บทที่ 2
โครงสร้างพื้นฐานของ Arduino IDE

โปรแกรม Arduino IDE ประกอบด้วยองค์ประกอบ 5 ส่วน คือโครงสร้างโปรแกรม (structure) ตัวแปร (variable) และค่าคงที่ (constant) ฟังก์ชันคำสั่ง (function) การใช้งานไลบรารี (Libraries) และโครงสร้างการควบคุม Arduino IDE

1. โครงสร้างโปรแกรม ประกอบด้วย

- **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
  เครื่องหมาย “{ }” เรียกว่าวงเล็บ (bracket) เป็นสัญลักษณ์ที่ใช้ขยายขอบเขตของคำสั่ง ความหมายคือถ้าไม่มี { } คำสั่งจะถูกดำเนินการเฉพาะบรรทัดแรกเท่านั้น ขณะที่ถ้ามี { } คำสั่งภายใน { } ทั้งหมดจะถูกดำเนินการ

2. ตัวแปร (variable) และค่าคงที่ (constant) ของ Arduino

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

Type	Byte	Range	Meaning
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	หมือนกับ char แต่เป็นบวกเท่านั้น

- **ตัวแปร 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 ตัวแปรประเภทนี้ใช้คล้ายกับเช็คว่าสวิตช์ถูกกดหรือไม่กด จะคล้ายกับการแสดงสถานะของ pin ขาออกของ Arduino ซึ่งจะแสดงค่าแรงดัน High หรือ Low แทน จริง หรือ เท็จ
- **ตัวแปรประเภท float และ double** float และ double ใช้สำหรับข้อมูลที่เป็นจำนวนทศนิยมและใช้เมื่อรับค่าประมาณของค่าจากการวัด ตัวแปรทั้งสองประเภทนี้มักจะใช้เพื่อคำนวณค่าที่ได้จากเซนเซอร์
- **ตัวแปรประเภท char และ string** char เก็บข้อมูลอักขระในช่องเก็บข้อมูล อาจเป็นตัวอักษร หรือ ตัวเลข string กำหนดตัวแปร string (อักษรหรือตัวเลขที่เรียงต่อกัน) จะกำหนดจากตัวแปรประเภท char มาเรียงต่อกัน

ค่าคงที่ (constant) คือ ค่าคงที่ใน Arduino คือค่าที่ถูกกำหนดไว้ล่วงหน้าและไม่สามารถเปลี่ยนแปลงได้ระหว่างการทำงาน เช่น การกำหนดพอร์ต I/O (INPUT หรือ OUTPUT) หรือการตั้งค่าคงที่ในการทำงานของบอร์ด (ดูในตารางในหัวข้อ 1.2)

3. ฟังก์ชันคำสั่งของ Arduino

ฟังก์ชั่นช่วยให้จัดโครงสร้างโปรแกรมที่เป็นชุดคำสั่ง (code) เพื่อทำงานแต่ละงานได้ กรณีทั่วไป เราจะสร้างฟังก์ชันเมื่อจำเป็นต้องดำเนินการเดียวกันหลายครั้งในโปรแกรม ฟังก์ชั่นจะประกอบด้วย ประเภท ชื่อฟังก์ชัน ตัวแปรขาเข้าของฟังก์ชัน และคำสั่งภายในฟังก์ชัน (statement) เช่น digitalRead(), digitalWrite(), analogRead(), analogWrite() ที่ใช้ในการอ่านและเขียนข้อมูลดิจิทัลหรืออนาล็อก ภายในคำสั่ง loop () (นั่นคือภายใน { } ที่ต่อจากคำสั่ง loop ()) ข้อความที่ไม่ใช่ตัวแปรจะเรียกว่าฟังก์ชัน ช่องว่างที่อยู่ใน () ที่ติดกับชื่อฟังก์ชัน เกิดจากการที่ฟังก์ชันเป็นประเภท void function (ฟังก์ชันที่ไม่ต้องมี input) เราไม่สามารถใช้ชื่อของฟังก์ชันต่อไปนี้ : setup, loop, if, for, switch เพราะคำเหล่านี้เป็นฟังก์ชันพื้นฐานที่ถูกกำหนดไว้แล้ว

