

1 LED燈模組應用

2 蜂鳴器模組應用

3 數位開關輸入控制

4 直流馬達控制

基礎程式控制

安裝完成圖控軟體 Cagebot_blockly後，快來試試看拖曳圖塊寫程式吧!

本章開始，將循序介紹使用Cagebot_blockly程式，來操作Cageboard控制板的各項配備元件功能，同時練習Arduino程式設計的語法與技巧。Cagebot_blockly的程式功能非常多樣，可支援Arduino的各種感測器以及應用。我們在介紹範例應用的同時，也循序漸進介紹程式積木的意義與用法。因此，在學習每個範例時，將列出新學習使用的程式積木以及它的功能說明。

Hi! 我是小幫手 柯吉寶
學習的過程中我會幫忙
大家解決問題喔!



拖曳圖塊好簡單，沒寫過程式，也能從零開始!
Arduino 語法對照，進階學習不卡關!



1 LED燈模組應用(閃爍、紅綠燈)

小試身手:七彩霓虹燈

2 蜂鳴器模組應用(Do Re Mi、救護車、小星星)

小試身手:瑪莉有隻小綿羊

3 數位開關輸入控制(開關控制LED明滅、按鈕與變數控制LED、 按鈕變數控制七彩霓虹燈)

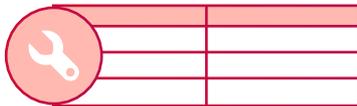
小試身手:按鈕聲光控制

4 直流馬達控制(雙馬達車測試、馬達變速測試)

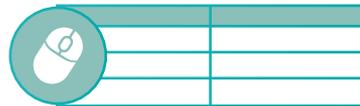
小試身手:唱歌跳舞動感探險車



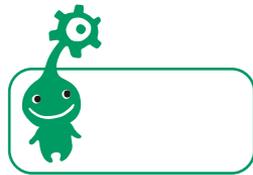
在本章節中，會有不同顏色的框格，各有不同的功能協助大家更容易學習。



粉紅色圖框 ▶ 積木樣式與功能說明



粉綠色圖框 ▶ 開始進行程式撰寫



綠色圖框 ▶ 小試身手試題

【LED燈模組應用】

1 輸出元件- LED燈模組應用

LED是最常見的電子零件，又稱為發光二極體，屬於Arduino的光「輸出裝置」。通常為單色，例如在一些控制板的電源附近的LED燈就是作為電源指示燈，顯示控制板的電源狀態；在連接USB時亮綠色燈告訴我們USB是接通的。離線時，關閉電源開關，在開啟時，亮綠色燈表示電源接通並且重置(Reset)程式。

另一種利用3色可混色顯示的全彩RGB LED，一般需要使用3支腳位來控制。例如使用串列式全彩LED控制晶片WS2812，僅使用1個數位的腳位，即可控制控制板多顆全彩RGB LED，做出多采多姿的顏色顯示，讓我們來試試吧！



1-1 程式範例：LED閃爍

此小節會使用到的積木樣式及功能說明，後續如有重覆積木則不重新說明



積木圖示	功能說明
	<ul style="list-style-type: none"> ● 「設定／迴圈」，對應 Arduino IDE專案中最基本的 <code>setup()</code> 與 <code>loop()</code> 兩個函式。 ● <code>setup()</code> 函式只會在一開始時被執行一次，<code>loop()</code> 函式反覆的執行 <code>loop()</code> 裡的程式。
	<ul style="list-style-type: none"> ● 可於左側積木控制區「顯示器」-「RGB LED」中找到。 ● WS2812控制D13腳位的燈數，第<#>燈、設定<#>顏色。
	<ul style="list-style-type: none"> ● 可於左側積木控制區「時間」找到。 ● 延遲時間，以毫秒(1/1000秒)為單位，延遲1000毫秒代表延遲1秒鐘。「時間」群組中有兩個時間延遲的程式積木，單位不同(毫秒、微秒)。兩者讓程式在延遲時間內只能停頓等待，而無法做其他的動作。



程式積木

LED閃爍

步驟說明



Step 1. 從「積木範例」開啟LED閃爍

Step 2. 在迴圈中命令D13的LED第1.2設定白色

Step 3. 延遲1000毫秒(1秒)

Step 4. 在迴圈中命令D13的LED第1.2設定黑色(暗)

Step 5. 延遲1000毫秒(1秒)

反覆執行

請依上傳燒錄程序，將程式上傳到有安裝LED的控制板。
您可以觀察到控制板上方的LED燈以1秒的間隔，不斷地閃爍。

再試試看選擇自己喜歡的
顏色燒錄吧!



1-2 程式範例：紅綠燈



一起來試試看！

讓探險車的LED燈，模擬紅綠燈的閃示，綠燈亮五秒後，閃黃燈間隔0.3秒共閃3次，紅燈亮5秒。



03.紅燈亮5秒

02.黃燈閃爍3次
(每次閃爍間隔0.3秒)

01.綠燈亮5秒

此小節會使用到的積木樣式及功能說明



積木圖示



功能說明

- 可於左側積木控制區「迴圈」中找到。
- 可設定迴圈重複執行次數的積木。
會重複跑【執行】裡面的程式積木動作，直到執行次數符合《重複》設定的重複次數為止。

接續程式範例1-1，我們進一步設計程式，讓LED燈如紅綠燈閃爍吧！

程式積木 紅綠燈	步驟說明
	<p>如左圖，將LED設定黑色的積木拖曳到設定中，這代表一開始LED是滅的。</p>
	<p>如左圖，再拉一組WS2812LED圖塊，製作綠燈亮5秒積木</p> <p>Step 1. 從設定顏色的框中，點選設定為綠色</p> <p>Step 2. 延遲秒數設為5000毫秒</p>



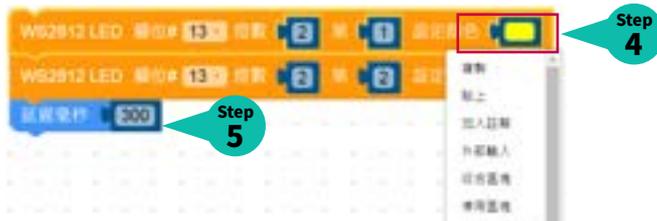
程式積木

紅綠燈

步驟說明



Step 3. 從左側「迴圈」群組，拖曳可設定迴圈重複執行次數的積木，重複執行次數設為 3。



Step 4. 複製一組做好的綠燈積木，製作亮黃燈積木 (滑鼠在LED積木上按右鍵，可以複製積木)
Step 5. 繼續修改成亮黃燈、間隔300毫秒(0.3秒)，並拖曳到重覆迴圈之中。
Step 6. 再複製一組積木，繼續修改成黑燈(熄滅)、間隔300毫秒(0.3秒)。



