

JAVA網路處理

劉治廷

前言

課程目標

- 知道什麼是API
- 使用Java內建套件URLConnection呼叫API
- 了解JSON及剖析JSON
- 物件化剖析結果及ArrayList操作

範例下載處

- <https://goo.gl/VKnHgg>
- 可以使用Clone or Download中Download Zip下載
- 建議：使用git抓取
git clone
https://github.com/silencecork/shuworkshop_late2017.git

網路處理

API

API

- Application Programming Interface
- 應用程式介面
- 網路中服務和服務之間以API作為資料交換的機制
- 最常見的資料交換格式為JSON或XML
- 現在手機也都使用API與各項服務相連
 - Google Speech API, Search API
 - Facebook Graph API
 - TensorFlow API

API使用流程



API

- 連接API的方式有以下幾種
 - GET
 - e.g. GET `http://api.com/user` 取得用戶資料
 - POST
 - e.g. POST `http://api.com/user` 建立用戶資料
 - PUT
 - e.g. PUT `http://api.com/user` 更新取代用戶資料
 - DELETE
 - e.g. DELETE `http://api.com/user` 刪除用戶資料
 - PATCH
 - e.g. PATCH `http://api.com/user` 修正用戶資料

API常用方式GET

- GET將參數帶在HTTP的Request網址中
 - 網址 `http://xxx.toright.com/api/?id=010101`
 - 封包

```
GET /?id=010101 HTTP/1.1
Host: xxx.toright.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-TW;
rv:1.9.2.13) Gecko/20101203 Firefox/3.6.13 GTB7.1 ( .NET CLR 3.5.30729)
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-tw,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: UTF-8,*
Keep-Alive: 115
Connection: keep-alive
```

API常用方式POST

- POST將參數帶在HTTP Request的封包中
 - 網址 <http://xxx.toright.com/api/insert>
 - 封包

```
POST / HTTP/1.1
Host: xxx.toright.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-TW;
rv:1.9.2.13) Gecko/20101203 Firefox/3.6.13 GTB7.1 ( .NET CLR
3.5.30729)
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-tw,en-us;q=0.7,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: UTF-8,*
Keep-Alive: 115
Connection: keep-alive

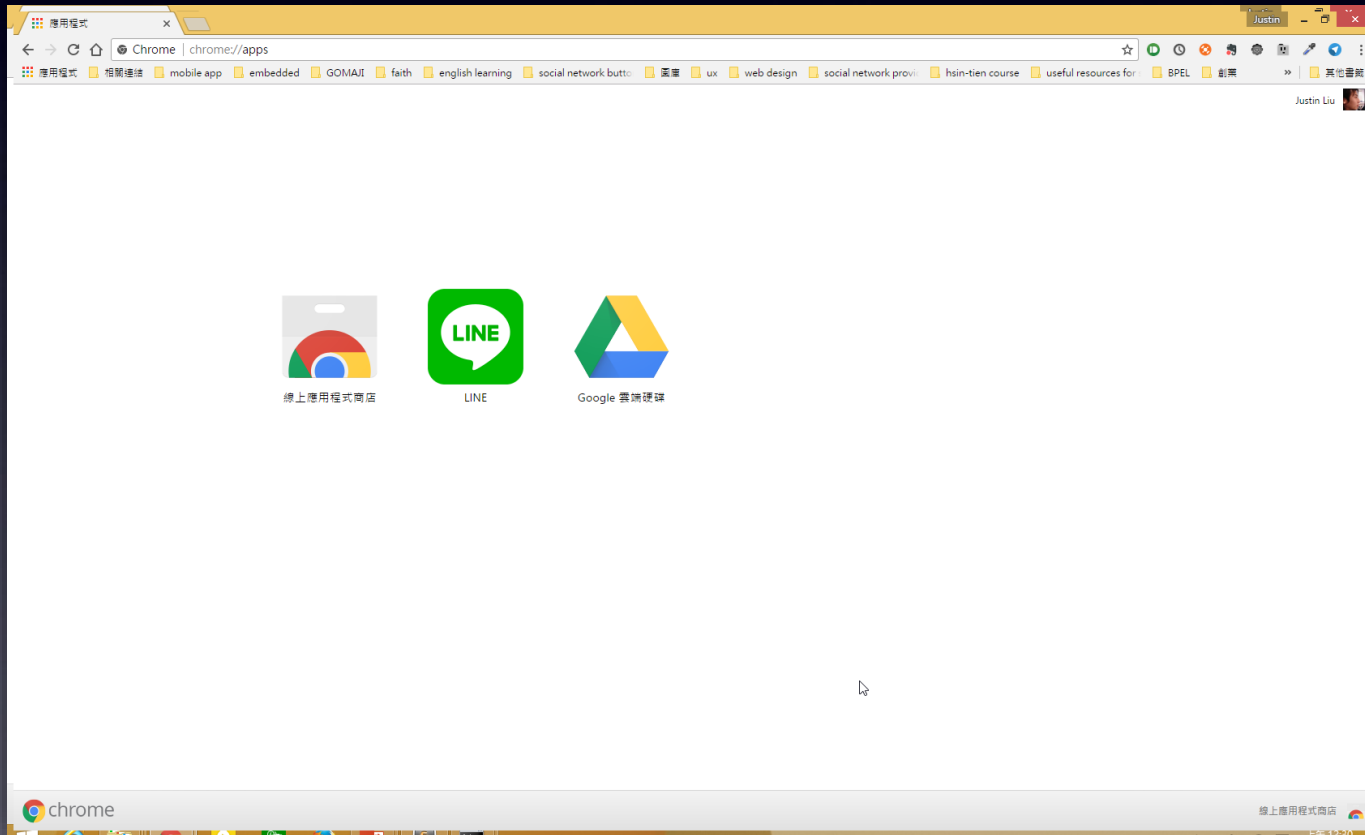
Content-Type: application/x-www-form-urlencoded
Content-Length: 9
id=010101
```

