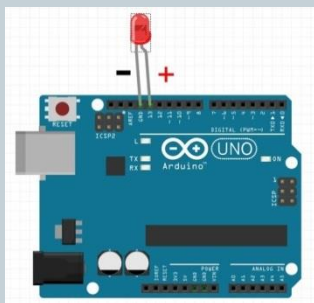


Arduino



智慧生活裝置

單元一入門課程--簡單智慧裝置

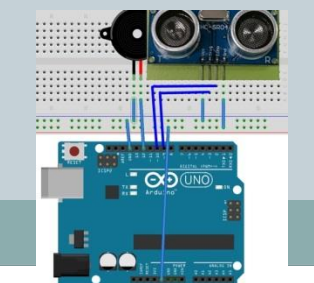
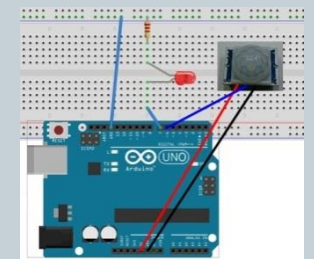


活動一：居家自動照明

- 練習一：LED燈閃爍(LED，練習程式基本結構，拆解問題)
- 練習二：外接麵包板
- 練習三：自動照明裝置(HC-SR501 人體紅外線模組,練習條件結構)

活動二：倒車雷達

- 練習一：蜂鳴器(蜂鳴器，練習重複結構)
- 練習二：播放不同音頻(陣列資料結構的程式設計實作)
- 練習三：倒車雷達加蜂鳴器(超音波，程式流程及結構綜合練習)



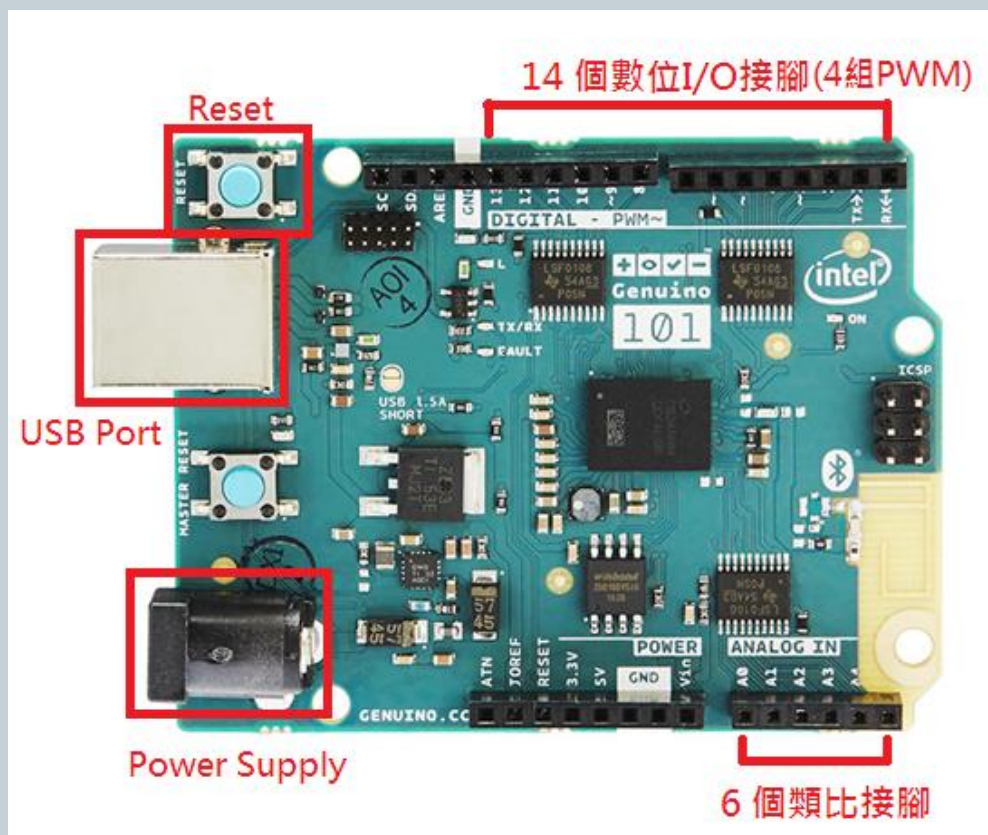
單元一入門課程--簡單智慧裝置



- 入門單元：藉由兩個簡單智慧型裝置居家自動照明及倒車雷達介紹Arduino開發環境、簡單電路開關設計及程式結構
- 單元簡介：介紹Arduino的基礎知識，教導學生建立開發環境。
- 具體目標則為利用簡單的程式控制外接電路板上的LED燈號、蜂鳴器、紅外線動作感測器、超音波感測器。並教導學生演算法基本概念—問題解析、流程控制
- 思考能力：了解、應用
- 運算思維能力：問題解析、資料表示（陣列）、樣式辨識、模擬及演算法設計