重複迴圈可以精簡指令，將重複執行一樣動作的積木，精簡成為重複次數。





程式積木

紅綠燈

步驟說明



Step 7. 複製綠燈亮5秒積木，修改為亮紅燈。



Step 8. 拖曳亮紅燈5秒積木到重覆迴圈下方，完成
Step 9. 按  下載儲存XML檔: 紅綠燈.XML。



因為是線上版本，如同下載檔案一般，所以
儲存位置在C:\Users\#\downloads 下載資
料夾中。



請依上傳燒錄程序，將程式上傳到控制板。
您可以觀察到控制板安裝的的LED燈以綠燈亮五秒後，閃黃燈間隔0.3秒共閃3次，紅燈亮5秒；然後重複執行。

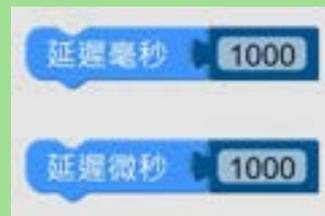


1-3 小試身手：七彩霓虹燈

請您設計程式，讓LED燈以每0.2秒間隔，變化紅、橙、黃、綠、藍、靛、紫七彩的顏色變化。



1秒(s)
= 1000 毫秒(ms)
= 1,000,000 微秒(μ s)
= 1,000,000,000 納秒(ns)
= 1,000,000,000,000 皮秒



圖控軟體積木使用毫秒與微秒作為計算時間的單位

【蜂鳴器模組應用】

2-1 輸出元件- 蜂鳴器模組應用介紹

蜂鳴器(Buzzer)是在電子產品中常用的發聲元件，可由Arduino控制發出任何頻率的聲音元件的「輸出裝置」。

蜂鳴器發聲的原理，是利用調頻PWM產生音頻來驅動蜂鳴器，讓空氣產生振動，便能發出聲音。只要適當地改變振動頻率，就可以產生不同音階的聲音。

經常使用的是無源蜂鳴器，通常用於數位腳位。所謂有源和無源中的「源」所指的並不是電源，而是震盪源。也就是說，有源蜂鳴器內部帶震盪電路，所以只要一通電就會叫。而無源蜂鳴器內部不帶震盪電路，所以如果單用直流信號無法讓它發聲。必須用20Hz-20kHz的方波去驅動蜂鳴器，才能發出聲音。



在這小節我們將學習蜂鳴器的發聲控制，蜂鳴器的發生頻率與延遲週期的控制，進而自行設計演奏不同的音樂。



2-2 程式範例：Do Re Mi

此小節會使用到的積木樣式及功能說明

積木圖示	功能說明
	<ul style="list-style-type: none"> • 可於左側積木控制區「蜂鳴器」中找到。 • 蜂鳴器發聲積木，指定蜂鳴器【腳位】，【聲音頻率】，【延遲週期】。



補充說明：

在蜂鳴器裡面會找到兩種模式，如右圖說明兩種都可以使用，調整頻率數值的靈活性會比選擇音階的多樣，可依照個人習慣使用喔~後續範例會使用填寫數值的積木做教學。



- 下拉選單選擇音階

- 自行填寫頻率數值，可搭配 04-21 頁的音階與頻率表使用

蜂鳴器Do Re Mi音階範例程式及說明積木

程式積木

Do Re Mi

步驟說明



- Step 1.** 請從左側積木控制區「蜂鳴器」群組中，拖曳與組合如左列的程式積木之一。左邊的程式兩者執行效果是一樣的，但是以標示唱名Do Re Mi的蜂鳴器積木只有一個音階。因此建議必要時參考音階表，直接填入發聲頻率數值，可以獲得更多的音樂變化。
- Step 2.** 您會發現程式中標示了延遲週期450，我們又設置延遲500毫秒，實際上是演奏450毫秒，間隔500毫秒，讓樂音有所區隔較為清晰。
- Step 3.** 左側兩個程式只能擇一使用，一個Arduino程式只能有一組的「設定/迴圈」。
- Step 4.** 按  下載儲存XML檔: 音樂-DoReMi.XML

請依上傳燒錄程序，將程式上傳到有安裝蜂鳴器控制板。
您可以觀察到蜂鳴器發出Do Re Mi，休止一拍，繼續重複執行。





由於Arduino程式會重複執行，吵個不停，這時候您只要上傳一個空迴圈，就可以停止聲音了！



空迴圈(甚麼也不做的程式)



程式範例：救護車



一起來試試看！

模擬救護車發出5次第五8度音Do與Fa各600毫秒，鳴笛的同時，前方的LED同步做紅色燈閃爍警示。

程式積木

救護車

步驟說明



Step 1. 請從左側積木「蜂鳴器」群組，拖曳與設計如左的程式積木，查表得到Do5頻率為523，Fa5頻率為698，修改參數。

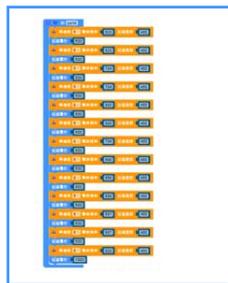
Step 2. LED指令與音頻放置一起，不用再另外加延遲delay()。

Step 3. 主程式拖曳放在「設定」之中，程式在設定區執行重複5次後，主迴圈loop()是空的，因此鳴笛與閃燈就停止了。

Step 4. 按  下載儲存XML檔：音樂-救護車.XML



以演奏音樂為例，你會發現連續幾個音符，程式就越來越長了。因此接著我們來介紹使用副程式積木，由主迴圈呼叫副程式，可以讓程式精簡易讀。



=

副程式

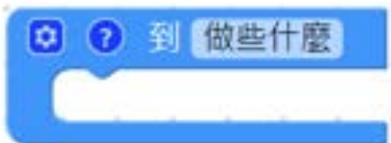
像資料夾的概念，把一長串積木程式包起來變成一個積木圖塊代表，並為這個代表積木命名，方便後續使用。

2-4 程式範例：小星星

此小節會使用到的積木樣式及功能說明



積木圖示



功能說明

- 可於左側積木控制區「副程式」中找到。
- 副程式積木，由主迴圈呼叫副程式，可以讓程式簡潔易讀。

蜂鳴器小星星範例程式及說明積木

程式積木 小星星

步驟說明

The screenshot shows the Blockly IDE interface. On the left, a sidebar contains various block categories. A red box highlights the 'New Program' button, with a callout bubble labeled 'Step 1'. In the workspace, a 'do something' block is highlighted with a red box. A red arrow points from this block to a 'do something' block in the workspace where the name 'partA' has been entered. A callout bubble labeled 'Step 2' points to this modified block. A dashed red arrow points from the modified block to a 'partA' block in the workspace.

Step 1.
開新檔案，可於左側積木控制區「副程式」群組，拖曳副程式。

Step 2.
將名稱修改為「partA」，您會發現灰色積木區會自動增加一個「partA」積木。



程式積木 小星星

步驟說明

The screenshot shows the Blockly programming environment. On the left, the 'Program' (程式積木) palette is visible, with the 'partB' block highlighted. The main workspace contains a sequence of blocks: a 'do something' (做些什麼) block, a 'return long' (回傳 long) block, an 'if-then' (如果...回傳) block, and two 'part' blocks labeled 'partA' and 'partB'. A red box highlights the top 'do something' block. A red arrow points from this box to the 'do something' block within the 'partA' loop. A yellow box highlights the 'do something' block within the 'partB' loop, with a green callout bubble labeled 'Step 3' pointing to it. A dashed red arrow points from the 'partB' block in the workspace to the 'partB' block in the 'partA' loop.