網路處理

前置作業

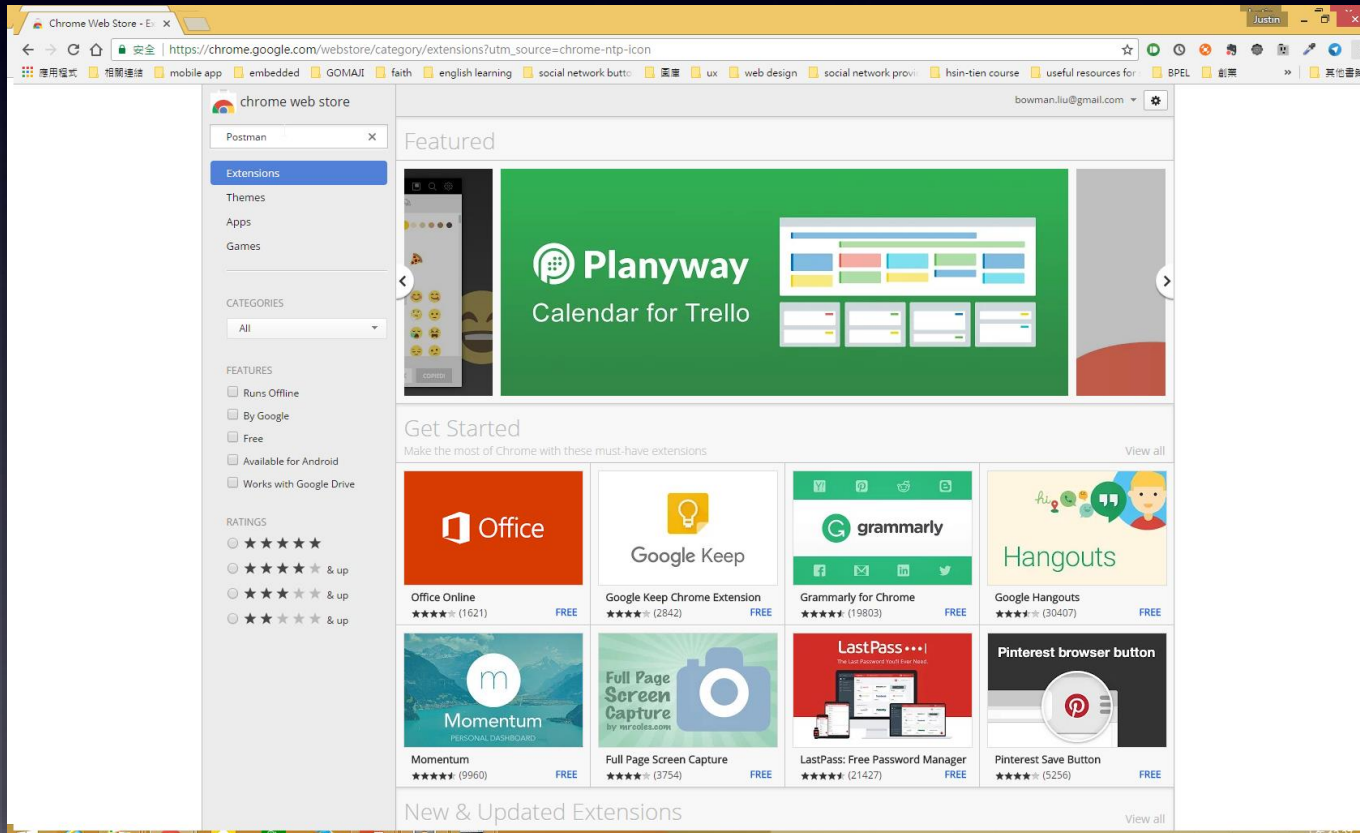
安裝POSTMAN

- 打開Chrome瀏覽器，選擇應用程式中的線上應用程式商店



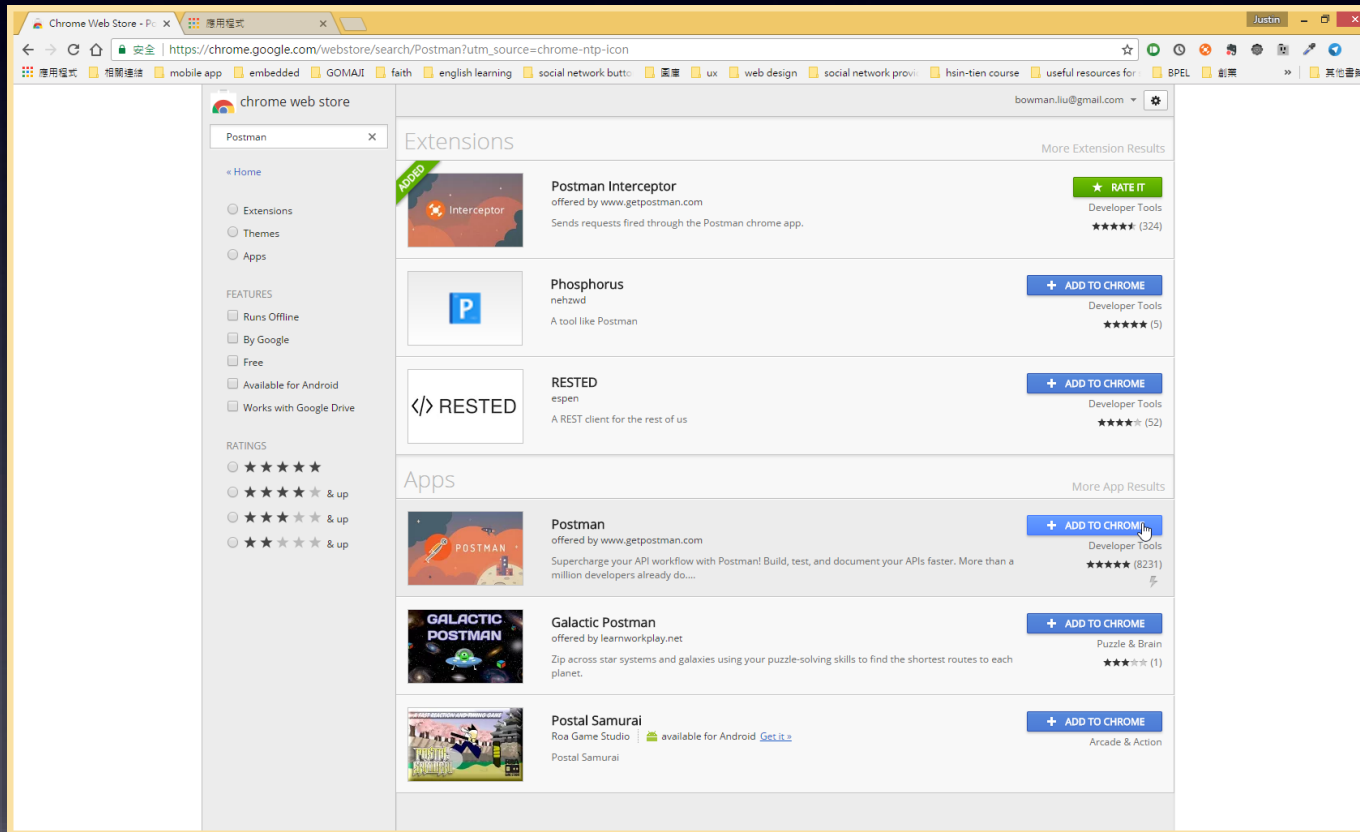
下載POSTMAN

- 在搜尋欄輸入postman並按下搜尋



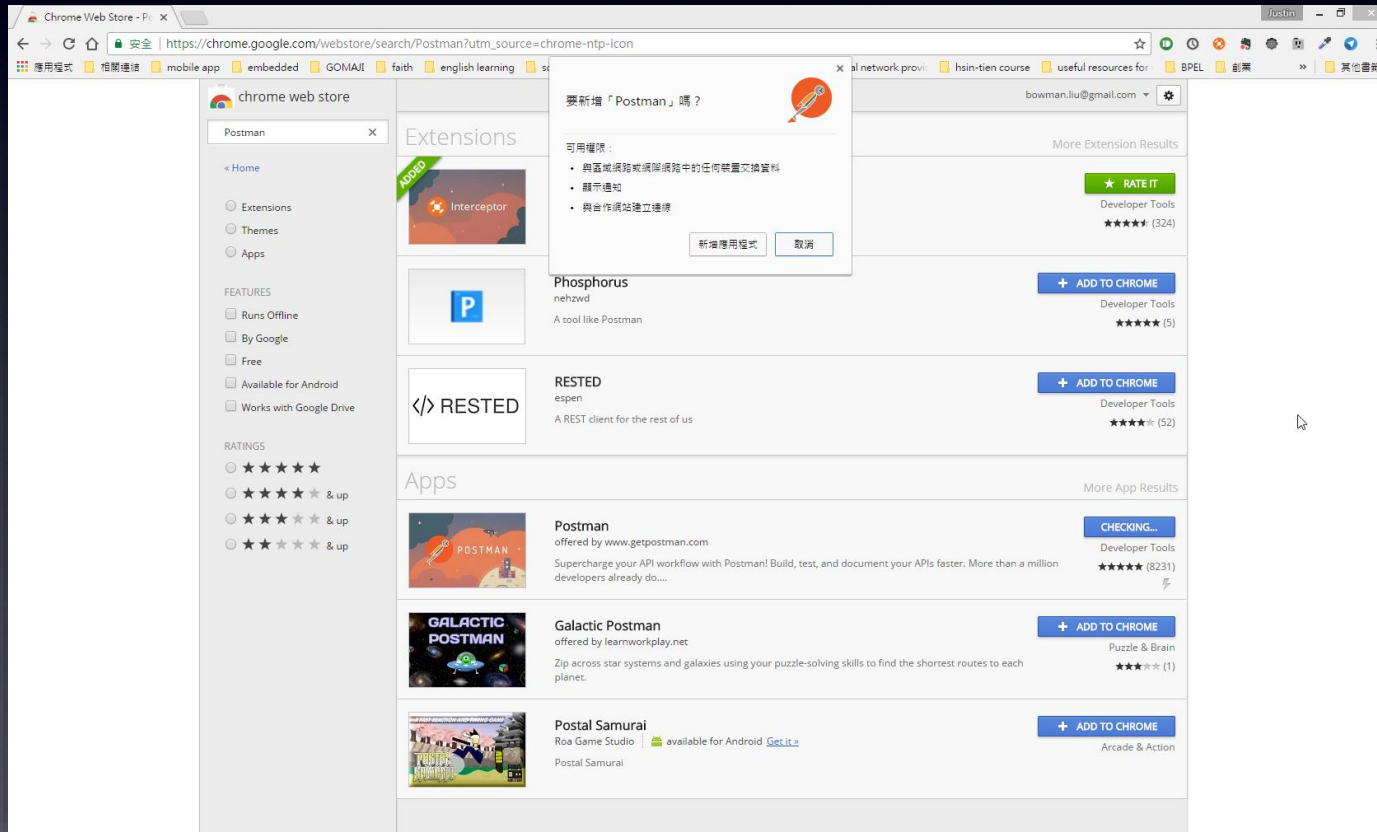
下載POSTMAN

- 找到Apps中的Postman，選擇Add To Chrome



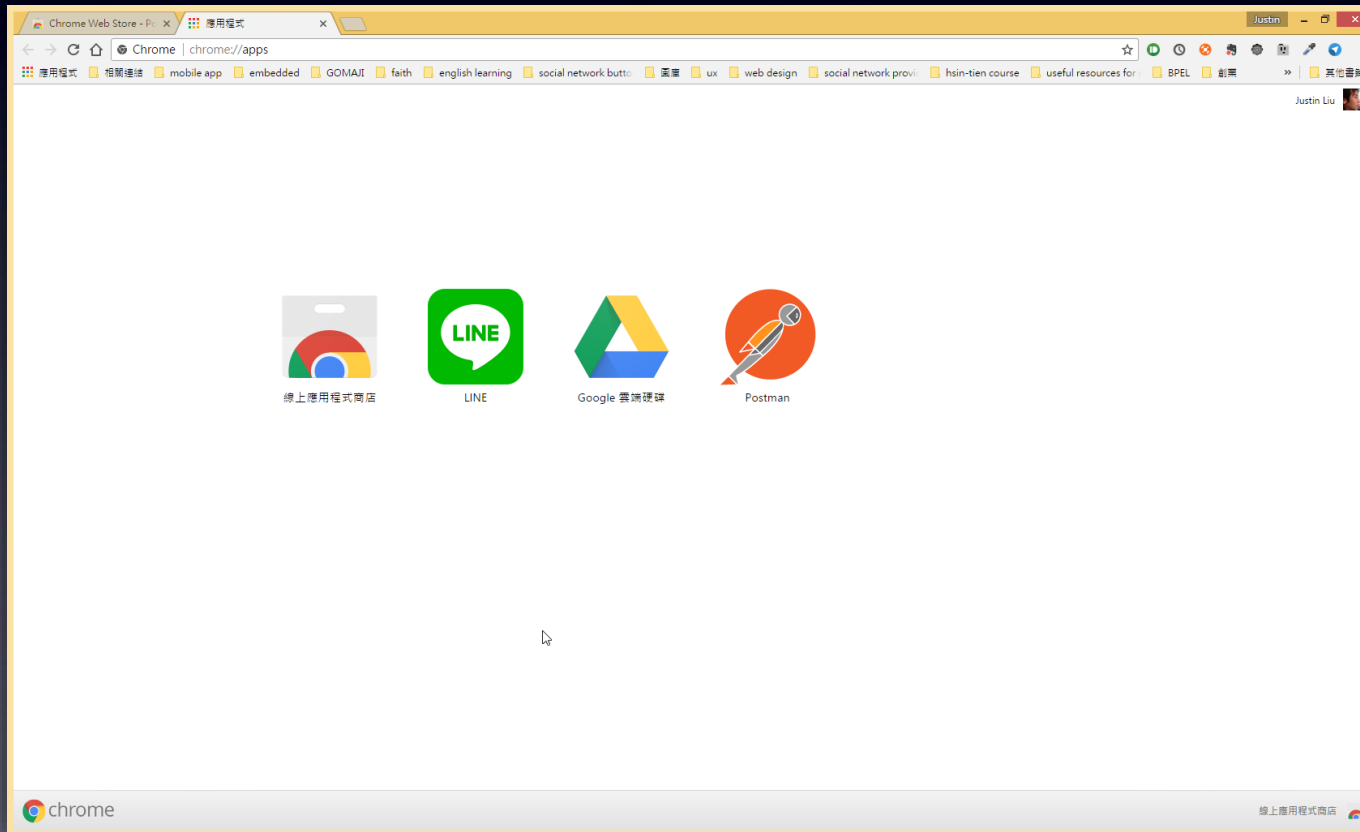
下載POSTMAN

- 選擇新增應用程式



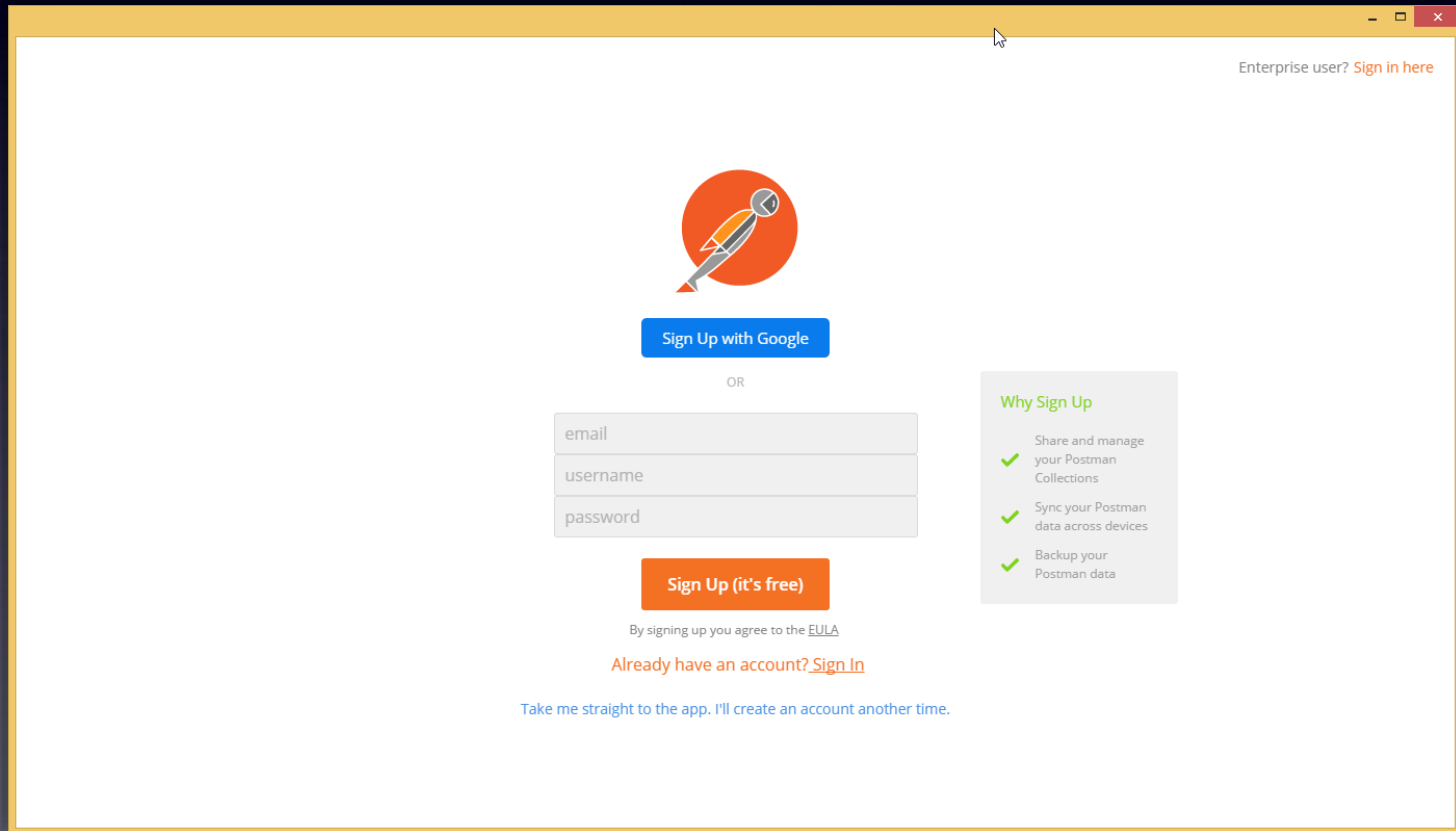
下載POSTMAN

- 安裝成功，點選一下開啟



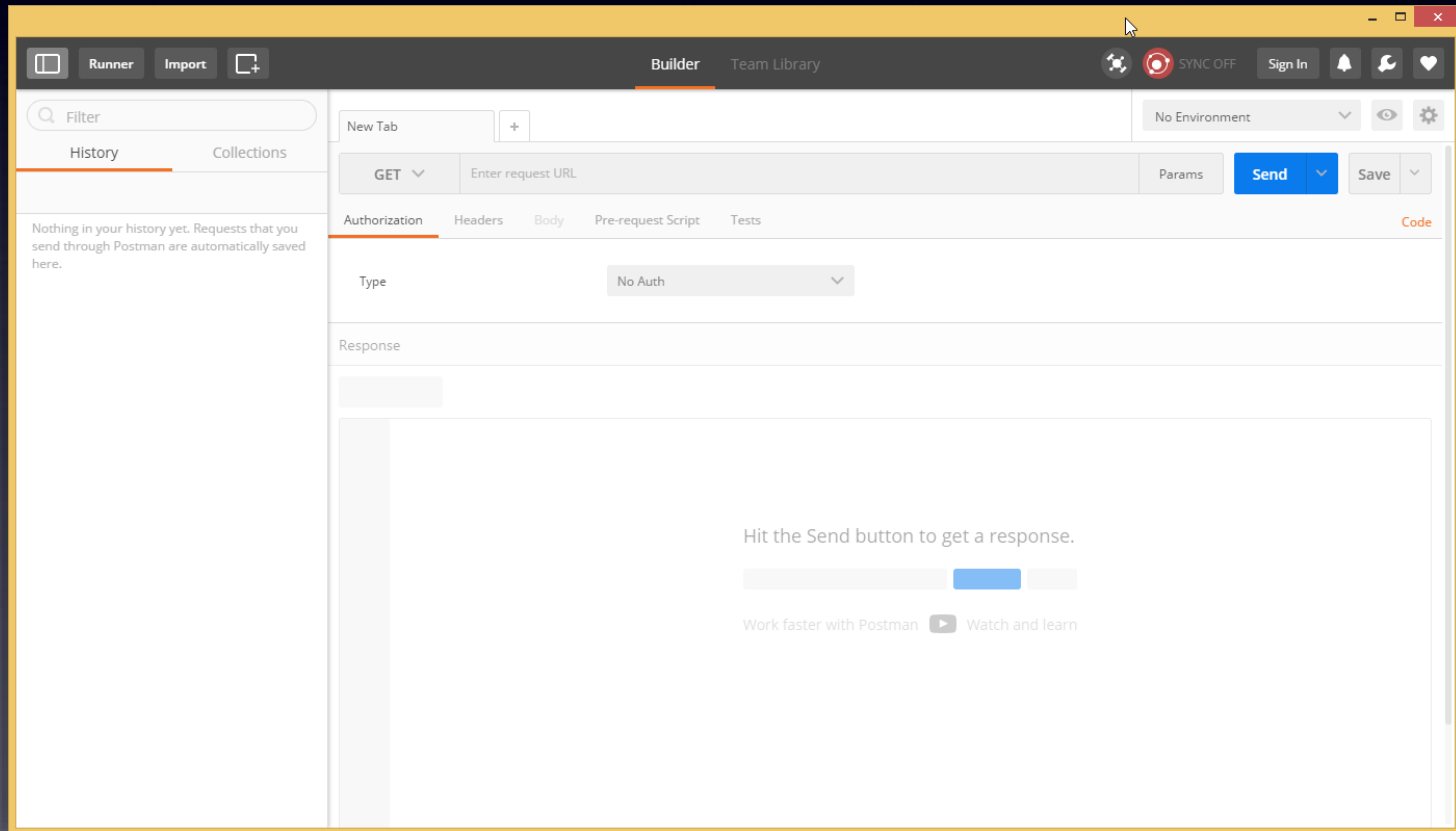
下載POSTMAN

- 選擇最下方Take me straight to the app...



下載POSTMAN

- 現在有可以測試API的工具了



下載POSTMAN

- 因著Google於12月初終止Chrome App的計畫
搜尋已經不再搜尋的到了
- Postman可直接到以下網址下載
