

# 實驗三 UART Send/Receive

## 實驗目的

- 了解如何使用 uart 端，收送封包

## 實驗步驟

### Step 1:

- components `SerialActiveMessageC` 宣告及拉線，通常在 configuration 內的 implementation
- 先宣告下列 component，而在這邊用 `as Serial` 只是利用 `Serial` 這簡短的字來替換原先 component 名稱，`components SerialActiveMessageC as Serial;`

### Step 2:

- 而在範例 `BaseStation`, module 為 `BaseStationP` 在 configuration 內，將 component 內的物件，拉線到已存在 component `SerialActiveMessageC`，表示 module 內 `BaseStationP` 有自訂 interface `UartSend` 與在 `SerialActiveMessageC` 中 interface 產生連結，當然你可以只選擇你想用的 interface 來連結，以下範例式連結了四個 interface

```
BaseStationP.UartSend -> Serial;
```

```
BaseStationP.UartReceive -> Serial;
```

```
BaseStationP.UartPacket -> Serial;
```

```
BaseStationP.UartAMPacket -> Serial;
```

### Step 3:

- 接著在 module `BaseStationP` 必須實作，先呼叫程式中有用到的 interface 而 `SerialActiveMessageC` 其內 provide 很多 interface，而以下為部分 interface

```
interface AMSend as UartSend[am_id_t id];
```

```
interface Receive as UartReceive[am_id_t id];
```

```
interface Packet as UartPacket; //封包格式設定，內含 interface 與下不同
```

```
interface AMPacket as UartAMPacket; //封包格式設定
```

#### Step 4:

將訊息送出 uart 端

call `UartSend.send[id]( 目的地 addr , 送出 data , sizeof(data) );`

`id` 型態為 `am_id_t` 為封包 type

目的地 `addr` => 可使用 `AM_BROADCAST_ADDR` 代表無特定目的地

送出 `data` 型態必須為 `message_t`

`sizeof(data)` 為 `data` 長度

而 `event void UartSend.sendDone[am_id_t id](message_t* msg, error_t error){}`

會回報 `uartsend` 是否成功

而 `uart` 端接收訊息較少用到，宣告完 `interface` 後，其接收到由 `uart` 端獲得資訊可用下列 `event` 獲得

`event message_t *UartReceive.receive[am_id_t id](message_t *msg, void *payload, uint8_t len){}`

`msg` 為所收資訊

`payload` 為其中內容

`len` 為 `payload` 長度

#### Step 5:

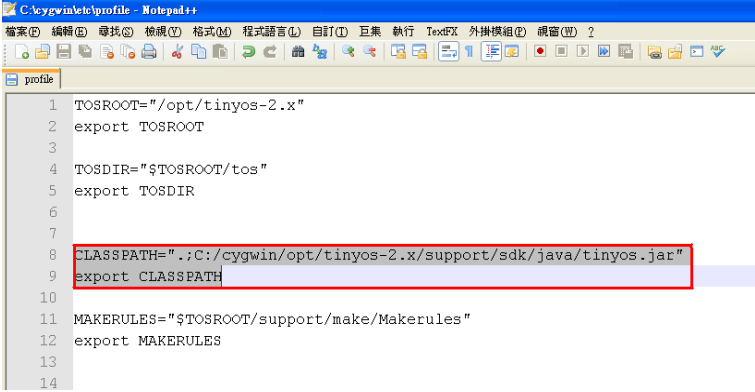
- 當你想寫的 `Uart` 收送程式寫好，且在 `sensor` 上燒好後，`Listen sensor` 送出 `Uart` 資訊必須依照下列步驟:

首先，先要確定你安裝 `cygwin` 底下 `java` 路徑是否有設對

檢查 `C:/cygwin/etc/profile` 將 `classpath` 改為下圖路徑

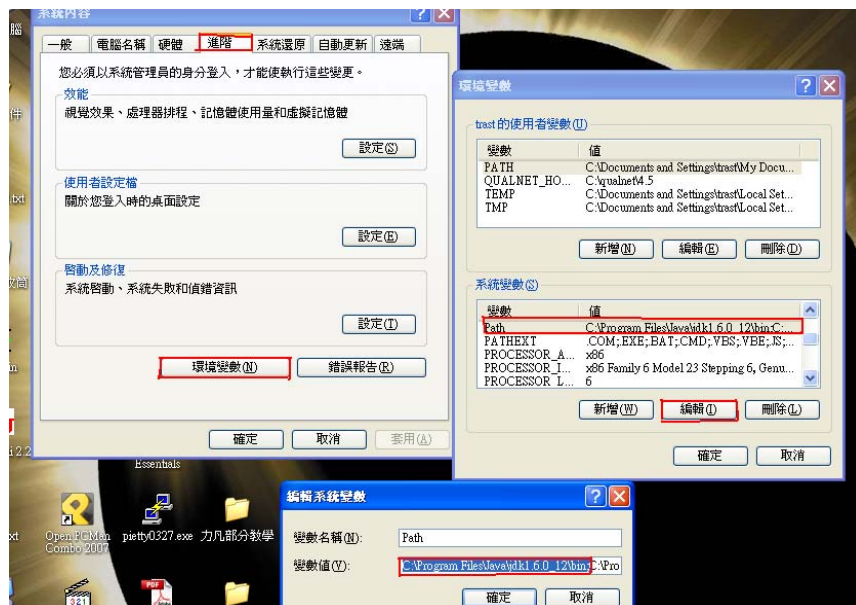
```
CLASSPATH=".;C:/cygwin/opt/tinyos-2.x/support/sdk/java/tinyos.jar"
```

```
export CLASSPATH
```

