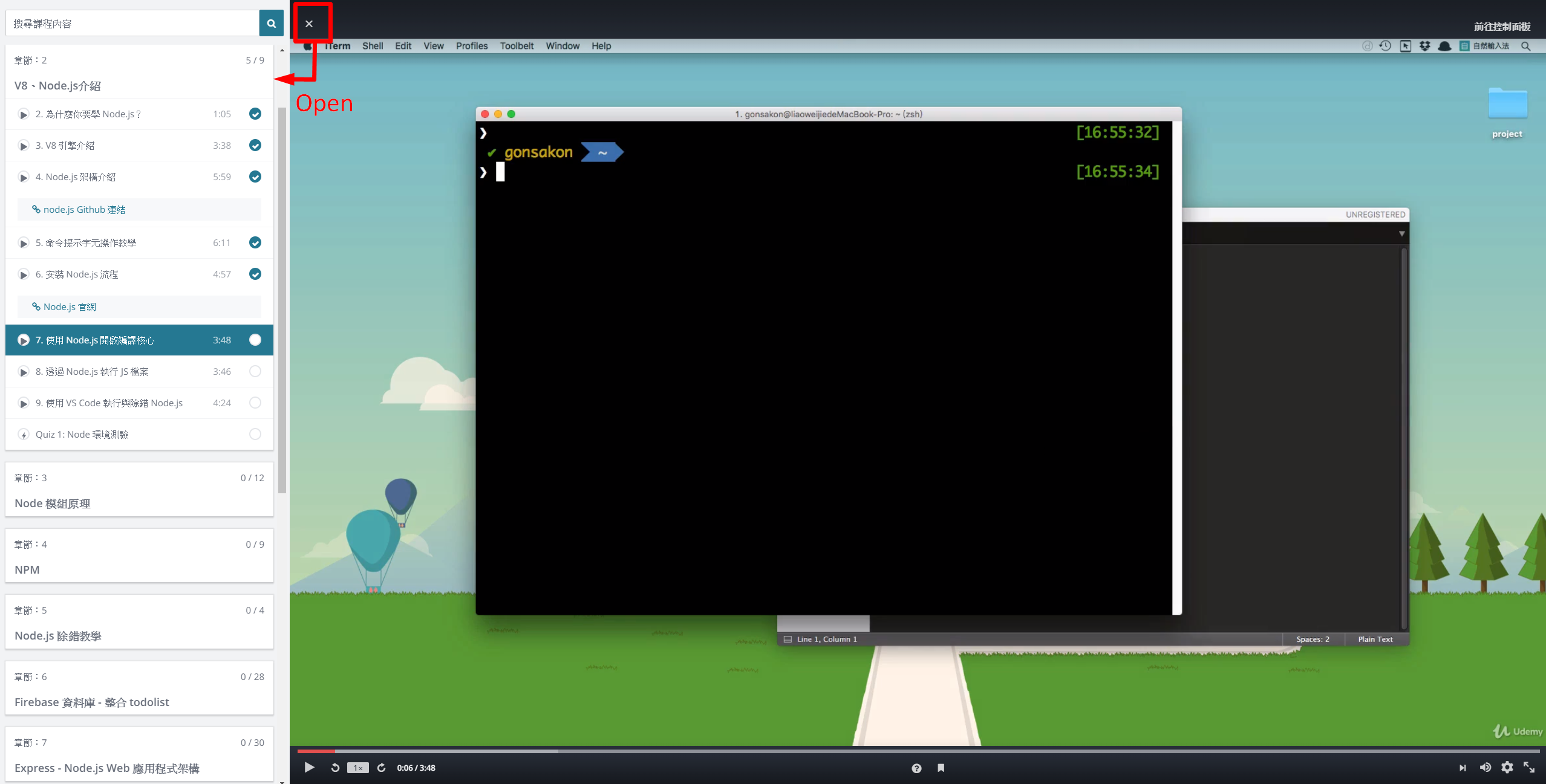
# Udemy nodejs 六角學院 教學

<https://www.udemy.com/teachnodejs/?couponCode=COUPON_2018>



# V8、Node.js介紹

## 2-5命令提示字元

要執行nodejs 都需要靠”命令提示字元”

首先來基本的

cd 移動

cd 把資料拖進來，可以直接呈現該path

顯示當層目錄

Window   
dir

Mac、linex

ls

## 2-6 安裝 Node.js 流程

由於我有安裝nvm 版本控制

所以這塊不照影片上做了

Cmd 查詢目前電腦node version

node --version

使用nvm 指令

查詢目前已安裝的node版本 (\*目前正在使用的版本)

nvm ls

查詢線上可安裝的版本

nvm ls available

安裝其他版本

nvm install \*.\*\*

ex > nvm install 8.11.1

切換版本

nvm ues \*.\*\*

來源參考:

<http://bubkoo.com/2017/01/08/quick-tip-multiple-versions-node-nvm/>

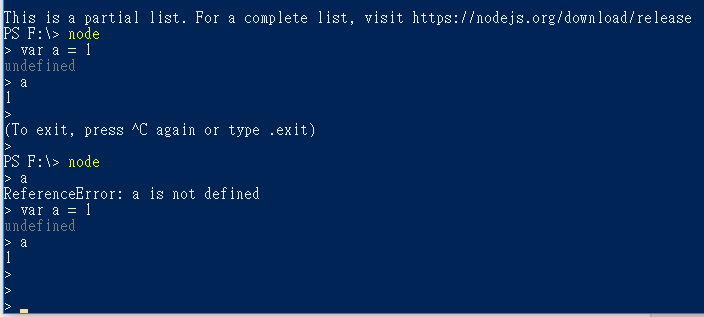
## 2-7 使用 Node.js 開啟編譯核心

使用nodejs的環境

Cmd > node

此時cmd 會變成 >

代表可以輸入javascript (像是chorme 的 console.log)



離開node環境

Ctrl+C

跳出 (To exit, press ^C again or type .exit)

Ctrl+C

## 2-8 透過 Node.js 執行 JS 檔案

首先cd 至 nodejs要執行的js path

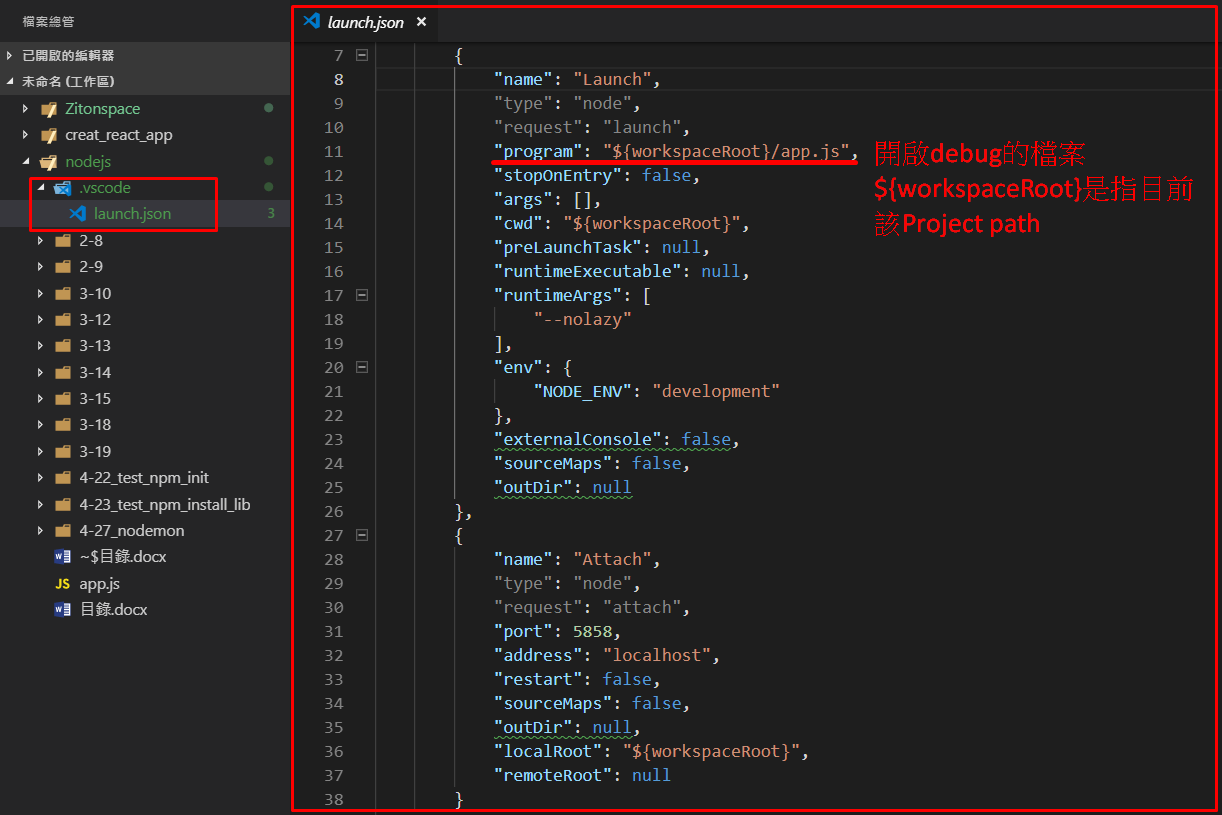
使用cmd 指令 > node .\\*\*\*.js



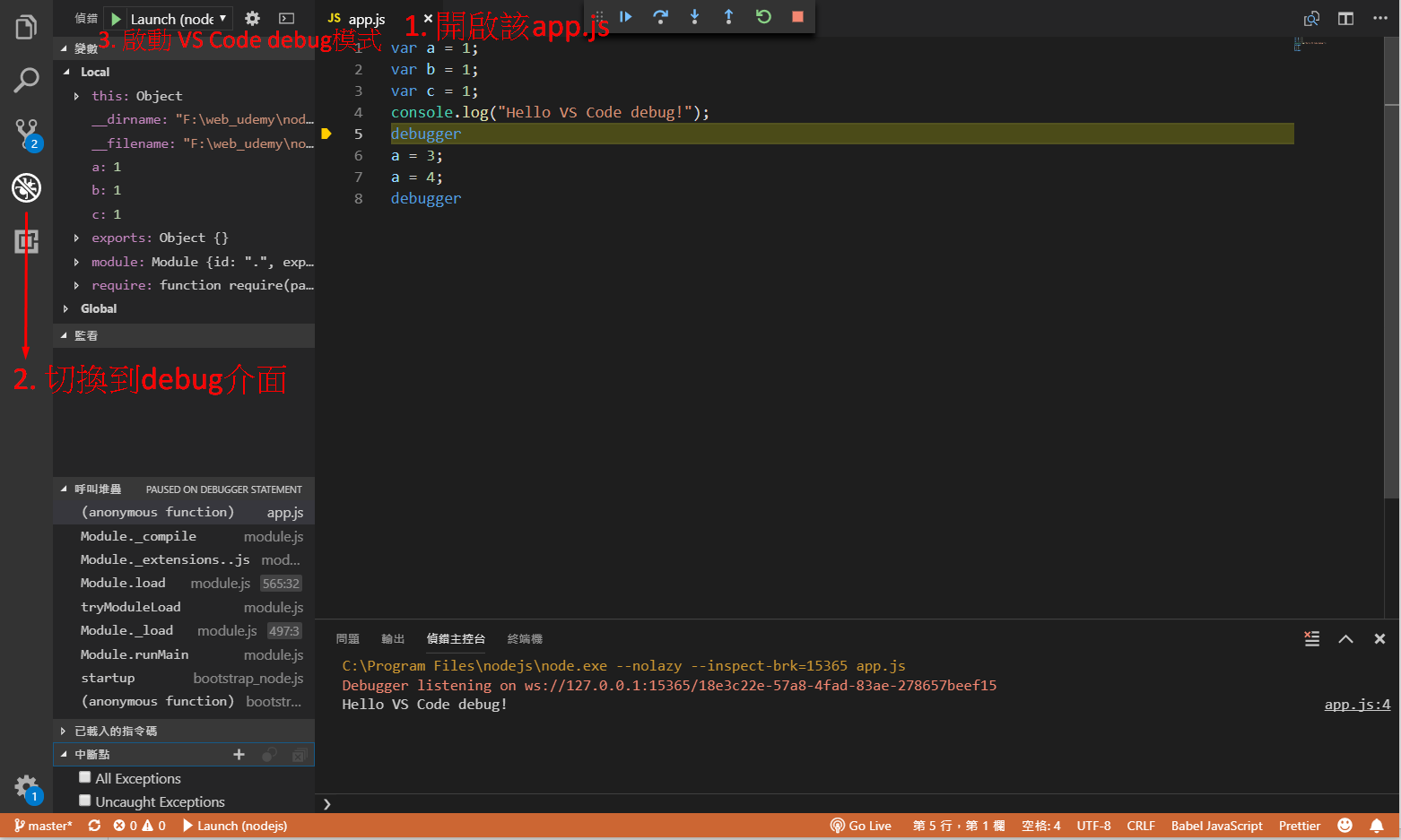
## 2-9 使用 VS Code 執行與除錯 Node.js

使用VS Code debug

首先VS Code會依賴一支.json 去開啟要debud的js



所以目前開啟debug的檔案在Project (nodejs folder)底下的app.js



5-31會有更詳細的debug方法

# Node 模組原理

## 3-10 Global 全域變數

Nodejs跟 js一樣有全域變數

Js為 window

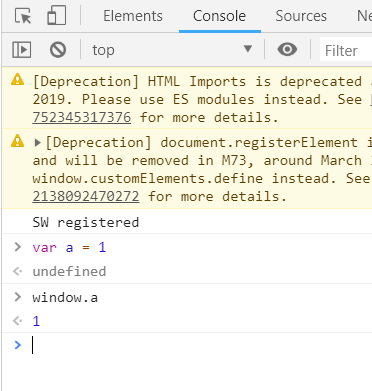
Nodejs 為 global

Js

一般在非function or Object中寫  
var a = 1;

這個a都會寫進window裡

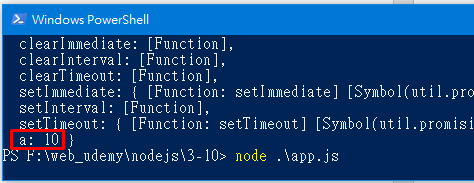
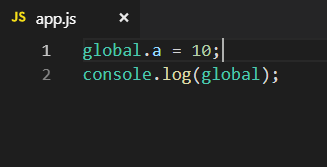
Ex.