Arduino

4

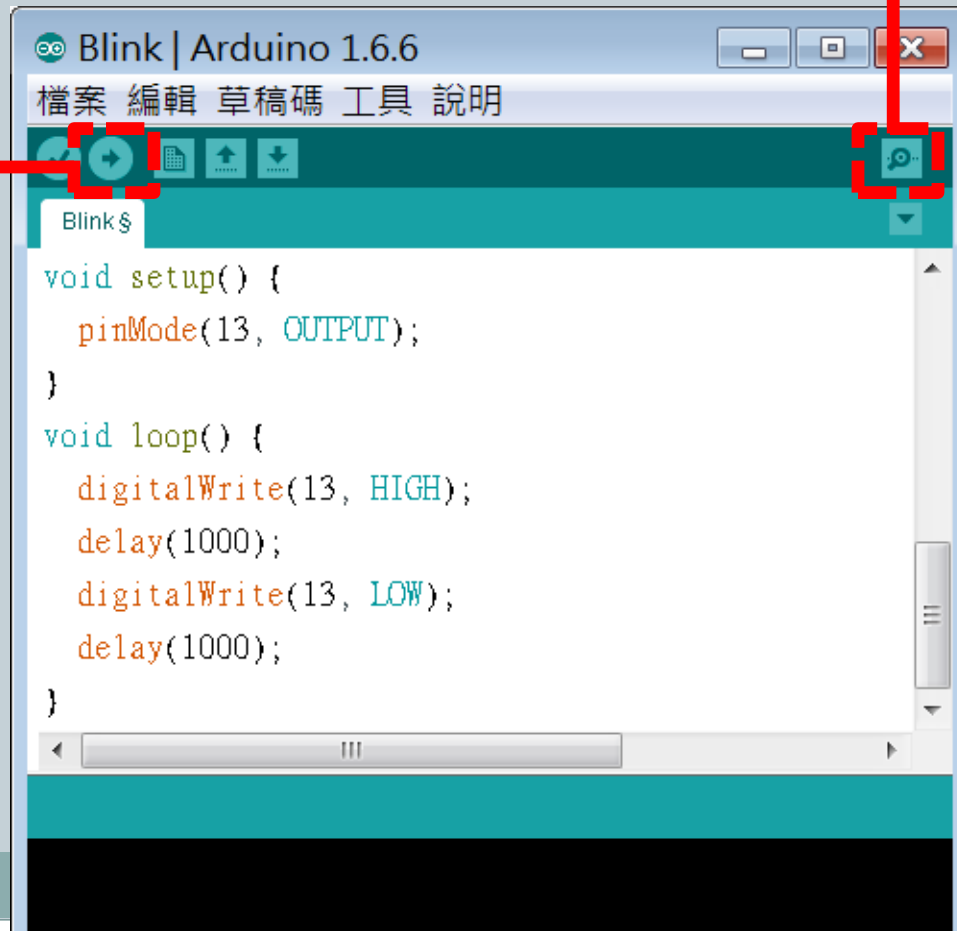


- 14 個數位輸入/輸出針腳
- 6 個類比輸入
- USB
- DC 電源插孔
(7V – 15V DC 輸入)

Arduino IDE

查看程式執行
後結果輸出

將程式上傳至
Arduino開發板



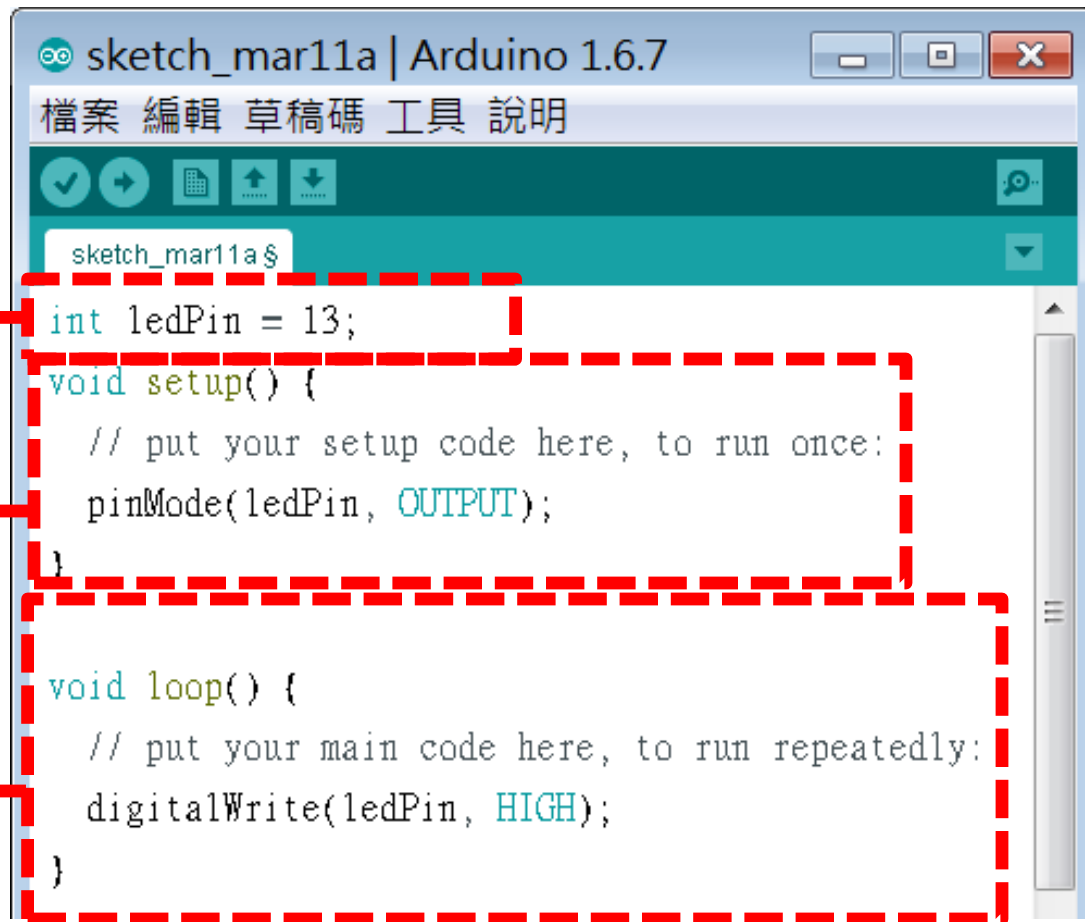
Arduino IDE

6

global變數定義區

setup()
啟動時被執行一次

loop()重複被執行



```
sketch_mar11a | Arduino 1.6.7
檔案 編輯 草稿碼 工具 說明
sketch_mar11a$
int ledPin = 13;
void setup() {
  // put your setup code here, to run once:
  pinMode(ledPin, OUTPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(ledPin, HIGH);
}
```

Arduino IDE



<pre>//定義變數紀錄LED連接於開發板的腳位 int ledPin = 13;</pre>	global 變數區
<pre>//設定腳位類型 pinMode(ledPin, OUTPUT);</pre>	setup 區
<pre>//設定腳位狀態 digitalWrite(ledPin, HIGH);</pre>	loop 區

