



**TEE3206**

# Design and Application of IoT

การออกแบบและประยุกต์ระบบไอโอที

## Unit 1: NodeMCU Overview

---

ผู้สอน: อาจารย์ ดร.ธัชชนนท์ ชุ่มแอนด์  
สาขาวิชาเทคโนโลยีไฟฟ้า มรภ.สวนสุนันทา



# Outline



## NodeMCU ESP8266

ข้อมูลเบื้องต้นของ  
NodeMCU ESP8266



## โครงสร้างของ IDE

โครงสร้างของโปรแกรม  
Arduino IDE



## การทดสอบ NodeMCU

การไหลทดสอบการเชื่อมต่อ  
NodeMCU กับคอมพิวเตอร์



## Arduino IDE

การติดตั้งโปรแกรม Arduino  
IDE



## การติดตั้ง NodeMCU

การเชื่อมต่อ NodeMCU กับ  
คอมพิวเตอร์



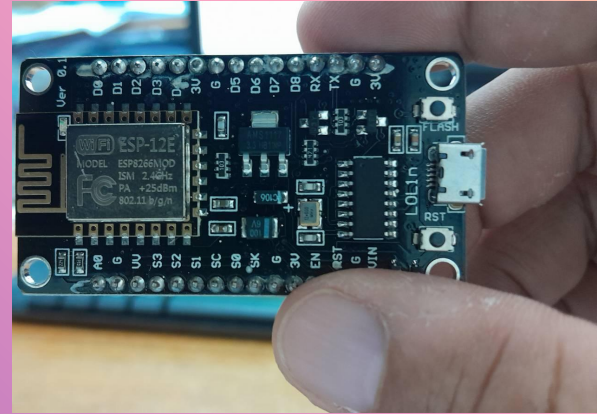
## โครงสร้างการควบคุมของ IDE

โครงสร้างการควบคุมของ  
โปรแกรม Arduino IDE

# NodeMCU EPS8266

# NodeMCU

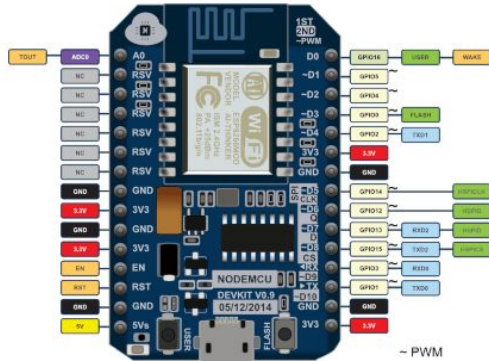
- NodeMCU คือ บอร์ด microcontroller คล้าย Arduino
- สามารถเชื่อมต่อกับ WiFi ได้
- มีราคาถูกมาก
- เหมาะแก่ผู้เริ่มต้นศึกษา IoT, เลิกทรอนิกส์ หรือนำไปใช้จริงในโปรเจคต่าง ๆ



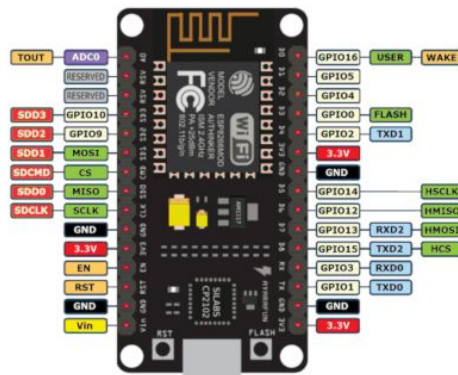
# NodeMCU

- NodeMCU มีชิป ESP8266 ทำให้เชื่อมต่อกับระบบ WiFi ได้
- เขียนโปรแกรมด้วย Arduino IDE ได้

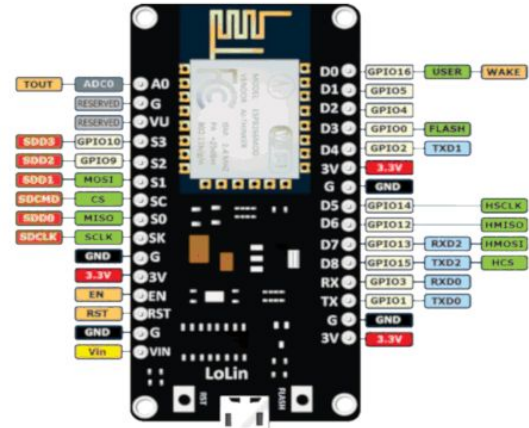
NodeMCU Devkit V0.9  
aka NodeMCU V1



NodeMCU Devkit V1.0  
aka NodeMCU V2



Lolin NodeMCU  
aka "NodeMCU V3"



# คุณสมบัติของ NodeMCU ESP8266

Microcontroller :