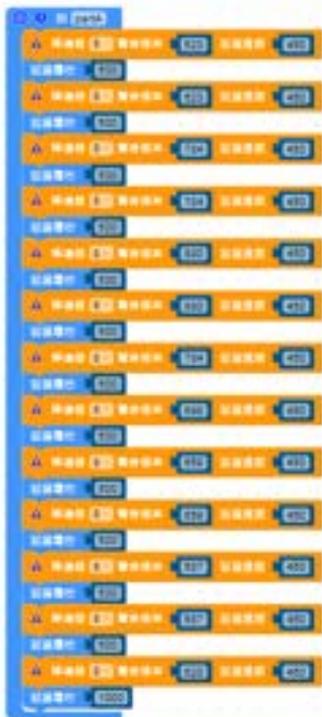
Step 3.
繼續增加副程式「partB」，您會發現灰色積木區會自動增加一個「partB」積木。



程式積木

小星星

步驟說明



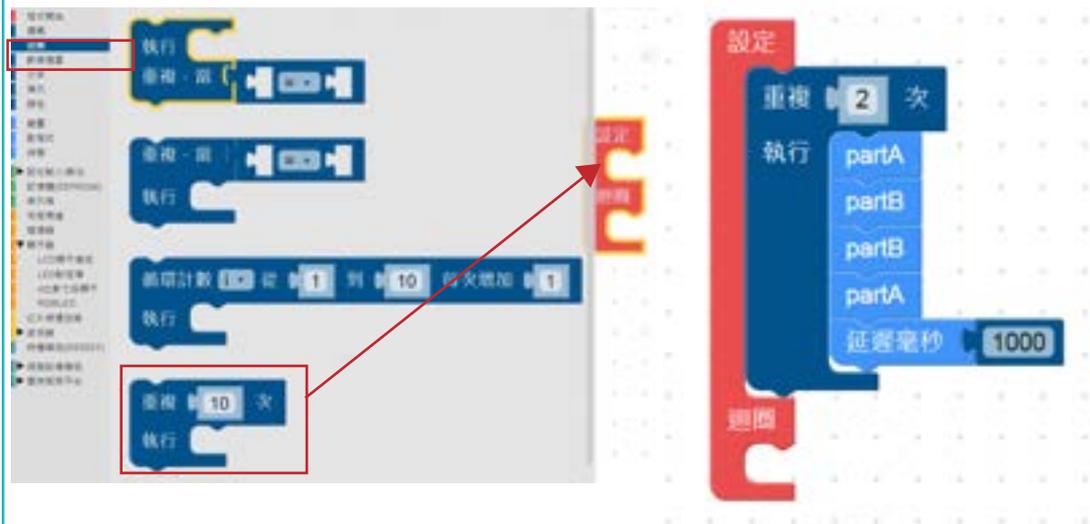
Step 4.

小星星1.2小節與7.8小節是一樣的。3.4小節與5.6小節是重複一樣的。因此，我們1.2小節的旋律程式放在「partA」，3.4小節放在「partB」。

程式積木 小星星

步驟說明

Step 5. 依照左圖完成。



這時候您再開啟積木範例→小星星，您就可以明白副程式的寫法。從主程式看到它是放在「設定」區重複2次，執行副程式「partA」「partB」「partB」「partA」，最後延遲1000毫秒。



2-5 小試身手：請設計一首演奏曲：瑪莉有隻小羊，演奏2次後停止。

瑪莉有隻小羊 Mary Had a Little Lamb

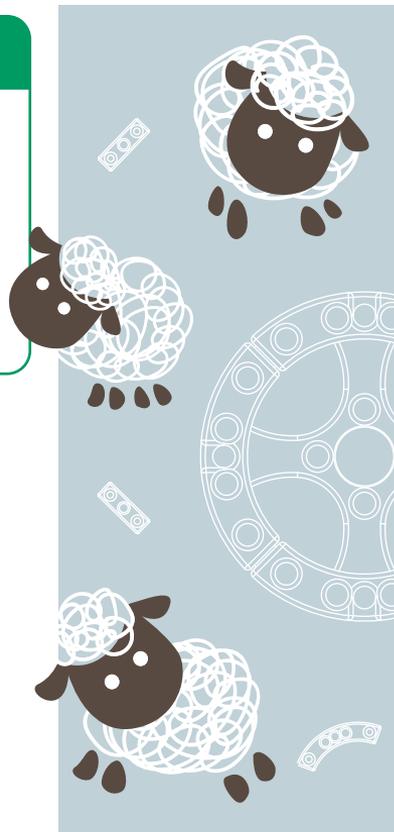
1=C $\frac{4}{4}$ ♩=96

| 3 2 1 2 | 3 3 3 - | 2 - 2 - | 3 3 3 - |

| 3 2 1 2 | 3 3 3 1 | 2 2 3 2 | 1 - - - ||

八度音域	半音	1	2	3	4	5	6	7	8	9	10	11	12
	唱名	Do	Do#	Re	Re#	Mi	Fa	Fa#	So	So#	La	La#	Si
	代號	C	CS	D	DS	E	F	FS	G	GS	A	AS	B
2	頻率	65	69	73	78	82	87	93	98	104	110	117	123
	簡譜	1̇		2̇		3̇	4̇		5̇		6̇		7̇
3	頻率	131	139	147	156	165	175	185	196	208	220	233	247
	簡譜	1		2		3	4		5		6		7
4	頻率	262	277	294	311	330	349	370	392	415	440	466	494
	簡譜	1		2		3	4		5		6		7
5	頻率	523	554	587	622	659	698	740	784	831	880	932	988
	簡譜	1̇		2̇		3̇	4̇		5̇		6̇		7̇
6	頻率	1047	1109	1175	1245	1319	1397	1480	1568	1661	1760	1865	1976
	簡譜	1̇		2̇		3̇	4̇		5̇		6̇		7̇

表·音階與頻率表



【數位輸入控制(開關)】

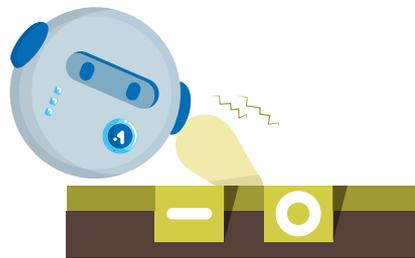
3-1 訊號的種類

數位訊號Digital & 類比訊號Analog

Arduino控制器接受的外部輸入訊號，可以分為二大類；一類是數位訊號(Digital)，另一類是類比訊號(Analog)。數位輸入訊號，它有兩種狀態：導通ON，不導通OFF。Arduino控制板以5V為高電位，因此輸入值為「0」代表低電位、0V。輸入值為「1」代表高電位、5V。我們整理數位輸入訊號的表示與判別方式如下：