 - https://www.google.com.tw/url?sa=t&rct=j&q=&esrc=s&source=web&cd=4&ved=0ahUKEwi4_OeFpfzXAhVDJ5QKHQv-AhYQFghCMAM&url=https%3A%2F%2Fchrome.google.com%2Fwebstore%2Fdetail%2Fpostman%2Ffhbjgbiflinjbdgggehcdcbncdddomop%3Fhl%3Dzh-TW&usg=AOvVaw3orE5cYNdV7xa-PuhONKhq

網路處理

試用API

試用API

- 本次範例為MixCloud
- API文件
 - <https://www.mixcloud.com/developers/>
- API範例
 - <https://api.mixcloud.com/search?q=party+time&type=cloudcast>

試用API

The screenshot displays the Postman web interface. At the top, there's a navigation bar with 'NEW', 'Runner', 'Import', and 'Builder' buttons. A notification banner states: 'Chrome apps are being deprecated. Download our free native apps for continued support and better performance. Learn more'. Below this, the 'MixCloud' tab is active, showing a collection named 'Google books'. The request method is set to 'GET' (highlighted with a red circle), and the URL is 'https://api.mixcloud.com/search?q=party+time&type=cloudcast' (also highlighted with a red rectangle). The 'Send' button is highlighted with a red rectangle. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab is active, displaying a JSON response in 'Pretty' format. The response status is '200 OK' and the