Clock Speed : 80 MHz

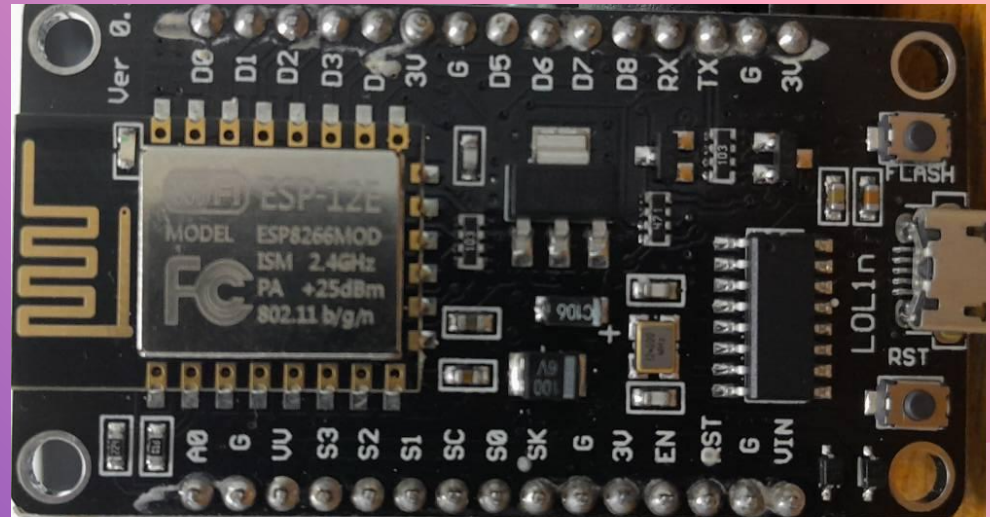
USB Connector : Micro USB

Operating Voltage: 3.3 V

Digital I/O : 11 Pins

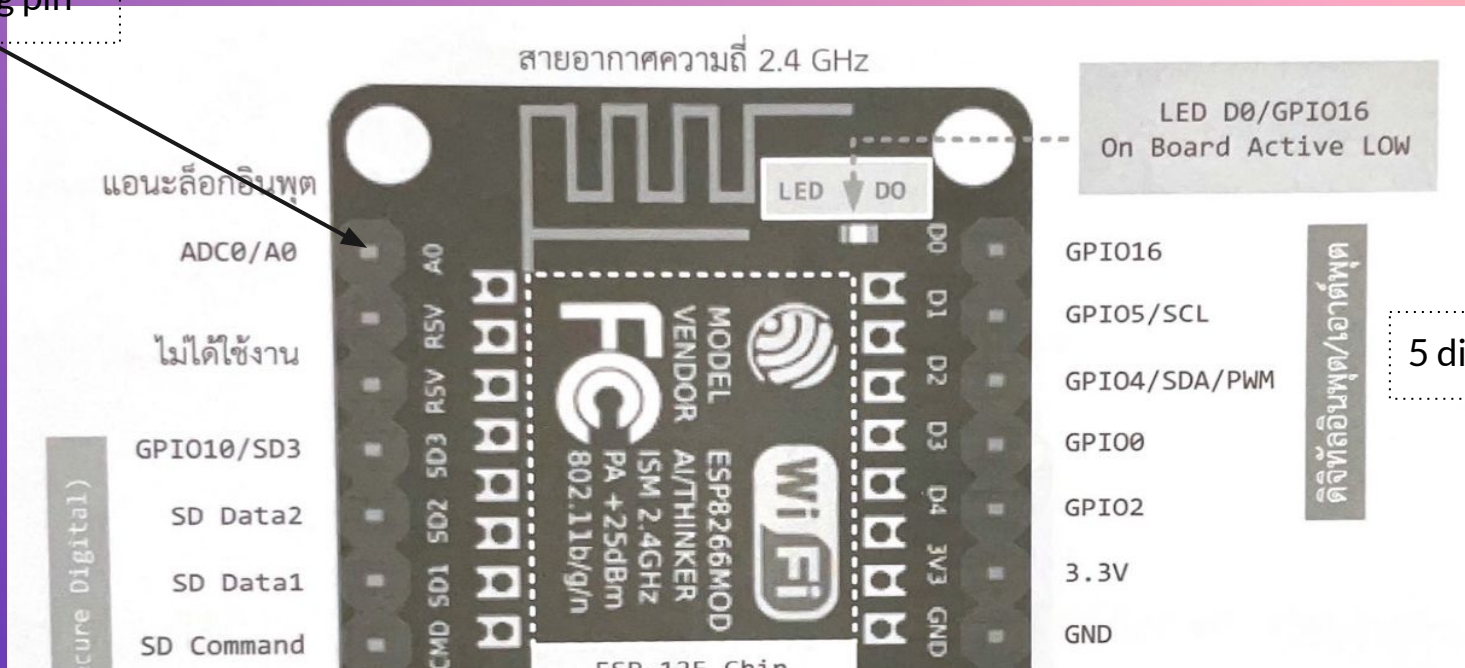
Analog Inputs : 1 Pin

WiFi : Built-in 802.11 b/g/n



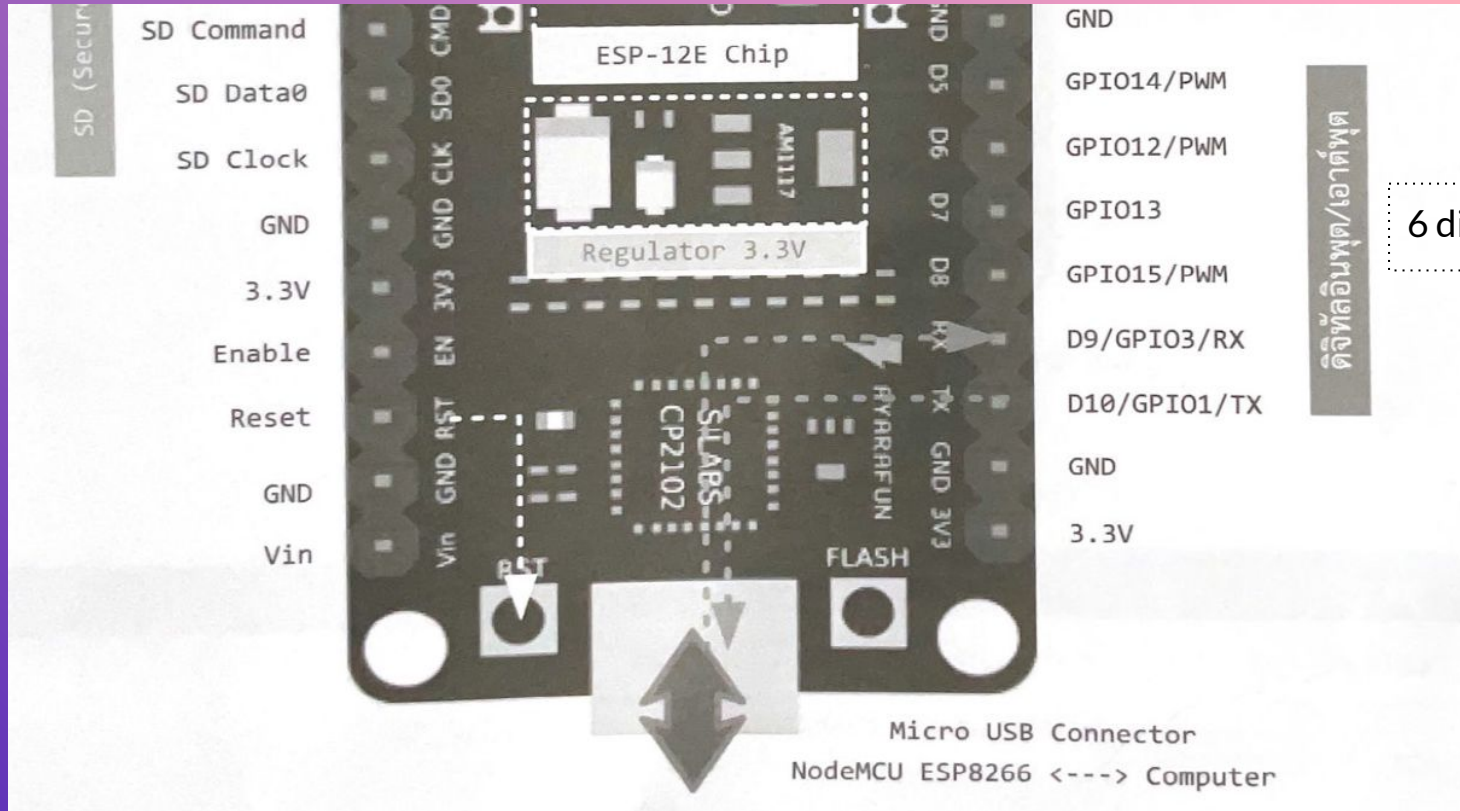
# ขาสัญญาณ NodeMCU ESP8266 (1)

1 analog pin



5 digital pins

# ขาสัญญาณ NodeMCU ESP8266 (2)



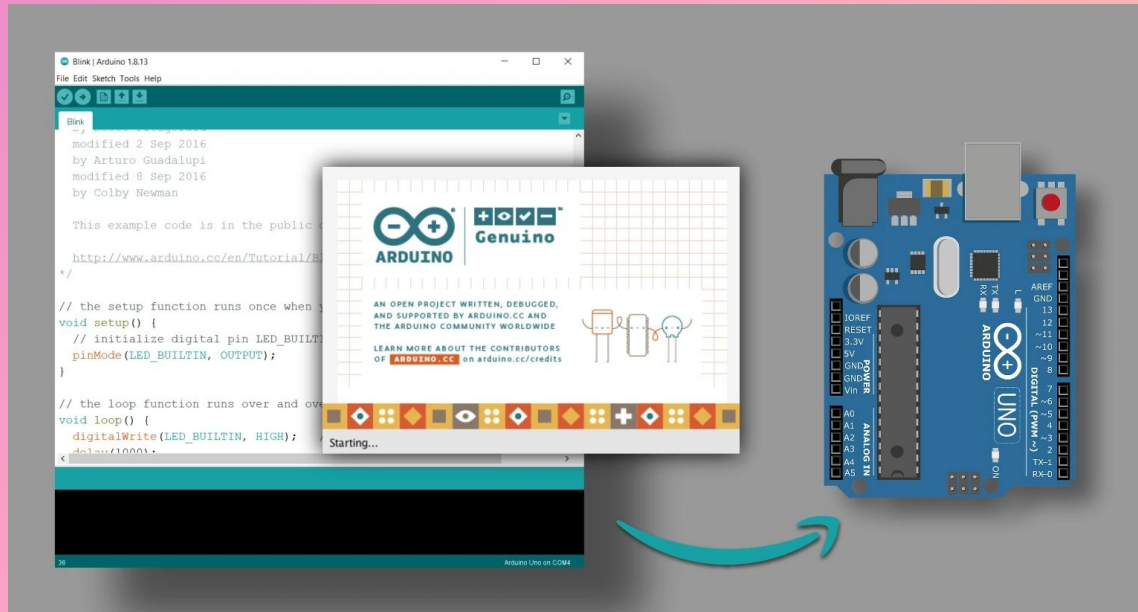
# Arduino IDE



Arduino เป็นแพลตฟอร์มอิเล็กทรอนิกส์ ที่ใช้ซอฟต์แวร์แบบเปิดเผยโค้ดต้นฉบับ (open source code) ซึ่งวางอยู่บนแนวคิดที่อาศัยความร่วมมือของนักพัฒนาทั่วโลกเพื่อสร้างซอฟต์แวร์ที่ดีกว่า

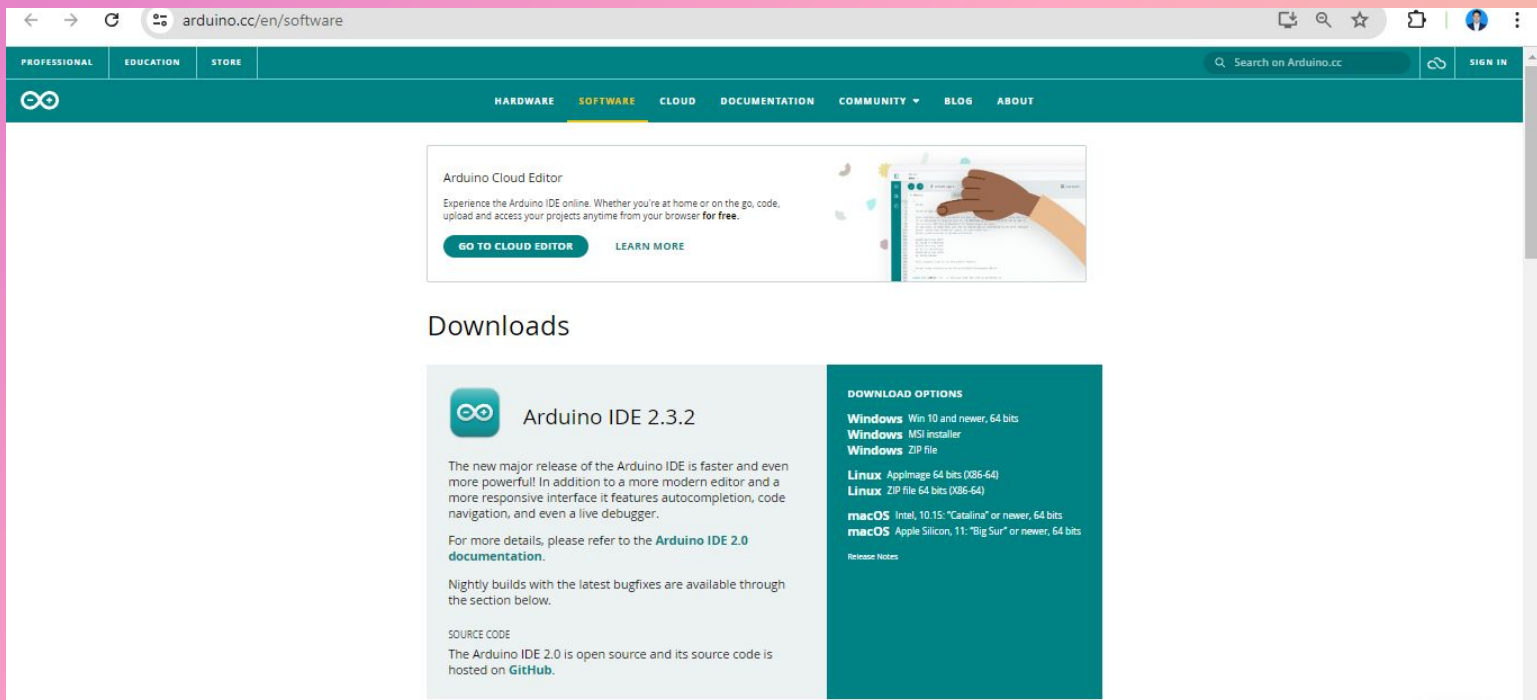
# Arduino IDE Installing

ซอฟต์แวร์สำหรับพัฒนา Arduino (IDE) เป็นสิ่งจำเป็นสำหรับการใช้งาน Arduino ใช้สำหรับเขียนโค้ดโปรแกรมและอัปโหลดลงไปที่บอร์ด Arduino



# Arduino IDE Installing

ดาวน์โหลดซอฟต์แวร์ Arduino IDE url: <https://www.arduino.cc/en/software>



The screenshot shows the Arduino website's software page. The browser address bar displays 'arduino.cc/en/software'. The navigation menu includes 'PROFESSIONAL', 'EDUCATION', 'STORE', 'HARDWARE', 'SOFTWARE' (highlighted), 'CLOUD', 'DOCUMENTATION', 'COMMUNITY', 'BLOG', and 'ABOUT'. A search bar is located on the right. The main content area features a promotional banner for 'Arduino Cloud Editor' with a 'GO TO CLOUD EDITOR' button and a 'LEARN MORE' link. Below this is a 'Downloads' section for 'Arduino IDE 2.3.2'. The download options are listed as follows:

DOWNLOAD OPTIONS	
Windows	Win 10 and newer, 64 bits
Windows	MSI installer
Windows	ZIP file
Linux	AppImage 64 bits (X86-64)
Linux	ZIP file 64 bits (X86-64)
macOS	Intel, 10.15: "Catalina" or newer, 64 bits
macOS	Apple Silicon, 11: "Big Sur" or newer, 64 bits

Release Notes

**Arduino IDE 2.3.2**

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

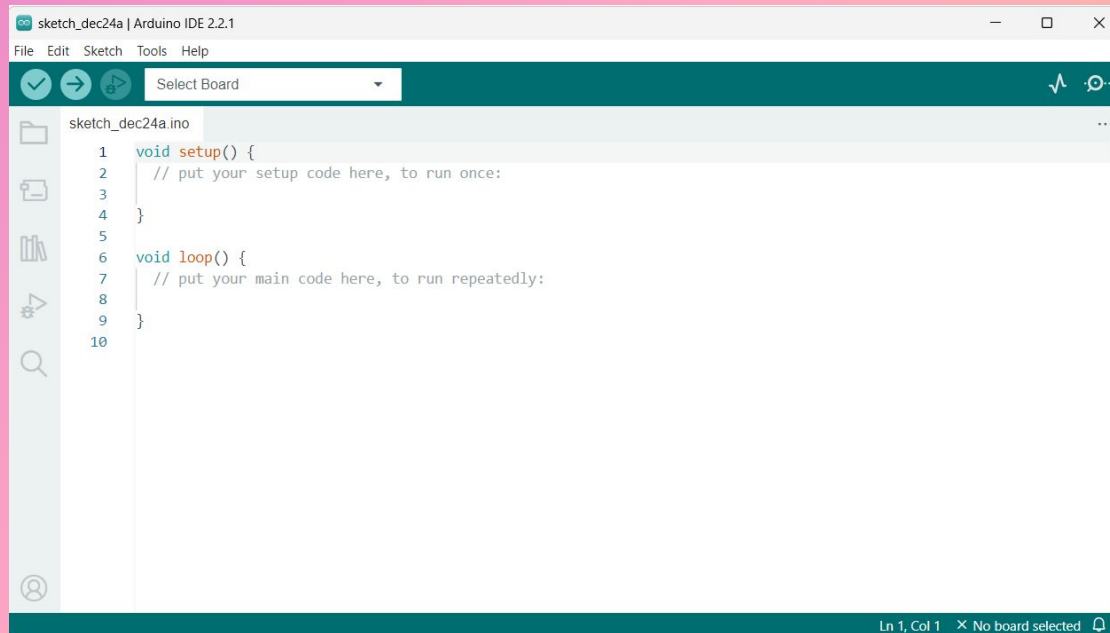
Nightly builds with the latest bugfixes are available through the section below.