4. โครงสร้างการควบคุม Arduino IDE

ใน Arduino IDE เป็นชุดคำสั่งที่ใช้ในการควบคุมการทำงานของโปรแกรมโดยการกำหนดลำดับขั้นตอนและเงื่อนไขการทำงาน ช่วยให้การทำงานของโปรแกรมมีความยืดหยุ่นและสามารถตอบสนองต่อสถานการณ์ต่างๆ ได้ โครงสร้างการควบคุมมีหลายประเภทที่สำคัญดังนี้

4.1. **โครงสร้าง if** คือเครื่องหมายเปรียบเทียบจำนวน เครื่องหมายที่สามารถใช้ใน if() ได้มีดังนี้

code	สัญลักษณ์คณิตศาตร์	ความหมาย
x==y	x=y	x มีค่าเท่ากับ y
x!=y	x≠y	x มีค่าไม่เท่ากับ y
x<y	x<y	x มีค่าน้อยกว่า y
x>y	x>y	x มีค่ามากกว่า y
x<=y	x≤y	x มีค่าน้อยกว่าหรือเท่ากับ y
x>=y	x≥y	x มีค่ามากกว่าหรือเท่ากับ y

4.2. **โครงสร้าง if else if และ else** เป็นโครงสร้างที่ให้ดำเนินการอย่างหนึ่งเมื่อเงื่อนไขสอดคล้อง และดำเนินการอีกอย่างเมื่อเงื่อนไขไม่สอดคล้อง

```
int x = 6;
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  if (x>5)
  {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
  delay(500);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(500);                       // wait for a second
 }
  else if (x>=3)
  {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on
  }
  else
  {
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off
  }
}
```

4.3. **โครงสร้าง for()** จะเป็นคำสั่งให้ทำซ้ำคำสั่งใน { } หลัง for() ตราบเท่าที่เงื่อนไขใน () 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() อยู่ใน loop() จะเป็นลักษณะของการทำซ้ำ ซ้อนกับคำสั่งทำซ้ำอีกที (ลอง upload code ด้านล่างแล้วสังเกตผล)

```
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);
8     delay(200);
9     digitalWrite(LED_BUILTIN, LOW);
10    delay(200);
11    }
12    delay(2000); // wait for 2 sec
13  }
```

4.4. **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);
9     delay(200);
10    digitalWrite(LED_BUILTIN, LOW);
11    delay(200);    // wait for 0.2 sec
12    i++;
13    }
14    delay(2000); // wait for 2 sec
15  }
```

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

```
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  for(int i=0;i<10;i++)
  {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(200);
  digitalWrite(LED_BUILTIN, LOW);
  delay(200);
  }
  delay(2000);
}
```

```
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  for(int i=0;i<10;i++)
  {
    if(i==5) break;
  digitalWrite(LED_BUILTIN, HIGH);
  delay(200);
  digitalWrite(LED_BUILTIN, LOW);
  delay(200);
  }
  delay(2000);
}
```

4.6. **โครงสร้าง switch() case** จะมีลักษณะคล้าย if และ else if แต่เงื่อนไขแต่ละกรณีจะเป็นตัวเลข 1,2,3… เท่านั้น ใช้เมื่อมีหลายเงื่อนไขที่ต้องการตรวจสอบ แต่ละเงื่อนไขจะถูกแยกออกมาเป็น "กรณี" (case) ใน switch()