time taken is '1784 ms'. The JSON data includes a 'paging' object with a 'next' URL and a 'data' array containing three items with 'tags' and 'url' fields.

```
1 {
2   "paging": {
3     "next": "https://api.mixcloud.com/search/?limit=20&offset=20&q=party+time&type=cloudcast"
4   },
5   "data": [
6     {
7       "tags": [
8         {
9           "url": "https://www.mixcloud.com/discover/top40/",
10          "name": "Top40",
11          "key": "/discover/top40/"
12        },
13        {
14          "url": "https://www.mixcloud.com/discover/edm/",
15          "name": "Edm",
16          "key": "/discover/edm/"
17        },
18        {
19          "url": "https://www.mixcloud.com/discover/progressive-edm/",
20          "name": "Progressive EDM",
21          "key": "/discover/progressive-edm/"
22        },
23        {
```

URLConnection

JAVA網路連線

JAVA網路連線

- HttpURLConnection
 - 1. 使用java.net.URL設定連線網址
 - 2. URL.openConnection()轉型為HttpURLConnection
 - 3. HttpURLConnection.setRequestMethod()設定方法
 - 4. HttpURLConnection.connect()進行連線
 - 5. HttpURLConnection.getResponseCode()取得狀態碼
 - 6. HttpURLConnection.getInputStream()取得回傳串流
 - 7. HttpURLConnection.getErrorStream()錯誤訊息串流
 - 8. 串流轉字串並呈現結果