**SOURCE CODE**

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

# Arduino IDE Installing

- ซอฟต์แวร์ IDE จะมีให้เลือกดาวน์โหลด 3 ประเภทคือ Windows, Mac และ Linux
- เมื่อติดตั้งโปรแกรมเรียบร้อยแล้ว เราจะเห็นหน้าต่างหลักของโปรแกรมดังรูป



The screenshot shows the Arduino IDE 2.2.1 interface. The window title is "sketch\_dec24a | Arduino IDE 2.2.1". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for "Check", "Next", "Run", and a "Select Board" dropdown menu. The main editor area displays a sketch named "sketch\_dec24a.ino" with the following code:

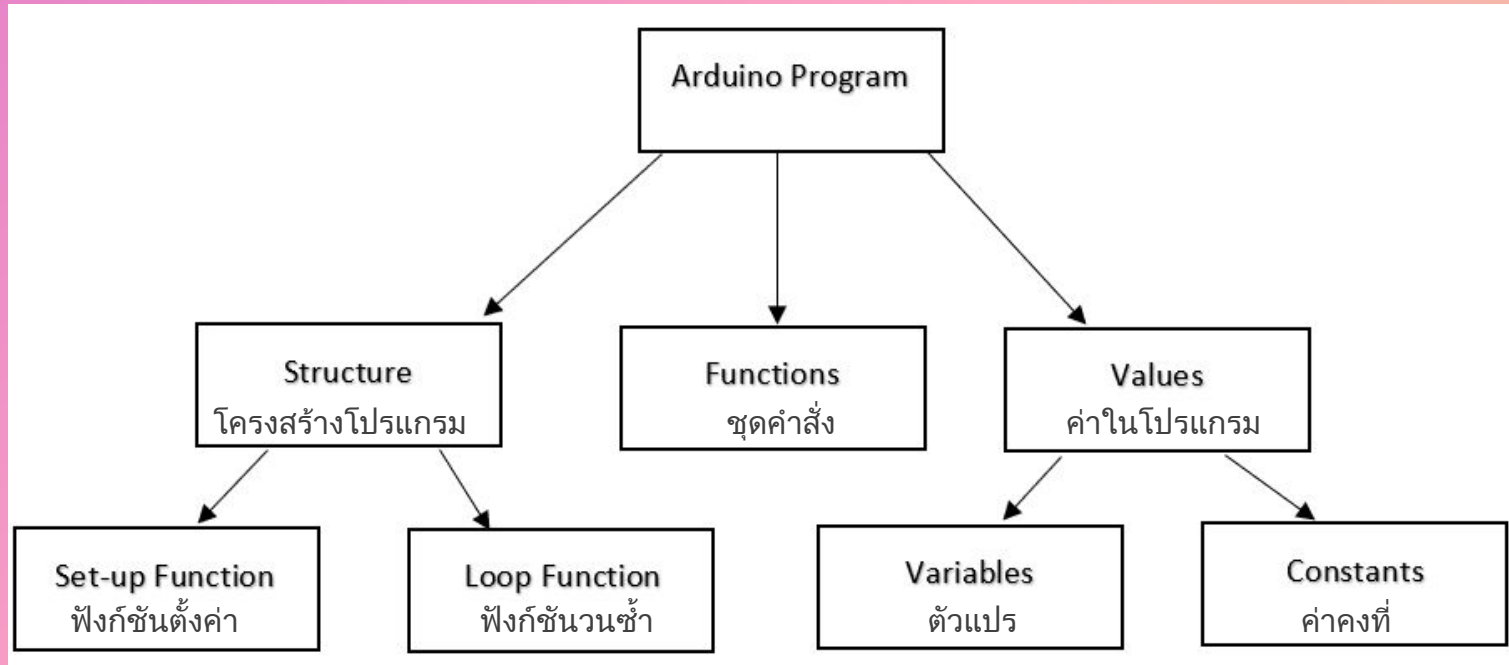
```
1 void setup() {  
2   // put your setup code here, to run once:  
3  
4 }  
5  
6 void loop() {  
7   // put your main code here, to run repeatedly:  
8  
9 }  
10
```

The status bar at the bottom indicates "Ln 1, Col 1" and "No board selected".

# โครงสร้างพื้นฐานของ **Arduino IDE**

# โครงสร้างของ Arduino IDE

โปรแกรม Arduino ประกอบด้วยองค์ประกอบ 3 ส่วน



# โครงสร้างของ Arduino IDE

```
1 void setup() {  
2   // put your setup code here, to run once:  
3  
4 }  
5  
6 void loop() {  
7   // put your main code here, to run repeatedly:  
8  
9 }  
10
```

- **void setup ()** เป็นคำสั่งเริ่มต้นของบอร์ด Arduino โดย Arduino จะถูกดำเนินการ (executed) 1 ครั้งเท่านั้น ภายใน { } ของ void setup นอกจากนี้ยังสามารถตั้งค่าตัวแปร กำหนด PIN เริ่มใช้ Library ฯลฯ
- **void loop ()** จะอยู่หลังจาก void setup () เป็นส่วนที่เราต้องการดำเนินการซ้ำ คำสั่งใน { } ของ void setup ของ void loop จะถูกดำเนินการซ้ำไปเรื่อยๆ จนกว่าจะปิดอุปกรณ์

# โครงสร้างของ Arduino IDE

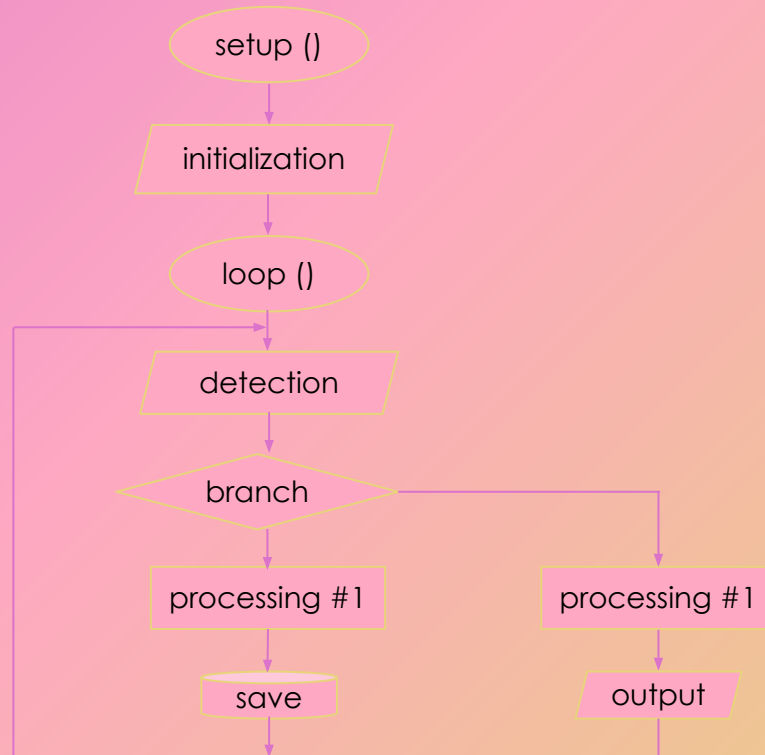
เครื่องหมาย “{ }” เรียกว่าวงเล็บ (bracket) เป็นสัญลักษณ์ที่ใช้ขยายขอบเขตของคำสั่ง ความหมายคือถ้าไม่มี { }

คำสั่งจะถูกดำเนินการเฉพาะบรรทัดแรกเท่านั้น ขณะที่ถ้ามี { } คำสั่งภายใน { } ทั้งหมดจะถูกดำเนินการ

```
1 void setup() {
2   pinMode(LED_BUILTIN, OUTPUT);
3 }
4 void loop() {
5   digitalWrite(LED_BUILTIN, HIGH);
6   delay(1000);
7   digitalWrite(LED_BUILTIN, LOW);
8   delay(1000);
9 }
10
```

# โครงสร้างของ Arduino IDE

รูปด้านล่างเป็นตัวอย่าง flowchart ของ Arduino ที่มีการแบ่งเงื่อนไขใน loop



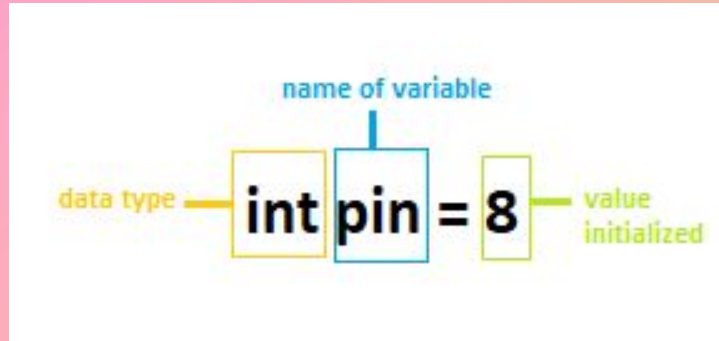
# ตัวแปร (variable) ของ Arduino IDE

- โดยปกติ Arduino จะใช้ int (integer, 16 bit) ในการระบุจำนวน
- แต่ยังสามารถใช้ตัวแปรประเภทอื่นในตารางด้านล่าง

Type	Byte	Range	
int	2	-32768 ถึง 32767	จำนวนบวกและจำนวนลบ
unsigned int	2	0 ถึง 65535	เหมือน int แต่เฉพาะจำนวนบวก
long	4	-2147483648 ถึง 2147483647	จำนวนบวกและจำนวนลบขนาดใหญ่
unsigned long	4	0 ถึง 4294967295	จำนวนบวกขนาดใหญ่
float	4	-3.4028235E+38 ถึง 3.4028235E+38	เลขทศนิยม ใช้รับค่าจากการวัดจริง
double	4	เหมือนกับ float	เหมือนกับ float
boolean	1	false (0) หรือ true (1)	ค่าจริงหรือเท็จ
char	1	-128 ถึง 127	ตัวอักษร หรือ ค่าระหว่าง -128 และ 127
byte	1	0 ถึง 255	เหมือนกับ char แต่เป็นบวกเท่านั้น

