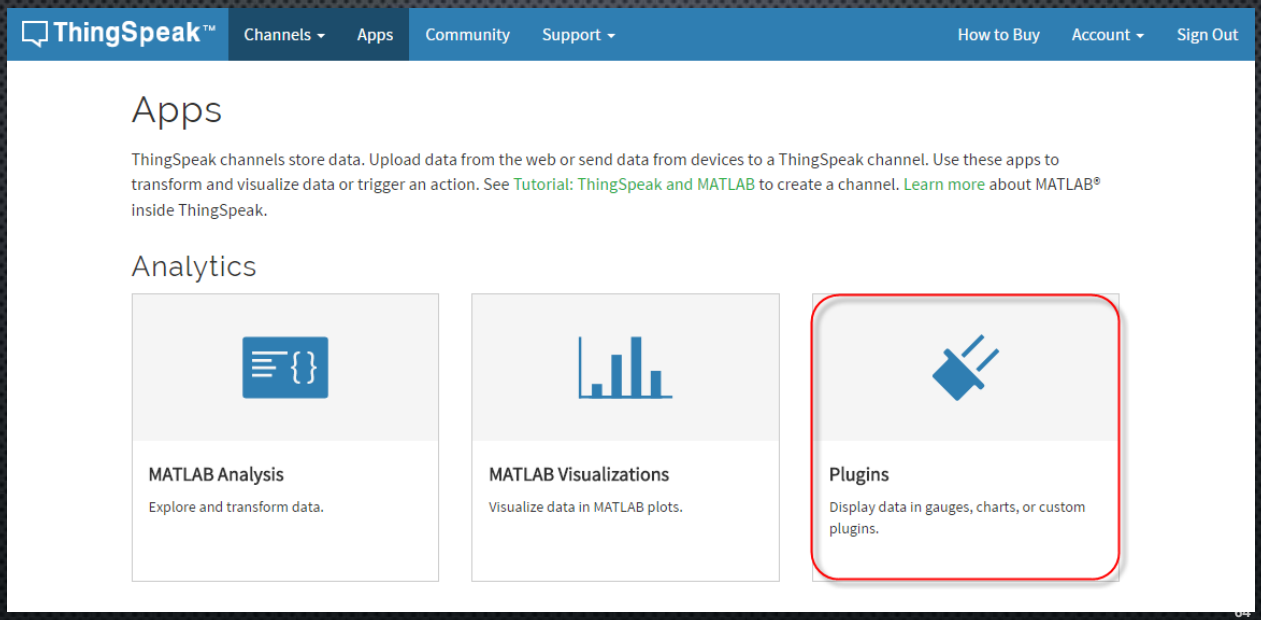

Workshop#2 sketch for test send data

```
void uploadThingsSpeak(float t, float h) {
  static const char* host = "api.thingspeak.com";
  static const char* apiKey = "YOUR_APIKEY";

  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    DEBUG_PRINTLN("connection failed");
    return;
  }
  // We now create a URI for the request
  String url = "/update/";
  //  url += streamId;
  url += "?key=";
  url += apiKey;
  url += "&field1=";
  url += t;
  url += "&field2=";
  url += h;

  DEBUG_PRINT("Requesting URL: ");
  DEBUG_PRINTLN(url);

  // This will send the request to the server
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
}

void readDht(DHT *dht, float *temp, float *humid) {

  if (dht == NULL) {
    DEBUG_PRINTLN(F("[dht11] is not initialised. please call initDht()
first."));
    return;
  }

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very
slow sensor)
  float h = dht->readHumidity();

  // Read temperature as Celsius
  float t = dht->readTemperature();
  // Read temperature as Fahrenheit
  float f = dht->readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    DEBUG_PRINTLN("Failed to read from DHT sensor!");
    return;
  }
```

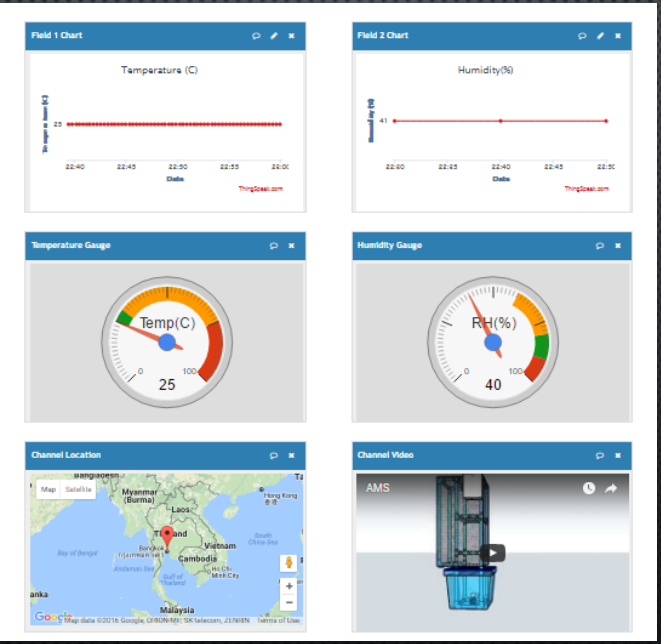
Workshop#2 sketch for test send data

```
// Compute heat index
  // Must send in temp in Fahrenheit!
  float hi = dht->computeHeatIndex(f, h);

  DEBUG_PRINT("Humidity: ");
  DEBUG_PRINT(h);
  DEBUG_PRINT(" %\t");
  DEBUG_PRINT("Temperature: ");
  DEBUG_PRINT(t);
  DEBUG_PRINT(" *C ");
  DEBUG_PRINT(f);
  DEBUG_PRINT(" *F\t");
  DEBUG_PRINT("Heat index: ");
  DEBUG_PRINT(hi);
  DEBUG_PRINTLN(" *F");

  *temp = t;
  *humid = h;
}
```


Customize visual board



Customize visual board



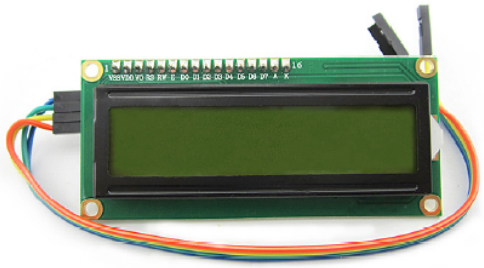
Workshop#3

- Display and control device
- Apply to use NETPIE
- Test send data
- Customize visual board and control

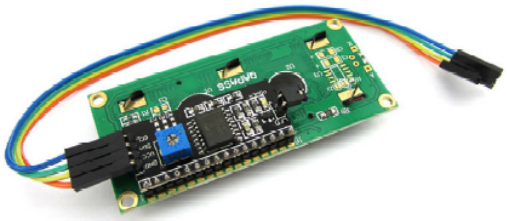
Workshop#3 LCD with I²C Module [Key feature]

- Display 16x2 Character
- I²C Interface with MCU
- 2 Address Bus of I²C
  - Chip PCF8574 -> 0x27
  - Chip PCF8574A -> 0x3F
- Use 4 Wires Vcc(+),GND(G),SDA,SCL
- Use +5V for Power

Workshop#3 LCD with I²C Module

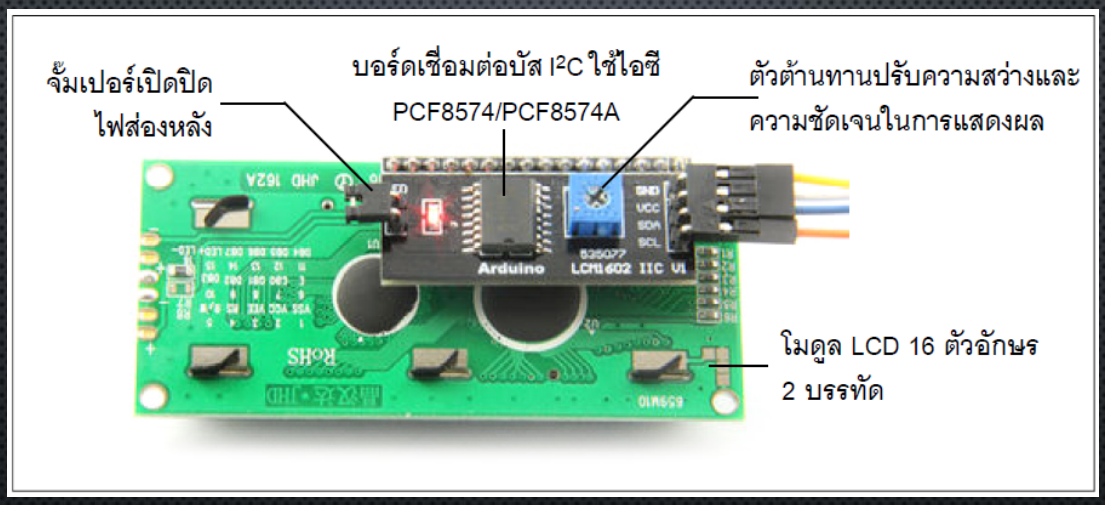


(ก) ภาพด้านหน้าของโมดูล LCD แบบบัส I²C

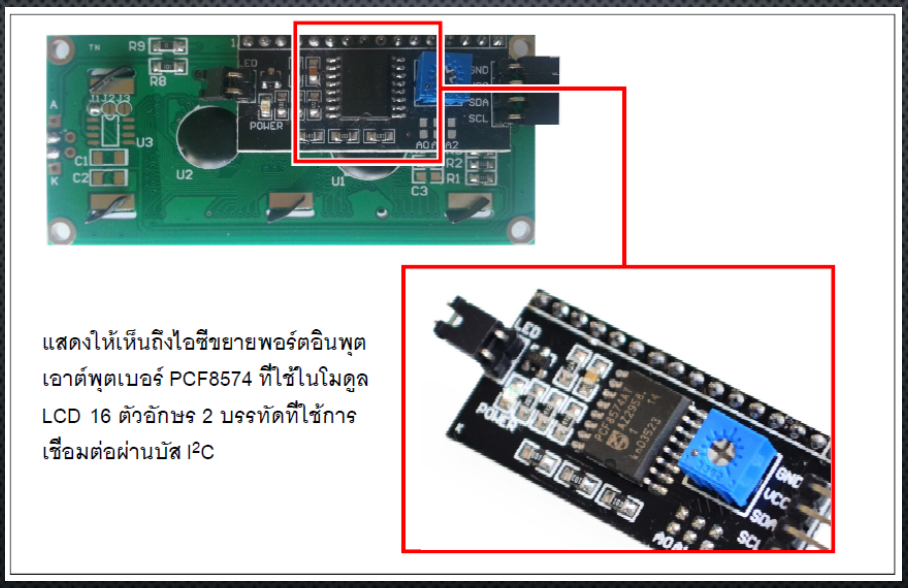


(ข) ภาพด้านหลังของโมดูล LCD แบบบัส I²C แสดงให้เห็นถึงบอร์ดเชื่อมต่อบัส I²C ที่ใช้ไอซีเบอร์ PCF8574 หรือ PCF8574A รวมถึงตัวต้านทานปรับค่าได้สำหรับปรับความชัดเจนในการแสดงผล

Workshop#3 LCD with I²C Module

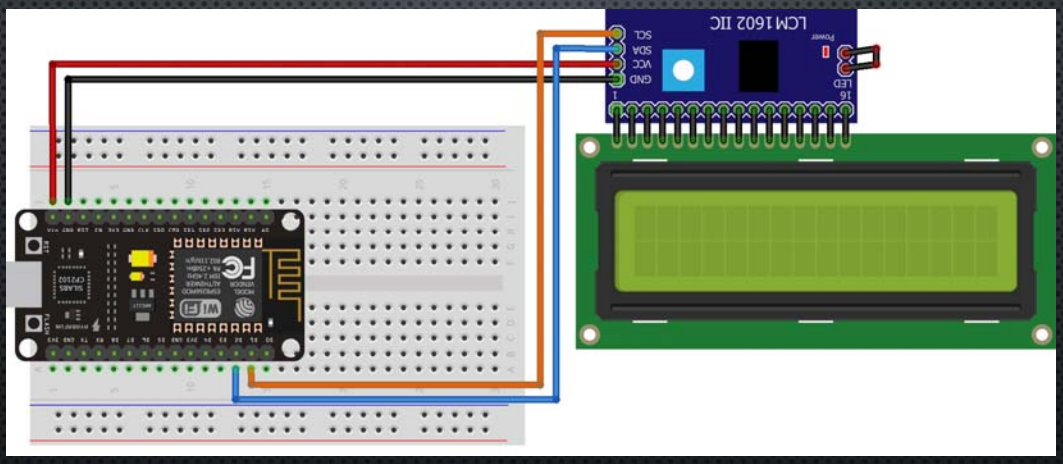


Workshop#3 LCD with I²C Module



แสดงให้เห็นถึงไอซีขยายพอร์ตอินพุตเอาต์พุตเบอร์ PCF8574 ที่ใช้ในโมดูล LCD 16 ตัวอักษร 2 บรรทัดที่ใช้การเชื่อมต่อผ่านบัส I²C