HTTP狀態碼

- 依照W3C定義，Http網路連線有通用代碼代表狀態
 - <https://developer.mozilla.org/zh-TW/docs/Web/HTTP/Status>
 - <https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html>
- 要記得200，是代表連線正常
 - `URLConnection.HTTP_OK`

串流轉字串

- API呼叫的結果多以串流回傳，需要將串流轉為可讀的字串，方便閱讀及剖析

```
private static String streamToString(InputStream in) throws IOException {
    ByteArrayOutputStream baos = new ByteArrayOutputStream();
    byte[] buf = new byte[8192];
    for (;;) {
        int nread = in.read(buf, 0, buf.length);
        if (nread <= 0) {
            break;
        }
        baos.write(buf, 0, nread);
    }
    in.close();
    baos.close();
    byte[] bytes = baos.toByteArray();
    return new String(bytes, StandardCharsets.UTF_8);
}
```

JSON

JSON格式介紹

JSON格式介紹

- API現在最常見的格式為JSON
 - JavaScript Object Notation
- 輕量化資料交換格式
- JSON主要由JSON Object與JSON Array組成
- 建議：可以使用範例Java取得回傳的字串進行了解
 - 線上JSON格式化工具 <https://jsonformatter.org/>

JSON OBJECT

- 以 { 開始，以 } 結束
- 內容 Key(索引值)與Value(內容)
- 內容可以是
 - 整數或浮點數、字串(要用""框起)、布林(true, false)、JSON Array、JSON Object
- 中間以冒號:分隔 { Key : Value }
- e.g.
`{"name" : "android"}`

JSON ARRAY

- 以 [開頭，以] 結果
- 中間內容有序列表，每個陣列物件以 , 分隔
- e.g.

```
{  
    "values": [  
        {"value", 1.0}, {"value", 2.0}, {"value", 3.0}  
    ]  
}
```

Eclipse +JSON

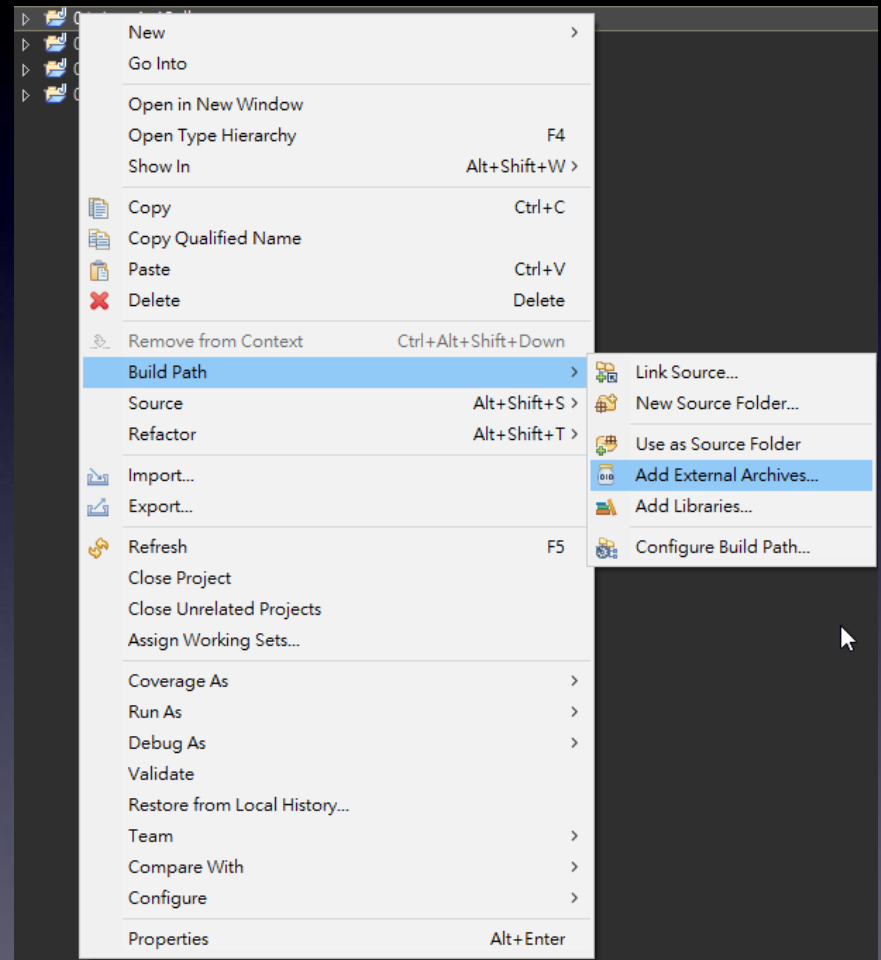
匯入第三方函式庫剖析JSON

匯入第三方函式庫剖析JSON

- 剖析JSON需要用到第三方函式庫
- 下載位置
 - <http://www.java2s.com/Code/Jar/j/Downloadjavajsonjar.htm>
- 下載後解壓縮，要記得擺放的位置
- 解壓縮後的檔案，副檔名應該是jar

