IoT Workshop

Imanol Gómez
imanolgomez.net
7th - 8th January 2019

1. Introduction

Why are you here?

What will you learn?

1. What is IoT?
2. Arduino IDE
3. ESP8266 (WiFi Module)
4. Blynk Platform
5. Sensing the world
6. Hackathon
7. Future Work

Repository

https://github.com/ImanolGo/IoTWorkshop2019

What is IoT?

Neither the ‘Thing’ nor the ‘Internet’ is new

THE RISE OF IOT

PROCEESSING SPEED

LIGHT-WEIGHT OS

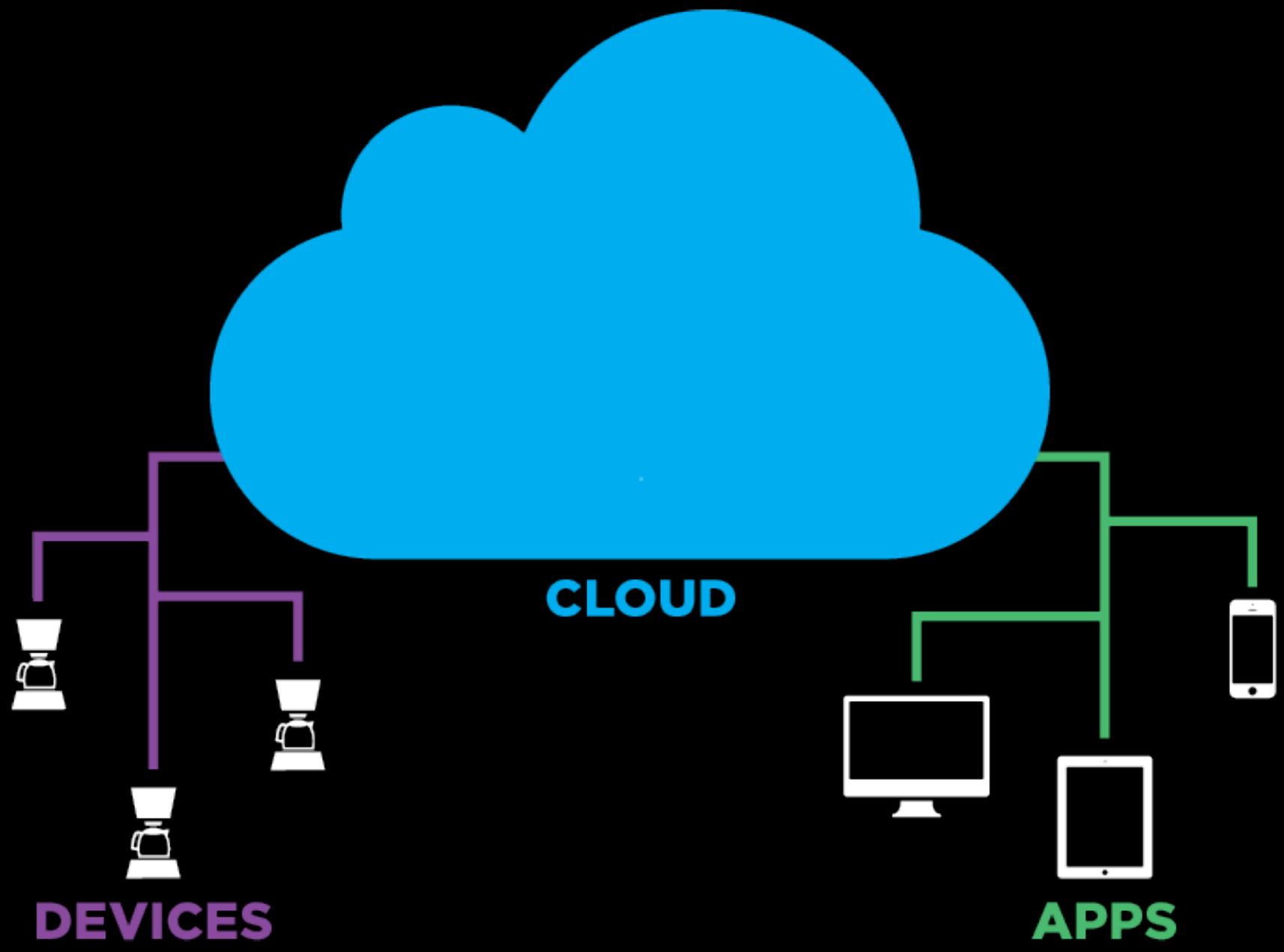
SMALL-FOOTPRINT PROTOCOLS

RESEARCH

CLOUD COMPUTING

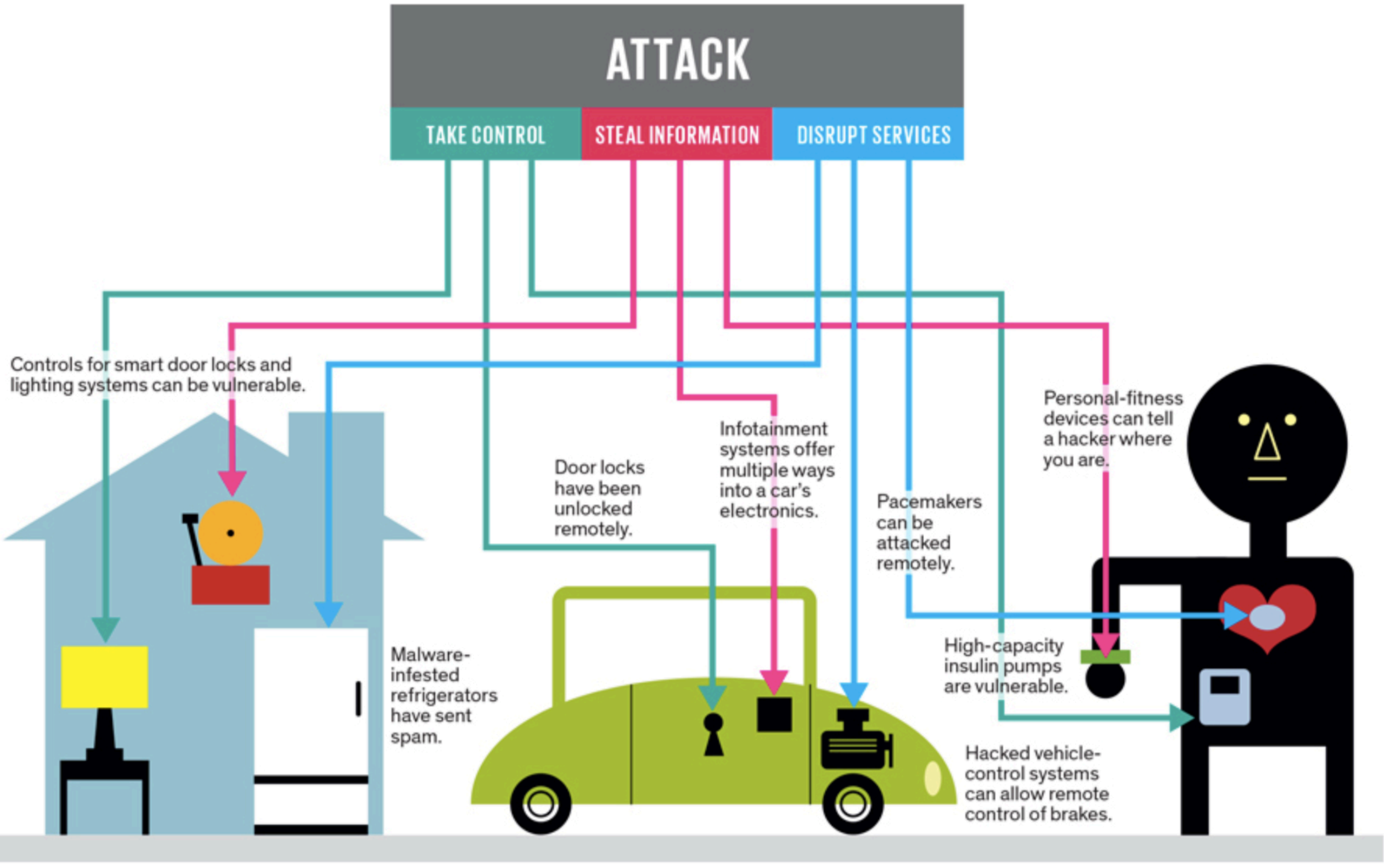
WIRELESS COMMUNICATION

How does it work?



The Cloud

Security


ATTACK
TAKE CONTROL
STEAL INFORMATION
DISRUPT SERVICES
Controls for smart door locks and lighting systems can be vulnerable.
Door locks have been unlocked remotely.
Infotainment systems offer multiple ways into a car's electronics.
Pacemakers can be attacked remotely.
Personal-fitness devices can tell a hacker where you are.
Malware-infested refrigerators have sent spam.
High-capacity insulin pumps are vulnerable.
Hacked vehicle-control systems can allow remote control of brakes.

Examples

Products

SMART THERMOSTATS

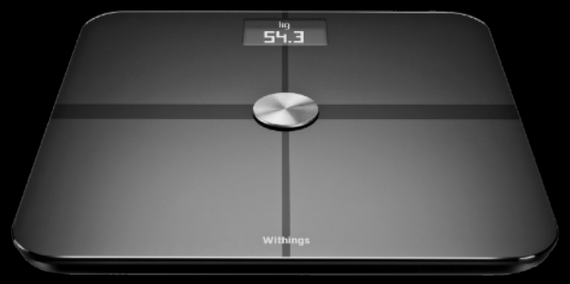
nest

ACTIVITY TRACKERS

NIKE

SMART LIGHTS

PHILIPS



CONNECTED SCALE

Withings

Applications


LOGISTICS
REAL TIME


MANUFACTURING
M2M


CITIES
INDUSTRY


HEALTH
BODY


ENVIRONMENT
PREVENTION


BUILDING
INFRASTRUCTURE

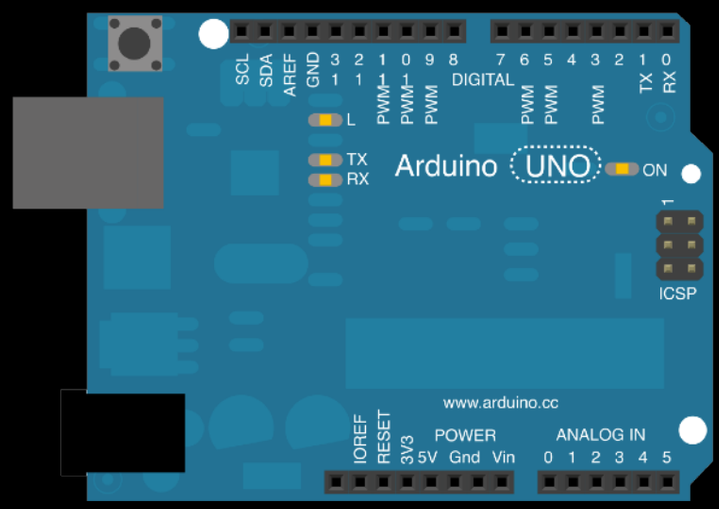

AGRICULTURE
CONTROL


SECURITY
DETECTION

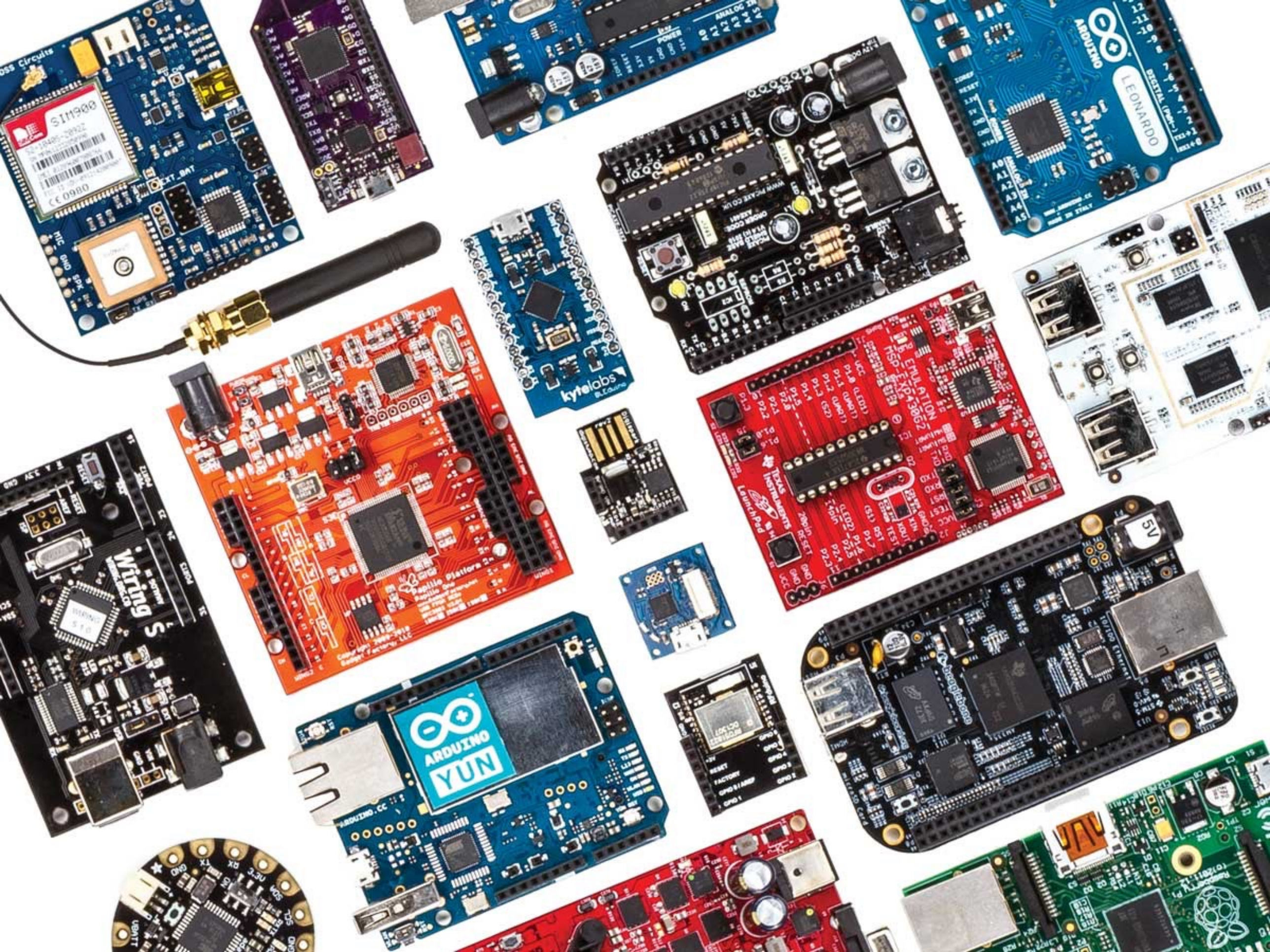
2. Arduino

ARDUINO

Software



Hardware


SIM900
ARDUINO
LEONARDO
ARDUINO
YUN
kytelabs
Wiring S
LaunchPad
5V

Arduino IDE

- Inexpensive
- Cross-platform
- Simple, clear programming environment
- Open source and extensible software
- Open source and extensible hardware

Installation

Install Arduino IDE

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

Download the Arduino IDE

ARDUINO 1.8.8

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board.
Refer to the Getting Started page for Installation instructions.