```
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;
}
```

```
void setup() {
    pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  int i=3;
  switch(i)
  {
  case 1:
  digitalWrite(LED_BUILTIN, HIGH);
  delay(200);
  digitalWrite(LED_BUILTIN, LOW);
  delay(200);   // wait for 0.2 se
  break;
  case 2:
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);   // wait for 1 sec
  break;
  case 3:
  digitalWrite(LED_BUILTIN, HIGH);
  break;
  }
}
```

5. การใช้งานไลบรารี (Libraries)

การใช้งานไลบรารี (Libraries) ใน Arduino เป็นการเพิ่มความสามารถให้กับโปรแกรมโดยการนำชุดคำสั่งหรือฟังก์ชันที่พัฒนาไว้ล่วงหน้าเข้ามาใช้งาน การใช้งานไลบรารีช่วยให้การเขียนโค้ดสะดวกขึ้นและลดความซับซ้อน โดยเฉพาะอย่างยิ่งเมื่อโปรแกรมต้องทำงานร่วมกับอุปกรณ์ที่มีความซับซ้อน เช่น เซ็นเซอร์ มอเตอร์ หรือการเชื่อมต่อไร้สาย

5.1. ขั้นตอนการติดตั้งและเรียกใช้ไลบราลี

1) ไลบรารีสามารถติดตั้งได้ผ่าน Arduino IDE โดยไปที่เมนู Sketch > Include Library > Manage Libraries แล้วค้นหาไลบรารีที่ต้องการ
2) สามารถดาวน์โหลดไลบรารีจากเว็บไซต์แล้วนำไฟล์ไลบรารี (.zip) มาติดตั้งโดยใช้เมนู Sketch > Include Library > Add .ZIP Library
3) หลังจากติดตั้งไลบรารีแล้ว เราสามารถเรียกใช้ไลบรารีโดยใช้คำสั่ง `#include` เพื่อนำไลบรารีเข้ามาในโปรแกรม เช่น

5.2. ตัวอย่างการใช้งานไลบรารี

5.2.1. ไลบรารีสำหรับเซ็นเซอร์อุณหภูมิและความชื้น DHT

ไลบรารี DHT.h ถูกใช้ในการอ่านค่าจากเซ็นเซอร์ DHT11 หรือ DHT22 ซึ่งเป็นเซ็นเซอร์วัดอุณหภูมิและความชื้น

5.2.2. ไลบรารีสำหรับการเชื่อมต่อ Wi-Fi ด้วย ESP8266

ไลบรารี ESP8266WiFi.h ใช้สำหรับการเชื่อมต่อ Wi-Fi บนโมดูล ESP8266

5.2.3. ไลบราลีสำหรับสร้างระบบ IoT

ไลบรารี blynk.h ใช้สำหรับเชื่อมต่อ ไมโครคอนโทรลเลอร์เช่น Arduino, ESP8266, และ ESP32 เข้ากับระบบเซิฟเวอร์ ซึ่งช่วยให้ผู้ใช้สามารถควบคุมและตรวจสอบอุปกรณ์จากระยะไกลผ่านแอปพลิเคชันบนสมาร์ทโฟน

6. บทสรุป

โปรแกรม Arduino IDE ประกอบด้วย 5 ส่วนหลัก:

- **โครงสร้างโปรแกรม**: ประกอบด้วย void setup() สำหรับการเริ่มต้นและ void loop() สำหรับการทำงานซ้ำ
- **ตัวแปรและค่าคงที่**: ใช้ประเภทต่าง ๆ เช่น int, float, boolean และ char เพื่อเก็บข้อมูลและค่าคงที่ที่ไม่เปลี่ยนแปลง
- **ฟังก์ชันคำสั่ง**: ใช้ฟังก์ชันต่าง ๆ เช่น digitalRead(), analogWrite() เพื่อจัดการกับข้อมูลดิจิทัลและอนาล็อก
- **โครงสร้างการควบคุม**: ใช้คำสั่งเช่น if, for, while, break, และ switch เพื่อควบคุมลำดับการทำงานของโปรแกรม

- **การใช้งานไลบรารี**: เพิ่มความสามารถให้โปรแกรมโดยใช้ชุดคำสั่งที่พัฒนาไว้ล่วงหน้า เช่น ไลบรารี DHT.h สำหรับเซ็นเซอร์อุณหภูมิ, ESP8266WiFi.h สำหรับ Wi-Fi, และ Blynk.h สำหรับการสร้างระบบ IoT

การเข้าใจและใช้งานองค์ประกอบเหล่านี้อย่างมีประสิทธิภาพจะช่วยให้สามารถพัฒนาโปรแกรม Arduino ที่มีความยืดหยุ่นและตอบสนองความต้องการของงานต่าง ๆ ได้อย่างครอบคลุมและถูกต้อง