在Nodejs 寫進global方式為

global.a = 1;

> node .\app.js



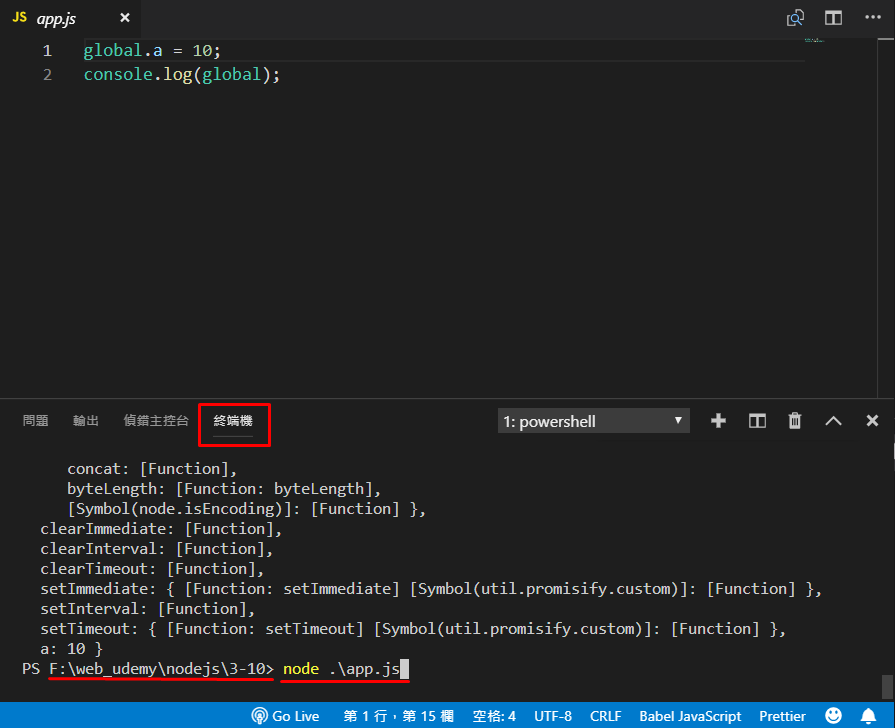
在Nodejs

var 是不會汙染nodejs的global區域的

因為之後結構方向為將每一支.js劃分為一個模組，互相include方式

## 3-11 VS CODE 執行 Node.js 應用程式

使用vs code 的cmd (終端機)

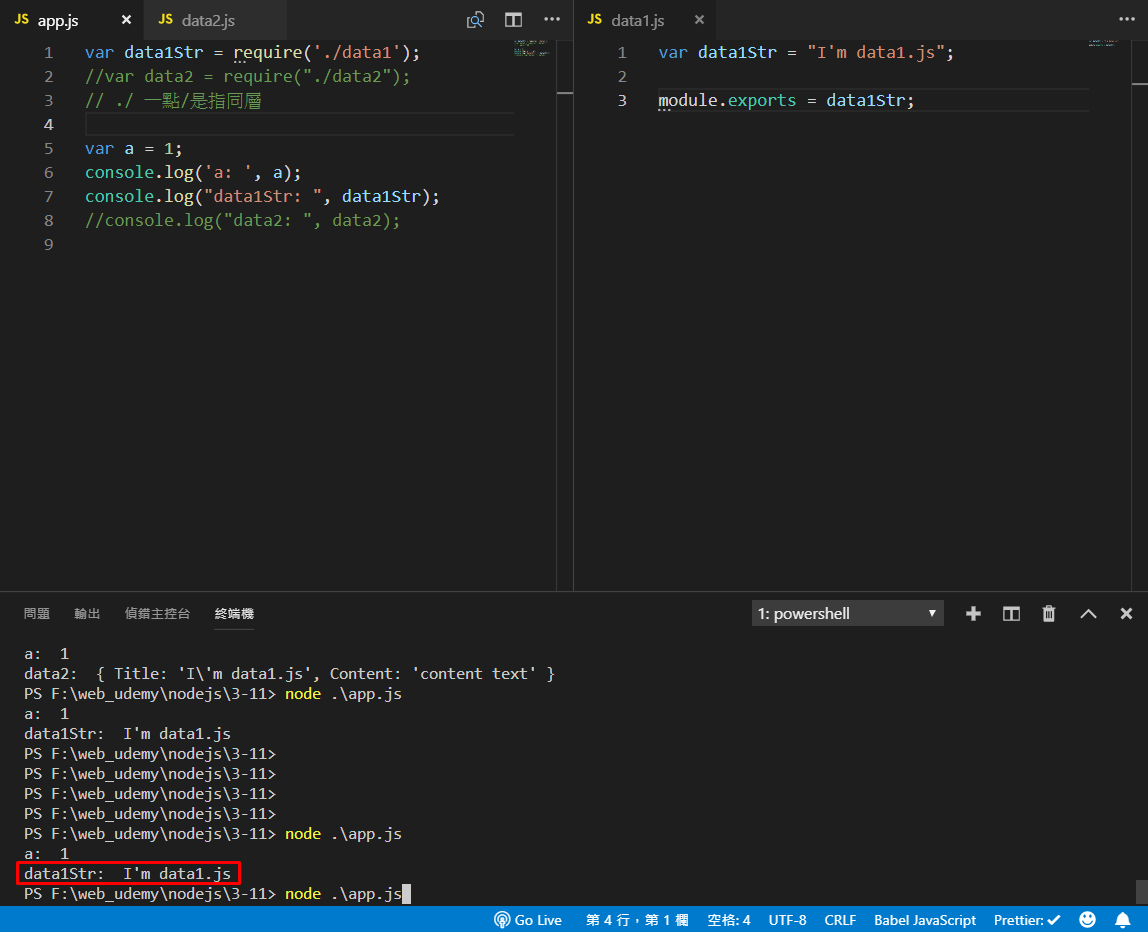


## 3-12 require、module exports 模組設計

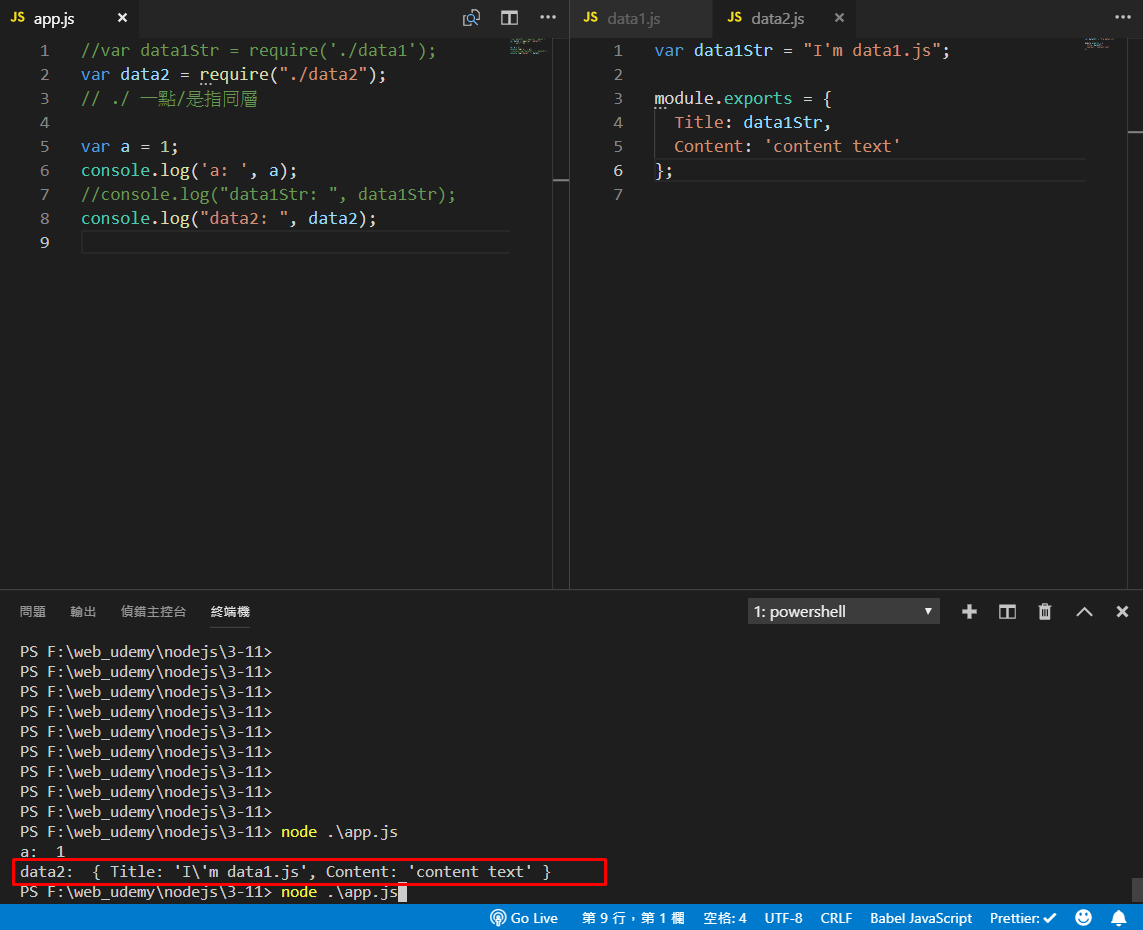
教學影片使用的ES5 require方式

但為了學習順利，目前還是依ES5方式去學習

單純require 一個var 變數



或是 require Object



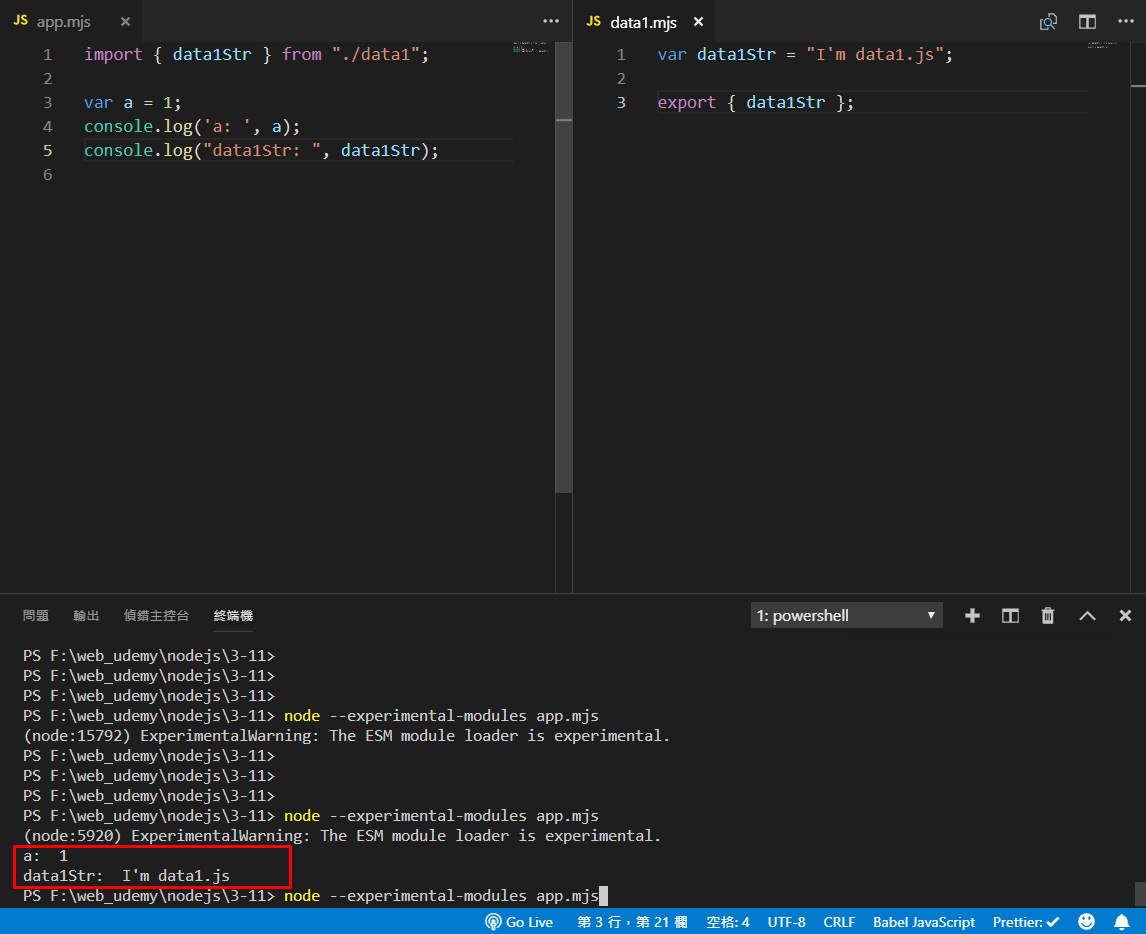
以下為ES6 的方式 import & exports

使用ES6 用Node

Cmd要改成使用此指令

node --experimental-modules \*\*.mjs

js部分也要需要改成 .mjs



該解決方式來源

<https://stackoverflow.com/questions/45854169/how-can-i-use-an-es6-import-in-node>

更多ES6 import & exports 解構方式

<http://es6.ruanyifeng.com/#docs/module>

<https://developer.mozilla.org/zh-TW/docs/Web/JavaScript/Reference/Statements/export>