Windows Installer, for Windows XP and up
Windows ZIP file for non admin install

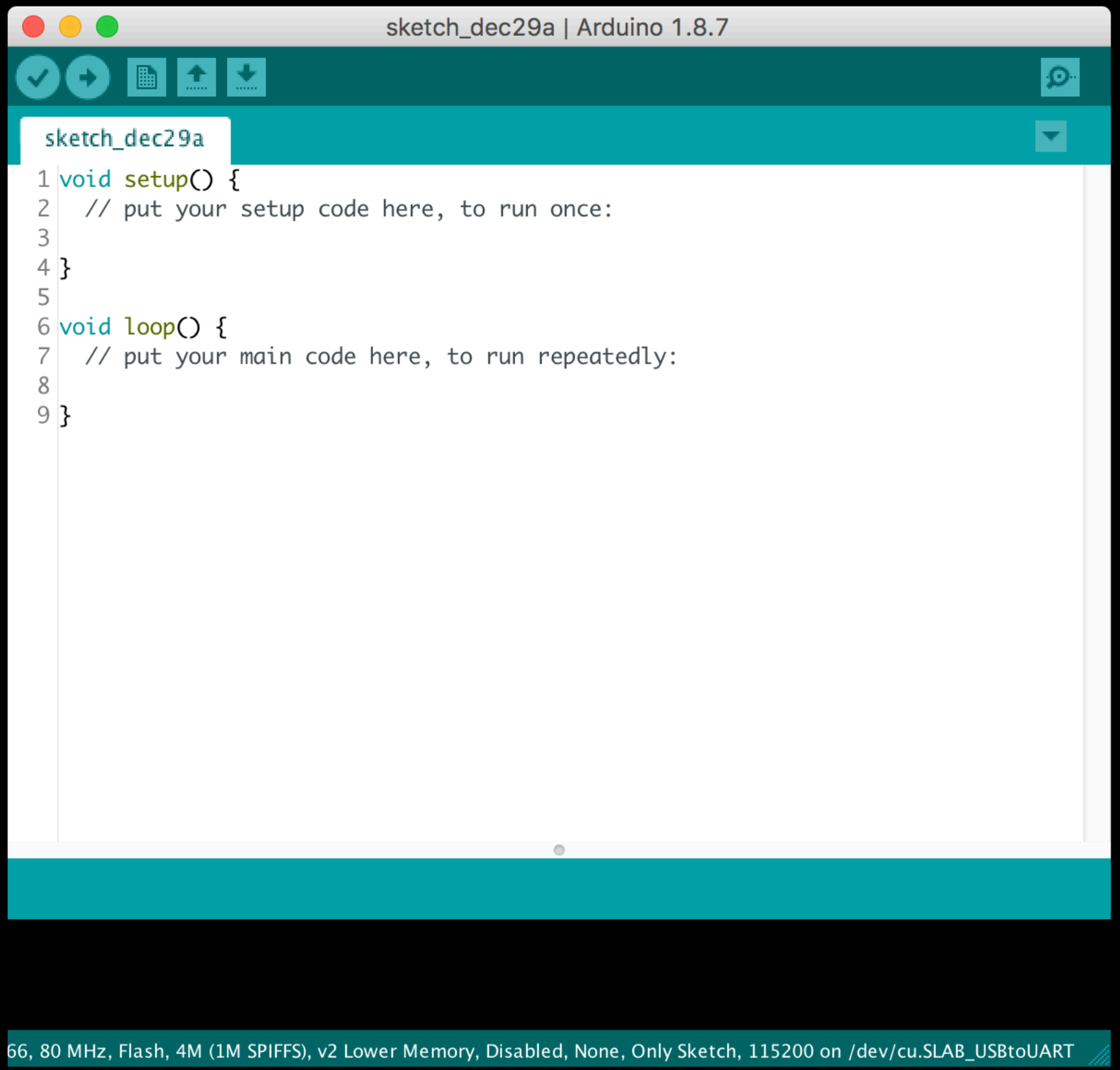
Windows app Requires Win 8.1 or 10
Get

Mac OS X 10.8 Mountain Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

Release Notes
Source Code
Checksums (sha512)

Open Arduino IDE



The Semicolon;

A semicolon needs to follow every statement written in the Arduino programming language.

```
int LEDpin = 9;
```

The Double Backlash For Single Line Comments //

- Comments are what you use to annotate code.
- Comments are meant to inform you and anyone else
- Comments will be ignored by the compiler

```
//This is the pin on the Arduino that the LED is plugged into
int LEDpin = 9
```

Curly Braces { }

- Curly braces are used to enclose further instructions carried out by a function
- There is always an opening curly bracket and a closing curly bracket

```
void loop() {   //this curly brace opens
//way cool program here
}    //this curly brace closes
```

Functions ( )

```
pinMode(13, OUTPUT);
//Sets the mode of an Arduino pin
```

```
millis( );
//Retrieves the length of time in milliseconds that the Arduino has been running
```

Void Setup( )

1. setup() only runs once.
2. setup() needs to be the first function in your Arduino sketch.
3. setup() must have opening and closing curly braces.

```
void setup( ) {   //the code between the curly braces is only run once for setup()
```

Void Loop( )

- loop() is repeated over and over again
- loop() function is where the body of your program will reside.

```
void loop( ) { //whatever code you put here is executed over and over
```

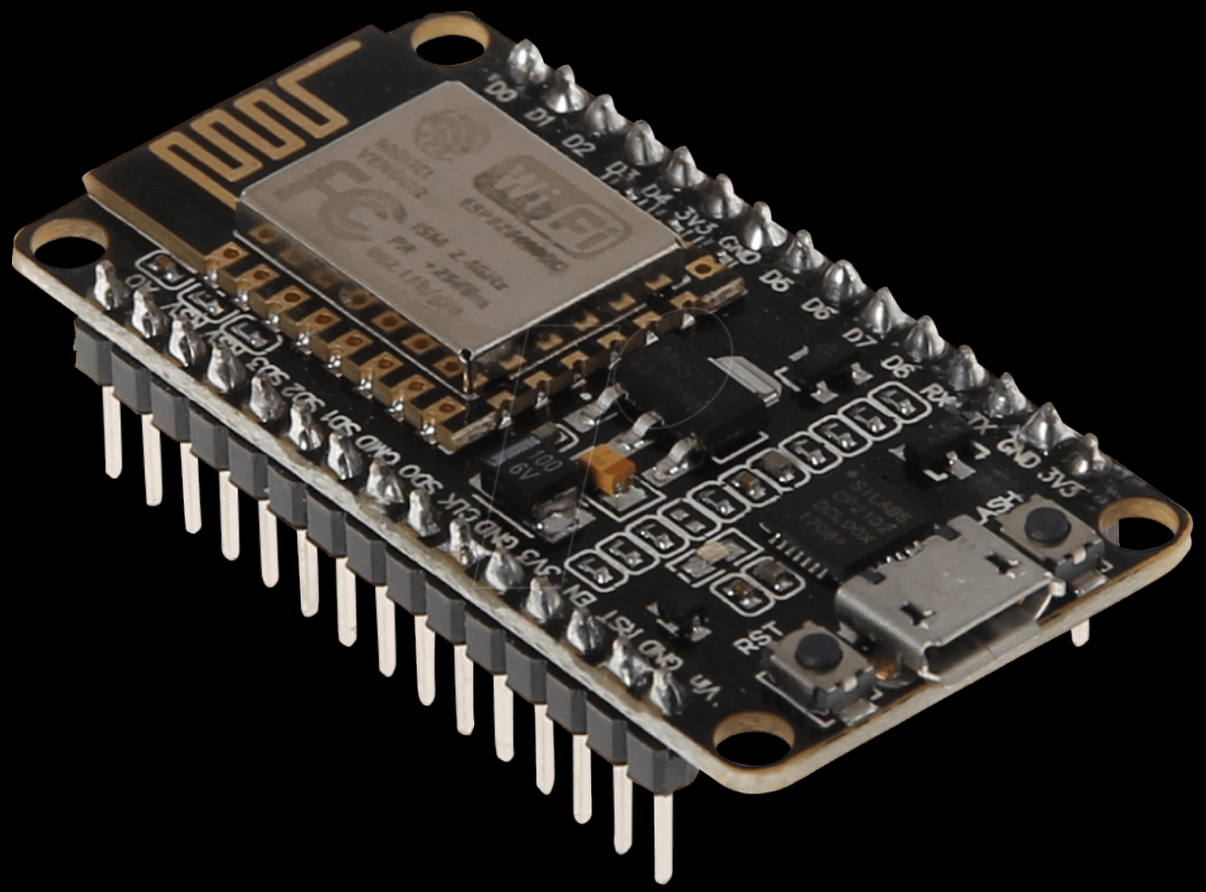
Example Syntax

DigitalReadSerial | Arduino 1.8.7

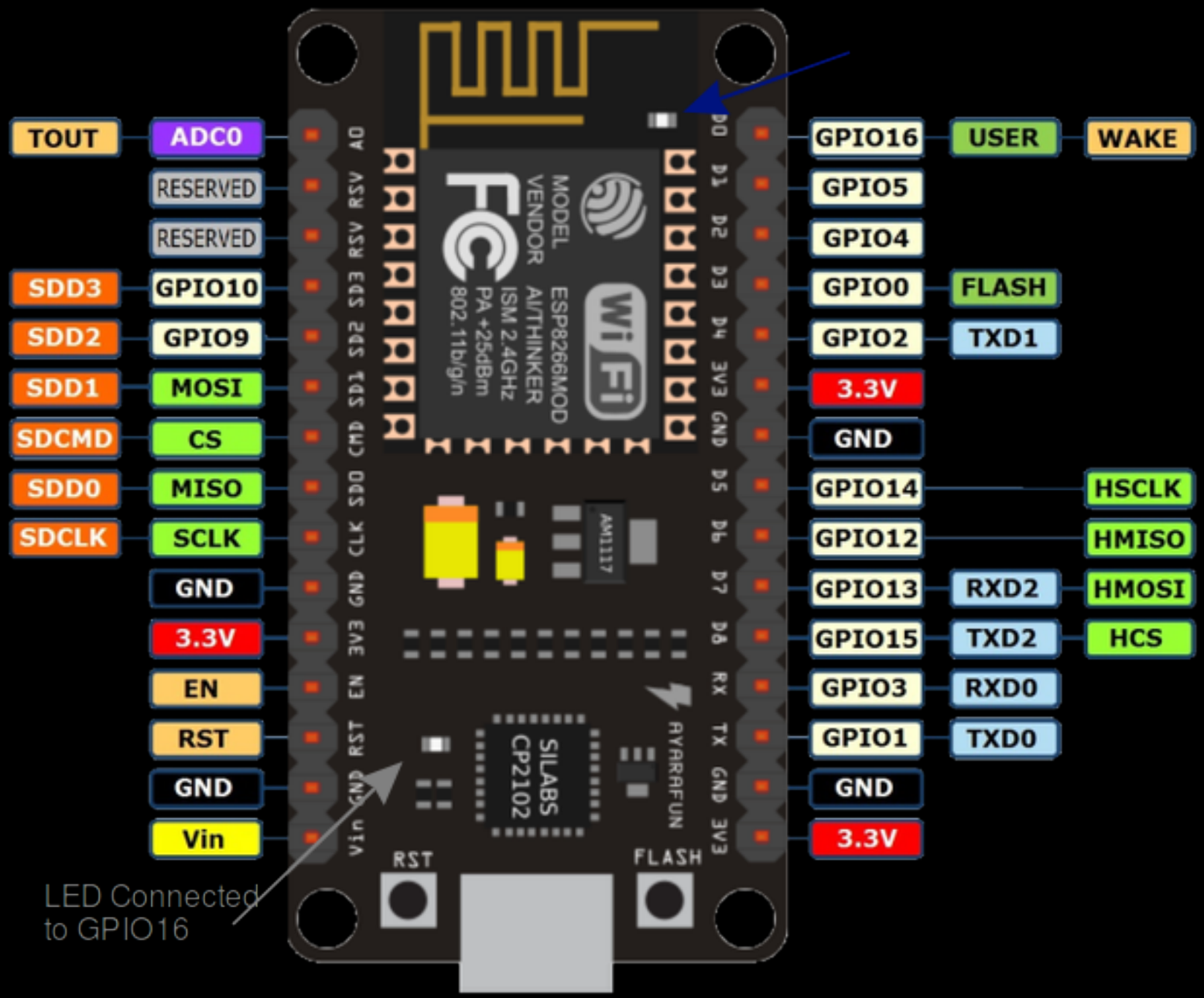
DigitalReadSerial §

```
/*
DigitalReadSerial

Reads a digital input on pin 2, prints the result to the Serial Monitor

This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/DigitalReadSerial
*/

// digital pin 2 has a pushbutton attached to it. Give it a name:
int pushButton = 2;

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // make the pushbutton's pin an input:
  pinMode(pushButton, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input pin:
  int buttonState = digitalRead(pushButton);
  // print out the state of the button:
  Serial.println(buttonState);
  delay(1);        // delay in between reads for stability
}
```

ESP8266


D0 D1 D2 D3 D4 3V3 GND D5 D6 D7 D8 RX TX GND 3V3
A0 RSV RSV SD3 SD2 SD1 CMD SD0 CLK GND 3V3 EN RST GND Vin
RST
FLASH

ESP8266 Pinout



Installing the Drivers

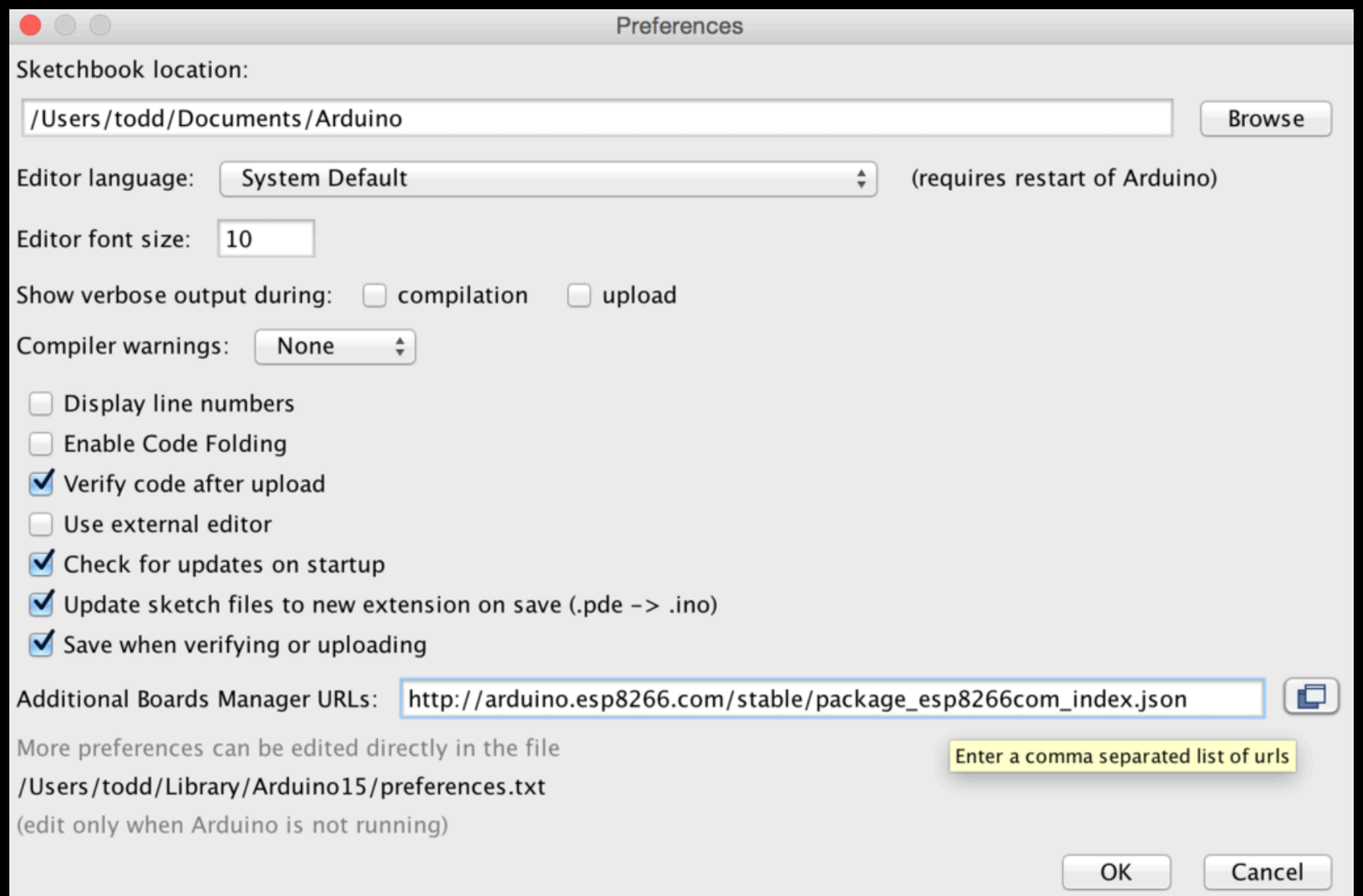
https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

