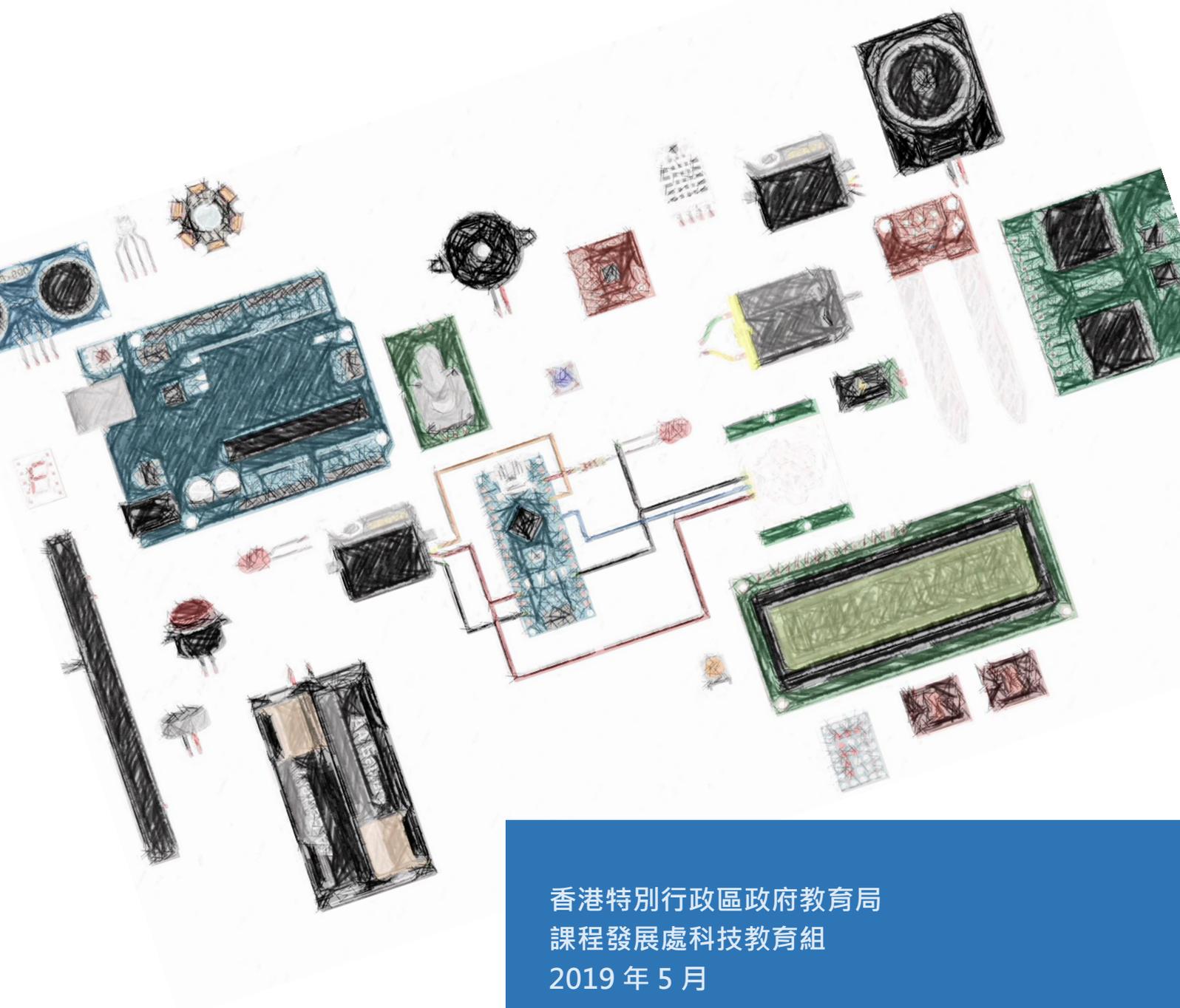


初中「機械人」學習資源

延伸單元一：機械人的通訊和自動化



香港特別行政區政府教育局
課程發展處科技教育組
2019年5月

如對本學與教資源有任何意見及建議，請致函：

香港九龍塘沙福道 19 號西座 W101 室

教育局課程發展處科技教育組

總課程發展主任（科技教育）

本學習資源版權，除在鳴謝頁所列舉的圖片外，全屬於香港特別行政區政府教育局擁有。

教育局歡迎學校等教育團體使用本學習資源的內容作非牟利的教學用途。任何情況下使用本學習資源，需作出鳴謝，教育局保留本學習資源版權。

未經本局事先允許，不能以任何形式使用其中教材作出版或其他用途，否則教育局將保留一切追究的權利。

© 版權所有 2019

本學習資源由香港機械人學院製作。

目錄

章節一：通訊和自動化基礎	P.3
章節二：無線通訊 – 藍牙	P.7
章節三：無線通訊 – APP Inventor	P.10
章節四：單元專題研習 – 拯救機械人	P.20

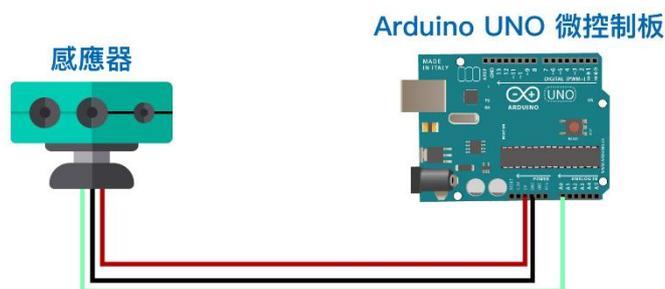
延伸單元一：機械人的通訊和自動化

章節一：通訊和自動化基礎

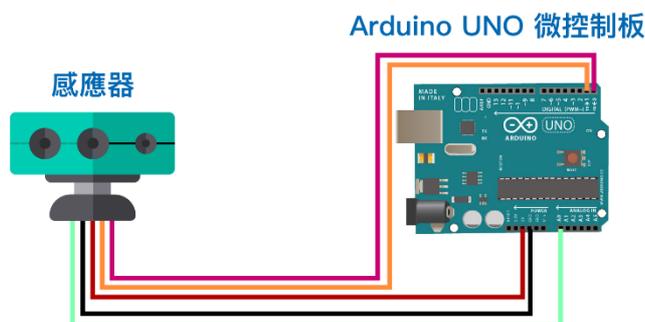
在設定微處理器、輸入和輸出的時候，通常會涉及資料傳輸的情況，例如上一個單元學過的溫濕度感應器，我們需要透過感應器讀取讀數，然後傳回微處理器。微處理器作分析之後作出有邏輯的輸出，中間涉及兩次資料傳輸。有時候我們會使用無線通訊硬件設計機械人，日常生活中，機械人與我們息息相關，例如：細心觀察地鐵站內的機械裝置都有機械人系統的存在。又例如：八達通機裝置已設定為自動化，能夠識別不同的八達通卡並作記錄或扣款，對人類生活有極大影響。這一單元會探討透過不同的通訊介面和自動化的基礎概念，讓我們能夠製作半自動化或全自動化的機械人。

I. 機械人的通訊

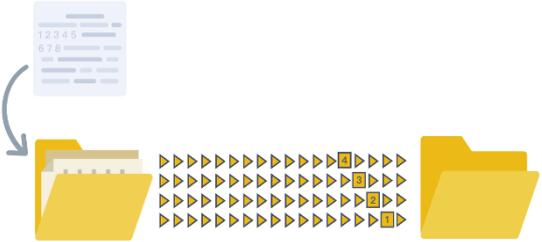
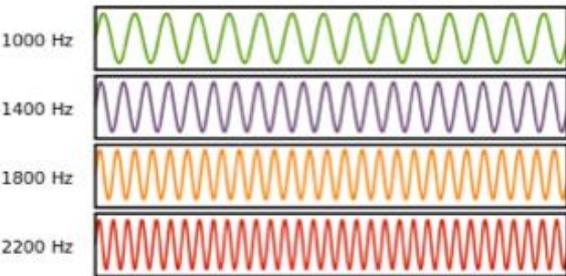
所謂通訊方法，即是在電子元件如感應器與控制板之間傳輸資料的方法。我們將會介紹三種常用的通訊介面：串行通訊、並行通訊以及無線通訊。串行通訊 (Serial Communication) 是一次發送一個位元數據的過程，與並行通信形成對比。並行通訊 (Parallel Communication) 是一種同時傳送多個二進制數字的方法。在並行通訊中，八個二進制數字 (byte) 作為一個整體發送，在具有多個並行信道的鏈路上發送。



串行通訊 (Serial Communication)