## 3-13 exports 模組設計

使用require兩種方式，與衝突可能

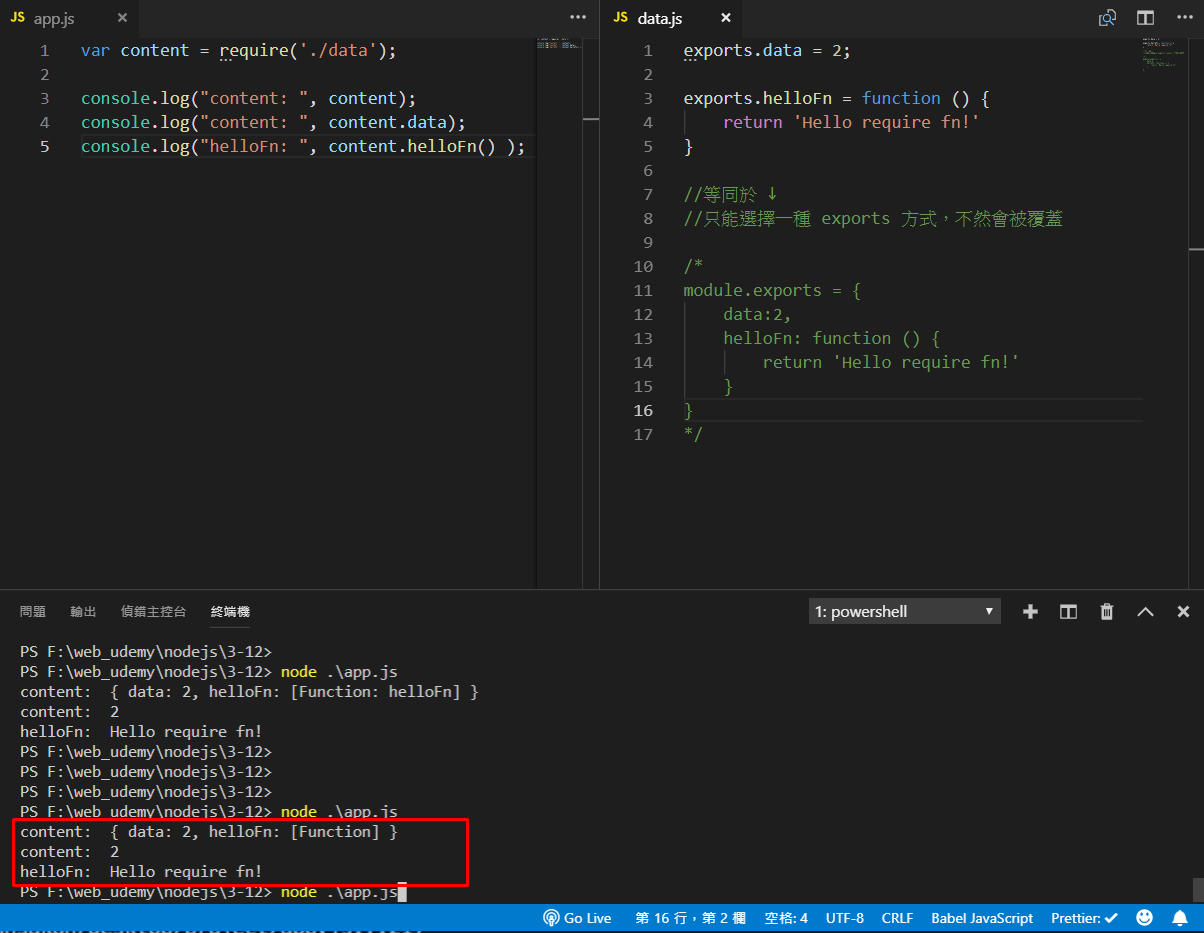
exports.data = 2;

and

module.exports = {

…

}

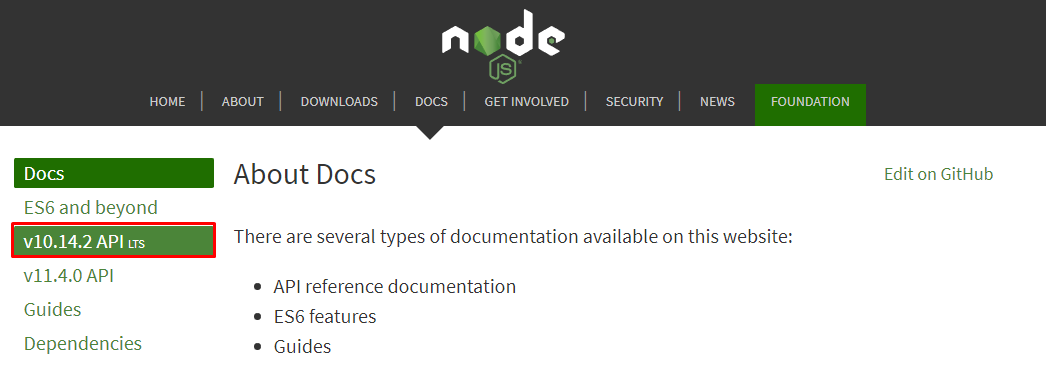


## 3-14 Node.js 核心模組 - createServer (上)

Nodejs 官方提供模組

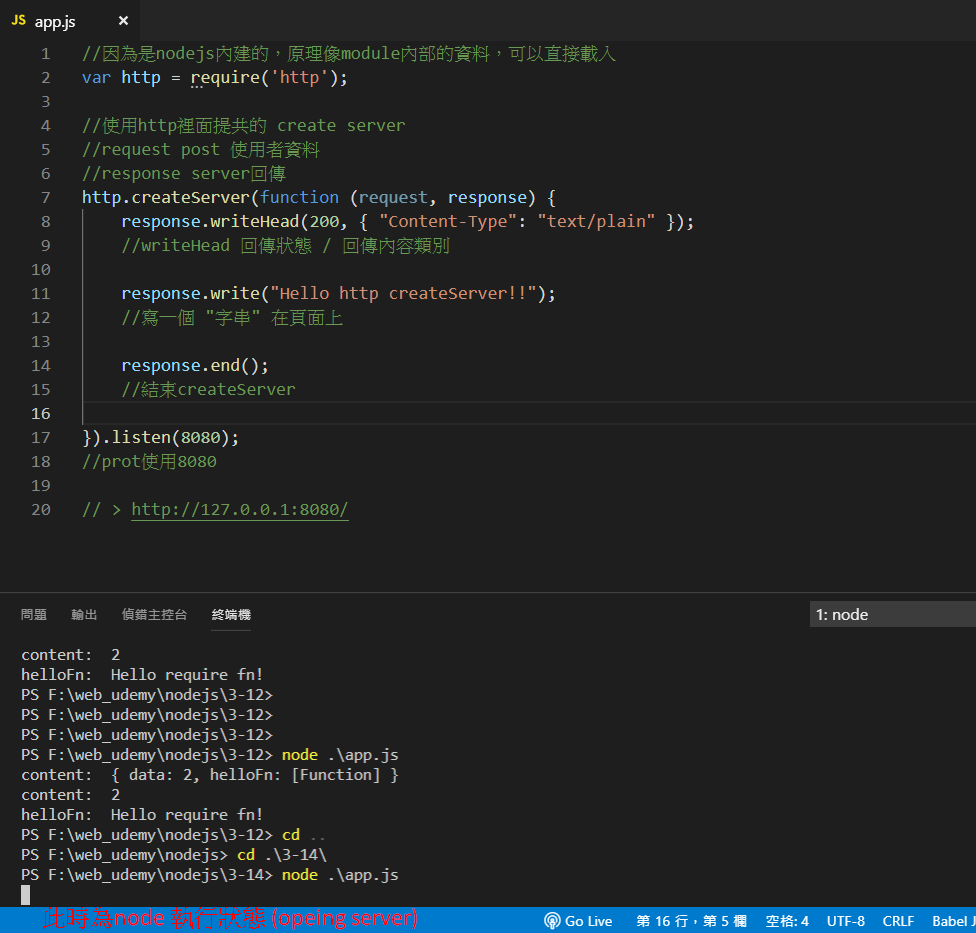
<https://nodejs.org/en/docs/>

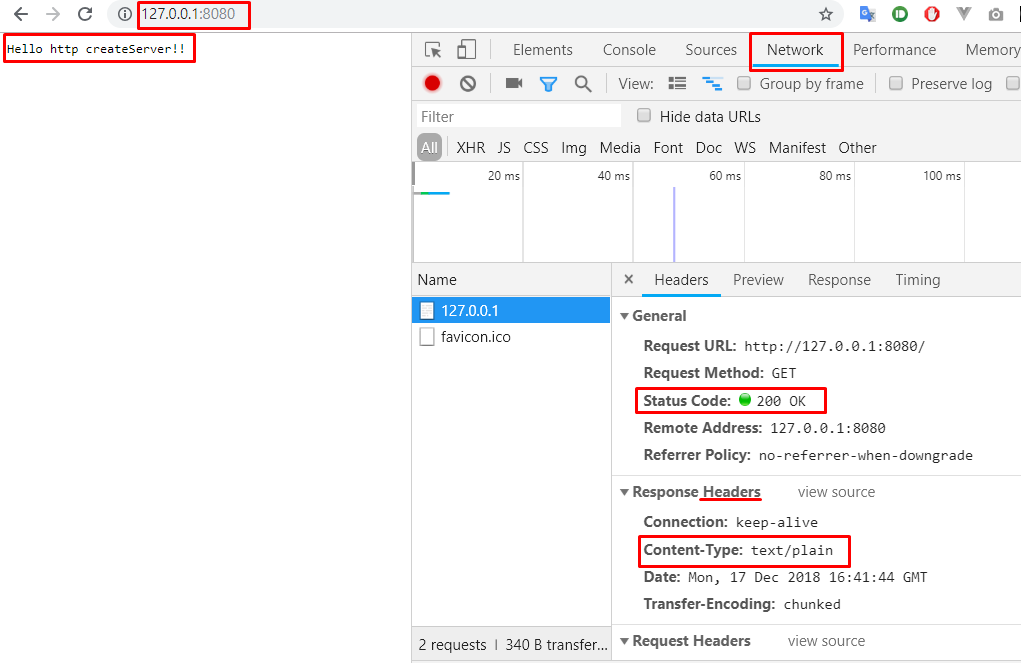
目前版本是 v10 (2018/12)



此次要使用nodejs提共的 http模組 來create servier

<https://nodejs.org/dist/latest-v10.x/docs/api/http.html#http_http_createserver_options_requestlistener>