Installing the ESP8266 Board

1. Open the preferences window from the Arduino IDE. Go to File > Preferences

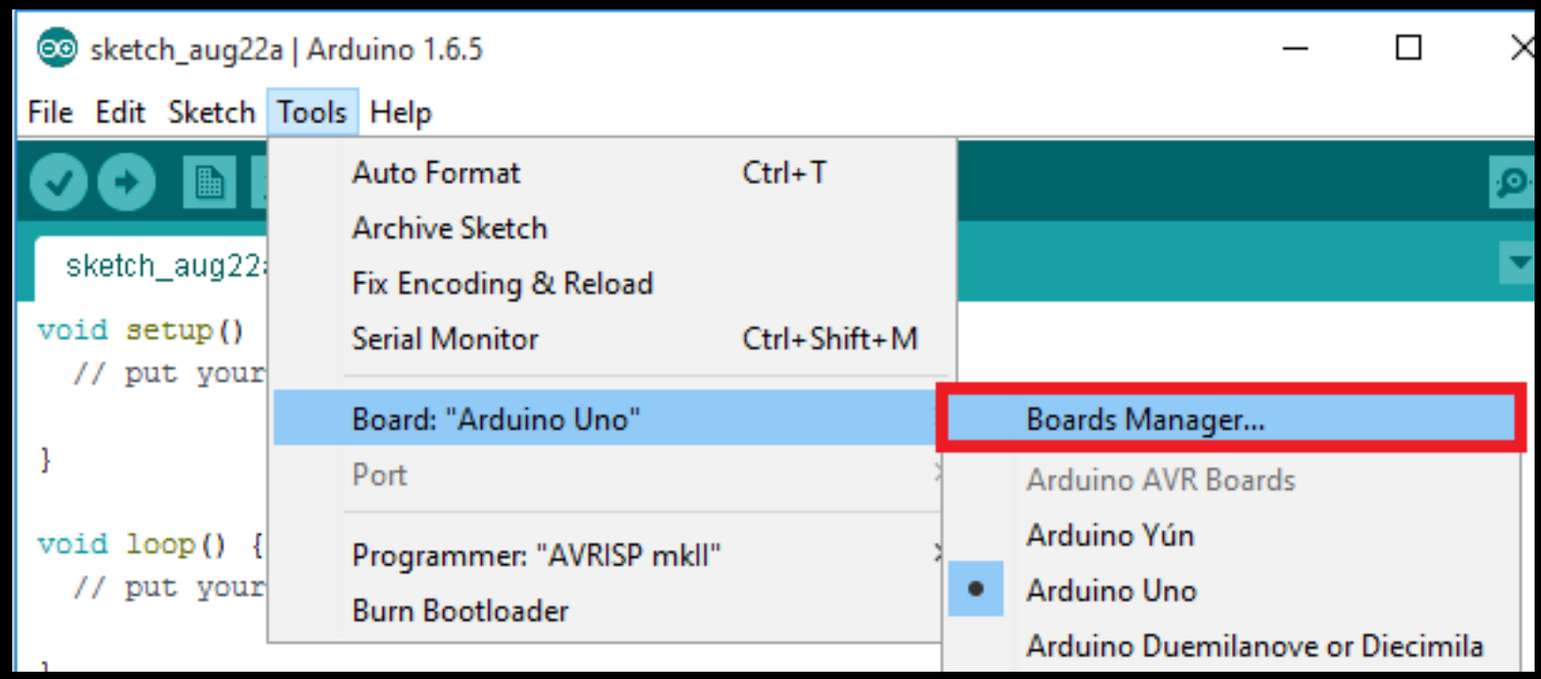
Installing the ESP8266 Board

2. Enter ***http://arduino.esp8266.com/stable/package_esp8266com_index.json*** into the "Additional Board Manager URLs" field as shown in the figure below. Then, click the "OK" button.



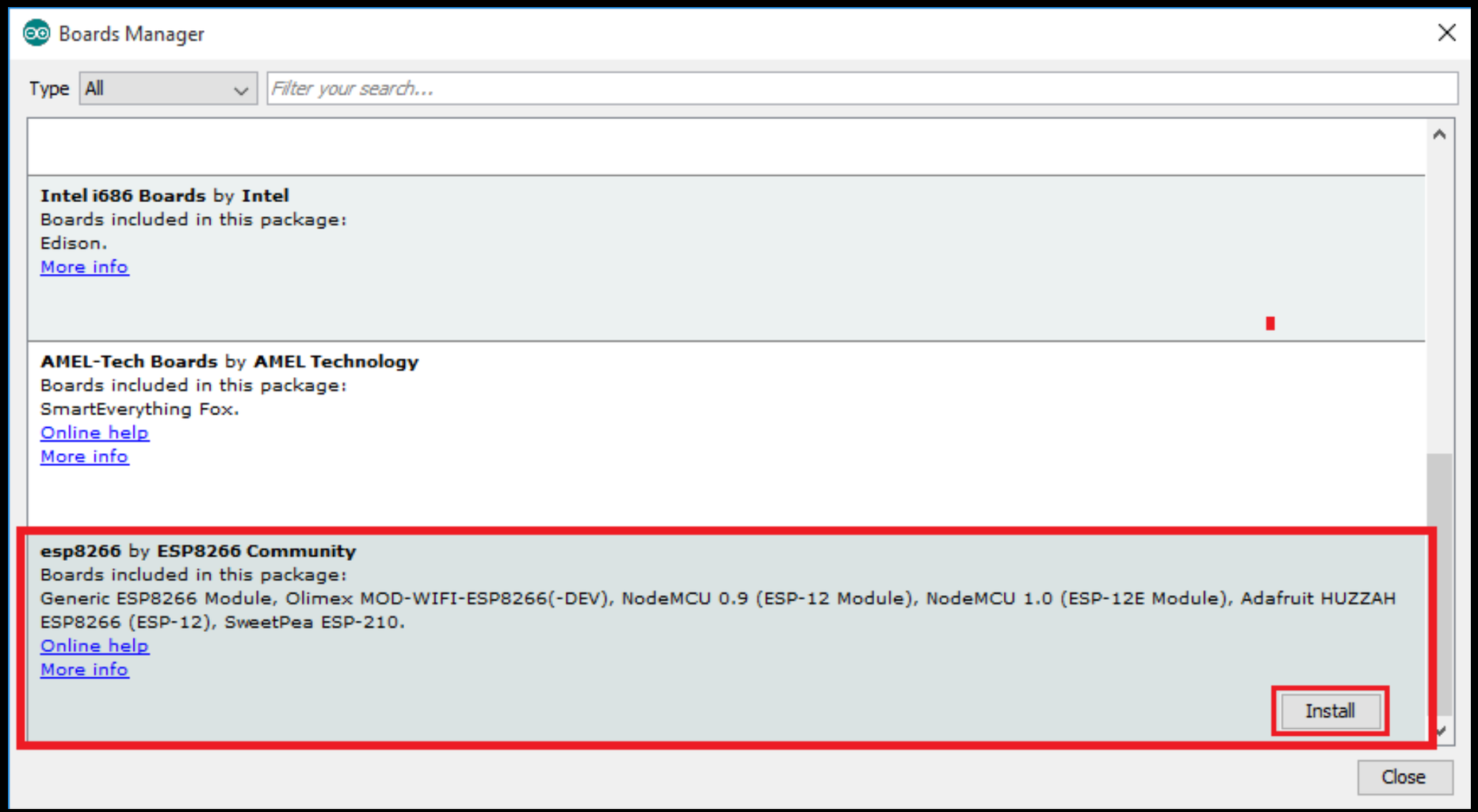
Installing the ESP8266 Board

3. Open boards manager. Go to Tools > Board > Boards Manager...



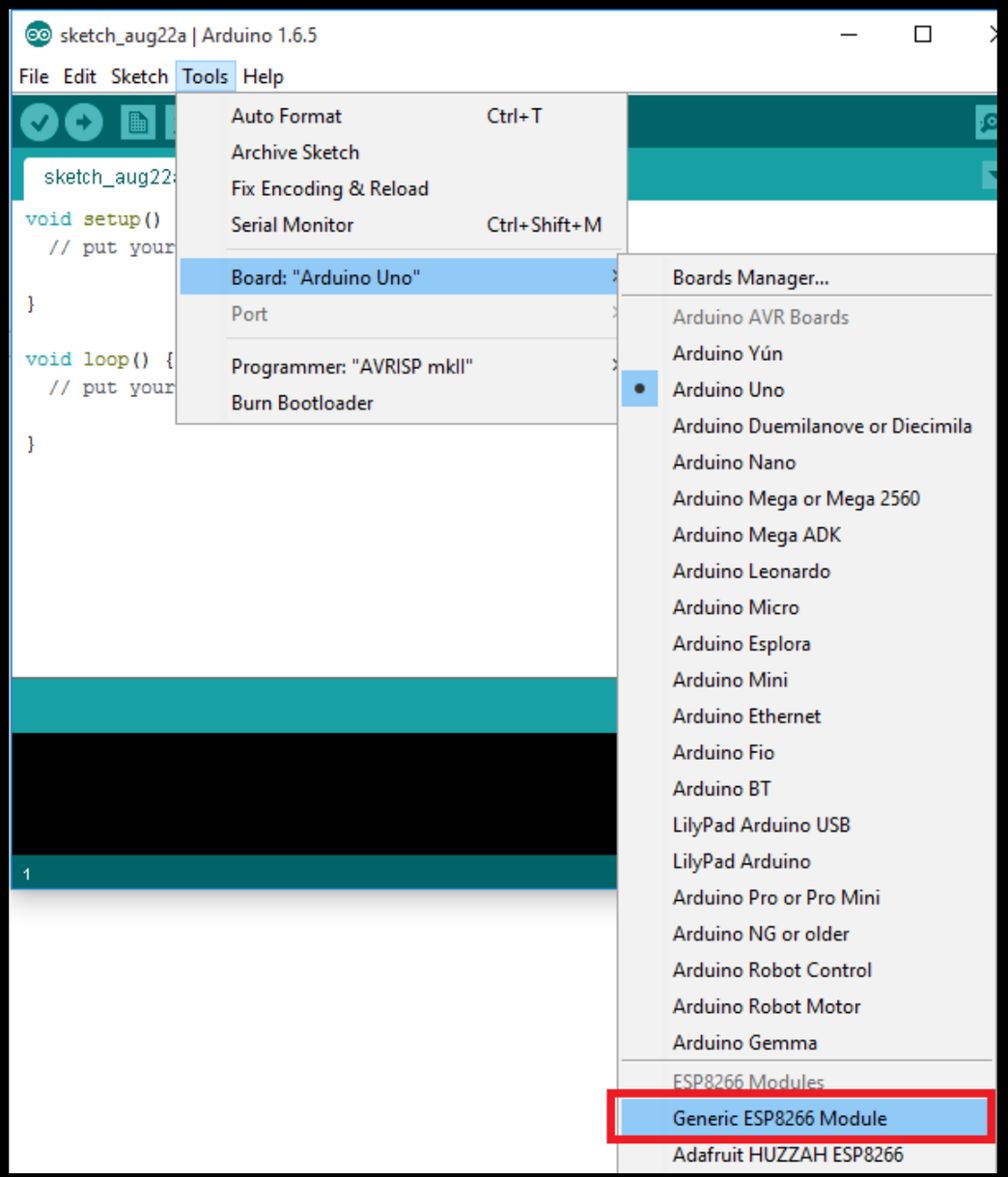
Installing the ESP8266 Board

4. Scroll down, select the ESP8266 board menu and install “esp8266”



Installing the ESP8266 Board

5. Choose your ESP8266 board from Tools > Board > Generic ESP8266 Module



Hello World!


Blink | Arduino 1.8.7
Blink
/*
Blink
Turns an LED on for one second, then off for one second, repeatedly.
Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
the correct LED pin independent of which board is used.
If you want to know what pin the on-board LED is connected to on your Arduino
model, check the Technical Specs of your board at:
https://www.arduino.cc/en/Main/Products
modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016
by Arturo Guadalupi
modified 8 Sep 2016
by Colby Newman
This example code is in the public domain.
http://www.arduino.cc/en/Tutorial/Blink
*/
// the setup function runs once when you press reset or power the board
void setup() {
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}
her HUZZAH ESP8266, 80 MHz, Flash, 4M (1M SPIFFS), v2 Lower Memory, Disabled, None, Only Sketch, 115200 on /dev/cu.SLAB_USBtoUART

3. Blynk

IoT System

1. A complete IoT system needs **hardware**
2. A complete IoT system needs **connectivity**
3. A complete IoT system needs **software**.
4. Finally, a complete IoT system needs a user **interface**.

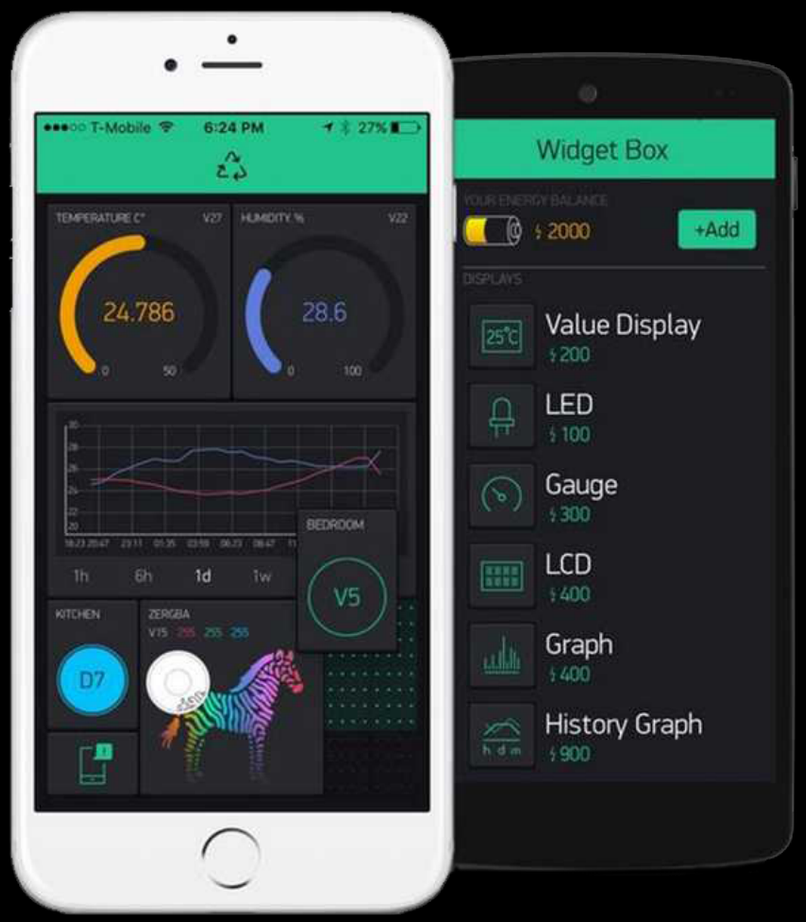
IoT Platform

- Connect hardware
- Handle different communication protocols
- Provide security and authentication for devices and users
- Collect, visualize, and analyze data
- Integrate with other web services

Platforms

1. **PlatformIO** - https://platformio.org/
2. **Oracle Cloud IoT** - https://cloud.oracle.com/iot
3. **Azure IoT Hub** - https://azure.microsoft.com/en-us/services/iot-hub/
4. **IBM IoT Watson** -https://www.ibm.com/internet-of-thing
5. **The Samsung ARTIK** - https://www.artik.io/

Blynk



Getting Started: http://www.blynk.cc/getting-started/

Blynk Server

No laptop required

Blynk app

Blynk Libraries

Internet Access of your choice
Ethernet, Wi-Fi, 3G ...

