**目的**

GPS Tracker用以記錄使用者的移動軌跡，設計構想為使用者身上的手錶，或其他穿戴式裝置，在使用者外出時可以記錄使用者的GPS位置至裝置內存記憶體中，待使用者休息時或移動到室內有網路處(Wifi/3G/4G)，此時可長按裝置上之按鈕，進行位置軌跡上傳至RainbowTree平台的動作。

**材料**

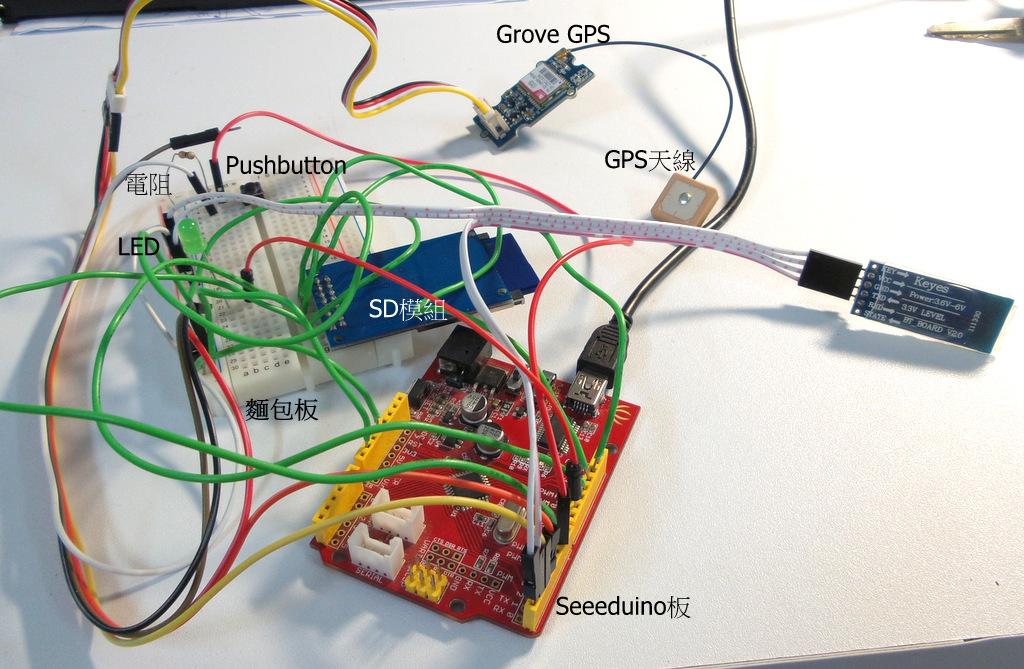
|  |  |
| --- | --- |
| Arduino主板 Seeeduino | × 1 |
| Grove GPS | × 1 |
| Android手機 | × 1 |
| SD模組 | × 1 |
| 藍芽模組 HC-06 | × 1 |
| 杜邦線、單心線 | × N |
| 麵包板 | × 1 |
| PushButton | × 1 |
| LED | × 1 |

**接線**

由於此專案外接三組模組，攝影機模組、SD模組、及藍芽模組，我們需要將Arduino板上之5V及GND拉線至麵包板上，再以並聯的方式來連結各個模組的VCC及GND針腳。

|  |  |
| --- | --- |
| **Grove GPS --- Arduino端**  GND---GND  5V---5V  TX---D2  RV---D3  此GPS模組連接介面為UART，我們使用SoftwareSerial設定GPS的TX、RX接腳為D2、及D3，另外設定GPS模組的baud rate為9600 | 20140828_163134.jpg20140828_163120.jpg |
| **SD模組---Arduino端**  +5---5V  CS---D4  MOSI---D11  SCK---D13  MISO---D12  GND---GND  SD模組的連接介面為SPI，此Arduino對應的SPI如上所列。 | 20140721_122305.jpg |
| **HC-06藍芽模組 --- Arduino端**  VCC---5V  GND---GND  TXD---D0  RXD---D1  \*HC-06的TX需接至Arduino主板上的RX(D0)  HC-06的RX需接至Arduino主板上的TX(D1)  \*由於需傳送檔案，我們事先透過AT command將baud rate設定為57600。 | 20140710_121952.jpg |

\*以上提供我們的接法，詳細可根據個人的專案設計來決定針腳的接法，但VCC/GND務必不要接反；另外SD模組接頭有時可能會接觸不良，造成檔案寫入錯誤，因此後來我們改用單心線來做連結，提供參考。