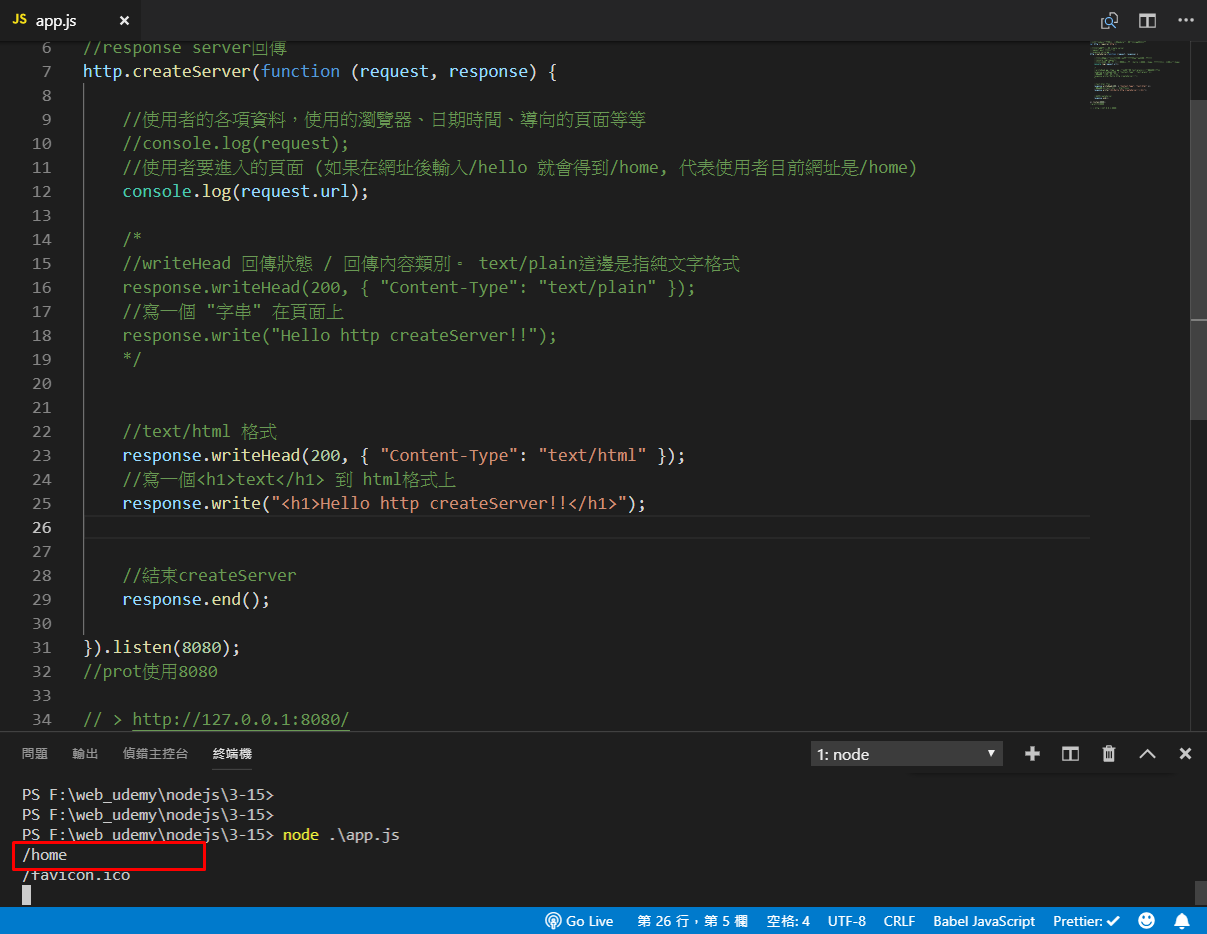




## 3-15 Node.js 核心模組 - createServer (中)

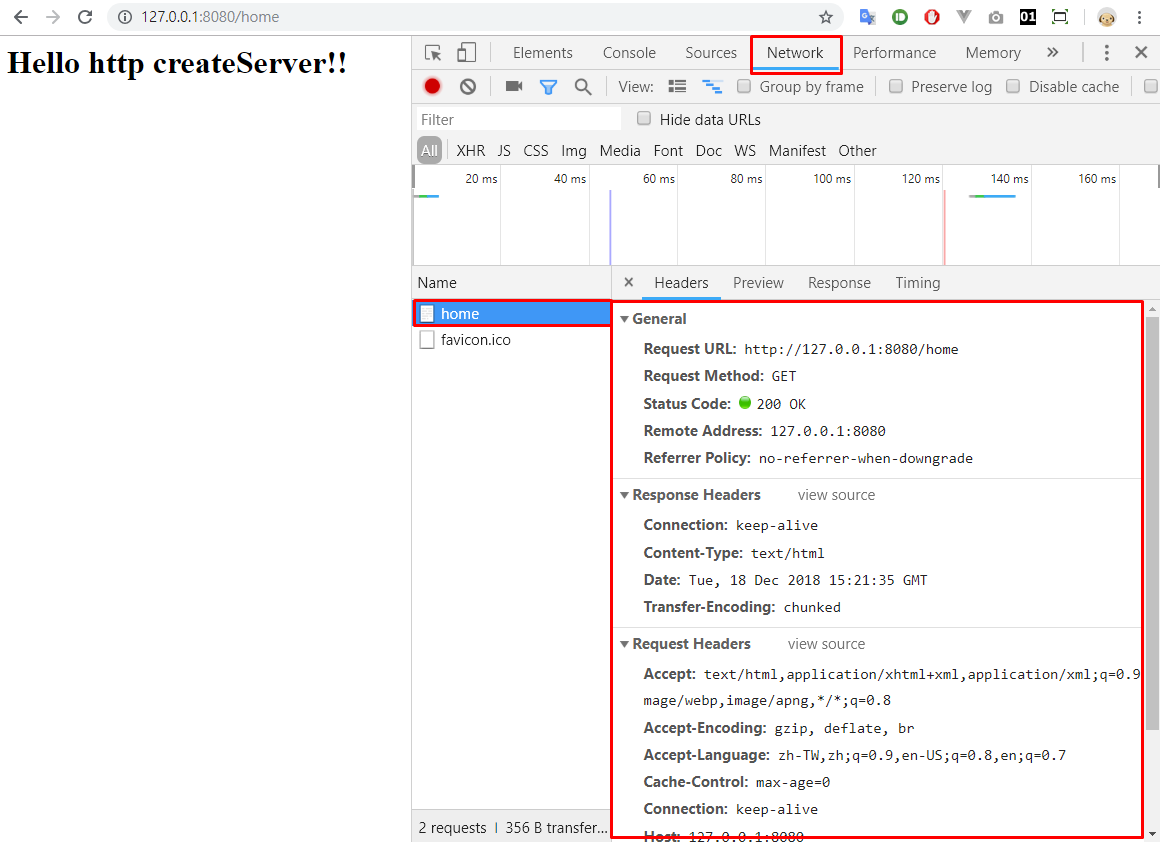
Request 與 response 分別解析





可以從Chrome Network

查看request & response 資料



## 3-17 Node.js 核心模組 - createServer (下)

解釋 127.0.0.1 : 8080 什麼意思

127.0.0.1 是指本機模擬web server 等同於localhost (非外部連結)

可參考: <https://zh.wikipedia.org/wiki/Localhost>

8080 (port or 通訊埠) 是指 本機 web 啟用的某一個軟體，佔據一個port

Ex. FTP 就常使用:21

:80 http

:3389 遠端桌面

一個相同port 只能應用一個軟體，例如:8080 已經被使用，那這段時間就不能再被其他軟體佔據。

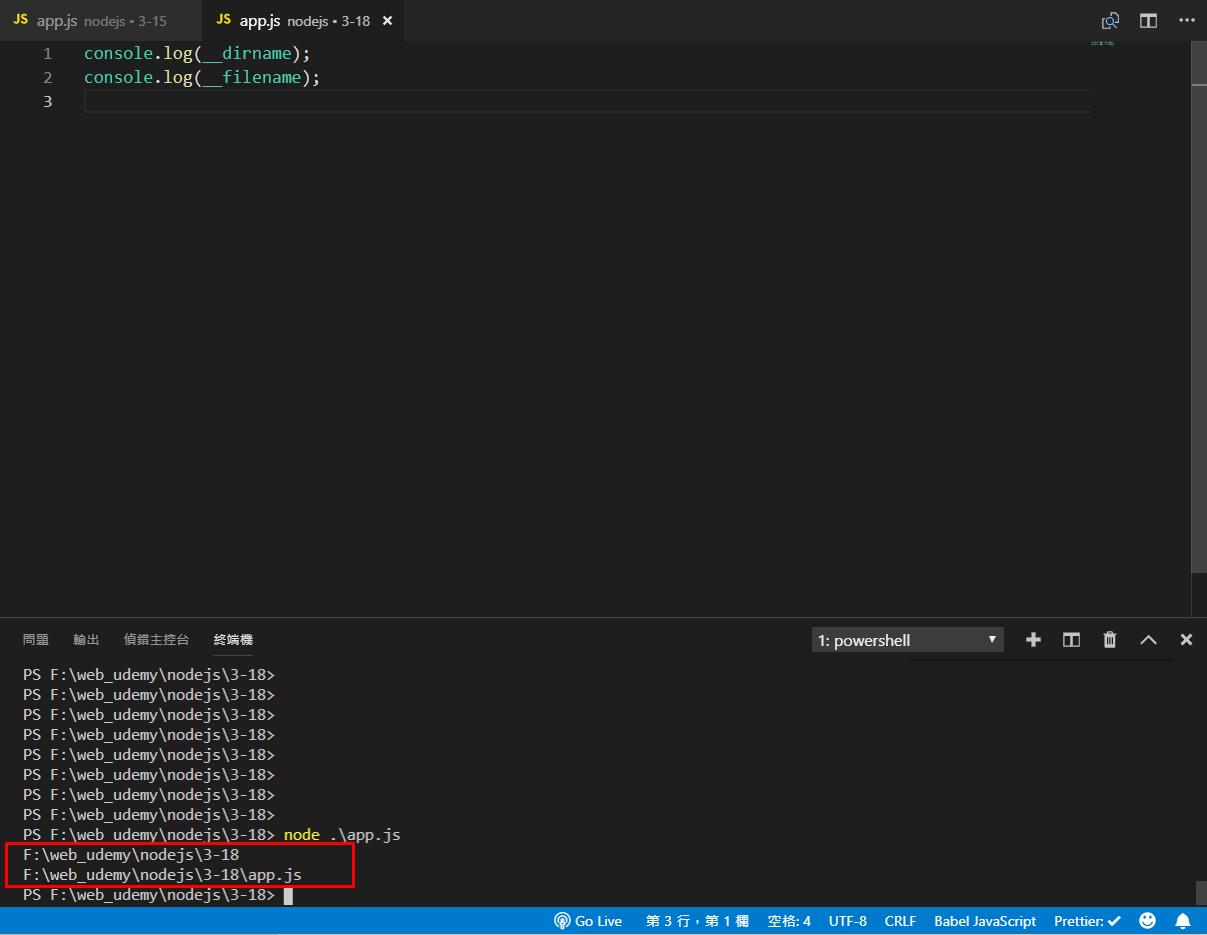
## 3-18 \_ \_dirname、\_ \_filename

解釋這兩個常使用的變數

這個是nodejs本身default會帶的變數

\_\_dirname > 目前porject的folder path

\_\_filename > 目前執行的哪一支.js



## 3-19 Node 模組 Path

Path 模組有許多查看路徑的功能

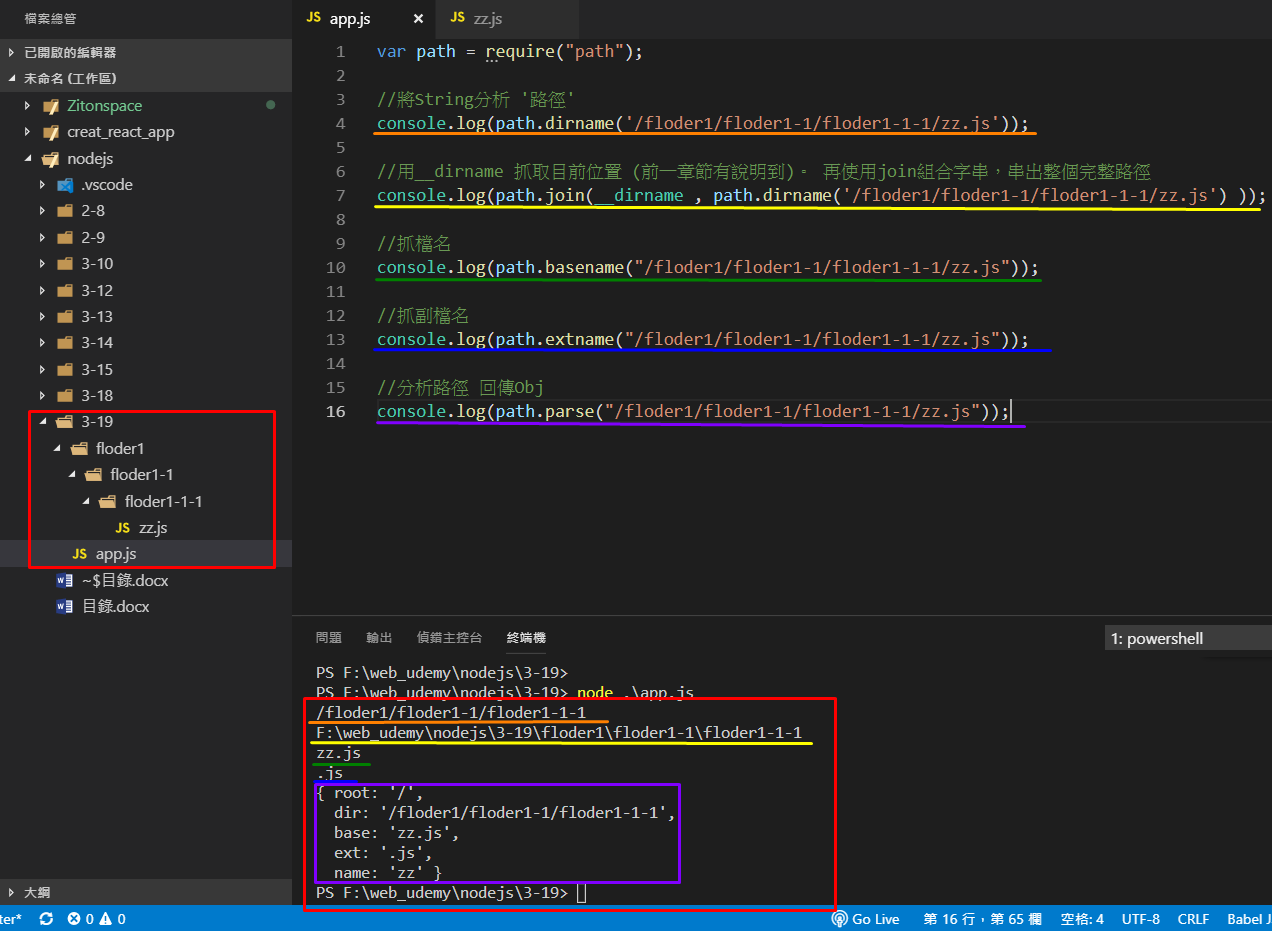
例如以下

dirname(“String path”)