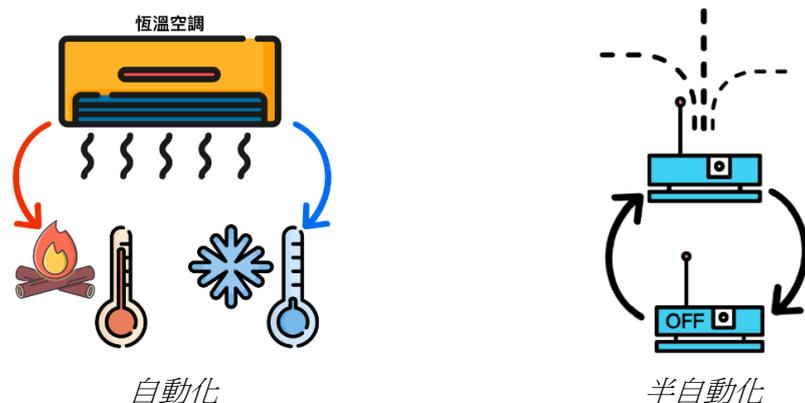
並行通訊 (Parallel Communication)

<p>串行通訊 (Serial Communication)</p>	<p>假如文件使用串行通訊的話，我們可以想像成只有一條管道同時傳輸四個數字，根據這個特性我們可以總結出：並行通訊速度比較慢，因為只有一條管道負責資料傳輸，所以設定上就比較簡單。</p>  <p>串行通訊在現實生活中的應用同樣十分廣泛，例如我們所使用的溫濕度感應器就有一條插腳，除了電源和接地，亦有另外一條線負責收集數據。</p>
<p>並行通訊 (Parallel Communication)</p>	<p>假如文件使用並行通訊的話，我們可以想像成有四條管道同時傳輸四個數字，根據這個特性我們可以總結出：並行通訊速度比較快，但因為同時有幾個管道負責資料傳輸，所以設定上比較複雜。</p>  <p>並行通訊在現實生活中的應用十分廣泛，例如我們在遊戲機所使用的控制桿就有五隻插腳，除了電源和接地，亦有另外三條線負責收集數據。早期的隨機存取記憶體 (RAM)，即一條長條型插腳的物體，同樣也利用了並行通訊的原理。</p>
<p>無線通訊 (Wireless Communication)</p>	<p>無線訊號基本上是利用不同頻段的電磁波來進行資料傳輸，像是收音機、無線電視、藍牙 (Bluetooth)、行動電話和無線網路都屬無線通訊。依據不同的頻率，無線通訊會有不同的使用特性和應用場合，例如民航機常用的 VHF (Very High Frequency)。</p> 

II. 自動化

自動化 (Automation) 可以被定義為在沒有人工協助的情況下執行過程或程序的技術。換言之，自動化或自動控制就是使用各種控制系統來操作設備，如機器、工廠的過程、鍋爐和熱處理爐，這些過程已經完全自動化。

自動控制系統的機制主要是反饋機制，我們可以用恆溫空調作為例子。恆溫空調的操作方法是預先設定一個特定的溫度，然後感應器會不斷監察溫度指標是否與我們設定的溫度一致。假如溫度比預設溫度低就會釋放暖氣，反之亦然。自動化在設計機械人中有不可或缺的地位，高階的機械人比賽往往是用自動化系統分高下，例如在世界機械人奧林匹克中，機械人通常需要自動行走及偵測特定物件，然後把一段特定物件拿起再放在適當的位置中。



除了自動化之外，我們還有半自動化。所謂半自動化，一般的意思是解作機械自動完成一次工作循環之後，就會再由人手決定是否重新開始一個新的工作。半自動化也在機械人設計中常見，上一課的專題演習－「清潔機械人」就是一個很好的例子。機械人起初會隨機行走並清潔地板，但之後需要人手把它重設，待有需要的時候重啟；這就是一個半自動化機械人的例子。在這個單元我們會談及利用不同通訊方法達到自動化和或半自動化的機械人。

III. 溫習問題

完成本課後你能夠回答以下的問題嗎？

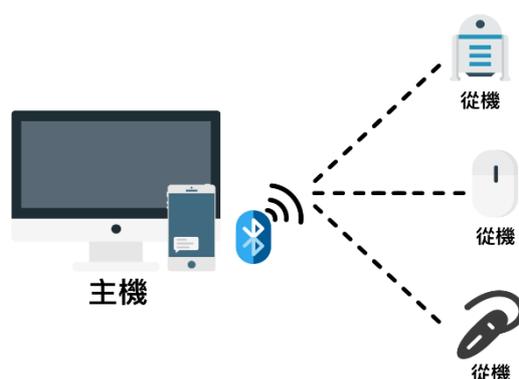
1. 在硬件（例如：Arduino）有那些種通訊方法？
2. 什麼是自動化？

章節二：無線通訊 – 藍牙

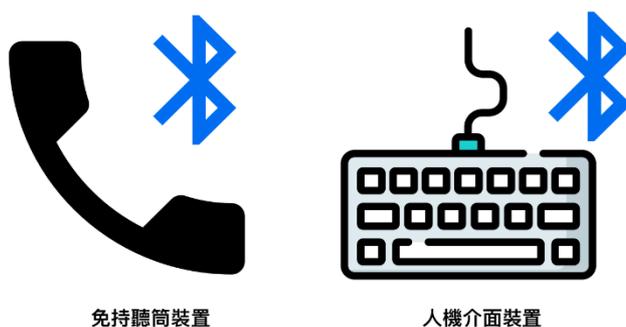
I. 認識藍牙（Bluetooth）



藍牙是一種無線通訊技術標準，一般用於流動設備例如智能電話或平板電腦，在一定距離交換數據。無線通訊的方法一般用特定頻率的無線電波發射數據，而藍牙就是利用一種特高頻率（2.4GHz）的無線電波頻率。藍牙技術由當時幾位電子業界龍頭如索尼愛立信（Sony Ericsson）、諾基亞（Nokia）、東芝（Toshiba）等公司所創立。他們創立了藍牙特別興趣小組（亦即藍牙技術聯盟的前身），希望開發一個成本低、效益高，亦可以在短距離範圍內無線連接的技術標準。

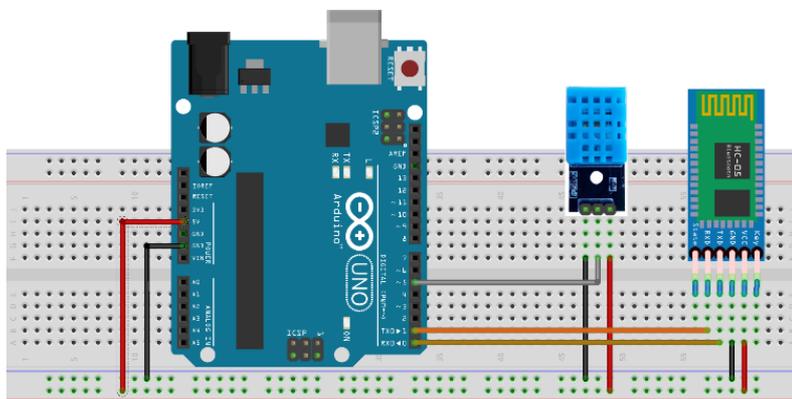


藍牙的應用層面非常廣泛，由可穿戴裝置、電腦、手機、心率監視器、智能家居至汽車都有藍牙技術的應用。藍牙裝置通常分為主機（Master）和從機（Slave）兩大類型，像是電腦和手機的藍牙裝置可以探索其他藍牙裝置並與其配對的，我們稱為主機。從機一般是被動的等待被連接，例如藍牙滑鼠、鍵盤、耳機、機械人等等。藍牙技術聯盟定出的藍牙規範種類有很多，例如：**人機介面裝置規範**，一般用於制定滑鼠鍵盤等；**免手持聽筒裝置規範**，一般用於行動裝置支援語音撥號和重撥，如智能電話。



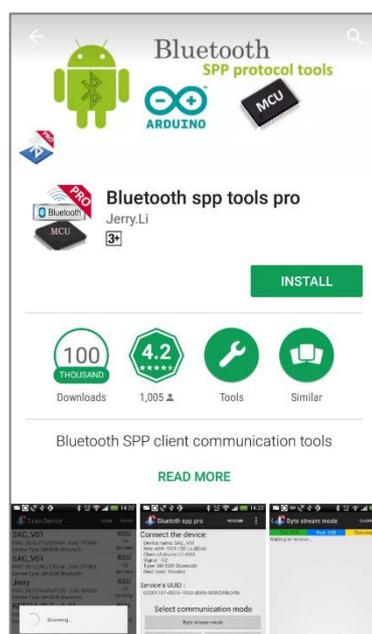
II. 動手做 – 藍牙模組（HC05）基本操作

同學需要使用藍牙模組，讓安卓（Android）手機與 Arduino 控制板進行溝通。這個課節看似複雜，但是程式碼跟之前與電腦作串行通訊溝通是差不多的。我們之前嘗試過 Arduino 控制板與電腦串行通訊，這次的任務就是 Arduino 控制板與藍牙模組串行通信，並發送溫濕度感應器的數據到手機上顯示。