\*燒錄程式請務必將各模組電源切斷，或是燒錄完畢後，再斷電重新將線路連通，以防模組等燒壞。藍芽模組接至Arduino板上的RX、TX，在燒錄程式時會影響，故請不要接上線路，待燒錄完畢，再將各模組接上通電執行。

**Arduino端程式操作**

這個專案的開發，主要的工作是解析GPS接收到的資料格式，我們採用了TinyGPS++這個libaray，這個library主要解析GGA及RMC兩種格式的資料，由GGA內取得位置、時間、海拔高度資料，由RMC內取得日期資料，使用者自行設定log資料的頻率，若想要將資料上傳，只要長按pushbutton，LED亮起時表示此時傳送GPS log檔至Android手機端，手機端收完檔案後讀取檔案，再將log的資料上傳至RainbowTree平台。若需要重啟GPS tracker功能，需按Seeeduino主板上Reset鍵重新log資料。

1. 由於GPS及藍芽模組皆為UART連接方式，我們在這邊透過SoftwareSerial設定GPS照相機模組的RX、TX接腳，藍芽模組就用Seeeduino板上的RX(D0)、TX(D1)連接。

|  |
| --- |
| SoftwareSerial GPSserial(2, 3); //rx,tx |

1. 設定Serial port(BlueTooth)及GPS照相機模組的baud rate及初始化各模組。

SD模組為SPI連接，程式預設CS接腳為D4，若使用Ethernet Shield，請記得將D10 output設為HIGH。我們透過AT command將HC-06模組的baud rate改為57600(HC-06出廠預設baud rate為9600，藍芽設備名稱為HC-06，配對密碼為1234，若想要修改baud rate，請參考HC-06 AT command的修改方式)。若想使用LED請另外指定pin腳，因pin13被SD模組所使用。

|  |
| --- |
| //==================================setup()====================================  void setup()  {  Serial.begin(57600); //Serial baud rate (bluetooth)  GPSserial.begin(9600); //GPS baud rate  pinMode(buttonPin, INPUT); //Set buttonPin as INPUT  pinMode(ledPin, OUTPUT); //Set ledPin as OUTPUT  // set initial LED state  digitalWrite(ledPin, LOW);  if (!SD.begin(4)) { //SD module initialization, CS=4  return;  }  myFile = SD.open("tracker.txt", FILE\_WRITE); //open GPS log file for writing...  if(myFile) //check if file exists  {  closedflag=1;  }  else  return;  } |

1. 重複執行Loop內的動作

讀取Pushbutton的值，並進行bounce的判斷。

* 1. 若pushbutton長按超過longPressTime，則進入藍芽傳輸模式，此時LED會亮起，待檔案傳送完畢後LED燈會熄滅。
  2. 接收GPS的資料，進行parsing的動作，GPS取得的資料格式為NMEA，此parsing library會判定GGA及RMC格式的資料以取得位置、時間、海拔高度、日期資料。

|  |
| --- |
| void loop()  {  long tmp;  int reading = digitalRead(buttonPin);  if (reading != lastButtonState) {  lastDebounceTime = millis();  }  if ((millis() - lastDebounceTime) > debounceDelay) { //check Debounce delay  if(reading)  {  tmp= millis()-currentTime;  if(tmp>=longPressTime) //if pushbutton is hold at "pressed" state more than longPressTime,  { //then...  digitalWrite(ledPin, HIGH); //LED light  if(closedflag=1)  {  myFile.close(); //close file  closedflag=0;  }  SendToBT(); //transfer file through Bluetooth  digitalWrite(ledPin, LOW); //after done transfer, LED OFF  return; //the program returns here, reset arduino to restart  }  }else  currentTime=millis();    // if the button state has changed: HIGH->LOW or LOW->HIGH  if (reading != buttonState) {  buttonState = reading;  }  }    lastButtonState = reading;  while (GPSserial.available() > 0) //Reading GPS data  {  s=GPSserial.read();  if (gps.encode(s)) //TinyGPS++ parsing data  displayInfo();  }  if (millis() > 5000 && gps.charsProcessed() < 10)  {  while(true);  }  } |