就可以傳入資料夾結構的路徑，可以將字串分析出

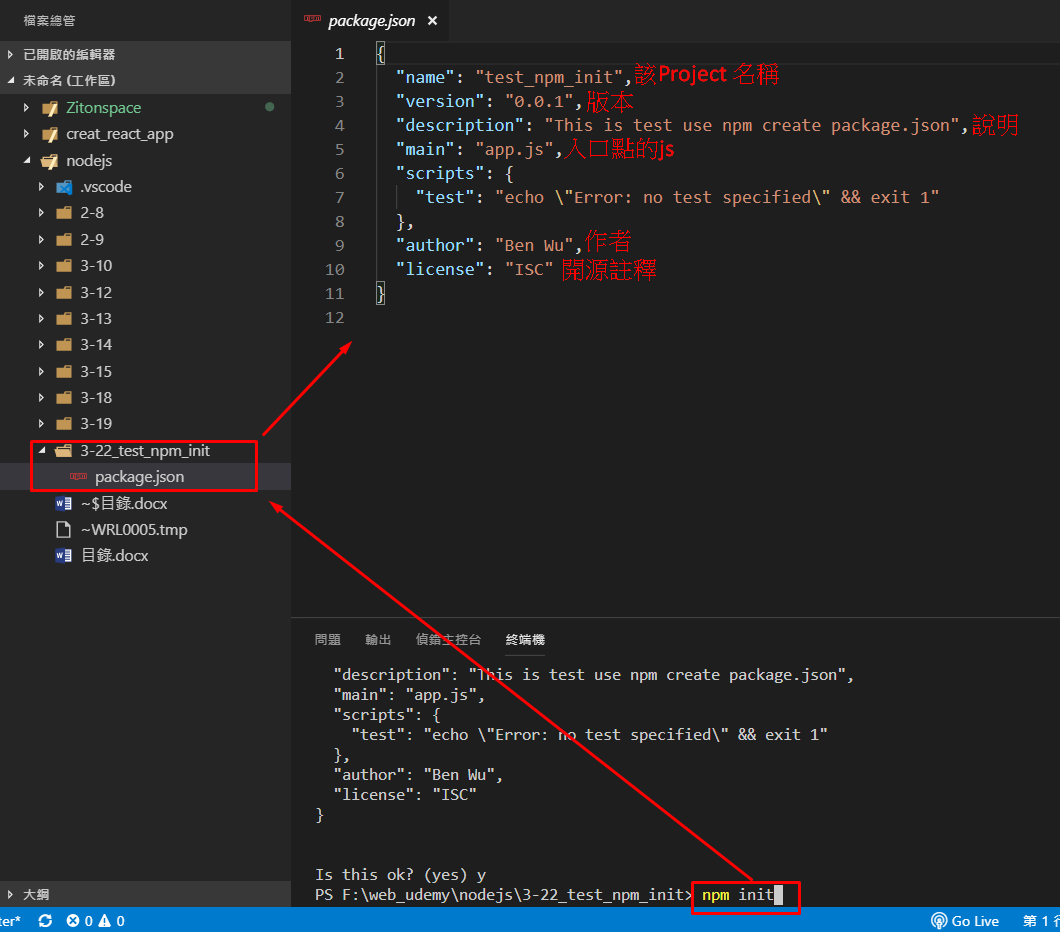
路徑、檔名、副檔名

當你載入 var path = require('path'); ，便可用下述語法取得檔案與目錄路

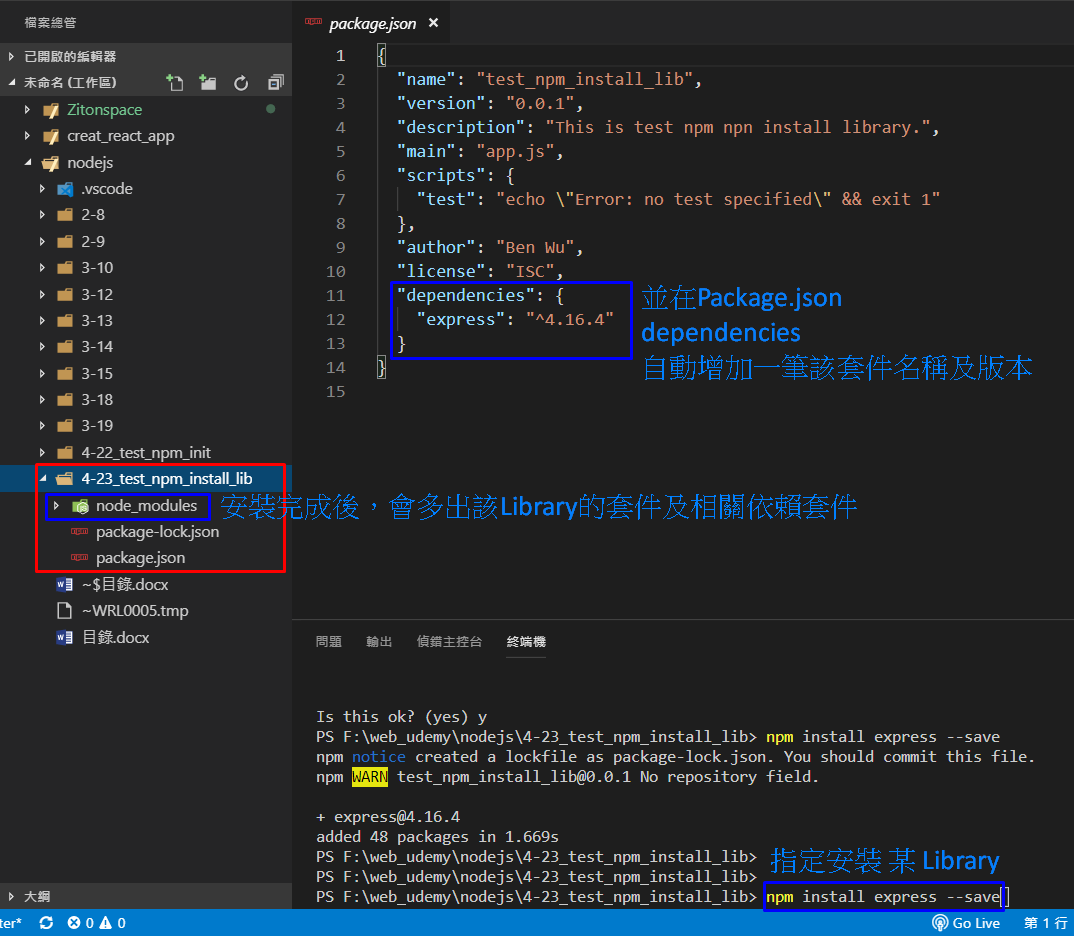


# NPM

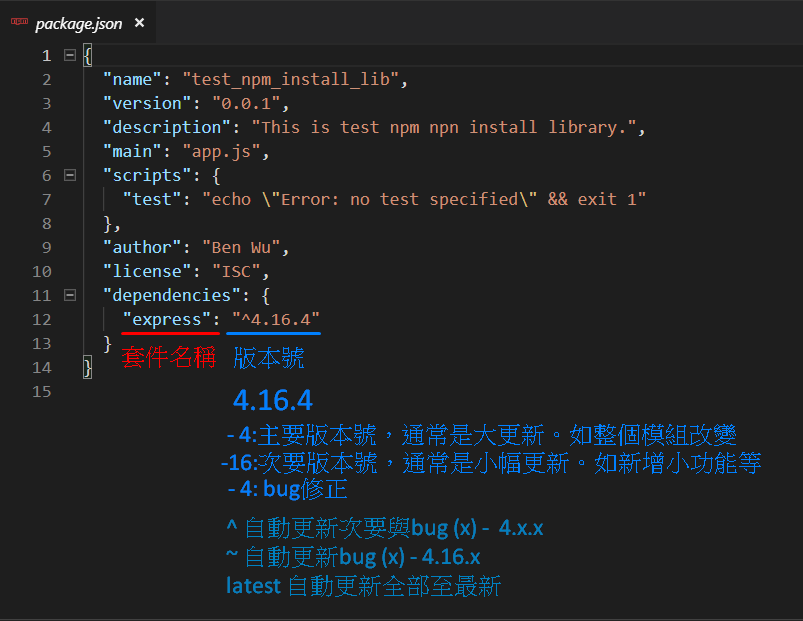
## 4-22 npm init - 開發自己的 package.json



## 4-23 安裝 npm 流程



## 4-24 NPM 版本號介紹



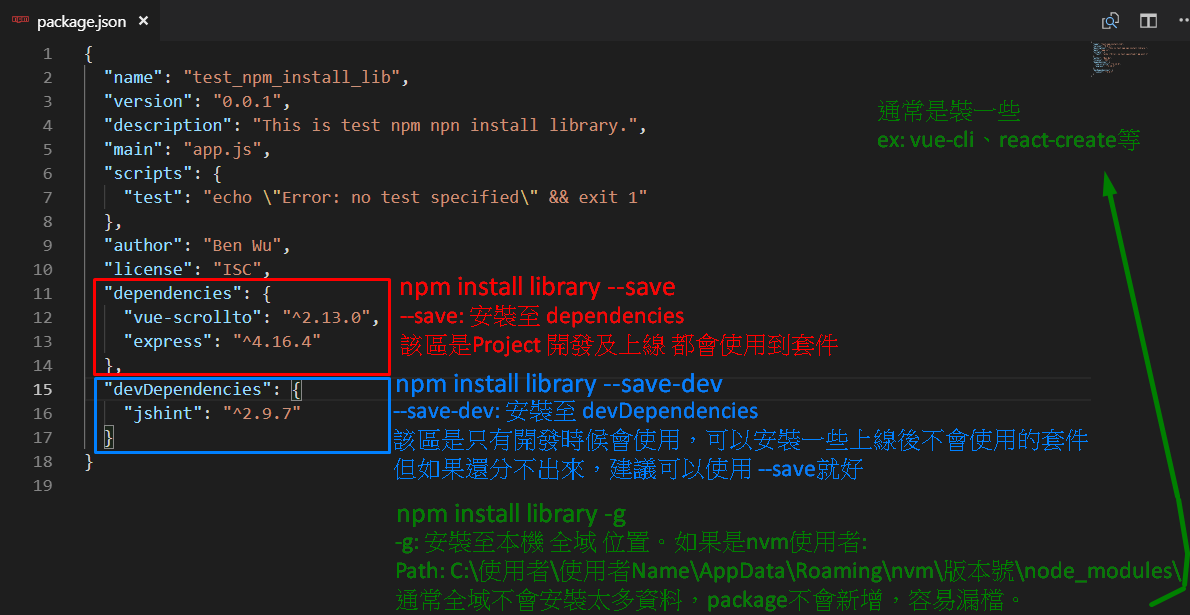
## 4-25 npm install 的妙用

一般來說，拿到別人的專案，不會有node\_modules 資料，因為太龐大了

npm install 指令是可以依賴該Project裡的package.json 內 dependencies安裝套件的項目

在去把node\_modules 安裝起來。

## 4-26 npm install -save、--save-dev、-g 差異



## 4-27 執行 NPM 內容流程

同 4-26

某一些方便開發的套件就可以安裝至全域

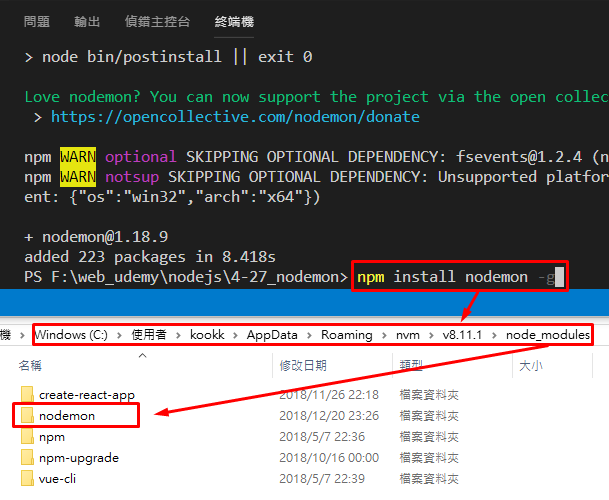
這邊介紹一個 套件

nodemon

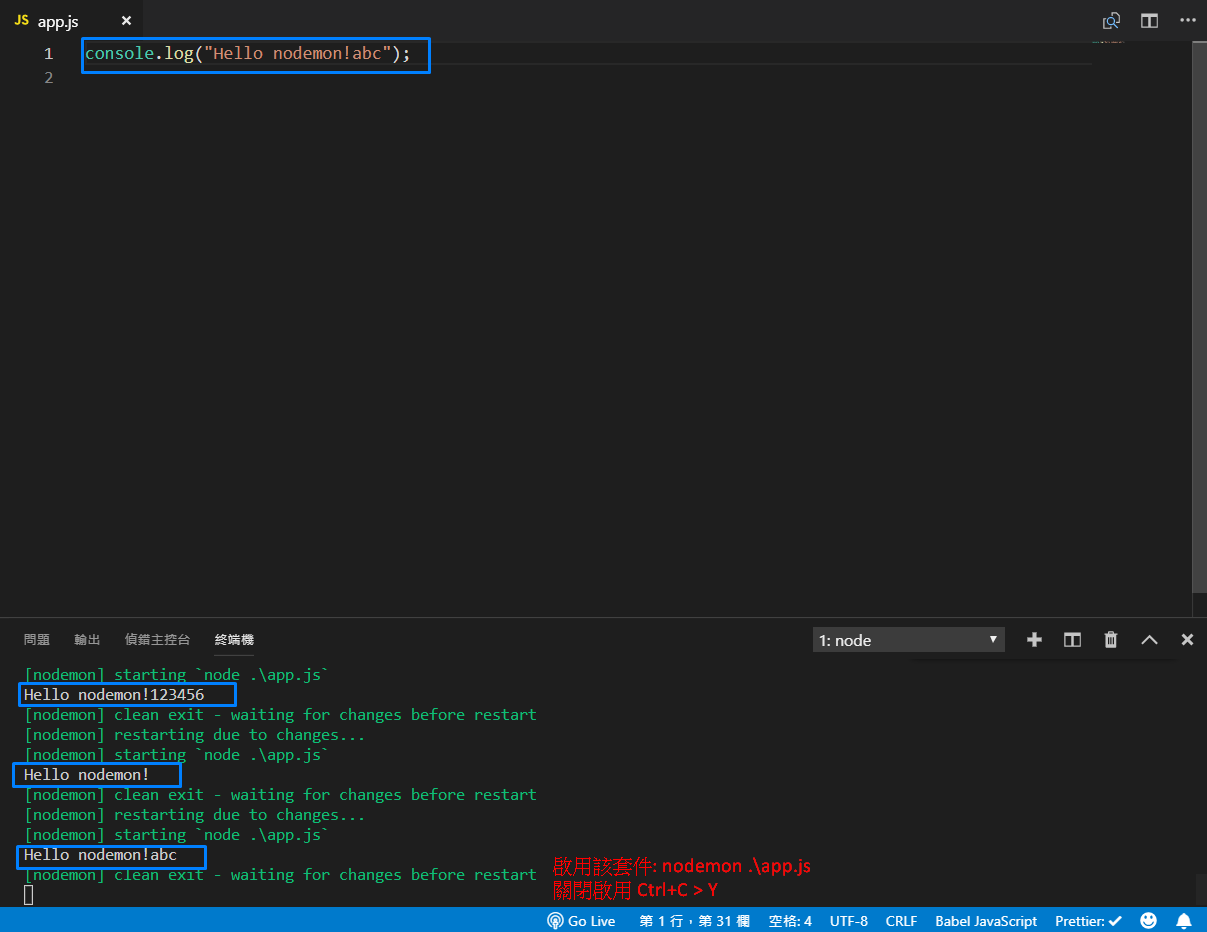
<https://www.npmjs.com/package/nodemon>