程式判別	導通	不導通
電路	ON	OFF
電壓	0V, LOW	5V, HIGH
數位訊號	0	1
邏輯, 布林	true	false

表. 數位訊號程式判別



數位訊號感測器

我們稱為「開關」的元件，訊號都是數位訊號。常見的開關如按鈕開關，微動開關，搖頭開關、指撥開關等。感測器中也有屬於數位開關型態，例如：

種類	感知	訊號
磁簧開關	磁力	導通ON、關閉OFF
霍爾開關	磁力	導通ON、關閉OFF
微動開關	碰撞、位置	有NC, NO可以選擇
滾珠、水銀開關	傾斜、位置	導通ON、關閉OFF
振動開關	振動	導通ON、關閉OFF
光遮斷開關	紅外線	導通ON、關閉OFF

表. 數位感測器舉例

類比訊號感測器

測量0~5V的電壓變化



轉換成0~1023的數值
讓程式做後續的判別應用

類比訊號感測器

至於在我們環境中可以偵測到的訊號大部分是類比訊號(Analog)，例如溫度、溼度的變化，聲音的大小等等，它是一種連續性的變化值。

因為我們的系統是以數位訊號處理，因此轉換類比訊號成爲數位訊號A/D (Analog to Digital)，在Arduino控制板的A0~A6共7個類比腳位用來做類比感測輸入。它可以測量0~5V的電壓變化轉換成0~1023的數值。

接下來的章節，我們以按鈕開關來練習控制程式流程~

將控制板配置的D12腳位安裝上按鈕開關，當處於復歸狀態，偵測D12腳位會得到回傳值爲「1」；當開關被按下時偵測D12腳位會得到回傳值爲「0」。



3-2 程式範例：使用序列埠監控視窗，讀取控制板回傳狀態

此小節會使用到的積木樣式及功能說明



積木圖示	功能說明
	<ul style="list-style-type: none"> ● 可於左側積木控制區「串列埠」中找到 ● 設定串列埠傳輸資料速率為9600 bps
	<ul style="list-style-type: none"> ● 可於左側積木控制區「串列埠、腳位輸入輸出-數位」中找到 ● 控制板印出讀取D12腳位的訊息後換行
	<ul style="list-style-type: none"> ● 可於左側積木控制區「文字」中找到 ● 輸入字串
	<ul style="list-style-type: none"> ● 可於左側積木控制區「文字」中找到 ● 字串組合

程式積木 使用序列埠監控視窗，讀取Cageboard控制板回傳狀態

步驟說明

- Step 1.** 開啟積木範例「類比讀入」。
- Step 2.** 設定串列埠傳輸資料速率為9600 bps。
- Step 3.** 由積木「腳位輸入/輸出」群組，拖曳「數位讀出腳位」積木，取代原「類比讀出腳位」積木。

The screenshot shows the Blockly IDE interface. On the left, a list of blocks is visible, with '類比讀入' (Analog Input) highlighted in a red box. A red callout bubble labeled 'Step 1' points to this block. In the workspace, a '設定串列埠' (Set Serial Port) block is shown with 'serial' selected and '9600 bps' set. A red callout bubble labeled 'Step 2' points to this block. Below it, a '類比讀出腳位' (Analog Read Pin) block is highlighted in a red box. A red callout bubble labeled 'Step 3' points to a '數位讀出腳位' (Digital Read Pin) block being dragged from the '腳位輸入/輸出' (Digital Input/Output) category. A red dashed arrow labeled '取代' (replace) points from the new block to the old one. The workspace also contains a '延遲毫秒' (Delay in milliseconds) block set to 1000 and an '印出訊息後換行' (Print message and then line feed) block.

程式積木 使用序列埠監控視窗，讀取Cageboard控制板回傳狀態

步驟說明

Step 4.

於左側積木程式區「文字」中找到「字串組合」與「輸入字串」的積木，並拖拉至圖示位置。

Step 5.

修改原積木程式，數位讀出腳位改為12，序列埠印出「字串組合」D12=與讀取值，以增加可讀性。

Step 6.

延遲300毫秒。

Step 7.

按  下載儲存XML檔: 序列埠讀取D12.XML

The screenshot shows the Blockly programming interface. On the left, the '文字' (Text) category contains blocks for '字串組合' (String Concatenation) and '輸入字串' (Input String). These are being dragged into the workspace. The workspace contains a sequence of blocks: a '設定串列埠' (Set Serial Port) block with 'serial' selected and '9600 bps' for the baud rate; an '印出訊息後換行' (Print Message and New Line) block with '字串組合' selected and the text 'D12=' followed by '數位讀出腳位' (Digital Read Pin) set to 12; and a '延遲毫秒' (Delay in milliseconds) block set to 300. Red dashed lines and callouts indicate the steps: Step 4 points to the '字串組合' and '輸入字串' blocks in the left palette; Step 5 points to the '數位讀出腳位' field in the '印出訊息' block, which is set to 12; Step 6 points to the '延遲毫秒' field, which is set to 300.

3-3

程式範例：按鈕控制LED，按一次LED亮，再按一次LED熄滅

此小節會使用到的積木樣式及功能說明



積木圖示	功能說明
	<ul style="list-style-type: none"> ● 可於左側積木控制區「迴圈」中找到。 ● 設定先執行工作、再判斷條件的迴圈積木。 ● 先執行一次迴圈內的工作內容，再判斷重複條件是否成立：當成立時再執行一次迴圈內的工作內容，否則跳出迴圈。
	<ul style="list-style-type: none"> ● 可於左側積木控制區「迴圈」中找到。 ● 設定先判斷條件、再執行工作的迴圈積木。 ● 先判斷執行迴圈的條件是否成立：當成立時執行一次迴圈內的工作內容，否則跳出迴圈。
	<ul style="list-style-type: none"> ● 可於左側積木控制區「邏輯」中找到 ● 回傳數學邏輯比較是否成立的積木。 ● 目前有等於(=)、不等於(≠)、小於(<)、小於或等於(\leq)、大於(>)、大於或等於(\geq)等六種數學判斷可選擇。

程式積木 按鈕控制LED燈

步驟說明



Step 1

開啟範例程式後，依照前述教學，拖拉、複製對應圖塊，修改成右圖樣式。



Step 2

Step 3

Step 4

Step 4

Step 5

- Step 1. 開啟積木範例「LED閃爍」修改。
- Step 2. 設定開始時LED為熄滅狀態。
- Step 3. 當D12=1，也就是按鈕未被按下，反覆執行「甚麼也不做」，直到按鈕被按下，D12=0時跳出重複迴圈。
- Step 4. 執行LED為亮白色狀態，延遲500毫秒。
- Step 5. 當D12=1，也就是按鈕未被按下，執行「甚麼也不做」，直到按鈕被按下，D12=0跳出重複迴圈。
- Step 6. 執行LED為熄滅狀態，延遲500毫秒
- Step 7. 按  下載儲存XML檔: 按鈕控制LED明滅.XML。

這個程式應用了迴圈來確認按鈕的按下，控制2段的LED明與滅的流程。但是這程式有一個缺點，就是當按著按鈕不放D12=0，LED會以500毫秒間隔明與滅；原因就是長按按鈕時，重複迴圈功能就失效了。我們來改良程式設計讓控制更穩定。

3-4 程式範例：按鈕與變數控制LED

此小節會使用到的積木樣式及功能說明



積木圖示

建立變數...