\*Arduino透過藍芽傳送檔案一樣是走Serial Port Profile(SPP)，因此我們需要重新設計我們所傳送的訊息格式。每次傳送資料時，會先傳送一個byte的OP code(MSG\_CMD、FILE\_SEND\_CMD)，接著的一個byte代表所要傳送的payload大小，再來才是傳送的資料。因此如下所示，前三行程式碼是用來傳送訊息(MSG\_CMD)，我們將所欲傳送的檔案名稱及檔案大小寫入藍芽Serial，接著後面三行是用來傳送檔案內容。

|  |
| --- |
| Serial.write(MSG\_CMD); //Message command  Serial.write(s.length()); //s: message string,  Serial.print(s);      Serial.write(FILE\_SEND\_CMD); //File content command  Serial.write(PIC\_PKT\_LEN); // PIC\_PKT\_LEN bytes are sent each time  Serial.write(wrbuffer, PIC\_PKT\_LEN); |

**Android手機端程式實作與操作** [**<創建自己的RainbowTree專案請按此>**](http://rainbowtree.here-apps.com/)

\*下載RainbowTree SDK，將其將入專案中libs目錄下

\*我們是改寫sample裡的BluetoothSPPDeviceList，這個範例提供完整的Bluetooth scan、配對、連線等流程。另外我們亦修改了BluetoothSPP library內的BluetoothService.java檔，修改了當Android端接收到藍芽訊息時的處理方式，當每次接收資料息時，收到的第一個byte會是control\_byte，用來判定目前收到的資料息為一般的訊息還是檔案內容，第二個byte為此次傳送資料的長度，接下來的bytes就是資料內容。

**BluetoothService.java**

|  |
| --- |
| public void run() {  //...  // Keep listening to the InputStream while connected  while (true) {  try {  int data = mmInStream.read();  if(section==0){  control\_byte=data;  section=1;  }else if(section==1){  payload\_count=data;  section=2;  }else{//section==2  arr\_byte.add(data);    if(arr\_byte.size()>=payload\_count){//done  buffer = new byte[arr\_byte.size()];  for(int i = 0 ; i < arr\_byte.size() ; i++) {  buffer[i] = arr\_byte.get(i).byteValue();  }  // Send the obtained bytes to the UI Activity  mHandler.obtainMessage(BluetoothState.MESSAGE\_READ  , buffer.length, control\_byte, buffer).sendToTarget();  //reset  //...  }  }  } catch (IOException e) {  //...  }  }  } |

**AndroidManifest.xml**

加入Internet與Bluetooth的uses-permission，另外由於需接收檔案，我們會將檔案片存至SD卡內，故亦需外部記憶空間的存取權限。

|  |
| --- |
| <uses-permission android:name=**"android.permission.INTERNET"** />  <uses-permission android:name=**"android.permission.BLUETOOTH"** />  <uses-permission android:name=**"android.permission.BLUETOOTH\_ADMIN"** />  <uses-permission android:name=**"android.permission.WRITE\_EXTERNAL\_STORAGE"** /> |

**Main.java**

import rainbowtree.android.PostExecuteListener及rainbowtree.android.RainbowtreeClient，宣告一個RainbowTree的instance，並加入RainbowTree平台的IP位址及API\_key。

宣告 HashMap map=new HashMap(); **，**用以存放所欲上傳平台的datastream

|  |
| --- |
| import com.hereapps.rainbowtree.android.PostExecuteListener;  import com.hereapps.rainbowtree.android.RainbowtreeClient;  **RainbowtreeClient client=new RainbowtreeClient("106.186.30.234","e095f4538bac4870");**  HashMap map=new HashMap(); |

當收到資料時，首先會先去解析是哪一種control\_byte，由於此App僅限用於接收GPS檔案並上傳RainbowTree，故我們將MSG\_mode設定為傳送檔案名稱及大小，當App端接收到包含檔案資訊的String並進行解析後，此時會打開新檔。再來Arduino端會開始傳送檔案內容，App端接收內容後會一併寫入檔案內。