Workshop#3 connect LCD with I2C



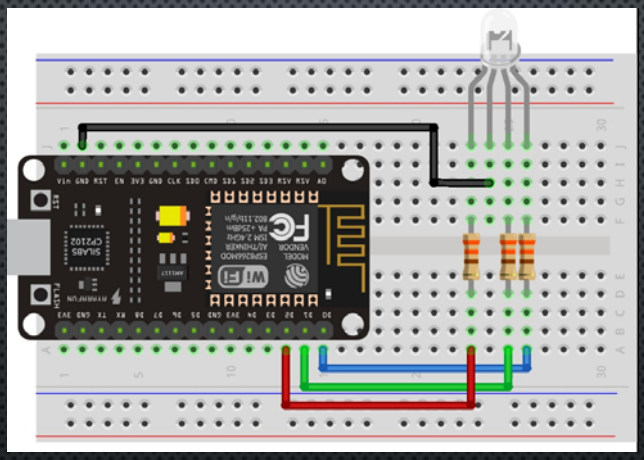
Workshop#3 sketch for test LCD

```
//SCL as D1/GPIO5
//SDA as D2/GPIO4
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
// Initial I2C-LCD
// Address is 0x27 (for PCF8574) or 0x3F (for PCF8574A)
// Type 16 characters 2 lines
//LiquidCrystal_I2C lcd(0x3F, 16, 2);
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup()
{
lcd.init(); // Start
lcd.backlight(); // Enable LED backlight
lcd.setCursor(0,0); // Set home cursor
lcd.print("L1-Hello World"); // Display message on line 1 (upper)
lcd.setCursor(0,1); // Set new position
lcd.print("L2-IoT Training"); // Display message on line 2 (lower)
}
void loop(){
 // put your main code here, to run repeatedly:
}
```

Workshop#3 LCD I²C serial workshop

```
#include <Wire.h>  // Comes with Arduino IDE
#include <LiquidCrystal_I2C.h>

//LiquidCrystal_I2C lcd(0x3F, 16,2);
LiquidCrystal_I2C lcd(0x27, 16,2);

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  lcd.init();
  for(int i = 0; i<3; i++)
  {
    lcd.backlight();
    delay(250);
    lcd.noBacklight();
    delay(250);
  }
  lcd.backlight();

  lcd.setCursor(0,0);
  lcd.print(".....");
  delay(1000);
  lcd.setCursor(0,1);
  lcd.print(".....");
  delay(8000);
}

void loop()
{
  {
  // when characters arrive over the serial port...
  if(Serial.available()){
    // wait a bit for the entire message to arrrive
    delay(300);
    // clear the screen
    lcd.clear();
    // read all the available characters
    while(Serial.available()>0)
    {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
   }
  }
}
```

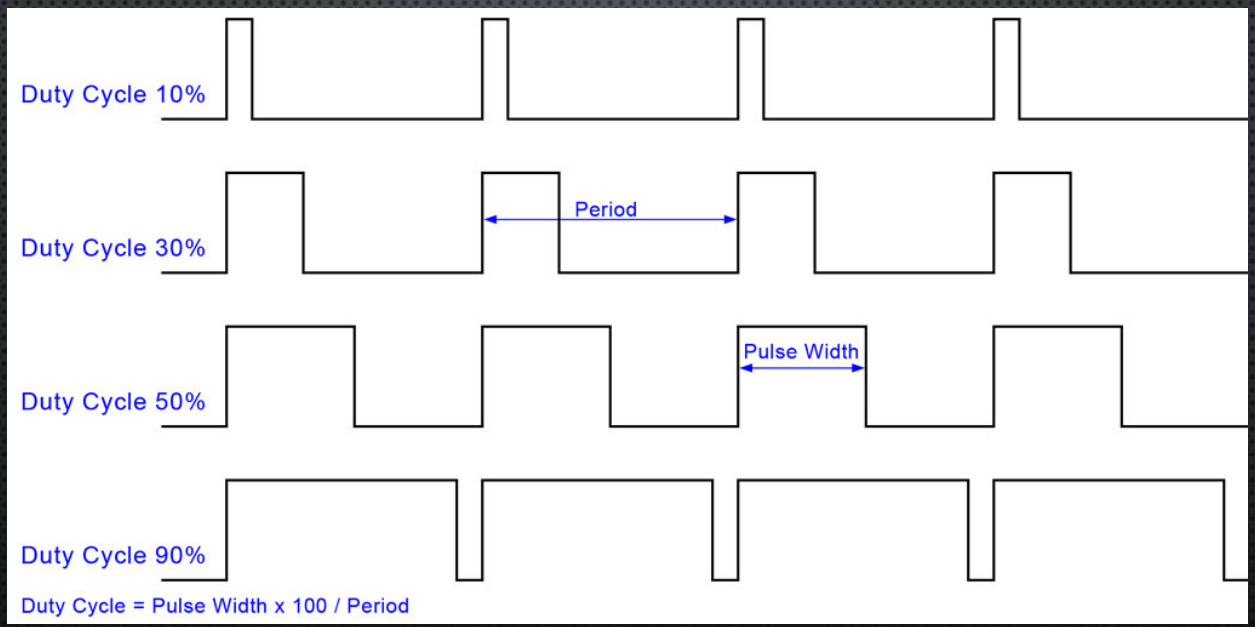

Workshop#3 connect LED 3 color (RGB)



Workshop#3 NodeMCU LED 3 color

```
#define B_pin D0 //Blue
#define G_pin D1 //Green
#define R_pin D2 //Red
int st_B = 0;
int st_G = 0;
int st_R = 0;
void setup() {
  // put your setup code here, to run once:
  pinMode(B_pin, OUTPUT); // Set pin as output
  pinMode(G_pin, OUTPUT);
  pinMode(R_pin, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  displayRGB(0, 0, 0); // Drive RGB LED with 3-bit data
  delay(2000); // Delay 2 seconds
  displayRGB(1, 0, 0);
  delay(2000);
  displayRGB(1, 1, 0);
  delay(2000);
  displayRGB(0, 1, 0);
  delay(2000);
  displayRGB(0, 1, 1);
  delay(2000);
  displayRGB(0, 0, 1);
  delay(2000);
  displayRGB(1, 0, 1);
  delay(2000);
  displayRGB(1, 1, 1);
  delay(2000);
}
void displayRGB(int R, int G, int B)
// Send 3-bit digital data to 3 output pins function
{
  digitalWrite(B_pin, B);
  digitalWrite(G_pin, G);
  digitalWrite(R_pin, R);
}
```


Workshop#3 Pulse Width Modulation (PWM)



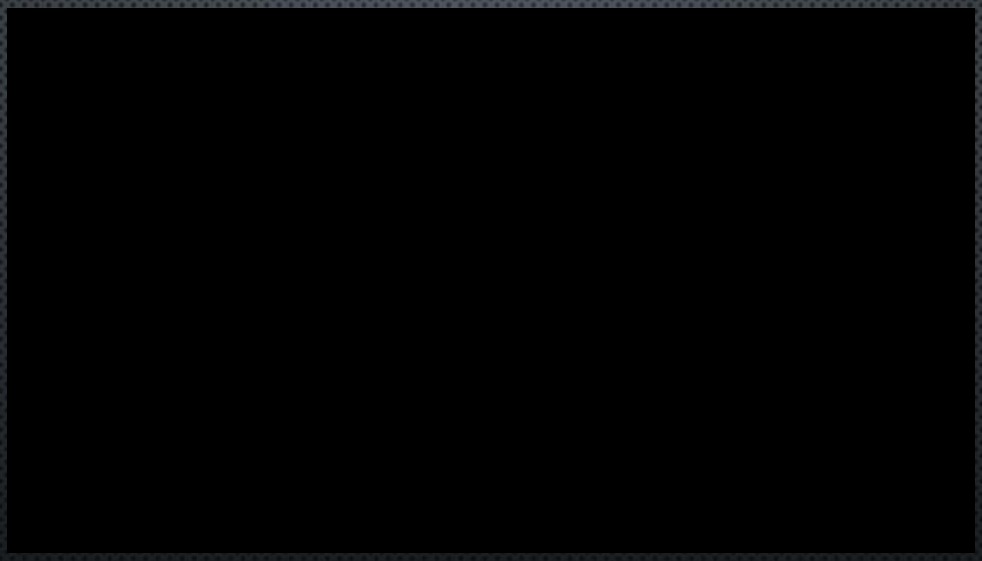
Workshop#3 NodeMCU LED 3 color PWM

```
#define B_pin D0 //Blue
#define G_pin D1 //Green
#define R_pin D2 //Red
int st_R = 255;
int st_G = 0;
int st_B = 0;
void setup() {
  // put your setup code here, to run once:
  pinMode(B_pin, OUTPUT); // Set pin as output
  pinMode(G_pin, OUTPUT);
  pinMode(R_pin, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  for (int i = 0; i < 256; i++) // Set loop
  {
    st_B = 0; // Clear BLUE data
    st_G++; // Increase GREEN data
    displayRGB(st_R, st_G, st_B); // Drive LED with PWM
    delay(30); // Short delay
  }
for (int i = 0; i < 256; i++)
{
  st_R--; // Decrease RED data
  displayRGB(st_R, st_G, st_B); // Drive LED with PWM
  delay(30); // Short delay
}
for (int i = 0; i < 256; i++)
{
  st_R = 0; // Clear RED data
  st_B++; // Increase BLUE data
  displayRGB(st_R, st_G, st_B); // Drive LED with PWM
  delay(30); // Short delay
}
for (int i = 0; i < 256; i++)
{
  st_G--; // Decrease GREEN data
  displayRGB(st_R, st_G, st_B); // Drive LED with PWM
  delay(30); // Short delay
}
```


Workshop#3 NodeMCU LED 3 color PWM

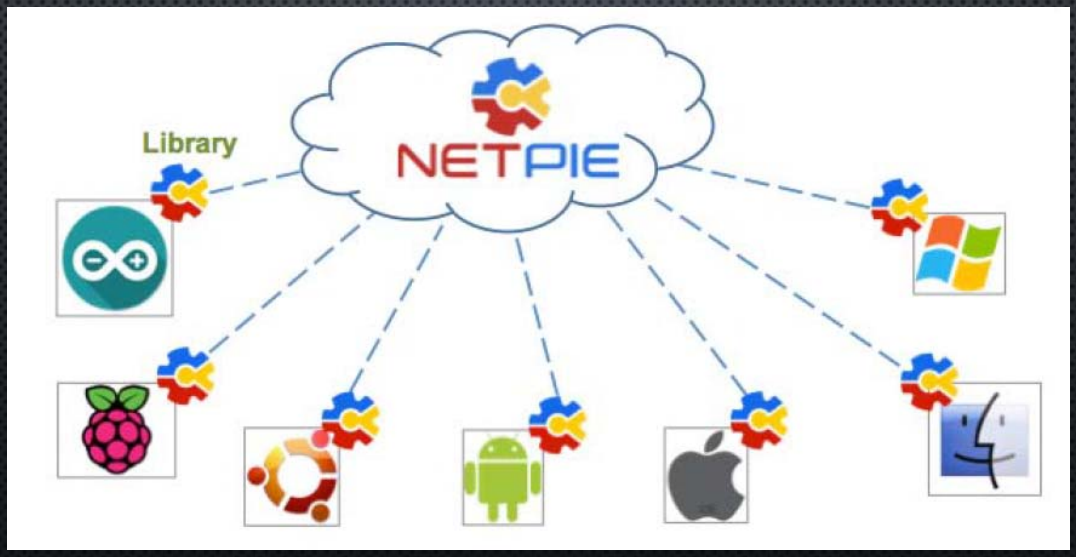
```
  for (int i = 0; i < 256; i++)
  {
    st_G = 0; // Clear GREEN data
    st_R++; // Increase RED data
    displayRGB(st_R, st_G, st_B); // Drive LED with PWM
    delay(30); // Short delay
  }
  for (int i = 0; i < 256; i++)
  {
    st_B--; // Decrease BLUE data
    displayRGB(st_R, st_G, st_B); // Drive LED with PWM
    delay(30); // Short delay
  }
}
void displayRGB(int R, int G, int B)
// Drive LED with PWM by using analogWrite function
{
  analogWrite(B_pin, B);
  analogWrite(G_pin, G);
  analogWrite(R_pin, R);
}
```

What is "NETPIE"