功能說明

- 可於左側積木控制區「變數」中找到
- 變量積木控制區，用以建立新的變數。

承上一步驟輸入「STATE」建立變數名稱後，自動出現宣告變數名稱、型態與初始值的積木。

《宣告》：設定變數名稱。

《當》：設定變數型態。

《資料》：設定變數的初始值。



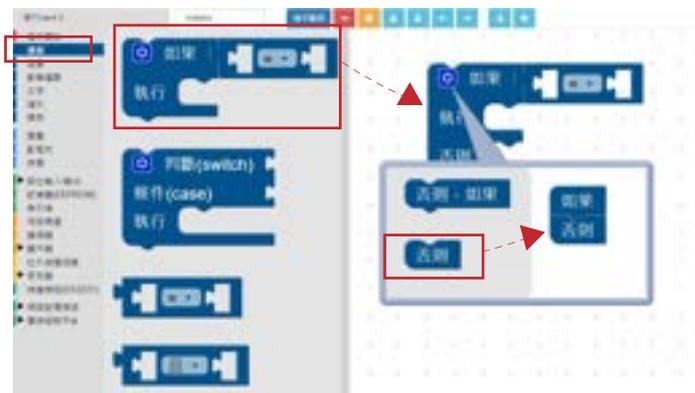
積木圖示



功能說明

- 變數型態：

long =長整數	float =浮點數
unsigned long =無符號長整數	int =整數
unsigned int =無符號整數	char =字元
string =字串	bool =布林邏輯
byte =位元	



- 可於左側積木控制區「邏輯」中找到，再按<齒輪>按鈕，修改條件式，增加一個條件 **否則**。
- 再按<齒輪>按鈕，收合改造選單

程式積木 按鈕與變數控制LED

步驟說明

Step 1. 開啟「按鈕控制LED明滅.XML」修改。

(前一小節完成的程式檔案)

Step 2. 設定開始時LED為熄滅狀態，建立新變數STATE，宣告變數STATE為布林bool型態，初始為0。

Step 3. 條件式如果D12=0，也就是按鈕被按下，執行「延遲20毫秒」(註1)，進入重複迴圈，直到再次按鈕D12=0才跳出迴圈。

Step 4. 執行變更變數STATE為<非>STATE，也就是當STATE=0，則改變成1；當STATE=1，則改變成0，互相切換。

Step 5. 條件式當STATE=1，執行LED為亮白色狀態。

Step 6. 否則STATE=0，執行LED為熄滅狀態。

Step 7. 按 下載儲存XML檔: 按鈕與變數控制LED.XML

程式經過修改後，如果按下按鈕，要一直到確認放開，才執行變數STATE變更狀態，由於指定變數STATE為布林bool型態，也就是非0即1 (布林bool型態就是0, 1 / true, false) 2種狀態。利用這2種狀態控制切換LED開與關。

這種程式控制模式十分穩定。因此，我們可以參照按鈕穩定的程式控制方式，設計各種不同的按鈕開關控制應用。

**註1:**

由於機械開關的特性，由某一接點投擲至另一接點的瞬間，會產生多次的彈跳現象，無論是接點由ON至OFF或由OFF至ON，接點會經過閉合→打開→閉合→打開→…最後到穩定狀態的過程，此一現象稱為「開關彈跳」現象，這會造成控制板的誤判。



(開關彈跳現象示意圖)



開關的彈跳時間通常不大於20毫秒(10毫秒~20毫秒)，因此加入程式「延遲20毫秒」，藉以消除開關彈跳。

3-5

程式範例：按鈕與變數控制七彩霓虹燈

請參考 1-3. 小試身手：七彩霓虹燈 與 3-4. 程式範例：按鈕與變數控制LED
改成每按一次按鈕，依序切換紅橙黃綠藍靛紫色彩變化。



此小節會使用到的積木樣式及功能說明



積木圖示

功能說明



- 先於左側積木控制區「變量」-「建立變數」，輸入STATE後可以得到左圖。
- 變數STATE型態整數，初始為-1

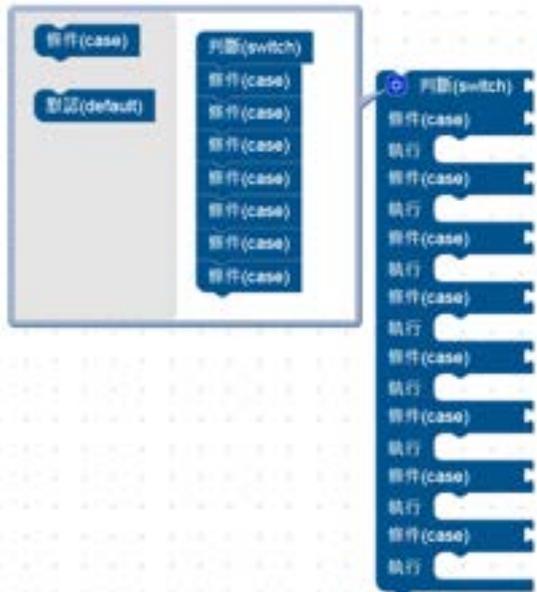


- 可於左側積木控制區「數學運算」中找到左圖深藍色外框。而STATE淺藍方塊可於「變量」中找到。
- 回傳值為STATE除以7的餘數

此小節會使用到的積木樣式及功能說明



積木圖示



功能說明



- 可於左側積木控制區「邏輯」中找到。
- 按積木左上齒輪按鈕來修改條件式，拖拉條件積木至判斷下，總共增加7個判斷條件，再按<齒輪>按鈕，收合改造選單。



程式積木 按鈕與變數控制七彩霓虹燈

步驟說明



Step 1. 開啟「按鈕與變數控制LED.XML」，再開啟「七彩霓虹燈.XML」，**注意：**彈出對話框要按<取消>，合併開啟2個程式在積木堆疊區。



Step 2. 變更變量STATE型態為整數，初始為-1



Step 3. 回傳值為STATE除以7的餘數

程式積木 按鈕與變數控制七彩霓虹燈

步驟說明

The image shows a Scratch script for controlling a 7-color rainbow LED. The script is divided into several sections:

- 設定 (Initialize):** Two 'WS2812 LED' blocks are used to initialize the LED strip. The first block sets the pin to 13, the number of LEDs to 2, and the color to red. The second block sets the pin to 13, the number of LEDs to 2, and the color to orange. Both blocks are labeled 'Step 4'.
- 迴圈 (Loop):** A '如果' (If) block is used to check if the digital input pin 12 is 0. This block is labeled 'Step 5'. Inside the loop, there is an '執行' (Execute) block for '設置毫秒' (Set milliseconds) to 20, labeled 'Step 6'. Below it is a '重複' (Repeat) block for '數位輸出數位' (Digital output digital) with the value 1, labeled 'Step 7'.
- 判斷 (Switch):** A '判斷 (switch)' block is used to check the value of the variable 'STATE' modulo 7. It has two cases:
 - 條件 (case) 0:** Labeled 'a', it contains two 'WS2812 LED' blocks. The first sets the color to red, and the second sets the color to orange.
 - 條件 (case) 1:** Labeled 'b', it contains two 'WS2812 LED' blocks. The first sets the color to orange, and the second sets the color to yellow.