|  |
| --- |
| if(control\_byte==0) //msg rx mode  {  String[] sensordata= message.split(",");  if(sensordata.length==2){  filename = sensordata[0];  filesize = Integer.parseInt(sensordata[1]);  //open file    File file = new File(fpath +"/"+ filename);  fpath=fpath+"/"+ filename;    cnt=filesize;  try{  fos = new FileOutputStream(file); //open file  }  catch (FileNotFoundException e) {  }    bt.send("GO"); //arduino starts transferring file when receive bt serial  }else ;  }else if(control\_byte==1) //file rx mode  {  if(cnt!=0)  {  try{ //one byte per writing  fos.write(data,0,data.length);  cnt=cnt-data.length;  }  catch (IOException ioe) {  }  if(cnt<=0)  {  try {  if (fos != null) {  fos.close();  }  }  catch (IOException ioe) {  }    String inputLine;  try {  BufferedReader br = new BufferedReader(new FileReader(fpath));  List<HashMap<String, Object>> valueMapList=new ArrayList<HashMap<String, Object>>();  List<Date> atList=new ArrayList<Date>();    while ((inputLine = br.readLine()) != null)  {  String[] gpsdata= inputLine.split(";");  if(gpsdata.length==3){  **map.put("sf4d36a802e8d11e4", gpsdata[0]);**  **map.put("tfa82a5902e8d11e4", gpsdata[1]);**  **map.put("vedcd5202e8d11e4", gpsdata[2]);**    valueMapList.add(map);  atList.add(new Date());    **client.postMultiDataAsync("b5f38c29a2e6472", atList, valueMapList, new PostExecuteListener(){**  @Override  public void onPostExecute(int arg0, String arg1) {  Log.d("debug", "statusCode="+arg0);  Log.d("debug", "resultMessage="+arg1);  }});  }  }  Toast.makeText(getApplicationContext(), "Done", Toast.LENGTH\_LONG).show();  Statusvalu1.setText("Done Uploading");  } catch (FileNotFoundException e) {  }  catch (IOException e) {  }  }  }  }else ; |

設定Location、Time、及Altitude的datastream id為**sf4d36a802e8d11e4，tfa82a5902e8d11e4，vedcd5202e8d11e4，**並傳入對應的資料

|  |
| --- |
| try {  BufferedReader br = new BufferedReader(new FileReader(fpath));  List<HashMap<String, Object>> valueMapList=new ArrayList<HashMap<String, Object>>();  List<Date> atList=new ArrayList<Date>();    while ((inputLine = br.readLine()) != null) //逐行讀取檔案內容  {  String[] gpsdata= inputLine.split(";"); //資料以“;”隔開  if(gpsdata.length==3){  **map.put("sf4d36a802e8d11e4", gpsdata[0]);** //將位置資料及其對應datastream id加入map  **map.put("tfa82a5902e8d11e4", gpsdata[1]);** //將時間資料及其對應datastream id加入map  **map.put("vedcd5202e8d11e4", gpsdata[2]);**  //將高度資料及其對應datastream id加入map    valueMapList.add(map); //加入以上三筆資料到valueMapList  atList.add(new Date()); //加入欲上傳的時間至atList內    //上傳多筆資料可用以下function--postMultiDataAsync，**"b5f38c29a2e6472"**”為Device id  //若僅上傳一筆資料，可用postDataAsync function  **client.postMultiDataAsync("b5f38c29a2e6472", atList, valueMapList, new PostExecuteListener(){**  @Override  public void onPostExecute(int arg0, String arg1) {  Log.d("debug", "statusCode="+arg0);  Log.d("debug", "resultMessage="+arg1);  }});  }  } |

**參考資料**

**Arduino**

[**http://arduino.cc/**](http://arduino.cc/en/Reference/SoftwareSerial)

**TinyGPS++**

[**http://arduiniana.org/libraries/tinygpsplus/**](http://arduiniana.org/libraries/tinygpsplus/)

**Grove GPS**

[**http://www.seeedstudio.com/wiki/Grove\_-\_GPS**](http://www.seeedstudio.com/wiki/Grove_-_GPS)

**SoftwareSerial**

[**http://arduino.cc/en/Reference/SoftwareSerial**](http://arduino.cc/en/Reference/SoftwareSerial)

**Arduino SPI**

[**http://arduino.cc/en/Reference/SPI**](http://arduino.cc/en/Reference/SPI)

**SD模組**

[**http://www.ltc.com.tw/print\_product\_info.php/products\_id/60605**](http://www.ltc.com.tw/print_product_info.php/products_id/60605)

**藍芽模組HC-06**

[**http://gsyan888.blogspot.tw/2014/03/arduino-hc-06-at-command.html**](http://gsyan888.blogspot.tw/2014/03/arduino-hc-06-at-command.html)

[**http://www.mikrokopter.de/ucwiki/en/HC-06**](http://www.mikrokopter.de/ucwiki/en/HC-06)

**Android-Bluetooth SPP Library**

[**https://github.com/akexorcist/Android-BluetoothSPPLibrary**](https://github.com/akexorcist/Android-BluetoothSPPLibrary)