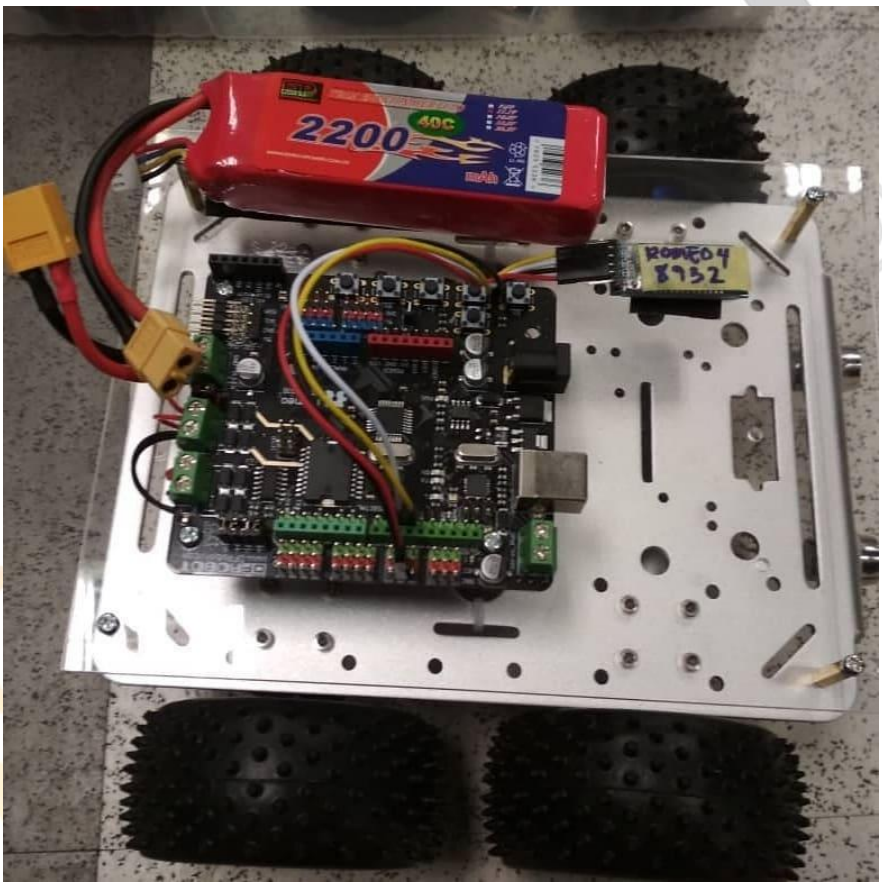
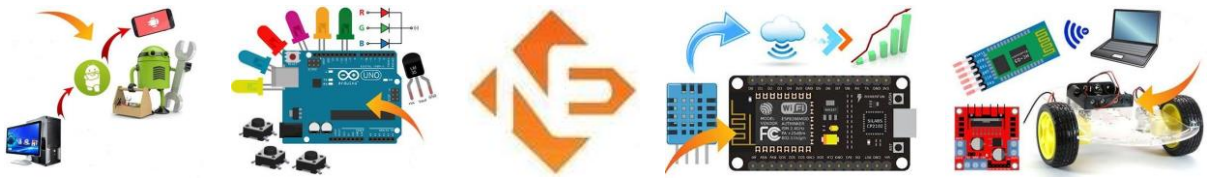
NADI
Eleczone solutions
Learn through electronic kits

Website :
http://nadieleczonesolutions.blogspot.com/
Facebook Page :
https://www.facebook.com/eleczone
Instagram :
http://instagram.com/eleczone

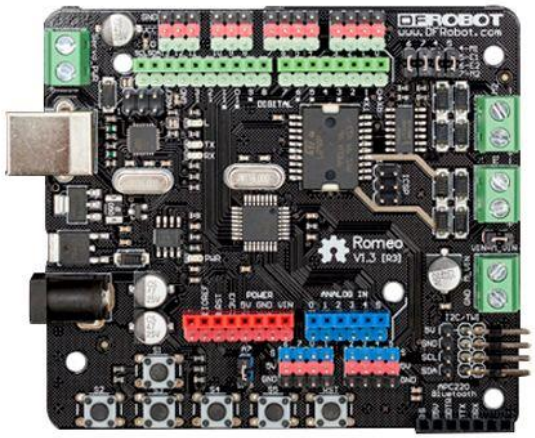
4 Wheeled Bluetooth Robot(4WBR)



Oleh NADI ELECZONE SOLUTIONS
AZARUL FAHMIN BIN AB HAMID
azarul@nadieleczone.com.my
013-4094377



Material List



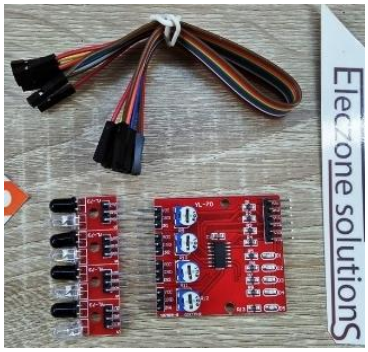
Arduino Romeo board

USB cable

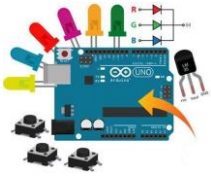
HC 05 Bluetooth

Jumper

4WD robot base



Line following module

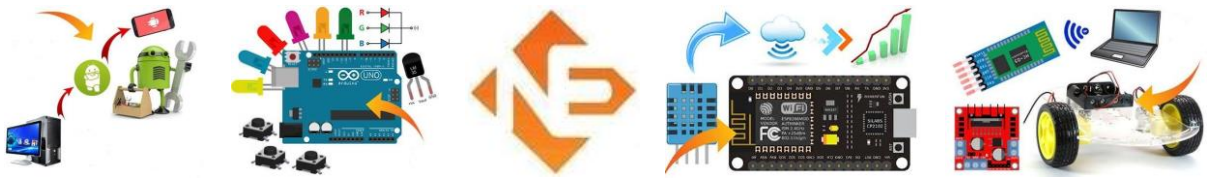




ARDUINO ROMEO Introduction

Romeo family is an All-in-One Robot control board especially designed for robotics applications from DFRobot. It benefits from the Arduino open source platform, it is supported by thousands of open source codes, and can easily be expanded with your Arduino shields. The integrated 2 way DC motor driver and wireless socket allows you to start your own robot project immediately without the need for an additional motor driver. Not just has the motor driver, Romeo is also designed to have extra power for servos which need more current.

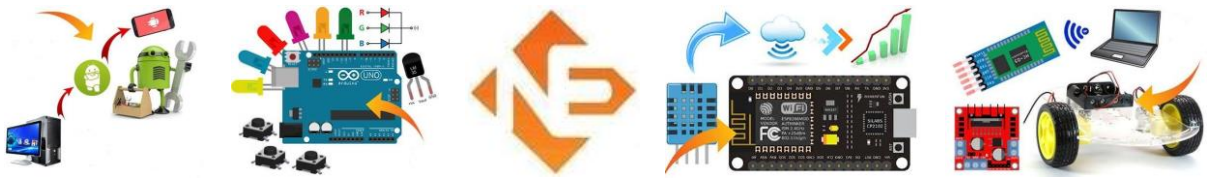
Romeo is also featured with DFRobot's standard 3 Pin-out designed and compatible with Gravity series sensors and actuators. Hundreds of sensors are now plug-play with Romeo. This is the first member in Romeo family that is born in 2009. It is not only the very first Arduino Robot Controller, but also the first Arduino-derived board in the market. The current version of Romeo is based on Arduino Uno.

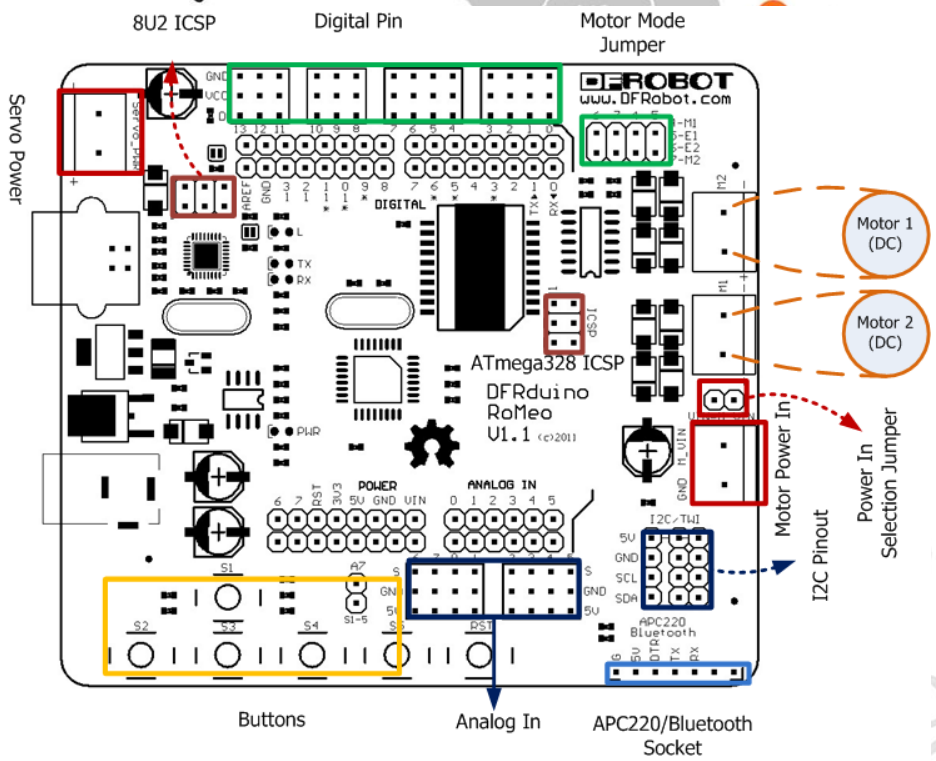
It has built in 2x2A DC motor drivers and socket for bluetooth / APC220 Radio Communication Module communications. The integrated io sensor pinout allows it connect hundreds of different Gravity compatible sensors and modules. It has servo connector which is a plug and play. It is the ideal controller to build your own robot.



Specification: -

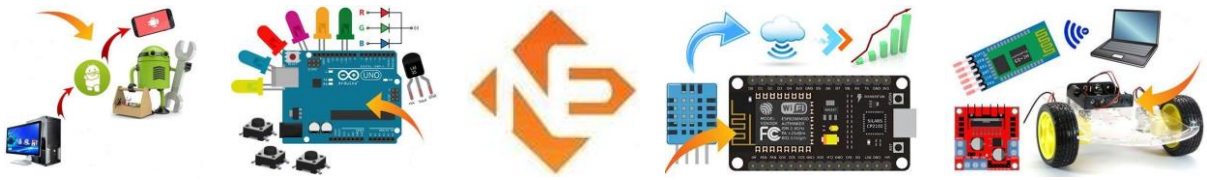
- Atmega 168/328
- 14 Channels Digital I/O
- 6 PWM Channels (Pin11,Pin10,Pin9,Pin6,Pin5,Pin3)
- 8 Channels 10-bit Analog I/O
- USB interface
- Auto sensing/switching power input
- ICSP header for direct program download
- Serial Interface TTL Level
- Support AREF
- Support Male and Female Pin Header
- Integrated sockets for APC220 RF Module and DF-Bluetooth Module
- Five I2C Interface Pin Sets
- Two way Motor Drive with 2A maximum current
- 5 key inputs
- DC Supply：USB Powered or External 7V~12V DC。
- DC Output：5V /3.3V DC and External Power Output
- Dimension：90x80mm





The picture shows the basic parts of the Arduino Romeo board equipped with the Atmega 168 and l298 motor driver IC.

- One Regulated Motor Power Input Terminal (6v to12v)
- One Unregulated Servo Power Input Terminal (you supply regulated 4v to 7.2v)
- One Servo input power selection jumper
- One Serial Interface Module Header for APC220/Bluetooth Module
- Two DC Motor Terminals – Handles motor current draw up to 2A, each terminal
- One I2C/TWI Port – SDA, SCL, 5V, GND
- One Analog Port with 8 analog inputs – Analog input 7 will be occupied when connecting "A7" jumper
- One General Purpose I/O Port with 13 I/O lines – 4,5,6,7 can be used to control motors
- One Reset Button
- Jumper bank to Enable/Disable Motor Control



Arduino IDE Software Installation

Step 1 : Go to website *https://www.arduino.cc/en/Main/Software*

Windows Installer
Windows ZIP file for non admin install

Windows app Get

Mac OS X 10.7 Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

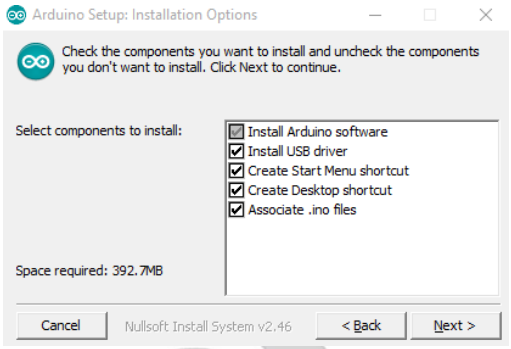
Release Notes
Source Code
Checksums (sha512)

Step 2 : Click *Download*, and choose *Windows Installer* to be download.

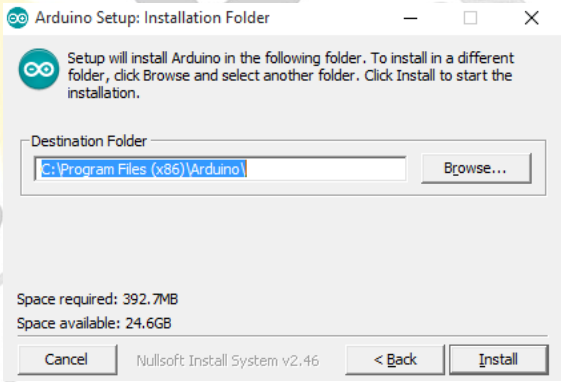
$25 $50 OTHER

JUST DOWNLOAD CONTRIBUTE & DOWNLOAD

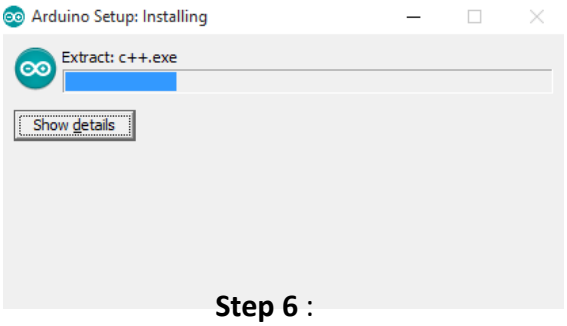
Step 3 : In the Download field, you can select JUST DOWNLOAD or CONTRIBUTE & DOWNLOAD to continue downloading the software.



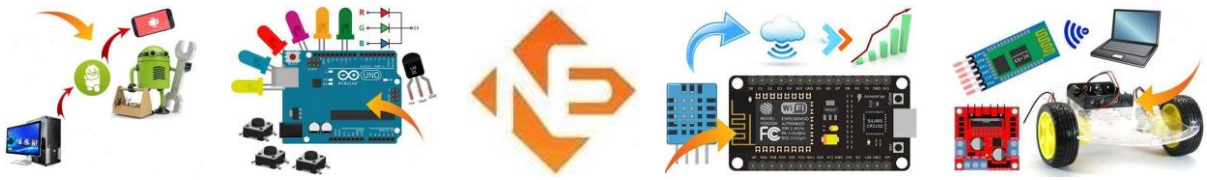
Step 4 :
After download, run the Arduino IDE software installer and click Next.



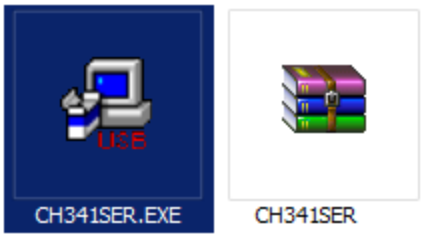
Step 5 : Click Install to start the software installation process.



Step 6 :
Wait for the Extract process to finish all files and software installation successfully.

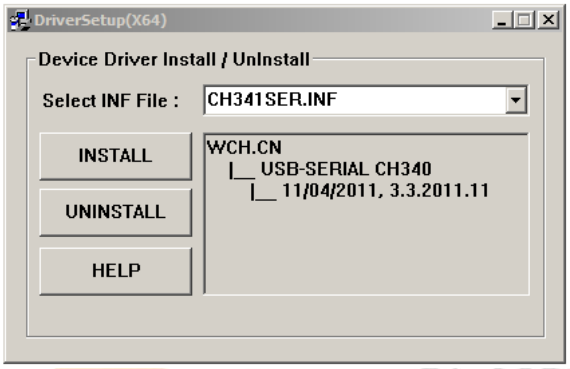


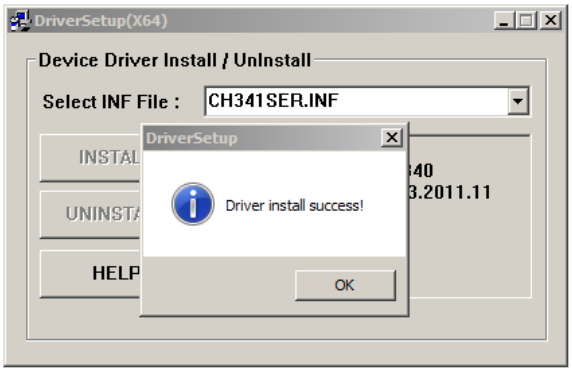
USB converter Driver for Arduino Romeo



Step 1 : Unzip file CH341SER

Step 2: Double-click on the CH341SER.EXE file to begin installing Arduino Romeo board driver.

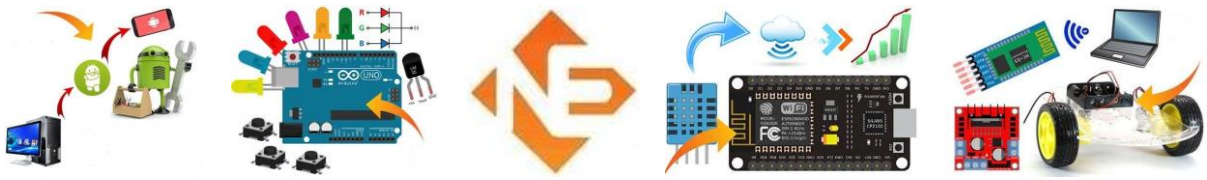




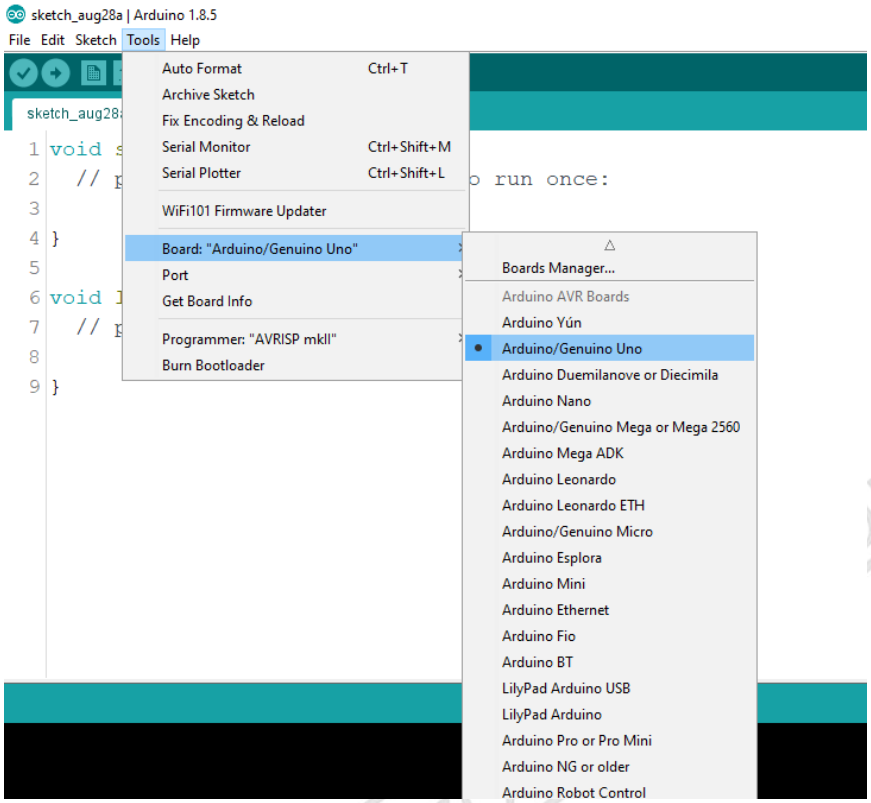
Step 3: Click on the INSTALL button to begin installing Arduino Uno board driver.

Step 4: Click the OK button after seeing the "Driver install success!" Message. This indicates that the Arduino Uno board driver is successfully installed.