燒錄程式時請緊記把藍牙模組從 Arduino 控制板上移除，因為當藍牙模組跟 Arduino 連接在一起時，藍牙模組會佔用了 Arduino Uno 上唯一的串行訊頻道，令電腦無法跟 Arduino 控制板進行溝通，因此會導致燒錄程式時出現錯誤。我們把藍牙模組的電源接線拔掉或拔掉藍牙模組的傳送（TX）接收（RX）接腳。HC05 的 TX 接腳與 Arduino 控制板的 RX 接腳相連，RX 接腳與 TX 接腳相連。

1. 使用 Google Play 搜尋“Bluetooth spp tools pro”，這個工具能讓手機透過藍牙變成 Serial monitor，從而達到控制 Arduino 控制板的目的。



- 把 Arduino 控制板接上電源，這個時候你會看見 HC05 上的 LED 燈會快速的閃爍，這代表 HC05 進入了配對模式。此時，同學可以拿出安卓手機，開啟藍牙功能，然後打開手機的藍牙列表，找尋 HC05 藍牙名稱，再連接 HC05。若出現要輸入連接密碼時，密碼一般而言都是 1234；若未能成功，可嘗試 0000。
- 以下是程式碼部分：

藍牙觀察溫濕度的程式碼全圖，可參考檔案 3_2_bluetooth_dht11

```

1 #include <dht.h>
2 #include <SoftwareSerial.h>
3 -----
4 SoftwareSerial BTSerial(10,11);
5 dht DHT;
6
7 int dht11_pin = 5;
8
9 float temp_result = 0;
10 -----
11 void setup() {
12     BTSerial.begin(9600);
13 }
14 -----
15 void loop() {
16     DHT.read11(dht11_pin);
17     temp_result = DHT.temperature;
18     BTSerial.print("the current temperature is ");
19     BTSerial.print(temp_result);
20     BTSerial.println("degree celsius");
21 }

```

程式內容說明

- 這段的程式主要是定義變數名稱及包含函數庫。#include <dht.h> 這句的意思是在程式中包含着在 dht.h 檔案內的一切函數定義及變數定義。沒有包含函數庫便不能使用一些快速的方法來提取數值。留意這句的指令是沒有分號的。第二句寫的 dht DHT; 是一個開啟溫濕度感應器的指令，如沒有執行這句，DHT.read11(dht11_pin) 便不能執行。最後，因程式讀回來的數值是包含小數，因此要注意在變數前方必須加上 float 以存儲小數點。

```

1 #include <dht.h>
2 #include <SoftwareSerial.h>
3
4 SoftwareSerial BTSerial(10,11);
5 dht DHT;
6
7 int dht11_pin = 5;
8
9 float temp_result = 0;

```

2. 開啟串行通訊用作發送訊息到藍牙模組。

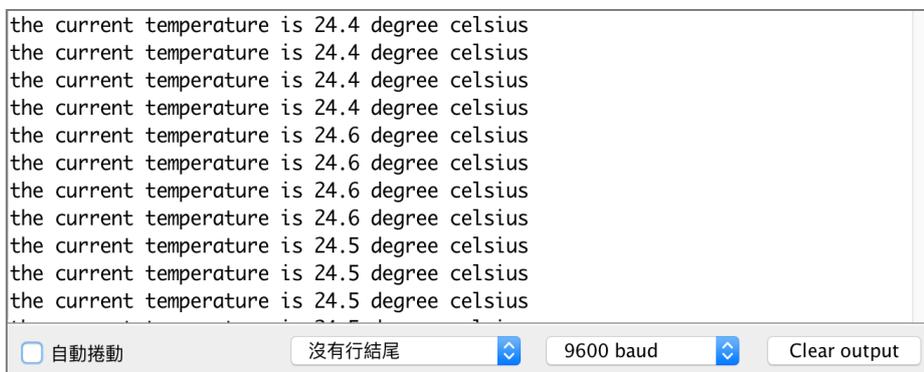
```
11 void setup() {  
12     BTSerial.begin(9600);  
13 }
```

3. 讀取溫度的數值，並把在 DHT11 收回來的數值，發送到串行通訊裝置，這裡即是藍牙模組。

```
15 void loop() {  
16     DHT.read11(dht11_pin);  
17     temp_result = DHT.temperature;  
18     BTSerial.print("the current temperature is ");  
19     BTSerial.print(temp_result);  
20     BTSerial.println("degree celsius");  
21 }
```

DHT.read11(dht11_pin)	DHT.temperature
更新 DHT.temperature 的數值	在 read11()執行後會更新溫度數值。

4. 開啟 Serial monitor 便可讀取到溫度的數值。



IV. 溫習問題

完成本課後你能夠回答以下的問題嗎？

1. 什麼是藍牙？
2. 你能夠列舉藍牙裝置的規範，並提出規範所設立的特徵嗎？

章節三：無線通訊 – APP Inventor

本次專題的任務是利用 APP Inventor 透過連接 HC-05 藍牙模組來控制 Arduino 上的伺服馬達轉動。同學需要留意，這個任務共涉及三個部分。第一，APP Inventor 的介面和程式設置。第二，使用 HC-05 藍牙模組連接電話並與 APP Inventor 進行溝通。第三，使用 APP Inventor 控制伺服馬達的轉動。

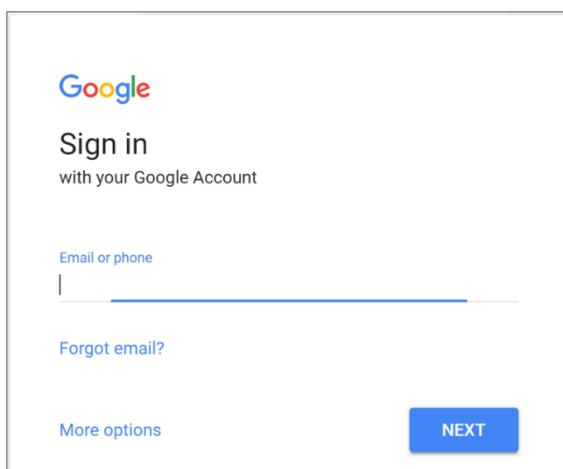
I. APP Inventor 的介面和程式設置

1. 在瀏覽器上輸入網址 <http://appinventor.mit.edu/explore> 進入 APP Inventor 網站。
2. 在 APP Inventor 網頁的右上方選擇 “Create apps!” 來製作新的手機應用程式。



The screenshot shows the MIT App Inventor website homepage. At the top, there is a navigation bar with the MIT App Inventor logo, "About", "News & Events", "Resources", and a "Create apps!" button. Below the navigation bar is a search bar with "Google Custom S" and a magnifying glass icon. The main content area features a video player with the title "MassTLC Distinguished Leadership Award 2017 - MIT App Inve..." and a play button. Below the video player, there is a statistics bar with the following data: Active Users: This Month: 637.6K, This Week: 279.3K, Today: 61.1K; Registered Users: 6.8M, Countries: 195, Apps Built: 24.0M. At the bottom of the statistics bar, it says "App Inventor code is open source". To the right of the video player, there are two promotional boxes. The first box is purple and titled "Thanks for helping us launch iOS!" with text thanking donors and mentioning a crowdfunding site. The second box is orange and titled "Introducing App Building Guides!" with text about a second App Building Guide developed by MIT App Inventor and Youth Radio, and a link to a Translation App Guide.

3. 使用 APP Inventor 前需要使用 Google 帳號登入。



4. 登入後會進入 APP Inventor 「我的專案」頁面。



5. 在我的專案頁面的左上方，按下新增專案，然後為新的專案更改名字。專案名稱可以改為“Bluetooth_Servo”，不過同學也可以改成其他名字。請留意，專案名稱必須以字母開頭。



6. 設定名字後按確定，一個新的 Bluetooth_Servo 專案便會新增在我的專案裡。打開 Bluetooth_Servo 專案後便會進入工作面板，上面顯示手機應用程式的顯示介面。



7. 同學可以在右邊元件屬性中選擇自己喜歡的設定，將自己的應用程式個人化。為方便起見，本次示範只會在「水平對齊」和「垂直對齊」選項中選擇「置中」，其他部分同學可以自行設定。



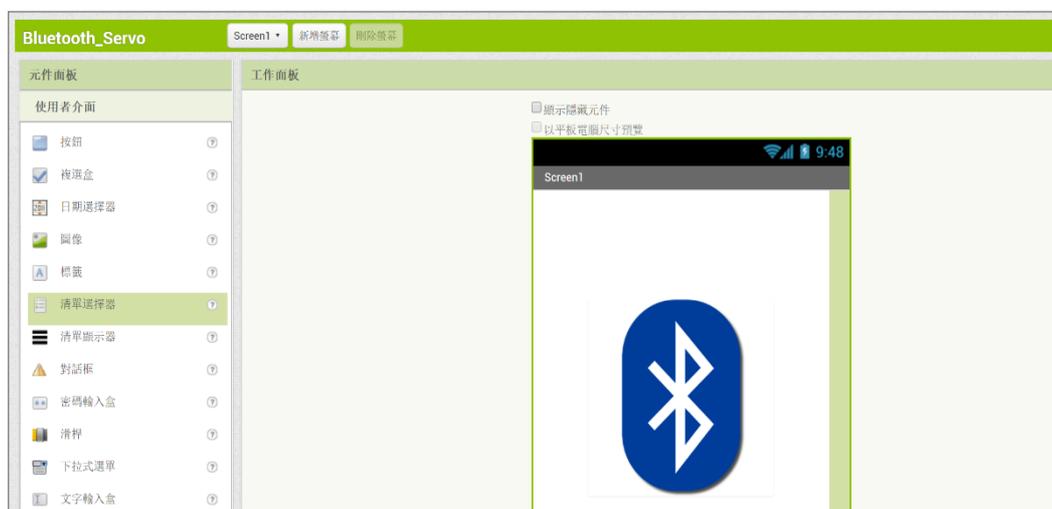
8. 從左邊元件面板選擇使用者介面，然後按下「清單選擇器」，把清單選擇器拖拉到中間工作面板的區域內。清單選擇器會出現在手機預覽屏幕的中間，而右邊元件清單也會多了清單選擇器 1。