Step 4. 設定開始時LED為熄滅狀態，宣告變數STATE為整數型態，初始值為-1

Step 5. 條件式如果D12=0，也就是按鈕被按下，執行「延遲20毫秒」，進入迴圈，直到D12=0按鈕再次被按下才跳出迴圈。

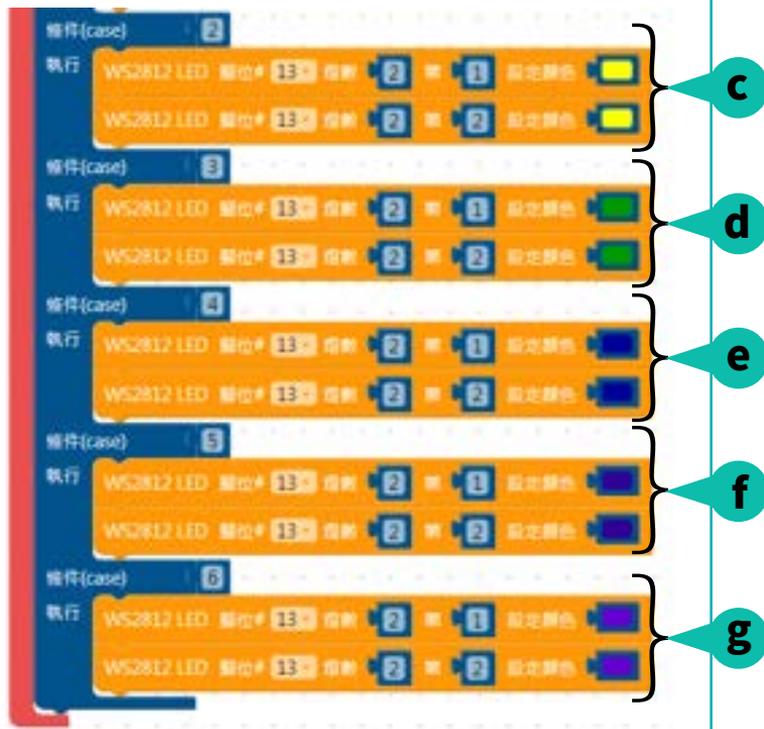
Step 6. 執行變更變數STATE為STATE+1，也就是當STATE=-1，執行1次則改變成0；再執行1次則STATE改變成1

Step 7. 條件式(switch)，回傳值為STATE除以7的餘數
Step 8.

- a. 當條件(case)為0，執行LED紅色
- b. 當條件(case)為1，執行LED橙色

程式積木 按鈕與變數控制七彩霓虹燈

步驟說明



- c. 當條件(case)為1，執行LED黃色
- d. 當條件(case)為2，執行LED綠色
- e. 當條件(case)為3，執行LED藍色
- f. 當條件(case)為4，執行LED靛色
- g. 當條件(case)為5，執行LED紫色

Step 9. STATE值累加=7時，求餘數得0，又回到程式條件(case)為0，執行LED紅色，依此重複執行

Step 10. 按 [↓](#) 下載儲存檔案: 按鈕控制七彩霓虹燈.XML



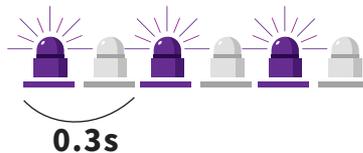
3-6 小試身手：請設計程式當第一次按下按鈕，演奏：瑪莉有隻小羊到結束，第二次按下按鈕，則LED亮紫色快閃3次間隔0.3秒，再回到等待按鈕，依此重複執行。



按下第一次按鈕



按下第二次按鈕



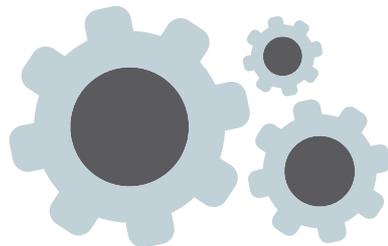
瑪莉有隻小羊 Mary Had a Little Lamb

$1 = C \frac{4}{4}$ $\text{♩} = 96$

3	2	1	2		3	3	3	-		2	-	2	-		3	3	3	-	
3	2	1	2		3	3	3	1		2	2	3	2		1	-	-	-	

可以叫出先前練習儲存的程式
也可重新再練習拉一次數值喔!





【直流馬達控制】

4-1 控制板直流馬達控制模式

在控制板上配置的馬達驅動與控制晶片TB6612FNG，內部包含兩組H橋式MOSFET電路，可驅動和控制兩個小型直流馬達，單一通道可輸出1.2A(極限3.2A)電流，輸出/輸入功率比值高達91.74%。將控制板的D5.D7用於控制直流馬達M01，D6.D4控制直流馬達M02，控制模式如下表：

馬達	腳位	轉向與轉速控制
M01	D7	高:順時針轉 / 低:逆時針轉
	D5	轉速:PWM值/停止0
M02	D4	高:順時針轉 / 低:逆時針轉
	D6	轉速:PWM值/停止0

表 . 控制板直流馬達控制模式

轉向以視線正向轉動軸心觀察為準

備祥儀公司製作的微型減速直流馬達，內裝金屬齒輪減速箱，耐磨、壽命長，特殊輸出軸承設計，金屬單出D型軸，輸出轉速誤差小(10%以內)。

微型減速直流馬達基本性能說明：

- 額定電壓: DC 4.5V
- 額定扭力: 0.8~1kg
- 空載電流: 0.3A以下
- 空載轉速: 160rpm ± 10%
- 適用電壓: DC 3V~6V
- 減速比: 1/55



4-2 模擬類比輸出PWM與馬達轉速控制

所謂PWM是(Pulse Width Modulation)的縮寫，是將訊號編碼於脈波寬度上的一種技術，此技術以數位方式來模擬類比訊號，也就是俗稱的變頻。PWM訊號中，脈波寬度在整個週期所占的高電位的波型在整個週期中所占的比例。Arduino Nano能夠做模擬類比輸出PWM 的數位接腳 (3, 5, 6, 9, 10, 11)，使用analogWrite() 函數，值由0~255，來調變電壓值。