Arduino程式的基本架構



- **void setup()**

```
{  
}
```

- 主要用來進行pin接腳類型的設定、變數的初始化等，這個函數區塊中的程式只有在程式開始執行時以及程式重新執行(reset)時會被呼叫執行一次。

Arduino程式的基本架構



- **void loop()**

```
{  
}
```

- 在執行完setup()函數中的所有設定及程式後，系統便會將控制權交給loop()函數，且自此之後便會重覆不斷的執行撰寫在loop()函數區塊中的程式，這也是Arduino程式的主要執行區塊，在此要特別強調，除非Arduino板的電力耗盡，否則，撰寫在loop()函數區塊中的程式碼將會反覆不斷的被執行，永不停止。

與電腦連接



- 點選桌面**Arduino**圖示



開啟**Arduino IDE**


- 使用**USB**傳輸線將開發板與電腦連接

- 點選 **工具** > **板子** > **Arduino/開發版型號**

- 點選 **工具** > **序列埠** > **COMxx(Arduino/開發版型號)**

測試



- 點選 **檔案** > **範例** > **01.Basics** > **Blink** > **Blink.ino**
- 點選上傳圖示  將程式燒錄至開發板
- 確認開發板上的**LED**閃爍



開啟與執行程式檔

- 點選 **檔案** > **開啟...**

從 Code 資料夾內選取 > **CH1_Blink** > **CH1_Blink.ino**

- 點選上傳圖示  將程式燒錄至開發板

- 確認開發板上的**LED**閃爍

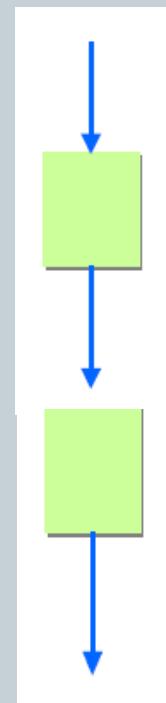


程式的基本控制結構



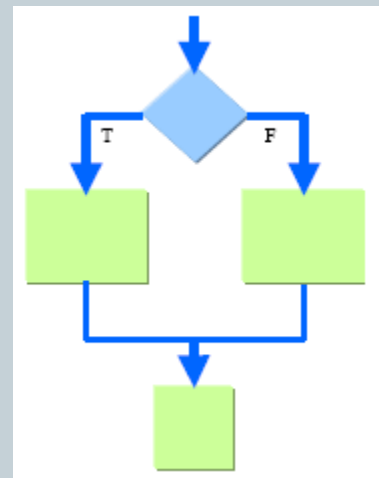
● 循序結構(Sequence)

- 程式由上而下，第一行指令敘述執行完後接著執行第二行，如此依序一行一行的執行。



● 選擇結構(Selection)

- 讓程式能依據條件判斷的結果分支執行，程式流程進入判斷的菱形符號後，會判斷測試條件是否成立。然後，依據判斷的結果選擇程式的流向。



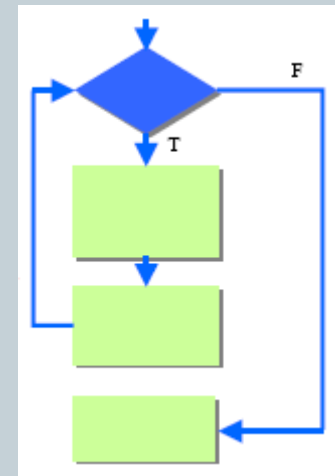
程式的基本控制結構



● 重複結構(Iteration)