這是一個可以不用每次修改儲存一次，都要重新在cmd ./node app.js

安裝至全域



啟用

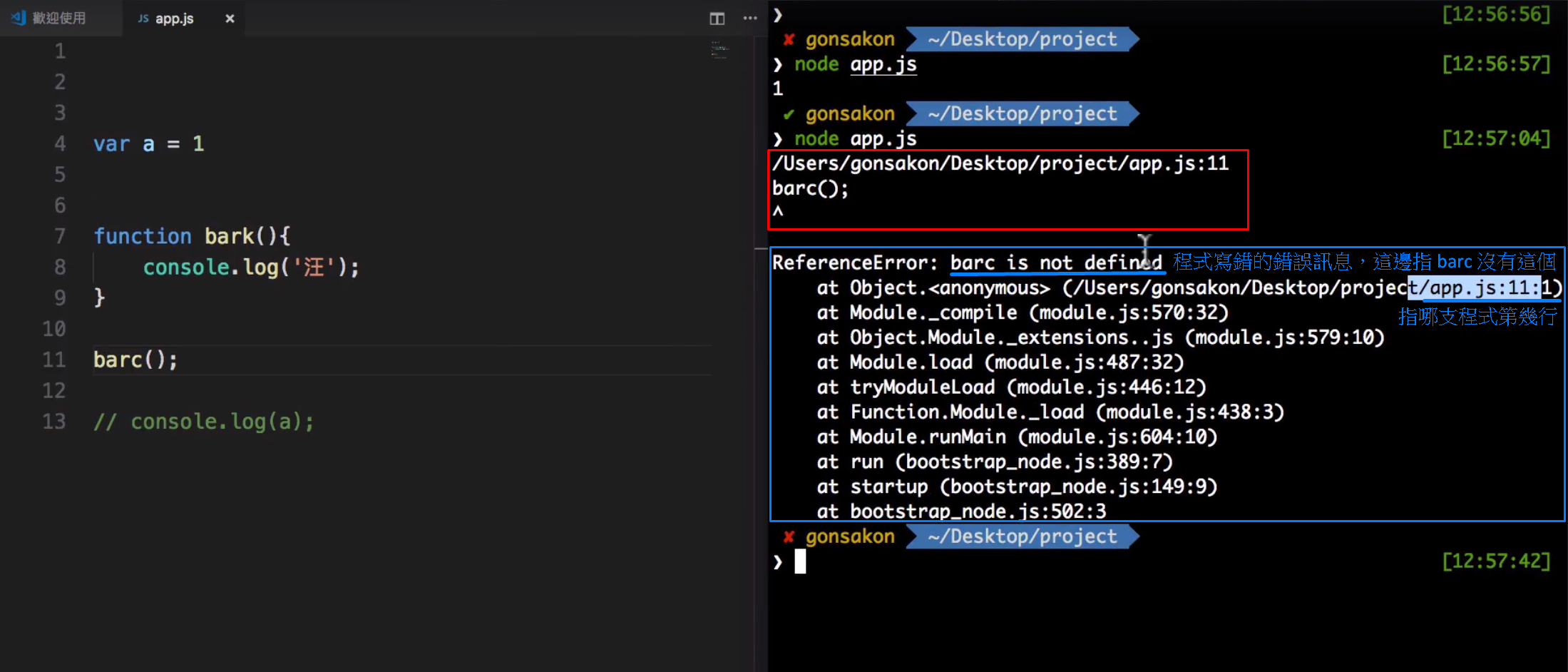


NPM 常用指令等



# Node.js 除錯教學

## 5-29 程式出錯莫驚慌，log 探索自己來



## 5-30 node 內建 chrome dev tools

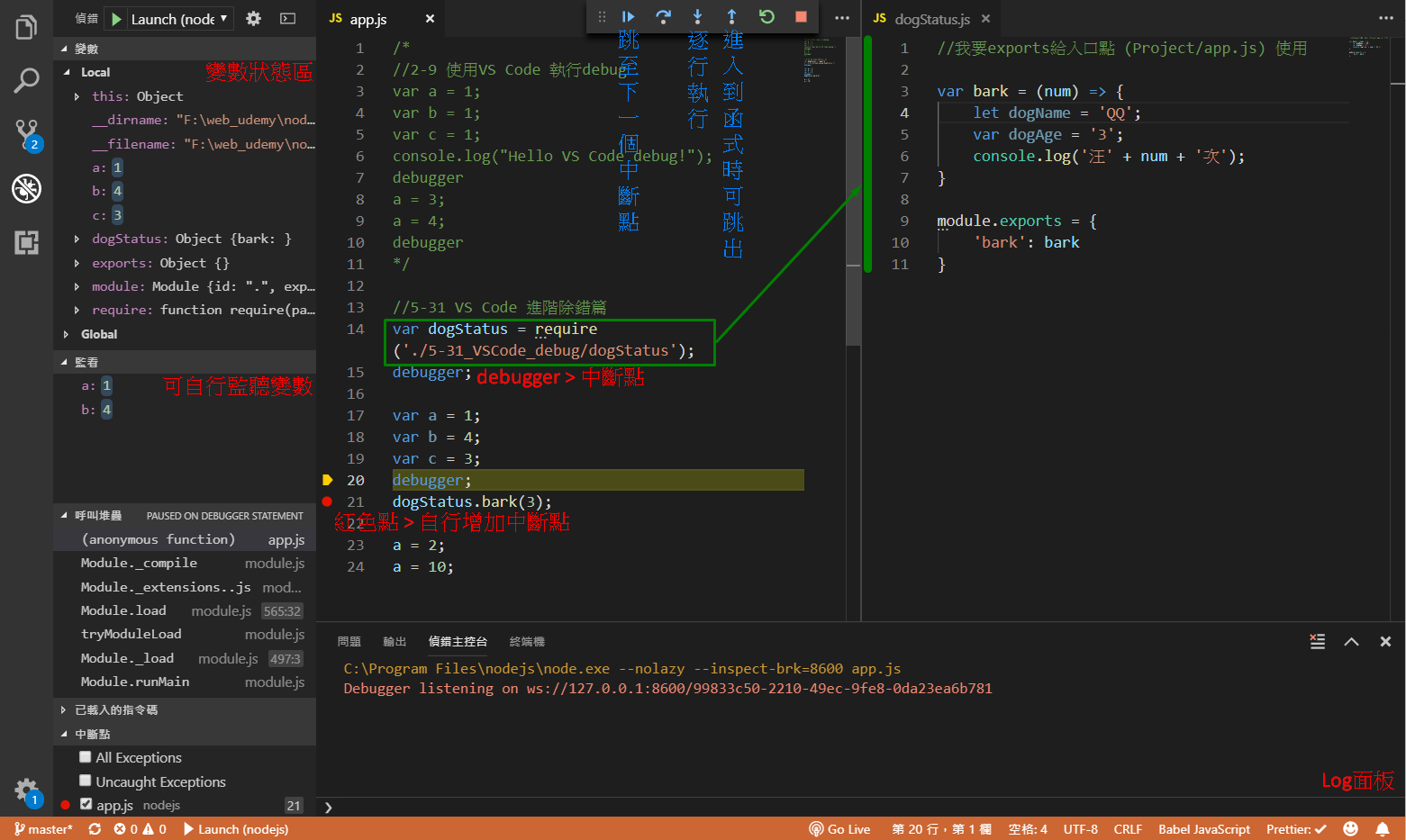
使用Chorme debug nodejs

有點問題，暫未新增內容

## 5-31 VS Code 進階除錯篇

繼2-9 開啟VS Code 方式

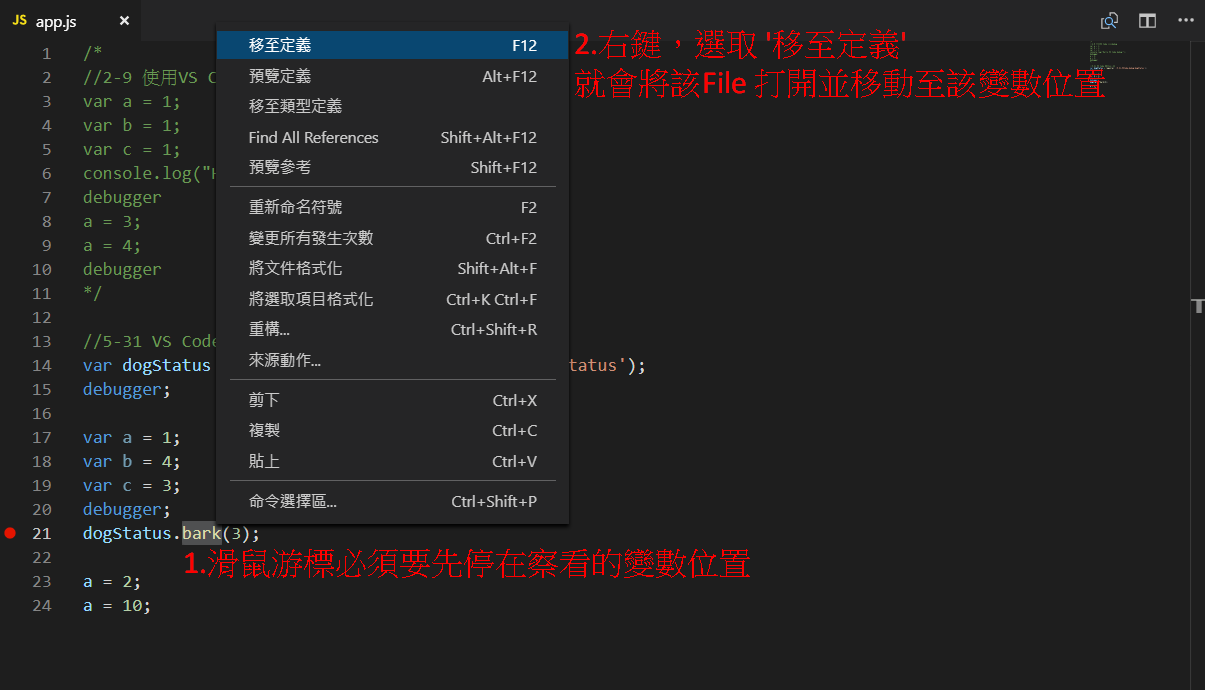
接下來詳細說明VS Code的除錯



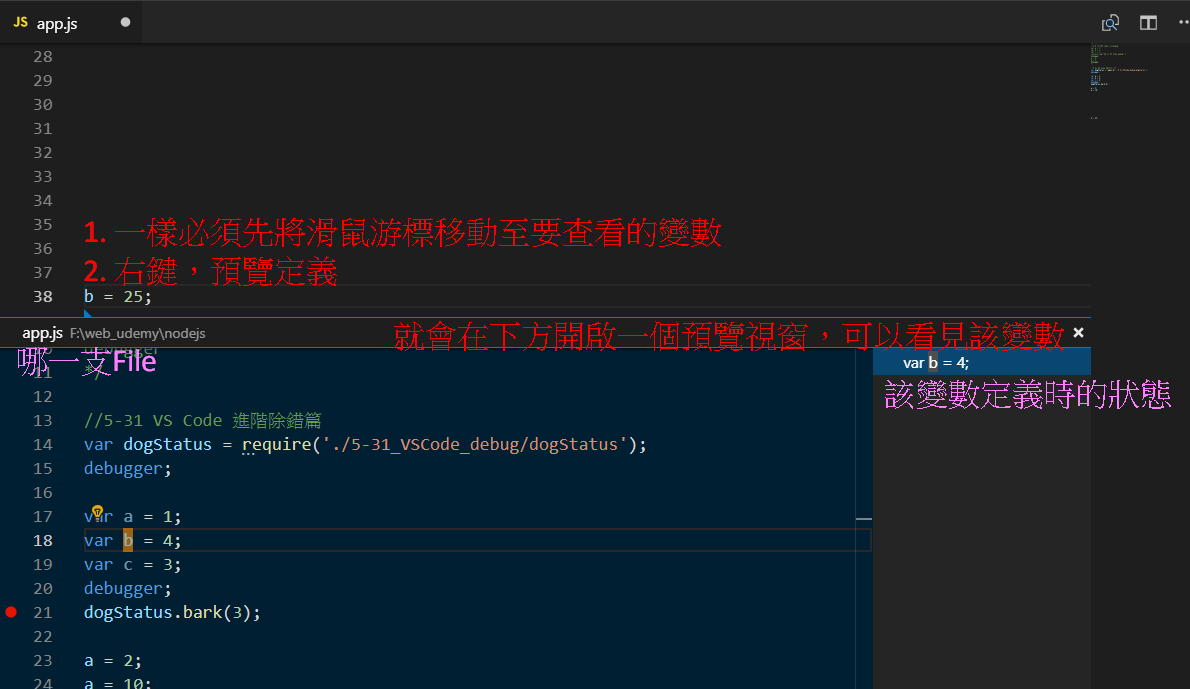
## 5-32 VS Code 定義瀏覽

使用VS Code 的快速瀏覽，可以快速移動到var 的變數位置或開啟其他require file 變數位置

移至定義



預覽定義

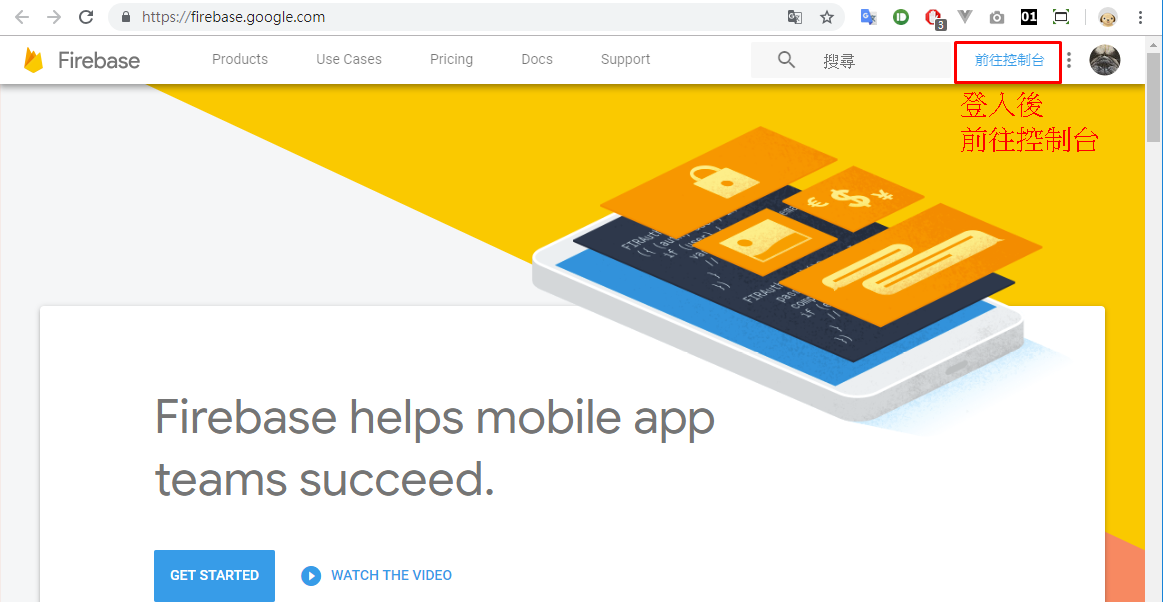


# Firebase 資料庫 - 整合 todolist

## 6-33 Firebase 服務介紹

官方網站: <https://firebase.google.com/>

官方API: <https://firebase.google.com/docs/reference/js/firebase.database.Reference?hl=zh-cn>