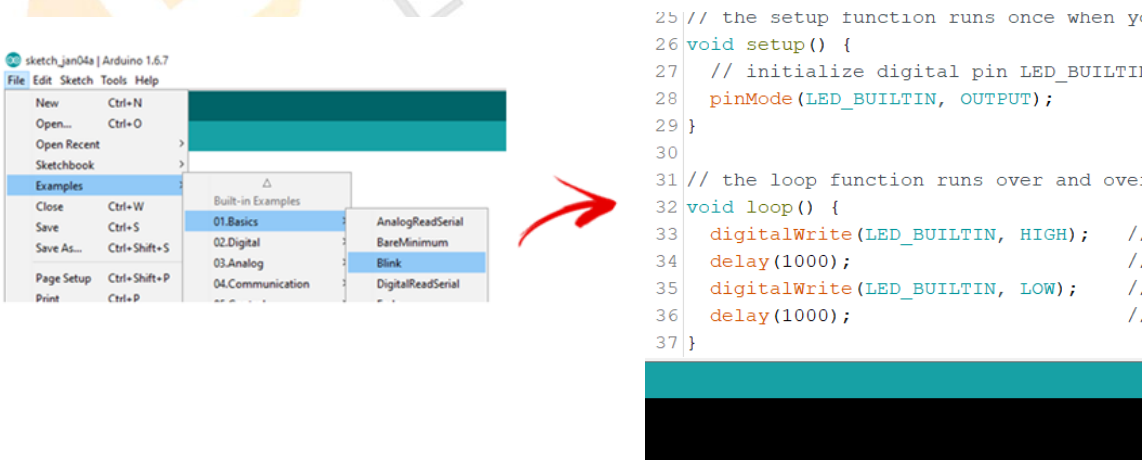
* Refer to Nadi Eleczone Youtube video for CH341 driver installation at https://www.youtube.com/user/Cakzone



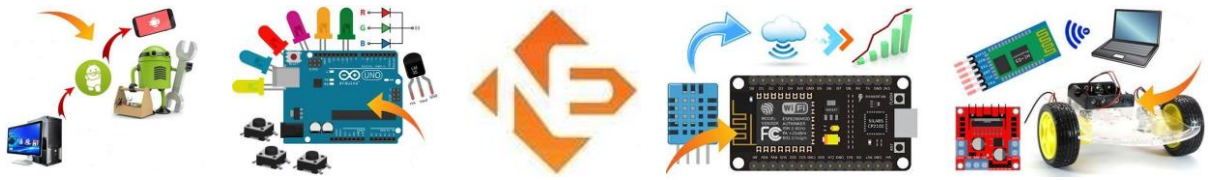
Upload Your First Program

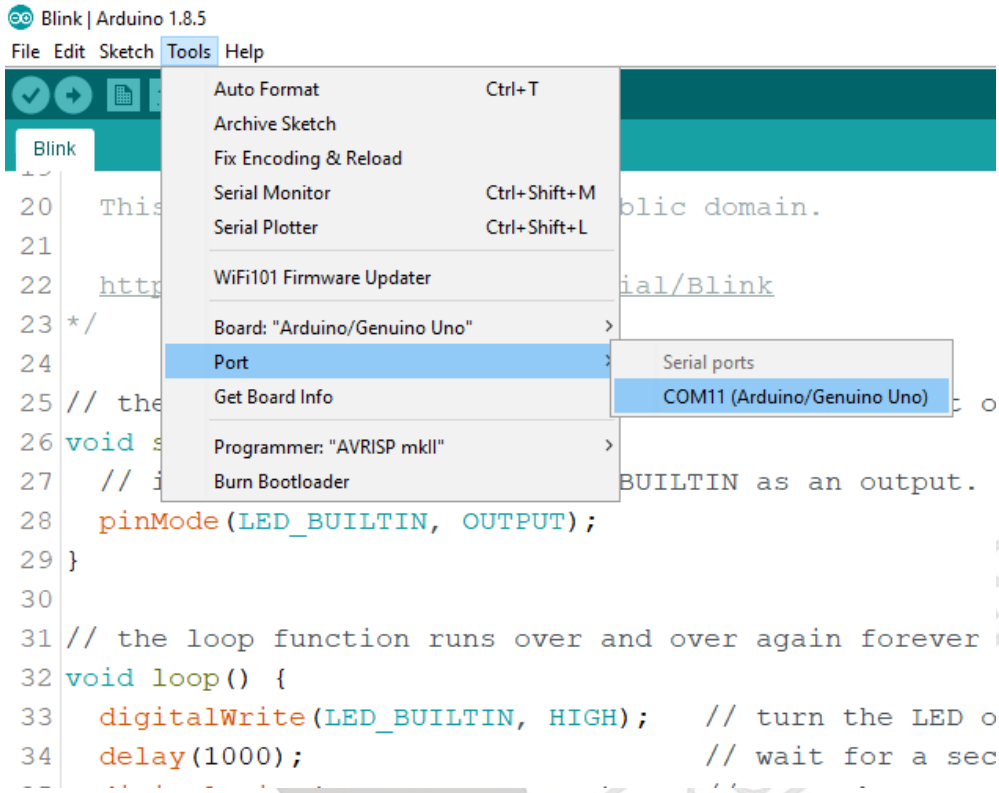


Step 1: Set the Arduino Romeo board connection with Arduino IDE, click on Tools >> Boards >> and choose Arduino/Genuino Uno

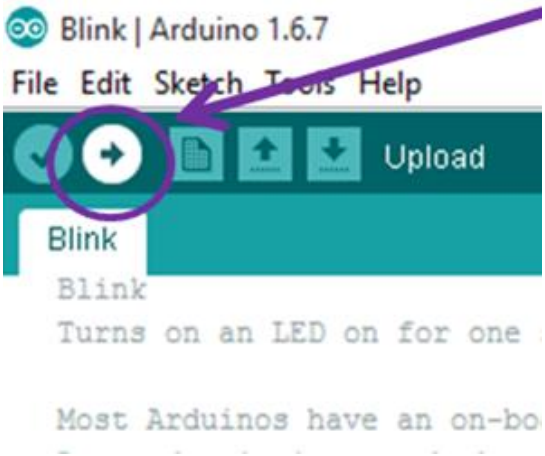


Step 2: Open an existing code example from Arduino IDE software by clicking File >> Examples >> 01 Basics >> Blink. And change the code like the red circle on the image.

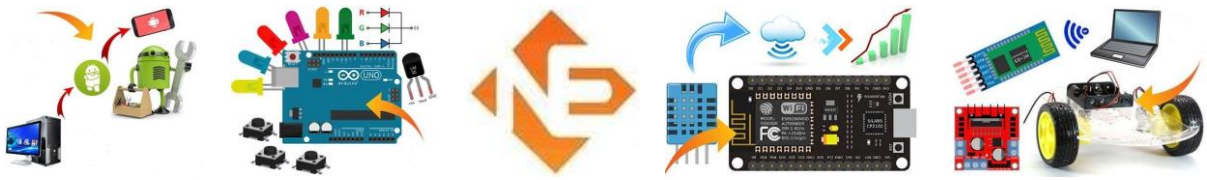




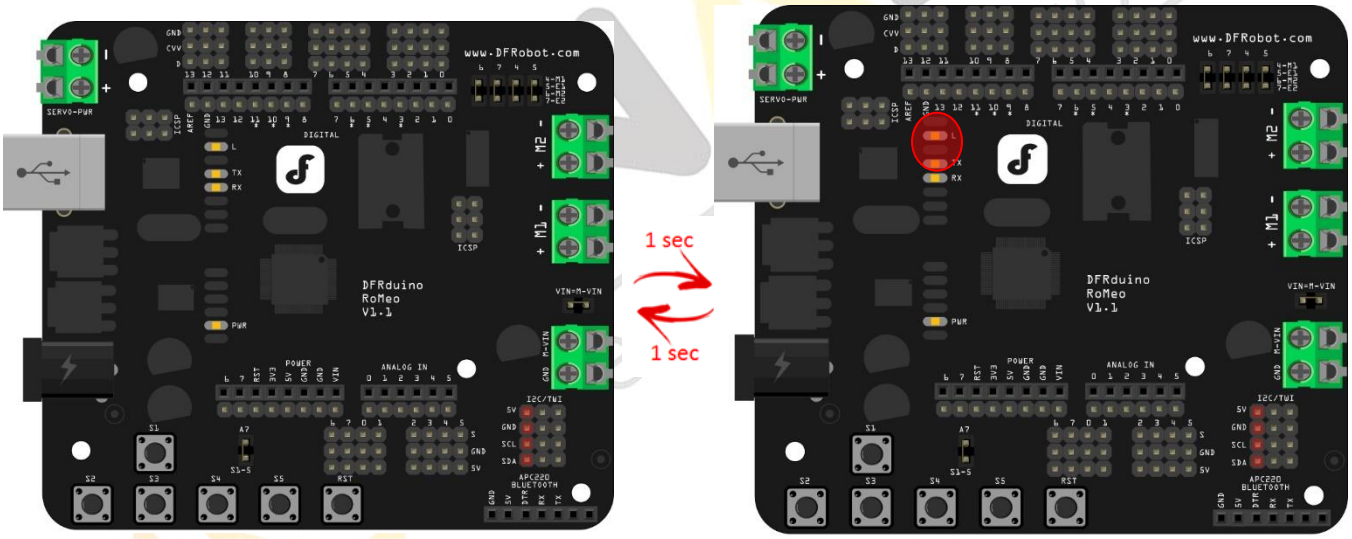
Step 3: Set Com port number Arduino Romeo board connection with Arduino IDE, click on Tools >> Ports >> and select COMX



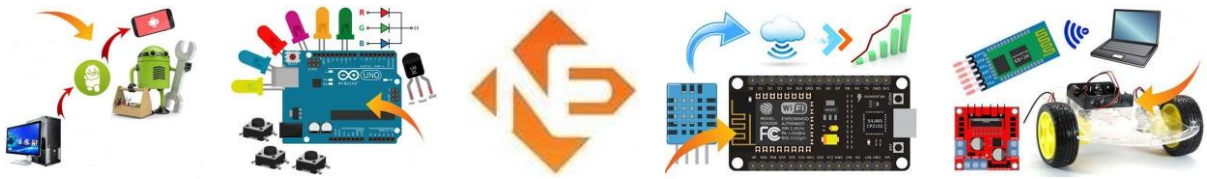
Step 4: Upload the code to the Arduino Romeo board by clicking on the Upload button.



Step 5: Done Uploading message displayed indicating code successfully uploaded to Arduino Romeo board.



Step 6: Visible lights on the Arduino Romeo board start blinking with a time interval for 1 second.



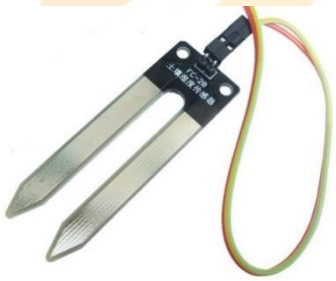
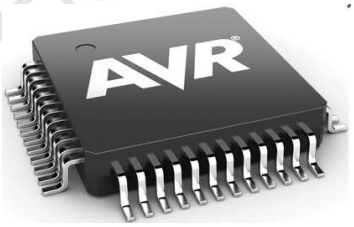
Input and *Output Concept*

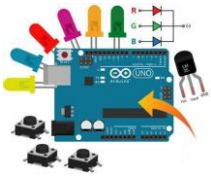
INPUT — Electronic components that produce voltage or current signals and can be used in Embedded system.

PROCESS — The control device will process the signal from the input as a reference in producing the output.

OUTPUT — The results of data manipulated by the processing system

Example of system
Automatic Groundwater System

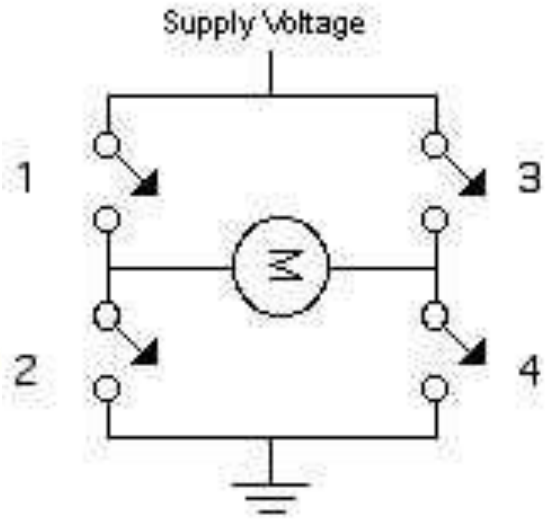
INPUT	PROCESS	OUTPUT
		
The soil moisture sensors will be immersed in the soil and emit a signal when the soil is dry.	The microcontroller chip will process the signal from the sensor and then control the water pipe for watering.	The water pipe will be activated when the soil is dry and requires drainage.

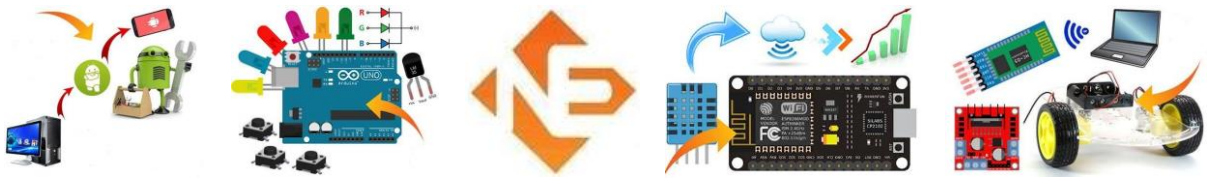

Project 1 – Controlling DC Motor : This project is to identify how to control the DC motor by using the instructions from Arduino Romeo. To make Arduino Romeo can control DC motor we need a DC motor driver module first. There are two easily controllable parameters of a DC motor, direction and speed. To control the direction, the polarity of the motor is reversed. To control the speed, the input voltage is varied using pulsewidth modulation.

Direction Control

To control a DC motor from a microcontroller, you use switching arrangement known as an H bridge. It looks like this:

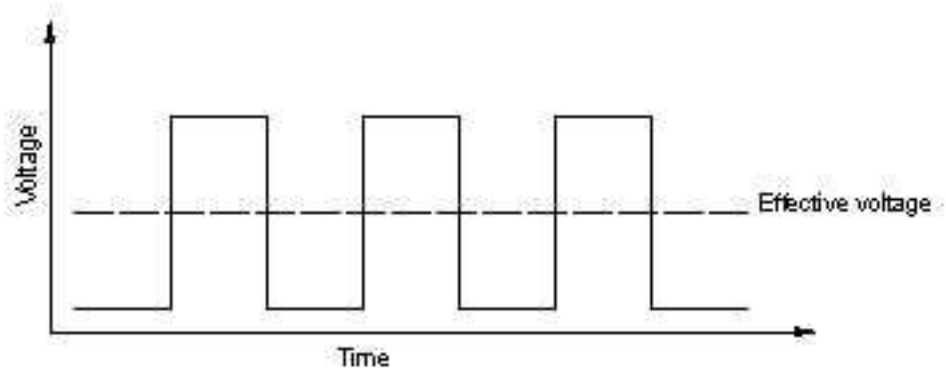


When switches 1 and 4 are closed and 2 and 3 are open, voltage flows from the supply to 1 to the motor to 4 to ground. When 2 and 3 are closed and 1 and 4 are open, polarity is reversed, and voltage flows from the supply to 3 to the motor to 2 to ground.

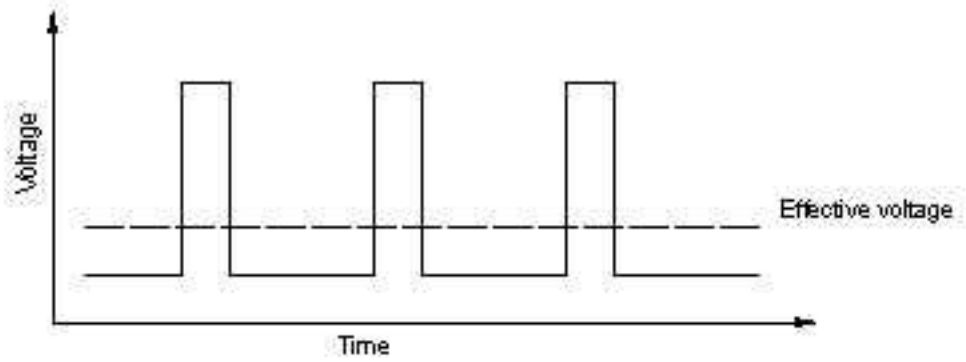


Speed

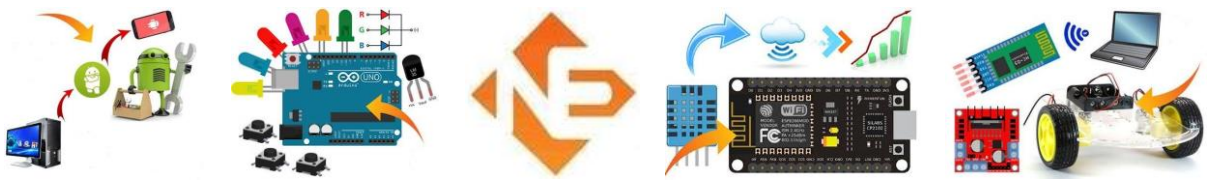
A DC motor's speed is proportional to the supplied voltage. If the voltage drops too far, the motor won't get enough power to turn, but within a certain range, usually 50% of the rated voltage, the motor will run at varying speeds. The most effective way to adjust the speed is by using pulsewidth modulation. This means that you pulse the motor on and off at varying rates, to simulate a voltage. Here are some examples of puleswidths and the voltages they would simulate:



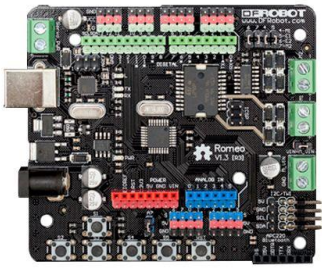
When the time that the voltage is high (the duty cycle) is half the total time in question, the effective voltage is about half the total voltage.



When the duty cycle is reduced to one quarter of the total time, the effective voltage is about one quarter of the total voltage.



List of needed material



Arduino
Romeo board

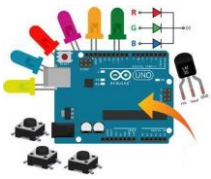
Jumper

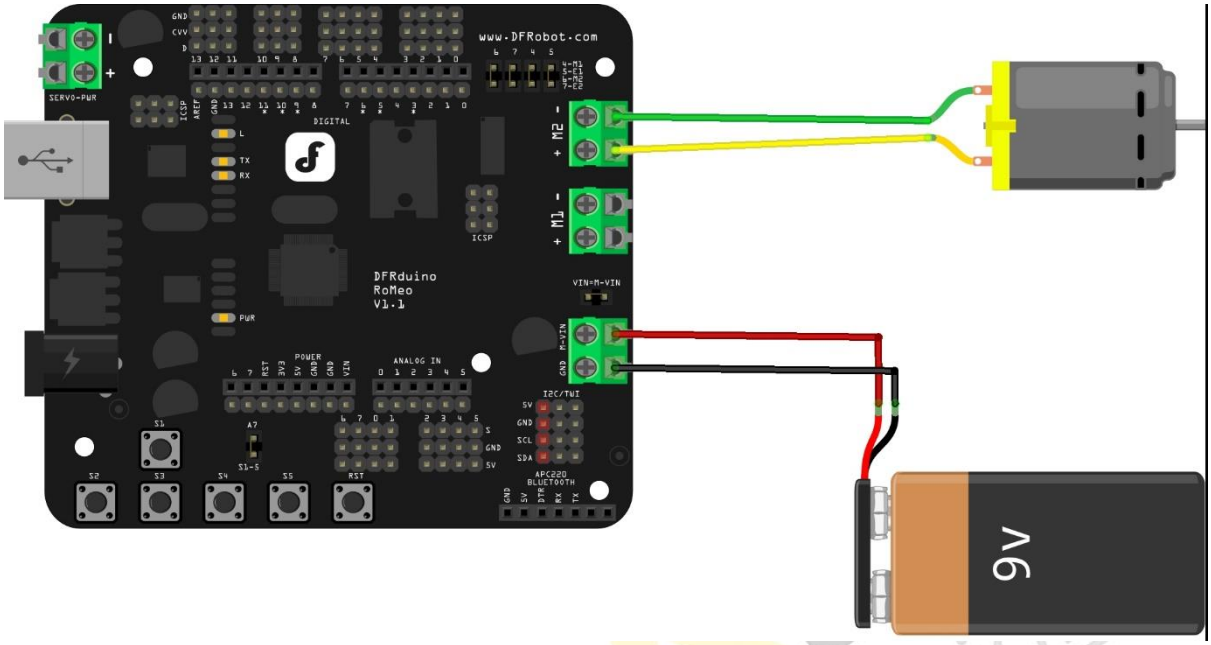
4WD Robot
Base

LiPo Baterry

DC motor

Build the Circuit

Learn the basic code: The basic code below is the code structure that should always be present in order for a code to be implemented.

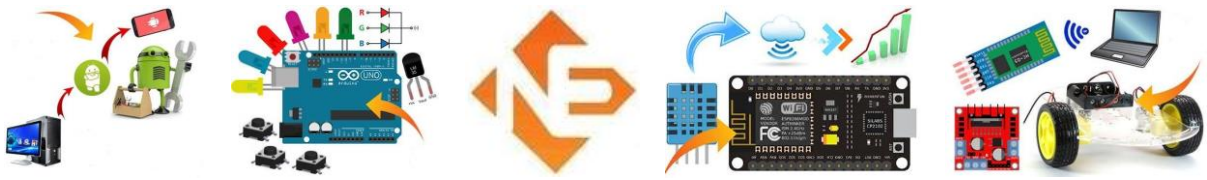
Basic of code

```
void setup()
{

}

void loop()
{


}
```

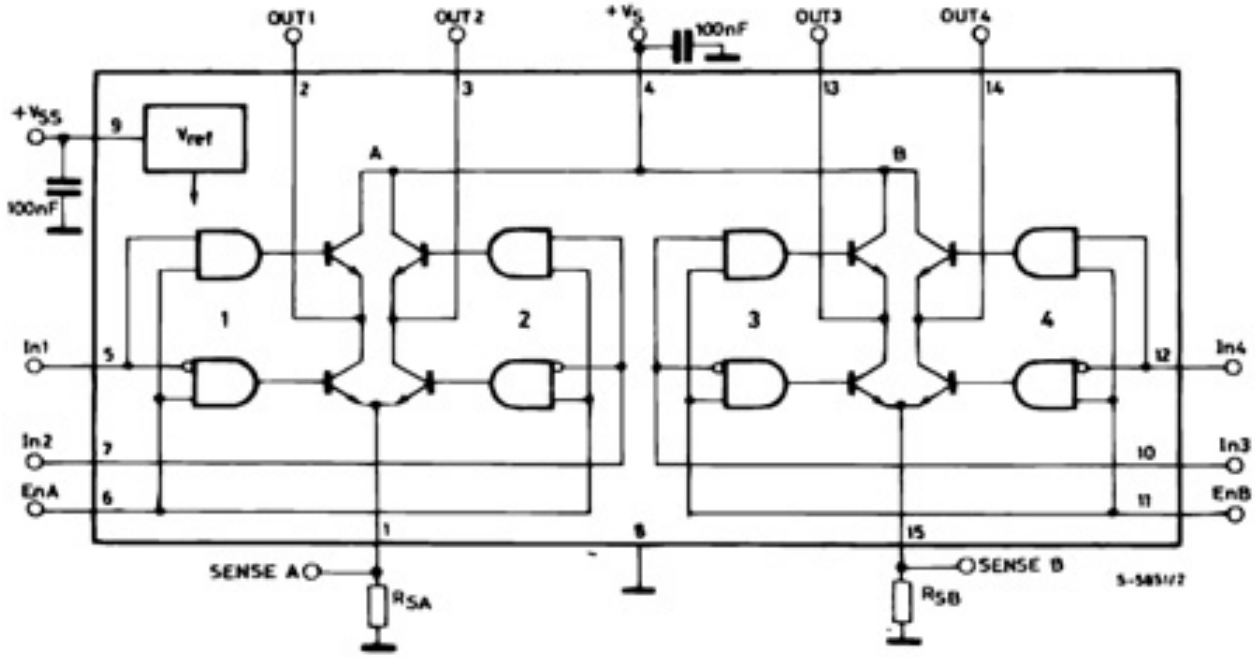
In the setup bracket, the code will only run once from top to bottom when the controller is power up.

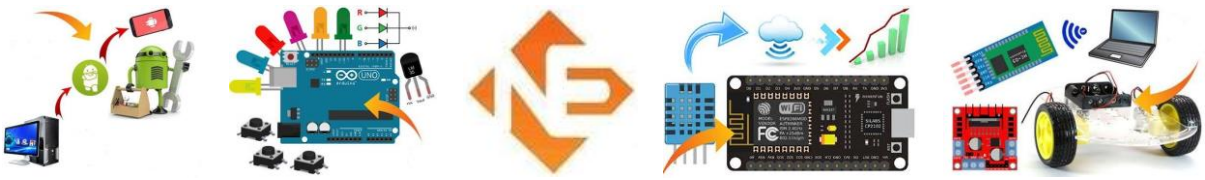
Within the loop bracket, the code will be executed from top to bottom and will be repeated until controller is shut down.