- 在程式執行時，常有重複執行某部分程式片段的需要，搭配條件判斷的機制，部分程式片段可重複執行多次，直到某測試條件發生為止，程式重複執行部分即構成迴圈。

○



單元一入門課程--簡單智慧裝置



活動一：居家自動照明

元件介紹



LED

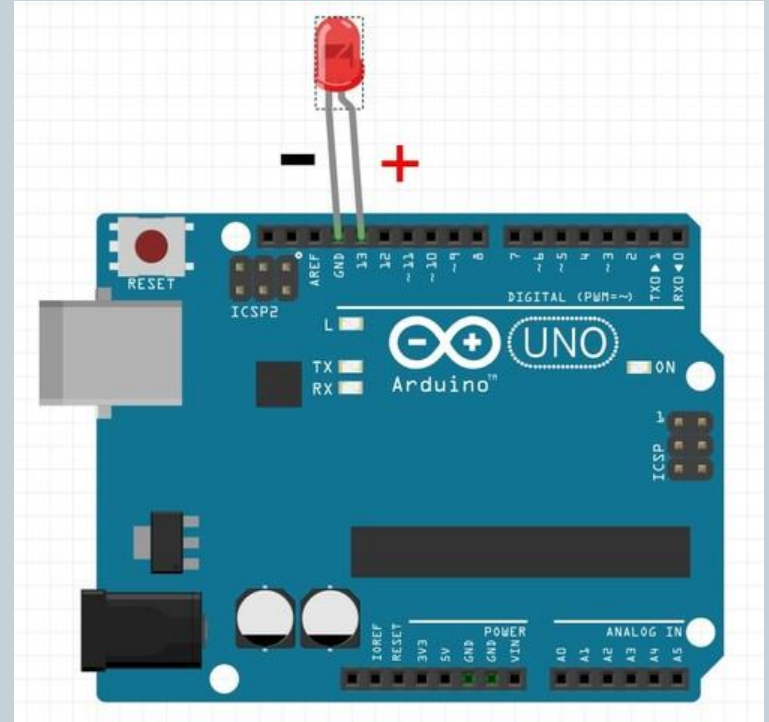
➤ OUTPUT 元件

- 長腳為正，短腳為負
- 需外接電阻避免燒毀



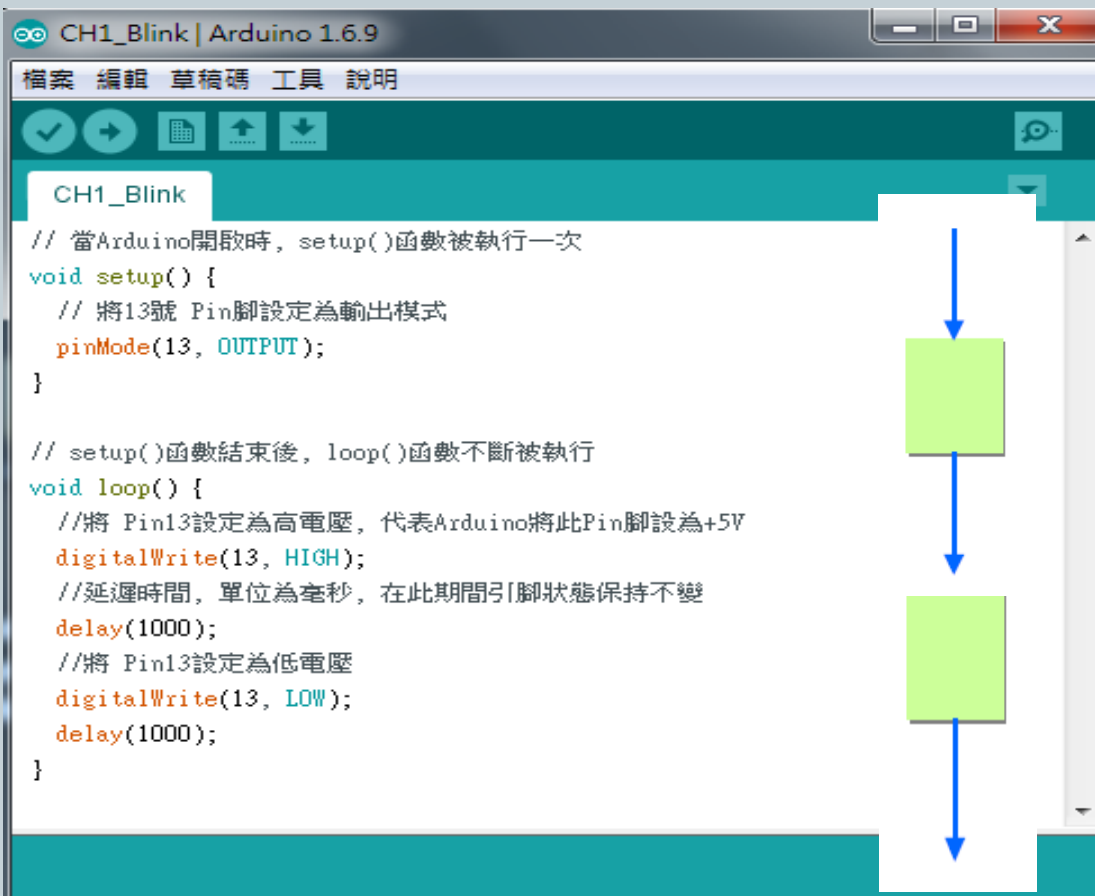
活動一之練習一：LED燈閃爍

- 讓一顆燈號閃爍，每隔一秒切換一次燈號。
- 把 LED 接到 Arduino 板子上，LED 長腳(陽極)接到 pin13，短腳(陰極)接到 GND



活動一之練習一：程式

先給學生 **pattern**，讓其改寫



```
CH1_Blink | Arduino 1.6.9
檔案 編輯 草稿碼 工具 說明

CH1_Blink
// 當Arduino開啟時, setup()函數被執行一次
void setup() {
  // 將13號 Pin腳設定為輸出模式
  pinMode(13, OUTPUT);
}

// setup()函數結束後, loop()函數不斷被執行
void loop() {
  //將 Pin13設定為高電壓, 代表Arduino將此Pin腳設定為+5V
  digitalWrite(13, HIGH);
  //延遲時間, 單位為毫秒, 在此期間引腳狀態保持不變
  delay(1000);
  //將 Pin13設定為低電壓
  digitalWrite(13, LOW);
  delay(1000);
}
```

pinMode(pin, mode)接腳
模式設定

- **pin**: 指定要進行模式設定的接腳
- **mode**: 可設定為INPUT or OUTPUT

➤ **pinMode(13, OUTPUT)**便是將第13接腳設定為OUTPUT模式，亦即可藉由pin 13 輸出訊號

活動一之練習一：程式

先給學生 **pattern**，讓其改寫



```
CH1_Blink | Arduino 1.6.9
檔案 編輯 草稿碼 工具 說明

CH1_Blink
// 當Arduino開啟時，setup()函數被執行一次
void setup() {
  // 將13號 Pin腳設定為輸出模式
  pinMode(13, OUTPUT);
}

// setup()函數結束後，loop()函數不斷被執行
void loop() {
  //將 Pin13設定為高電壓，代表Arduino將此Pin腳設定為+5V
  digitalWrite(13, HIGH);
  //延遲時間，單位為毫秒，在此期間引腳狀態保持不變
  delay(1000);
  //將 Pin13設定為低電壓
  digitalWrite(13, LOW);
  delay(1000);
}
```

- **digitalWrite(pin, value)**
輸出數位訊號
 - **pin**: 指定要輸出數位訊號的接腳編號
 - **value**: 設定要輸出的訊號為LOW或HIGH
- **digitalWrite(13, HIGH)**
便是由第13接腳輸出HIGH訊號

活動一之練習一：程式

先給學生pattern，讓其改寫



```
CH1_Blink | Arduino 1.6.9
檔案 編輯 草稿碼 工具 說明

CH1_Blink
// 當Arduino開啟時，setup()函數被執行一次
void setup() {
  // 將13號 Pin腳設定為輸出模式
  pinMode(13, OUTPUT);
}

// setup()函數結束後，loop()函數不斷被執行
void loop() {
  //將 Pin13設定為高電壓，代表Arduino將此Pin腳設為+5V
  digitalWrite(13, HIGH);
  //延遲時間，單位為毫秒，在此期間引腳狀態保持不變
  delay(1000);
  //將 Pin13設定為低電壓
  digitalWrite(13, LOW);
  delay(1000);
}
```

- Arduino的運作非常快速，在送完一個HIGH訊號後，立即再送一個LOW訊號，LED燈也就隨即滅掉，為了要讓LED燈亮滅的時間都能維持久一點，可分別在輸出HIGH和LOW訊號的程式後各加入一個延遲的函數指令 `delay()`