## 6-34 環境講解

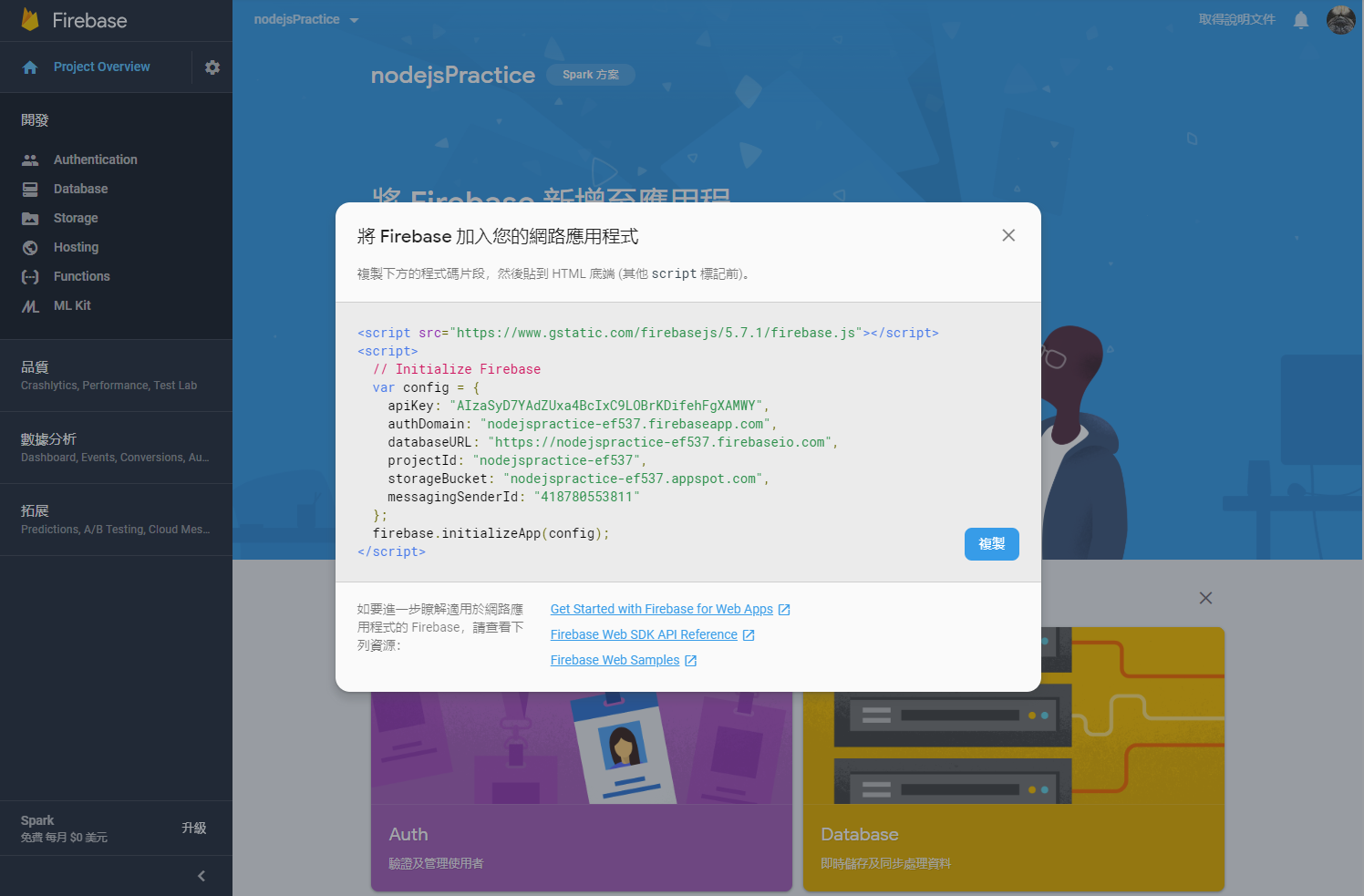
請參考影片

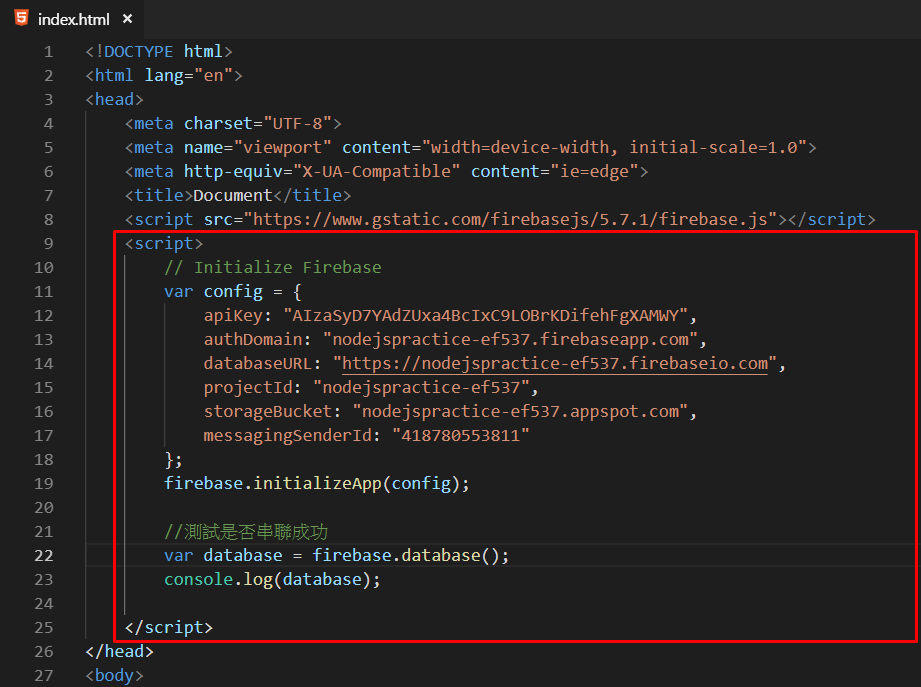
<https://www.udemy.com/teachnodejs/learn/v4/t/lecture/8380900?start=58>