L298 Specification



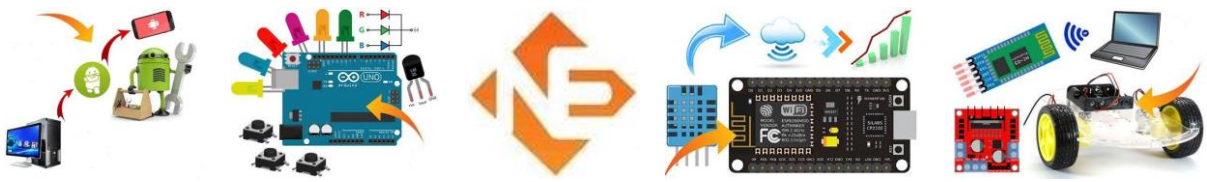
The L298 is an integrated monolithic circuit in a 15- lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage





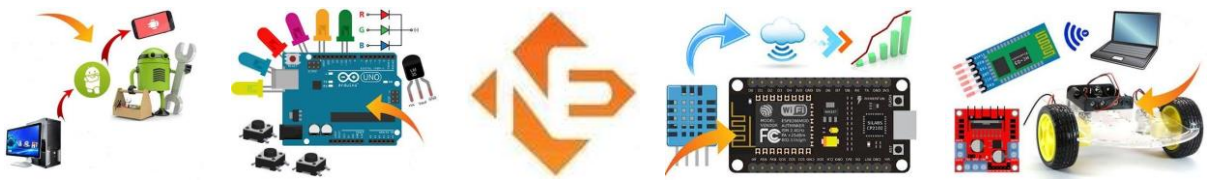
Characteristic: -

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA
$V_{CEsat\ (H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat\ (L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V



Programming Code

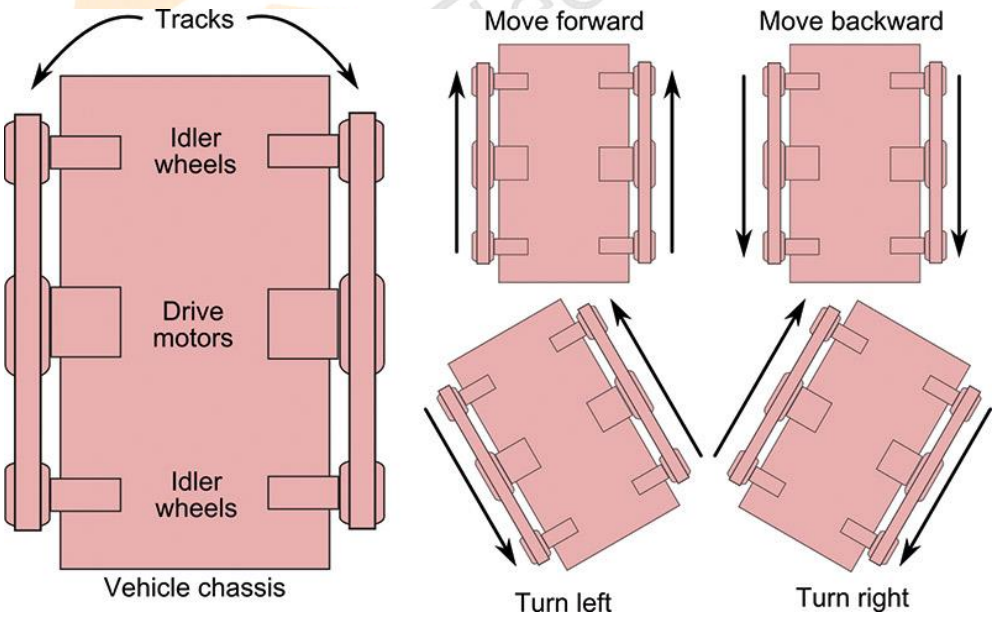
```
#define M1 4
#define E1 5
void setup() {
  // put your setup code here, to run once:
  pinMode(M1, OUTPUT);
  pinMode(E1, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(E1, HIGH);
  digitalWrite(M1, HIGH);
  delay(2000);
  digitalWrite(E1, LOW);
  digitalWrite(M1, LOW);
  delay(1000);
  digitalWrite(E1, HIGH);
  digitalWrite(M1, LOW);
  delay(2000);
  digitalWrite(E1, LOW);
  digitalWrite(M1, LOW);
  delay(1000);
}
```

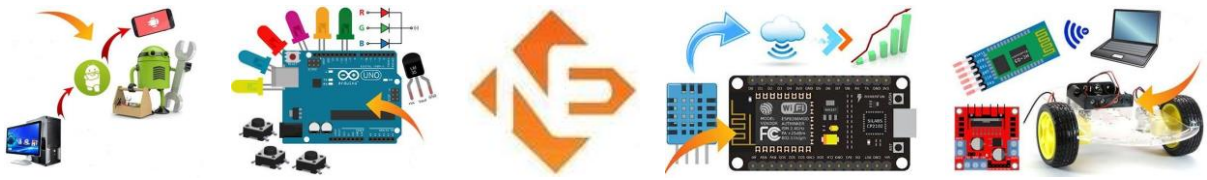


Project 2 – Controlling Robot Direction

: This project is to identify how to control the Robot direction by using the instructions from Arduino Romeo. Wheeled robots are robots that navigate around the ground using motorized wheels to propel themselves. This design is simpler than using treads or legs and by using wheels they are easier to design, build, and program for movement in flat, not-so-rugged terrain. They are also more well controlled than other types of robots. Disadvantages of wheeled robots are that they cannot navigate well over obstacles, such as rocky terrain, sharp declines, or areas with low friction. Wheeled robots are most popular among the consumer market, their differential steering provides low cost and simplicity. Robots can have any number of wheels, but three wheels are sufficient for static and dynamic balance. Additional wheels can add to balance; however, additional mechanisms will be required to keep all the wheels in the ground, when the terrain is not flat. For this robot direction instruction, we are using Serial Monitor of Arduino IDE as an input. We will send a capital character such as A, B, C and D to make the robot move forward, backward, left, right and stop.

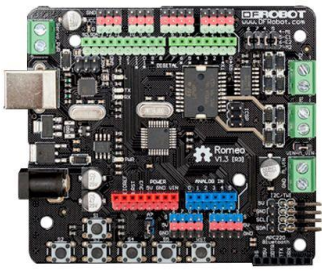
Direction Control





To move forward the left DC motor will rotate counter-clockwise (CCW) and the right motor will rotate clockwise (CW). Then, to move backward the left DC motor will rotate CW and the right motor will rotate CCW. After that, to move left both of DC motor will rotate CW. Lastly, to move left both of DC motor will rotate CCW

List of needed material



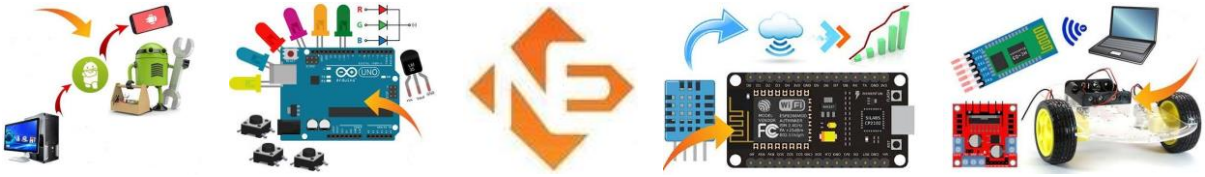
Arduino Romeo board

Jumper

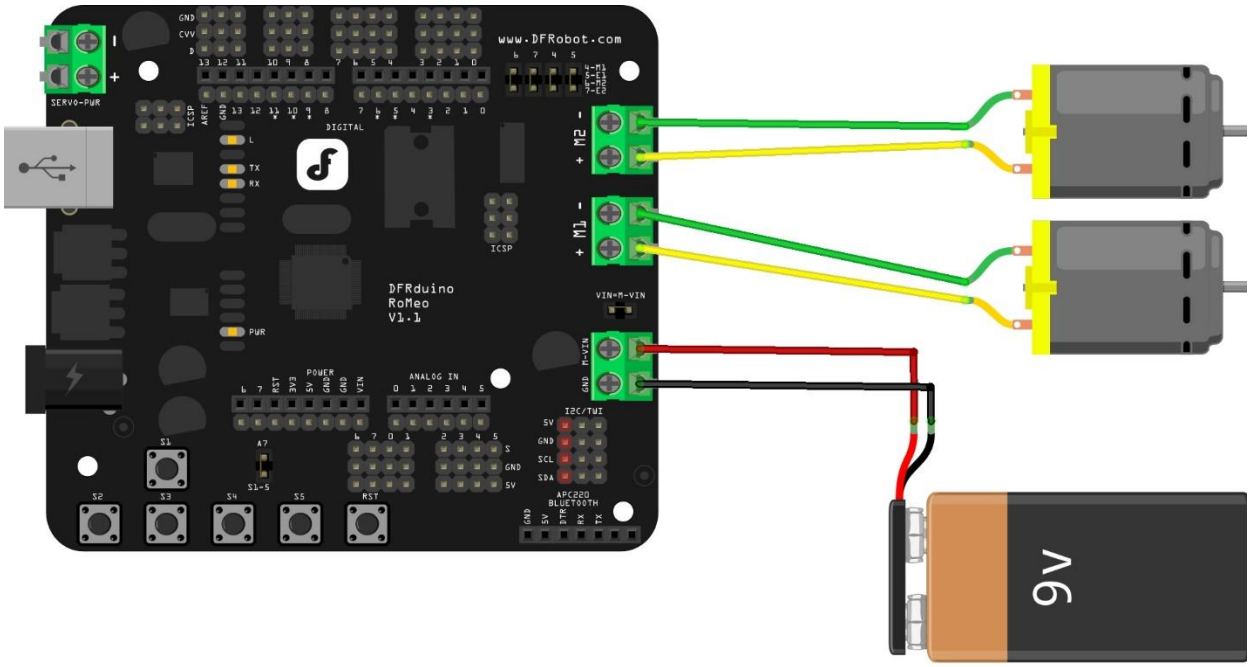
4WD Robot Base

LiPo Baterry

DC motor

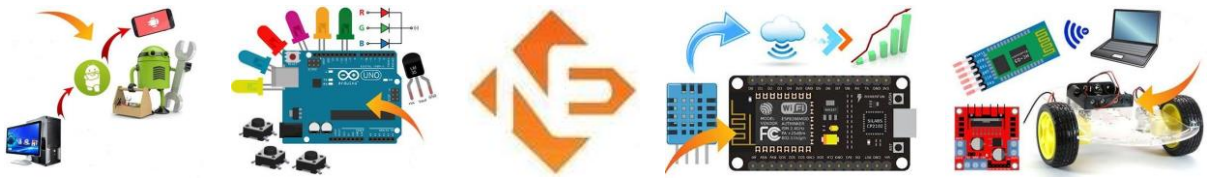


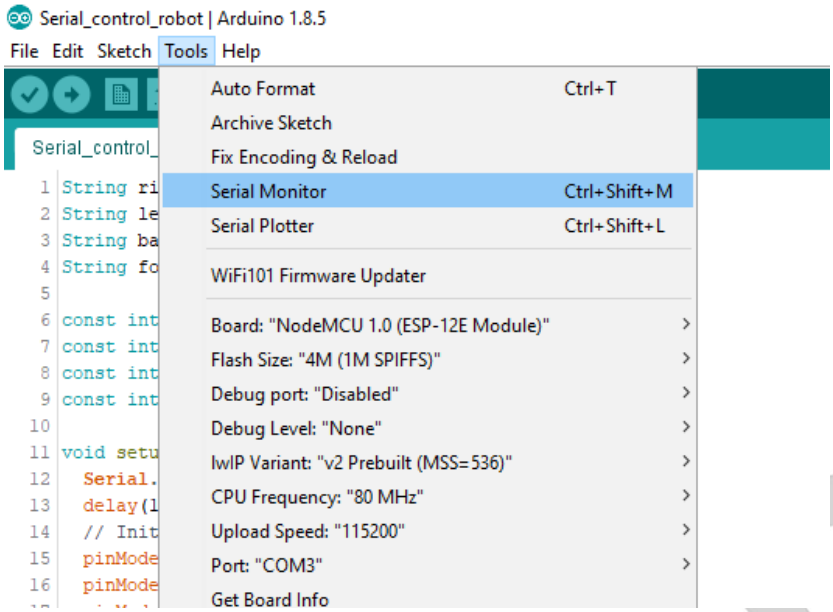
Build the Circuit



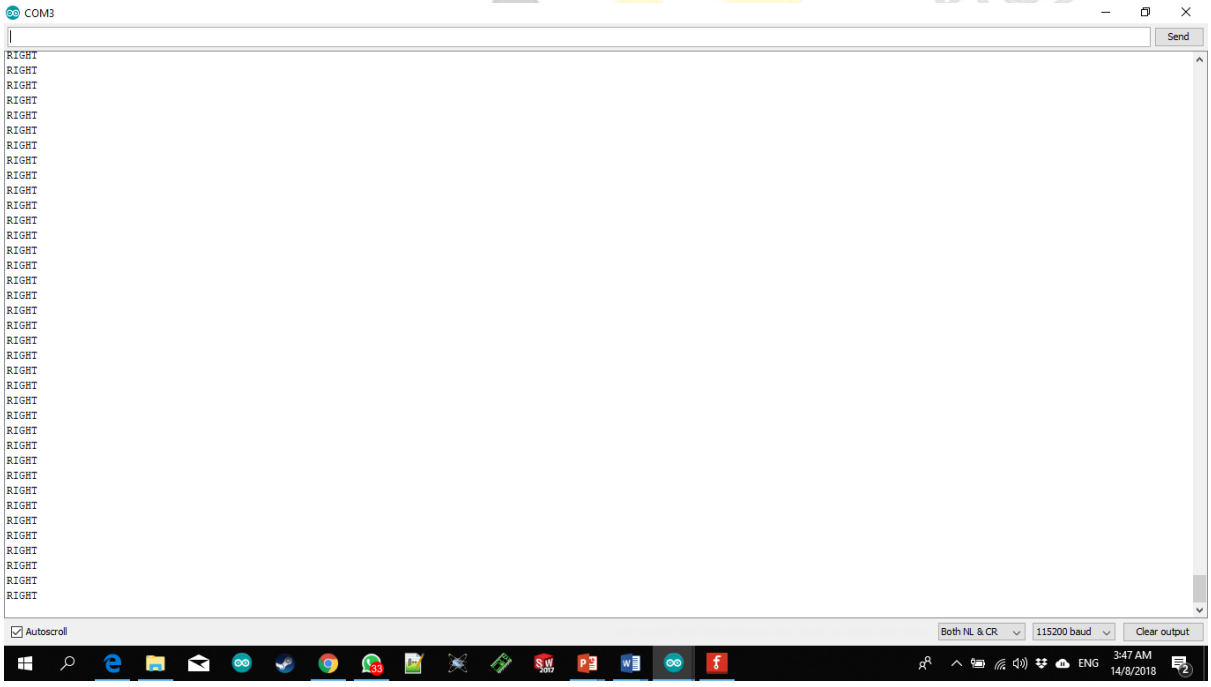
Arduino Serial Monitor

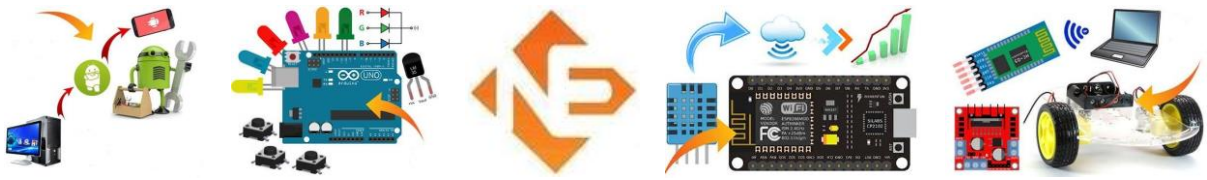
The Arduino IDE has a feature that can be a great help in debugging sketches or controlling Arduino from your computer's keyboard. **The Serial Monitor** is a separate pop-up window that acts as a separate terminal that communicates by receiving and sending Serial Data. Serial Data is sent over a single wire (but usually travels over USB in our case) and consists of a series of 1's and 0's sent over the wire. Data can be sent in both directions. Recognizing the Arduino IDE serial monitor, this function lets you see the value given by Arduino Romeo board to the computer via USB medium and send data from computer to Arduino Romeo board. After clicking on the upload button and receiving the Done Uploading message, you need to click on Tools >> Serial Monitor. Make sure the baudrate on the program and Serial Monitor are the same.





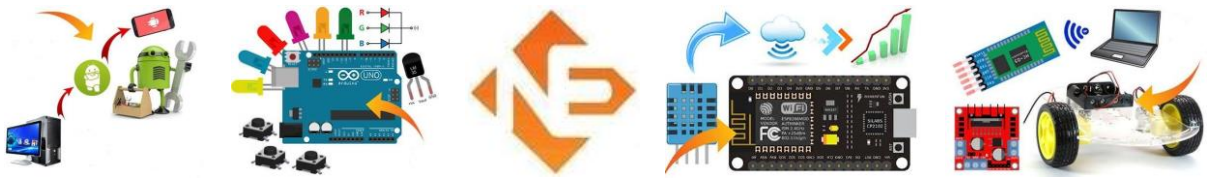
Choosing Serial Monitor and you will get pop up window like the below images.



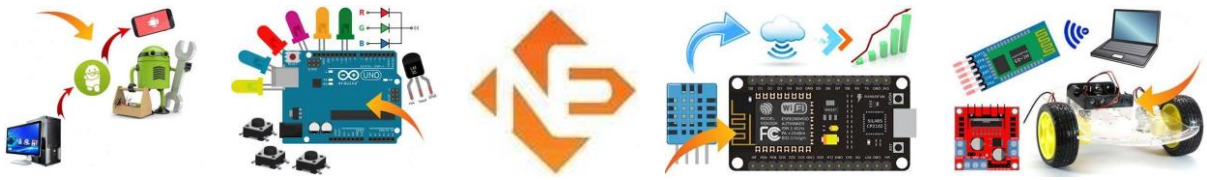


Programming Code

```
#define M1 4
#define E1 5
#define M2 7
#define E2 6

void setup() {
  Serial.begin(9600);
  pinMode(M1, OUTPUT);
  pinMode(E1, OUTPUT);
  pinMode(M2, OUTPUT);
  pinMode(E2, OUTPUT);
}

void loop() {
char in = Serial.read();
Serial.println(in);
if(in == 'A'){
  digitalWrite(E1, HIGH);
  digitalWrite(M1, HIGH);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, HIGH);
  delay(100);
}
if(in == 'B'){
  digitalWrite(E1, HIGH);
  digitalWrite(M1, LOW);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, HIGH);
  delay(100);
}
if(in == 'C'){
  digitalWrite(E1, HIGH);
  digitalWrite(M1, HIGH);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, LOW);
  delay(100);
```

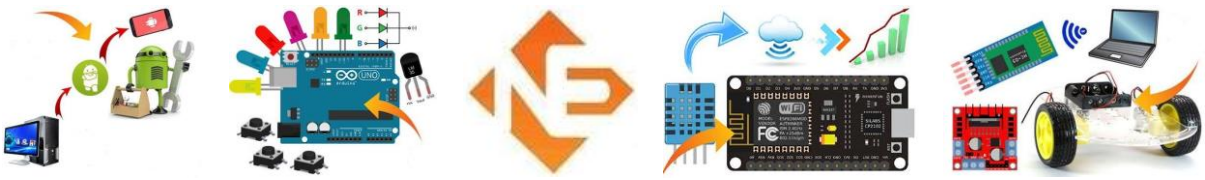
```
}
if(in == 'D'){
   digitalWrite(E1, HIGH);
  digitalWrite(M1, LOW);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, LOW);
  delay(100);
}
if(in == 'Z'){
   digitalWrite(E1, LOW);
  digitalWrite(M1, LOW);
  digitalWrite(E2, LOW);
  digitalWrite(M2, LOW);
  delay(100);
}
}
```

Project 3 – Bluetooth Connectivity: We have learnt about wired serial communication. For this part, we will learn about wireless serial communication with HC-05 and how to use it. At the end of this lesson, we will learn about how to control a robot using our smartphone. But first, we need to learn about bluetooth module HC-05. Before that, for your information Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz) from fixed and mobile devices, and building personal area networks (PANs). Invented by Dutch electrical engineer Jaap Haartsen, working for telecom vendor Ericsson in 1994, it was originally conceived as a wireless alternative to RS-232 data cables. Bluetooth is managed by the Bluetooth Special Interest Group (SIG), which has more than 30,000 member companies in the areas of telecommunication, computing, networking, and consumer electronics. The IEEE standardized Bluetooth as IEEE 802.15.1, but no longer maintains the standard. The Bluetooth SIG oversees development of the specification, manages the qualification program, and protects the trademarks. A manufacturer must meet Bluetooth SIG standards to market it as a Bluetooth device. A network of patents apply to the technology, which are licensed to individual qualifying devices.

Bluetooth HC-05

HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module,designed for transparent wireless serial connection setup.The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication.This serial port bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband.



Foot Note – Pairing and connecting with Bluetooth module is different..!!