- `delay(ms)` 讓程式暫停一段時間

其中

ms: 指定要程式暫停多少毫秒
因此可加入 `delay(1000);` 來讓亮滅各維持一秒鐘。

運算思維導向教學



問題解析 Decomposition

- 能將大問題分解成足以解決的小問題



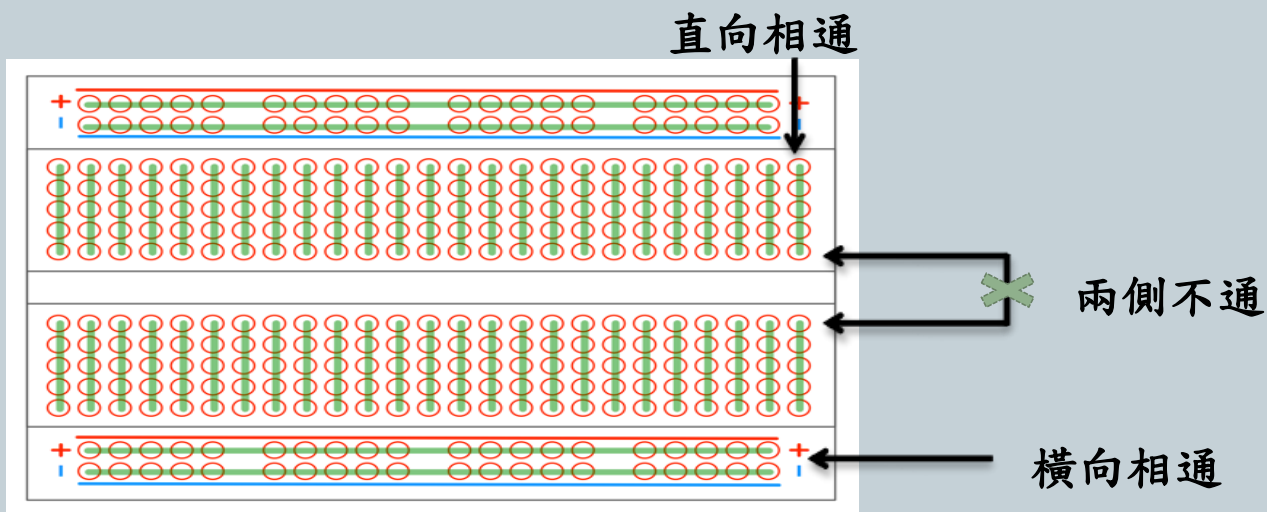
- 可分解成
 - LED 亮
 - delay
 - LED 暗

元件介紹



麵包板

當欲連接至開發板上相同腳位，但腳位不夠使用時，可接於麵包板相通位置上後，再連接至開發板。

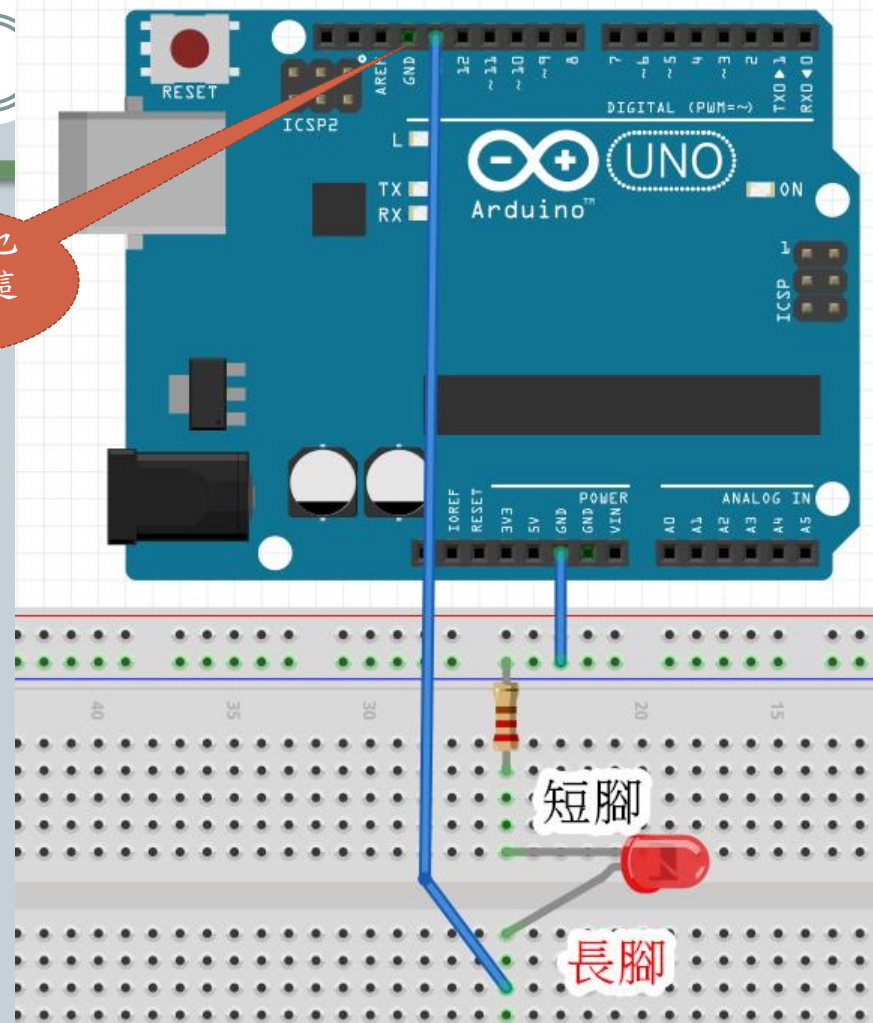


活動一之練習二：外接麵包板

