

# 分散式系統

## Lab: MQTT

112753138 資碩工一 周宸宇

### Basic Information

(Type names, student IDs, and your department name below)

### 安裝 MQTT 伺服器 mosquitto

1. 若您使用 Windows 請到 <https://mosquitto.org/download/> 下傳並安裝適合的版本，系統預設安裝到 C:\program files\mosquitto\下。安裝過程中，建議不要勾選 install as service。
2. 若您使用 Mac，請先確定您有安裝 homebrew (參考 <https://brew.sh/>)  
安裝完 brew 後，請開啟終端機，輸入 brew install mosquitto
3. 如果無法安裝者，可使用公用測試伺服器 mqtt://test.mosquitto.org:1883，但可能會額外收到奇怪訊息或是連線速度較慢。

### 啟動與測試 MQTT 伺服器

1. 啟動 MQTT 伺服器

Windows: 開啟命令列視窗(cmd)，切換目錄至 C:\Program Files\mosquitto 之後，執行 mosquitto.exe (雙擊滑鼠)

Mac: 開啟命令列視窗，請參考這裡的 Step 2.:

<https://oranwind.org/post-post-2/>或依次執行以下

```
brew services list
```

```
brew link mosquitto
```

```
brew services start mosquitto -d
```

```
brew services restart mosquitto
```

2. 測試 Listener: 新增一個 mqtt listener 來訂閱「EVENT」這個 topic

Windows: 在命令列視窗(cmd) C:\Program Files\mosquitto 下執行  
mosquitto\_sub -t EVENT

Mac: 開啟終端機，執行 mosquitto\_sub -t EVENT

3. 測試 Publisher: 新增一個 mqtt message publisher，發佈訊息到 EVENT 這個 topic。

Windows: 在命令列視窗(cmd) C:\Program Files\mosquitto 下執行  
mosquitto\_pub -t EVENT -m hello。此時，mqtt listener 中應該會收到 hello 訊息。

Mac: 在終端機執行 mosquitto\_pub -t EVENT -m hello。此時，mqtt listener 中應該會收到 hello 訊息。

## 寫作 MQTT Listener

1. 建立一個名為「mqttlabb」的空專案，新增一個 package.json 檔案
2. package.json 對於 JavaScript 專案來說是專案設定檔，可以在其中載明所使用的 library，之後可透過 npm 指令自動下傳。package.json 內容如下:

```
{
  "name": "mqttlabb",
  "version": "1.0.0",
  "dependencies": {
    "mqtt": "*"
  }
}
```

3. 請使用 npm install 安裝所需要的函式庫。
4. 將 cep-template.js 更名為 cep.js，參考程式中的註解與下面說明的範例程式，完成一個簡單的 MQTT CEP Listener。

由 EVENT 中接收輸入的事件，事件共四種: DO(DOOR ON), DF(DOOR OFF), CO(CAB ON), CF(CAB OFF)。偵測連續的「CO->CF->DO->DF」的 sequence，當這四個事件先後發生時，就往 EVENT 送出一個「LEAVE」的事件。注意下列事項:

- 這四個事件只需要依次須序出現，不一定要連續出現，中間若插入任何事件時，應該予以忽略。例如:「CO->AA->CF->BB->DO->DF」在 DF 偵測到之後，也需要輸出「LEAVE」。
- 匹配完最後的 DF，並輸出「LEAVE」後，系統應該要回到從 CO 開始匹配「CO->CF->DO->DF」的 sequence。
- 提示:單一事件匹配到的時候，利用 JavaScript 的 shift 方法，將第一個元

素移除。全部匹配完成時，`eventSeq.length` 為 0 (陣列中元素都匹配完了)。

- 可使用 `mosquitto_sub/mosquito_pub` 進行測試，例如 `mosquitto_pub -t EVENT -m CO`

程式樣板請見附檔: `cep-template.js`

將您的 `cep.js` 所有程式碼貼在這裡:

```
const mqtt = require('mqtt');
const client = mqtt.connect();
client.subscribe('EVENT'); //請填入適當的值
let eventSeq = ['CO', 'CF', 'DO', 'DF'];

client.on('message', function (topic, message) {
  let item = eventSeq[0]; // 取得 eventSeq 的第一個元素
  // 除錯用
  // message 是 Buffer 型別，要用 message.toString() 才能取得字串值
  console.log('input: ' + message.toString());
  //message.toString() 是 -m 後面的訊息

  if (message.toString() !== 'LEAVE'){
    console.log('expected: ' + item);
    console.log('-----');
  }

  //if (...) ...; // 如果 message 和 item 匹配，將匹配元素從陣列移出 (使用 eventSeq.shift())
  // message 是 Buffer 型別，要用 message.toString() 才能取得字串值
  if (message.toString() == item) {
    eventSeq.shift();
    if (eventSeq.length !== 0){
      console.log('Seq ' + eventSeq);
    }
  } else {
    console.log('Seq empty');
  }
  console.log('-----');
```

```
}

//if (...) { // 如果全部匹配
    // 送出 LEAVE 到 EVENT topic 並回復 eventSeq 的內容
    if (eventSeq.length == 0) {
        client.publish('EVENT', 'LEAVE');
        eventSeq = ['CO', 'CF', 'DO', 'DF'];
    }
});
```