KEY or EN – Used for entering setting mode (Using HIGH bit)

VCC – Power Supply for the HC-05 (3.6 V-5.0 V)

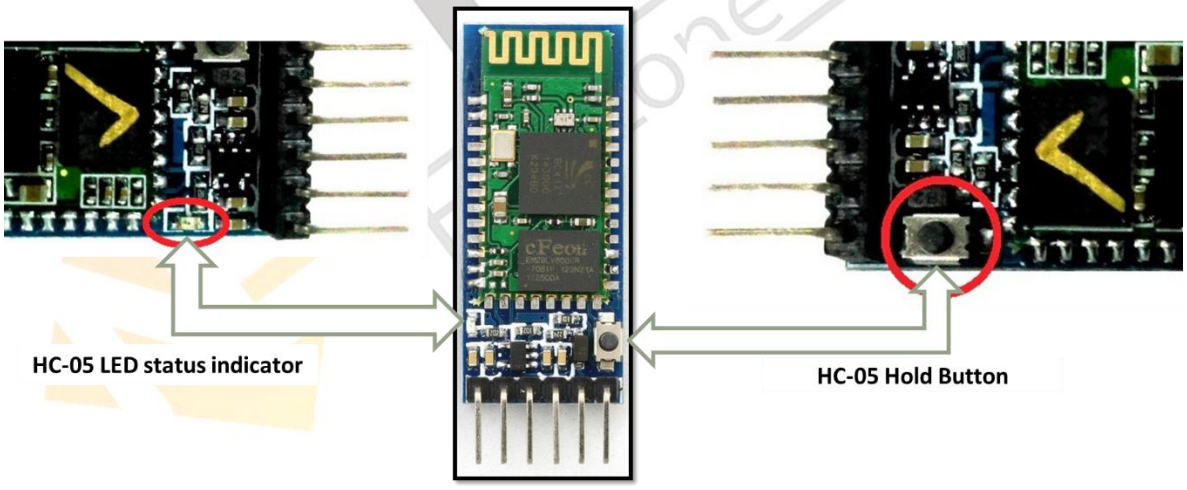
GND – Ground

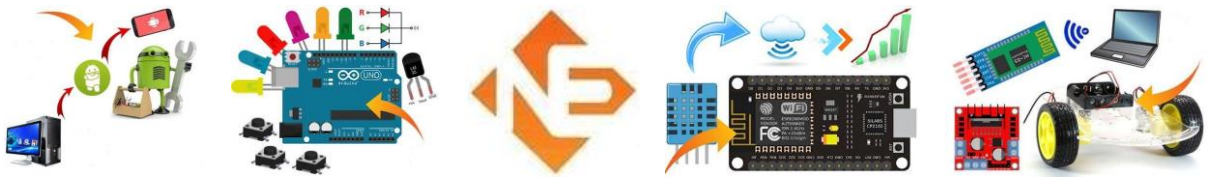
TXD – Transmitting part for HC-05

RXD – Receiving part for HC-05

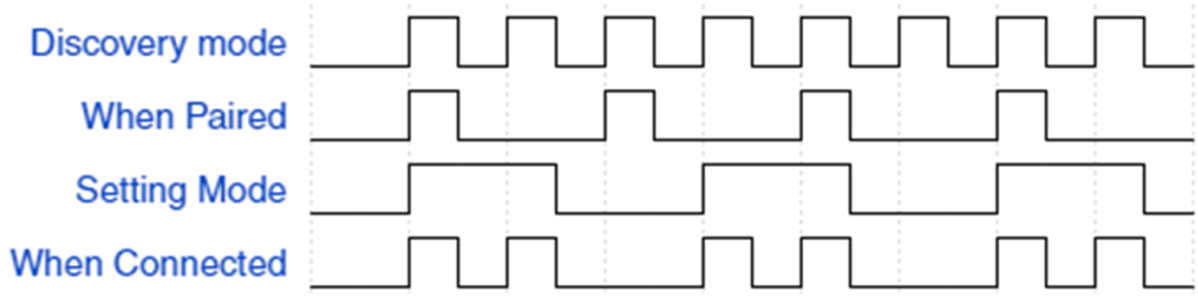
STATE/LED – Indicator for connection status

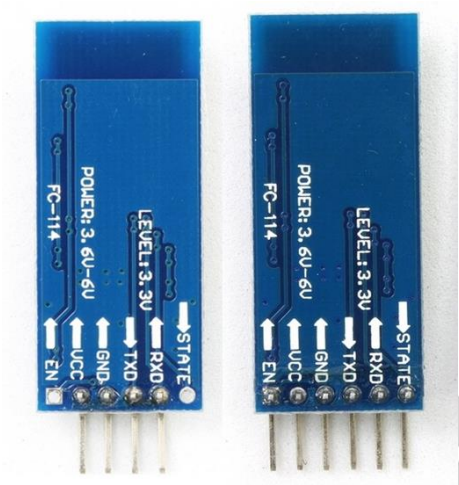
LOW for disconnected and HIGH for connected





HC-05 LED status indicator





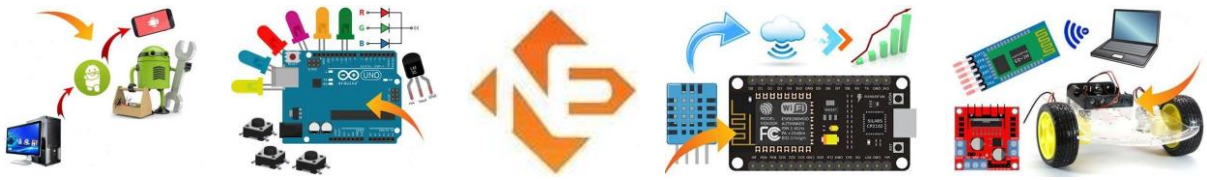
Different between HC-05 and HC-06

- HC-06 just have 4 pins (VCC, GND, TX, RX). No KEY/EN and STATE pin.
- It means that HC-06 cannot enter setting mode.
- It also means that we cannot change the parameter of the bluetooth.
- Bluetooth module is in slave mode by default.
- That makes the HC-06 just stay in slave mode.
- Make sure you know what bluetooth module you are choosing.

Setting the bluetooth HC-05 module

Setting the bluetooth means that we want to change the bluetooth parameter such as :

- Name
- Password
- Role (Master/Slave)
- Baud rate
- Connection Mode
- Etc.

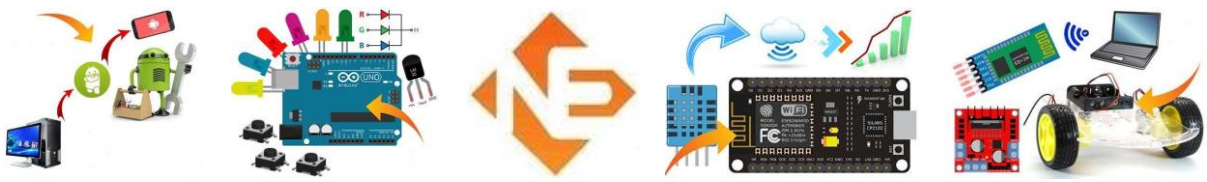


AT Command

- AT command is a standard command set developed by Dennis Hayes to control modems in 1981.
- For the Bluetooth module, AT command is used to just set the control parameters.
- Unlike modem that has many different kind of command such as to:
  - Dialling, hanging up, messaging, GPS connectivity, Internet connectivity and many more.
- For the bluetooth at command you can refer the document manual about the at command.

Scan here to download the at command set manual

Or follow this link:
https://goo.gl/XvljRl

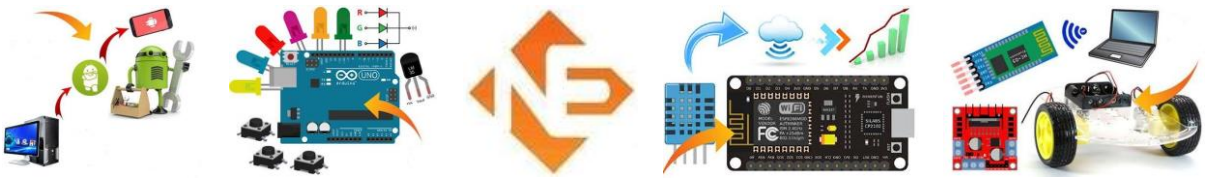


Software to perform AT Command

- We used software sscom32 (aka the terminal) to perform AT command for the bluetooth module.
- It's because SSCOM's serial buffer is more stable. Plus the serial monitor is customizable.
- For our cases the terminal is used as medium communication between PC and the microcontroller.
- See the next slide for downloading the software or simply visit the website: https://goo.gl/R22Gz0

Scan here to download the software

Or follow this link:
https://goo.gl/ik6Ygr

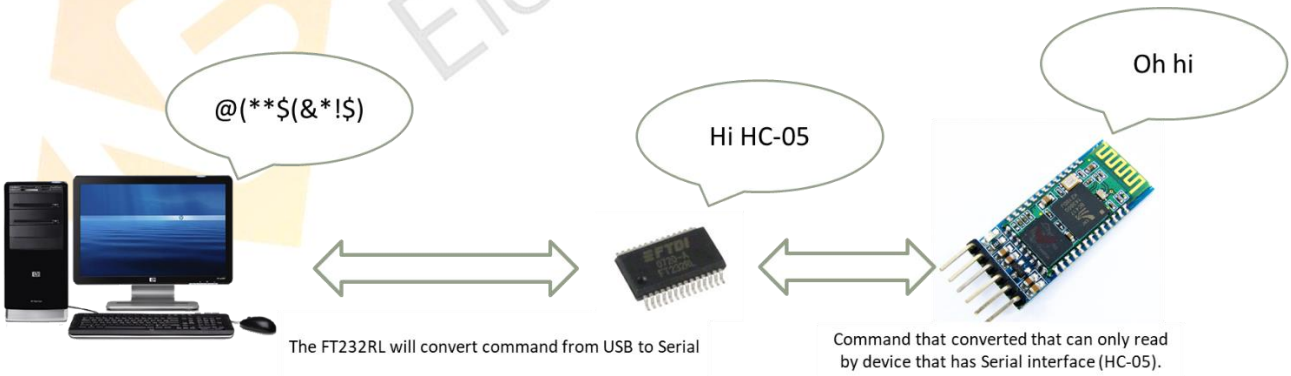


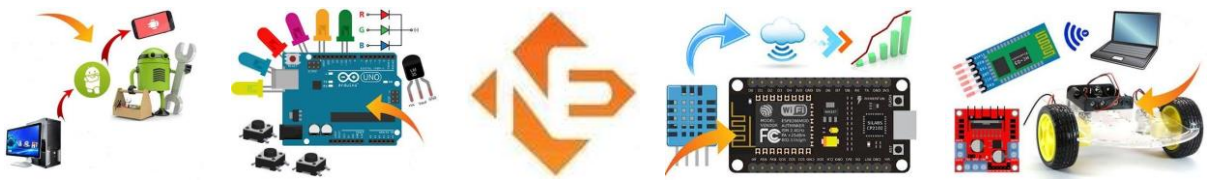
Hardware to perform AT Command

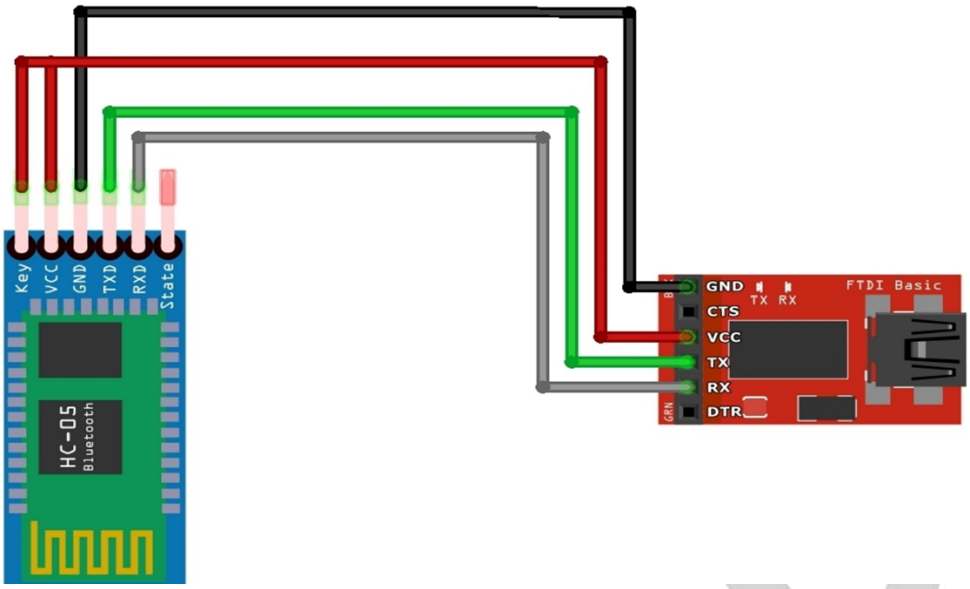
- To send command from PC to Bluetooth Module, we need to use USB to serial converter.
- USB to serial converter will convert any data from PC to serial port that connected.
- When it comes to serial in embedded, usually it consist of TX (transmitter) and RX (receiver).
- There is two technique that can be used to perform AT command for the bluetooth :
  - Using **FT232RL chip** as USB to serial converter
  - Using **ATmega8up at Arduino** as USB to Serial converter
- Even there is different technique but the goal is same, to send command from PC to bluetooth module.

Using FT232RL chip as USB to serial converter

- The FT232RL chip or we call it as FTDI chip that commonly used in USB to Serial interface.
- Widely used in microcontroller application, very cheap and easy to configure.



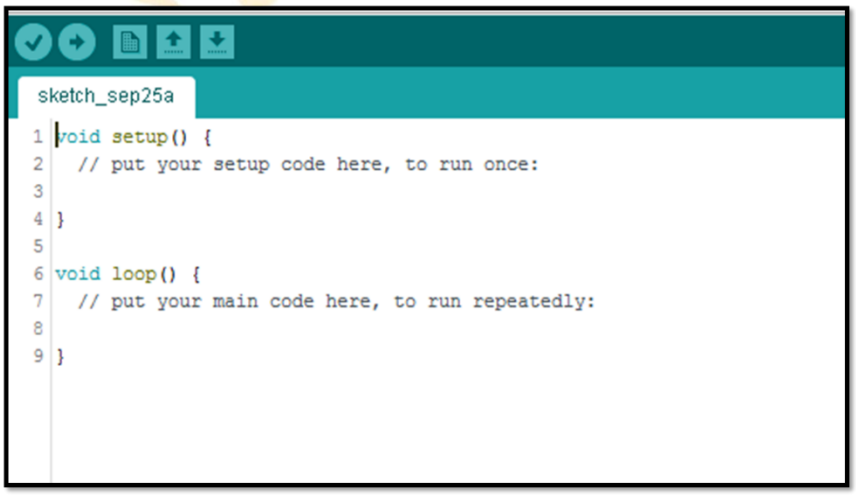


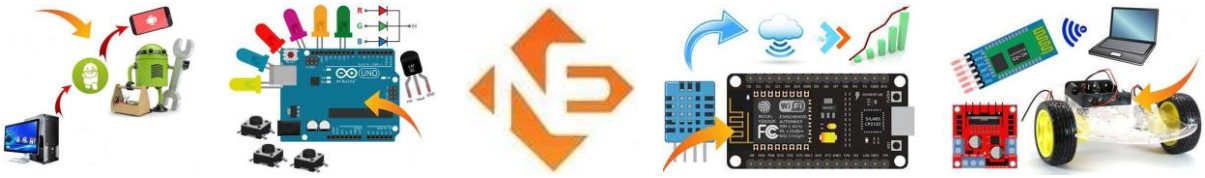


Using Arduino as USB to serial converter

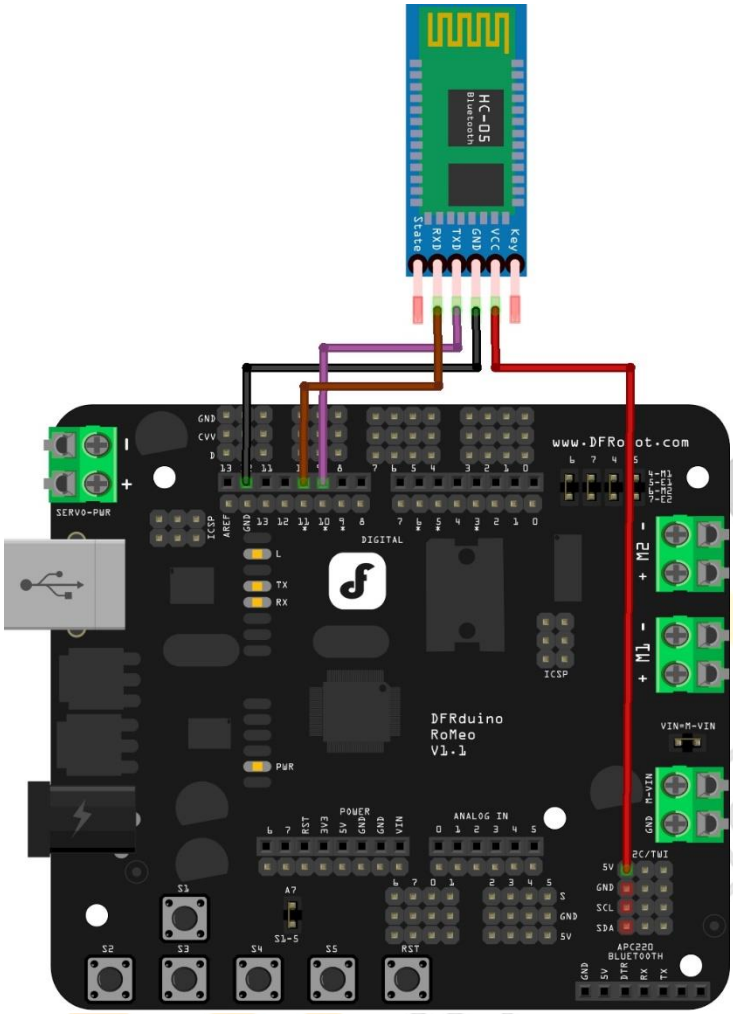
- ATMega8up at Arduino UNO R3 act as USB to Serial Converter.
- ATmega8up convert data from USB to Serial When uploading process of the code to ATmega328p.
- To use the ATmega8up alone without atmega328, just upload blank sketch or remove atmega32p.

Uploading a blank sketch to Arduino



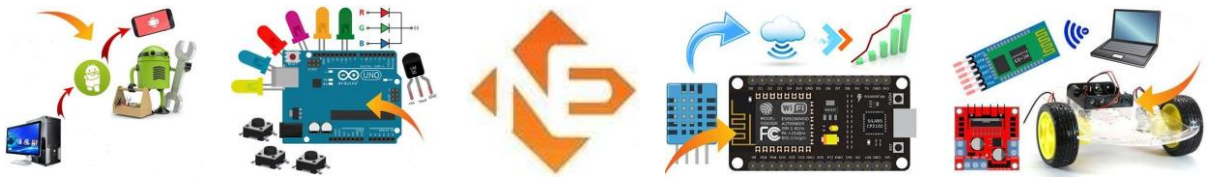


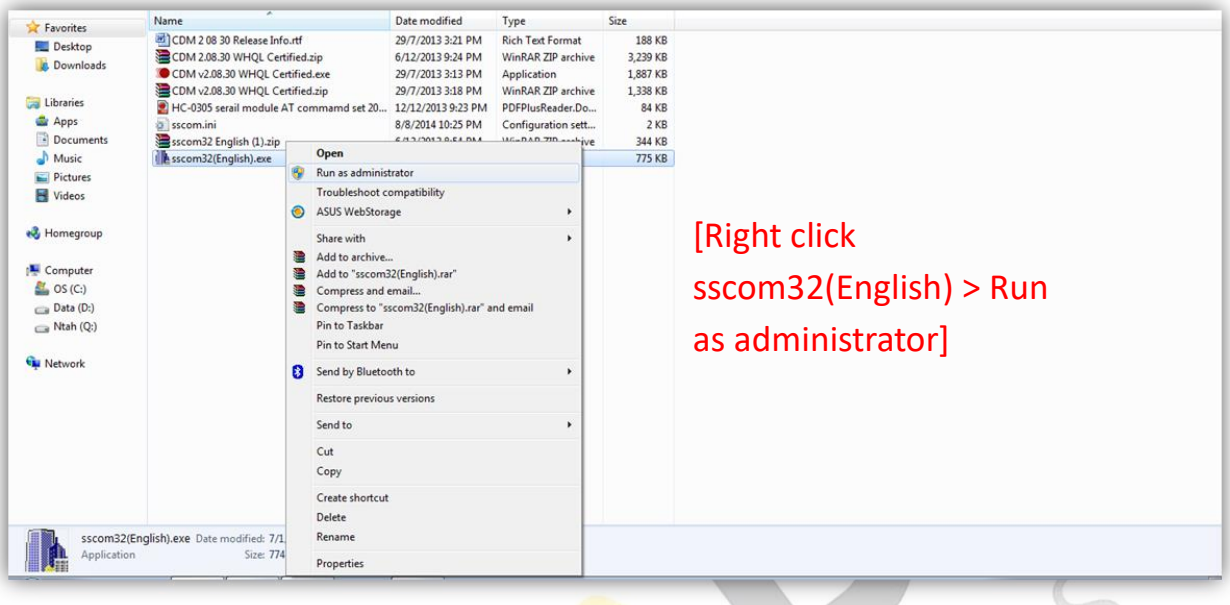
Circuit:



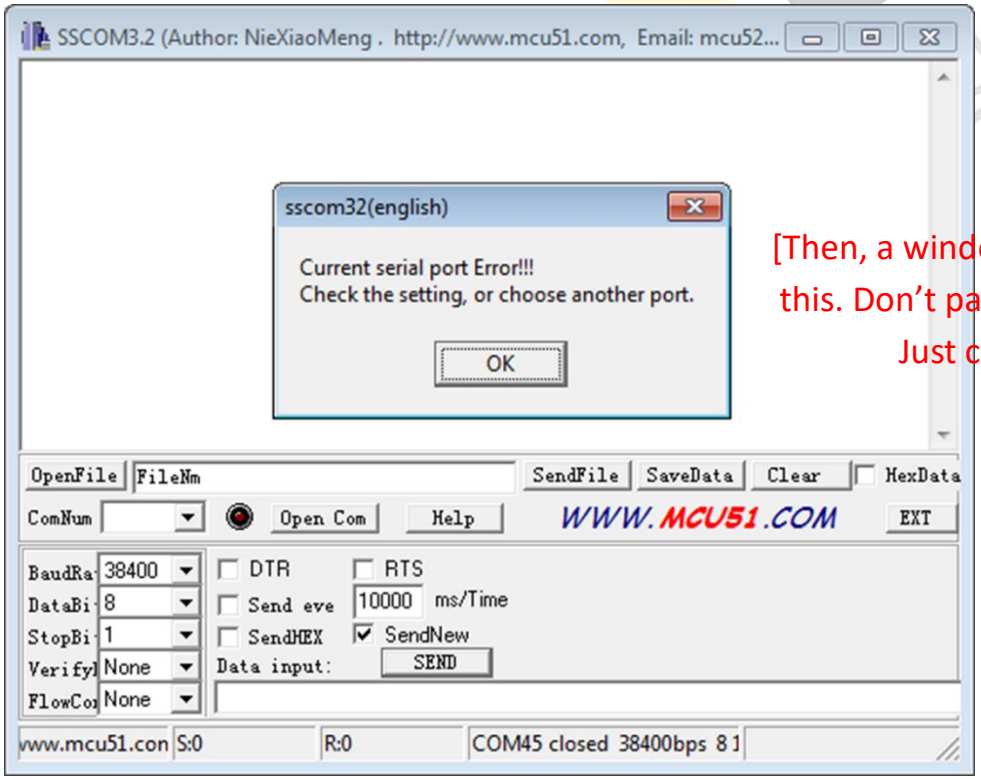
AT Command Using sscom32.exe

- There are many type of terminal that can be used to performed AT command
- In this case, we're using the sscom32.exe as terminal.