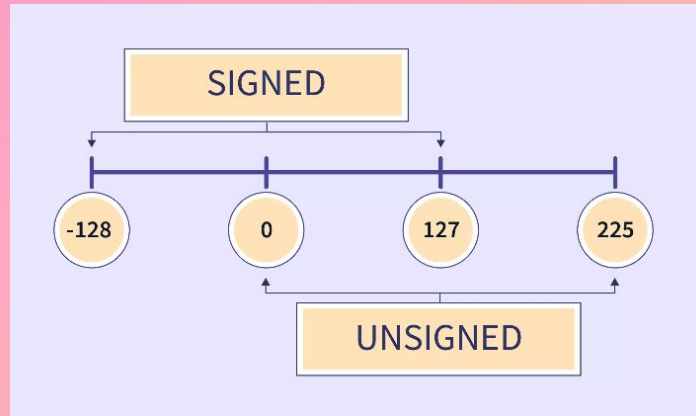
## ตัวแปรประเภท int

- ถ้าไม่ได้คำนึงถึงสมรรถนะหรือประสิทธิภาพการเก็บข้อมูล และถ้าค่าสูงสุด และค่าต่ำสุดอยู่ในขอบเขต เราไม่จำเป็นต้องใช้ตัวแปรที่เป็นเลขทศนิยม และควรจะใช้ตัวแปรประเภท `int`
- ตัวอย่าง Arduino ส่วนใหญ่ใช้ตัวแปรประเภท `int`



## ตัวแปรแบบ signed และ unsigned

- บางครั้งจำเป็นต้องมีจำนวนลบ จำนวนจึงถูกแบ่งประเภทเป็นแบบมีเครื่องหมาย +- (signed) และไม่มีเครื่องหมาย (ไม่ติดลบ) (unsigned)
- ปกติไม่ว่าค่าจะเป็น +- หรือเป็น + อย่างเดียว เราจะใช้ signed ยกเว้นค่าเป็นบวกและค่าสูงสุดเกินขอบเขตของ signed จึงจะใช้ unsigned



## ตัวแปรประเภท boolean

- Boolean จะเป็น เท็จ (false) หรือ จริง (true) อย่างใดอย่างหนึ่ง
- ตัวแปรประเภทนี้ใช้คล้ายกับเช็คสวิทช์ถูกกดหรือไม่กด
- จะคล้ายกับการแสดงสถานะของ pin ขาออกของ Arduino ซึ่งจะแสดงค่าแรงดัน High หรือ Low แทน จริง หรือ เท็จ

<b>True</b>	<b>False</b>
<b>1</b>	<b>0</b>
<b>HIGH</b>	<b>LOW</b>

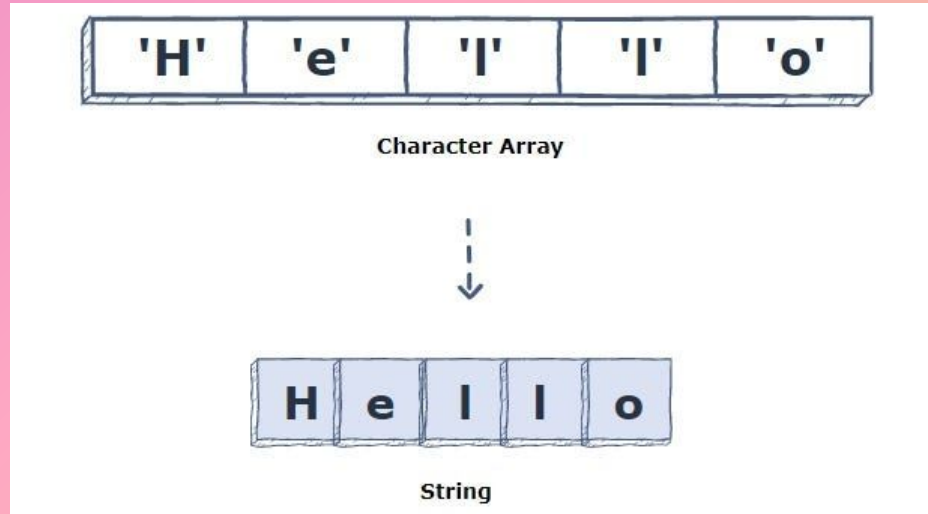
## ตัวแปรประเภท float และ double

- float และ double ใช้สำหรับข้อมูลที่เป็นจำนวนทศนิยมและใช้เมื่อรับค่าประมาณของค่าจากการวัด
- ตัวแปรทั้งสองประเภทนี้มักจะใช้เพื่อคำนวณค่าที่ได้จากเซนเซอร์



# ตัวแปรประเภท char และ string

- **char** เก็บข้อมูลอักขระในช่องเก็บข้อมูล อาจเป็นตัวอักษร หรือ ตัวเลข
- **string** กำหนดตัวแปร string (อักขระหรือตัวเลขที่เรียงต่อกัน) จะกำหนดจากตัวแปรประเภท **char** มาเรียงต่อกัน



## ฟังก์ชันของ Arduino IDE

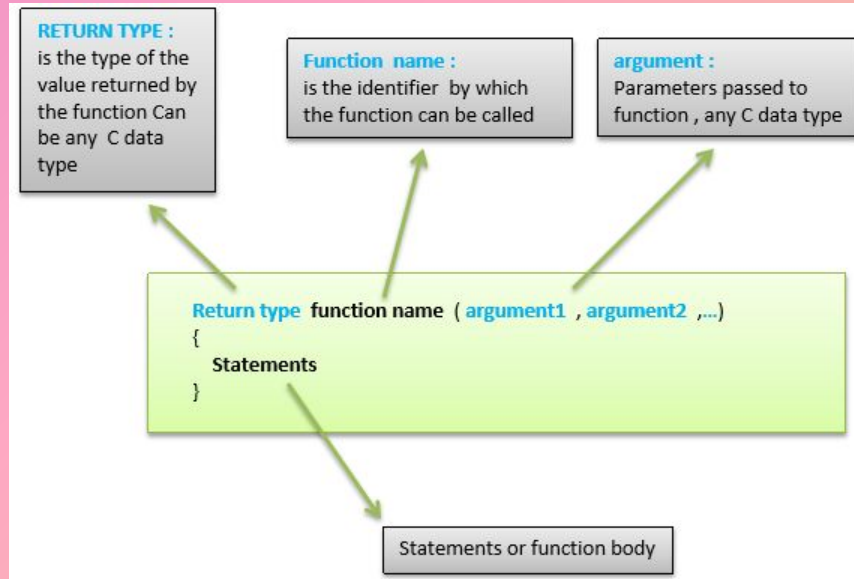
- ฟังก์ชันช่วยให้จัดโครงสร้างโปรแกรมที่เป็นชุดคำสั่ง (code) เพื่อทำงานแต่ละงานได้
- กรณีทั่วไป เราจะสร้างฟังก์ชันเมื่อจำเป็นต้องดำเนินการเดียวกันหลายครั้งในโปรแกรม



# ฟังก์ชันของ Arduino IDE

ฟังก์ชันจะประกอบด้วย ประเภท ชื่อฟังก์ชัน ตัวแปรขาเข้าของฟังก์ชัน และคำสั่งภายในฟังก์ชัน

(statement)



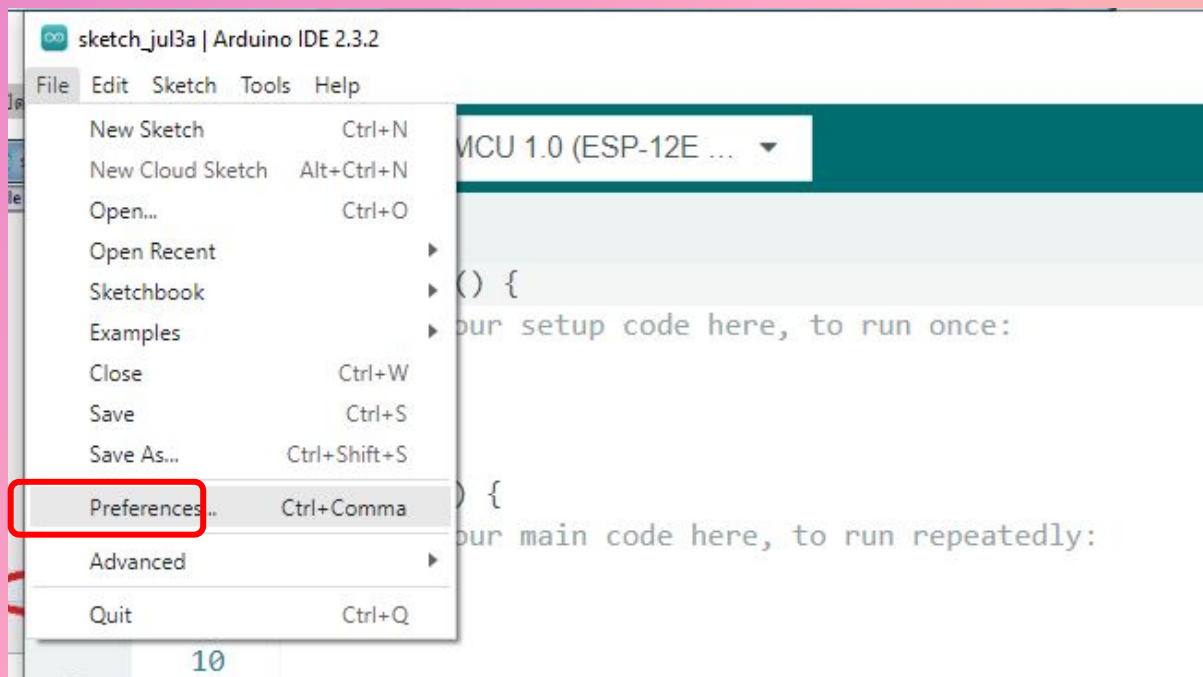
# ฟังก์ชันของ Arduino IDE

- ภายในคำสั่ง loop () (นั่นคือภายใน { } ที่ต่อจากคำสั่ง loop ()) ข้อความที่ไม่ใช่ตัวแปรจะเรียกว่าฟังก์ชัน
- ช่องว่างที่อยู่ใน () ที่ติดกับชื่อฟังก์ชัน เกิดจากการที่ฟังก์ชันเป็นประเภท void function ( ฟังก์ชันที่ไม่ต้องมี input)
- เราไม่สามารถใช้ชื่อของฟังก์ชันต่อไปนี้ : setup, loop, if, for, switch เพราะคำเหล่านี้เป็นฟังก์ชันพื้นฐานที่ถูกกำหนดไว้แล้ว

```
void loop() {  
  readAD();           // Read the analog port  
  outPORT();         // Control the output to digital port  
}
```