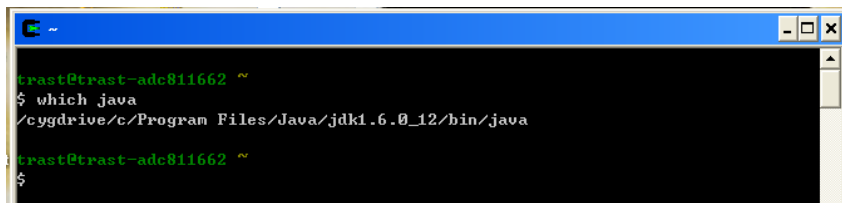


```
C:\cygwin\etc\profile - Notepad++
檔案(F) 編輯(E) 尋找(S) 檢視(V) 格式(O) 程式語言(L) 自訂(I) 巨集 執行 文本FX 外掛模組(P) 視窗(W) ?
profile
1 TOSROOT="/opt/tinyos-2.x"
2 export TOSROOT
3
4 TOSDIR="${TOSROOT}/tos"
5 export TOSDIR
6
7
8 CLASSPATH=".;C:/cygwin/opt/tinyos-2.x/support/sdk/java/tinyos.jar"
9 export CLASSPATH
10
11 MAKERULES="${TOSROOT}/support/make/Makerules"
12 export MAKERULES
13
14
```

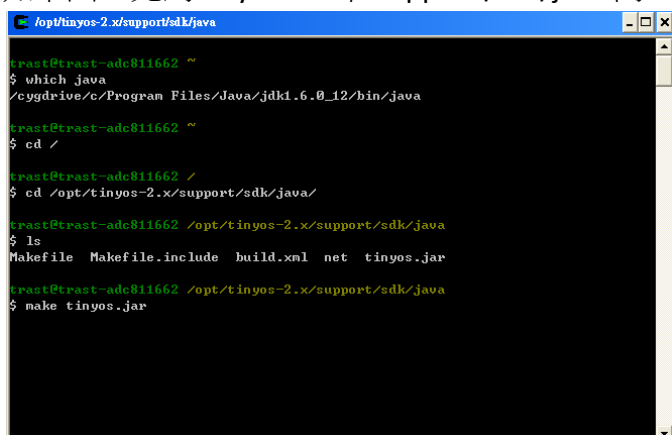
再來 右鍵點選 我的電腦=>內容=>進階=>環境變數=>系統變數內的 path => 編輯 => 在變數值的欄位 =>將游標移到最前面加上你安裝 java sdk 的 path 而在這裡我 java sdk 安裝路徑為此 C:\Program Files\Java\jdk1.6.0\_12\bin;



之後可利用在 cygwin 下輸入 which java 檢查 java 路徑是否有設對

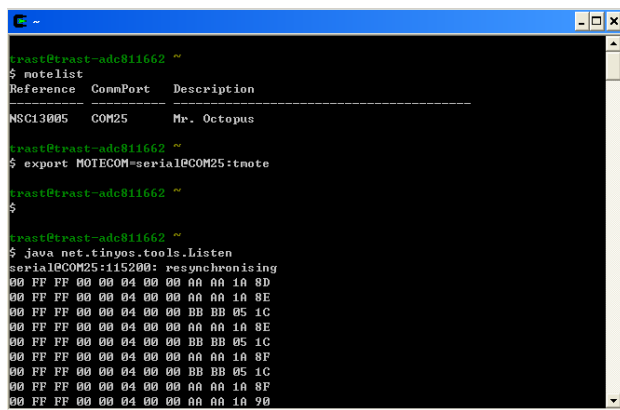


若路徑設對 要使用 java library 前，還需編譯 java library 如下圖，先到 tinyos-2.x 下 support /sdk/java 內



輸入 make tinyos.jar 若沒發生錯誤，就可再依下圖讀取封包資訊了 但若有 error 資訊產生 請檢查上述之前步驟是否有問題

下圖為要讀取 sensor 所傳到 Uart 端資訊方法:



```
trast@trast-adc811662 ~  
$ motelist  
Reference  ComnPort  Description  
-----  
NSC13005   COM25      Mr. Octopus  
  
trast@trast-adc811662 ~  
$ export MOTECOM=serial@COM25:tmote  
  
trast@trast-adc811662 ~  
$  
  
trast@trast-adc811662 ~  
$ java net.tinyos.tools.Listen  
serialCOM25:115200: resynchronize  
00 FF FF 00 00 04 00 00 0A 0A 1A 0D  
00 FF FF 00 00 04 00 00 0A 0A 1A 0E  
00 FF FF 00 00 04 00 00 0B 0B 05 1C  
00 FF FF 00 00 04 00 00 0A 0A 1A 0E  
00 FF FF 00 00 04 00 00 0B 0B 05 1C  
00 FF FF 00 00 04 00 00 0A 0A 1A 0F  
00 FF FF 00 00 04 00 00 0B 0B 05 1C  
00 FF FF 00 00 04 00 00 0A 0A 1A 0F  
00 FF FF 00 00 04 00 00 0A 0A 1A 00
```

輸入以下指令

motelist //列出本機電腦上插的 Sensor Comport

export MOTECOM=serial@COM25:tmote //開啓要 listen 的 Sensor Comport

java net.tinyos.tools.Listen // tinyos 內提供 java 的 listen library 可聽 sensor 資訊

若所燒錄程式確定有在 Uart 端有送出資訊，就可聽到封包資訊了

## 學習成果

- 可以在其他實驗中，用 uart 端收送自訂封包，藉以檢查封包結果是否如預期

## 參考資料

- Component, interface 細節 command，event 部分可在 <http://www.tinyos.net/tinyos-2.x/doc/nescdoc/telosb/> 查取範例程式 BaseStation。