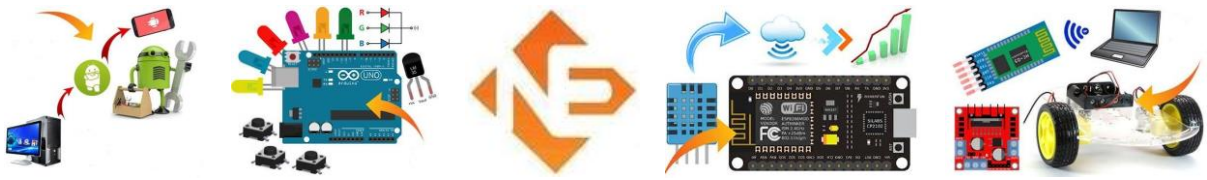


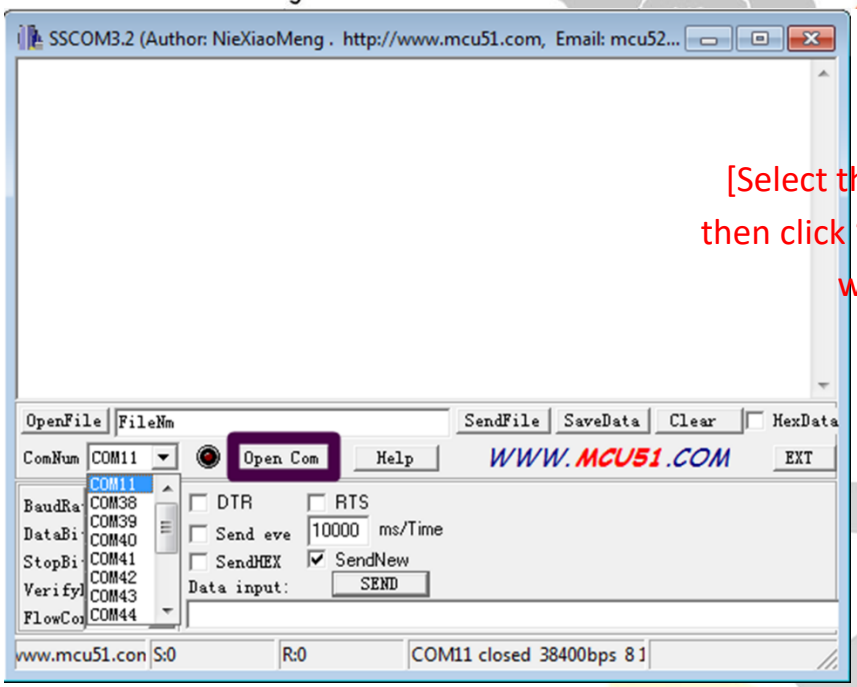


[Right click sscom32(English) > Run as administrator]

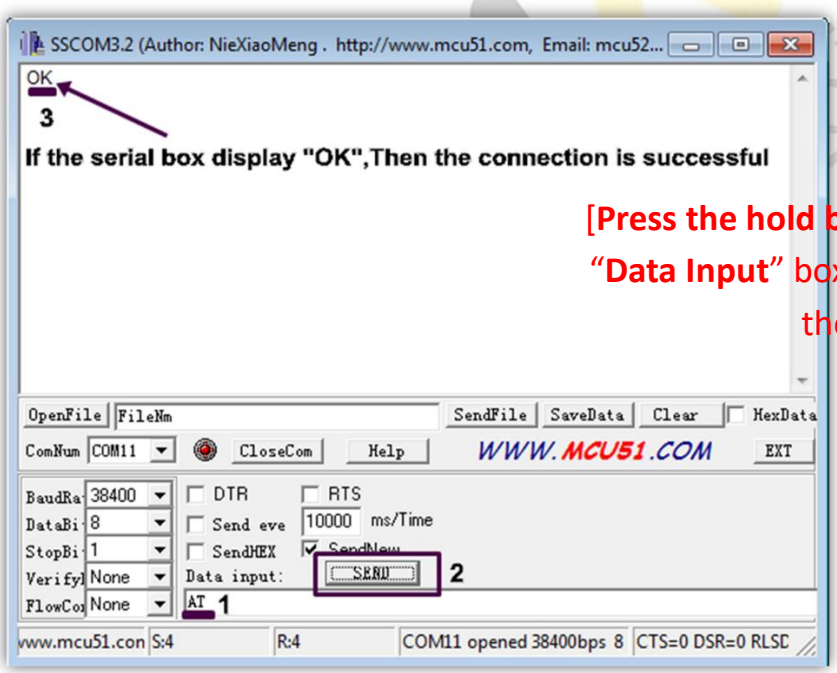


[Then, a window will pop up like this. Don't panic with the error, Just click "OK".]

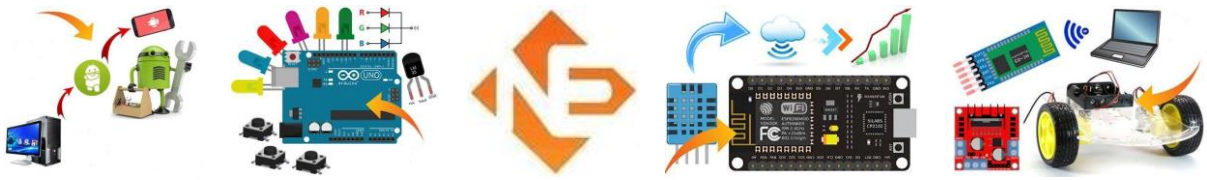


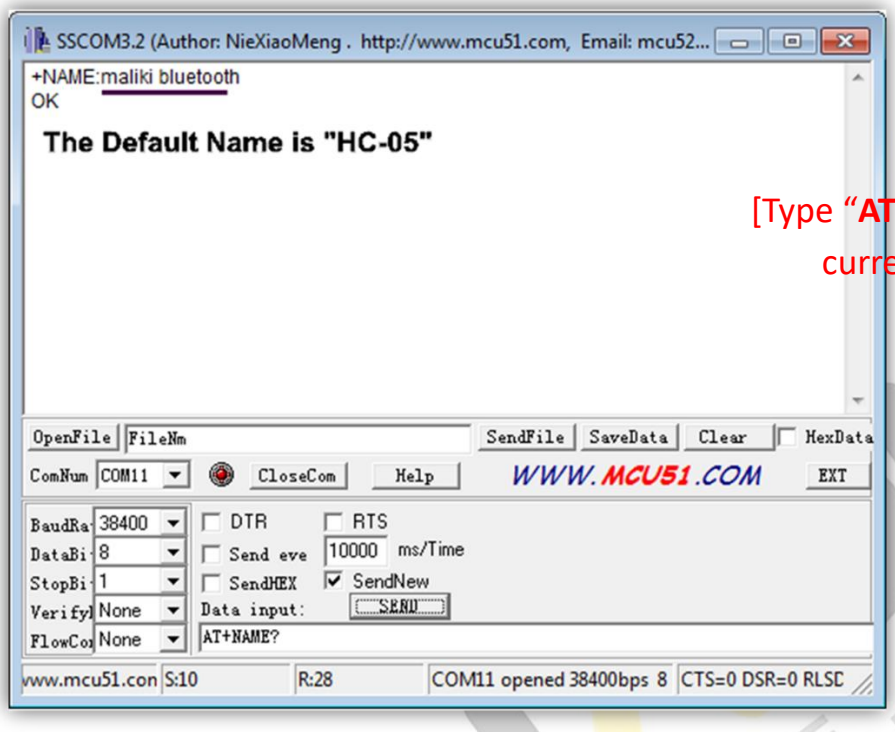


[Select the Arduino COM Port and then click "Open Com". The black dot will change to red.]

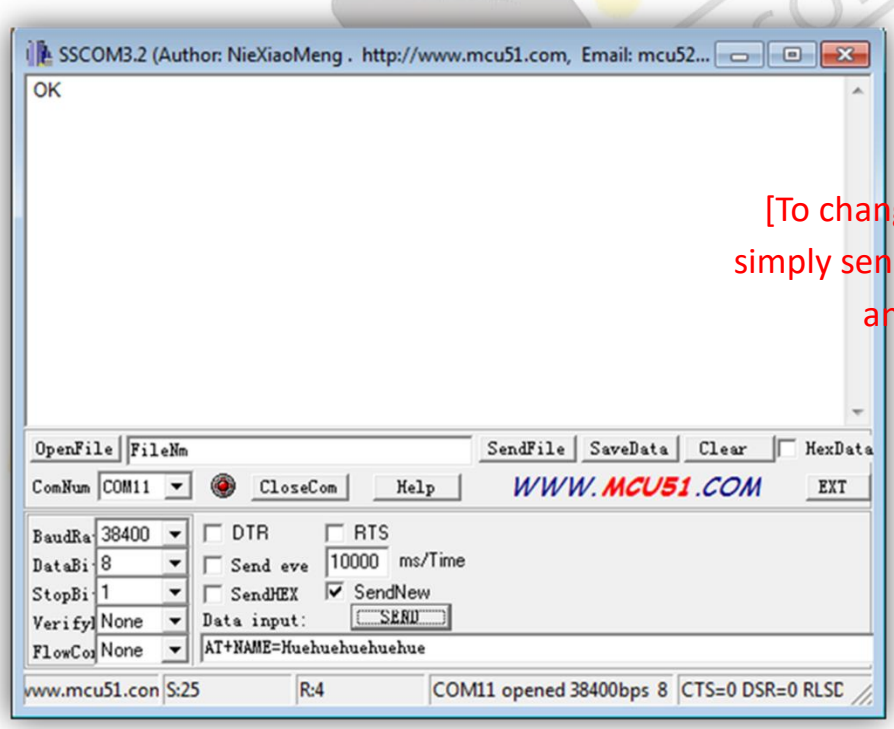


[**Press the hold button on HC-05**, Type AT in "**Data Input**" box and press "**SEND**" to send the command.]

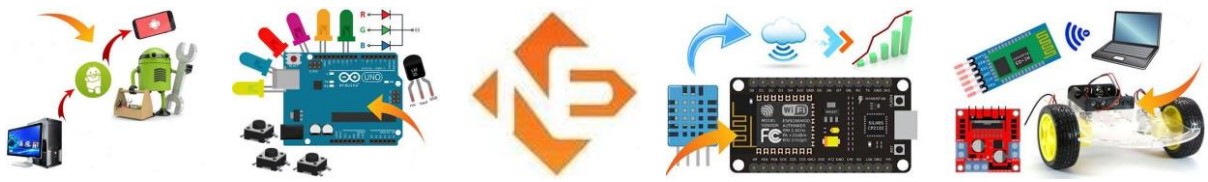


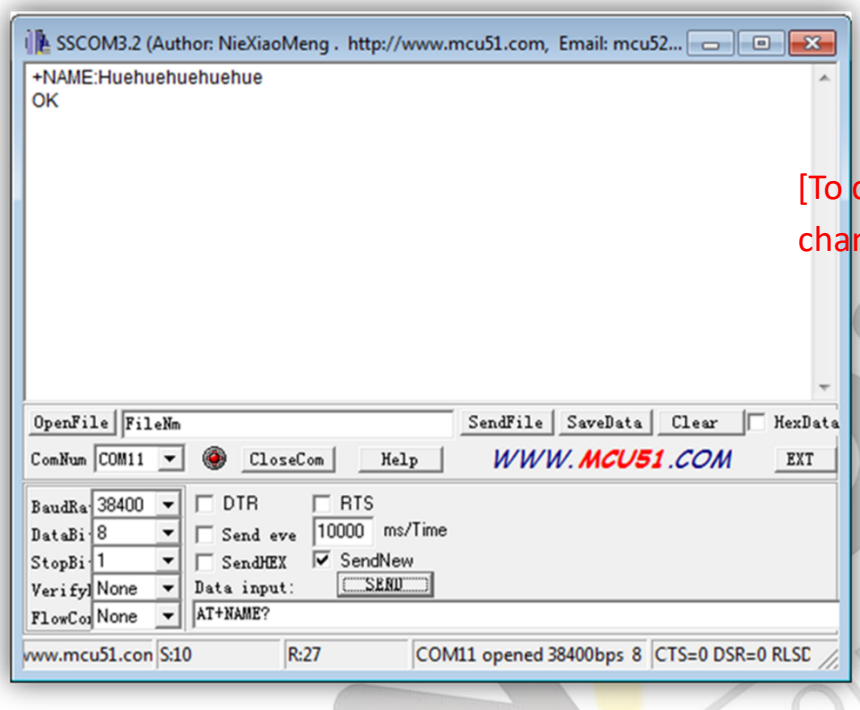


[Type "**AT+NAME**?" to identify the current bluetooth name]

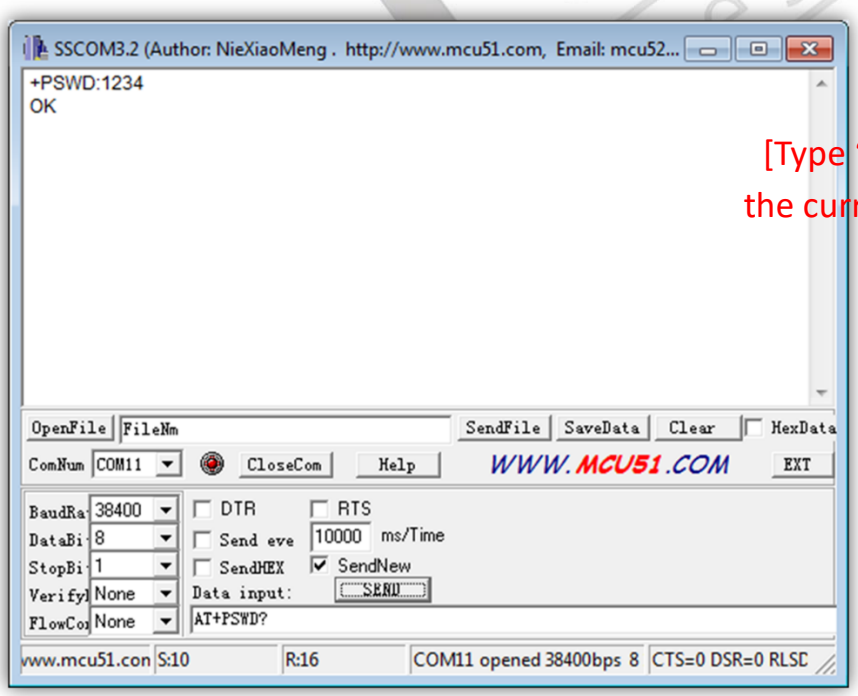


[To change the bluetooth name, simply send "**AT+NAME=*any name***" and press "**SEND**".]

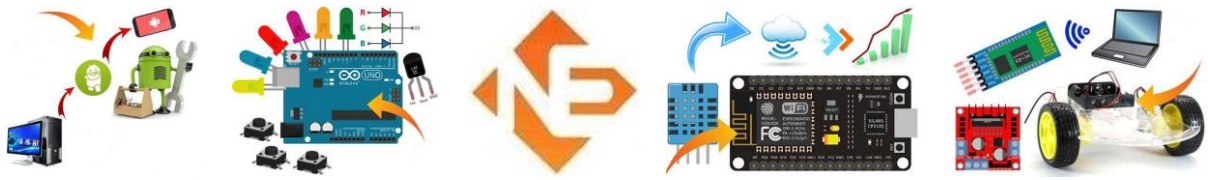


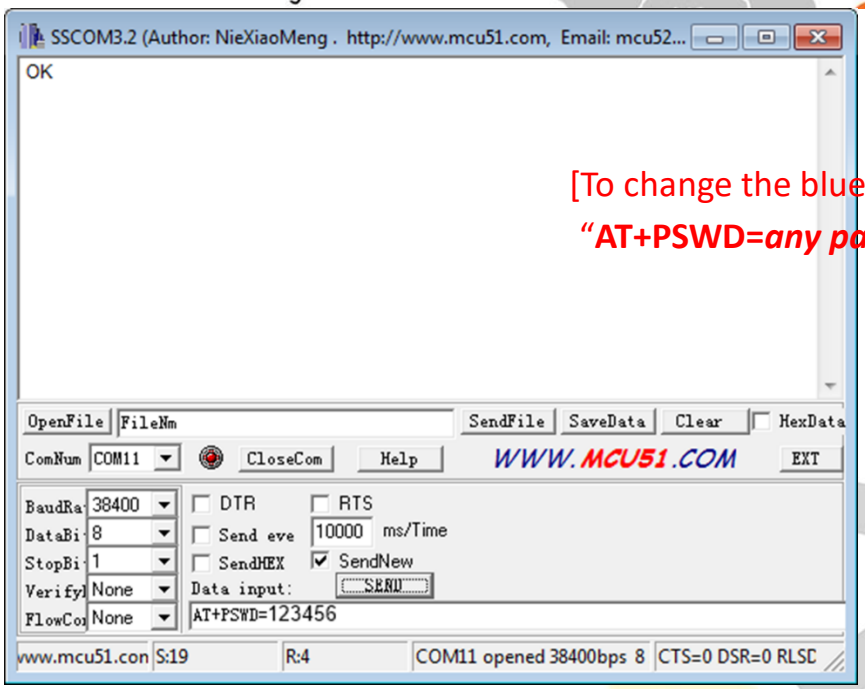


[To check if the name have been changed send "**AT+NAME?**" and press "**SEND**".]

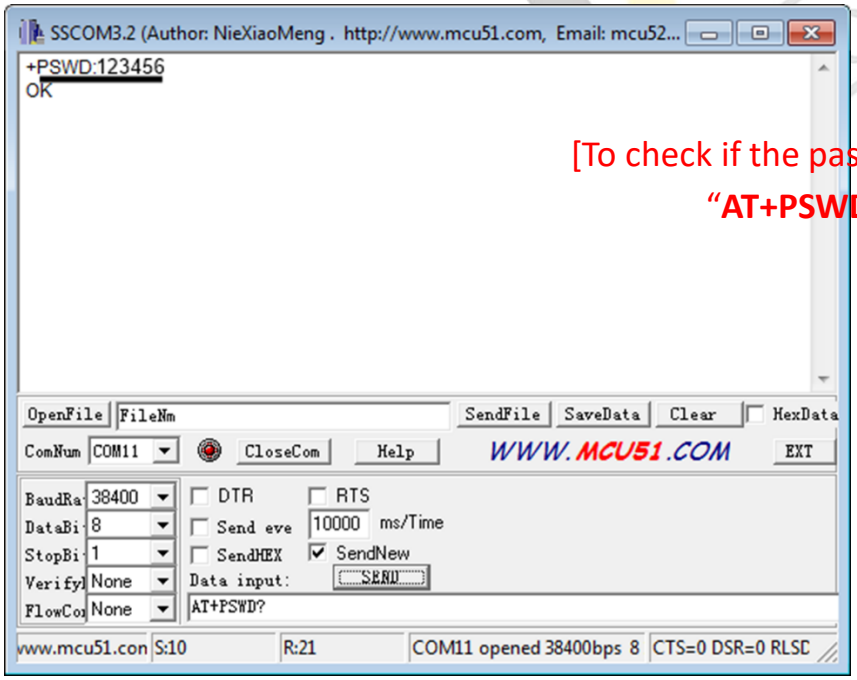


[Type "**AT+PSWD**?" to identify the current bluetooth password]

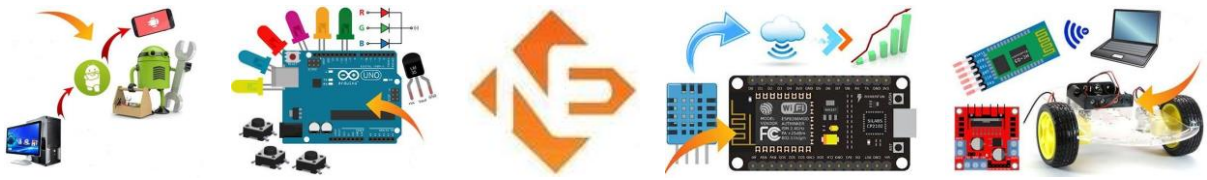




[To change the bluetooth password, simply send "**AT+PSWD=*any password***" and press "**SEND**".]



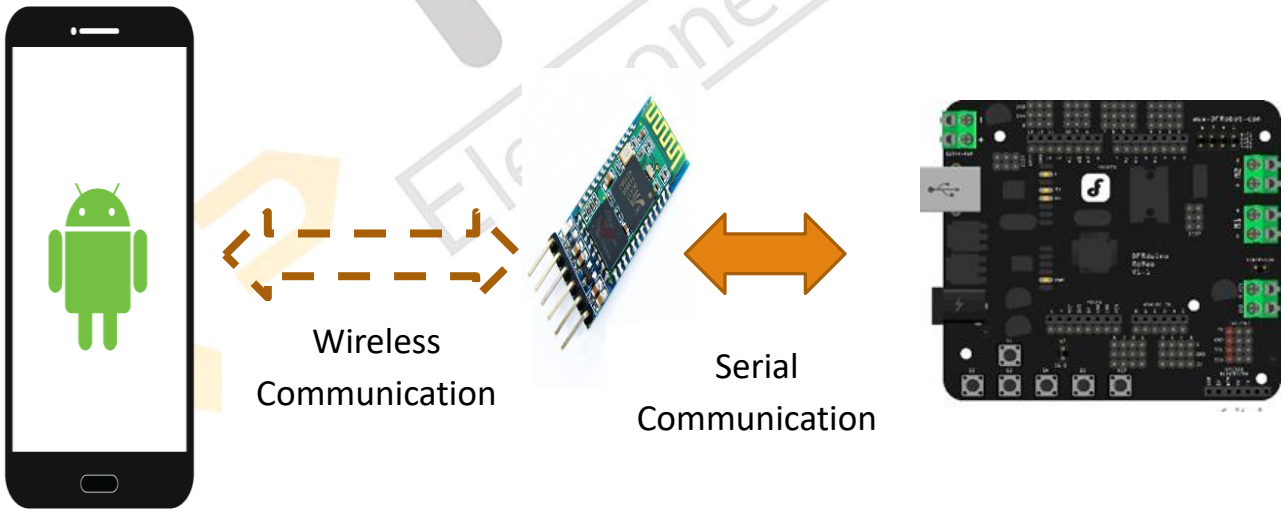
[To check if the password have been changed send "**AT+PSWD?**" and press "**SEND**".]

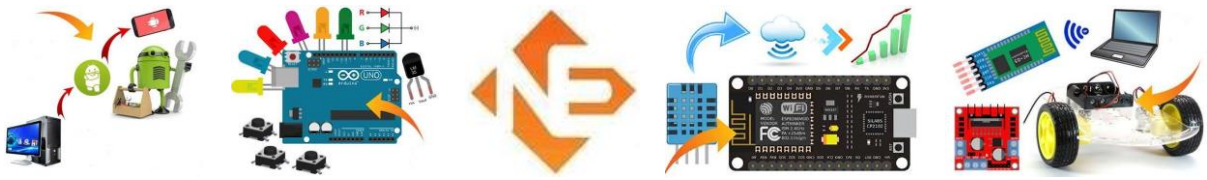


Project 4 – Bluetooth Mobile Apps Robot Control

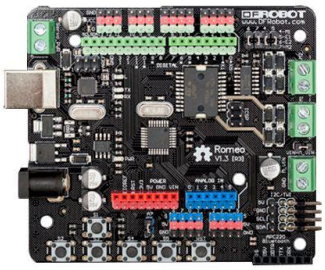
Connectivity

- There are many ways to connect to the bluetooth module.
- Nowadays, there are many devices that have bluetooth connectivity.
- Any device that have bluetooth connectivity can connect to bluetooth module.
- In this case, we're using Android smartphone to connect to the bluetooth.