Blynk Documentation

https://docs.blynk.cc/

Steps Arduino

1. Install Arduino Software

   http://www.arduino.org/software

2. Download Blynk Library

   https://github.com/blynkkk/blynk-library/releases/latest

3. Install Library

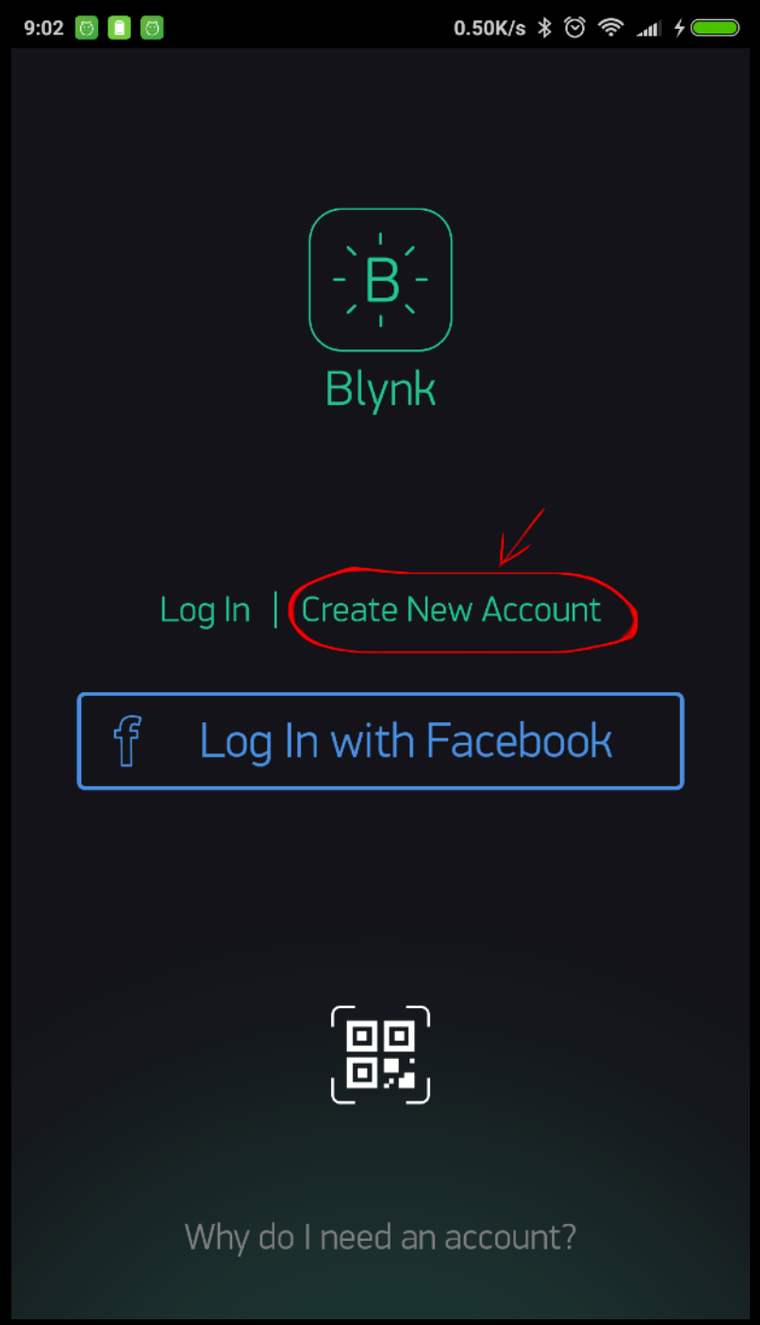
   http://www.arduino.cc/en/guide/libraries

Steps Blynk

1 . Install Blynk Apps for iOS or Android.

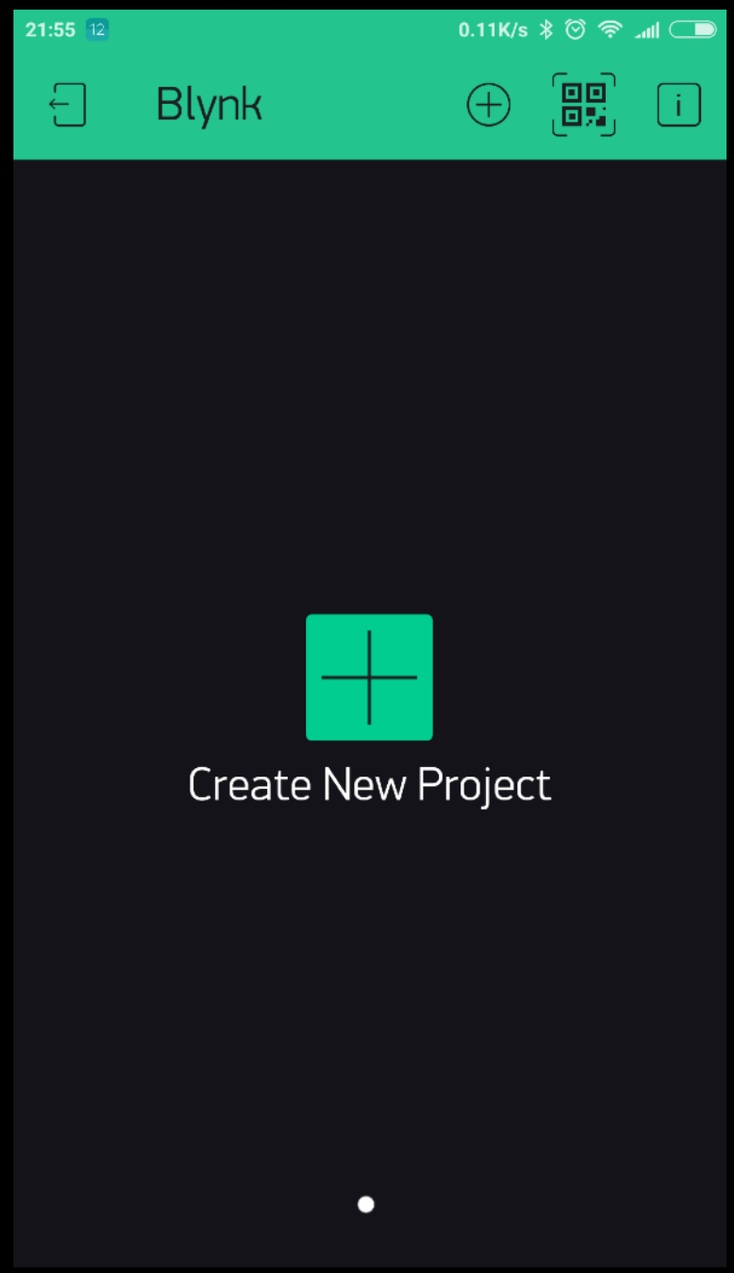
Steps Blynk

2 . Create Blynk Account



Steps Blynk

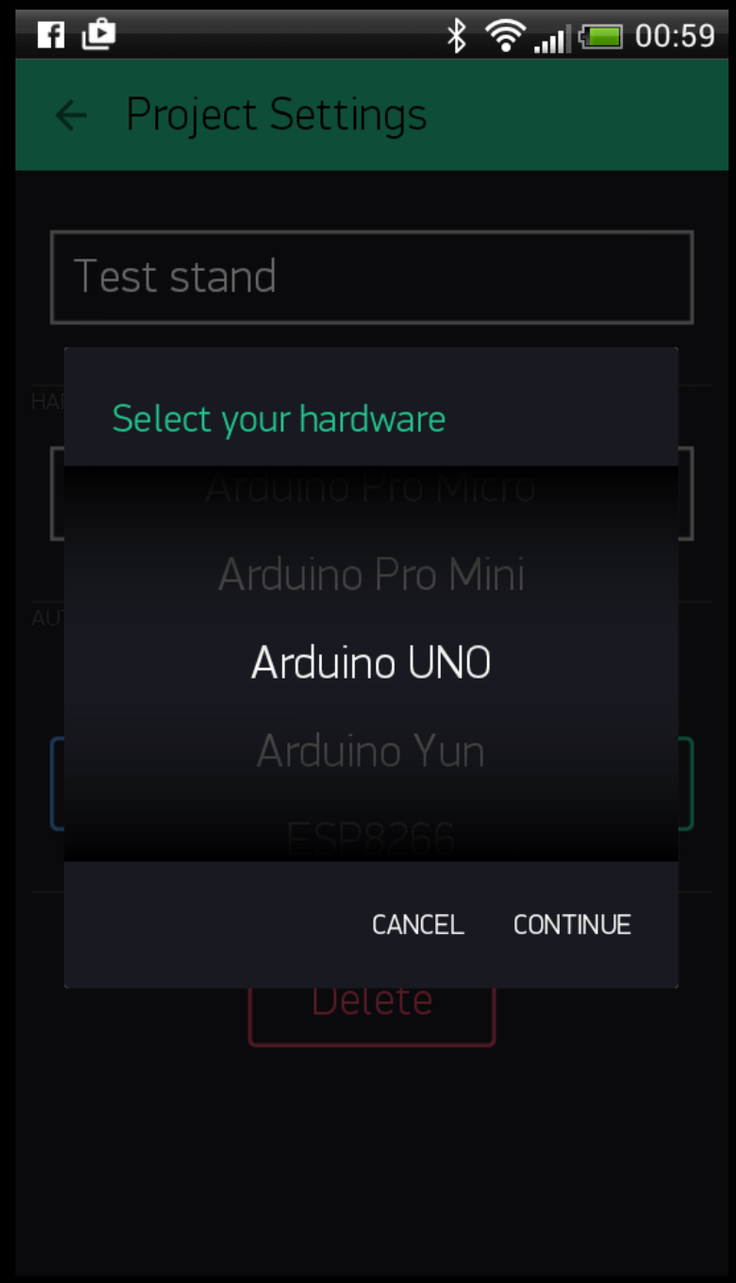
3 . Create a New Project



Steps Blynk

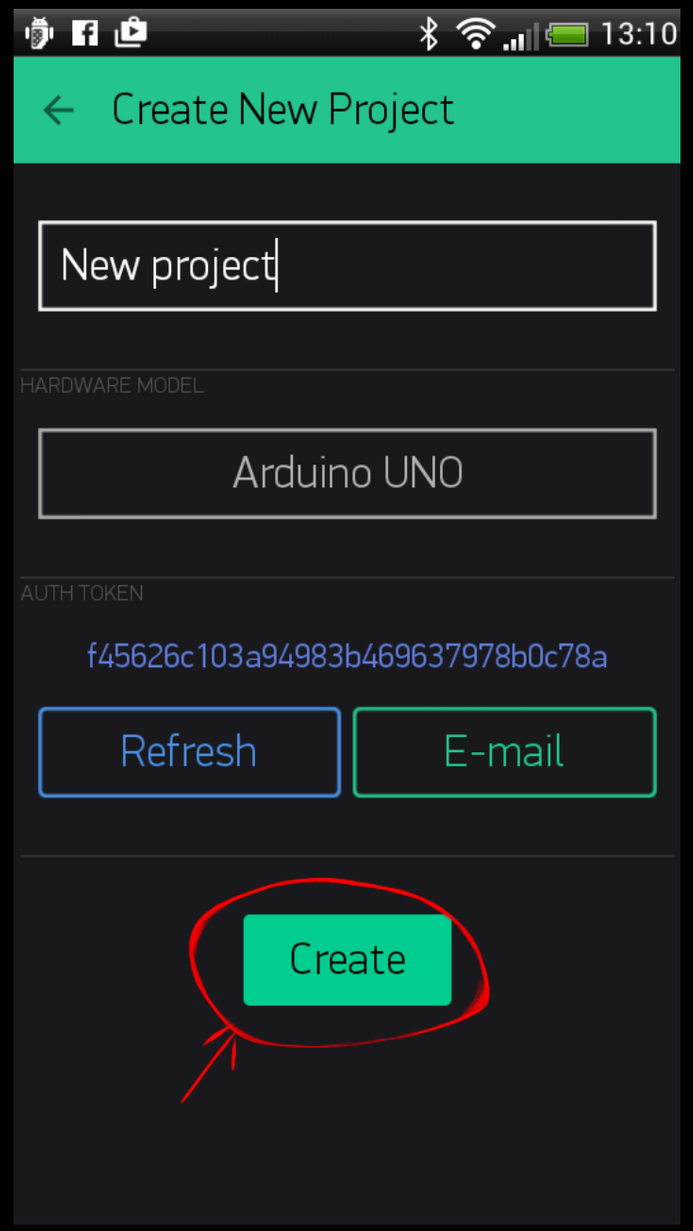
4 . Choose your hardware:

ESP8266



Steps Blynk

5 . Auth Token



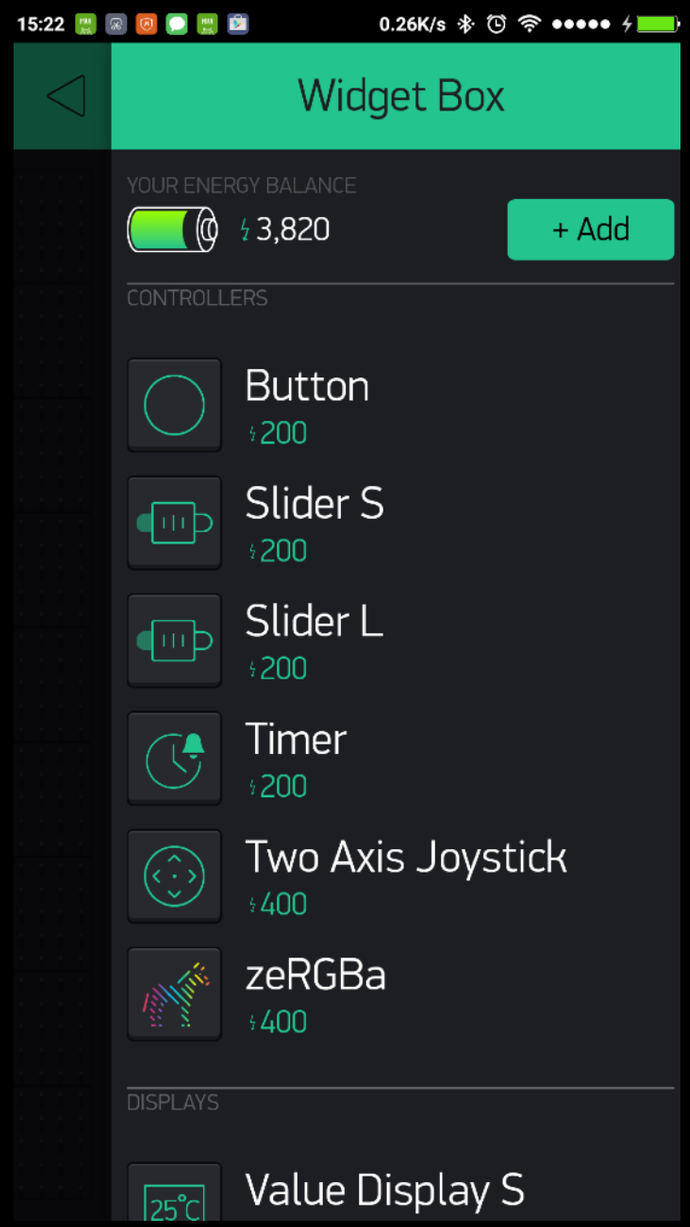
Steps Blynk

6 . Add a Widget

1. Drag-n-Drop Tap and hold the Button Widget to drag it to the new position.

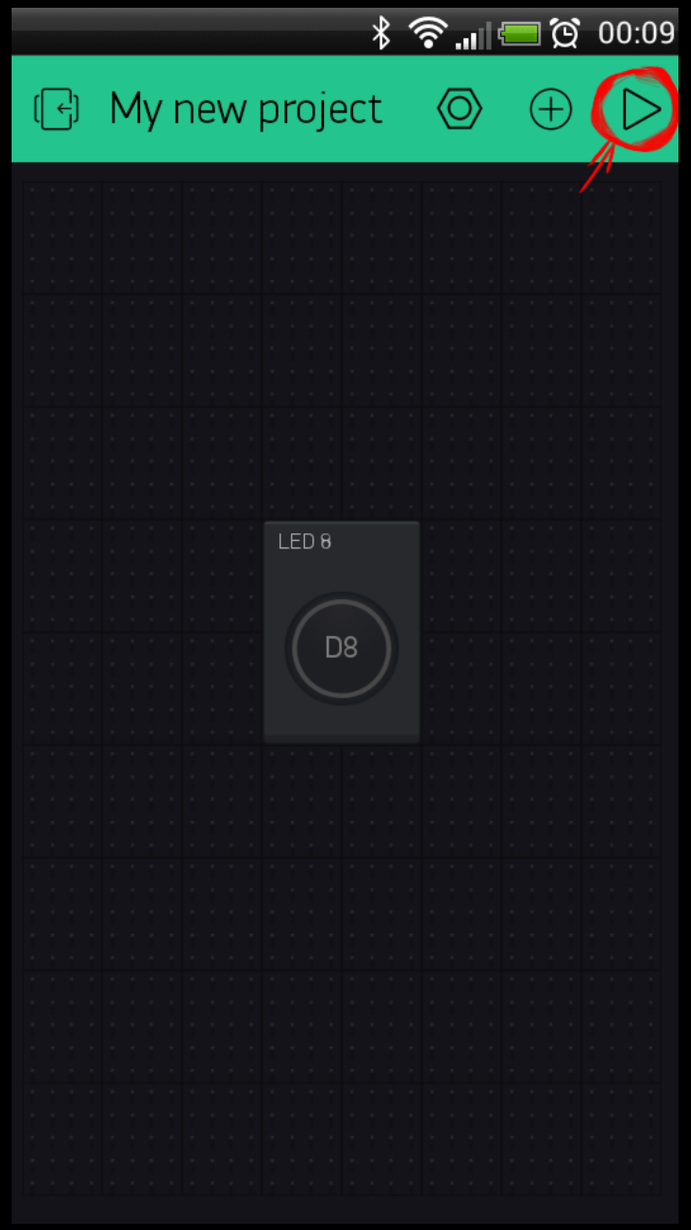
2. Widget Settings Each Widget has it's own settings. Tap on the widget to get to them .