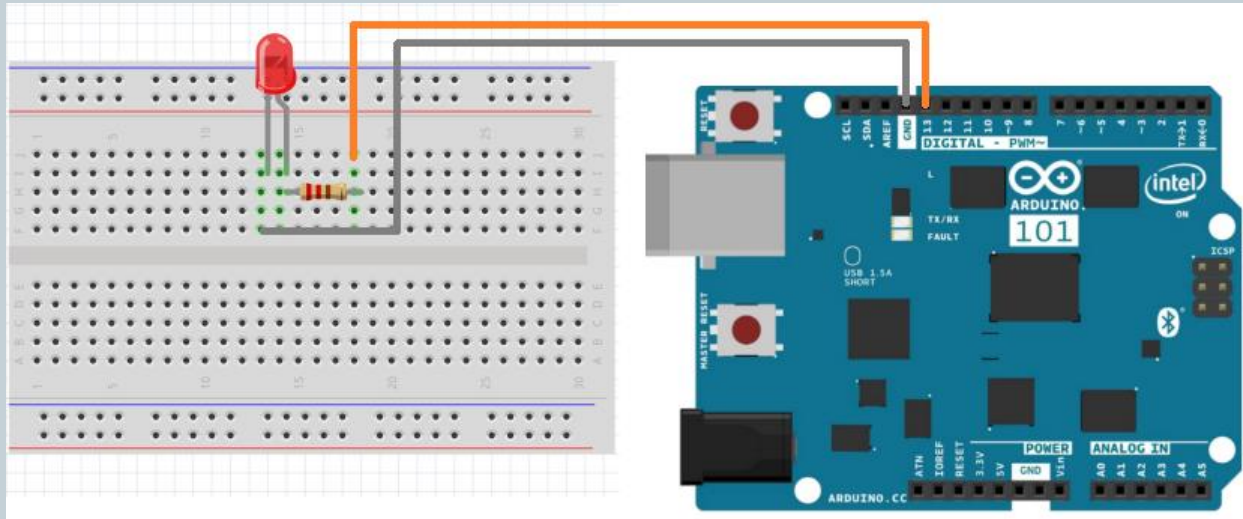
電路：

- 將LED正極(長腳)串聯220歐姆電阻，接至Pin13
- LED負極(短腳)接至GND

GND也可以接這裡



麵包板也可以這樣接



延伸練習



1. 改變LED燈頻率
2. 請加入另一個LED燈(或兩個)做出特殊的創意表現

- 設置紅綠燈系統

(綠燈1秒>黃燈1秒>紅燈1秒)

[綠燈]亮一秒，然後熄滅

[黃燈]亮一秒，然後熄滅

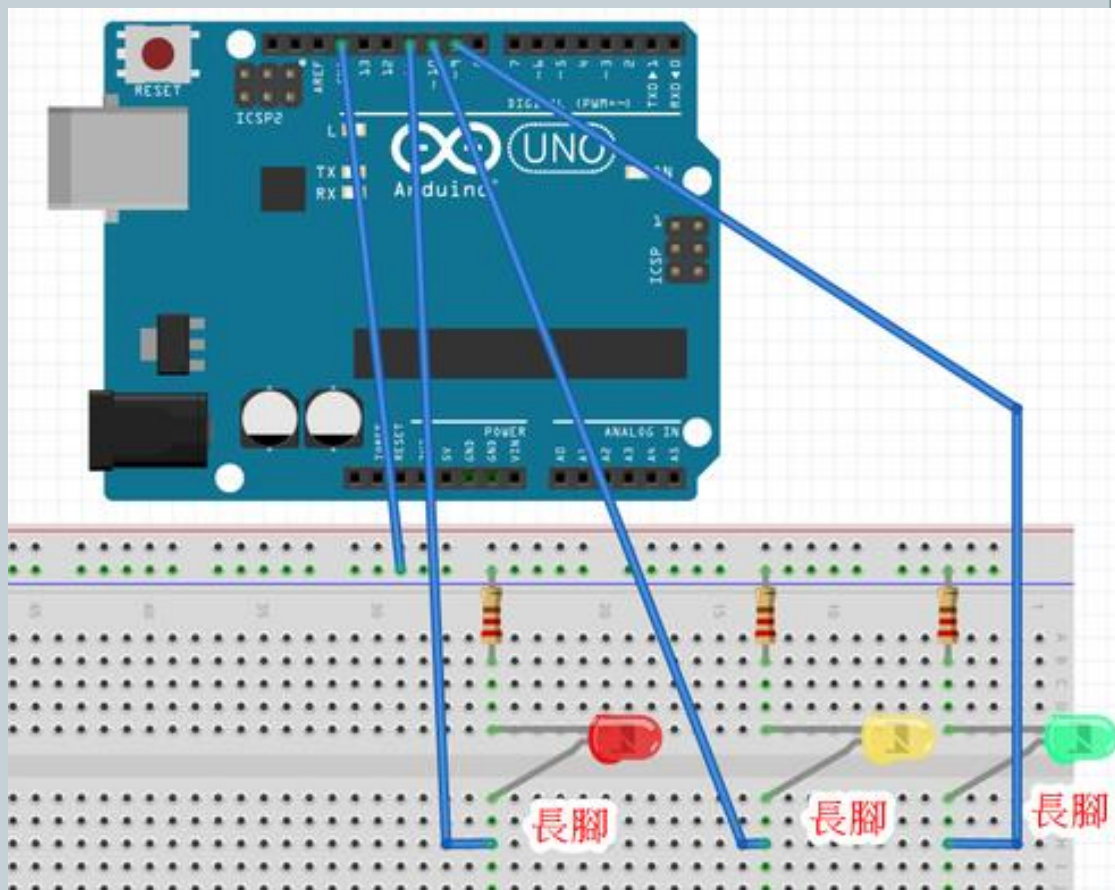
[紅燈]亮一秒，然後熄滅

- 設置紅綠燈系統

(綠燈5秒>黃燈2秒>紅燈3秒)

- 綠、黃、紅LED輪流亮，黃燈要閃爍

(綠燈亮三秒，黃燈閃爍，每閃一次是0.1秒，紅燈亮三秒)