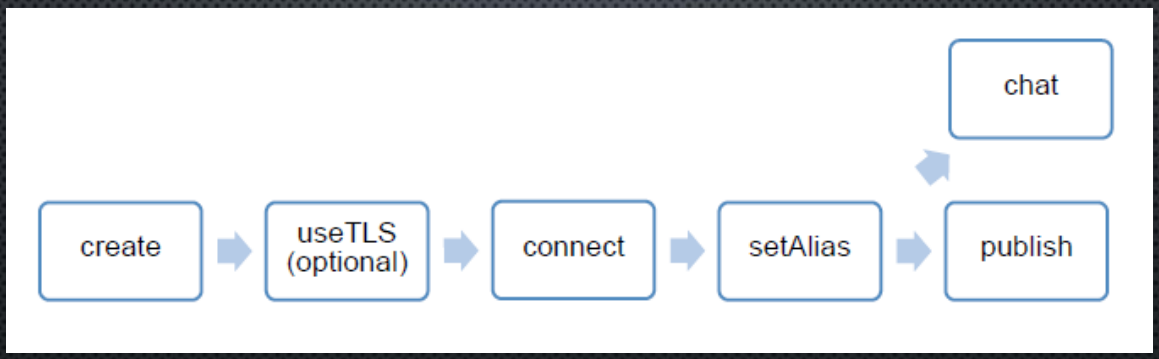


About NETPIE

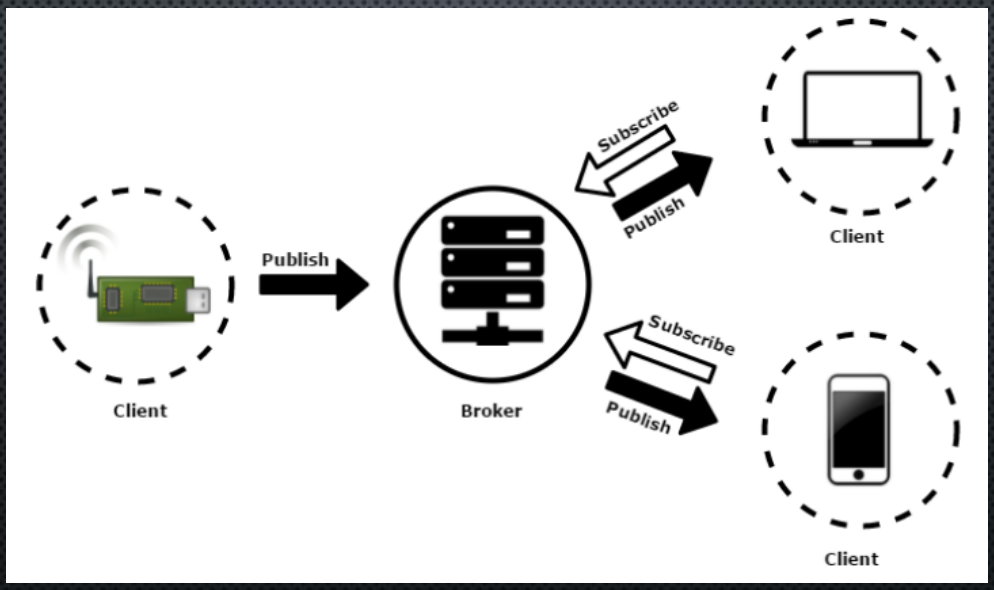
NETPIE = Cloud Platform + Microgear Libraries



Basic function of Microgear



MQTT (MQ TELEMETRY TRANSPORT)



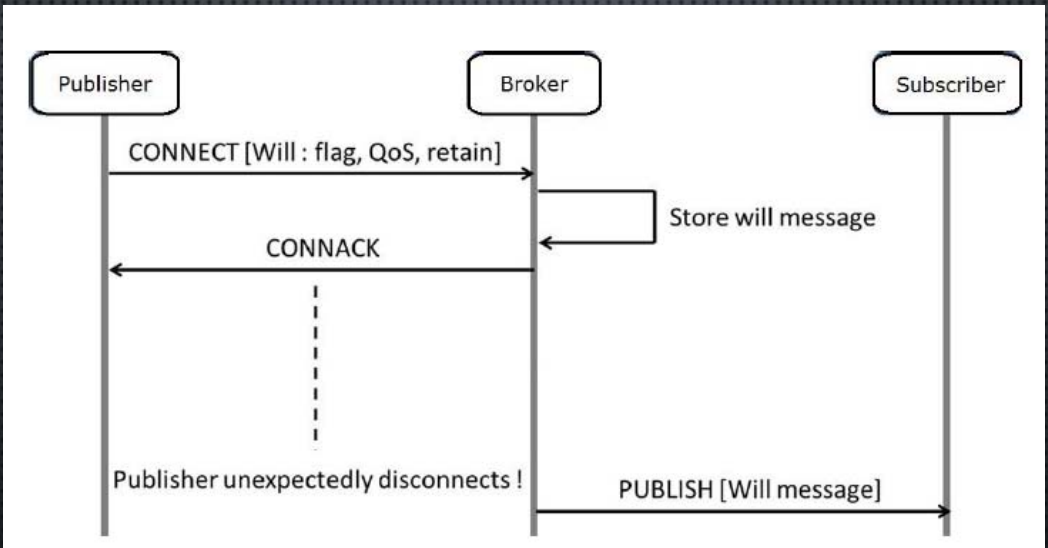
MQTT Topics

- UTF-8 String
- File Path -> *myhome/floor-one/room-c/temperature*
- Single-Lever Wildcard (+)
  - *myhome/floor-one/+/temperature*
- Multi-Level Wildcard (#)
  - *myhome/floor-one/#*

MQTT Connections

Control Packets	Sender		Description
	Broker	Client	
CONNECT		X	Request connect
CONNACK	X		Acknowledge connect
PUBLISH	X	X	Message for Publish
PUBACK	X	X	Published (QoS Level 1)
PUBREC	X	X	Published (QoS Level 2)
PUBREL	X	X	Published and Remove message(QoS Level 2)
PUBCOM	X	X	Publish complete and delete status (QoS Level 2)
SUBSCRIBE		X	Request Subscribe
SUBACK	X		Acknowledge Subscribe
UNSUBSCRIBE		X	Cancel Subscribe
UNSUBACK	X		Acknowledge Cancel Subscribe
PINGREQ		X	PING Request
PINGRESP	X		PING Response
DISCONNECT		X	Cancel connect

Last Will Message



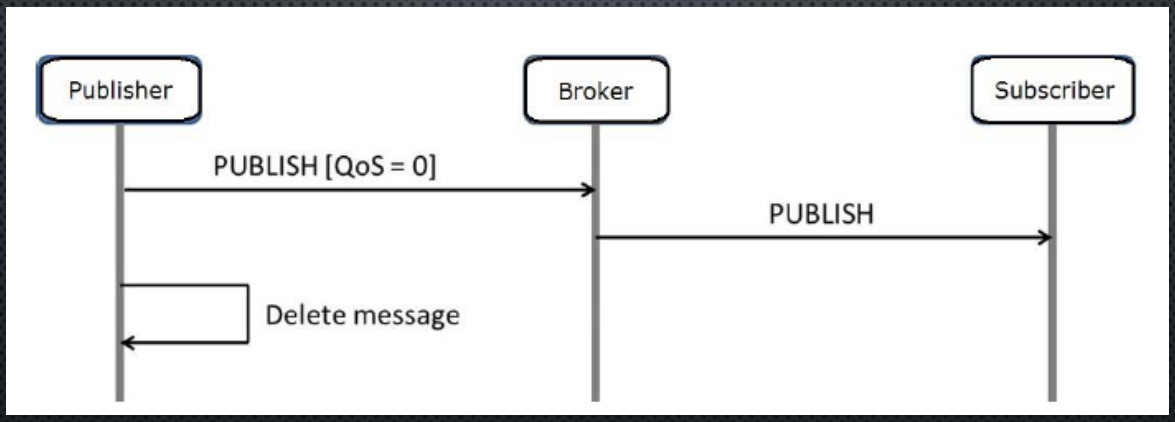
MQTT Quality of Service (QoS)