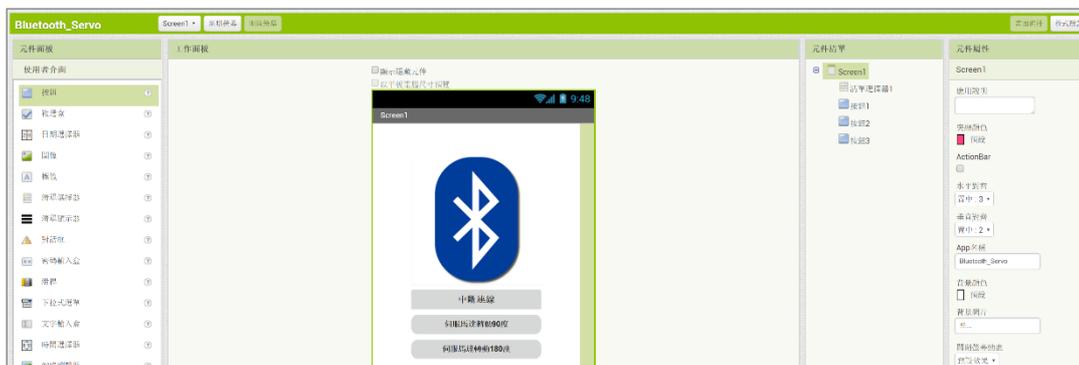
9. 我們可以為清單選擇器 1 加上圖片，讓使用者更加容易明白應用程式的各個部份。要加上圖片，我們可以在右邊的元件屬性中選擇圖像一欄，並按下上傳文件選擇自己想要的圖片。



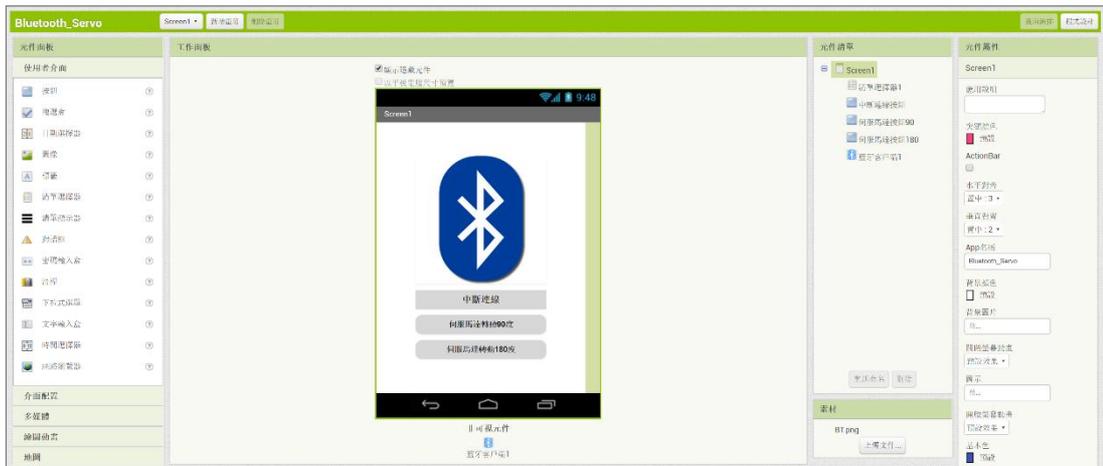
10. 加入相片後可以調較相片的大小和加入文字。



11. 接著，在左邊元件面板中的使用者介面內選擇「按鈕」，並把按鈕拉到工作面板的中間。我們可以在右邊元件屬性中為按鈕調整大小和輸入文字。我們一共要製作三個按鈕，分別稱為中斷連線按鈕、伺服馬達轉動 90 度按鈕和伺服馬達按鈕轉動 180 度按鈕。請留意，中斷連線按鈕的可見性我們預設為「否」。



12. 完成設定按鈕後，我們需要為應用程式加入藍牙設定。我們在左邊元件面板上選擇「通訊」，再選擇「藍牙客戶端」，並把藍牙客戶端拉到工作面板的中間，我們會看見藍牙客戶端出現在預覽屏幕下方，即非可視元件當中。