如右側說明，雖然控制轉速PWM值可以由0~255，但是馬達有啟動電壓的限制，經過實測，Mars E1 戰神探險車配置的微型直流馬達，約在PWM值50左右才能轉動，最高值為255。而讓馬達停止，則PWM值輸入0。



PWM與analogWrite()值的對應

0% Duty Cycle - analogWrite(0)



25% Duty Cycle - analogWrite(64)



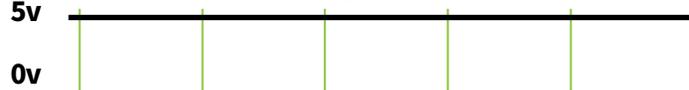
50% Duty Cycle - analogWrite(127)



75% Duty Cycle - analogWrite(191)



100% Duty Cycle - analogWrite(255)

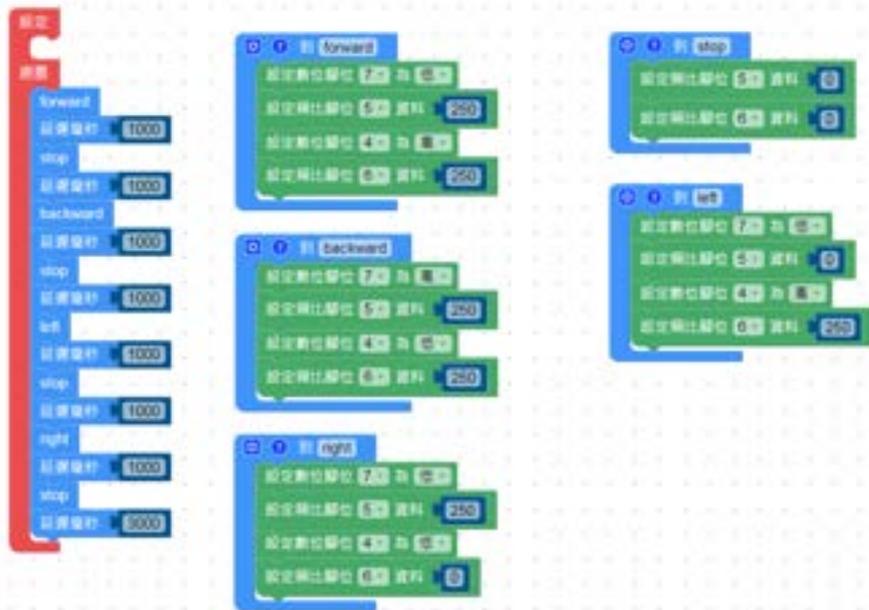


4-3 程式範例：雙馬達車測試

確認戰神探險車左馬達連接M01插座，右馬達連接M02插座



積木圖示



功能說明

Step 1.

開啟積木範例「雙馬達車測試」

本程式採用副程式結構，從主程式判讀，流程為：

前進(forward)	1 秒鐘
停止(stop)	1 秒鐘
後退(backward)	1 秒鐘
停止(stop)	1 秒鐘
左轉(left)	1 秒鐘
停止(stop)	1 秒鐘
右轉(right)	1 秒鐘
停止(stop)	3 秒鐘



積木圖示



功能說明

Step 2.

如左圖，我們要命令戰神探險車前進時，左馬達 (M01)要逆時針轉，右馬達 (M02)要順時針轉

Step 3.

副程式「forward」

左馬達(逆時針轉) D7=低，D5=250(PWM)

右馬達(順時針轉) D4=高，D6=250(PWM)



程式積木

步驟說明



Step 4.

副程式「backward」

左馬達(順時針轉) D7=高, D5=250(PWM)

右馬達(逆時針轉) D4=低, D6=250(PWM)



Step 5.

副程式「left」

左馬達(停止) D7=低, D5=0

右馬達(順時針轉) D4=高, D6=250(PWM)



程式積木



步驟說明

Step6.**副程式「right」**

左馬達(逆時針轉) D7=低，D5=250(PWM)

右馬達(停止) D4=低，D6=0

**Step7.****副程式「stop」**

左馬達(停止) D5=0

右馬達(停止) D6=0

Step8.

請上傳燒錄，並移除USB線，開啟控制板電源開關，將戰神探險車放在地板上測試看看！

4-4

程式範例：測試馬達變速

在這個範例，我們將學習使用變數來變更PWM值，並用序列埠監控視窗讀取PWM值，這可以測試得到控制戰神探險車移動速度的數據，作為進階控制時，設計最佳化的依據。

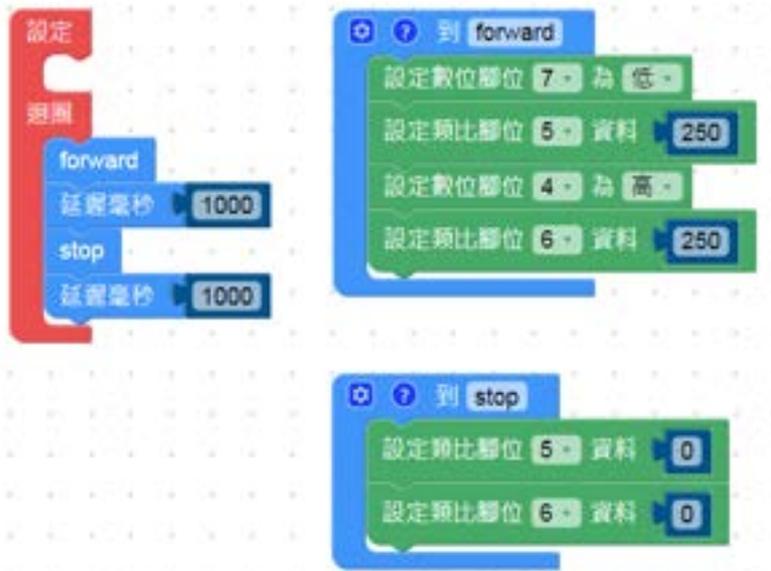
此小節會使用到的積木樣式及功能說明

積木圖示	功能說明
	<ul style="list-style-type: none"> • 可於左側積木控制區「變量」中找到 • 建立變數PWM，宣告PWM型態整數，初始為0
	<ul style="list-style-type: none"> • 可於左側積木控制區「迴圈」中找到 • 迴圈群組，建立循環計數迴圈i從0到255，每次增加5。變數i是區域變數，僅對計數迴圈有效