0 At Most Once

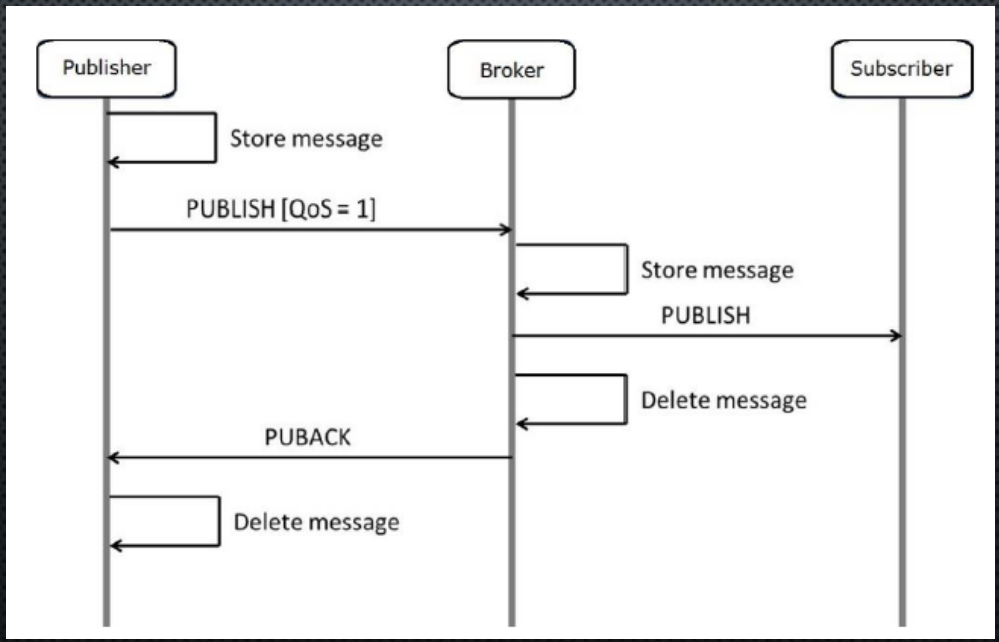
1 At Least Once

2 Exactly Once

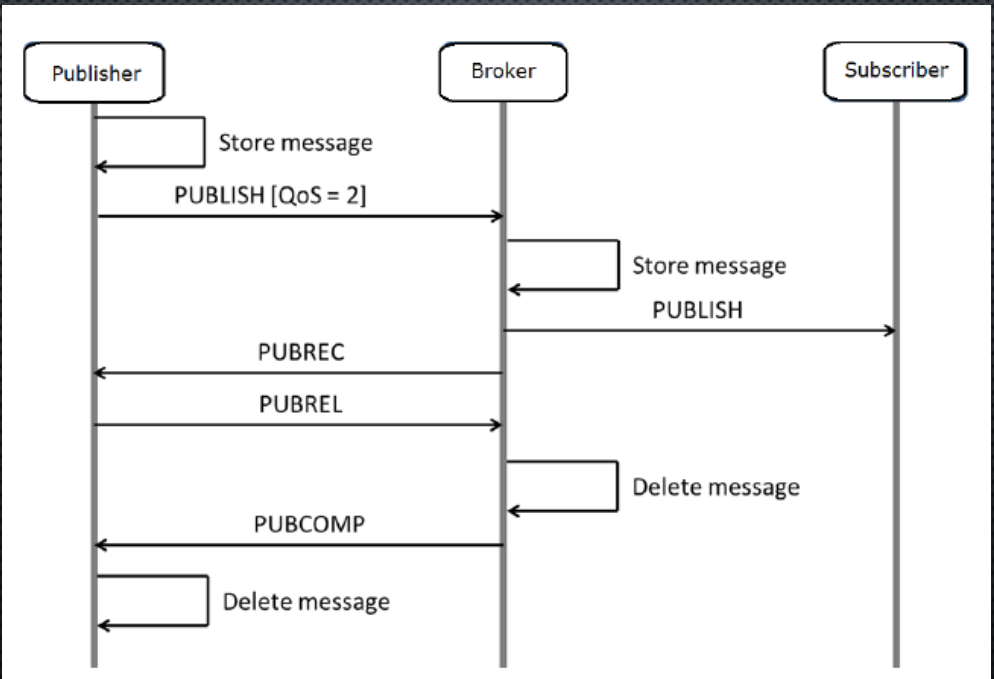
At Most Once QoS 0



At Least Once QoS 1



Exactly Once QoS 2

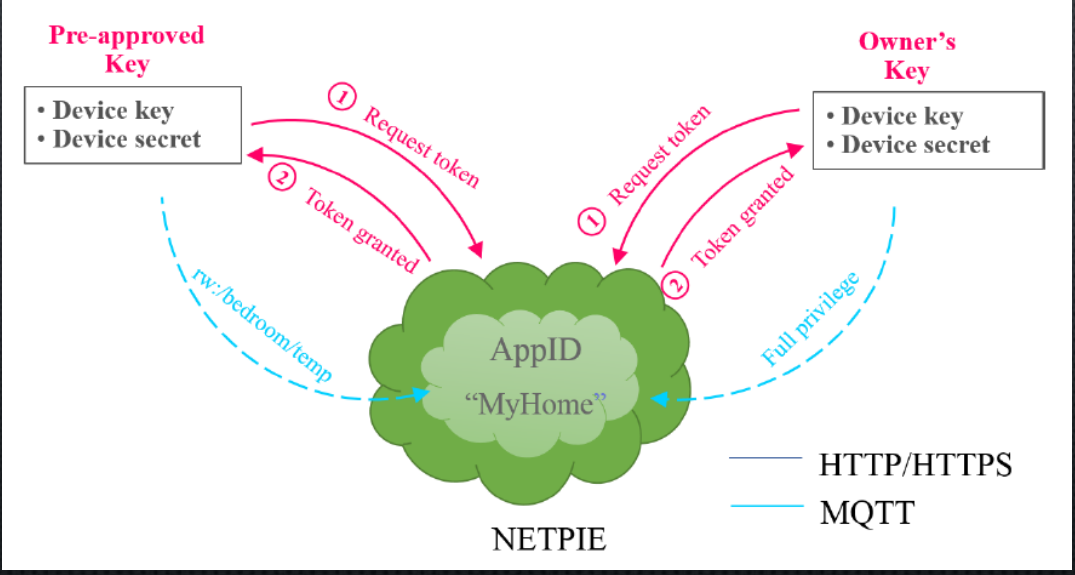


Authorization and Authentication of NETPIE

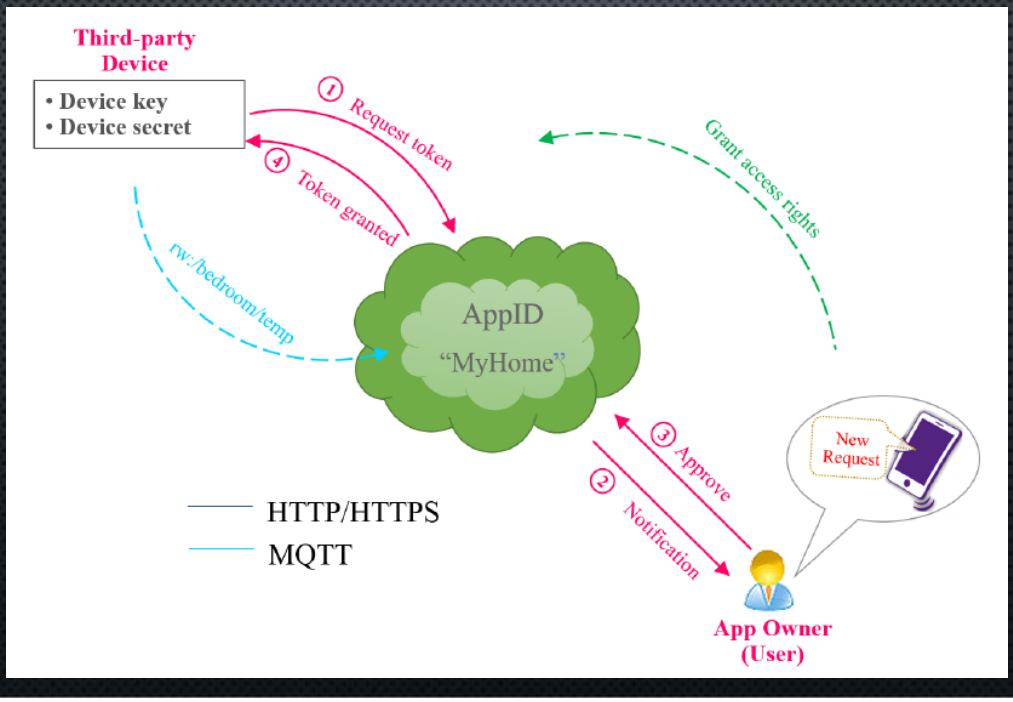
NETPIE Connection process

1. • Register NETPIE Web Portal
   • Create Application (AppID)
2. • Add devices under that Application
   • Get Key and Key Secret put in your Code of Microgear
3. • Microgear Automatic connect with NETPIE
   • Use Key and Key Secret for receive Token for connection
4. • 2 Option for user
   • Pre-approved Key
   • Third-Party Key

Pre-approved Key



Third-Party Key



Type of Key

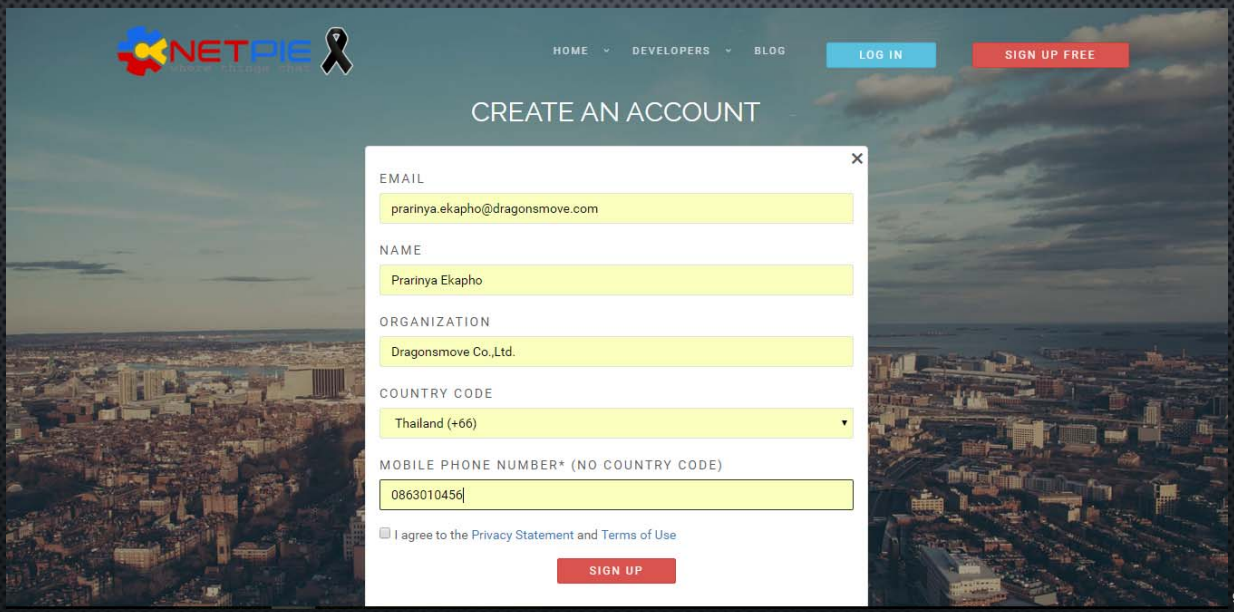
Device Key

Session Key

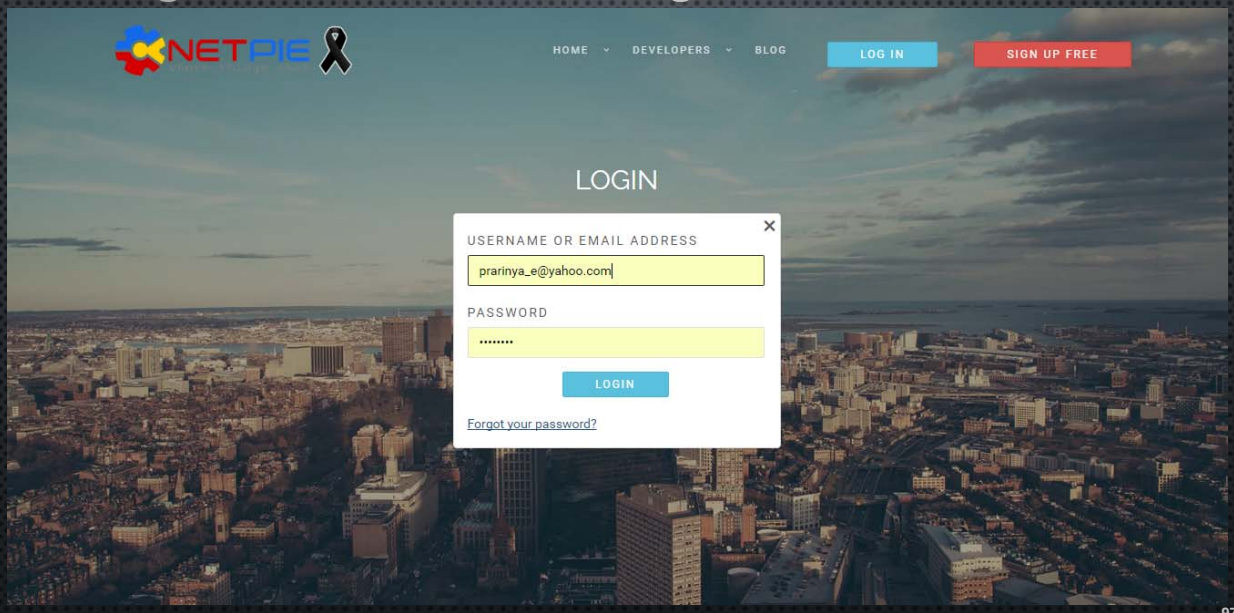
Scope for access NETPIE MQTT Broker

1	• rw:/topic • rw:/home/bedroom/temperature
2	• r:/topic • r:/home/#
3	• w:/topic • w:/home/+/temperature

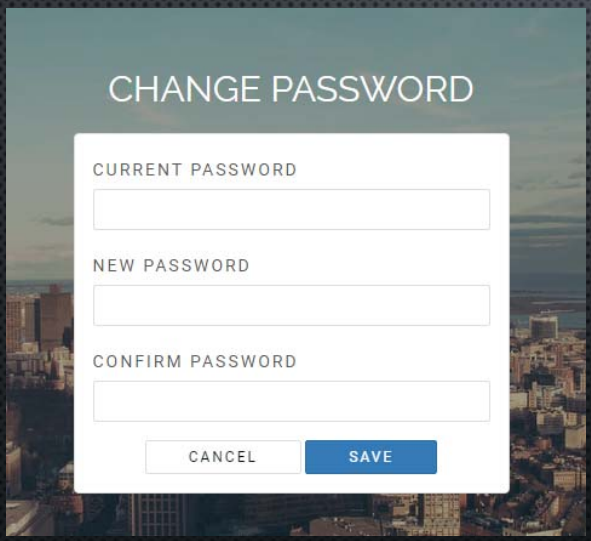
Apply Member NETPIE (https://netpie.io/sign_up)