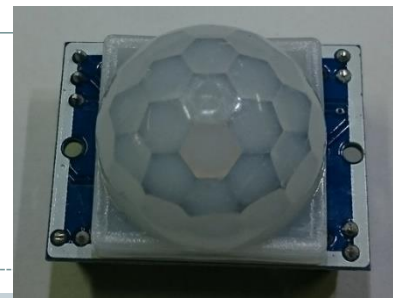
活動一之練習三：自動照明裝置



□ 感應亮燈

- 當有人靠近時，就會自動亮燈，人離開後燈就自動熄滅

元件介紹

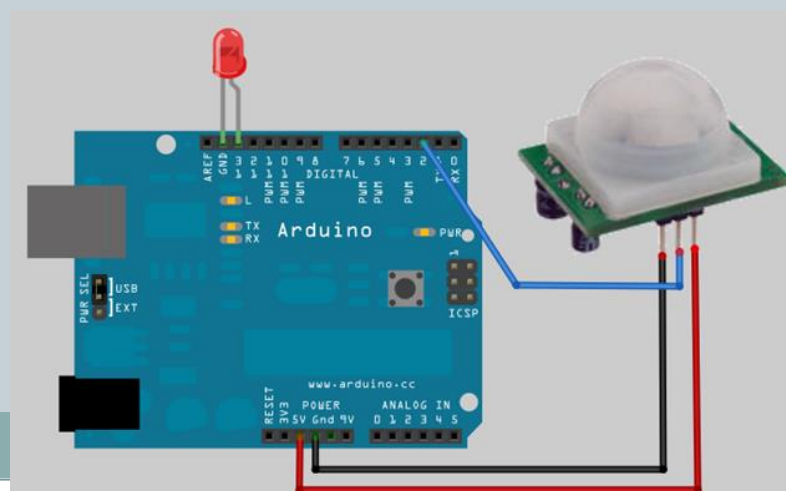
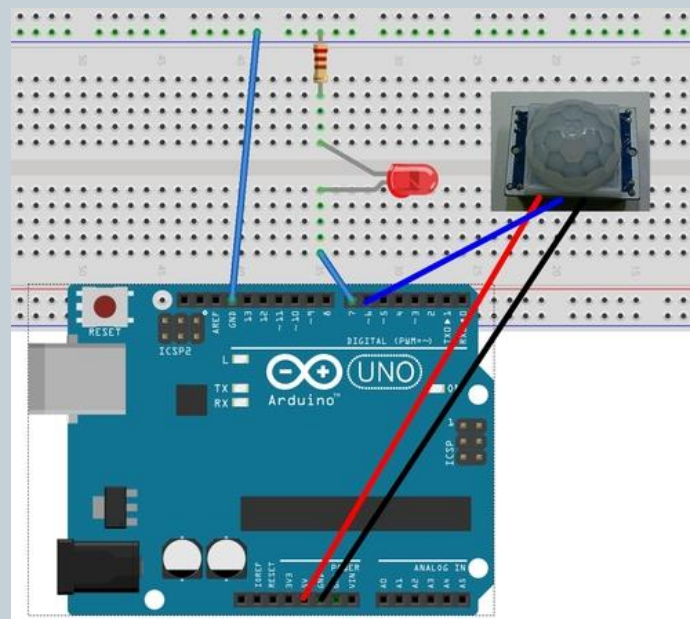


HC-SR501 人體紅外線模組

- 紅外線動作感測器 (Passive Infrared Sensor) 又稱為(PIR Motion Sensor)
 - 屬於被動式的紅外線裝置
 - 感應器本身不會發射紅外線光束
 - 原理就是利用物體發射出來的紅外線的變化，來感應物體的移動
 - 一般利用人體會發出紅外線的特性，常用來當作感應人體的感測器。
- 用於防盜系統
 - 如有人入侵屋內便響警報的紅外警報器
- 用於自動照明裝置
 - 如玄關、走廊、樓梯間或車庫門口等不常有人走動的的地方，將紅外線感應器和燈具裝在這些地方，只要有人靠近就會自動開燈照明，人離開後就自動關燈省電。

活動一之練習三：自動照明裝置電路圖

- HC-SR501 人體紅外線感測器有3個接腳，VCC接5V，GND接地，中間OUT接數位腳D6當成Arduino的輸入訊號。
- 另外接一組紅色LED當作感應燈，正極接數位腳D7



活動一之練習三：mBlock



- 當感測器感應到有人經過時，數位腳位D2會出現高電位(數值1)，這時就讓數位腳位D13變成高電位，紅色LED亮起來。
- 沒有人時，感測器沒有感應，數位腳位D13會維持低電位，紅色LED就不會亮。

變數



- 所謂變數即是一塊預先保留的記憶體空間，可在程式執行過程中，將資料或運算的結果儲存起來，供程式後續處理之用
- 變數在使用前必須先進行**宣告**，包括設定變數的名稱、設定變數的資料型態，以及設定變數的初始值等
- 變數宣告語法：
 - 資料型態 變數名稱;
 - ✦ 如：int score ;
 - 資料型態 變數名稱 = 初始值;
 - ✦ 如：int score = 70 ;

程 式 -- 變 數



```
01  /*
02  PIR("Passive Infrared Sensor") Motion Sensor,
03  紅外線動作感測器，或稱人體紅外線感測器
04  */
05
06  const int PIRSensor = 2;    // 紅外線動作感測器連接的腳位
07  const int ledPin = 13;     // LED 腳位
08
09  int sensorValue = 0;       // 紅外線動作感測器訊號變數
10
11  void setup() {
12    pinMode(PIRSensor, INPUT);
13    pinMode(ledPin, OUTPUT);
14  }
15
16  void loop(){
17    // 讀取 PIR Sensor 的狀態
18    sensorValue = digitalRead(PIRSensor);
19
20    // 判斷 PIR Sensor 的狀態
21    if (sensorValue == HIGH) {
22      digitalWrite(ledPin, HIGH); // 有人，開燈
23    }
24    else {
25      digitalWrite(ledPin, LOW); // 沒人，關燈
26    }
27  }
```