List of needed material



Arduino Romeo board

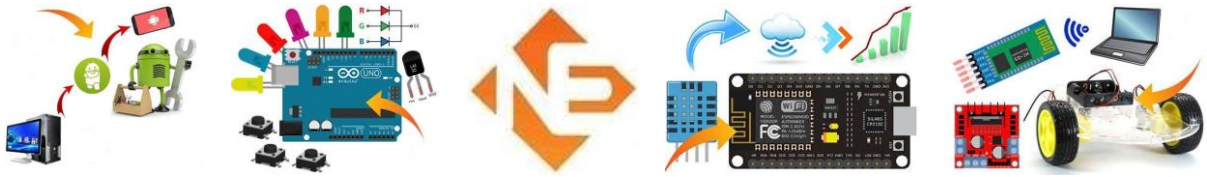
Bluetooth HC-05

Jumper

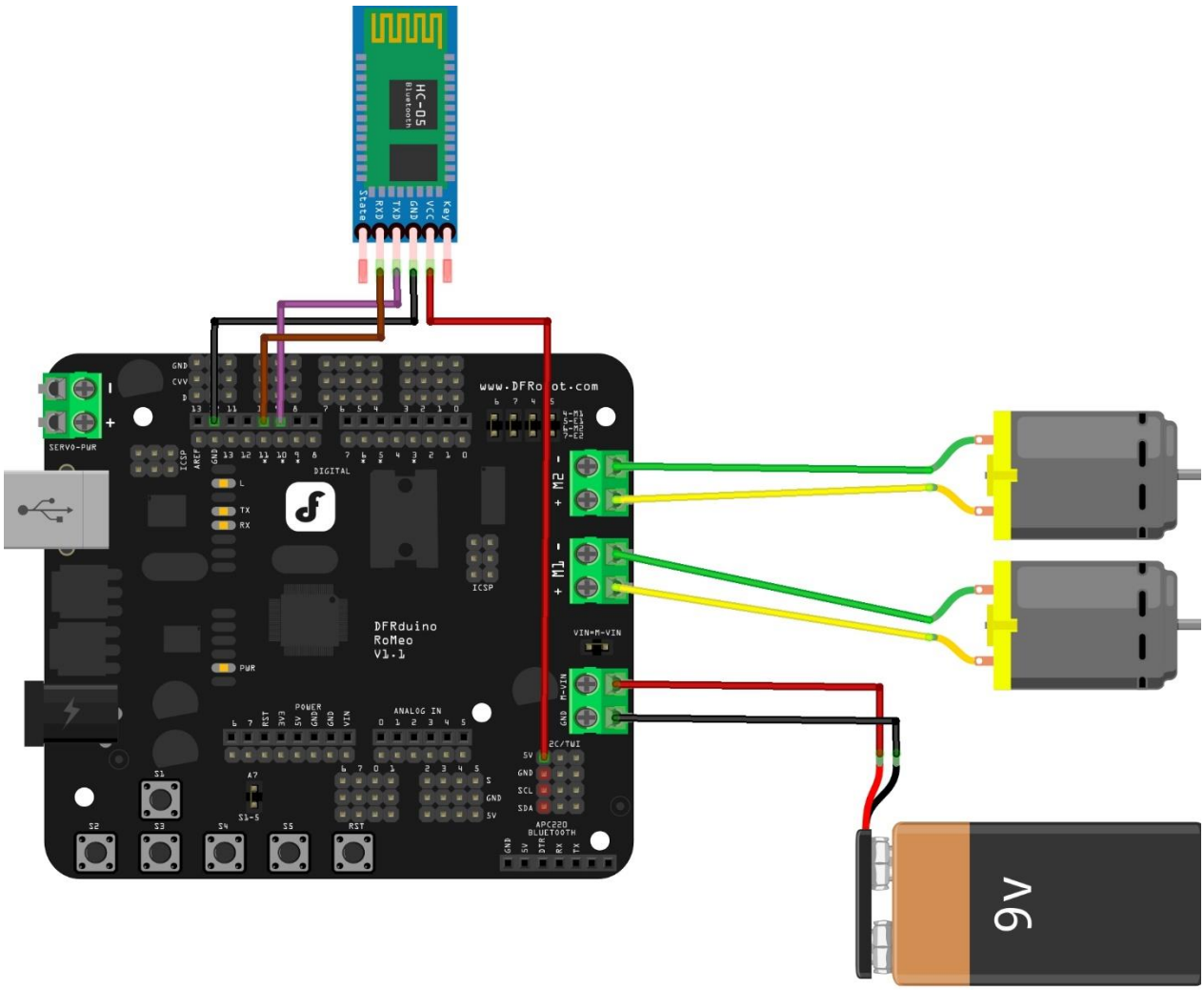
4WD Robot Base

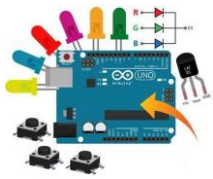
LiPo Baterry

DC motor



Build the Circuit



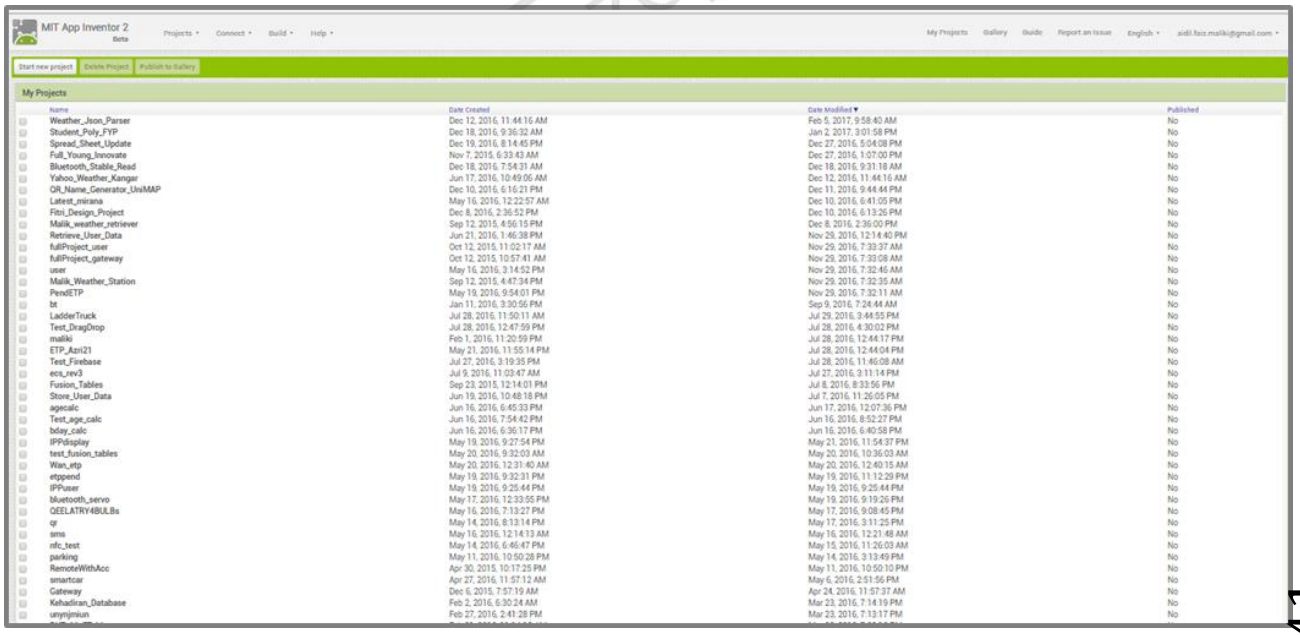

MIT App Inventor

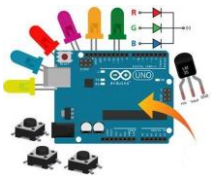
- To enable the smartphone to communicate with HC-05, we have to develop and android apps.
- We are using the MIT app inventor as our framework to develop the apps.
- Don't worry, there's no need to master any programming language when using the MIT app inventor.
- The framework used is just based on drag and drop concept. So it's kind of easy for us.

MIT App Inventor

Creating an account

- First of all, open the website first: ai2.appinventor.mit.edu
- We need to create MIT app inventor account that affiliate with our google account.

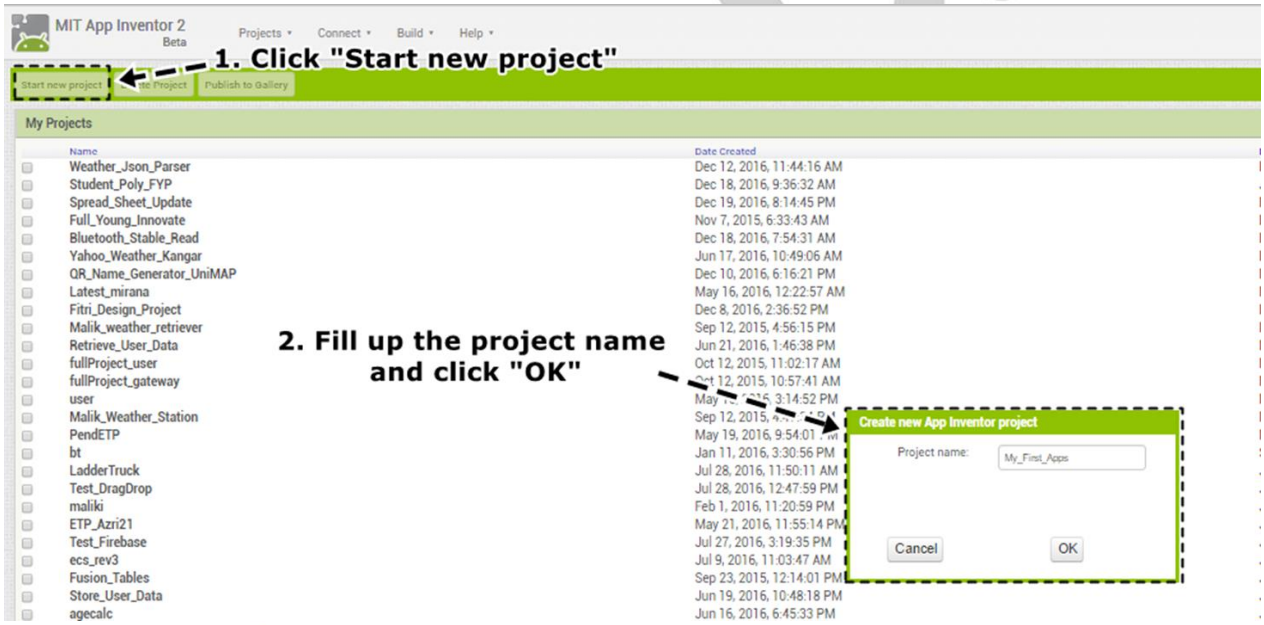



Developing our first apps

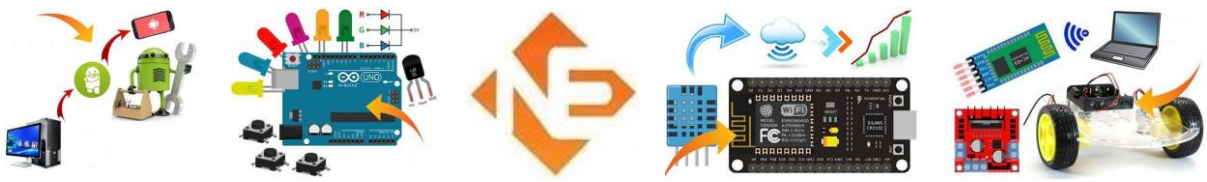
- First of all, open the website first: ai2.appinventor.mit.edu
- We need to create MIT app inventor account that affiliate with our google account.
- We are going to develop our first apps that have username and password function.
- If both username and password is true, the screen will turn to green. Otherwise it will become red.
- Let's get started.

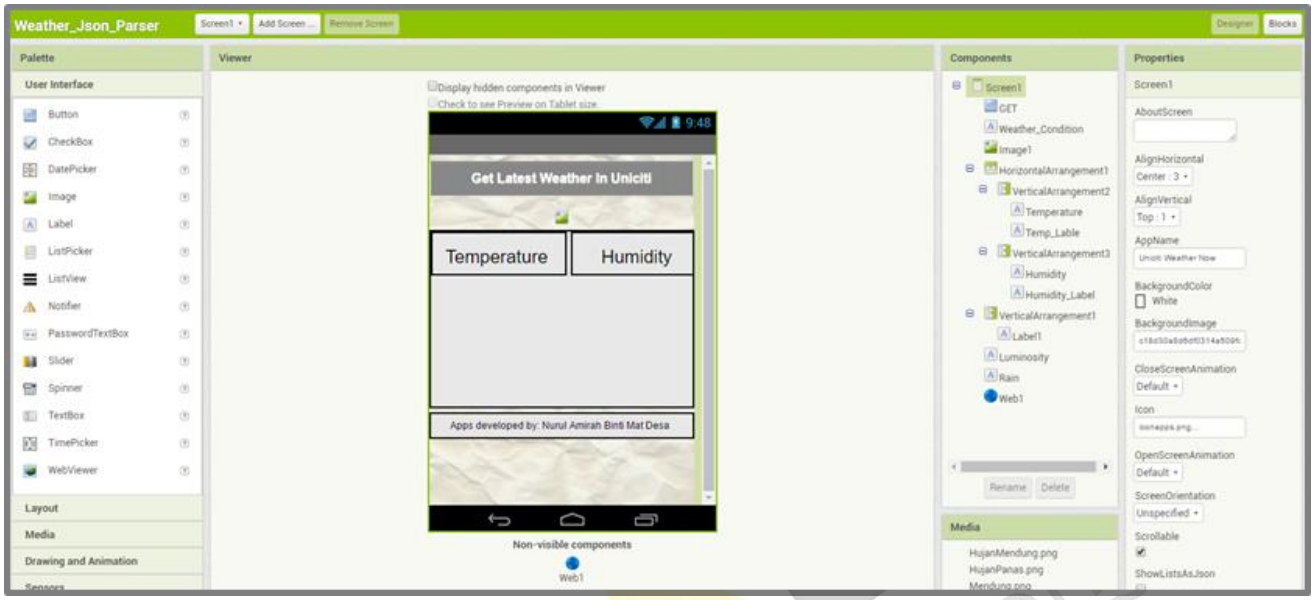
Creating new Project



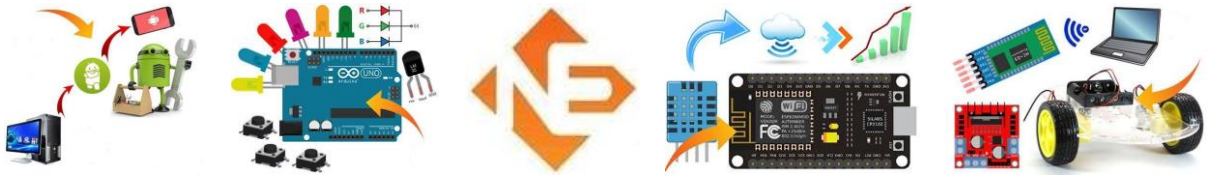
Layout designing

- For the first part, we have to design a layout which act as the User interface between apps and User.
- In designing the layout, we just need to drag and drop the block or we call it as "component".
- The component that used determine the function of our apps.
- The component used also determine how me make the input/output elements in the application.

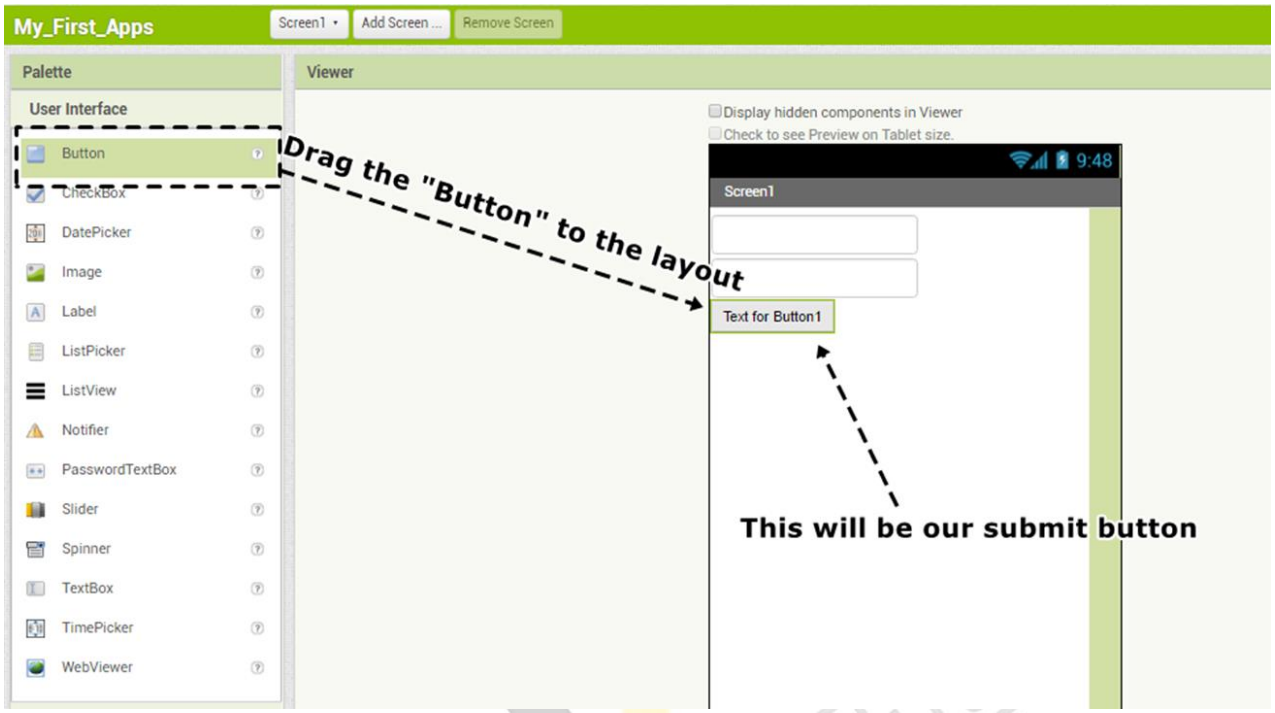




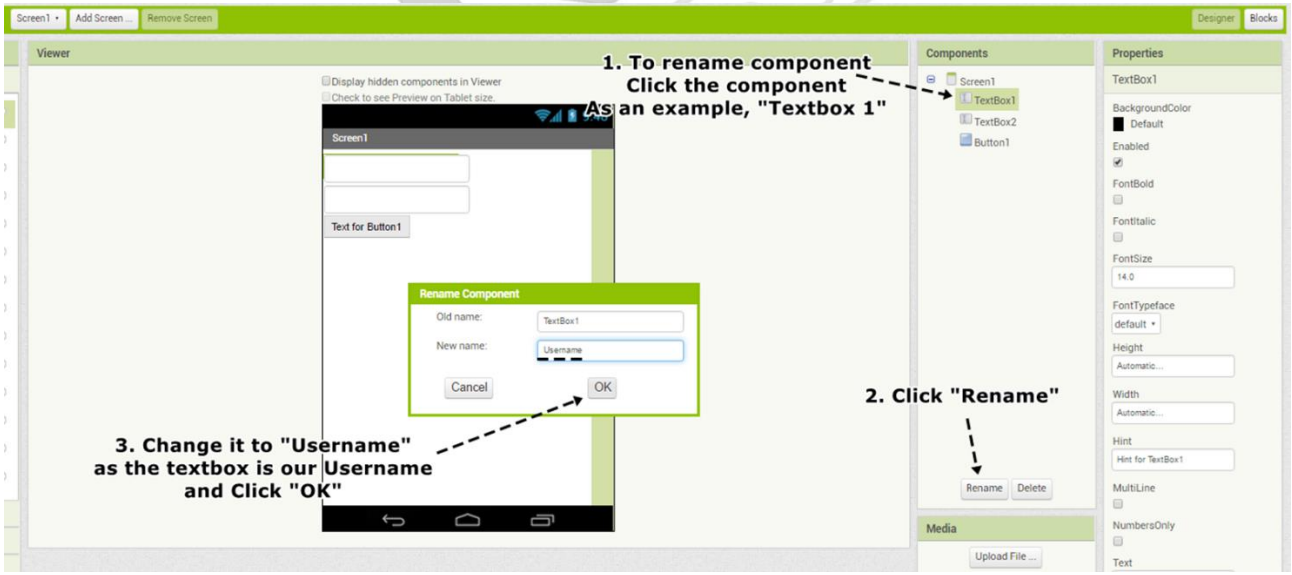
Design Layout (1st part)

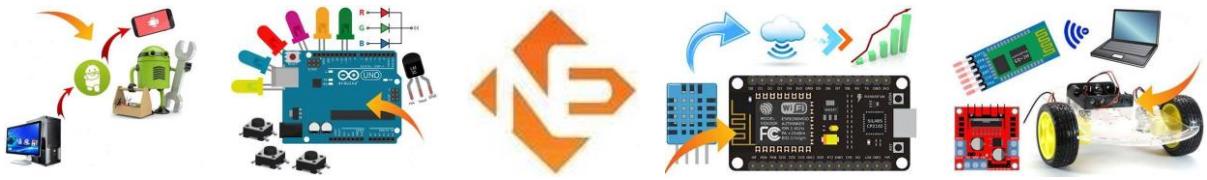


Design Layout (2nd part)

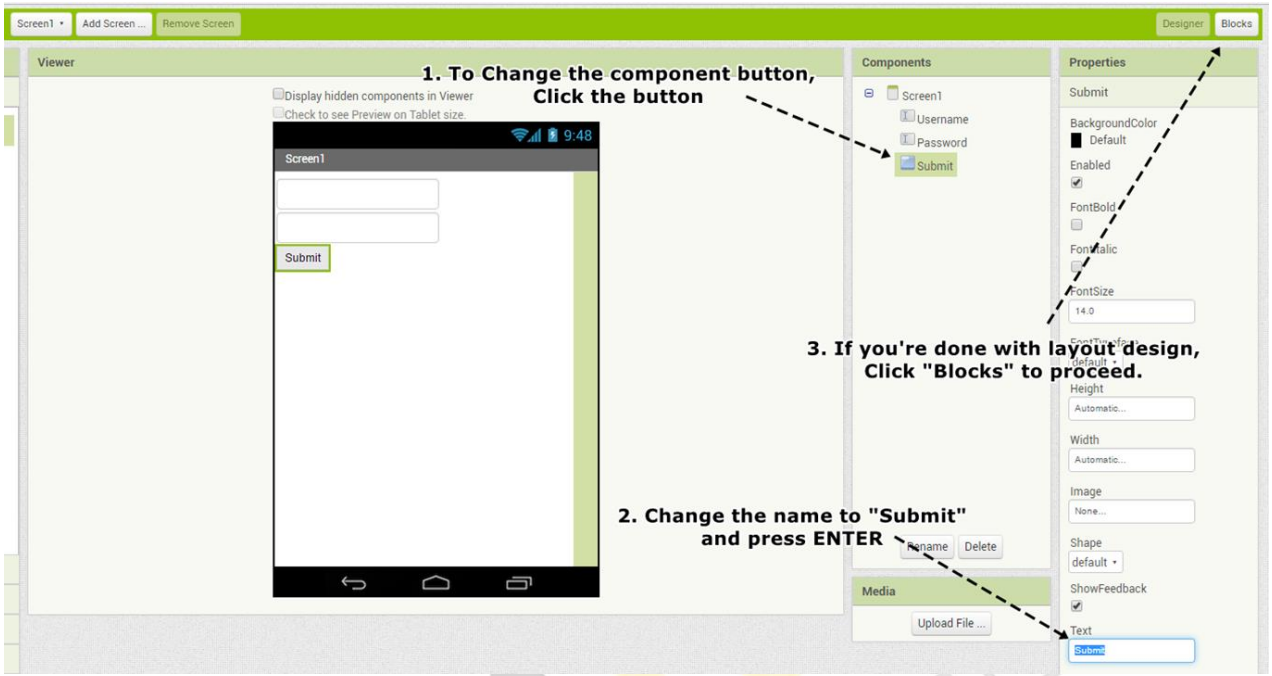


Design Layout (3rd part)