3, Set D0



Steps Blynk

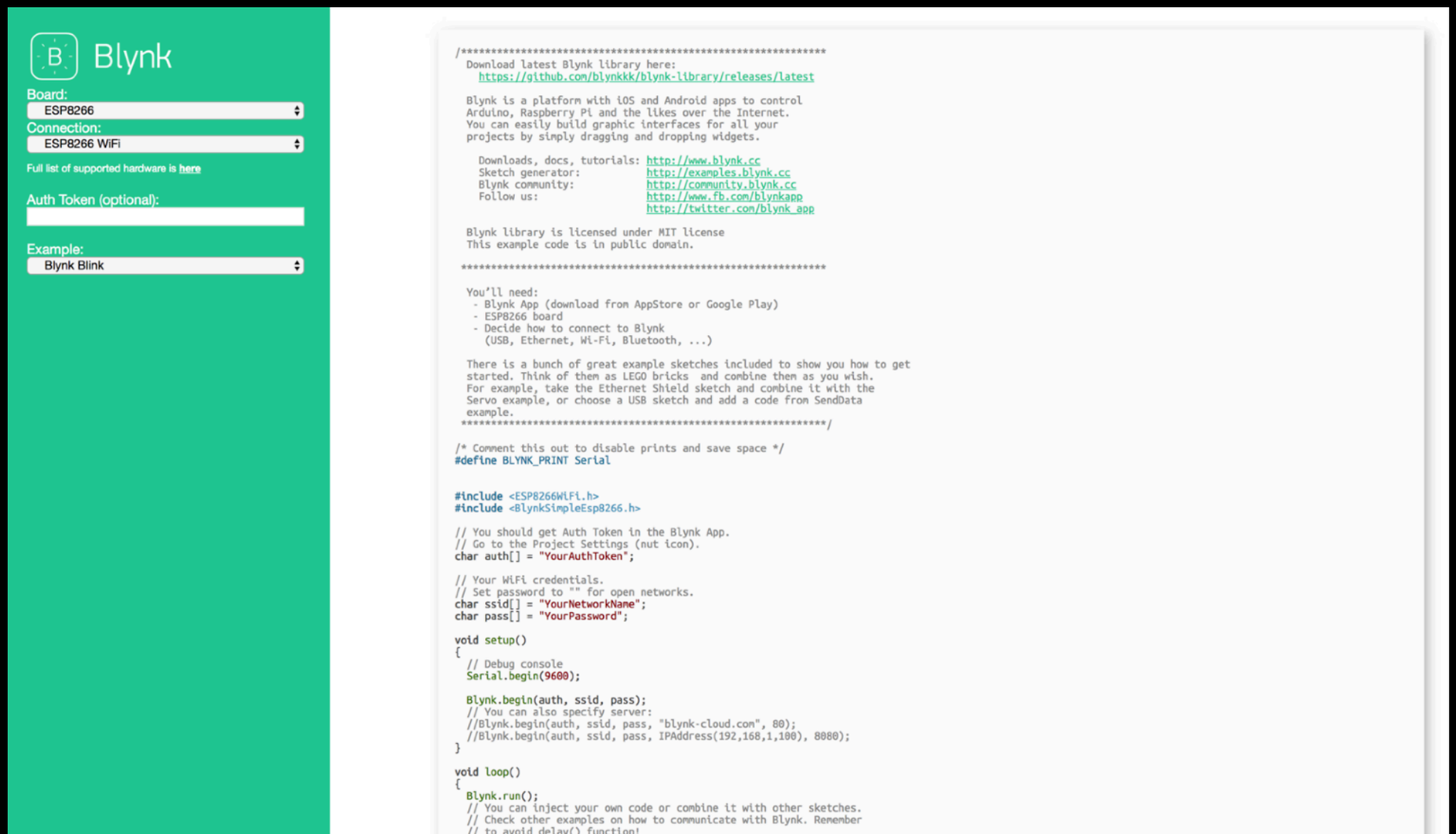
7 . Run The Project



Steps Code

1 . Sketch Code Builder (https://examples.blynk.cc)

1. Choose Your board: ESP8266
2. Choose your connection: ESP8266 WiFi

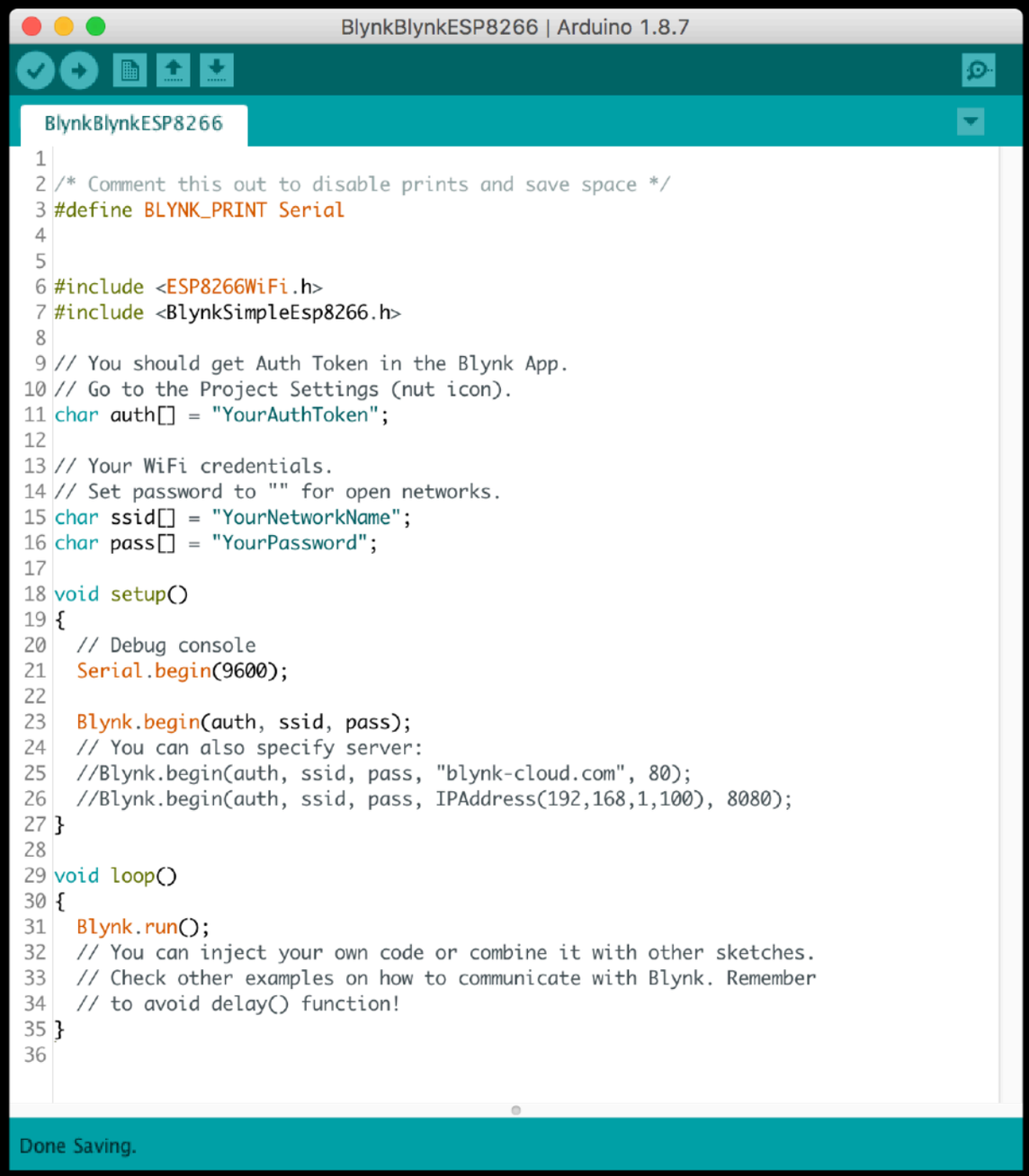



```
/*************************************************************
  Download latest Blynk library here:
    https://github.com/blynkkk/blynk-library/releases/latest

  Blynk is a platform with iOS and Android apps to control
  Arduino, Raspberry Pi and the likes over the Internet.
  You can easily build graphic interfaces for all your
  projects by simply dragging and dropping widgets.

    Downloads, docs, tutorials: http://www.blynk.cc
    Sketch generator:           http://examples.blynk.cc
    Blynk community:            http://community.blynk.cc
    Follow us:                  http://www.fb.com/blynkapp
                                http://twitter.com/blynk_app

  Blynk library is licensed under MIT license
  This example code is in public domain.

 *************************************************************

  You'll need:
   - Blynk App (download from AppStore or Google Play)
   - ESP8266 board
   - Decide how to connect to Blynk
     (USB, Ethernet, Wi-Fi, Bluetooth, ...)

  There is a bunch of great example sketches included to show you how to get
  started. Think of them as LEGO bricks  and combine them as you wish.
  For example, take the Ethernet Shield sketch and combine it with the
  Servo example, or choose a USB sketch and add a code from SendData
  example.
 *************************************************************/

/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial


#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

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

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

void setup()
{
  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}

void loop()
{
  Blynk.run();
  // You can inject your own code or combine it with other sketches.
  // Check other examples on how to communicate with Blynk. Remember
  // to avoid delay() function!
```

Steps Code

2 . Create new sketch

1. Copy example code
2. Auth Token
3. WiFi Credentials



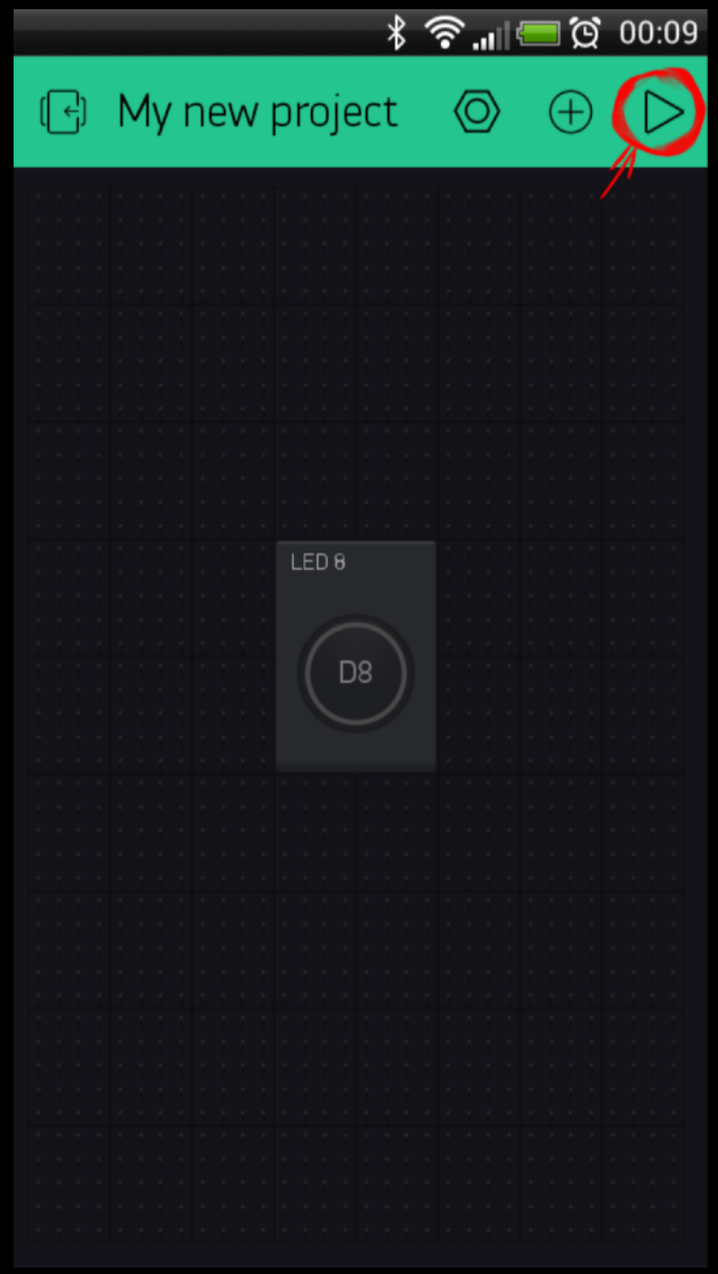
Steps Code

3 . Upload Sketch

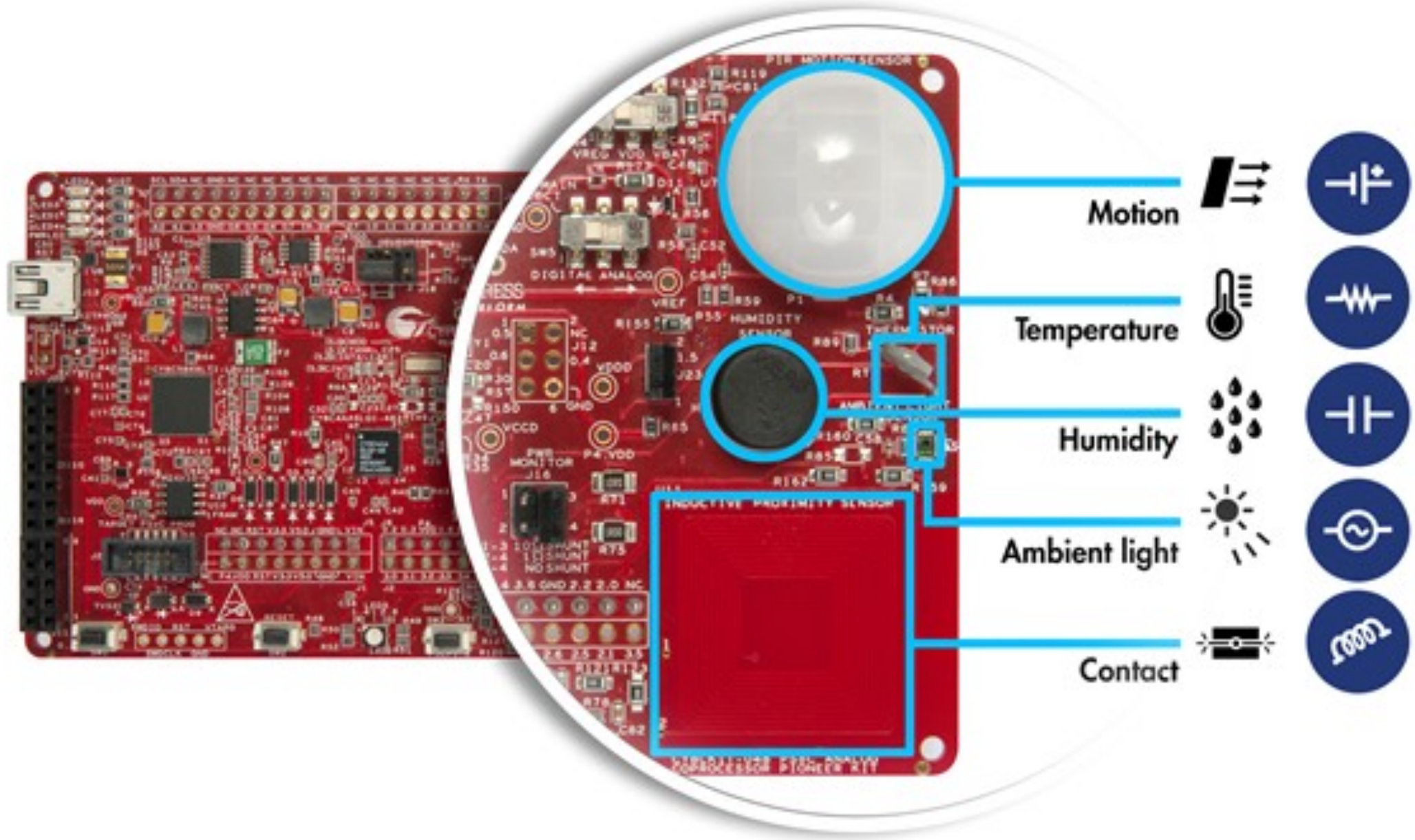
1. Open serial monitor in Arduino IDE.