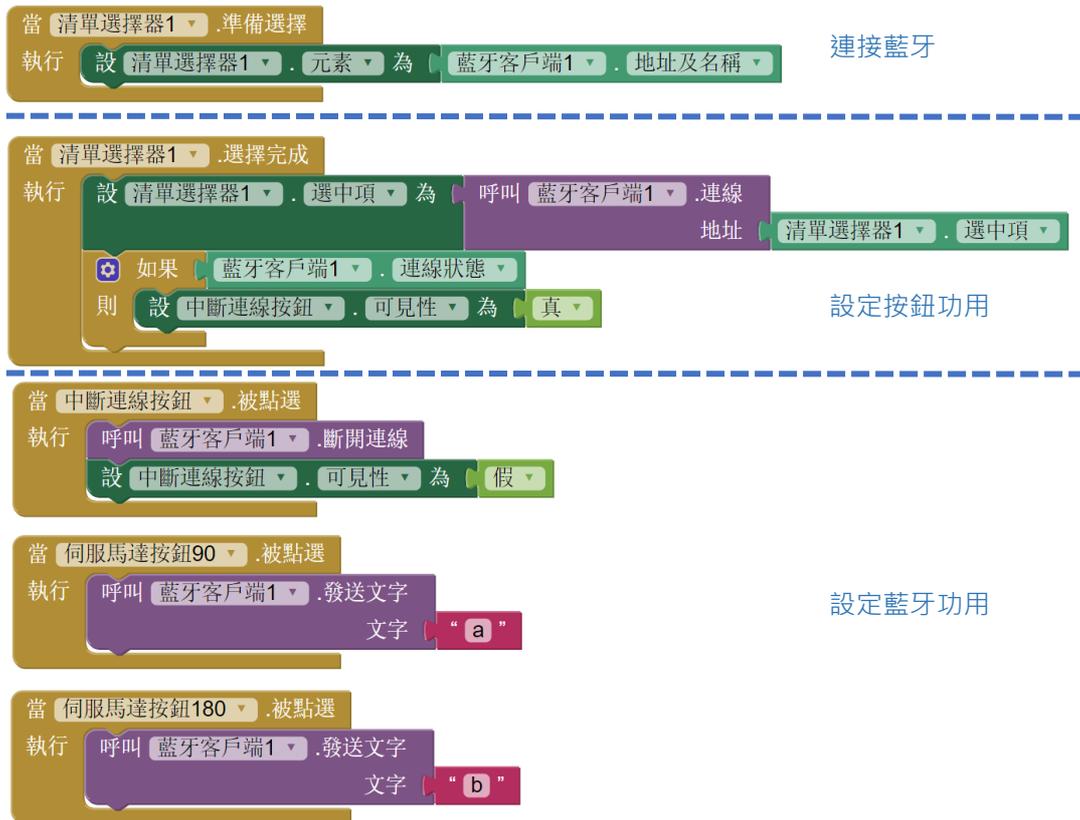


13. 完成設定使用者介面後，我們需要為應用程式編程。按下右上方「程式設計」按鈕便會跳至程式編寫



14. 從左邊方塊區域中我們可以拖拉程式方塊到工作面板上進行編程。根據程式方塊的類別和顏色，我們製作以下程式。





連接藍牙

設定按鈕功用

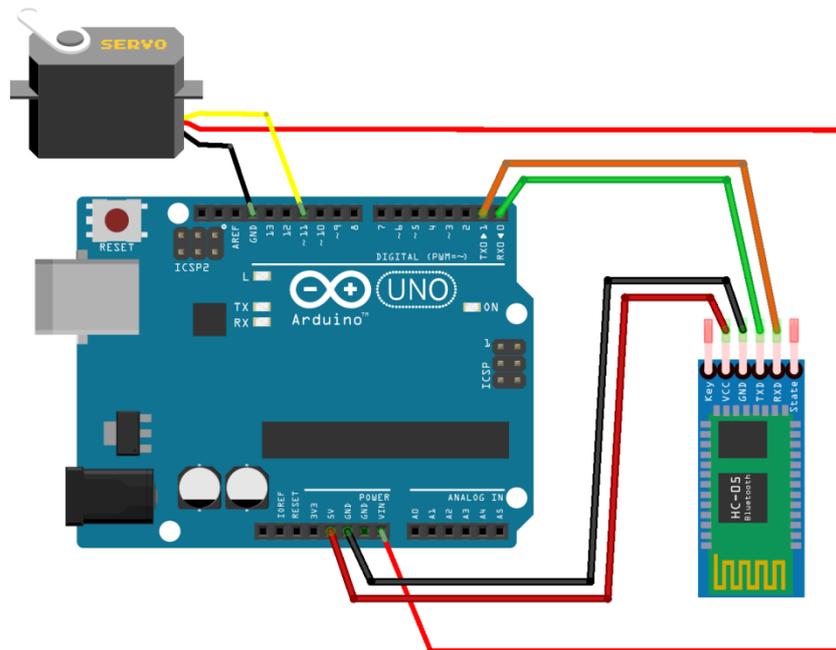
設定藍牙功用

程式共分為三部分，分別是設定藍牙、連接藍牙和設定藍牙功用。首先，在設定藍牙部分，我們在清單選擇器定義藍牙需要連接的地址和名稱。當我們在應用程式選擇需要連接的藍牙地址後，程式便會透過藍牙客戶端連接指定的藍牙。若藍牙成功連線，我們便可以透過按下伺服馬達轉動 90 度按鈕和伺服馬達轉動 180 度按鈕傳送信息到 Arduino。在以上程式中，當我們按下伺服馬達轉動 90 度按鈕，電話便會透過藍牙向 Arduino 傳遞一個英文字母 “a”；當我們按下伺服馬達轉動 180 度按鈕，電話便會透過藍牙向 Arduino 傳遞一個英文字母 “b”。

完成所有操作後，我們可以按下中斷連線按鈕來中斷藍牙連接。

II. Arduino 連接 HC-05 藍牙模組和伺服馬達

首先，根據以下方法連接 Arduino、HC-05 藍牙模組和伺服馬達。



HC-05 藍牙模組和伺服馬達連接 Arduino 的方法非常簡單。藍牙模組只需要連接四個腳位，分別是 VCC 腳位連接到 Arduino 的 5V、GND 腳位連接到 Arduino 的 GND、TXD 腳位連接到 Arduino 的 RXD 及 RXD 腳位連接到 Arduino 的 TXD。而伺服馬達方面，紅色和黑色的電源線分別連接到 Arduino 的 5V 和 GND，黃色或白色的訊號線則連接到 Arduino 的 11 號腳位。

III. 使用 APP Inventor 控制伺服馬達的轉動

首先，我們需要在 Arduino IDE 上輸入以下程式，然後把程式上載到 Arduino UNO 板上。

藍牙控制伺服馬達的程式碼全圖，可參考檔案 3_3_bluetooth_servo_motor

```
1 #include <Servo.h>
2                                     加入程式庫·定義
3 int servoPin = 11;
4                                     變數
5 Servo Servo1;
6 -----
7 void setup() {
8     Serial.begin(9600);
9     Servo1.attach(servoPin);
10 }
11 -----
12 void loop(){
13     Servo1.write(0);
14     if (Serial.available() > 0){
15         char data = Serial.read();
16         if (data == 'a'){
17             Servo1.write(0);
18             delay(2000);
19             Servo1.write(90);
20             delay(2000);
21             Servo1.write(0);
22         }
23         else if (data == 'b'){
24             Servo1.write(0);
25             delay(2000);
26             Servo1.write(180);
27             delay(2000);
28             Servo1.write(0);
29         }
30     }
31 }
```

程式內容說明

1. 加入程式庫，定義變數。

```
1 #include <Servo.h>
2
3 int servoPin = 11;
4
5 Servo Servo1;
```

在第一部分中，我們加入了伺服馬達程式庫，並定義一個名叫 `servoPin` 的變數。在以上程式中，我們首先加入之前在程式庫管理員裡下載名叫 `Servo` 的程式庫，

之後把數字 11 儲存在 servoPin 這個變數裡。完成後，我們使用 “Servo Servo1” 來啟動以上加入的伺服馬達程式庫。

2. 設定序列埠連接速率和定義伺服馬達腳位。

```
7 void setup() {  
8   Serial.begin(9600);  
9   Servo1.attach(servoPin);  
10 }
```

在第二部分中，我們設定序列埠連接速率為 9600。由於這個任務需要 Arduino UNO 和藍牙模組 HC-05 進行通訊，所以兩個模組之間需要有相同的序列埠連接速率才可以互相連接和溝通。接著，我們透過 “Servo1.attach” 來定義伺服馬達連接 Arduino 的訊號腳位位置。在第一部分中，訊號腳位定義在 ArduinoUNO 板上的 11 號腳位。

3. 根據藍牙接收的訊號控制伺服馬達。

```
12 void loop(){  
13   Servo1.write(0);  
14   if (Serial.available() > 0){  
15     char data = Serial.read();  
16     if (data == 'a'){  
17       Servo1.write(0);  
18       delay(2000);  
19       Servo1.write(90);  
20       delay(2000);  
21       Servo1.write(0);  
22     }  
23     else if (data == 'b'){  
24       Servo1.write(0);  
25       delay(2000);  
26       Servo1.write(180);  
27       delay(2000);  
28       Servo1.write(0);  
29     }  
30   }  
31 }
```

第三部分是整個 Arduino 程式的關鍵，因為程式會根據從藍牙模組接收的訊號來控制伺服馬達的轉動。在 APP Inventor 裡，我們已經設定當伺服馬達轉動 90 度按鈕被按下時，電話會透過藍牙傳送一個英文字母 “a”；當伺服馬達轉動 180 度按鈕被按下時，電話會透過藍牙傳送一個英文字母 “b”。因此，在 Arduino 程式裡我們透過分析藍牙模組接收的英文字母是 “a” 還是 “b” 來判斷伺服馬達應該轉 90 度還是 180 度。程式中，我們以 “Servo1.write” 來控制馬達的轉動角度。一開始我們重設馬達的角度，確保每一次馬達轉動前都是同一個地方開始。接著，我們會使用 “Serial.available” 檢查藍牙是否已經連接裝置。如果成功連接，我們利用 “Serial.read()” 來閱讀藍牙模組接收的信息。當接收的信息是英文字母 “a”，我們使

用 “Servo1.write” 來控制伺服馬達轉動 90 度。若接收的信息是英文字母 “b”，我們便使用 “Servo1.write” 來控制伺服馬達轉動 180 度。完成 Arduino 程式後，我們便可以開啟電話藍牙，使用電話 APP Inventor 的應用程式連接 HC-05 藍牙模組。連接後我們便可以使用 APP Inventor 控制伺服馬達的轉動。