# การติดตั้ง NodeMCU

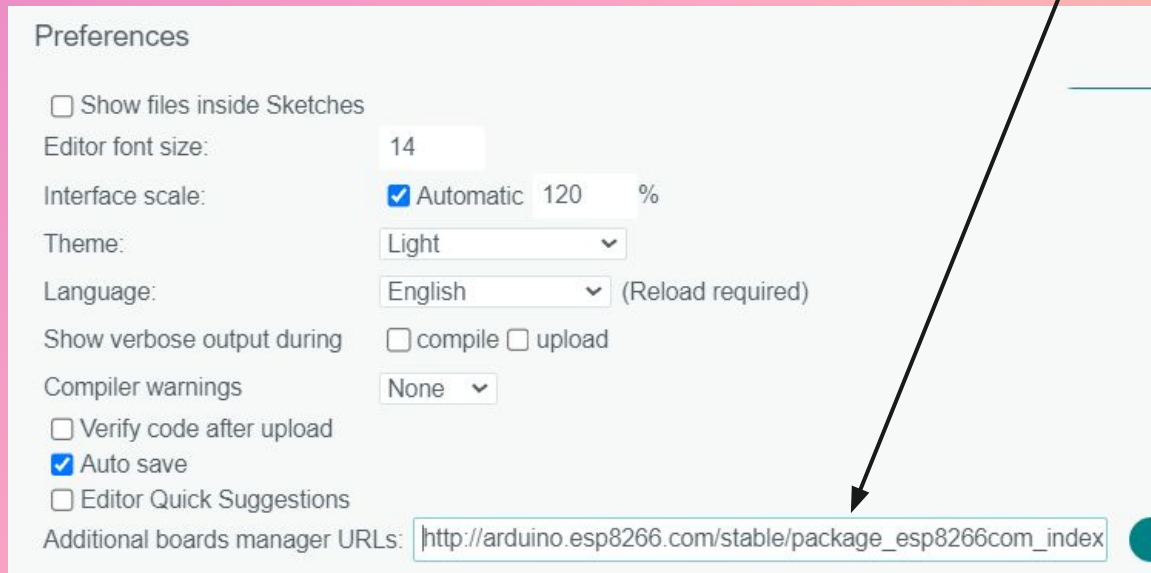
1. เปิดโปรแกรม Arduino IDE คลิกไปที่เมนู File -> Preferences



# การติดตั้ง NodeMCU

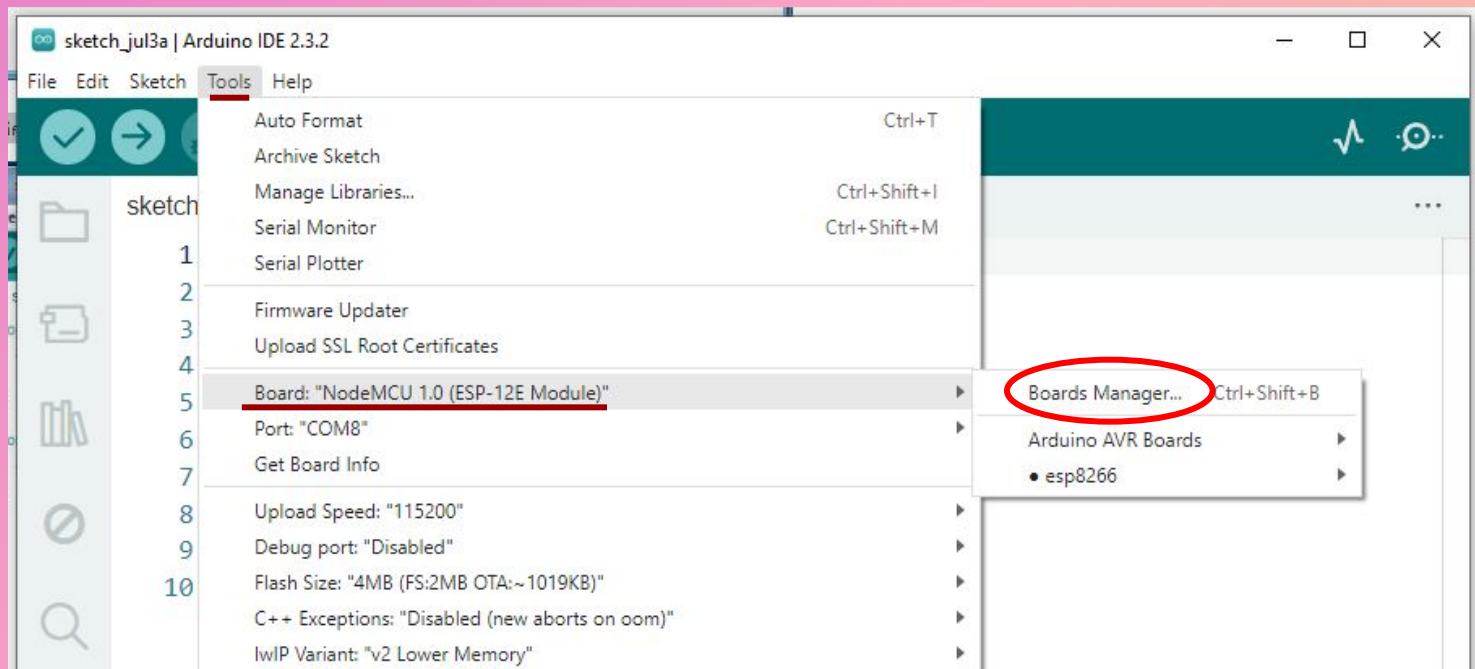
2. เพิ่ม [http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)

ลงในช่อง **Additional Boards Manager URLs** ดังภาพ แล้วกด **OK**



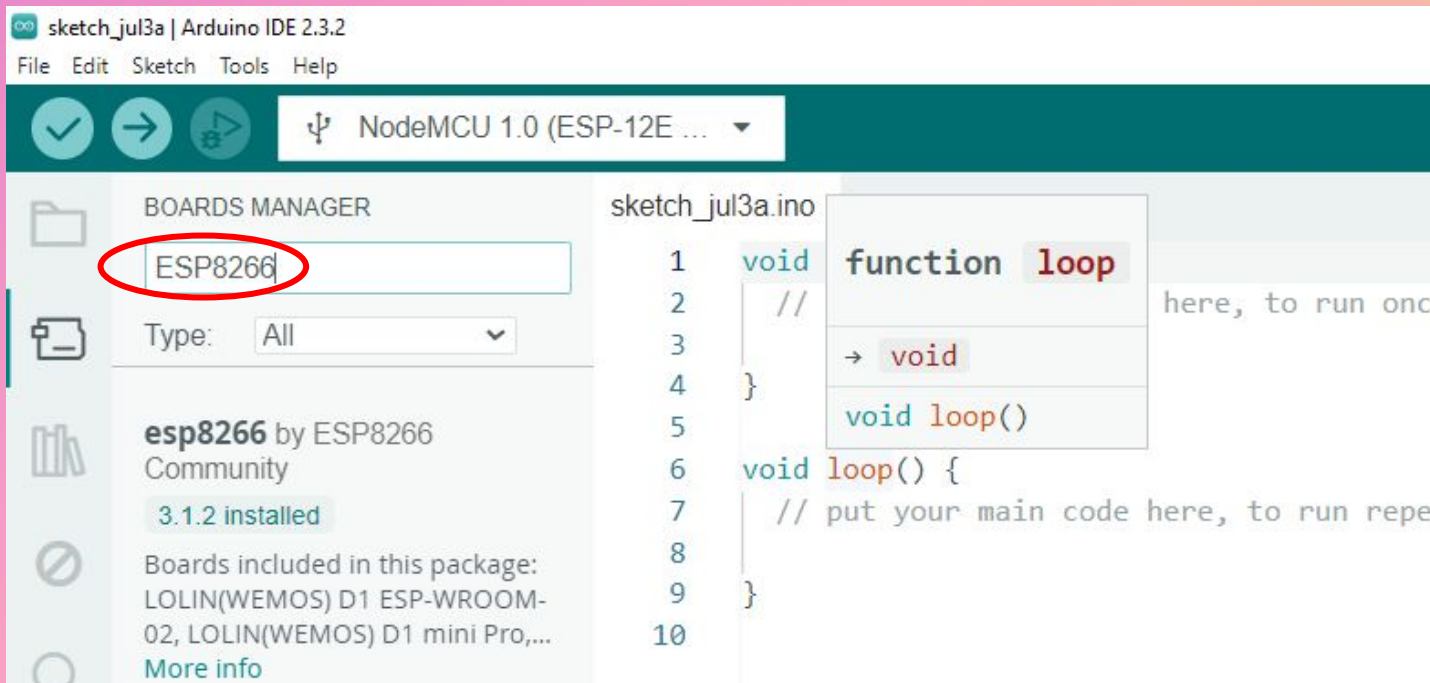
# การติดตั้ง NodeMCU

3. คลิกไปที่เมนู Tools -> Board -> Board Manager



# การติดตั้ง NodeMCU

4. พิมพ์คำว่า ESP8266 ลงในช่อง และกด Install



The screenshot shows the Arduino IDE interface. The title bar reads "sketch\_jul3a | Arduino IDE 2.3.2". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains icons for a checkmark, a right arrow, and a play button, followed by a search icon and a dropdown menu showing "NodeMCU 1.0 (ESP-12E ...)".

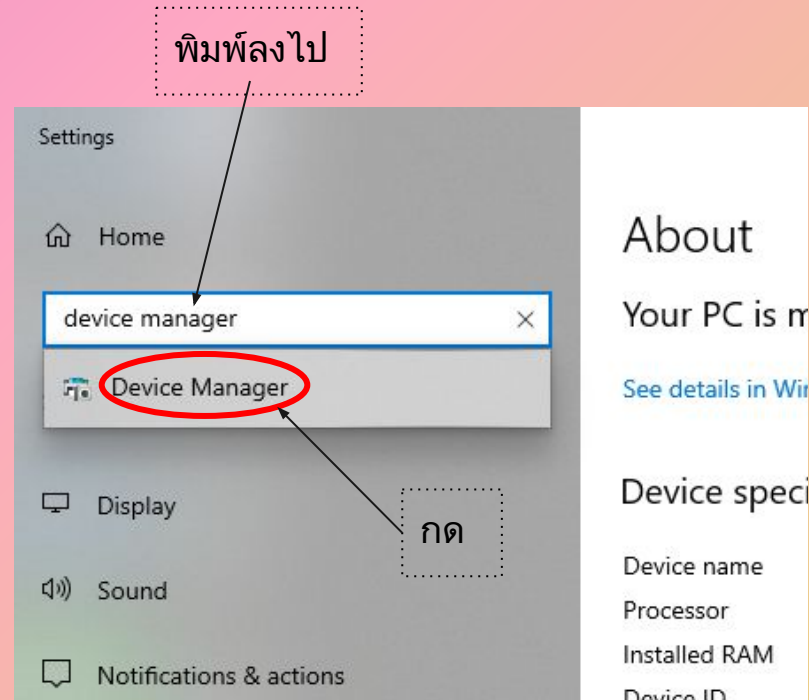
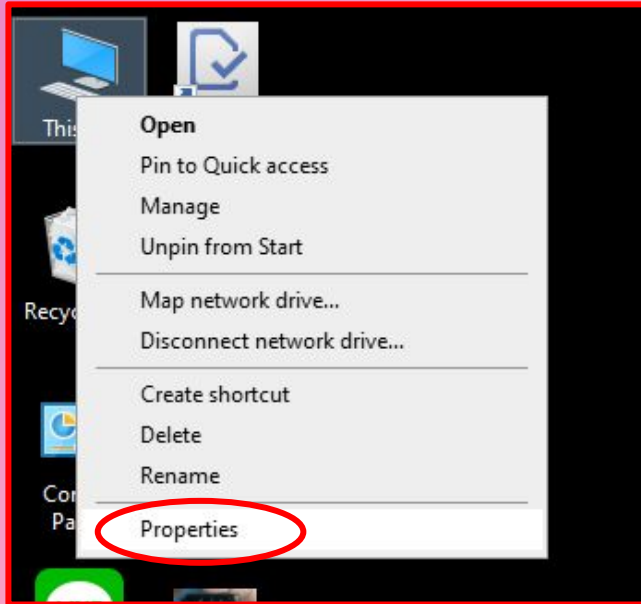
The "BOARDS MANAGER" panel is active. The search bar contains the text "ESP8266", which is circled in red. Below the search bar, the "Type:" dropdown is set to "All". The search results show "esp8266 by ESP8266" with a "Community" tag and a "3.1.2 installed" badge. Below this, it lists "Boards included in this package: LOLIN(WEMOS) D1 ESP-WROOM-02, LOLIN(WEMOS) D1 mini Pro,..." and a "More info" link.