Arduino的選擇結構

- if (條件判斷式)

```
{
```

```
  // action A
```

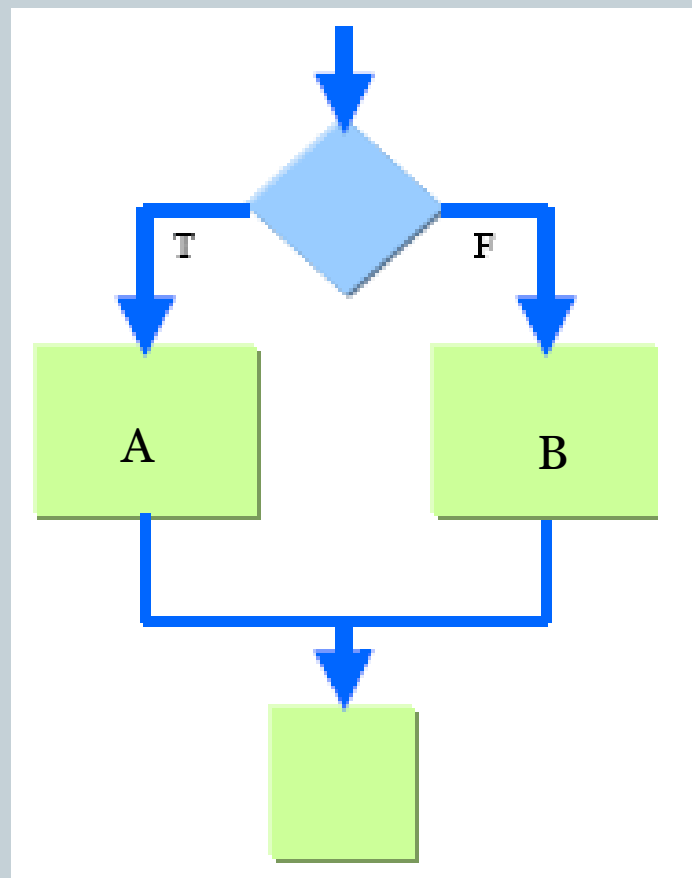
```
}
```

```
else
```

```
{
```

```
  // action B
```

```
}
```

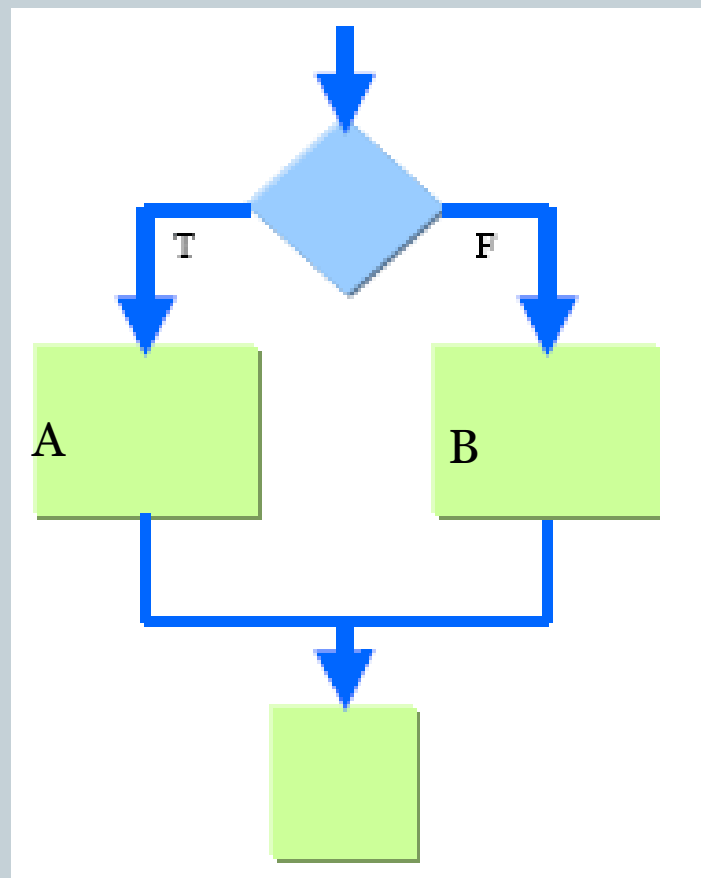


Arduino的選擇結構



● 關係運算子

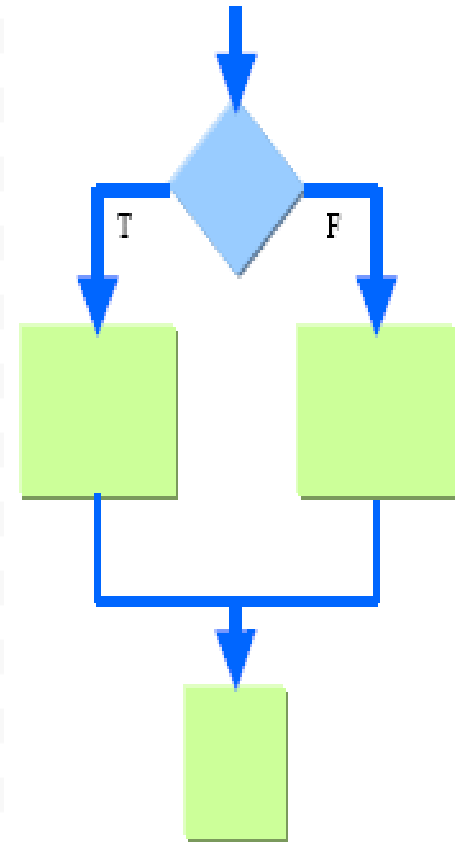
- == 左邊的值等於右邊的值
- != 左邊的值不等於右邊的值
- > 左邊的值大於右邊的值
- >= 左邊的值大於或等於右邊的值
- < 左邊的值小於右邊的值
- <= 左邊的值小於或等於右邊的值



程式



```
01 /*  
02  PIR("Passive Infrared Sensor") Motion Sensor,  
03  紅外線動作感測器，或稱人體紅外線感測器  
04  */  
05  
06  const int PIRSensor = 2;    // 紅外線動作感測器連接的腳位  
07  const int ledPin = 13;     // LED 腳位  
08  
09  int sensorValue = 0;       // 紅外線動作感測器訊號變數  
10  
11  void setup() {  
12    pinMode(PIRSensor, INPUT);  
13    pinMode(ledPin, OUTPUT);  
14  }  
15  
16  void loop(){  
17    // 讀取 PIR Sensor 的狀態  
18    sensorValue = digitalRead(PIRSensor);  
19  
20    // 判斷 PIR Sensor 的狀態  
21    if (sensorValue == HIGH) {  
22      digitalWrite(ledPin, HIGH); // 有人，開燈  
23    }  
24    else {  
25      digitalWrite(ledPin, LOW); // 沒人，關燈  
26    }  
27  }
```



延伸活動



- 自己改改看，修改程式為自己最愛的作品