V. 溫習問題

完成本課後你能夠回答以下的問題嗎？

1. 什麼是 App Inventor ？
2. 藍牙是屬於那一種通訊方式？
3. 你能夠設定 Arduino 與 App Inventor 進行互動嗎？

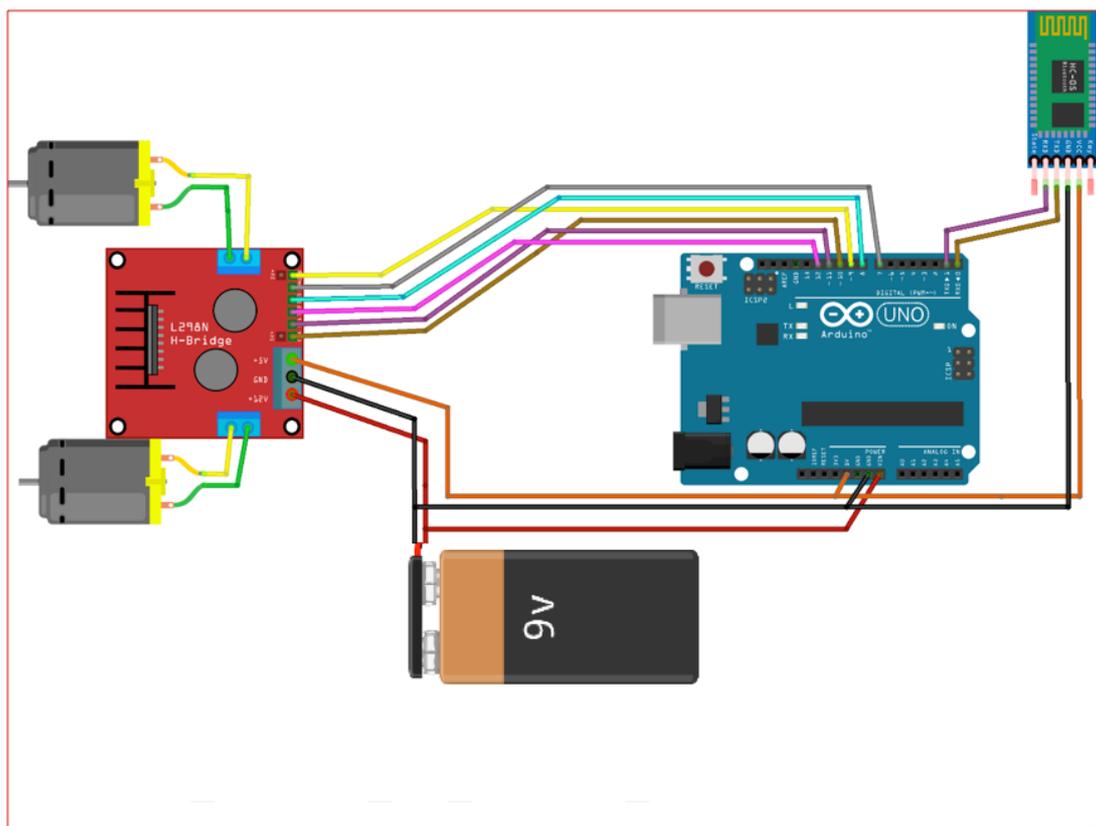
章節四：單元專題研習 – 拯救機械人

工程設計階段/步驟	相關知識	探究和設計考慮	應用相關知識和探究結果
界定問題 (確認要求和限制)	瞭解災難現場的情況十分危險，如果要人手去拯救，拯救人員不但會有生命危險，而且速度也不夠快。因此需要機器人的幫助		釐定明確的設計要求和限制 (參考資料一的例子)
研究	<ul style="list-style-type: none"> 不同無線的效能及傳輸距離 拯救時需完成哪些任務 甚麼傳感器能完成尋找生命的線索 甚麼傳感器能完成拯救任務 		選擇處理方法 (如在課堂上進行，可選一項或多項，視乎教學目標、課時及材料限制而定)
構想設計	研究資料顯示，藍牙傳輸在短距離運作效能最高	預測：自動機械人能增加拯救效率	先使用簡單的程式測試感應器及機械人的運作
測試模型	利用兩隻機械人分別去尋找生命跡象及拯救生命。兩台機械人透過藍牙溝通，加快拯救行動	<p>如何有效加快拯救生命的行動</p> <ul style="list-style-type: none"> 公平測試方法：找出機械人行走最快的程序 獨立變數：不同的程序 因變數：時間的長短 控制變數：場地、馬達功率、電池電壓等 	
解決設計/製作 測試過程中遇到的問題		程式必須讓行動機械人能成功發送有效資訊給拯救機械人，讓拯救機械人能順利拯救人質	將這些發現納入為是項工程設計的必要程序

分析及評估 測試結果及 出現的問題		利用傳感器及程序，讓機械人能貼近黑線行走，並評估設計的速度是否達標	參考分析及評估所得，研究改良方法
改良	除了安裝單個紅外線感應器外，亦可安裝多個不同感應器，幫助機械人快速並有效行動	利用電腦視覺及人工智能以改善機械人效能，包括能有效跟隨黑線行走	一起探討現今科技是如何制作拯救用的機械人

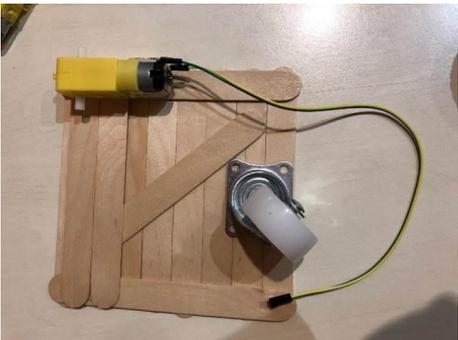
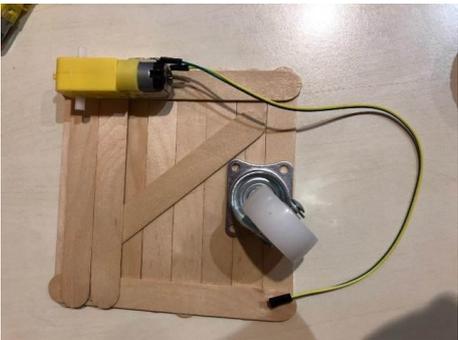
一 機械人電路製作

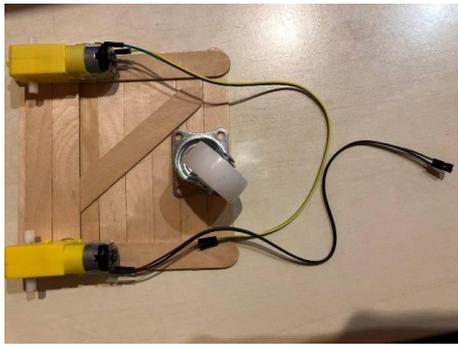
我們根據以下方法連接藍牙、馬達、馬達控制模組、9V 電池和 Arduino。完成後我們將把電子零件安裝在清潔機械人的車底上。同學請注意，因為我們要推動馬達，因此會建議使用較高電壓來推動馬達。由於 Arduino Uno 有穩壓模組，所以使用 9V 的電池直接接上 Arduino 便可推動控制板。



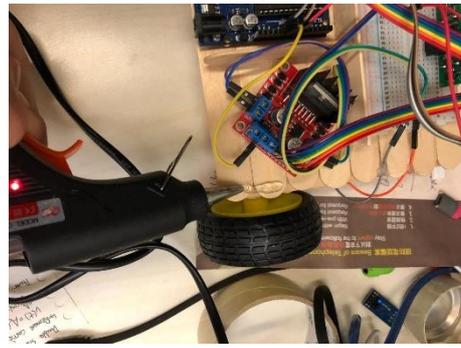
二 拯救機械人車身結構製作

同學可根據自己的構思，為你的拯救機械人設計及製作其外殼，用以放置 Arduino 板及其他電子原件。以下是製作拯救機械人車底的參考：

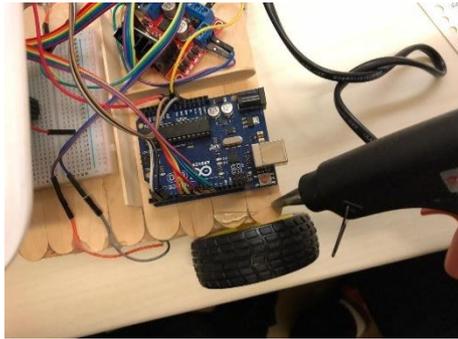
<p>1. 拿出 8 條雪條棍，並且排列在一起</p> 	<p>2. 使用熱溶膠槍塗滿熱溶膠在雪條棍上</p> 
<p>3. 塗滿後的雪條棍</p> 	<p>4. 把它壓在整排雪條棍，並固定雪條棍</p> 
<p>5. 最後加上斜的雪條棍作固定</p> 	<p>6. 使用熱溶膠槍把萬向輪貼在雪條棍上</p> 
<p>7. 在後方貼上 2 條雪條棍加高位置</p> 	<p>8. 把黃色馬達貼在機械人的後方</p> 
<p>9. 重覆以上動作</p>	<p>10. 在機械人左邊馬達旁的木條塗熱溶膠</p> 