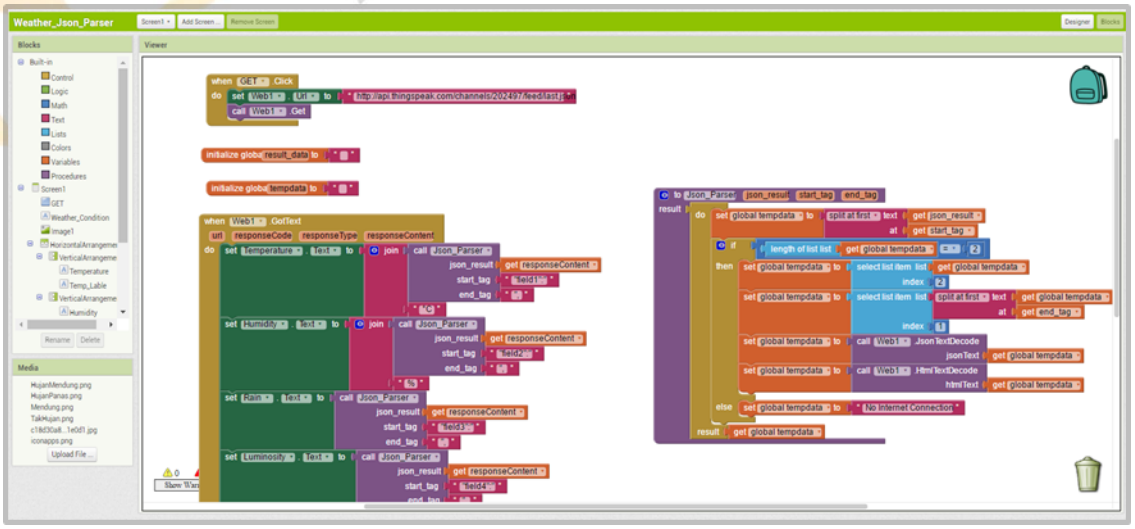


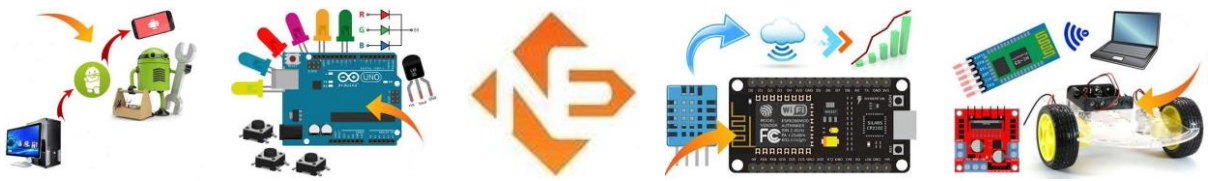
Design Layout (4th part)



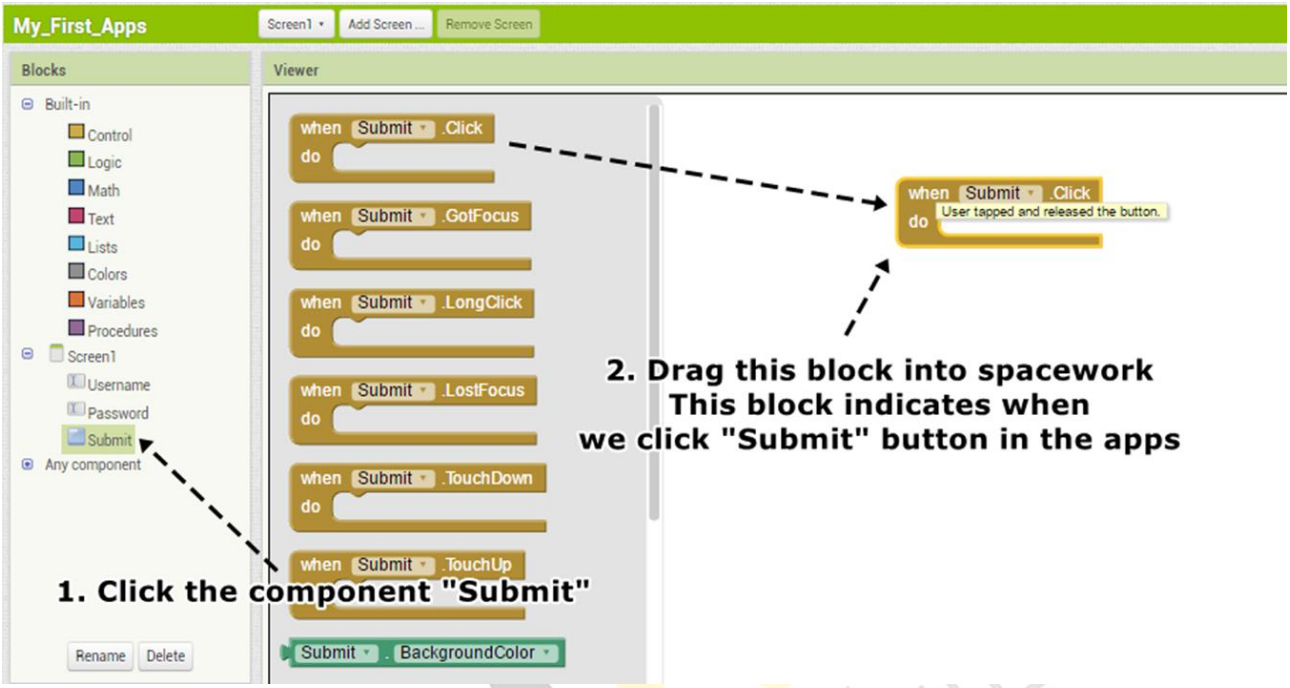
Block

- After designing the layout, we can proceed to block part.
- The block part is just as the translation from long shut line of coding which is easy for us.
- We just have to pick the right block to meet the function that we need.
- In short, the block act as the response or input processor from our layout (The Code).

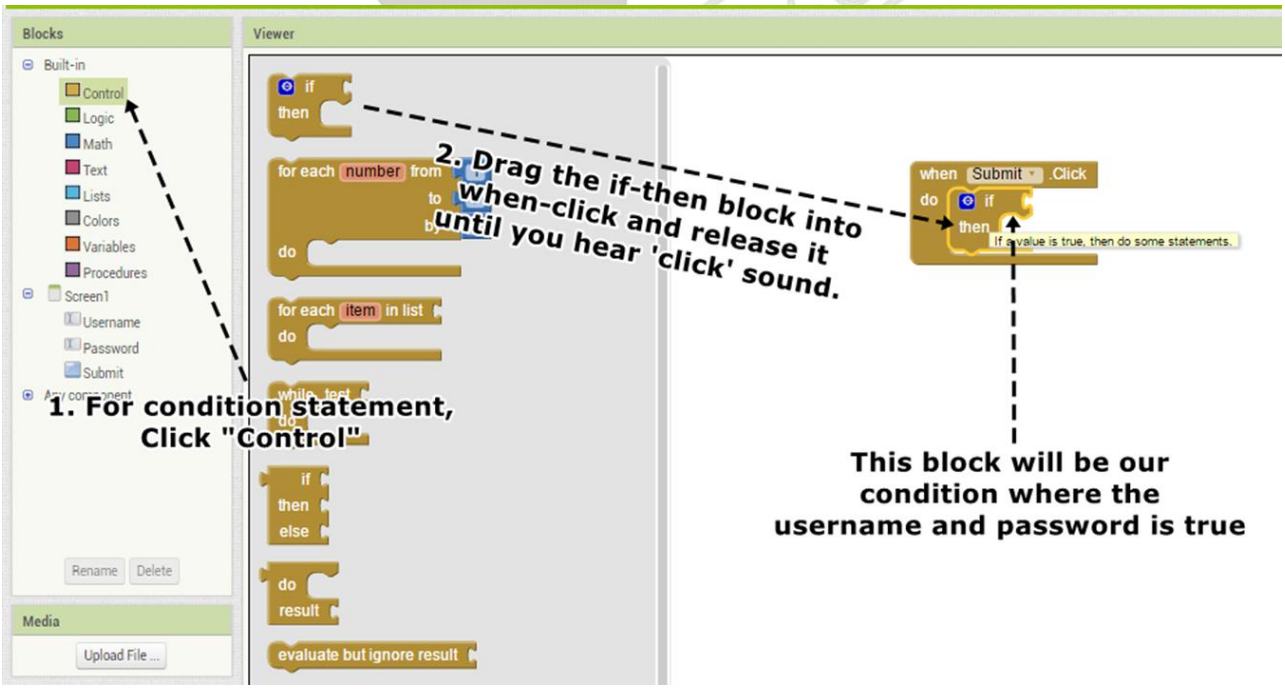


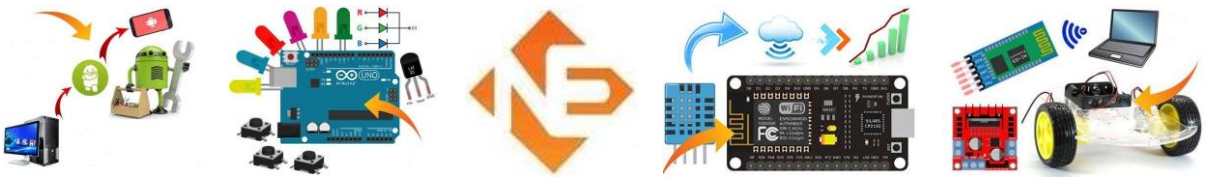


Block (1st Part)

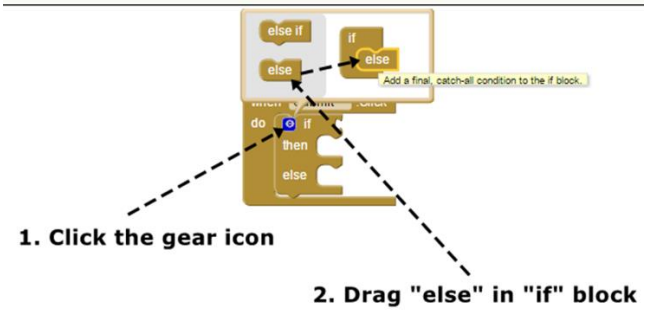


Block (2nd Part)

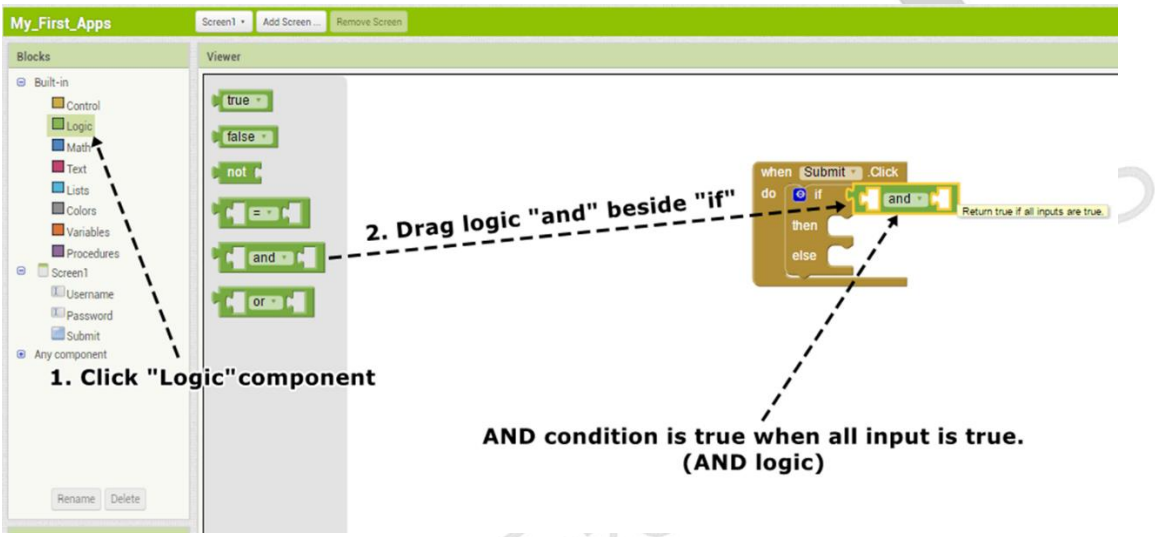




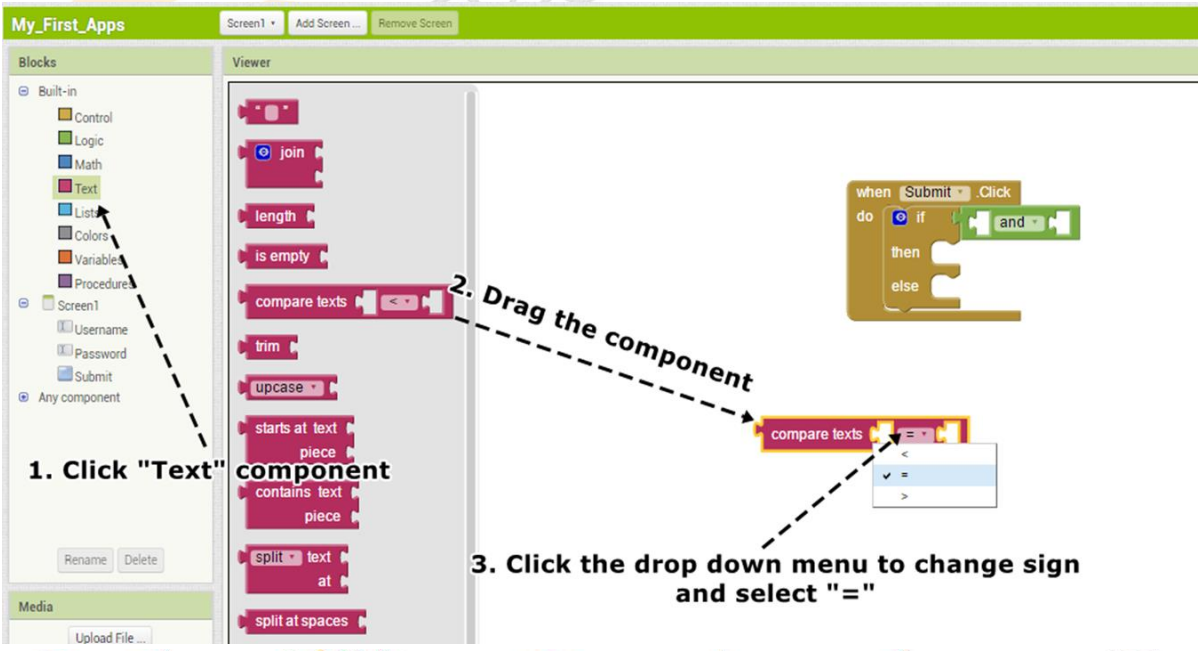
Block (3rd Part)

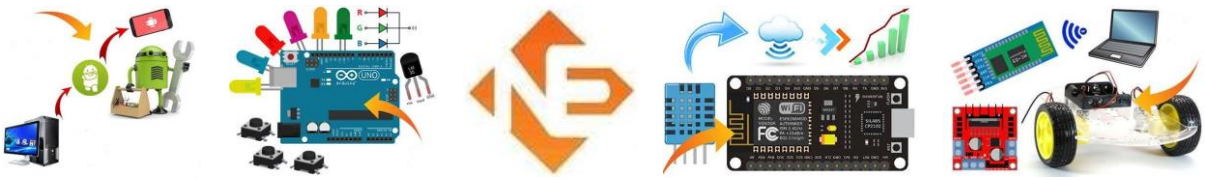


Block (4th Part)



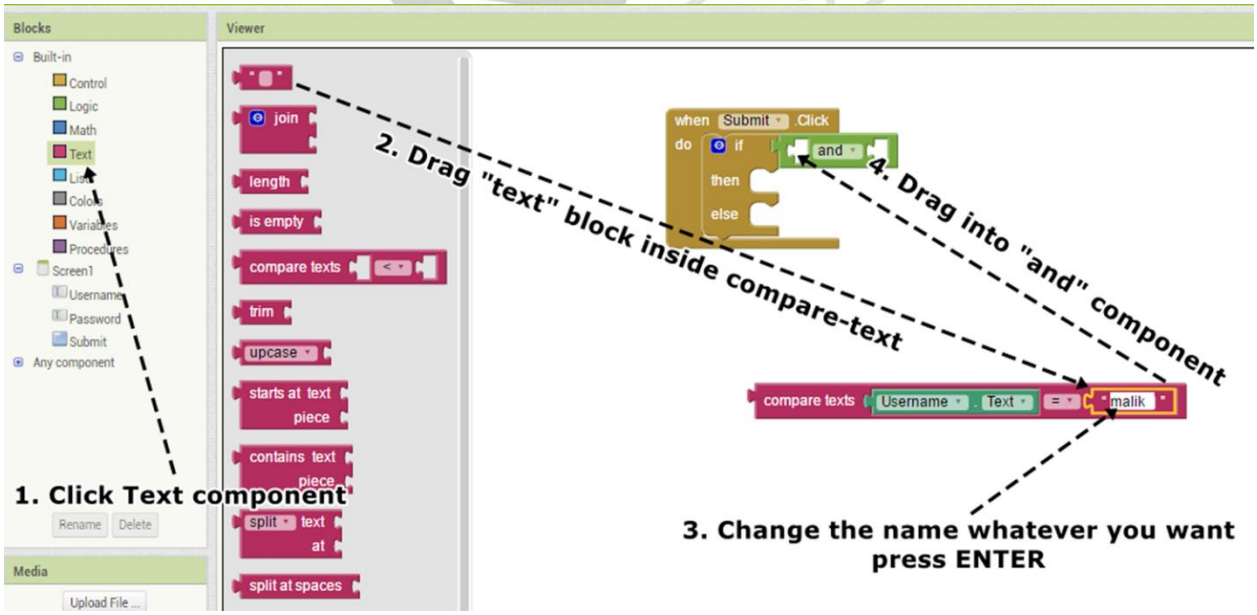
Block (5th Part)

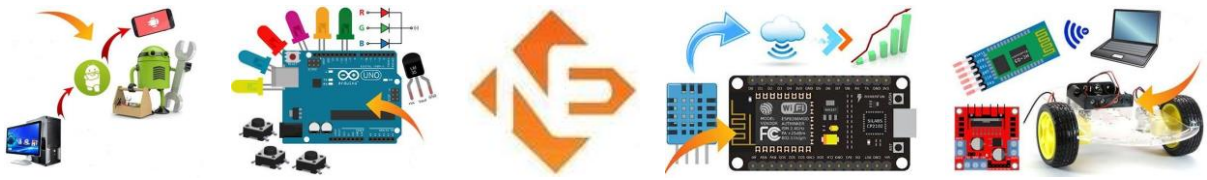




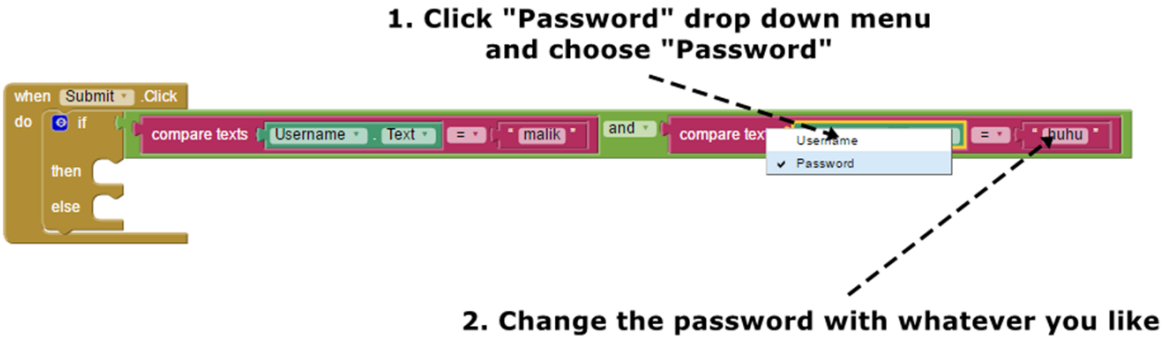
Block (6th Part)

Block (7th Part)

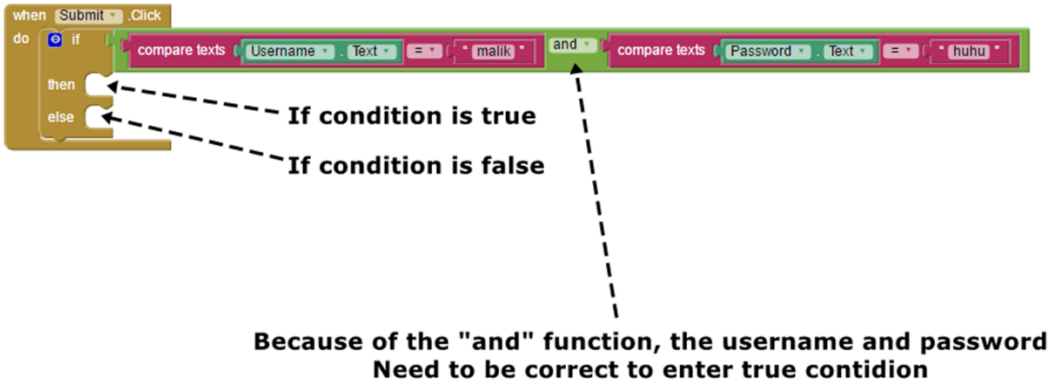




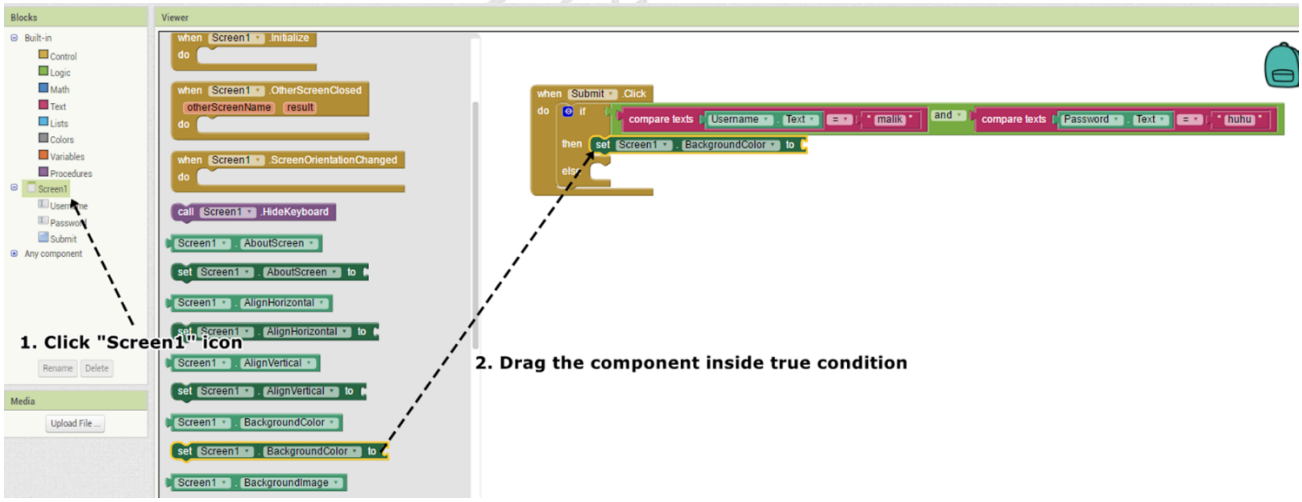
Block (8th Part)

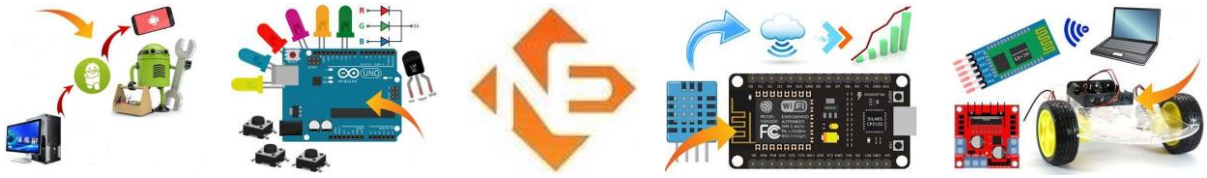


Block (8th Part) Cont.