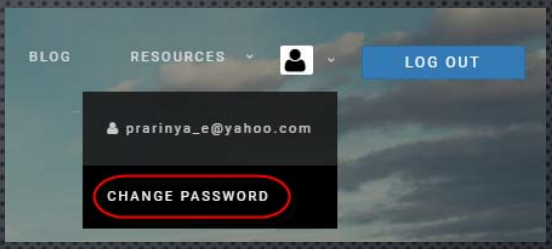


Waiting SMS for OTP First Login

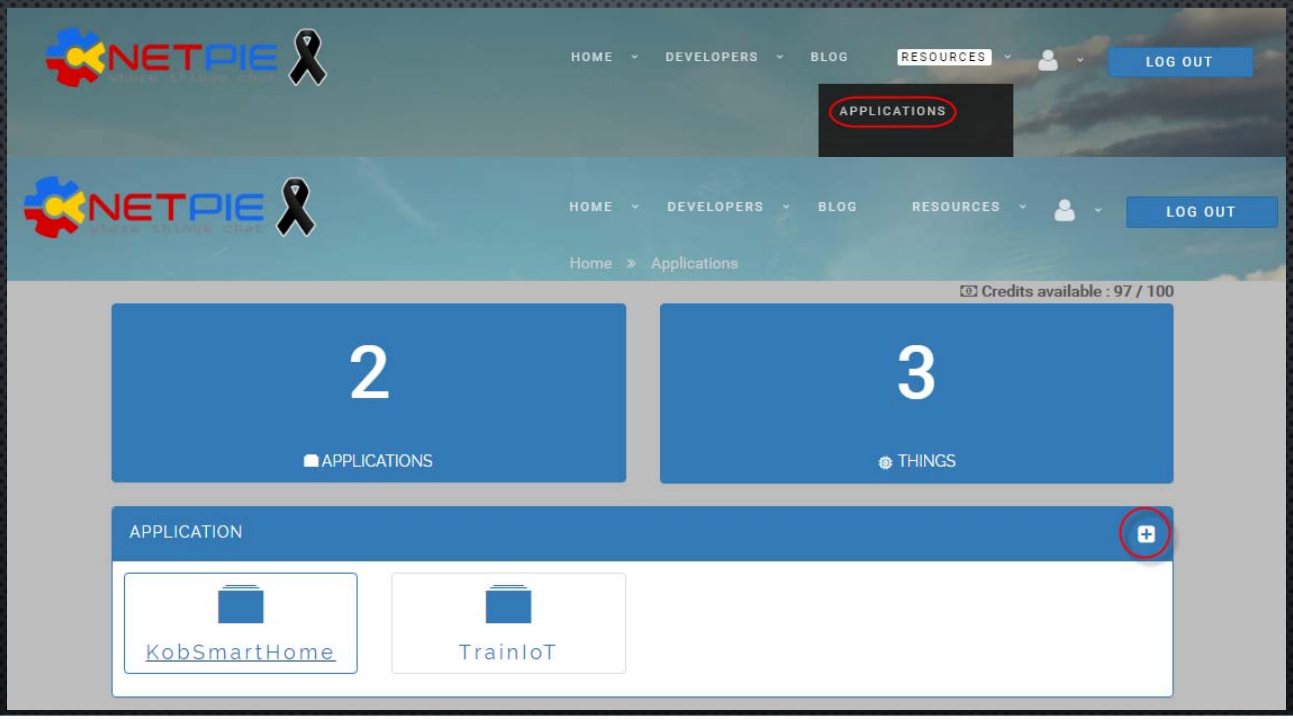


Change Password

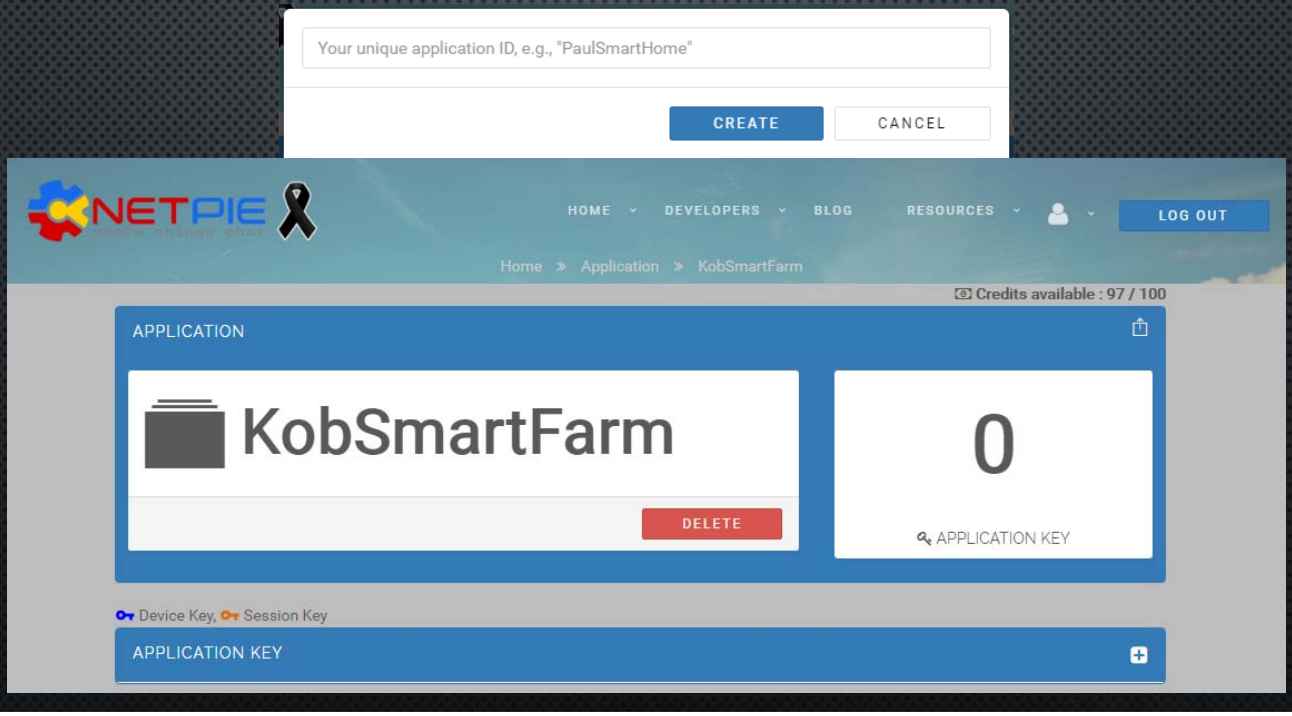




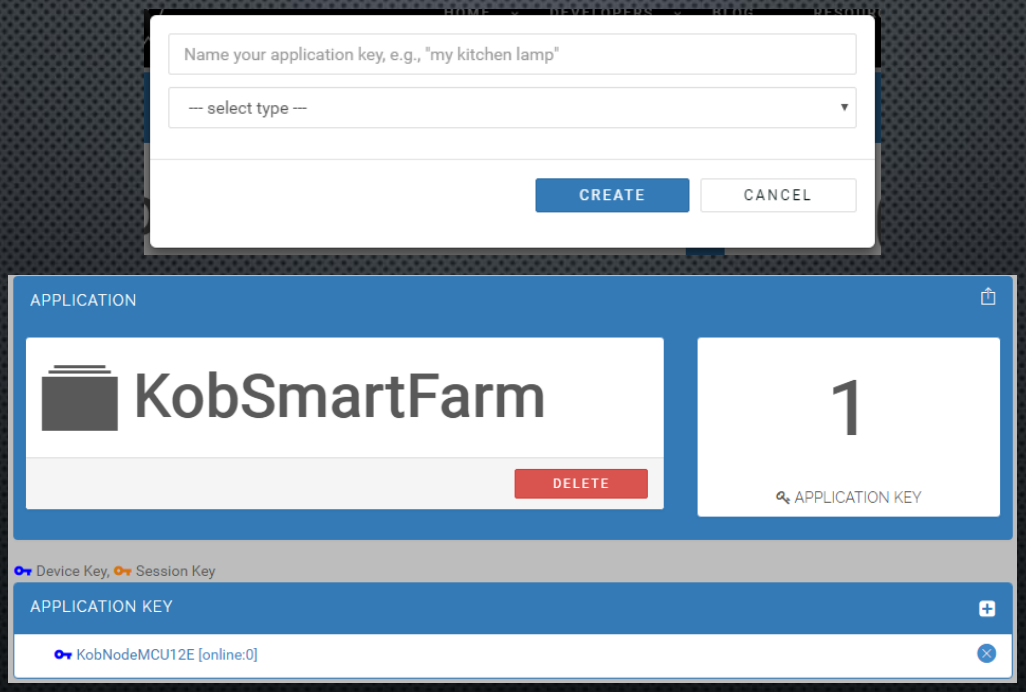
Create Application



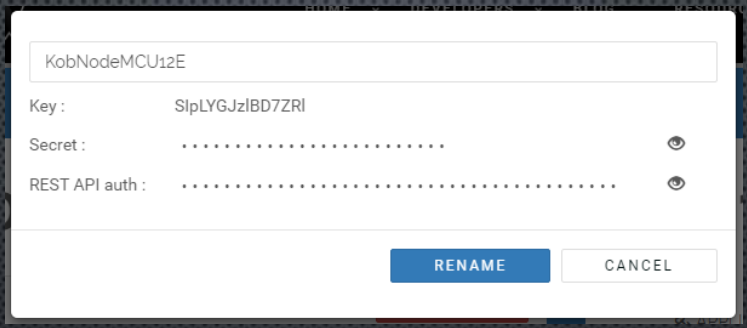
Create Application



Create Application Key



Create Application Key



Workshop#3 NodeMCU NETPIE blink

```
/*  NETPIE ESP8266 blink                          */

#include <ESP8266WiFi.h>
#include <MicroGear.h>

const char* ssid     = “SSID"; //your ssid
const char* password = “PASSWORD"; //your password

#define APPID   “YOUR_APPID" //your AppID
#define KEY     “YOUR_KEY" //your Key
#define SECRET  “YOUR_SECRET" //your Secret
#define ALIAS   “netpieblink"

WiFiClient client;

int timer = 0;
char state = 0;
MicroGear microgear(client);
```

```
/* If a new message arrives, do this */
void onMsghandler(char *topic, uint8_t* msg, unsigned int
msglen) {
  Serial.print("Incoming message --> ");
  msg[msglen] = '\0';
  Serial.println((char *)msg);
// If messaage = 1 On LED, If 0 LED off
  if(*(char *)msg == '1'){
    digitalWrite(LED_BUILTIN, LOW); // LED on
   }else{
    digitalWrite(LED_BUILTIN, HIGH); // LED off }
  }
}


/* When a microgear is connected, do this */
void onConnected(char *attribute, uint8_t* msg, unsigned
int msglen) {
  Serial.println("Connected to NETPIE...");
  /* Set the alias of this microgear ALIAS */
  microgear.setAlias(ALIAS);
}
```


Workshop#3 NodeMCU NETPIE blink

```
void setup() {
  /* Add Event listeners */
  /* Call onMsghandler() when new message arraives */
  microgear.on(MESSAGE,onMsghandler);

  /* Call onConnected() when NETPIE connection is
established */
  microgear.on(CONNECTED,onConnected);

  Serial.begin(115200);
  Serial.println("Starting...");

  pinMode(LED_BUILTIN, OUTPUT); //Set LED Built-in to
Output

  /* Initial WIFI, this is just a basic method to configure WIFI
on ESP8266.                       */
  /* You may want to use other method that is more
complicated, but provide better user experience */
  if (WiFi.begin(ssid, password)) {
    while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
    }
  }
```

```
 Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  /* Initial with KEY, SECRET and also set the ALIAS here */
  microgear.init(KEY,SECRET,ALIAS);

  /* connect to NETPIE to a specific APPID */
  microgear.connect(APPID);
}

void loop() {
  /* To check if the microgear is still connected */
  if (microgear.connected()) {
    Serial.println("connected");

    /* Call this method regularly otherwise the connection
may be lost */
    microgear.loop();
```

Workshop#3 NodeMCU NETPIE blink

```
if (timer >= 1000) {
        Serial.println("Publish...");

        //chat myself with invert stage
        if(state==0){
          microgear.chat(ALIAS,state);
          state = 1;
        }else{
          microgear.chat(ALIAS,state);
          state = 0;
        }
        timer = 0;
      }
      else timer += 100;
    }
    else {
      Serial.println("connection lost, reconnect...");
      if (timer >= 5000) {
        microgear.connect(APPID);
        timer = 0;
      }
      else timer += 100;
    }
    delay(100);
}
```


Workshop#4
Another way to control IoT devices

- Control IoT via NETPIE
- Control IoT via Blynk

Workshop#4 IoT Switch use NETPIE

1. • NodeMCU
   • Use file NETPIE_IoT_Switch.ino (Device Key)
2. • HTML5
   • Use file switch.html (Session Key)
3. • AppID, KEY, SECRET
   • Save and upload file to NodeMCU
4. • Open Serial Monitor
   • Open file switch.html on Browser
5. • Test press switch on browser

Workshop#4 NETPIE_IoT_Switch.ino

```
/*  NETPIE ESP8266 blink                          */
#include <ESP8266WiFi.h>
#include <MicroGear.h>

const char* ssid     = "......"; //your ssid
const char* password = "....."; //your password

#define APPID   "....." //your AppID
#define KEY     "....." //your Device Key
#define SECRET  "....." //your Device Secret Key
#define ALIAS   "....." //your Alias

WiFiClient client;

int timer = 0;
MicroGear microgear(client);

/* If a new message arrives, do this */
void onMsghandler(char *topic, uint8_t* msg, unsigned int
msglen) {
  Serial.print("Incoming message --> ");
  msg[msglen] = '\0';
  Serial.println((char *)msg);
// If messaage = 1 On LED, If 0 LED off
  if(*(char *)msg == '1'){
    digitalWrite(LED_BUILTIN, LOW); // LED on
    microgear.chat("switch","1");
   }else{
    digitalWrite(LED_BUILTIN, HIGH); // LED off
    microgear.chat("switch","0");
  }
}

/* When a microgear is connected, do this */
void onConnected(char *attribute, uint8_t* msg, unsigned
int msglen) {
  Serial.println("Connected to NETPIE...");
  /* Set the alias of this microgear ALIAS */
  microgear.setAlias(ALIAS);
}


void setup() {
  /* Add Event listeners */

  /* Call onMsghandler() when new message arraives */
  microgear.on(MESSAGE,onMsghandler);
```

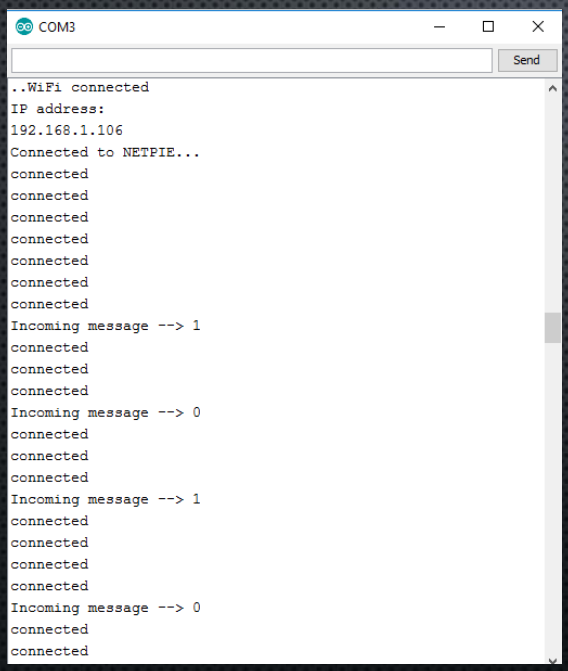
Workshop#4 NETPIE_IoT_Switch.ino

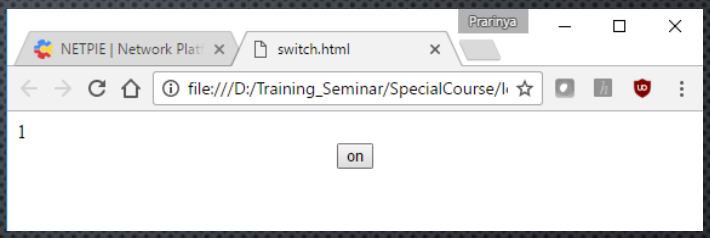
```
 /* Call onConnected() when NETPIE connection is established */
  microgear.on(CONNECTED,onConnected);

  Serial.begin(115200);
  Serial.println("Starting...");

  pinMode(LED_BUILTIN, OUTPUT); //Set LED Built-in to Output

  /* Initial WIFI, this is just a basic method to configure WIFI on ESP8266.                       */
  /* You may want to use other method that is more complicated, but provide better user experience */
  if (WiFi.begin(ssid, password)) {
    while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
    }
  }

  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

 /* Initial with KEY, SECRET and also set the ALIAS here */
  microgear.init(KEY,SECRET,ALIAS);
  /* connect to NETPIE to a specific APPID */
  microgear.connect(APPID);
}

void loop() {
  /* To check if the microgear is still connected */
  if (microgear.connected()) {
    Serial.println("connected");

    /* Call this method regularly otherwise the connection may be lost */
    microgear.loop();
    timer = 0;
    } else {
     Serial.println("connection lost, reconnect...");
     if (timer >= 5000) {
        microgear.connect(APPID);
        timer = 0;
     }
     else timer += 100;
    }
  delay(100);
}
```


Workshop#4 switch.html

```
<script src="https://cdn.netpie.io/microgear.js"></script>
<script>
      const APPID = "……."; //your AppID
      const KEY = "……."; //your Session Key
      const SECRET = "…….";//your Session Secret Key
      const ALIAS = "…….";//your Alias
var microgear = Microgear.create({
      key: KEY,
      secret: SECRET,
      alias : ALIAS
      });
function toggle() {
      if(document.getElementById("button").innerText=="off"){
            microgear.chat('KobNodeMCU12E','1');
      } else{   // Alias of your device
            microgear.chat('KobNodeMCU12E','0');
      }
}
 microgear.on('message',function(topic,msg) {
      document.getElementById("data").innerHTML = msg;
      if(msg=="1"){
      document.getElementById("button").innerText="on";
            }else if(msg=="0"){
      document.getElementById("button").innerText="off";
            }
      });
      microgear.on('connected', function() {
            microgear.setAlias(ALIAS);
            document.getElementById("data").innerHTML = "Now I am connected with netpie...";
      });
microgear.connect(APPID);
</script>
<div id="data">____</div>
<center>
<button onclick="toggle()" id="button">off</button>
</center>
```