11. 在機械人右邊馬達旁的木條塗熱溶膠



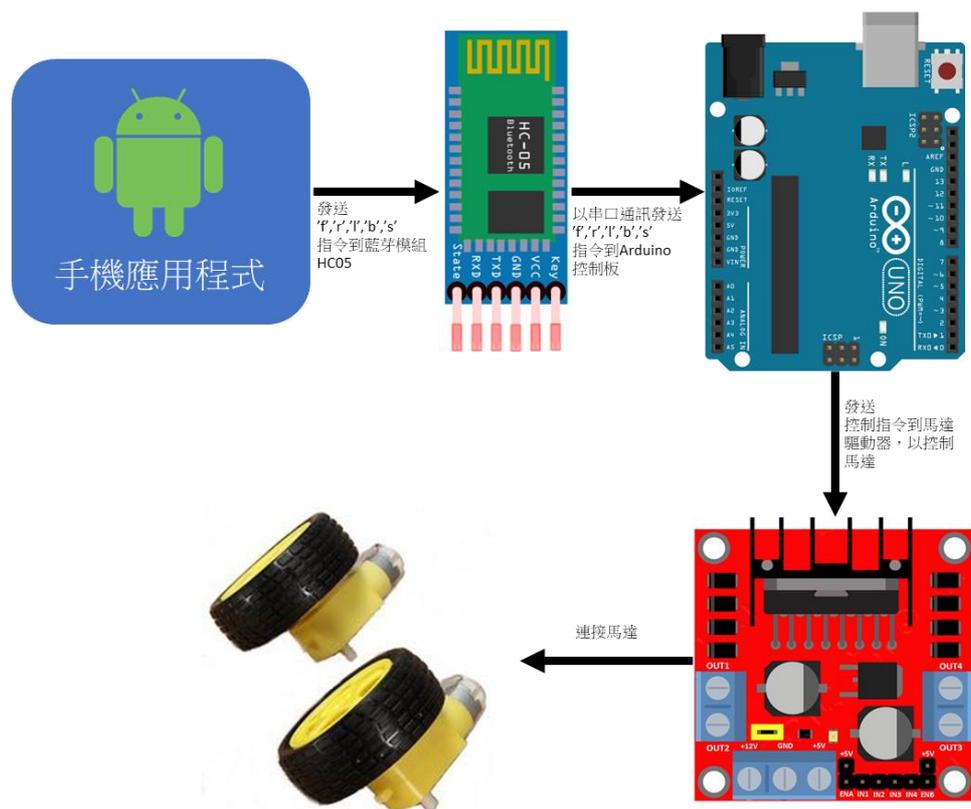
12. 先把塑膠蓋剪開並塗熱溶膠



13. 最後替機械人加上後蓋



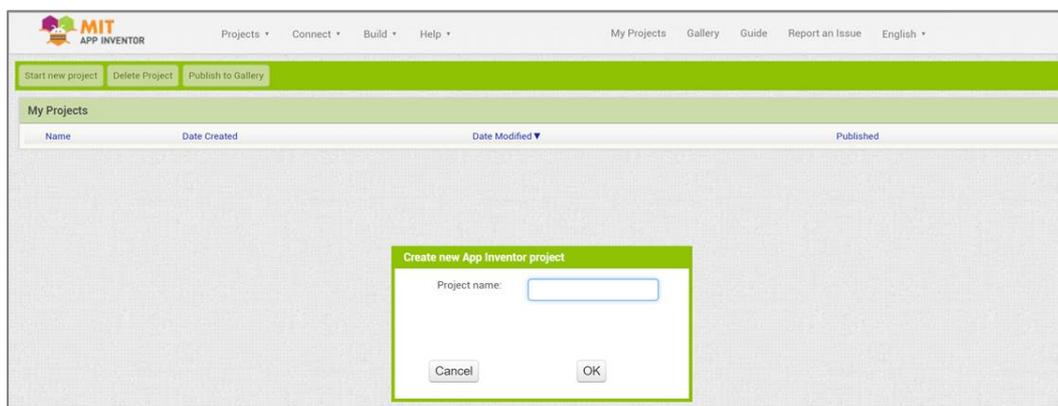
三 程式編寫



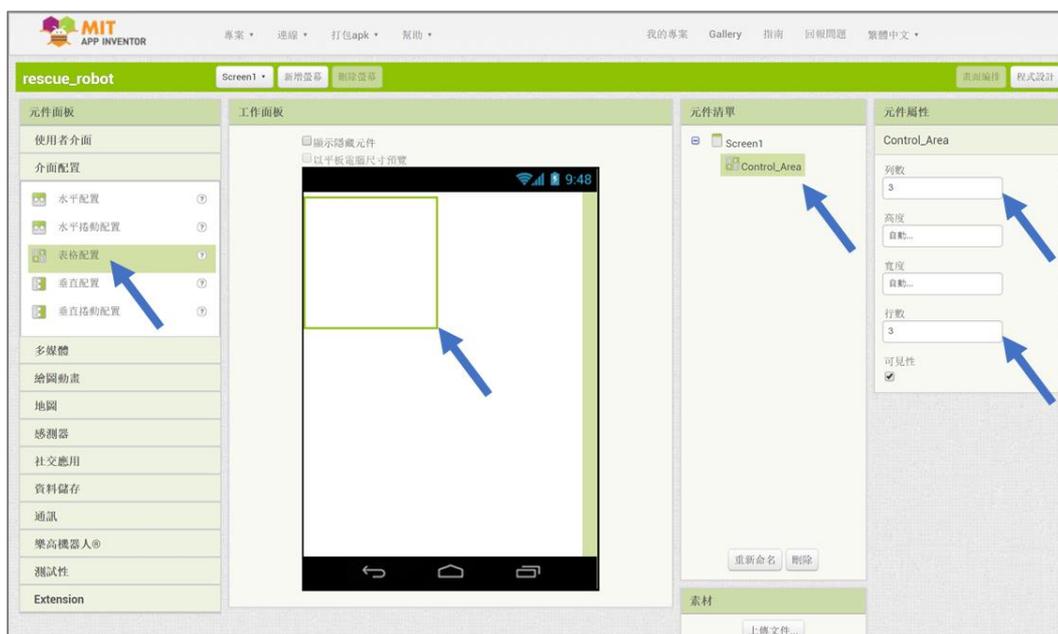
上述例子中，我們以“f”來代表 Forward 的指令、“b”來代表 Backward 的指令、“r”來代表 Right 的指令、“l”來代表 Left 的指令及“s”來代表 Stop 的指令。Arduino 控制板從藍牙模組 HC05 上取得指令後再發送控制訊號到馬達驅動器，驅動器接收到訊號後便會作出反應，驅動馬達旋轉。

設計好專案的流程後，便可以開始著手設計及編寫安卓手機應用程式的介面和邏輯。以下是編寫安卓應用程式的步驟。跟上一個動手作差不多，我們要先設計介面，後編寫邏輯運算。

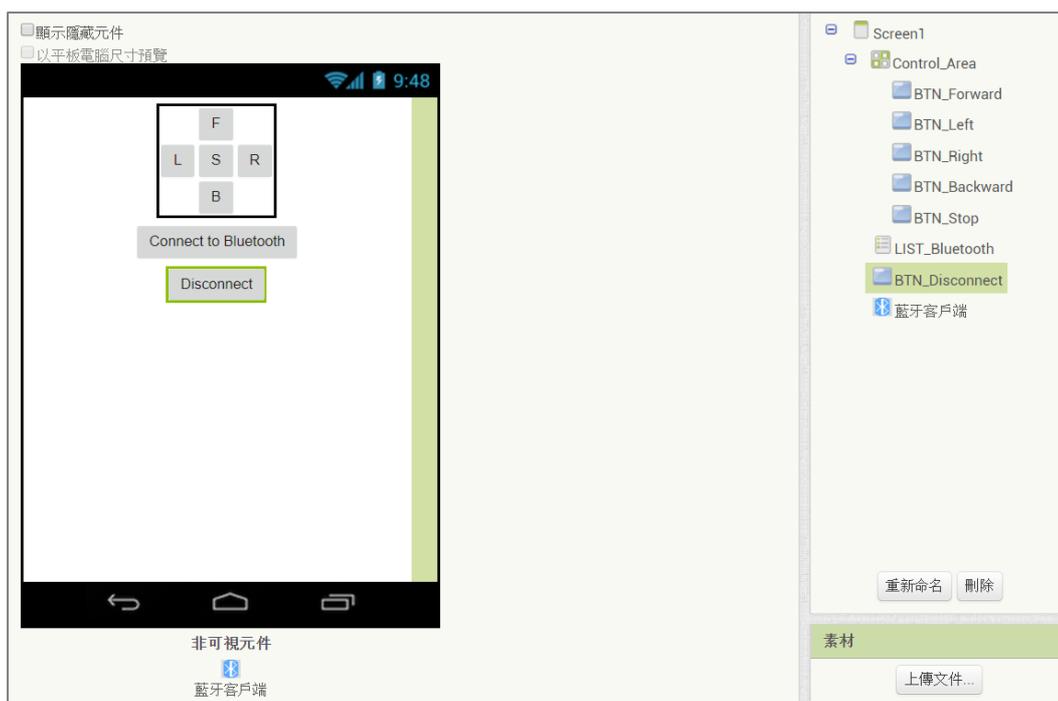
1. 先開啟一個新的專案，並為這個專案命名。



2. 打開介面後，先從元件面板下的介面配置中拖出表格配置，這個表格是為方向鍵作定位之用。我們有前、後、左、右與停五個按鈕，因此我們需要在表格上輸入三格列數及三格行數，以把這些按鈕都放下。在元件清單上會看到表格配置，為了讓同學更易理解及開發得更快，我們通常都會建議把元件名稱更改。



3. 定義好介面配置後，便可以拉進更多元件去完成應用程式。同學們可從元件面板上拖拉五個按鈕到元件清單上，並排列出前、後、左、右與停止。跟一般藍牙的應用程式一樣，我們需要一個清單選擇器用以顯示藍牙搜尋到的設備、一個藍牙客戶端的非可視元件及一個中斷的按鈕。



4. 按鈕的屬性例如按鈕的大小、文字等都可以在元件屬性上找到。如要更改按鈕上的文字，我們要從元件屬性中找到文字的一欄，並在文字輸入方塊上輸入你想要的名稱。為了讓程式變得更美，同學們可以在線 Screen1 的屬性中的水平對齊選取置中，這樣可以令元件在面板都放在中間。中斷連線的按鈕只在連線成功的時候才會顯示，因此同學們可以先在元件屬性上的可見性別除，待在邏輯編程時才打開可見性。程式介面可算完成。