Steps Code

4 . Run the Blynk project



4. Sensors


Motion
Temperature
Humidity
Ambient light
Contact

Finger-Pulse sensor Module

Hall sensor Module

Microphone sensor Module

Obstacle avoidance sensor Module

Passive Buzzer Module

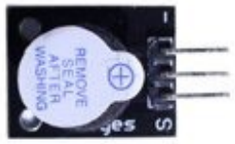
Flame sensor Module

Colorful Auto-flash Module

infrared-transmit Module

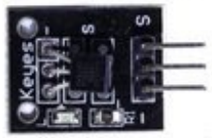
common-cathode RED&GREEN LED Module

Linear-Hall Sensor Module



Active buzzer Module

Magnet-ring sensor Module



18b20 temperature sensor Module

RGB LED Module

Analog-temperature sensor Module

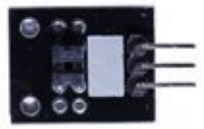
Magic-ring Module

tilt-switch Module

Shock-switch sensor Module

Rotate-encode Module

Relay Module



Light break sensor Module

Magnetic spring Module

Digital-Temperature sensor Module

High-sensitive voice sensor Module

Photo resistor sensor Module

Metal touch sensor Module

hydrargyrum-switch sensor Module

Infrared-receive sensor Module

RGB LED Module

Push button Module

humiture sensor Module

Laser-transmit Module

two-color commoncathode LED Module

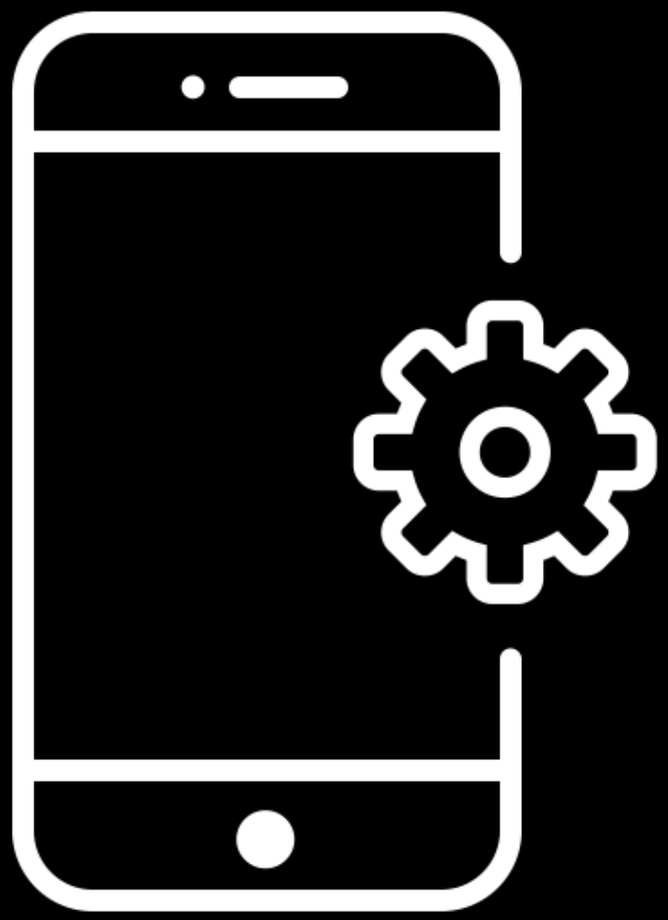
Knock sensor module

Tracking sensor Module

Joystick PS2 Module

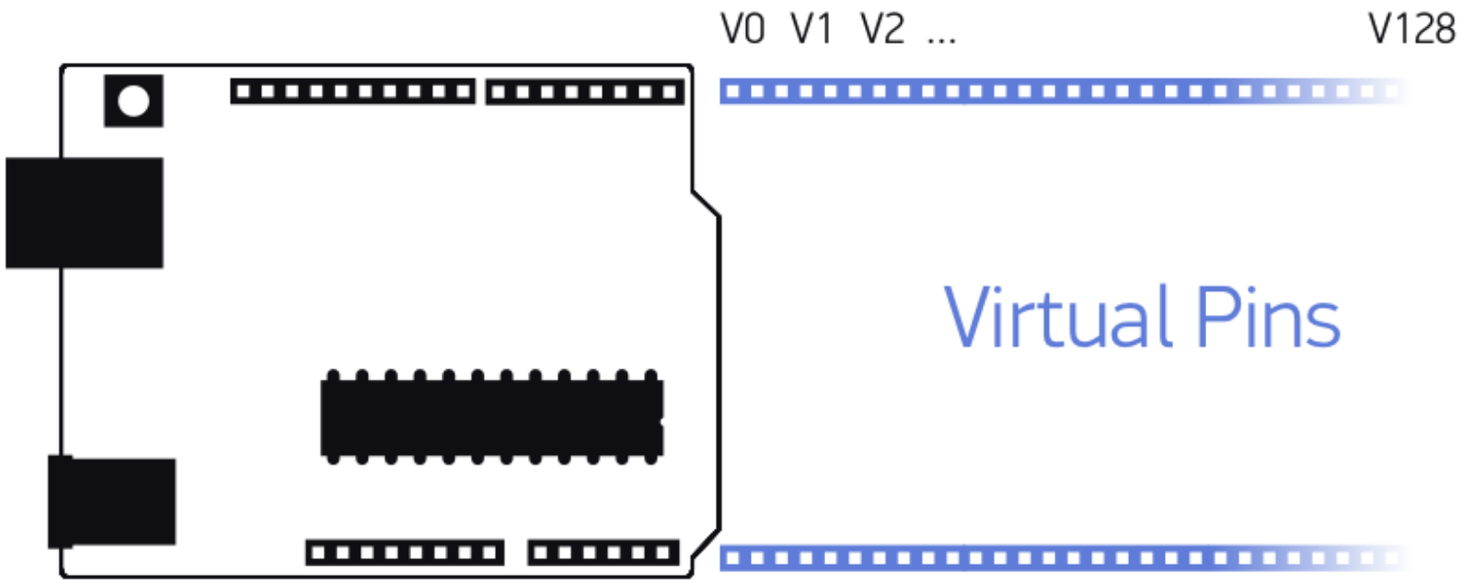
Analogy-Hall sensor Module

Smartphone Sensors



- Accelerometer
- Gravity
- Barometer/Pressure
- Humidity
- Light
- Proximity
- Temperature
- GPS

Virtual Pins



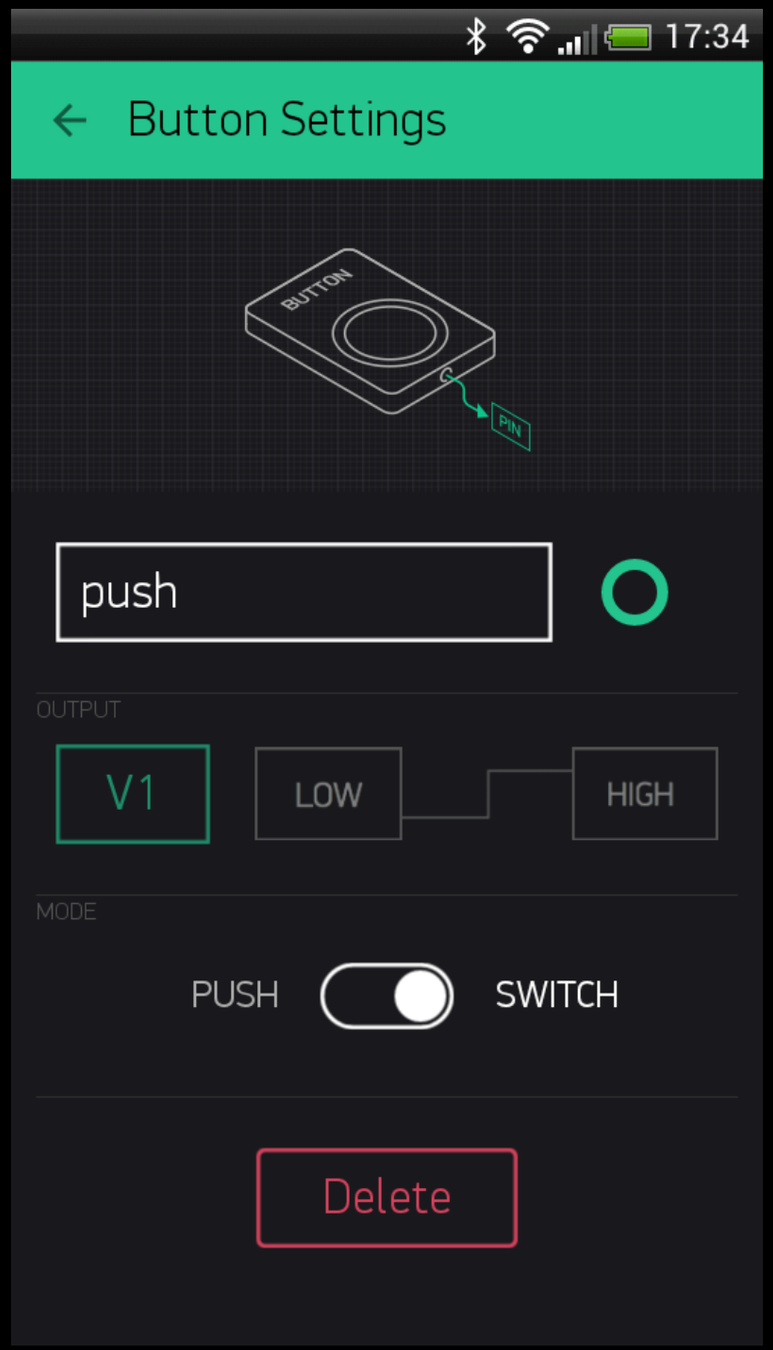
Get Data

GetDataESP8266 | Arduino 1.8.7

GetDataESP8266 §

```
/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial


#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

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

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

// This function will be called every time Slider Widget
// in Blynk app writes values to the Virtual Pin 1
BLYNK_WRITE(V1)
{
  int pinValue = param.asInt(); // assigning incoming value from pin V1 to a variable
  // You can also use:
  // String i = param.asStr();
  // double d = param.asDouble();
  Serial.print("V1 Slider value is: ");
  Serial.println(pinValue);
}

void setup()
{
  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}

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

Get Data


17:34
Button Settings
BUTTON
PIN
push
OUTPUT
V1
LOW
HIGH
MODE
PUSH
SWITCH
Delete

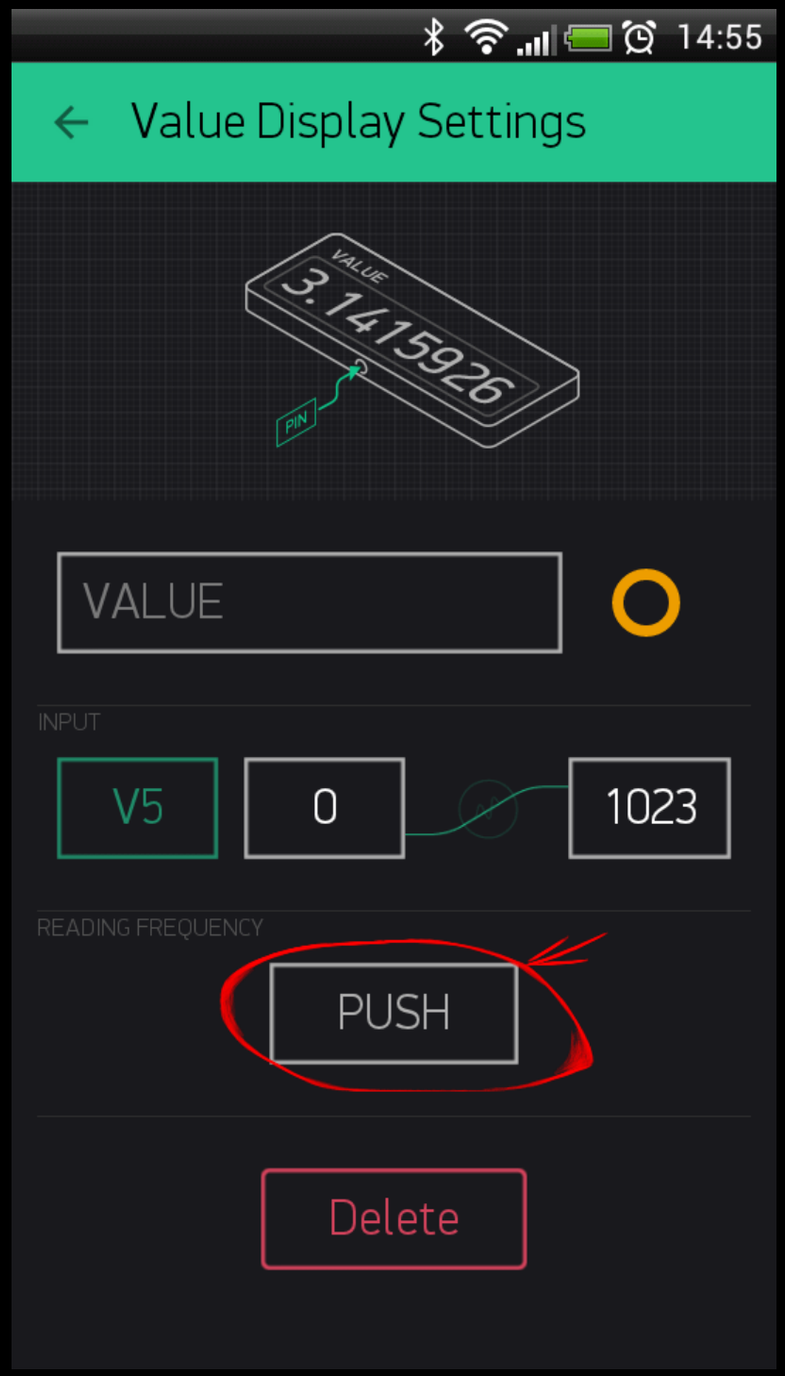
Push Data

PushDataESP8266 | Arduino 1.8.7

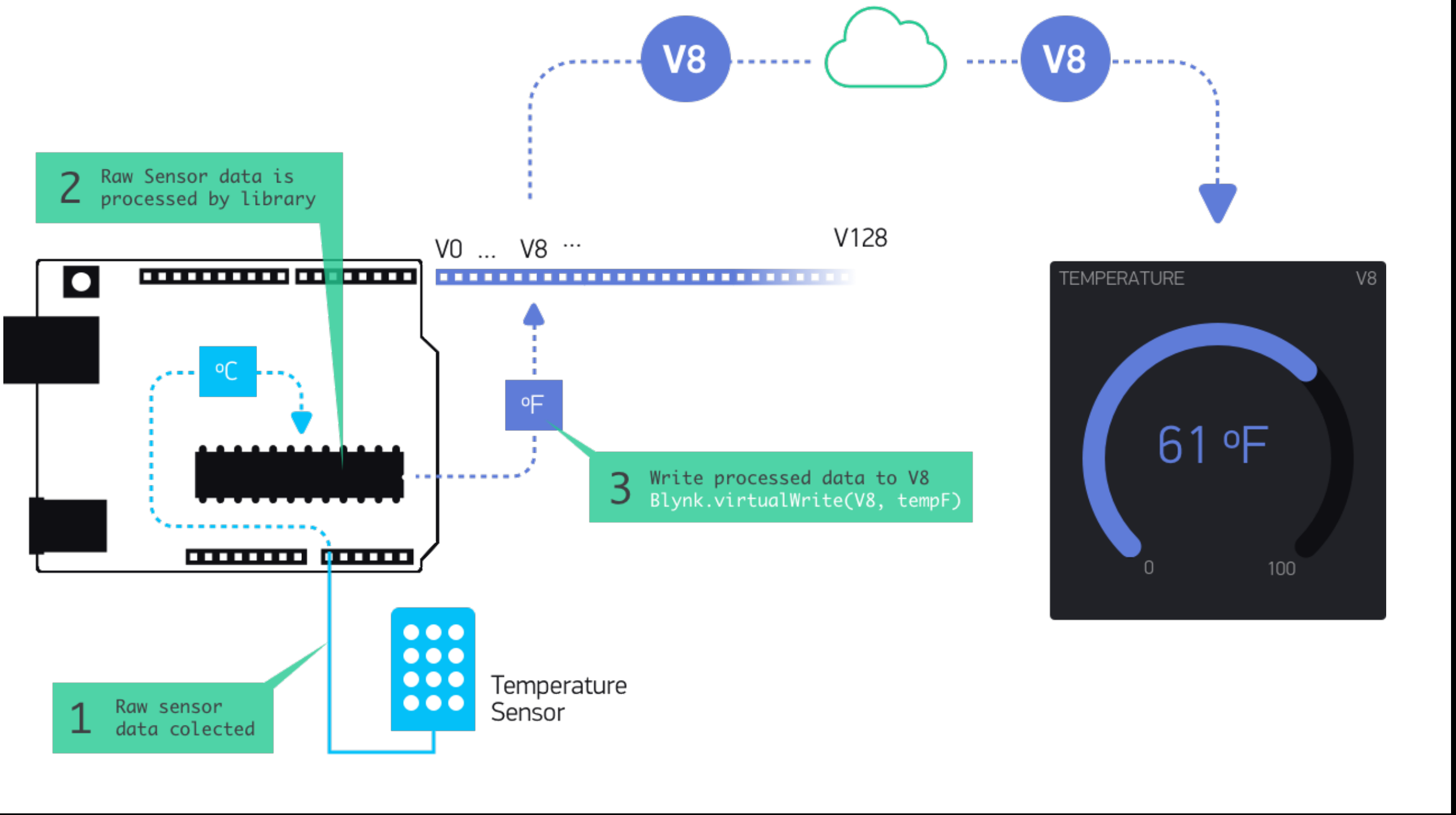
PushDataESP8266