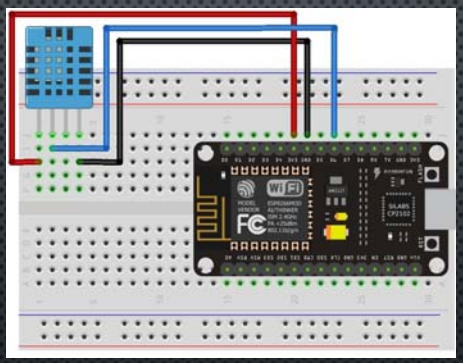
Workshop#4 IoT Switch





Workshop#4 Send data to NETPIE Freeboard

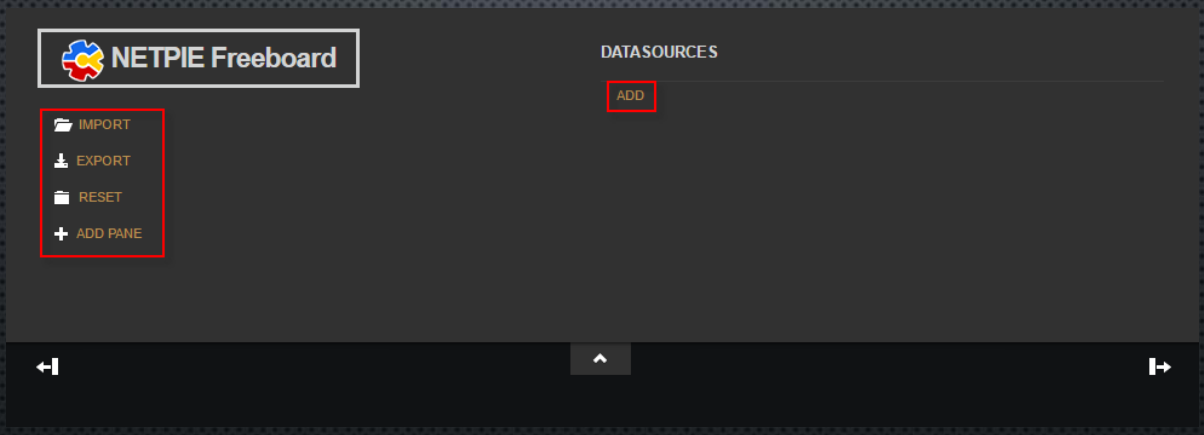
Hardware



Software

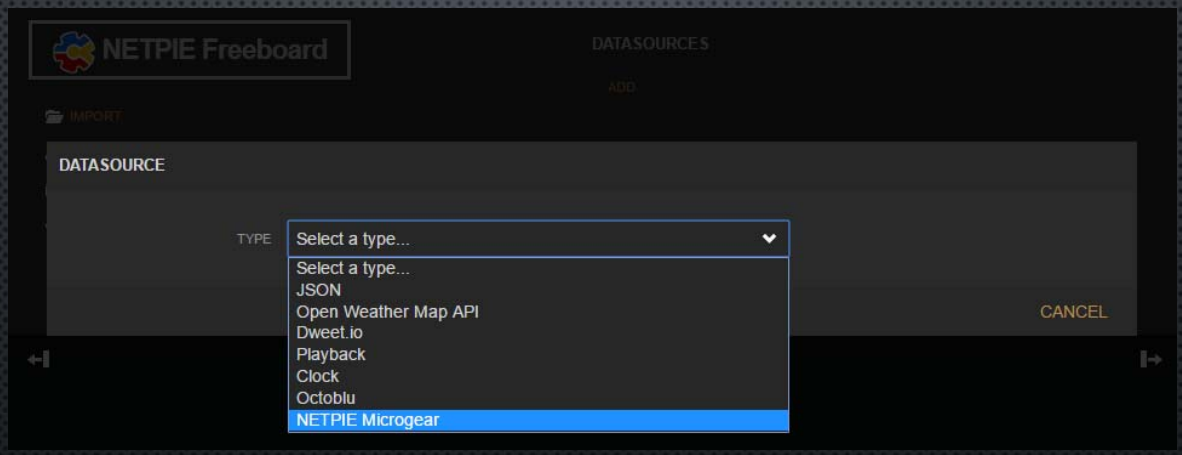
Upload program NETPIE_Freeboard.ino to board

Workshop#4 NETPIE Freeboard

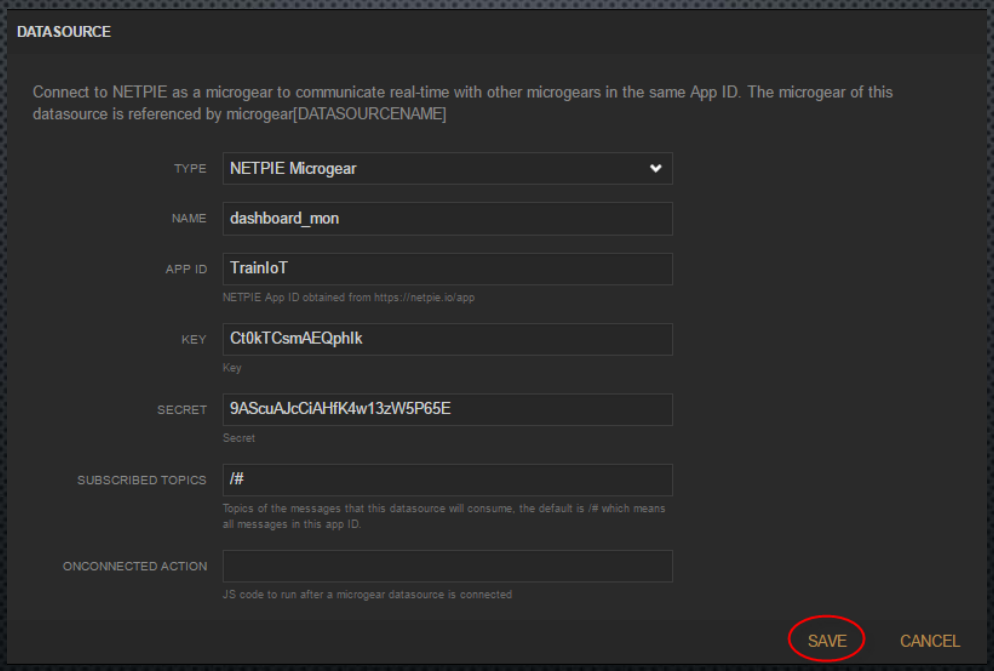


Before to next step Upload file NETPIE_Freeboard.ino to board

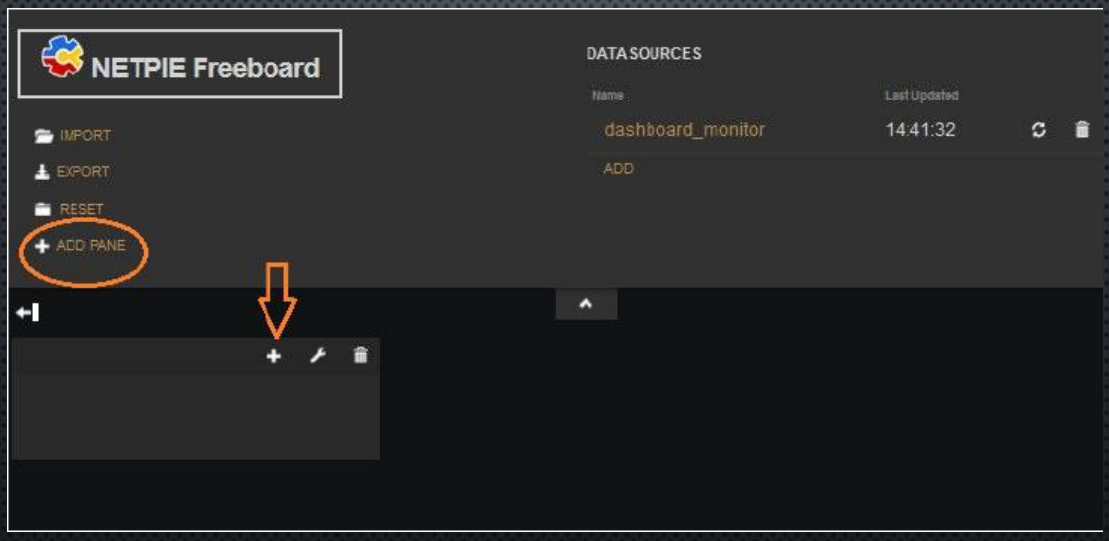
Workshop#4 NETPIE Freeboard



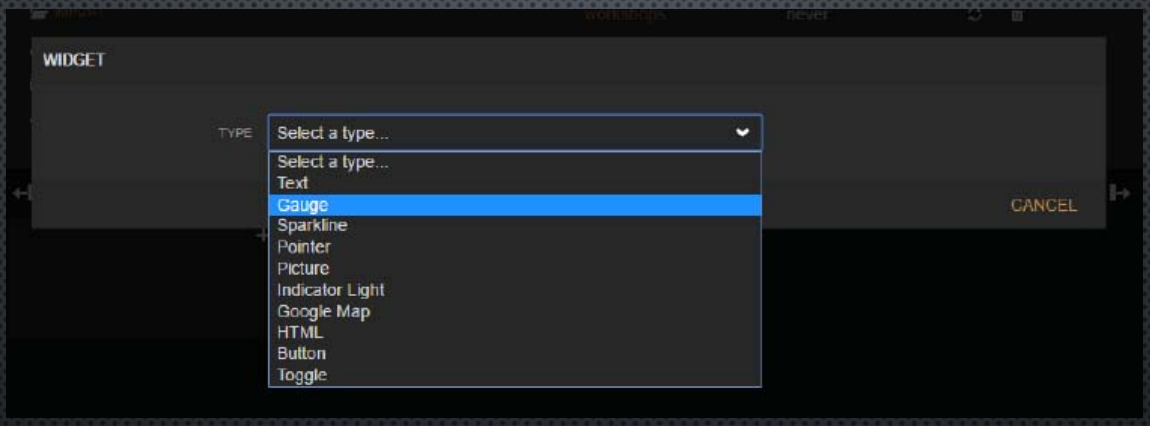
Workshop#4 NETPIE Freeboard



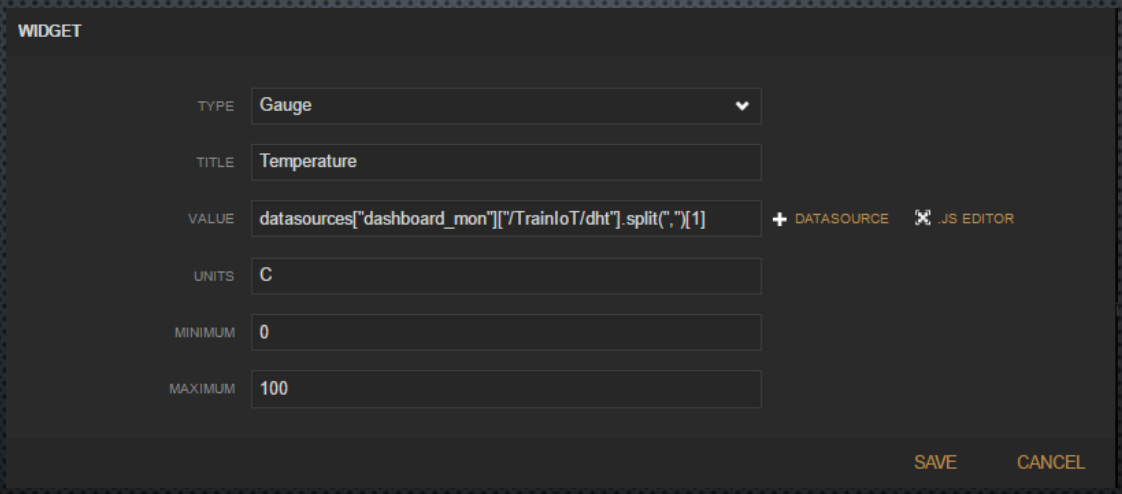
Workshop#4 NETPIE Freeboard



Workshop#4 NETPIE Freeboard



Workshop#4 NETPIE Freeboard



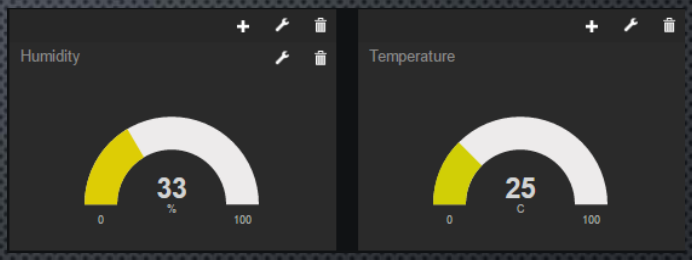
Workshop#4 NETPIE Freeboard

Widget 1:

- TITLE : Humidity
- VALUE : datasources["YourDatasourceName"]["/YourAppID/dht"].split(",")[0]
- UNIT : %
- MINIMUM : 0
- MAXIMUM : 100