The main editor window shows the code for "sketch\_jul3a.ino". The code is as follows:

```
1 void function loop
2 // here, to run onc
3
4 }
5
6 void loop() {
7 // put your main code here, to run repe
8
9 }
10
```

# การติดตั้ง NodeMCU

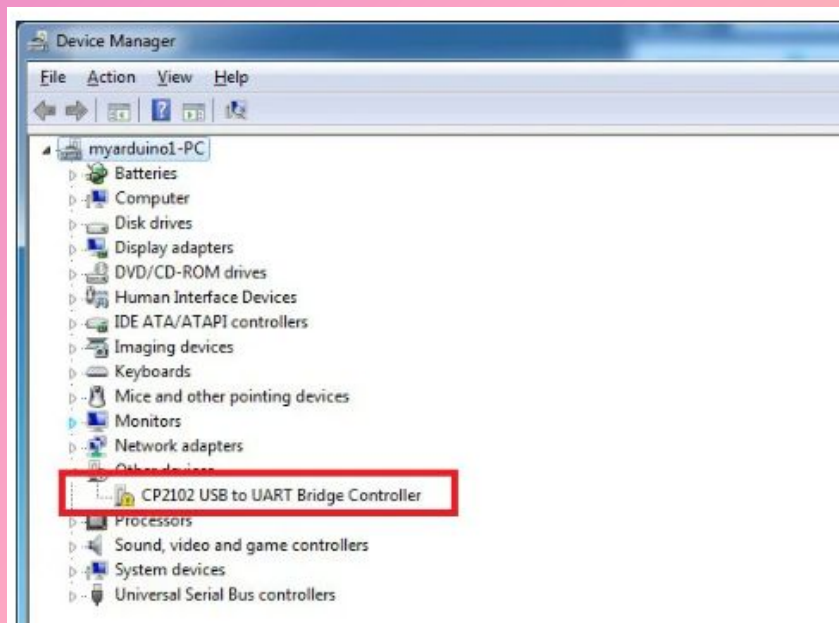
5. เสียบบอร์ด NodeMCU/ESP8266 เข้ากับคอมพิวเตอร์ จากนั้นไปที่ Properties -> Device Manager



# การติดตั้ง NodeMCU

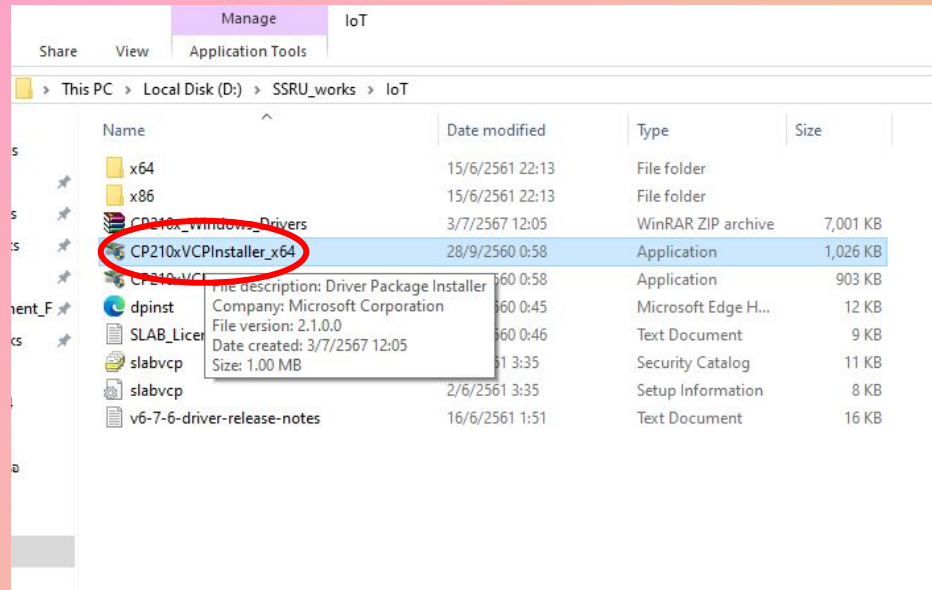
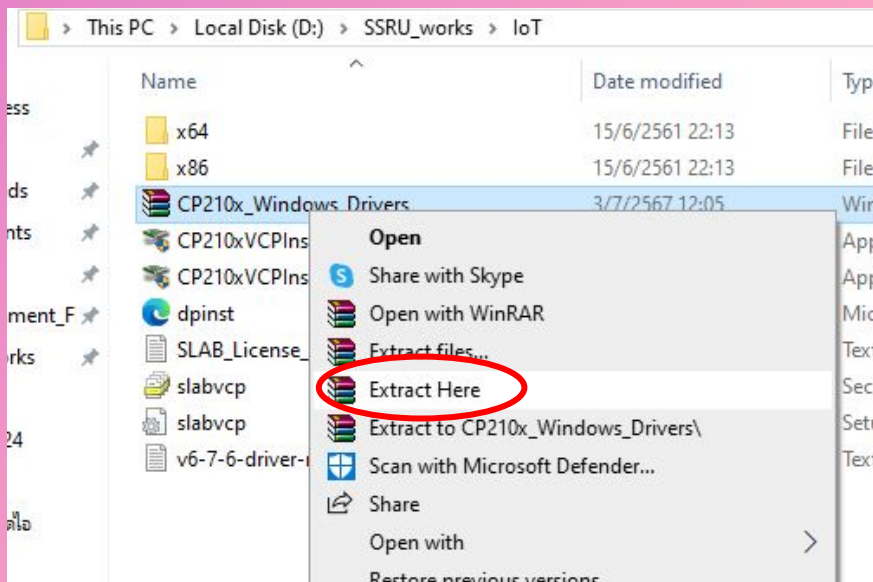
6. ขึ้นเครื่องหมายตกใจ ให้เราติดตั้ง Driver ก่อน โหลดได้จากลิงค์

[http://www.mediafire.com/file/l47semi8ek3o978/CP210x\\_Windows\\_Drivers.zip/file](http://www.mediafire.com/file/l47semi8ek3o978/CP210x_Windows_Drivers.zip/file)



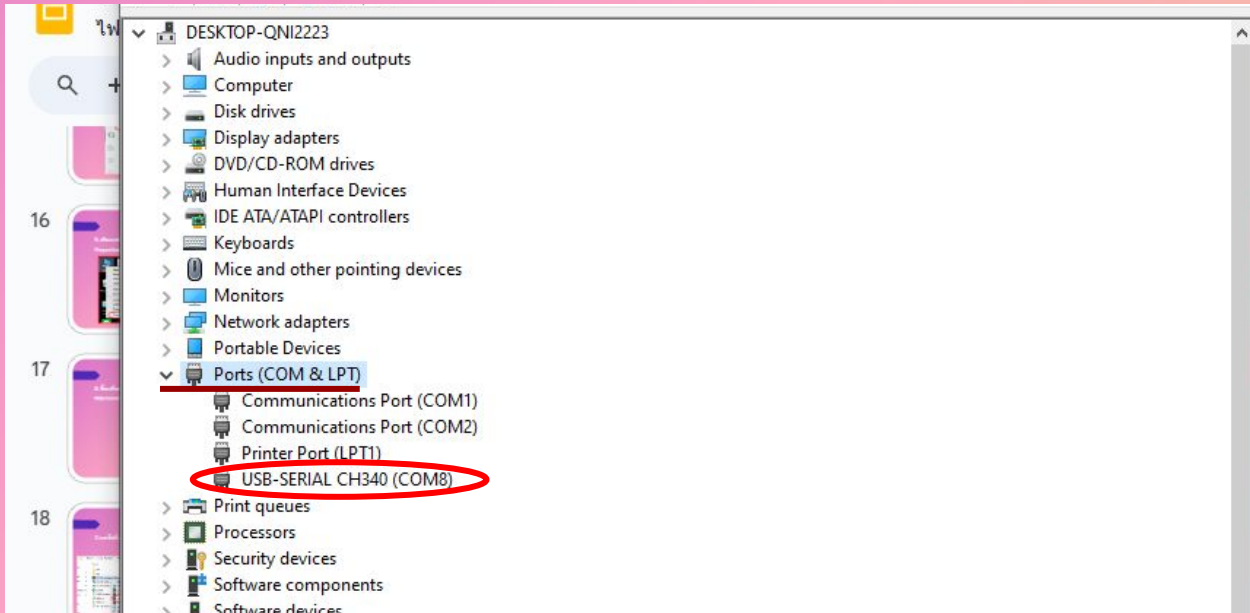
# การติดตั้ง NodeMCU

## 7. แยกไฟล์ แล้ว install



# การติดตั้ง NodeMCU

8. กลับไปที่ Device Manager ตรวจสอบที่ Ports จะเห็นว่าตอนนี้  
คอมพิวเตอร์รู้จัก NodeMCU/ESP8266 แล้ว ให้จำหมายเลข Port ไว้



# การติดตั้ง NodeMCU

9. กลับไปที่ Arduino ide ไปที่เมนู Tools เพื่อตั้งรุ่นบอร์ด NodeMCU 1.0

The screenshot shows the Arduino IDE interface. The 'Tools' menu is open, and the 'Board' option is selected, which has opened the 'Boards Manager' window. In the 'Boards Manager' window, the 'esp8266' board family is selected, and the 'NodeMCU 1.0 (ESP-12E Module)' is highlighted with a red circle and a checkmark.