```
/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial


#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

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

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

// Use Virtual pin 5 for uptime display
#define PIN_UPTIME V5

// This function tells Arduino what to do if there is a Widget
// which is requesting data for Virtual Pin (5)
BLYNK_READ(PIN_UPTIME)
{
  // This command writes Arduino's uptime in seconds to Virtual Pin (5)
  Blynk.virtualWrite(PIN_UPTIME, millis() / 1000);
}

void setup()
{
  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}

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

Push Data


14:55
Value Display Settings
VALUE
3.14159926
PIN
VALUE
INPUT
V5
0
1023
READING FREQUENCY
PUSH
Delete

Temperature



Temperature

DhtESP8266 | Arduino 1.8.7

DhtESP8266 §

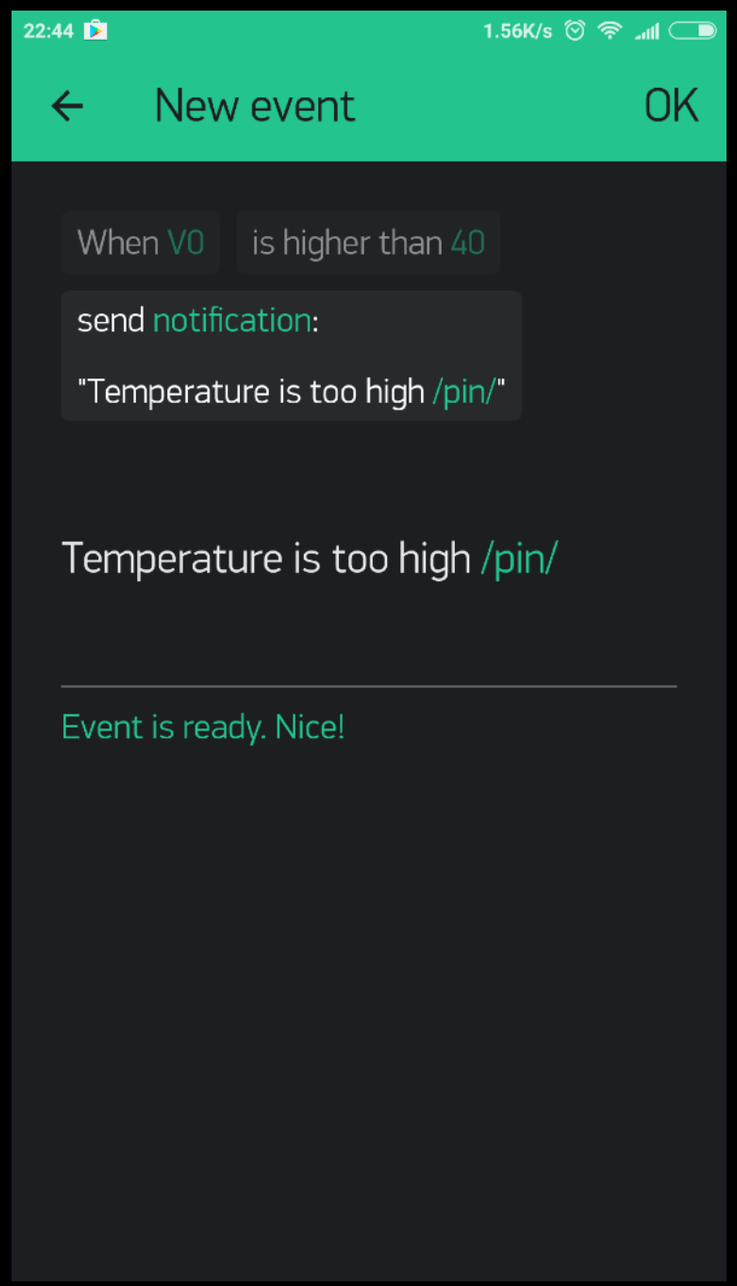
```
#include <BlynkSimpleEsp8266.h>
#include <DHT.h>

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

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YourNetworkName";
char pass[] = "YourPassword";

#define DHTPIN 2          // What digital pin we're connected to

// Uncomment whatever type you're using!
#define DHTTYPE DHT11     // DHT 11
//#define DHTTYPE DHT22   // DHT 22, AM2302, AM2321
//#define DHTTYPE DHT21   // DHT 21, AM2301

DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;

// This function sends Arduino's up time every second to Virtual Pin (5).
// In the app, Widget's reading frequency should be set to PUSH. This means
// that you define how often to send data to Blynk App.
void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  // You can send any value at any time.
  // Please don't send more that 10 values per second.
  Blynk.virtualWrite(V5, h);
  Blynk.virtualWrite(V6, t);
}

void setup()
{
  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);

  dht.begin();

  // Setup a function to be called every second
  timer.setInterval(1000L, sendSensor);
}

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

Eventor

```
float t = dht.readTemperature();
if (isnan(t)) {
  return;
}
if (t > 40) {
  Blynk.notify(String("Temperature is too high: ") + t);
}
```

Eventor