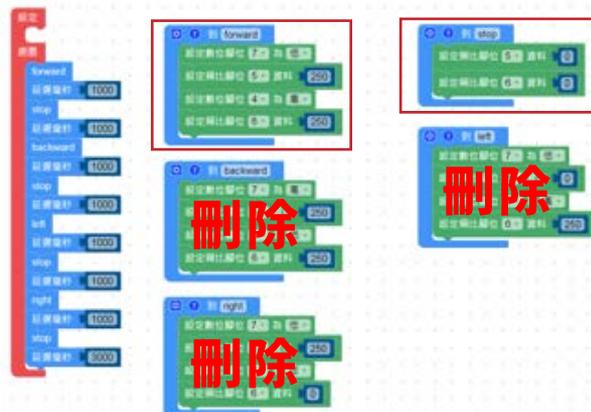
程式積木 測試馬達變速

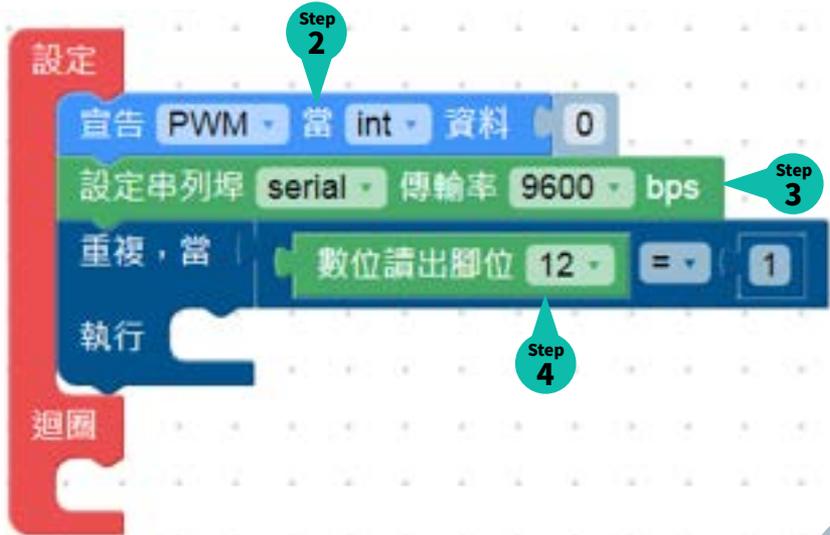
步驟說明



Step 1.

開啟積木範例「雙馬達車測試」，保留前進(forward)與停止(stop)，其餘刪除。



程式積木 測試馬達變速	步驟說明
 <p>設定</p> <p>宣告 PWM 當 int 資料 0</p> <p>設定串列埠 serial 傳輸率 9600 bps</p> <p>重複, 當 數位讀出腳位 12 = 1</p> <p>執行</p> <p>迴圈</p>	<p>Step 2. 建立變數PWM，宣告PWM型態為整數，初始為0，拖曳積木到「設定」中</p> <p>Step 3. 從串列埠群組，拖曳設定串列埠積木到「設定」中</p> <p>Step 4. 從迴圈群組中，拖曳《重複，當…執行》的迴圈至「設定」中，設計讀取按鈕D12被按下才脫離，進入主迴圈loop()</p>



這是貼心的小技巧，避免燒錄後，戰神探險車會自己亂跑，要學起來喔！

程式積木 測試馬達變速

步驟說明

**Step 5.**

拖曳與建立循環計數迴圈，變數i從0到255，每次增加5

Step 6.

賦值變數PWM=i值

Step 7.

從「串列埠」中找到「印出訊息後換行」，加上「文字」中的「字串組合」，將訊息”PWM=”+變數PWM列印到序列埠

Step 8.

執行副程式forward，延遲1000毫秒

Step 9.

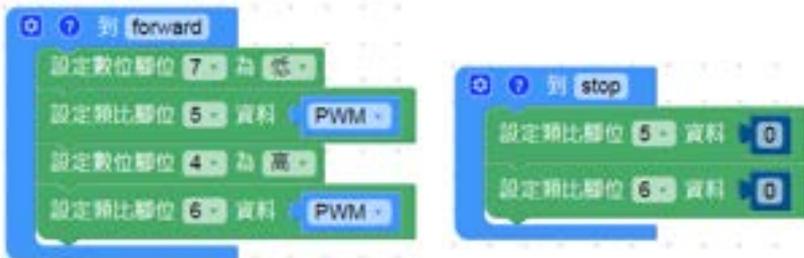
執行副程式stop，延遲1000毫秒


 程式積木 **測試馬達變速**

步驟說明


Step 10.

複製與修改循環計數迴圈，變數 i 從 255 到 0，每次增加 -5


Step 11.

修改副程式 forward，D5 與 D6 值 = PWM 值

Step 12.

副程式 stop 不變更

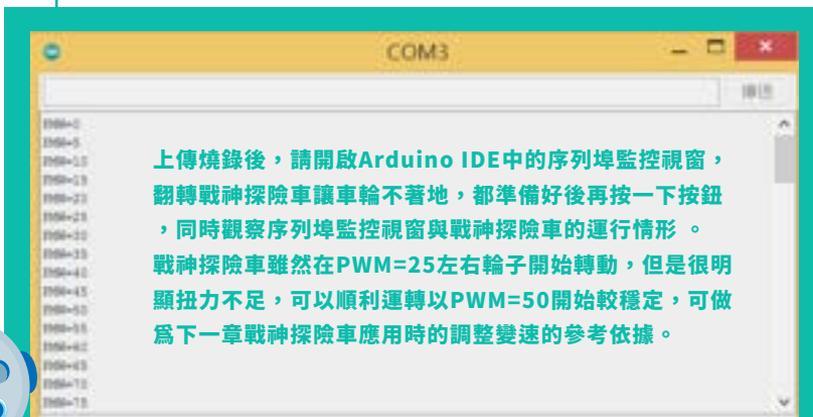
程式積木 測試馬達變速

步驟說明



Step 13.
重新組合程式積木，完成

Step 14.
按 [↓](#) 下載儲存檔案：測試馬達變速.XML



上傳燒錄後，請開啟Arduino IDE中的序列埠監控視窗，翻轉戰神探險車讓車輪不著地，都準備好後再按一下按鈕，同時觀察序列埠監控視窗與戰神探險車的運行情形。戰神探險車雖然在PWM=25左右輪子開始轉動，但是很明顯扭力不足，可以順利運轉以PWM=50開始較穩定，可作為下一章戰神探險車應用時的調整變速的參考依據。





4-5 小試身手：請設計唱歌跳舞動感探險車。

01.
蜂鳴器演奏音樂



02.
LED燈隨著音樂撥放
變換顏色



請綜合本章基礎控制所學的所有功夫，以範例積木：「雙馬達車測試」做基礎。設計一台會唱歌跳舞，轉轉身體的探險車。當按下按鈕，開始唱歌閃LED燈，演奏每一小節音樂，LED隨著變換顏色，然後左右搖搖0.5秒。

03.
左右搖晃0.5秒



恭喜您完成全部的學習囉！



The logo for CAGEBOT, featuring the brand name in a bold, sans-serif font with a stylized robot head icon integrated into the letter 'A'.

CAGEBOT

A central message box with a dashed border containing congratulatory text. The background is a vibrant yellow and orange gradient with various geometric patterns and icons like gears, a curved sensor strip, and a cube.

恭喜您已完成教學手冊所有章節
未來，請依照所學繼續擴充學習
增加其他感測器應用