**Tools Menu:**

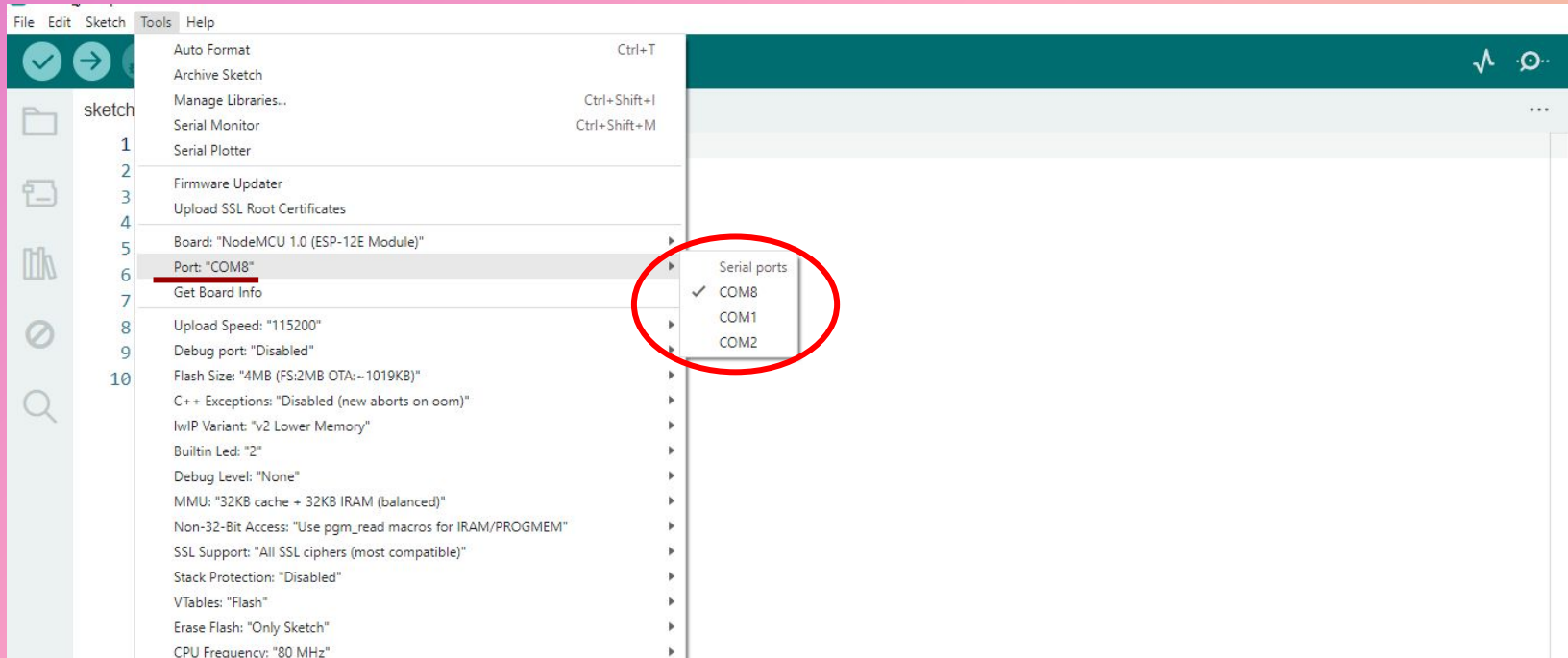
- Auto Format (Ctrl+T)
- Archive Sketch
- Manage Libraries... (Ctrl+Shift+I)
- Serial Monitor (Ctrl+Shift+M)
- Serial Plotter
- Firmware Updater
- Upload SSL Root Certificates
- Board: "NodeMCU 1.0 (ESP-12E Module)"
- Port: "COM8"
- Get Board Info
- Upload Speed: "115200"
- Debug port: "Disabled"
- Flash Size: "4MB (FS:2MB OTA:~1019KB)"
- C++ Exceptions: "Disabled (new aborts on oom)"
- lwIP Variant: "v2 Lower Memory"
- Builtin Led: "2"
- Debug Level: "None"
- MMU: "32KB cache + 32KB IRAM (balanced)"
- Non-32-Bit Access: "Use pgm\_read macros for IRAM/PROGMEM"
- SSL Support: "All SSL ciphers (most compatible)"
- Stack Protection: "Disabled"
- VTables: "Flash"
- Erase Flash: "Only Sketch"

**Boards Manager:**

- Generic ESP8285 Module
- 4D Systems gen4 IoT Range
- Adafruit Feather HUZZAH ESP8266
- Amperka WiFi Slot
- Arduino
- DOIT ESP-Mx DevKit (ESP8285)
- Digistump Oak
- ESPduino (ESP-13 Module)
- ESPECTRO Core
- ESPino (ESP-12 Module)
- ESPRESSO Lite 1.0
- ESPRESSO Lite 2.0
- ITEAD Sonoff
- Invent One
- LOLIN(WEMOS) D1 ESP-WROOM-02
- LOLIN(WEMOS) D1 R2 & mini
- LOLIN(WEMOS) D1 mini (clone)
- LOLIN(WEMOS) D1 mini Lite
- LOLIN(WEMOS) D1 mini Pro
- LOLIN(WeMos) D1 R1
- Lifely Agrumino Lemon v4
- NodeMCU 0.9 (ESP-12 Module)
- ✓ NodeMCU 1.0 (ESP-12E Module)
- Olimex MOD-WIFI-ESP8266(-DEV)
- Phoenix 1.0

# การติดตั้ง NodeMCU

## 10. เลือกหมายเลขพอร์ตให้ตรงกับใน device manager



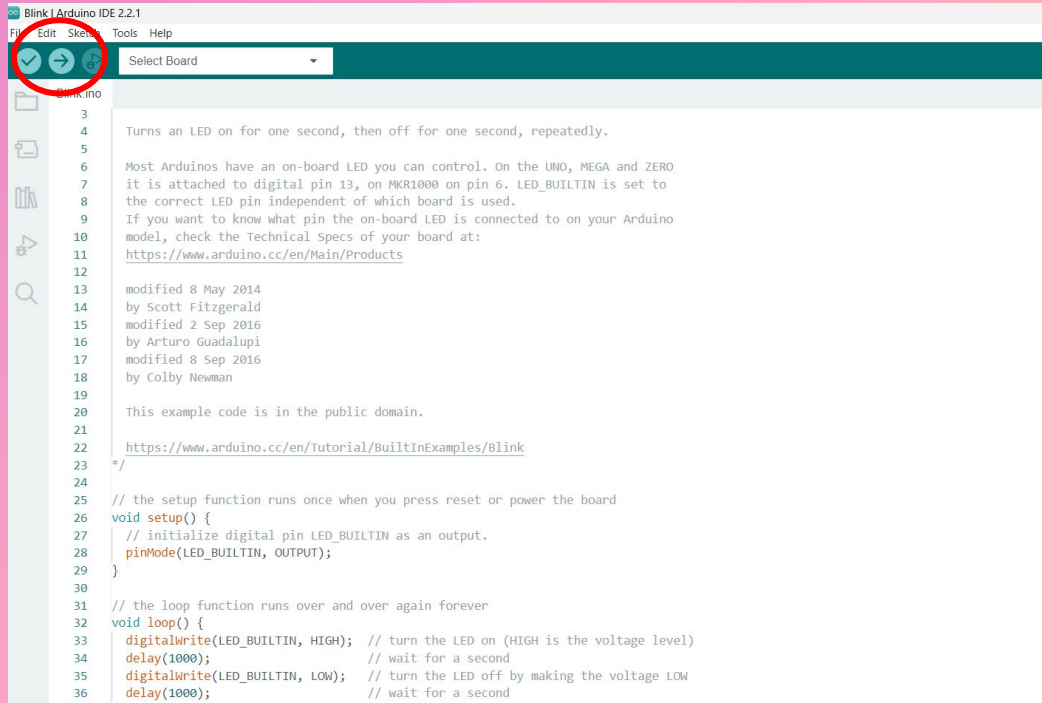
# การทดสอบ NodeMCU

- เรามักทดสอบการเชื่อมต่อโดยใช้ **Blink Sketch**
- **Blink Sketch** เป็นโปรแกรมแรกของผู้เริ่มต้นใช้งานบอร์ด Arduino ผมคิดว่าในวงการนี้รู้จักเจ้า Blink กันทุกคน
- **Blink** เป็น Sketch ที่สำคัญในการตรวจสอบว่าบอร์ดที่ใช้ทำงานปกติ หรือ เข้ากับ IDE ที่คุณกำลังใช้อยู่หรือไม่



# Blink Sketch

2. จะปรากฏหน้าต่างตามรูป ให้กด upload



The screenshot shows the Arduino IDE 2.2.1 interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu bar is a toolbar with several icons. The 'Upload' icon, which is a right-pointing arrow, is circled in red. To the right of the toolbar is a dropdown menu labeled 'Select Board'. The main workspace contains the following code:

```
3
4 Turns an LED on for one second, then off for one second, repeatedly.
5
6 Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
7 it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
8 the correct LED pin independent of which board is used.
9 If you want to know what pin the on-board LED is connected to on your Arduino
10 model, check the Technical Specs of your board at:
11 https://www.arduino.cc/en/Main/Products
12
13 modified 8 May 2014
14 by Scott Fitzgerald
15 modified 2 Sep 2016
16 by Arturo Guadalupi
17 modified 8 Sep 2016
18 by Colby Newman
19
20 This example code is in the public domain.
21
22 https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27   // initialize digital pin LED_BUILTIN as an output.
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34   delay(1000); // wait for a second
35   digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
36   delay(1000); // wait for a second
```

## Blink Sketch

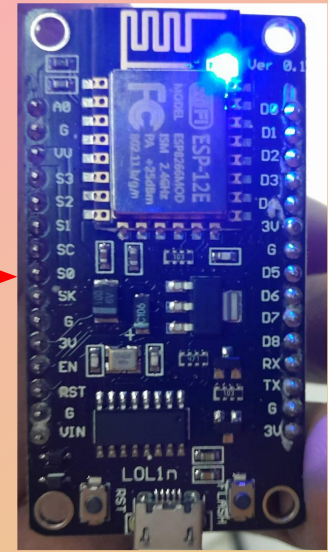
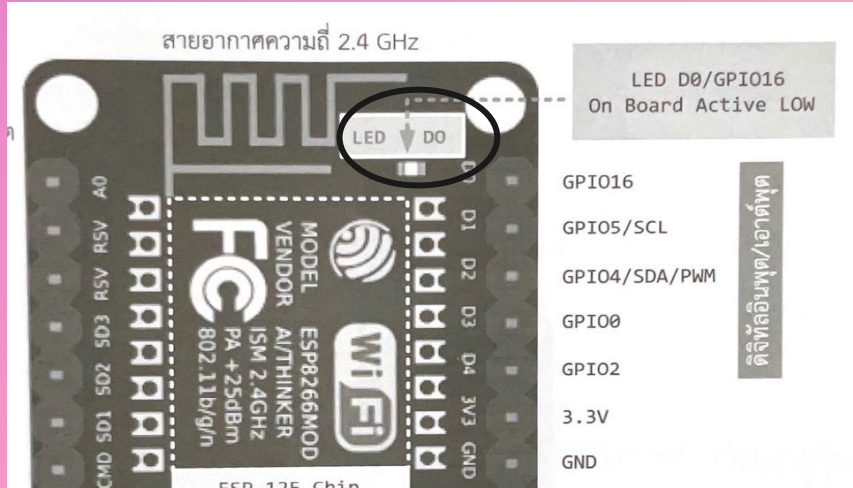
```
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27     // initialize digital pin LED_BUILTIN as an output.
28     pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
33     digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
34     delay(1000); // wait for a second
35     digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
36     delay(1000); // wait for a second
37 }
38
```



ตัวอักษรหลัง // คือ comment  
ไม่ใช่ code คำสั่ง ไม่มีผลกับ  
การทำงานของบอร์ด Arduino