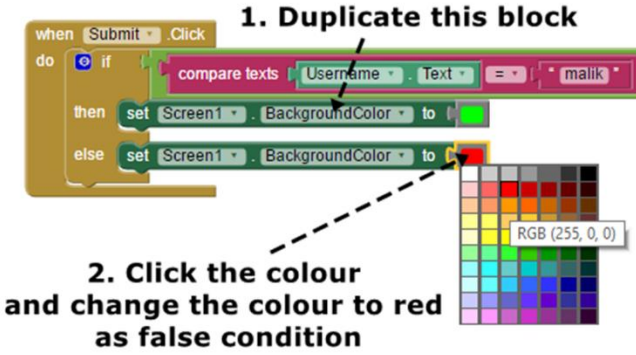


Block (9th Part)

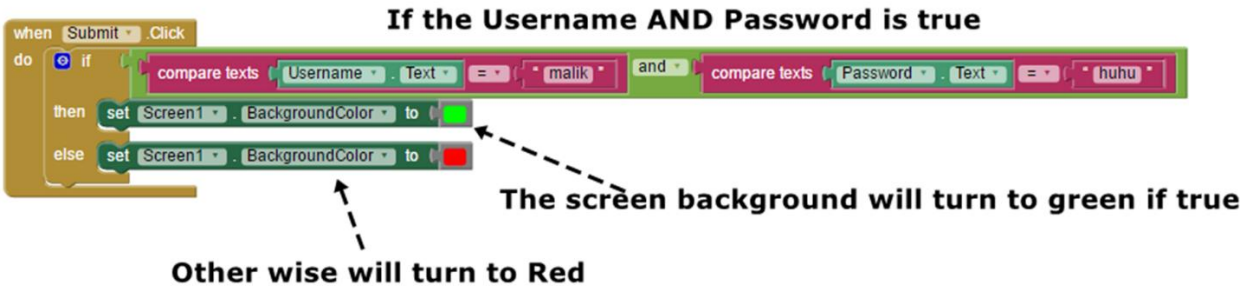




Block (Last Part)

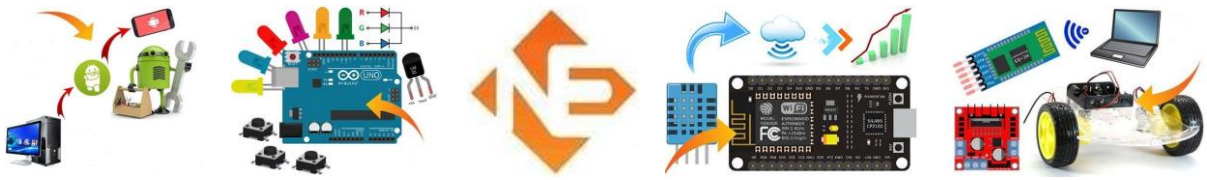


Block (Full)



How to get the apps..??

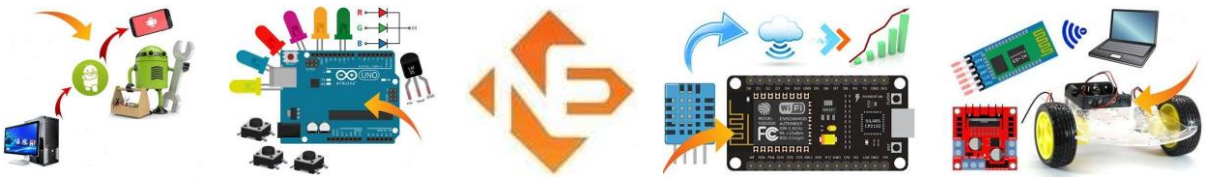
- To get the apps, you can whether download it or connect to the apps using Wi-Fi with same network.
- To download the apps :
  - Click Build > App (provide QR code for .apk) To generate the QR code and download using phone
  - Click Build > App (save .apk to my computer)
- To Connect with apps through Wi-Fi, click Connect > AI Companion
- If you currently developing the apps, you're recommended to connect with the apps through Wi-Fi.
- If you have finished with the apps and tested with functionality, you can download the apps.



Result

When Username and Password is TRUE

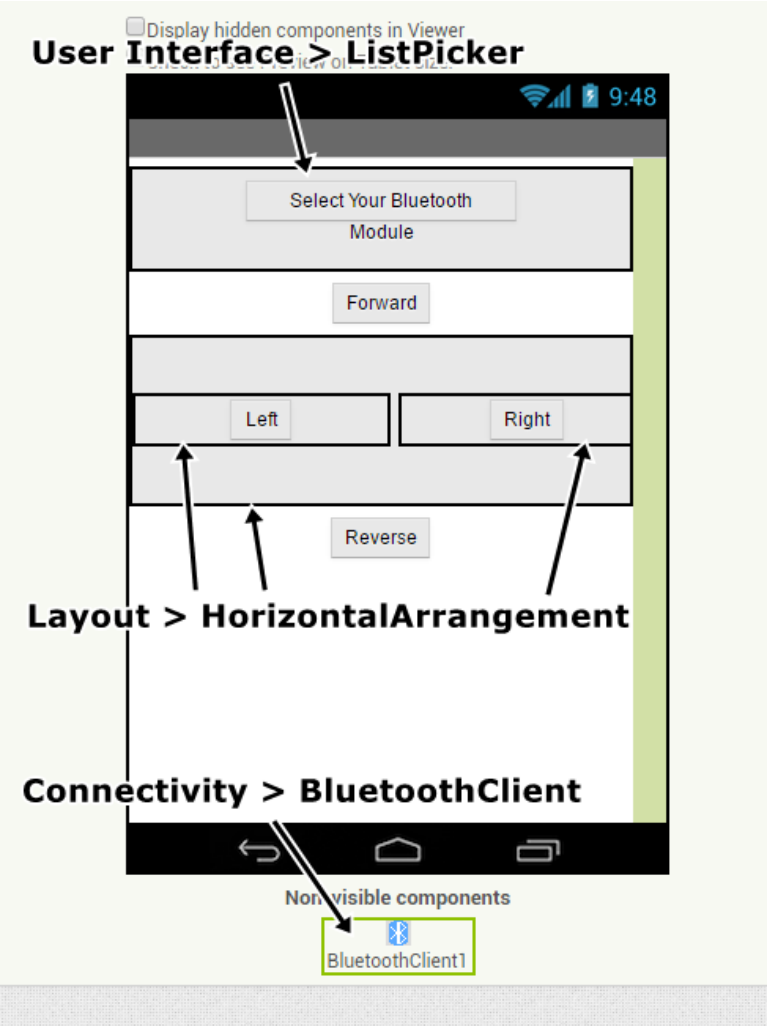
When Username and Password is FALSE

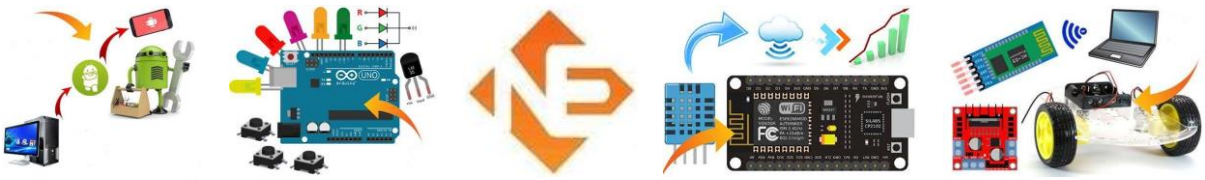


The Real Deal Is Here

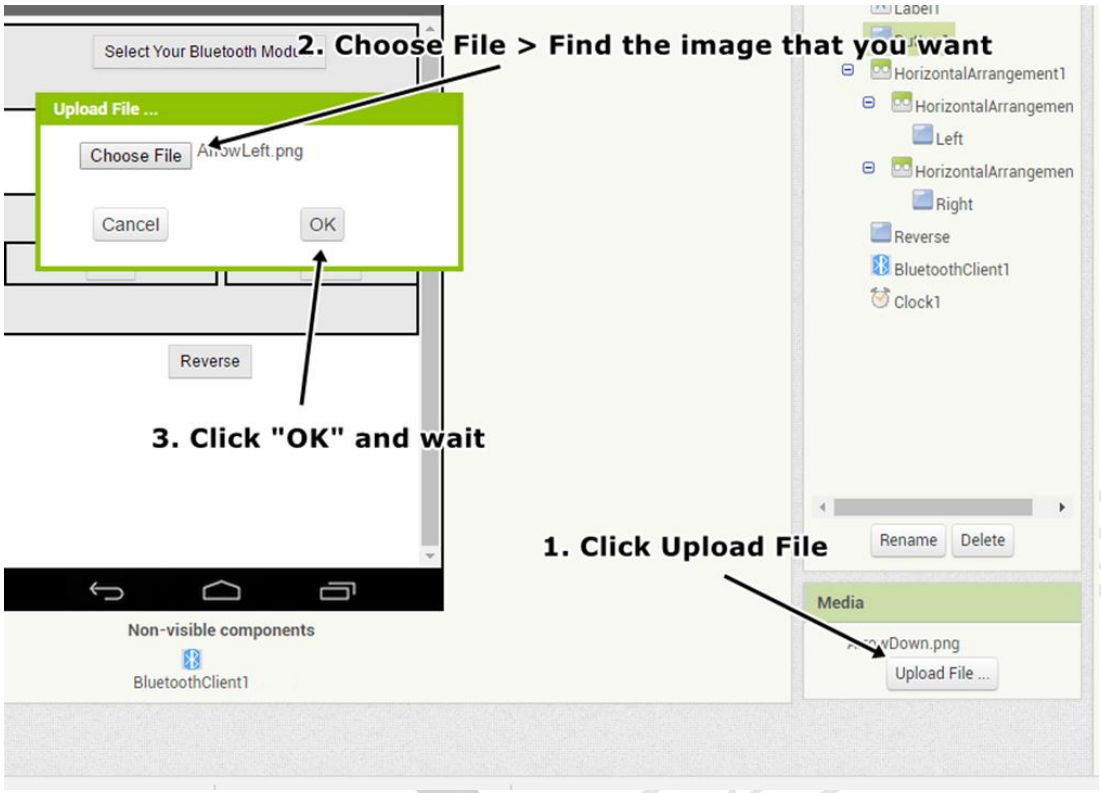
- Congratulation for completing your first apps.
- Now let's move on to the real deal. We have to create an apps with bluetooth connectivity.
- The apps will be our controller for our robot via bluetooth connectivity.
- Let's Get Started and create your new project.

Design Layout

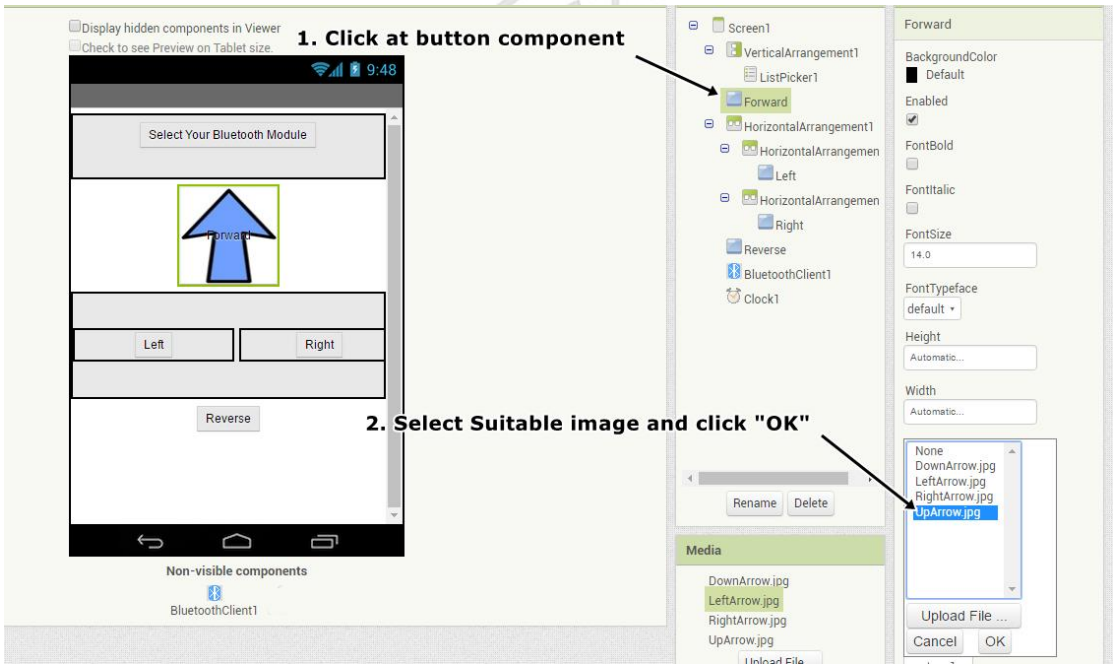


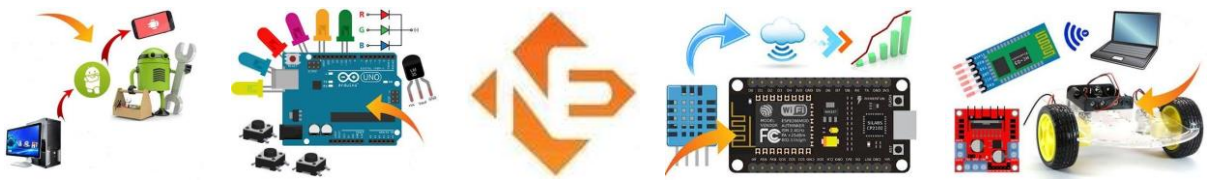


How to change button pic (1st Step)

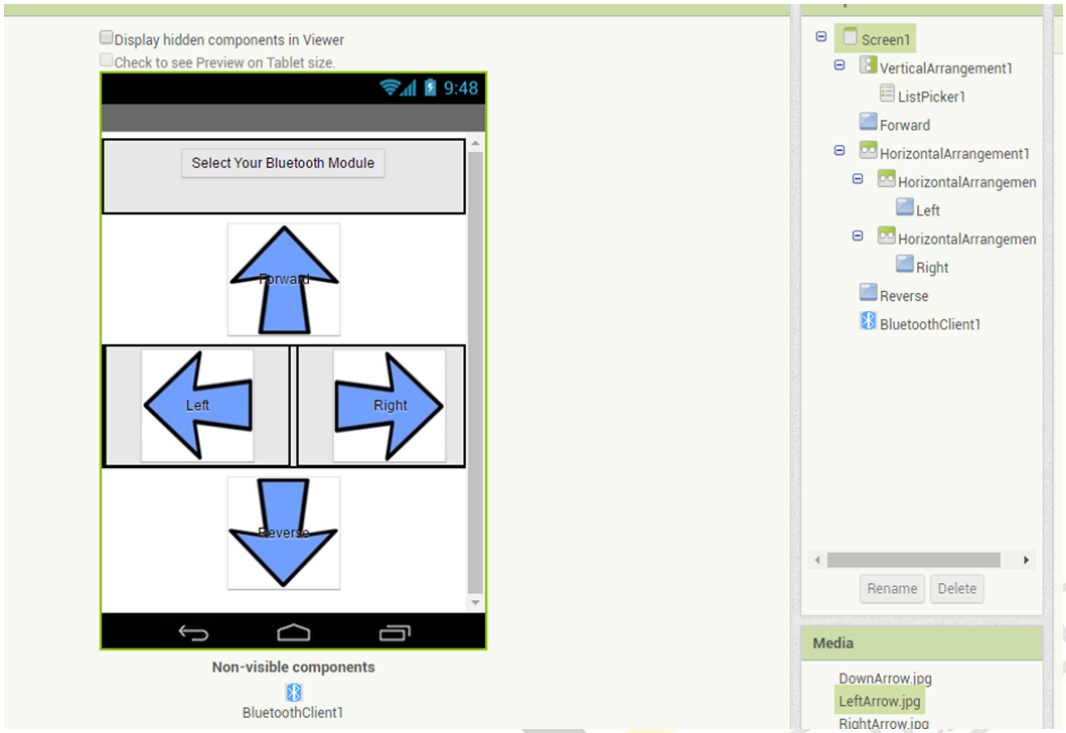


How to change button pic (2nd Step)

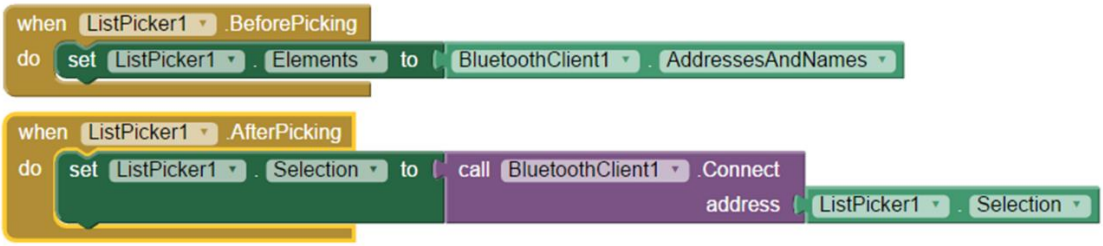




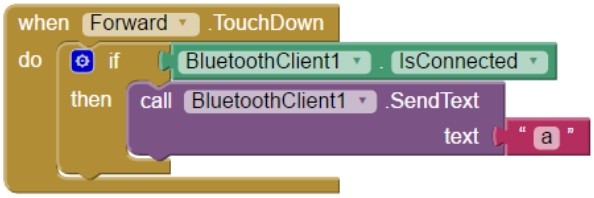
Final Look

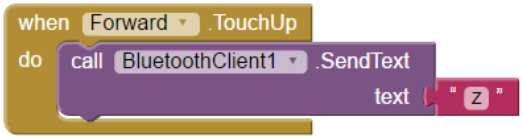


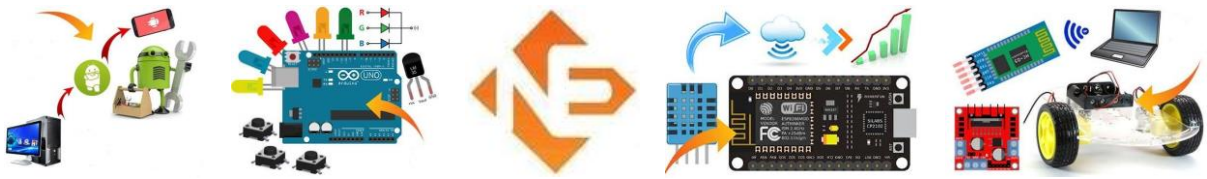
Block (1st Part) – Connecting to Bluetooth



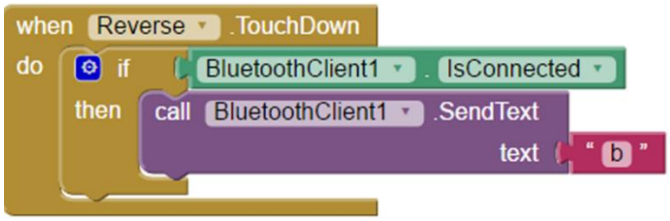
Block (2nd Part) – Move Forward

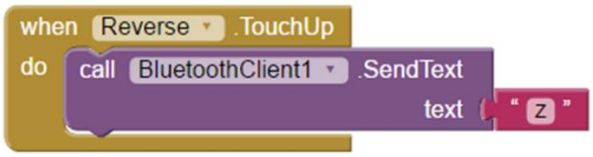




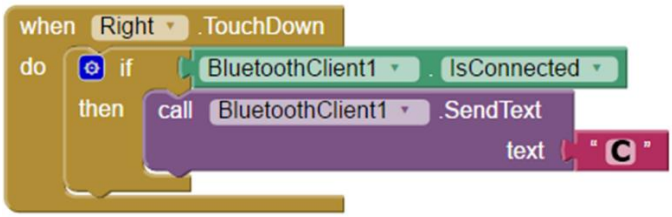


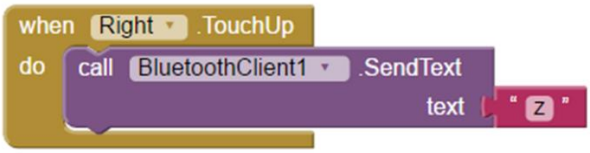
Block (3rd Part) – Move Backward



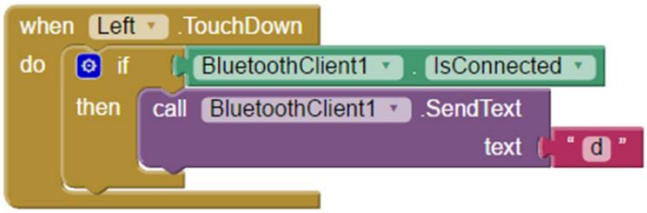


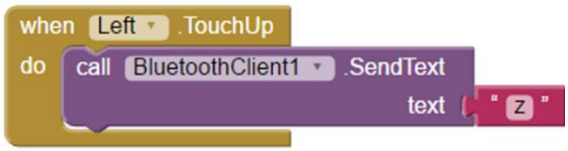
Block (4th Part) – Move Right

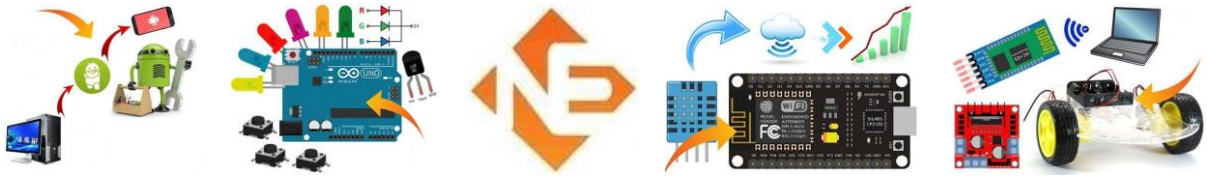




Block (5th Part) – Move Left

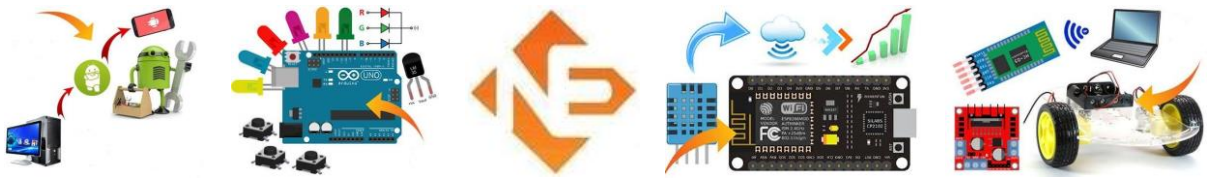






Programming Code

```
#include <SoftwareSerial.h>
#define M1 4
#define E1 5
#define M2 7
#define E2 6

SoftwareSerial blue(10,11);

void setup() {
  blue.begin(9600);
  Serial.begin(9600);
  pinMode(M1, OUTPUT);
  pinMode(E1, OUTPUT);
  pinMode(M2, OUTPUT);
  pinMode(E2, OUTPUT);
}

void loop() {
char in = blue.read();
Serial.println(in);
if(in == 'a'){
  digitalWrite(E1, HIGH);
  digitalWrite(M1, HIGH);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, HIGH);
  delay(100);
}
if(in == 'b'){
  digitalWrite(E1, HIGH);
  digitalWrite(M1, LOW);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, HIGH);
  delay(100);
}
```

```
if(in == 'c'){
  digitalWrite(E1, HIGH);
  digitalWrite(M1, HIGH);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, LOW);
  delay(100);
}
if(in == 'd'){
   digitalWrite(E1, HIGH);
  digitalWrite(M1, LOW);
  digitalWrite(E2, HIGH);
  digitalWrite(M2, LOW);
  delay(100);
}
if(in == 'z'){
   digitalWrite(E1, LOW);
  digitalWrite(M1, LOW);
  digitalWrite(E2, LOW);
  digitalWrite(M2, LOW);
  delay(100);
}
}
```