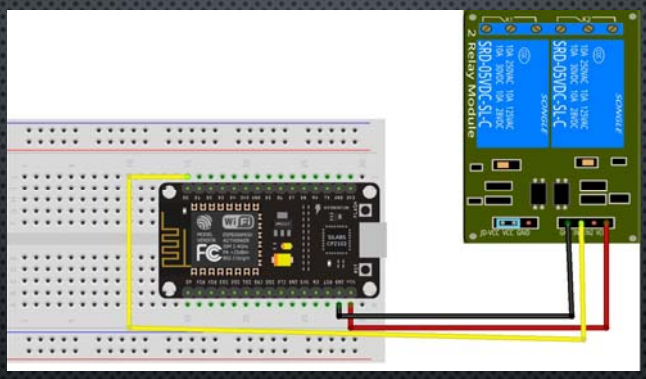
Widget 2:

- TITLE : Temperature
- VALUE : datasources["YourDatasourceName"]["/YourAppID/dht"].split(",")[1]
- UNIT : C
- MINIMUM : 0
- MAXIMUM : 50



Workshop#4 Control devices with NETPIE Freeboard

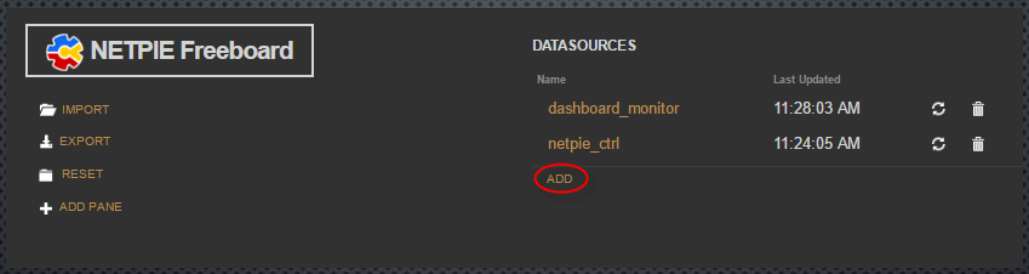
Hardware



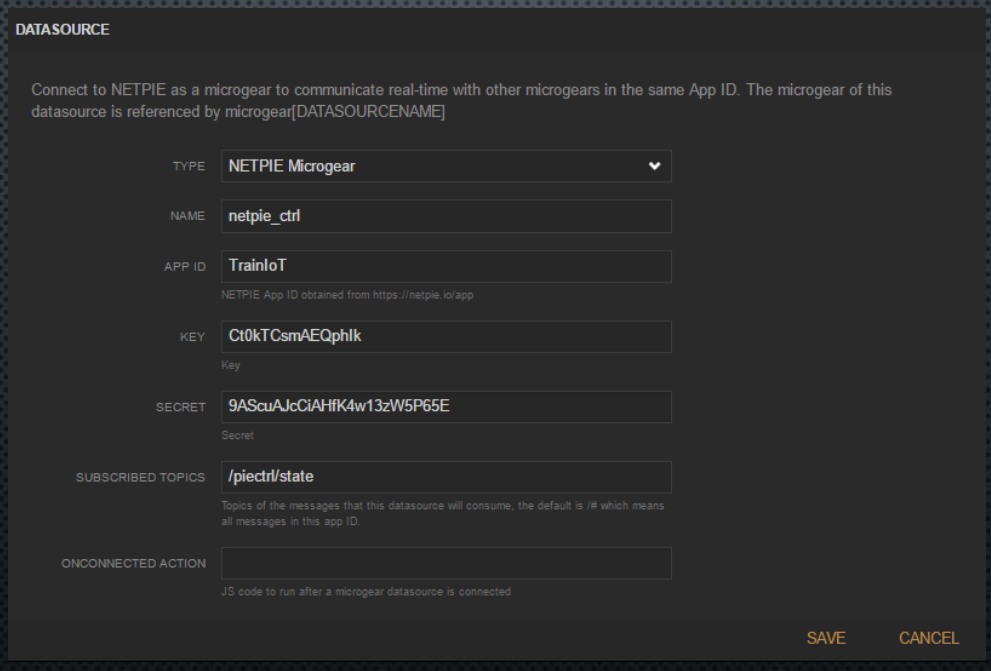
Software

Upload program NETPIE_Ctrl _Freeboard.ino to board

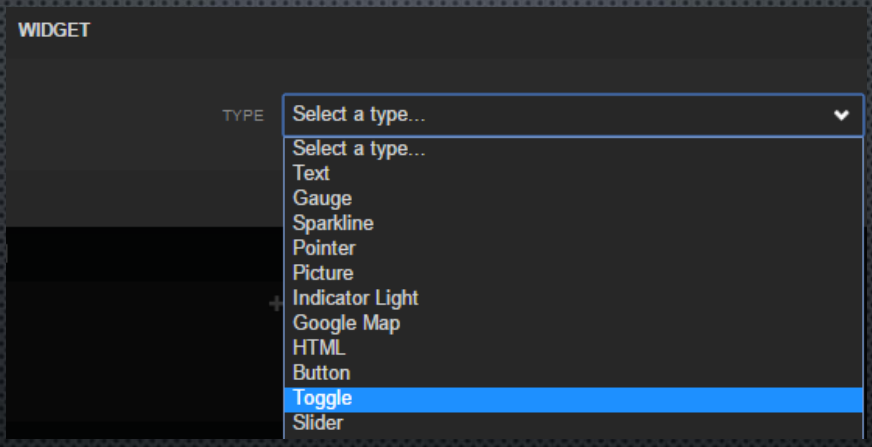
Workshop#4 Control devices with NETPIE Freeboard



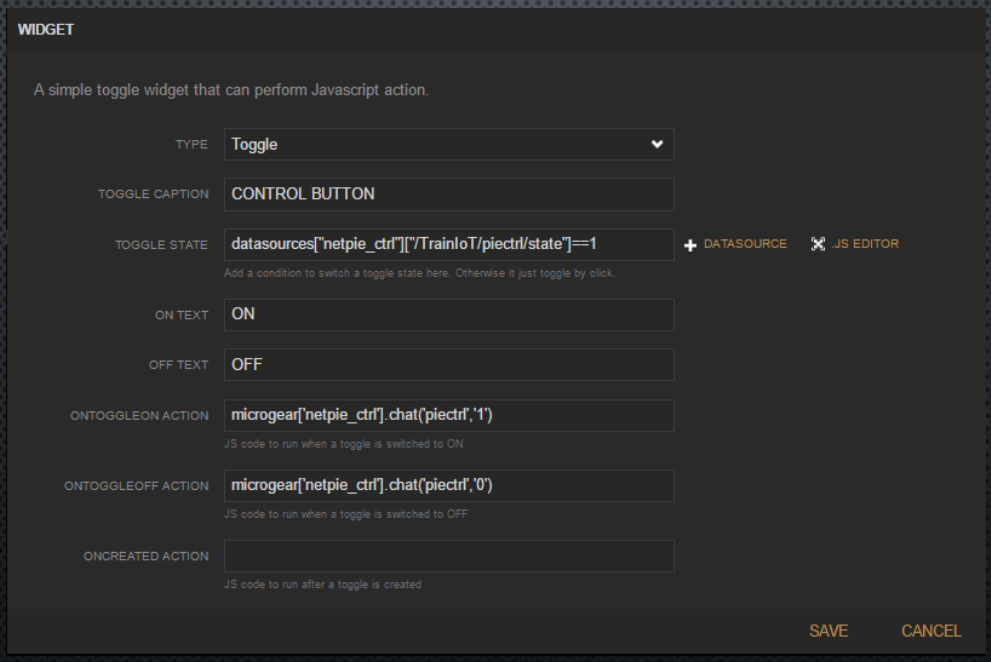
Workshop#4 Control devices with NETPIE Freeboard



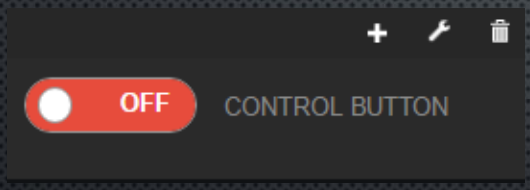
Workshop#4 Control devices with NETPIE Freeboard

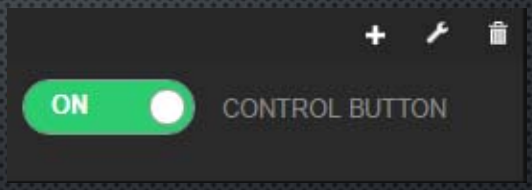


Workshop#4 Control devices with NETPIE Freeboard

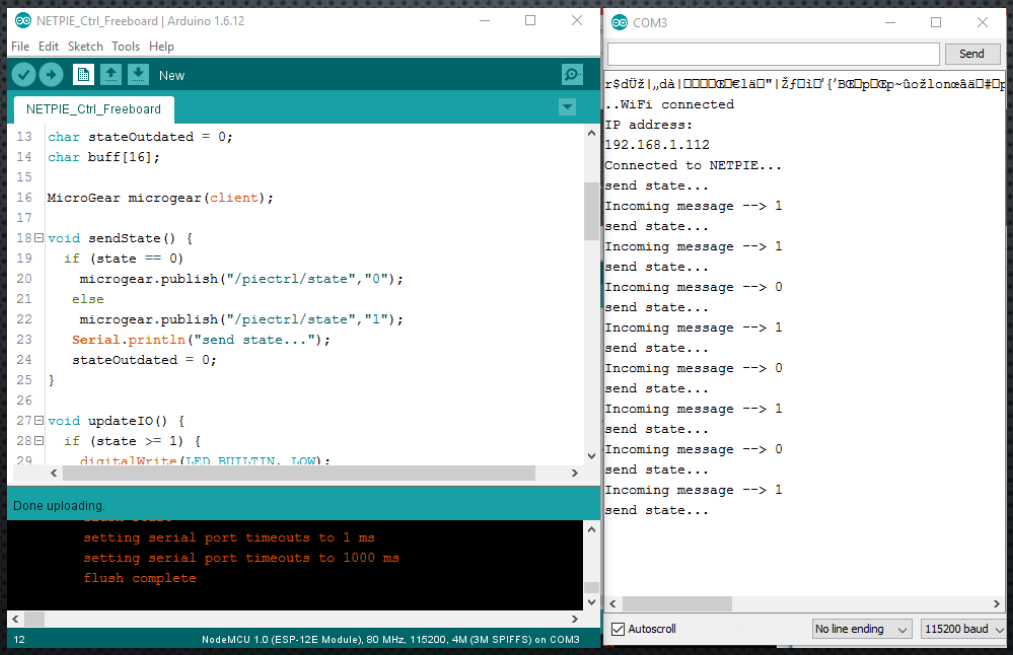


Workshop#4 Control devices with NETPIE Freeboard

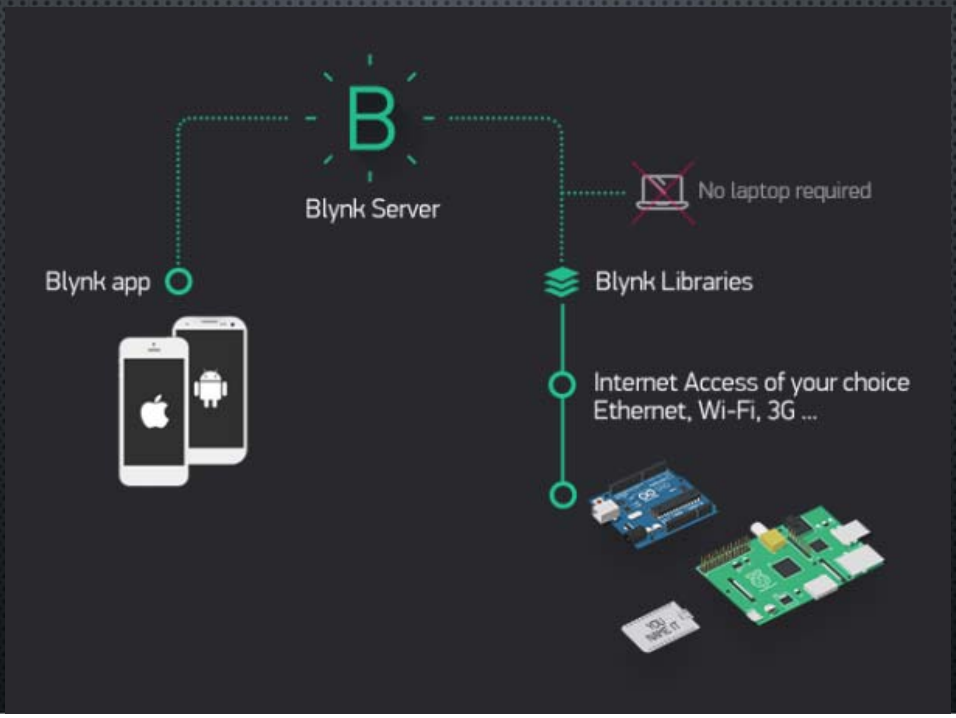




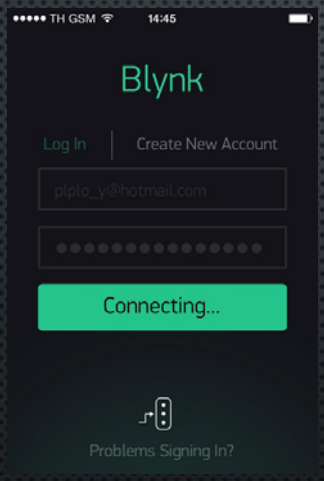
Workshop#4 Control devices with NETPIE Freeboard