# Blink Sketch

- เมื่อ upload เสร็จแล้ว LED D0/GPIO16 จะกะพริบ 1 sec และดับ 1 sec สลับกันไป



# โครงสร้างการควบคุม ของ **IDE**

# โครงสร้าง if

เครื่องหมายเปรียบเทียบจำนวนที่สามารถใช้กับ if() มีดังตาราง

code	สัญลักษณ์คณิตศาสตร์	ความหมาย
<code>x==y</code>	$x=y$	x มีค่าเท่ากับ y
<code>x!=y</code>	$x\neq y$	x มีค่าไม่เท่ากับ y
<code>x&lt;y</code>	$x<y$	x มีค่าน้อยกว่า y
<code>x&gt;y</code>	$x>y$	x มีค่ามากกว่า y
<code>x&lt;=y</code>	$x\leq y$	x มีค่าน้อยกว่าหรือเท่ากับ y
<code>x&gt;=y</code>	$x\geq y$	x มีค่ามากกว่าหรือเท่ากับ y

## โครงสร้าง if

ให้ดัดแปลงโปรแกรมในสไลด์ที่ 4 ให้ได้เงื่อนไขดังนี้

group	เงื่อนไข
1	ไฟกระพริบเมื่อ x มีค่าไม่เกิน 1 เวลาติด/ดับ 0.1 วินาที
2	ไฟกระพริบเมื่อ x มีค่าไม่เท่ากับ 2 เวลาติด/ดับ 0.2 วินาที
3	ไฟกระพริบเมื่อ x มีค่าติดลบ เวลาติด/ดับ 0.2 วินาที
4	ไฟกระพริบเมื่อ x มีค่าไม่เป็นบวก เวลาติด/ดับ 0.2 วินาที

# โครงสร้าง if และ else

โครงสร้าง if และ else เป็นโครงสร้างที่ให้ดำเนินการอย่างหนึ่งเมื่อเงื่อนไขสอดคล้อง และดำเนินการอีกอย่างเมื่อเงื่อนไขไม่สอดคล้อง

```
1  int x = 3;
2  void setup() {
3      pinMode(LED_BUILTIN, OUTPUT);
4  }
5  void loop() {
6      if (x>5)
7      {
8          digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
9          delay(500); // wait for a second
10         digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
11         delay(500); // wait for a second
12     }
13     else
14     {
15         digitalWrite(LED_BUILTIN, HIGH); // turn the LED on
16     }
17 }
```

## โครงสร้าง if และ else

ให้ตัดแปลงโปรแกรมในสไลด์ที่ 7 ให้ได้เงื่อนไขดังนี้

group	เงื่อนไข
1	ไฟกระพริบเมื่อ x มีค่าไม่เท่ากับ 3 ไม่เช่นนั้นให้ติดตลอด
2	ไฟกระพริบเมื่อ x มีค่าไม่เกิน 6 ไม่เช่นนั้นให้ติดตลอด
3	ไฟกระพริบเมื่อ x มีค่าไม่เป็นบวก ไม่เช่นนั้นให้ติดตลอด
4	ไฟกระพริบเมื่อ x มีค่าติดลบ ไม่เช่นนั้นให้ติดตลอด

# โครงสร้าง if และ else if

โครงสร้าง if และ else if เป็นโครงสร้างแบบ 2 เงื่อนไขขึ้นไป โดยพิจารณาตามลำดับ เช่น ถ้า if() สอดคล้อง จะไม่สนใจเงื่อนไข else if()

```
1  int x = 6;
2  void setup() {
3    pinMode(LED_BUILTIN, OUTPUT);
4  }
5  void loop() {
6    if (x>5)
7    {
8      digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
9      delay(500);                       // wait for a second
10     digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
11     delay(500);                       // wait for a second
12   }
13   else if (x>=3)
14   {
15     digitalWrite(LED_BUILTIN, HIGH); // turn the LED on
16   }
17   else
18   {
19     digitalWrite(LED_BUILTIN, LOW); // turn the LED off
20   }
21 }
22
```

# โครงสร้าง if และ else

ให้ตัดแปลงโปรแกรมในสไลด์ที่ 9 ให้ได้เงื่อนไขดังนี้

group	เงื่อนไข
1	ไฟกระพริบเมื่อ $x=3$ ไม่เช่นนั้นให้ไฟติดตลอดเมื่อ $x>0$ ไม่เช่นนั้นให้ไฟดับตลอด
2	ไฟกระพริบเมื่อ $x\leq 6$ ไม่เช่นนั้นให้ไฟติดตลอดถ้า $x\leq 10$ ไม่เช่นนั้นให้ไฟดับตลอด
3	ไฟกระพริบเมื่อ $x\leq 0$ ไม่เช่นนั้นให้ไฟติดตลอดถ้า $x=0$ ไม่เช่นนั้นให้ไฟดับตลอด
4	ไฟกระพริบเมื่อ $x<0$ ไม่เช่นนั้นให้ไฟติดตลอดถ้า $x>0$ ไม่เช่นนั้นให้ไฟดับตลอด

# โครงสร้าง for

โครงสร้าง for() จะเป็นคำสั่งให้ทำซ้ำคำสั่งใน { } หลัง for() トラบเท่าที่เงื่อนไขใน () เป็นจริง มีรูปแบบการใช้คำสั่งดังนี้  
for(ค่าเริ่มต้น, เงื่อนไข, การเพิ่มค่าต่อรอบ) เช่น

```
for(int i=0;i<3;i++)
```

```
{...}
```

- ❑ int i=0 หมายถึงกำหนดตัวแปร i เริ่มต้นเท่ากับ 0
- ❑ i<3 เงื่อนไขที่จะให้ดำเนินการคำสั่งใน {...}
- ❑ i++ คือเมื่อดำเนินการคำสั่งใน {...} 1 ครั้งแล้วให้ i เพิ่ม 1 ค่า

## โครงสร้าง for

โครงสร้าง for() อยู่ใน loop() จะเป็นลักษณะของการทำซ้ำ ซ้อนกับคำสั่งทำซ้ำอีกที (ลอง upload code ด้านล่างแล้วสังเกตุผล)

```
1 void setup() {
2   |   pinMode(LED_BUILTIN, OUTPUT);
3 }
4 void loop() {
5   for (int i=0;i<3;i++)
6     {
7     digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
8     delay(200); // wait for 0.2 sec
9     digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
10    delay(200); // wait for 0.2 sec
11  }
12  delay(2000); // wait for 2 sec
13 }
```

## โครงสร้าง for

ให้ตัดแปลงโปรแกรมในสไลด์ที่ 12 ให้ได้เงื่อนไขดังนี้

group	เงื่อนไข
1	ไฟกระพริบด้วยเวลาติด/ดับ 0.3 วินาที จำนวน 3 รอบ แล้วดับ 5 วินาที วนซ้ำไปเรื่อย ๆ
2	ไฟกระพริบด้วยเวลาติด/ดับ 0.1 วินาที จำนวน 6 รอบ แล้วดับ 3 วินาที วนซ้ำไปเรื่อย ๆ
3	ไฟกระพริบด้วยเวลาติด/ดับ 0.2 วินาที จำนวน 5 รอบ แล้วดับ 1 วินาที วนซ้ำไปเรื่อย ๆ
4	ไฟกระพริบด้วยเวลาติด/ดับ 0.4 วินาที จำนวน 4 รอบ แล้วดับ 4 วินาที วนซ้ำไปเรื่อย ๆ

# โครงสร้าง while

โครงสร้าง while() จะมีลักษณะการใช้งานคล้าย for() แต่ต่างกันที่กำหนดค่าเริ่มต้นก่อน while() และกำหนดการเพิ่มค่าที่บรรทัดสุดท้ายใน { }

```
1 void setup() {
2   | | pinMode(LED_BUILTIN, OUTPUT);
3 }
4 void loop() {
5   int i=0;
6   while (i<3)
7   {
8     digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
9     delay(200); // wait for 0.2 sec
10    digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
11    delay(200); // wait for 0.2 sec
12    i++;
13  }
14  delay(2000); // wait for 2 sec
15 }
16
```

## โครงสร้าง while

ให้ตัดแปลงโปรแกรมในสไลด์ที่ 14 ให้ได้เงื่อนไขดังนี้

group	เงื่อนไข
1	ไฟกระพริบด้วยเวลาติด/ดับ 0.1 วินาที จำนวน 6 รอบ แล้วดับ 3 วินาที วนซ้ำไปเรื่อย ๆ
2	ไฟกระพริบด้วยเวลาติด/ดับ 0.2 วินาที จำนวน 5 รอบ แล้วดับ 1 วินาที วนซ้ำไปเรื่อย ๆ
3	ไฟกระพริบด้วยเวลาติด/ดับ 0.4 วินาที จำนวน 4 รอบ แล้วดับ 4 วินาที วนซ้ำไปเรื่อย ๆ
4	ไฟกระพริบด้วยเวลาติด/ดับ 0.3 วินาที จำนวน 3 รอบ แล้วดับ 5 วินาที วนซ้ำไปเรื่อย ๆ

# โครงสร้าง break

โครงสร้าง break ใช้สำหรับออกจาก for หรือ while ก่อนเงื่อนไขที่กำหนดไว้ มักจะใช้คู่กับ if() เพื่อกำหนดเงื่อนไขว่าตอนไหนจะ break

```
1 void setup() {
2   |   pinMode(LED_BUILTIN, OUTPUT);
3   }
4 void loop() {
5   for(int i=0;i<10;i++)
6   {
7     digitalWrite(LED_BUILTIN, HIGH);
8     delay(200);
9     digitalWrite(LED_BUILTIN, LOW);
10    delay(200);
11  }
12  delay(2000);
13 }
14
```

```
1 void setup() {
2   |   pinMode(LED_BUILTIN, OUTPUT);
3   }
4 void loop() {
5   for(int i=0;i<10;i++)
6   {
7     if(i==5) break;
8     digitalWrite(LED_BUILTIN, HIGH);
9     delay(200);
10    digitalWrite(LED_BUILTIN, LOW);
11    delay(200);
12  }
13  delay(2000);
14 }
15
```

# โครงสร้าง switch

โครงสร้าง switch() จะมีลักษณะคล้าย if และ else if แต่เงื่อนไขแต่ละกรณีจะเป็นตัวเลข 1,2,3... เท่านั้น

```
switch(i)
{
  case 1: (เงื่อนไขนี้ทำงานเมื่อค่า i=1)
    คำสั่งต่าง ๆ สำหรับ case 1
    break;
  case 2: (เงื่อนไขนี้ทำงานเมื่อค่า i=2)
    คำสั่งต่าง ๆ สำหรับ case 2
    break;
  case 3: (เงื่อนไขนี้ทำงานเมื่อค่า i=3)
    คำสั่งต่าง ๆ สำหรับ case 3
    break;
}
```

หลังจากชุดคำสั่งของแต่ละ case จะต้องใส่ break; ด้วยเสมอ

## โครงสร้าง switch-case

```
1 void setup() {
2   | | pinMode(LED_BUILTIN, OUTPUT);
3 }
4 void loop() {
5   int i=3;
6   switch(i)
7   {
8     case 1:
9     digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
10    delay(200); // wait for 0.2 sec
11    digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
12    delay(200); // wait for 0.2 sec
13    break;
14    case 2:
15    digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
16    delay(1000); // wait for 1 sec
17    digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
18    delay(1000); // wait for 1 sec
19    break;
20    case 3:
21    digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
22    break;
23  }
24 }
```