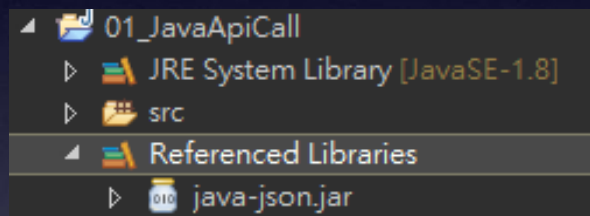
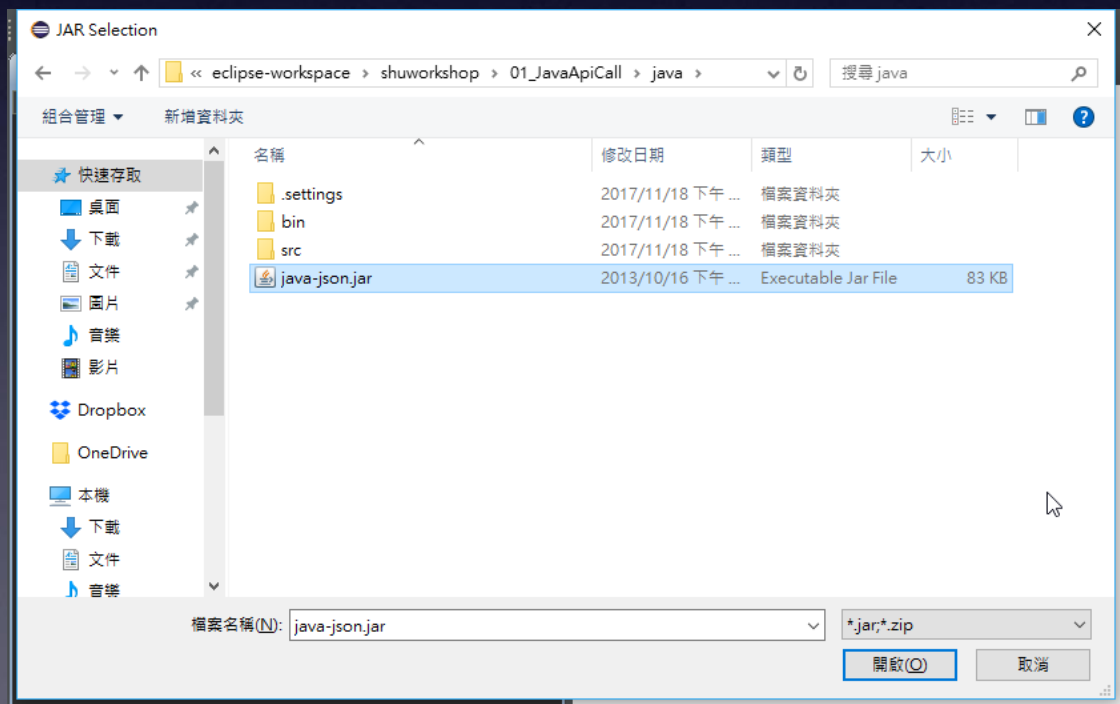
匯入第三方函式庫剖析JSON

- 對專案按下滑鼠右鍵
- 選擇Build Path -> Add External Archives



匯入第三方函式庫剖析JSON

- 跳出檔案選擇視窗，選擇剛剛下載的JSON jar
- 選擇後按下開啟
- 接下來專案的Referenced Libraries就會有出現jar了



JSON

剖析JSON

剖析JSON

- 重點提示
 - 取得Int `JSONObject.optInt()`
 - 取得String `JSONObject.optString()`
 - 取得JSONObject `JSONObject.optJSONObject()`
 - 取得JSONArray `JSONObject.optJSONArray()`
- 注意！JSON的剖析，多練習就會了

Object + ArrayList

剖析結果的處理

剖析結果的處理

- 回傳資料物件化，JSONObject取出需用的資料，就對應一個Class來儲存

```
{  
  "userId": 1,  
  "id": 1,  
  "title": "...",  
  "body": "..."  
}
```

```
public class BasicObject {  
    int userId;  
    int id;  
    String title;  
    String body;  
}
```

```
BasicObject obj = new BasicObject();  
obj.userId = response.optInt("user_id");  
obj.id = response.optInt("id");  
obj.title = response.optString("title");  
obj.body = response.optString("body");
```

剖析結果的處理

- 需用的資料整理完後經常也是多個物件，建議使用 ArrayList 來處理
- 因為陣列(Array)長度是固定的，對於多次呼叫API來整理結果比較麻煩
- ArrayList的使用方式重點提示
 - 建立 `new ArrayList<物件>()`
 - 填入資料 `ArrayList.add(物件)`, `add(索引值, 物件)`
 - 取出資料 `ArrayList.get(索引值)`
 - 移除資料 `ArrayList.remove(索引值)`
 - 取得ArrayList的長度 `ArrayList.size()`

Homework

作業

作業說明

- 建立簡單的Java應用程式，呼叫Google book API
 - <https://www.googleapis.com/books/v1/volumes?q=<關鍵字>>
- 應用程式啟動時，要詢問使用者要搜尋甚麼書籍
- 使用者的輸入是帶入API<關鍵字>的參數
- 取得的結果為JSON，剖析後要建立成物件
- 物件儲存的資訊包含書名、圖片位置、網頁連結
 - 這部分請去了解API回傳的結果
- 物件儲存於ArrayList
- 跑回圈掃描ArrayList，印出其中每個物件的書名、圖片位置、網頁連結

Q & A