```
float t = dht.readTemperature();
Blynk.virtualWrite(V0, t);
```



Dash


10:29
Logger
TEMP °C V21
28.8
HUMIDITY V22
34.1
MOIST 2 V23
102
MOIST 2 V24
84
1d
1w
1m
3m
All
MOIST 1 A4
2197
0
4095
MOIST 2 A5
2749
0
4095
BUTTON
D7
BUTTON
V25


orange
08:43
91 %
Garden
30 MINS
60 MINS
90 MINS
120 MIN
V11
V12
V13
V14
STATUS 1 V5
ON
STATUS 2 V6
ON
STATUS 3 V7
off
STATUS 4 V8
ON
LEFT 1 V1
19
LEFT 2 V2
49
LEFT 3 V3
0
LEFT 4 V4
114
SELECTED CHANNEL
V0: 4
STOP
V18
STOP ALL
V9
STATUS V10
off
UPTIME V30
766
RESET
V31

Libelium Smart World

Air Pollution
Control of CO_2 emissions of factories, pollution emitted by cars and toxic gases generated in farms.

Forest Fire Detection
Monitoring of combustion gases and preemptive fire conditions to define alert zones.

Wine Quality Enhancing
Monitoring soil moisture and trunk diameter in vineyards to control the amount of sugar in grapes and grapevine health.

Offspring Care
Control of growing conditions of the offspring in animal farms to ensure its survival and health.

Sportsmen Care
Vital signs monitoring in high performance centers and fields.

Structural Health
Monitoring of vibrations and material conditions in buildings, bridges and historical monuments.

Smartphones Detection
Detect iPhone and Android devices and in general any device which works with Wifi or Bluetooth interfaces.

Perimeter Access Control
Access control to restricted areas and detection of people in non-authorized areas.

Radiation Levels
Distributed measurement of radiation levels in nuclear power stations surroundings to generate leakage alerts.

Electromagnetic Levels
Measurement of the energy radiated by cell stations and and WiFi routers.

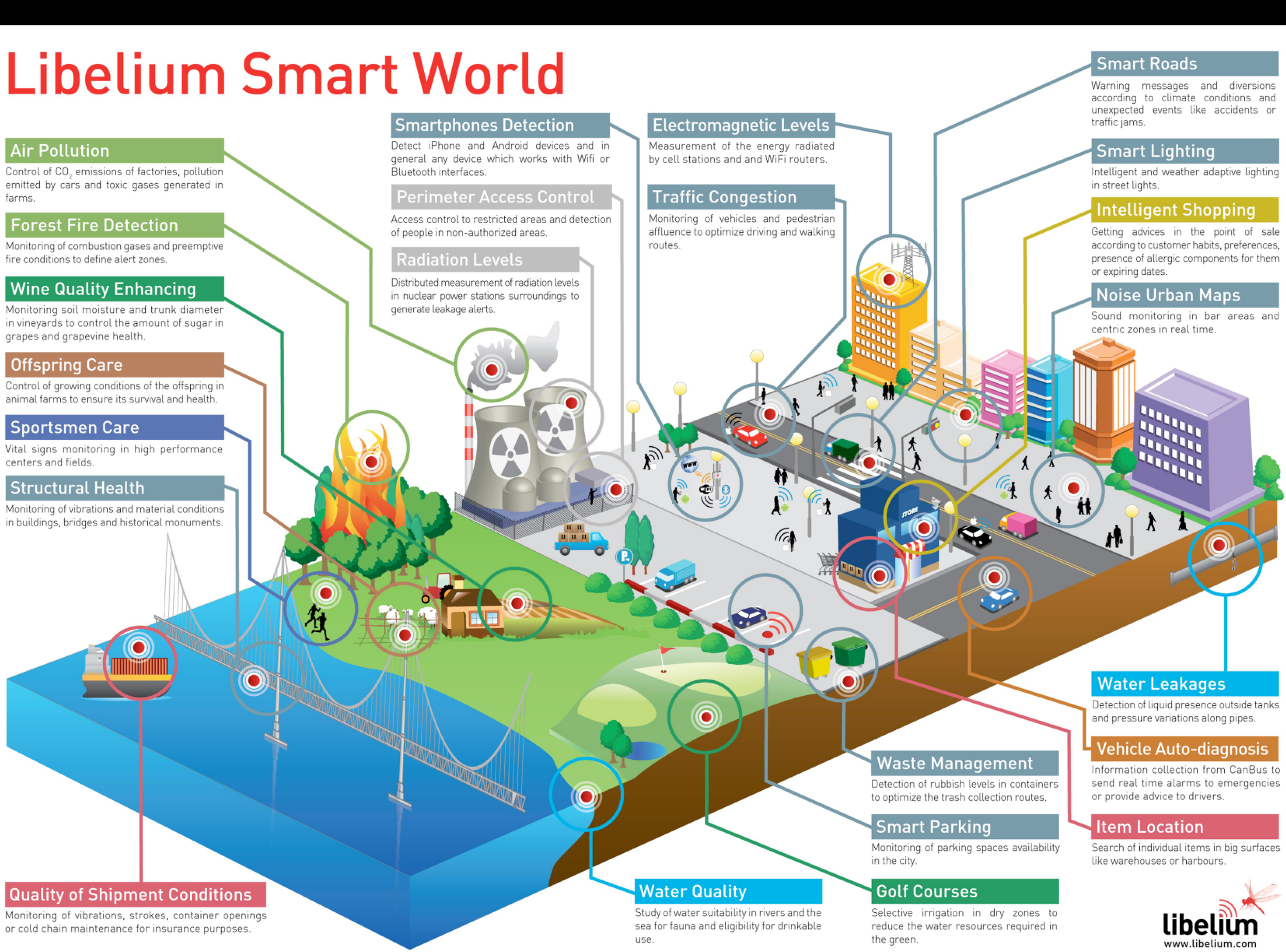
Traffic Congestion
Monitoring of vehicles and pedestrian affluence to optimize driving and walking routes.

Smart Roads
Warning messages and diversions according to climate conditions and unexpected events like accidents or traffic jams.

Smart Lighting
Intelligent and weather adaptive lighting in street lights.

Intelligent Shopping
Getting advices in the point of sale according to customer habits, preferences, presence of allergic components for them or expiring dates.

Noise Urban Maps
Sound monitoring in bar areas and centric zones in real time.



Water Leakages
Detection of liquid presence outside tanks and pressure variations along pipes.

Vehicle Auto-diagnosis
Information collection from CanBus to send real time alarms to emergencies or provide advice to drivers.

Item Location
Search of individual items in big surfaces like warehouses or harbours.

Waste Management
Detection of rubbish levels in containers to optimize the trash collection routes.

Smart Parking
Monitoring of parking spaces availability in the city.

Golf Courses
Selective irrigation in dry zones to reduce the water resources required in the green.

Water Quality
Study of water suitability in rivers and the sea for fauna and eligibility for drinkable use.

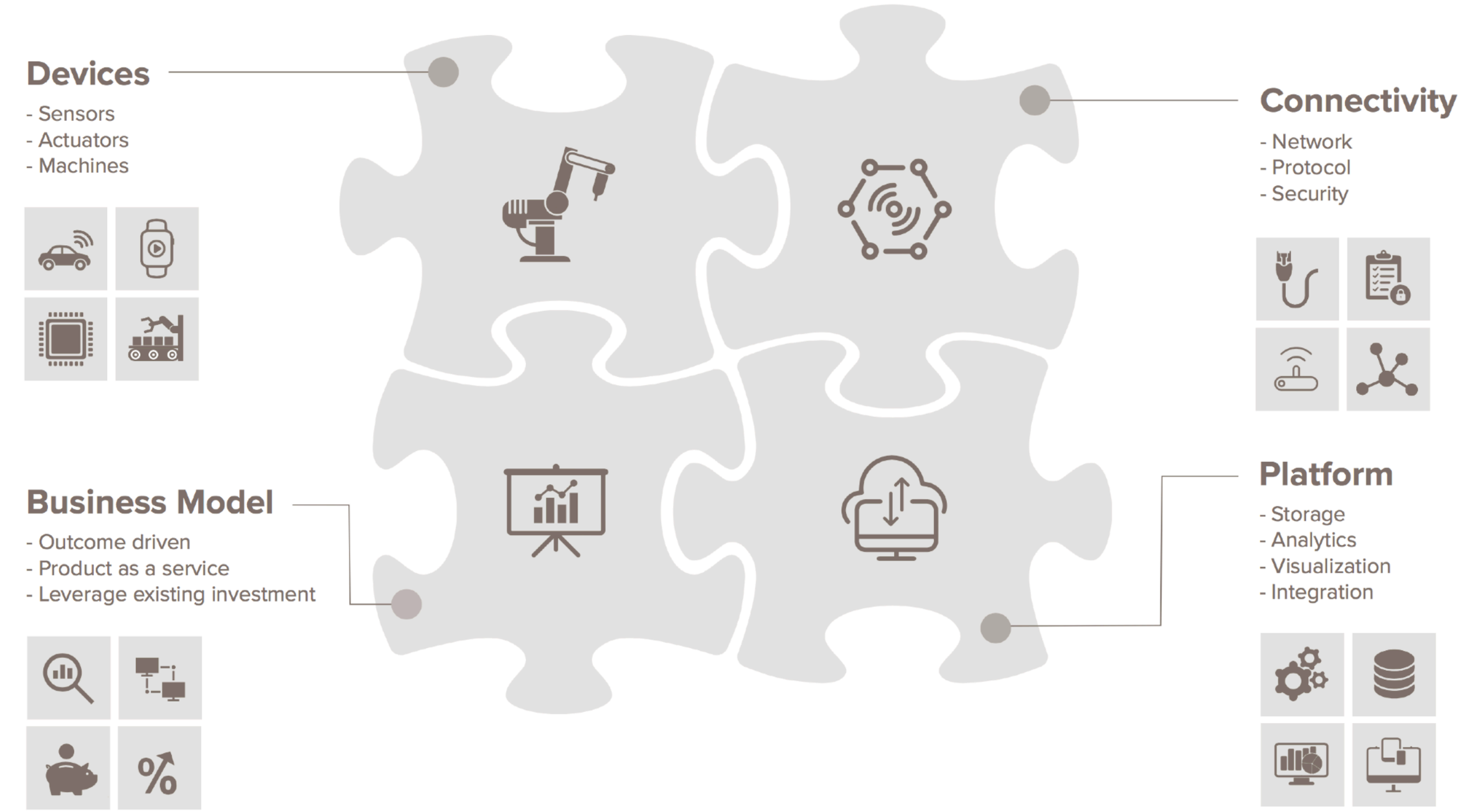
Quality of Shipment Conditions
Monitoring of vibrations, strokes, container openings or cold chain maintenance for insurance purposes.

libelium
www.libelium.com

5. Hackathon

6. Future Work

Parts of an IoT System



Platform

1. **PlatformIO** - https://platformio.org/
2. **ThingerIO -** https://thinger.io/
3. **Oracle Cloud IoT** - https://cloud.oracle.com/iot
4. **Azure IoT Hub** - https://azure.microsoft.com/en-us/services/iot-hub/
5. **IBM IoT Watson** -https://www.ibm.com/internet-of-thing
6. **The Samsung ARTIK** - https://www.artik.io/

Devices

1. **Reliability -** warranty
2. **Scalability:** Getting ready for the big time
3. **Costs:** Counting Pennies
4. **Power consumption:** Every milliamp matters
5. **Security:** You will be attacked

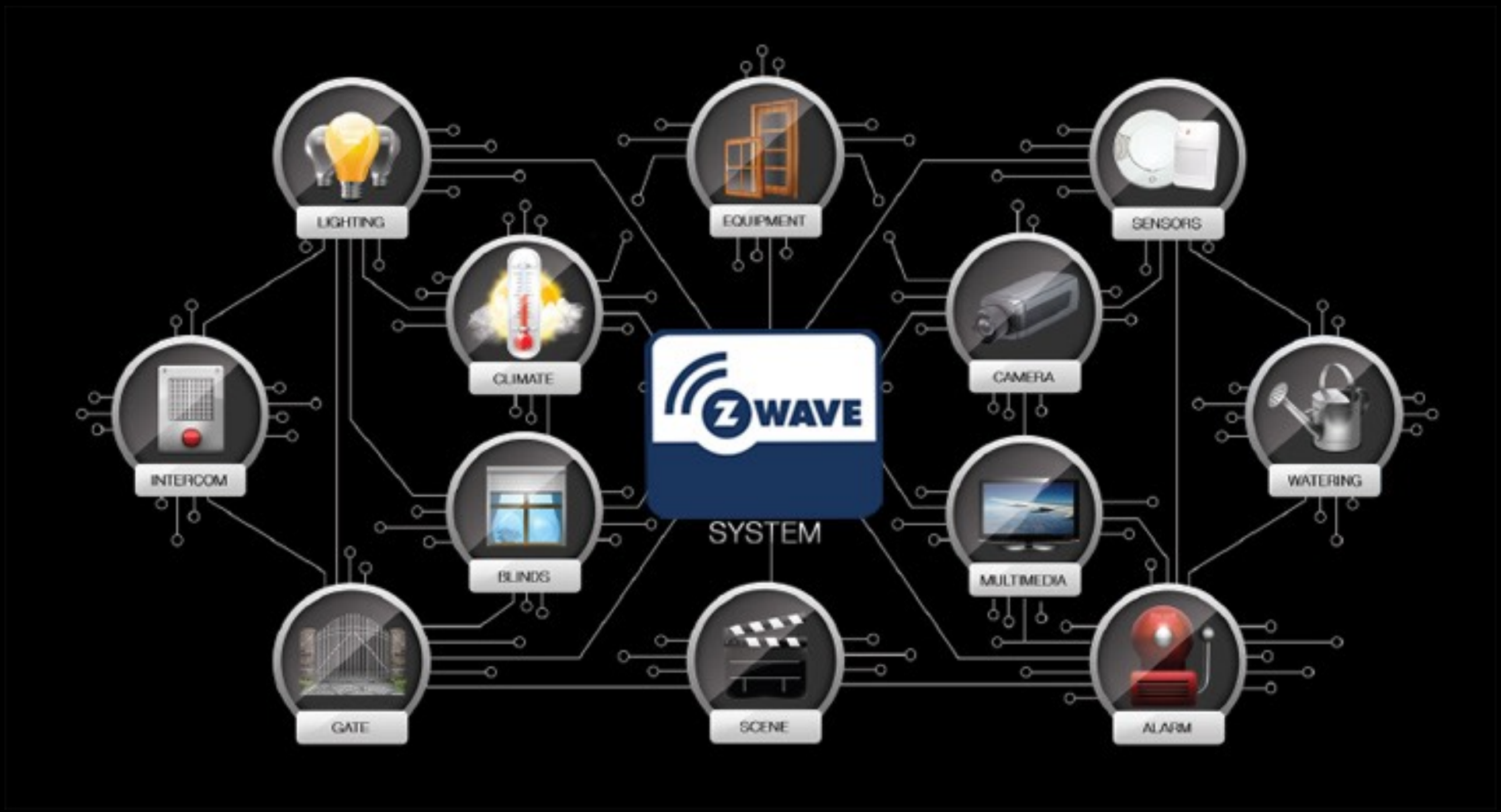
Connectivity

LoRaWAN

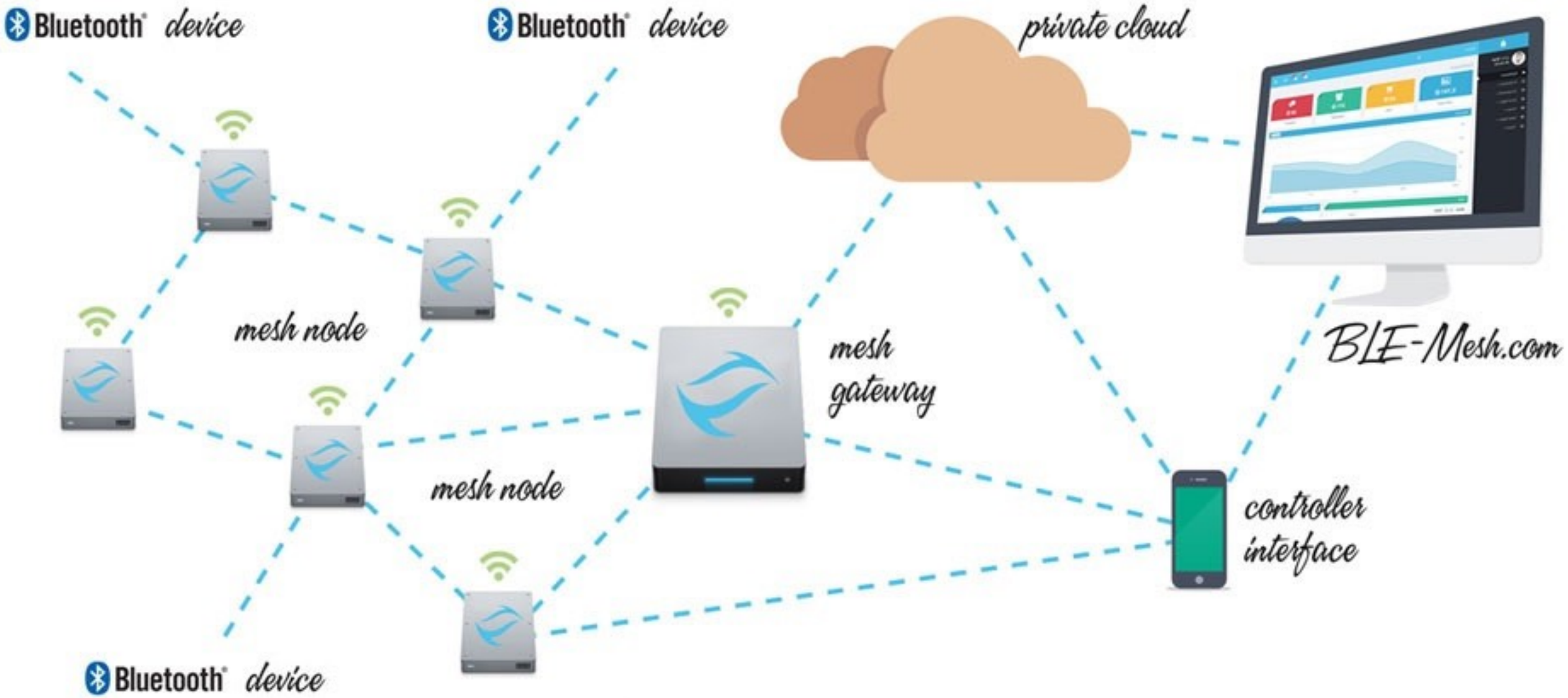
ZigBee


Home
Input
Control
Health
ZigBee®
Energy
Retail
Building
Light
Telecom

Z-Wave


LIGHTING
EQUIPMENT
SENSORS
CLIMATE
CAMERA
INTERCOM
WATERING
ZWAVE
SYSTEM
BLINDS
MULTIMEDIA
GATE
SCENE
ALARM

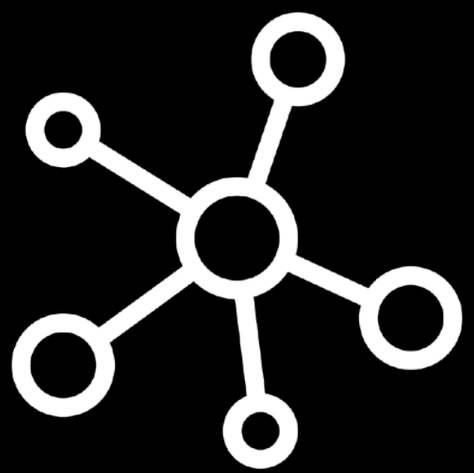
BLE Mesh


Bluetooth device
Bluetooth device
private cloud
mesh node
mesh node
mesh gateway
BLE-Mesh.com
controller interface
Bluetooth device

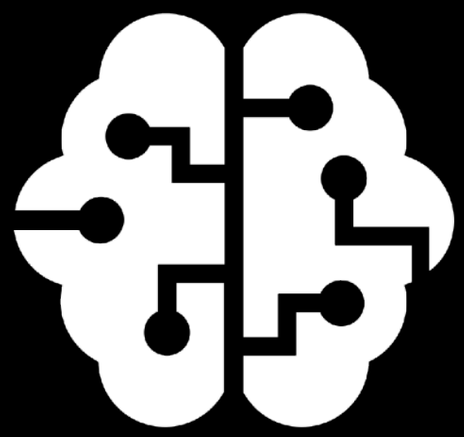
Business Model

1. **Subscription Model**
2. **Outcome-Based Model**
3. **Asset-Sharing Model**
4. **IoT Products as a Proxy to Sell Another Product**
5. **IoT Products as a Vehicle to Monetize Data**

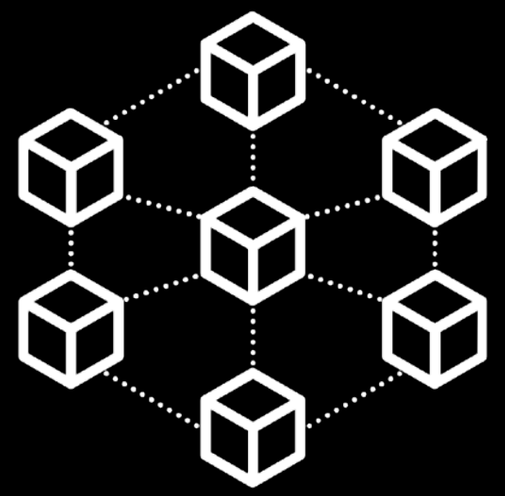
New Holy Trinity



IoT

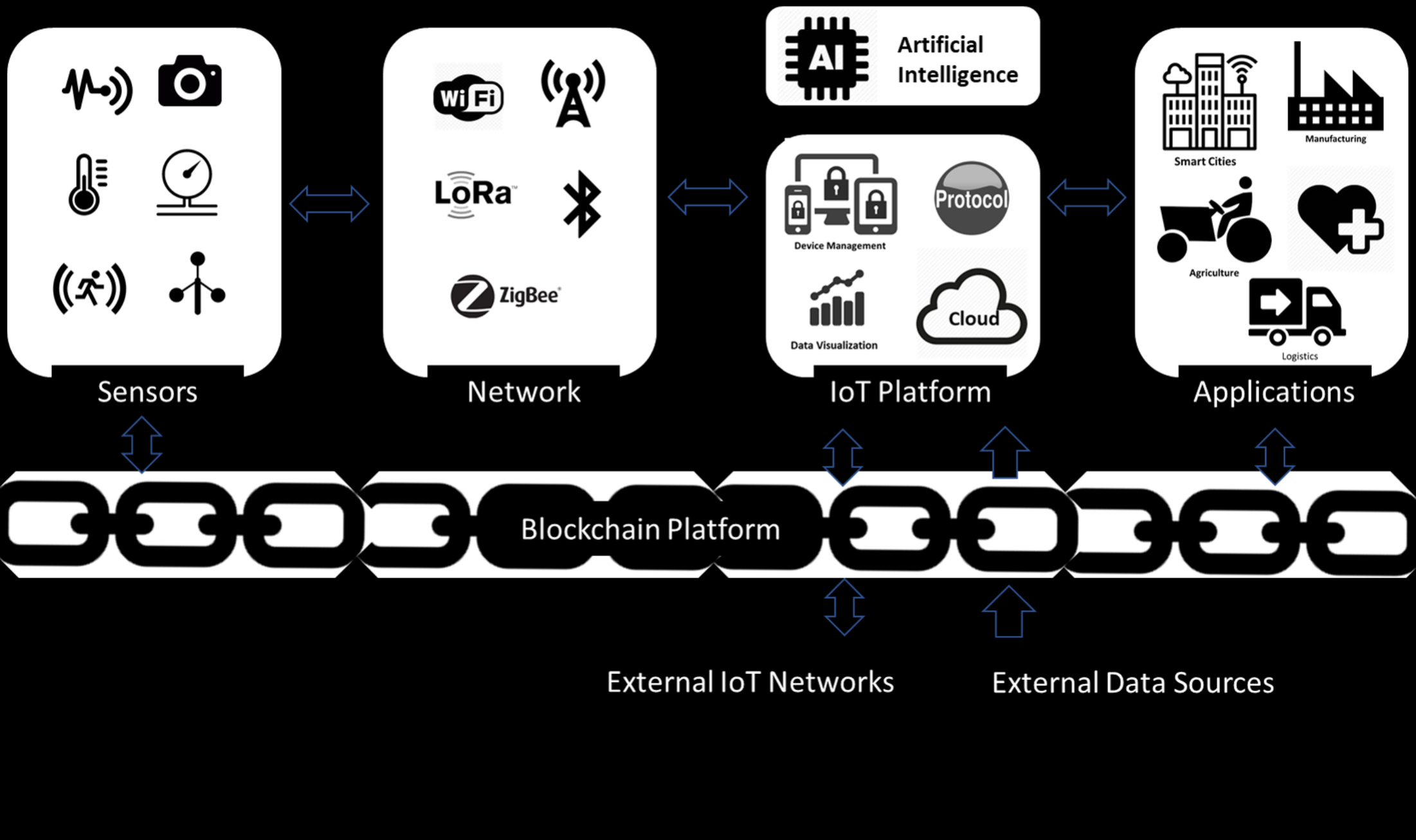


AI



Blockchain

Summary


AI
Artificial Intelligence
WiFi
LoRa
ZigBee
Device Management
Protocol
Data Visualization
Cloud
Smart Cities
Manufacturing
Agriculture
Logistics
Sensors
Network
IoT Platform
Applications
Blockchain Platform
External IoT Networks
External Data Sources

“In theory, theory and practice are the same. In practice, they are not.”

- Albert Einstein -

Questions?

Imanol Gómez

imanolgomez.net

yo@imanolgomez.net