5. 程式邏輯主要可分為兩小部分，首先要處理藍牙連線部分的程式方塊。

```

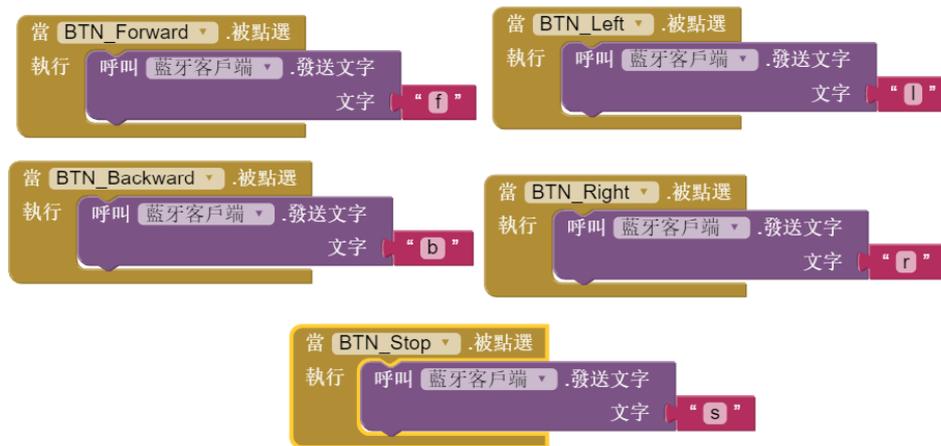
當 LIST_Bluetooth .準備選擇
執行 設 LIST_Bluetooth .元素 為 藍牙客戶端 .地址及名稱

當 LIST_Bluetooth .選擇完成
執行 設 LIST_Bluetooth .選中項 為 呼叫 藍牙客戶端 .連線
    地址 LIST_Bluetooth .選中項
    如果 藍牙客戶端 .連線狀態
    則 設 BTN_Disconnect .可見性 為 真

當 BTN_Disconnect .被點選
執行 呼叫 藍牙客戶端 .斷開連線
    設 BTN_Disconnect .可見性 為 假
  
```

這三段程式碼能讓手機與藍牙模組溝通。第一組程式方塊是指當用家按下清單按鈕時，把清單內的每個元素都變成藍牙接收端的地址及名稱。第二組是指用家在選擇完成時，會讓藍牙連接他所選擇的地址及名稱。此外，程式亦將中斷連線的按鈕變為可見，讓用家能在適當的時候把手機與藍牙模組中斷連接。

6. 這五個按鈕性質一樣，因此它們的程式幾乎都是一樣的。當按鈕被點下的時候，會發送一個文字到藍牙模組。



7. 手機應用程式部分便完成了，接下來便是 Arduino 控制板的程式部分。

拯救機械人的程式碼全圖，可參考檔案 3_4_rescue_robot

```
1 const int motorENA = 10;
2 const int motorIN1 = 11;
3 const int motorIN2 = 12;
4
5 const int motorENB = 9;
6 const int motorIN3 = 8;
7 const int motorIN4 = 7;
8
9 byte incomingByte = 0;
10 -----
11 void setup() {
12   Serial.begin(9600);
13   pinMode(motorENA, OUTPUT);
14   pinMode(motorIN1, OUTPUT);
15   pinMode(motorIN2, OUTPUT);
16   pinMode(motorENB, OUTPUT);
17   pinMode(motorIN3, OUTPUT);
18   pinMode(motorIN4, OUTPUT);
19   digitalWrite(motorENA, HIGH);
20   digitalWrite(motorENB, HIGH);
21 }
22 -----
```

定義腳位變數

定義腳位運作方式及
開啟串行通訊頻道

```

23 void loop() {
24   if(Serial.available()) {
25     incomingByte = Serial.read();
26   }
27   -----
28   if(incomingByte == 's') {
29     digitalWrite(motorIN1, LOW);
30     digitalWrite(motorIN2, LOW);
31     digitalWrite(motorIN3, LOW);
32     digitalWrite(motorIN4, LOW);
33   }
34   else if(incomingByte == 'f') {
35     digitalWrite(motorIN1, HIGH);
36     digitalWrite(motorIN2, LOW);
37     digitalWrite(motorIN3, HIGH);
38     digitalWrite(motorIN4, LOW);
39   }
40   else if(incomingByte == 'b') {
41     digitalWrite(motorIN1, LOW);
42     digitalWrite(motorIN2, HIGH);
43     digitalWrite(motorIN3, LOW);
44     digitalWrite(motorIN4, HIGH);
45   }
46   else if(incomingByte == 'r') {
47     digitalWrite(motorIN1, HIGH);
48     digitalWrite(motorIN2, LOW);
49     digitalWrite(motorIN3, LOW);
50     digitalWrite(motorIN4, HIGH);
51   }
52   else if(incomingByte == 'l') {
53     digitalWrite(motorIN1, LOW);
54     digitalWrite(motorIN2, HIGH);
55     digitalWrite(motorIN3, HIGH);
56     digitalWrite(motorIN4, LOW);
57   }
58 }

```

每次執行時從手機更新指令

接受指令後執行動作

8. 這段正是為了程式的常數而定義的程式。這段程式把接上了的腳位定義成文字，讓撰寫程式的時候能更易使用及更易明白。程式中選取了第 9 及第 10 的腳位作為連接馬達驅動器的 ENA 及 ENB 腳位，這是因為這兩個腳位能多輸出模擬訊號。此外，亦定義了一個名為 incomingByte 的變數，用以放下從藍牙模組收回來的訊號，以在之後的程式中能夠作出決定。

```

1 const int motorENA = 10;
2 const int motorIN1 = 11;
3 const int motorIN2 = 12;
4
5 const int motorENB = 9;
6 const int motorIN3 = 8;
7 const int motorIN4 = 7;
8
9 byte incomingByte = 0;

```

9. 這段程式只會執行一次，用以設定串行通訊的傳輸速度、腳位用途及設定。因為我們要發送訊息到馬達驅動器，因此必須把腳位的模式設定為輸出模式，以傳送控制訊號。最後的兩個 `digitalWrite()` 是給馬達驅動器指示兩個馬達都是使用的，因此會看見 `motorENB`, `motorENA` 的腳位裏長期輸出高電壓。

```
11 void setup() {
12   Serial.begin(9600);
13   pinMode(motorENA, OUTPUT);
14   pinMode(motorIN1, OUTPUT);
15   pinMode(motorIN2, OUTPUT);
16   pinMode(motorENB, OUTPUT);
17   pinMode(motorIN3, OUTPUT);
18   pinMode(motorIN4, OUTPUT);
19   digitalWrite(motorENA, HIGH);
20   digitalWrite(motorENB, HIGH);
21 }
```

10. 這段程式是放在 `loop()` 的函數裡，令它不停運行。這是一個簡單的如果句構邏輯，意思是指當串行通訊有訊息傳入時，程式便要讀取串行通訊的訊息，並把訊息存入 `incomingByte` 的變數中，然後等待進一步的動作。

```
23 void loop() {
24   if(Serial.available()) {
25     incomingByte = Serial.read();
26   }
```

11. 接下來的程式看似複雜，但細心留意程式其實有很多重覆的成份，程式中的每一個指令都是跟據以下的表格填寫。例如我們希望機械人停下來時，同學們應該要對 `IN1`、`IN2`、`IN3`、`IN4` 這四個腳位輸出低電壓以作出停止的指令。

	motorIN1	motorIN2	motorIN3	motorIN4
Forward (f)	HIGH	LOW	HIGH	LOW
Right (r)	HIGH	LOW	LOW	HIGH
Back (b)	LOW	HIGH	LOW	HIGH
Left (l)	LOW	HIGH	HIGH	LOW
Stop (s)	LOW	LOW	LOW	LOW

```

28  if(incomingByte == 's') {
29      digitalWrite(motorIN1, LOW);
30      digitalWrite(motorIN2, LOW);
31      digitalWrite(motorIN3, LOW);
32      digitalWrite(motorIN4, LOW);
33  }
34  else if(incomingByte == 'f') {
35      digitalWrite(motorIN1, HIGH);
36      digitalWrite(motorIN2, LOW);
37      digitalWrite(motorIN3, HIGH);
38      digitalWrite(motorIN4, LOW);
39  }
40  else if(incomingByte == 'b') {
41      digitalWrite(motorIN1, LOW);
42      digitalWrite(motorIN2, HIGH);
43      digitalWrite(motorIN3, LOW);
44      digitalWrite(motorIN4, HIGH);
45  }
46  else if(incomingByte == 'r') {
47      digitalWrite(motorIN1, HIGH);
48      digitalWrite(motorIN2, LOW);
49      digitalWrite(motorIN3, LOW);
50      digitalWrite(motorIN4, HIGH);
51  }
52  else if(incomingByte == 'l') {
53      digitalWrite(motorIN1, LOW);
54      digitalWrite(motorIN2, HIGH);
55      digitalWrite(motorIN3, HIGH);
56      digitalWrite(motorIN4, LOW);
57  }
58  }

```

12. 程式部分也完成了，最後就是把硬件連接上