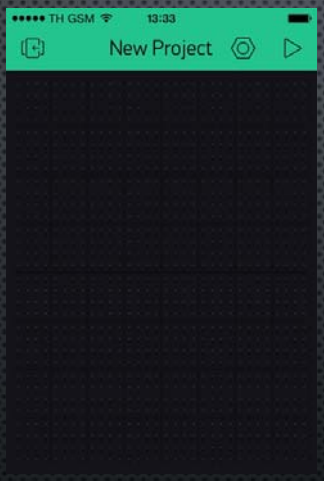


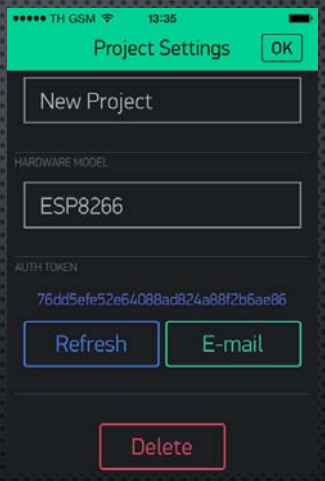
Workshop#4 Control IoT via Blynk

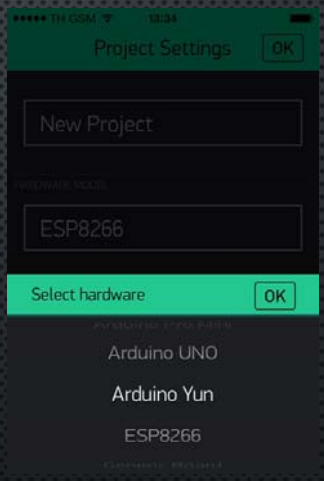


Workshop#4 Control IoT via Blynk









Workshop#4 Control IoT via Blynk

Auth Token for IoT Switch project and device New Device

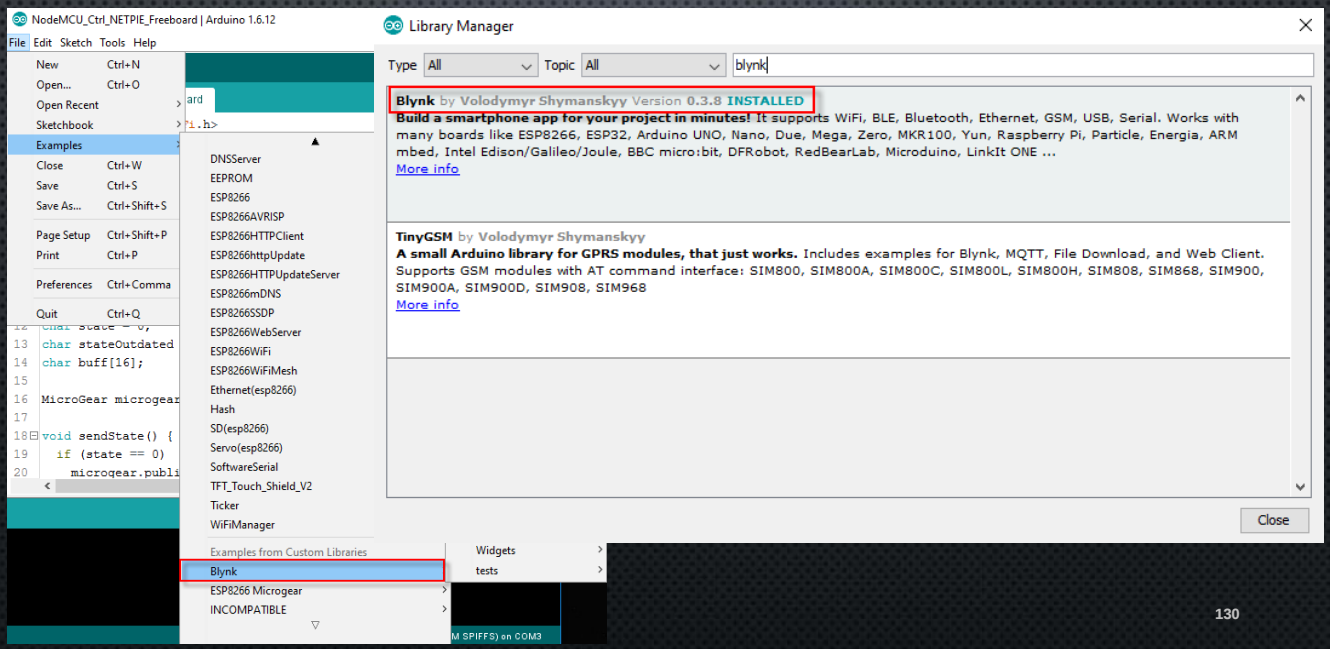
dispatcher@blynk.io

To prarinya_e@yahoo.com

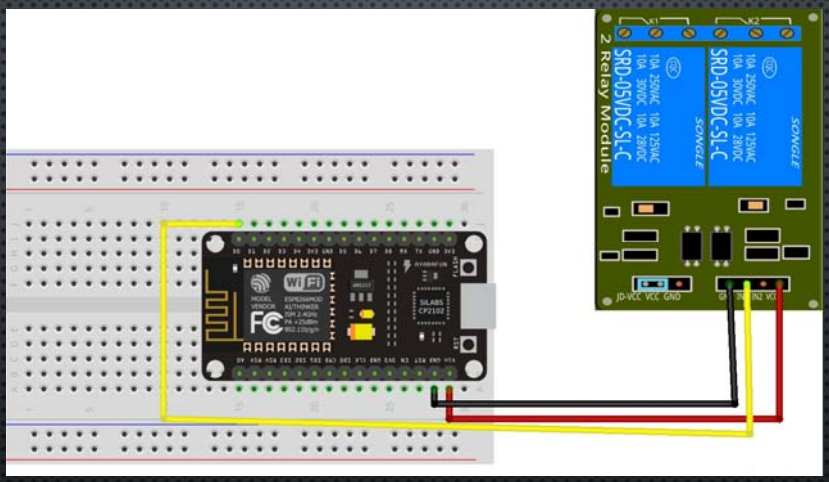
Auth Token for IoT Switch project "7c66a2b373a148b8a9b9470a0da38f87"

Happy Blynking!
-
Getting Started Guide -> http://www.blynk.cc/getting-started
Documentation -> http://docs.blynk.cc/
Latest Blynk library -> https://github.com/blynkkk/blynk-library/releases/download/v0.4.1/Blynk_Release_v0.4.1.zip
Latest Blynk server -> https://github.com/blynkkk/blynk-server/releases/download/v0.20.1/server-0.20.1.jar
-
http://www.blynk.cc
twitter.com/blynk_app
www.facebook.com/blynkapp

Workshop#4 Control IoT via Blynk

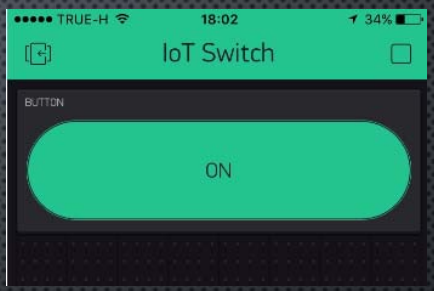


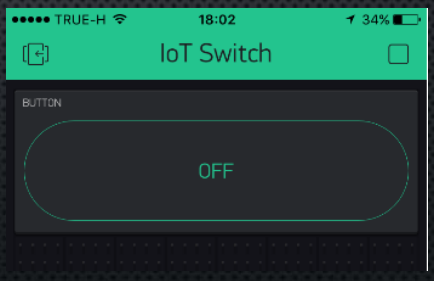
Workshop#4 Control IoT via Blynk



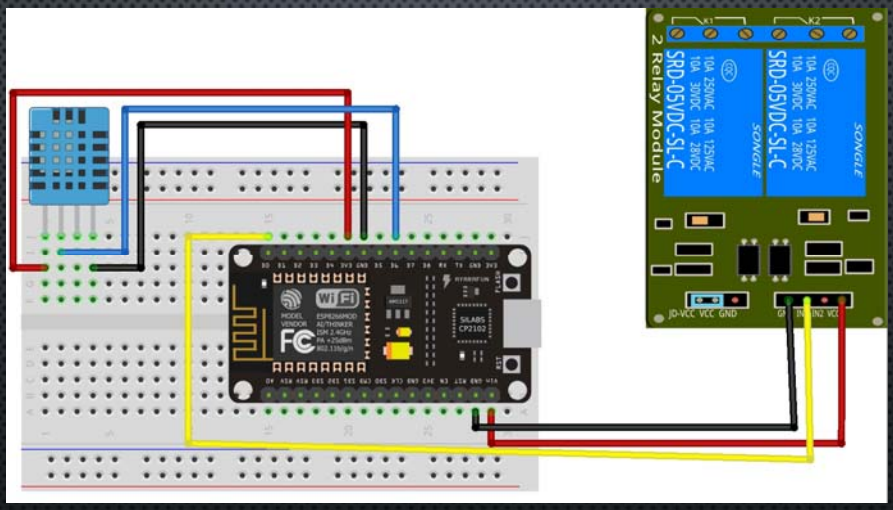
Workshop#4 Control IoT via Blynk

```
#define BLYNK_PRINT Serial    // Comment this out to
disable prints and save space
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = “your_authentication_Key";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = “......"; // your SSID
char pass[] = “....."; //your password

void setup()
{
  Serial.begin(115200);
  Blynk.begin(auth, ssid, pass);
}

void loop()
{
  Blynk.run();
}
```





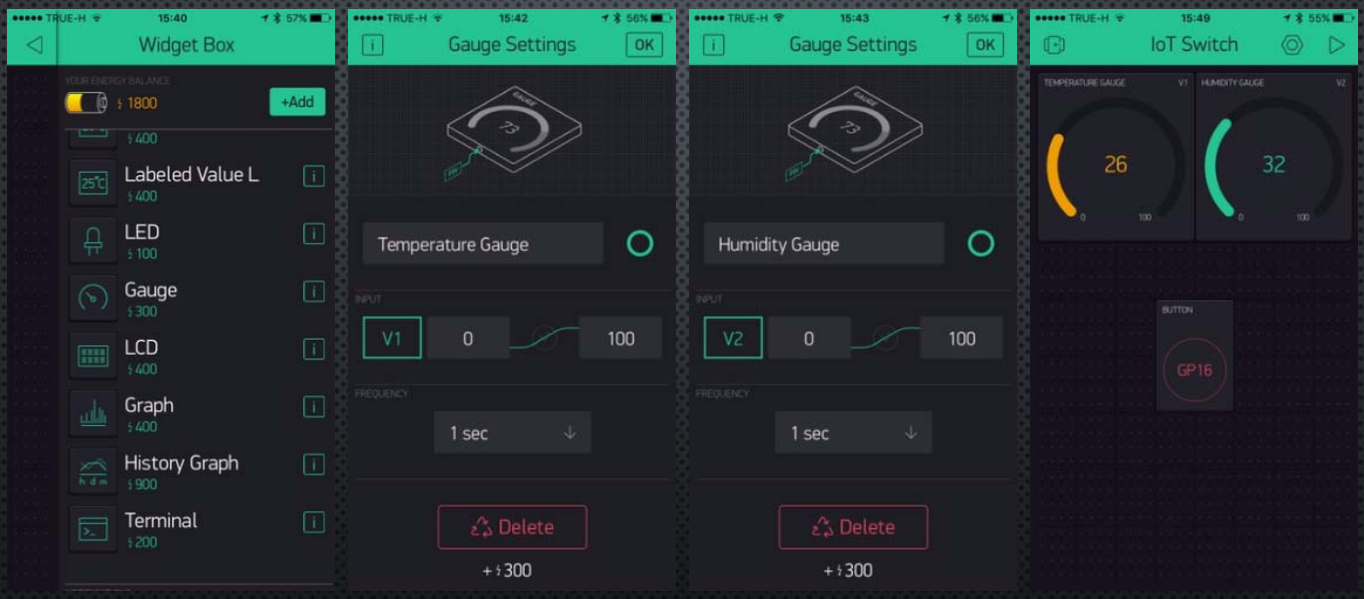
Workshop#4 Control IoT via Blynk



Workshop#4 Control IoT via Blynk

```
#define BLYNK_PRINT Serial    // Comment this out to
disable prints and save space
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <SimpleTimer.h>

#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN          12    //GPIO12 Or D6 pin
#define DHTTYPE         DHT11

DHT_Unified dht(DHTPIN, DHTTYPE);

char auth[] = "your_Authentication_Key";

SimpleTimer timer;
void setup()
{
  Serial.begin(115200);
  Blynk.begin(auth, "your_SSID", "your_Password");

  dht.begin();

  // Setup a function to be called every second
  timer.setInterval(5000L, sendUptime);
}

void sendUptime()
{
  Blynk.virtualWrite(V5, millis() / 1000);
  sensors_event_t event;
  dht.temperature().getEvent(&event);
  if (!isnan(event.temperature)) {
    Blynk.virtualWrite(V1, event.temperature);
  }
  dht.humidity().getEvent(&event);
  if (!isnan(event.relative_humidity)) {
    Blynk.virtualWrite(V2, event.relative_humidity);
  }
}

void loop()
{
  Blynk.run();
  timer.run();
}
```

Workshop#4 Control IoT via Blynk



Question....

Further resources

- https://www.arduino.cc/
- https://thingspeak.com/
- https://netpie.io/
- https://espressif.com/en/products/hardware/esp8266ex/overview
- http://www.esp8266.com/
- http://www.blynk.cc/
- https://www.rs-online.com/designspark/rs-toolbox

Course feedback and instructor evaluation

Final Question....

Email: prarinya.e@gmail.com