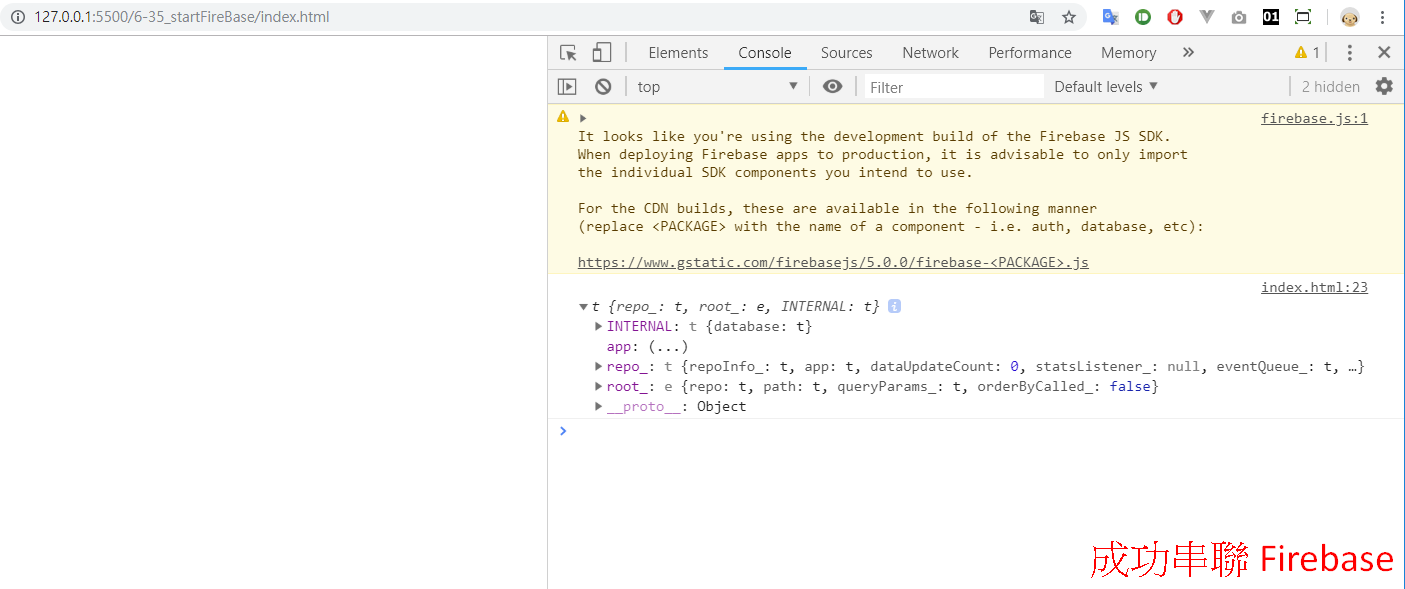
## 6-35 資料庫環境設定

快速將靜態Html串聯 firebase 的 database



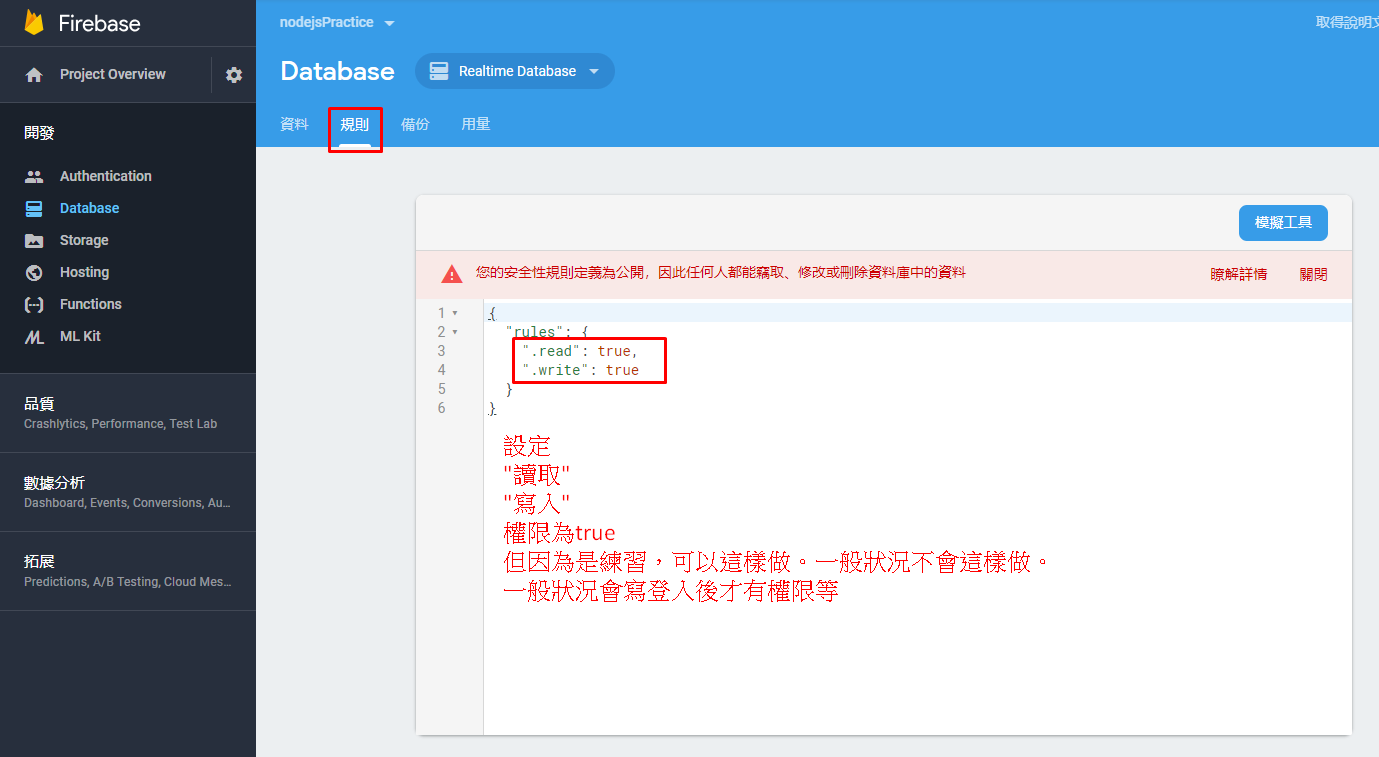


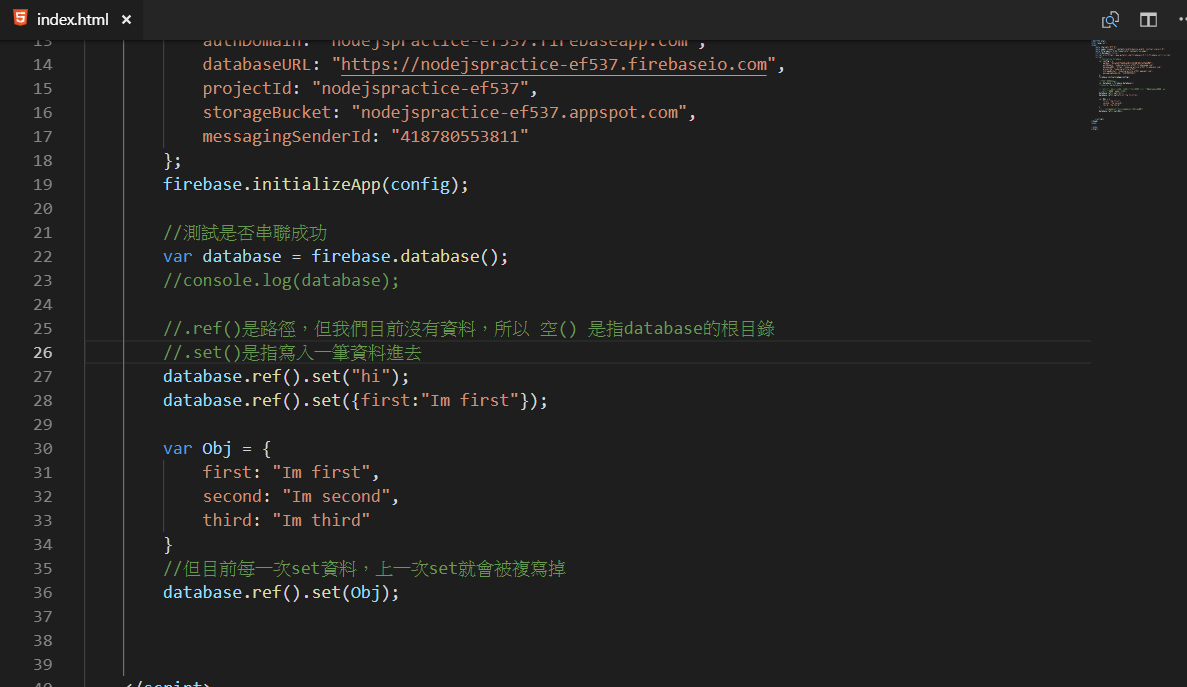


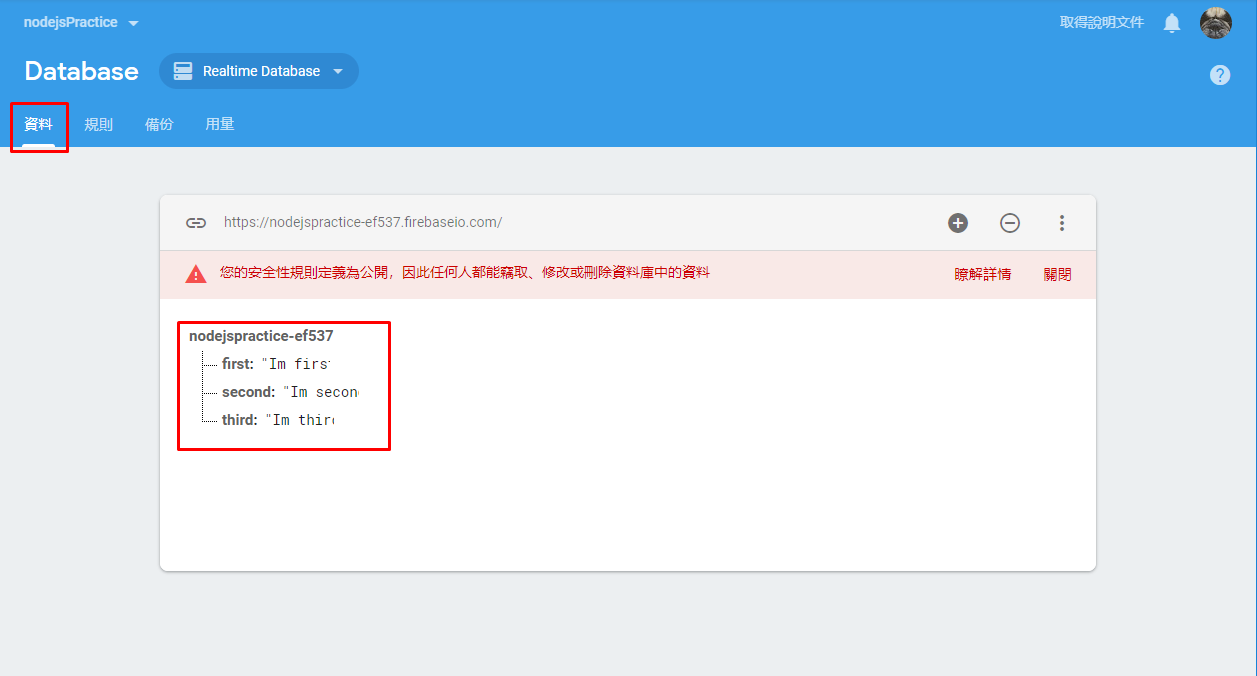


## 6-36 ref (路徑)、set(新增) – 上

從靜態html寫入資料到Firebase

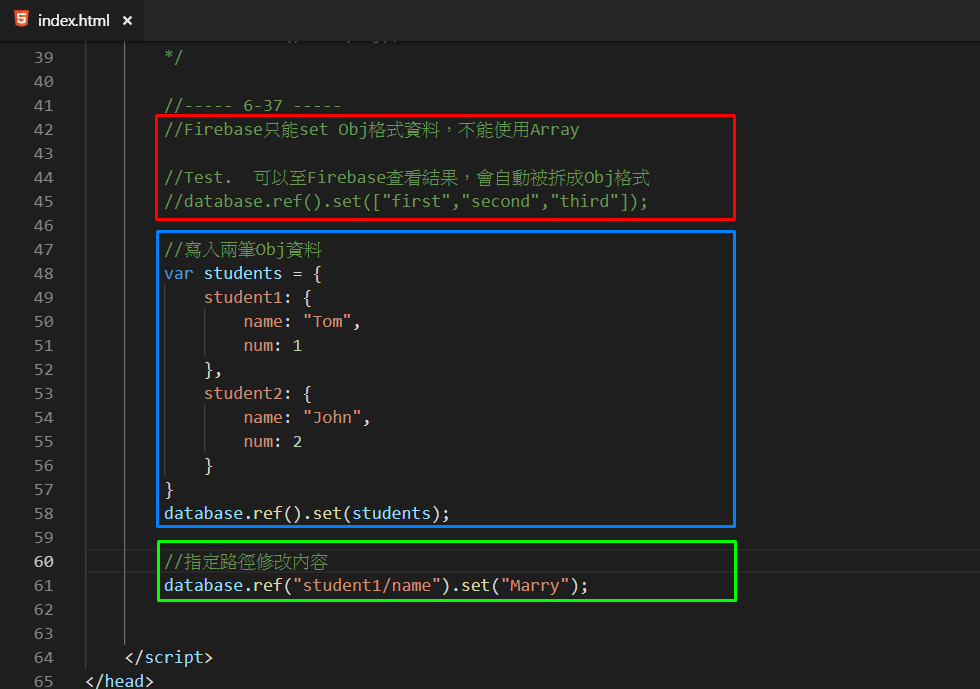


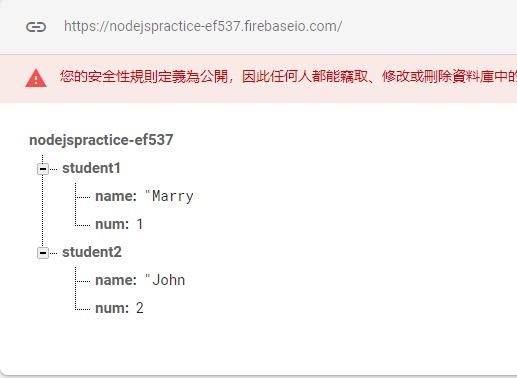




## 6-37 ref (路徑)、set(新增) – 中



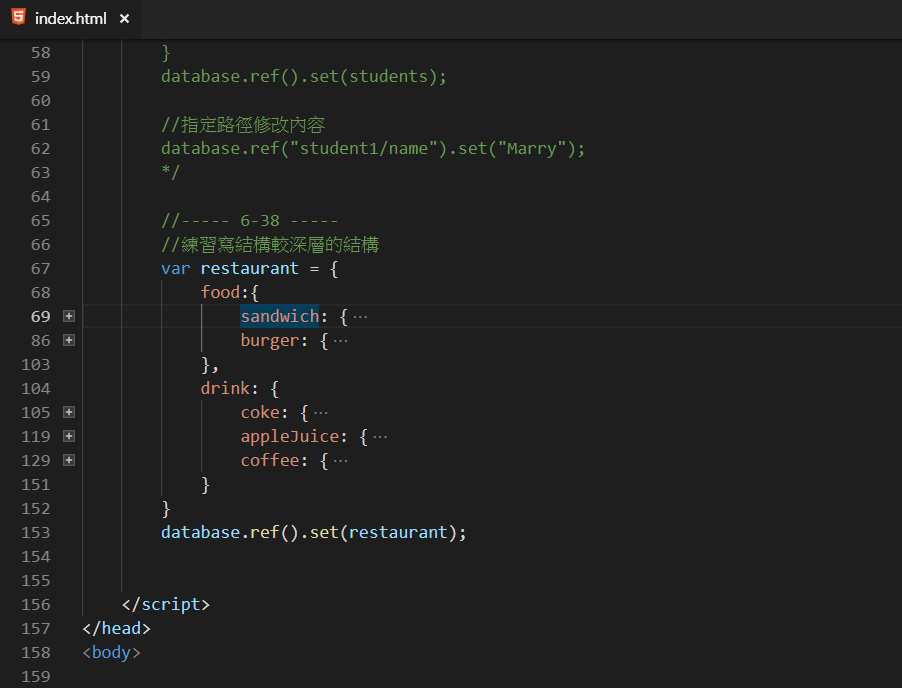




## 6-37 ref (路徑)、set(新增) – 下

練習寫結構較深層的結構

|  |
| --- |
| var restaurant = {  food:{  sandwich: {  beef: {  price: 50  },  chicken: {  price: 40  },  pork: {  price: 40  },  fishMeat: {  price: 50  },  generalSeaFood: {  price: 80  }  },  burger: {  beef: {  price: 50  },  chicken: {  price: 40  },  bacon: {  price: 45  },  fishMeat: {  price: 50  },  generalSeaFood: {  price: 80  }  },  },  drink: {  coke: {  small: {  capacity: "300ml",  price: 20  },  moddle: {  capacity: "600ml",  price: 30  },  large: {  capacity: "1000ml",  price: 45  }  },  appleJuice: {  small: {  capacity: "300ml",  price: 35  },  moddle: {  capacity: "600ml",  price: 60  }  },  coffee: {  americanCoffee: {  small: {  capacity: "200ml",  price: 30  },  moddle: {  capacity: "500ml",  price: 50  }  },  latte: {  small: {  capacity: "200ml",  price: 35  },  moddle: {  capacity: "500ml",  price: 55  }  }  }  }  } |



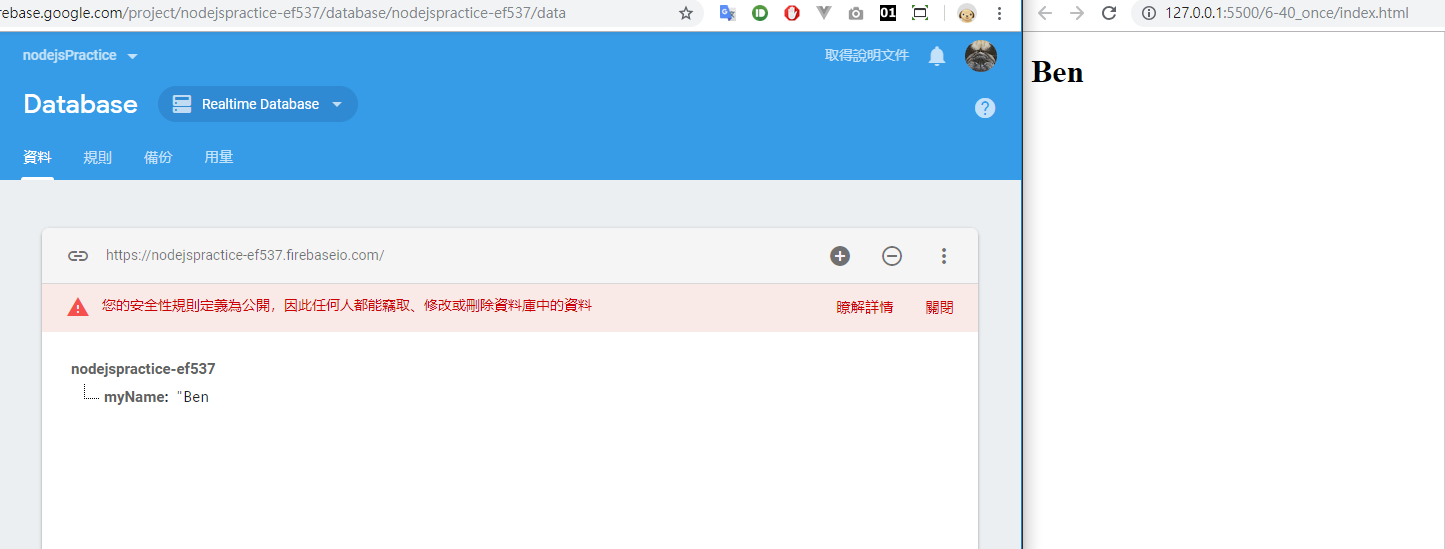


## 6-40 once - 顯示資料到網頁上

.once 讀取一次資料庫的資料

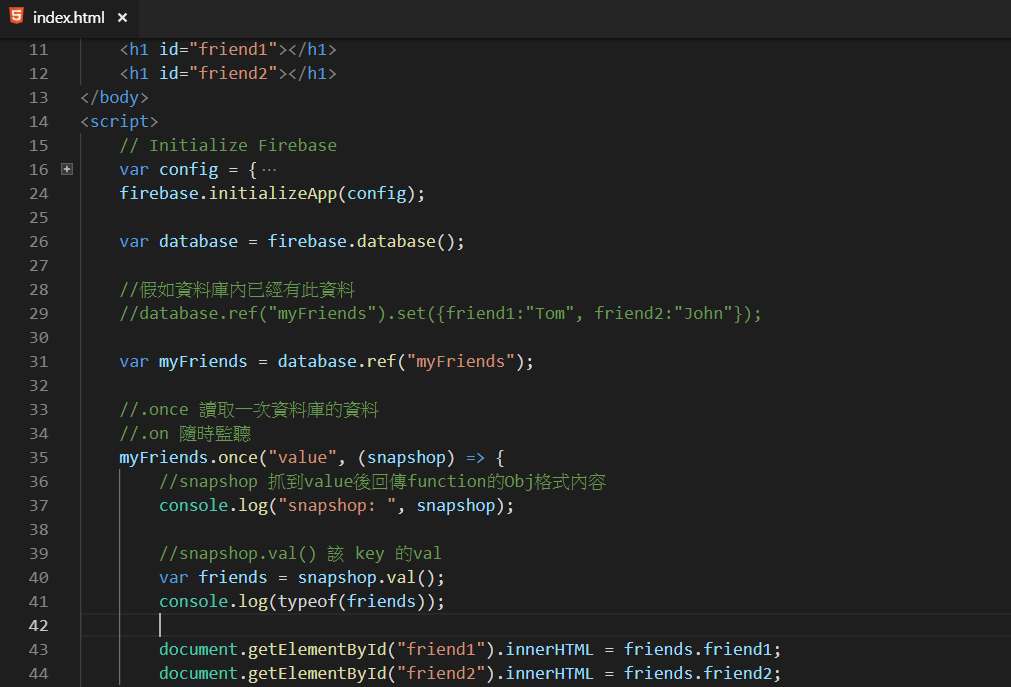
.on 隨時監聽

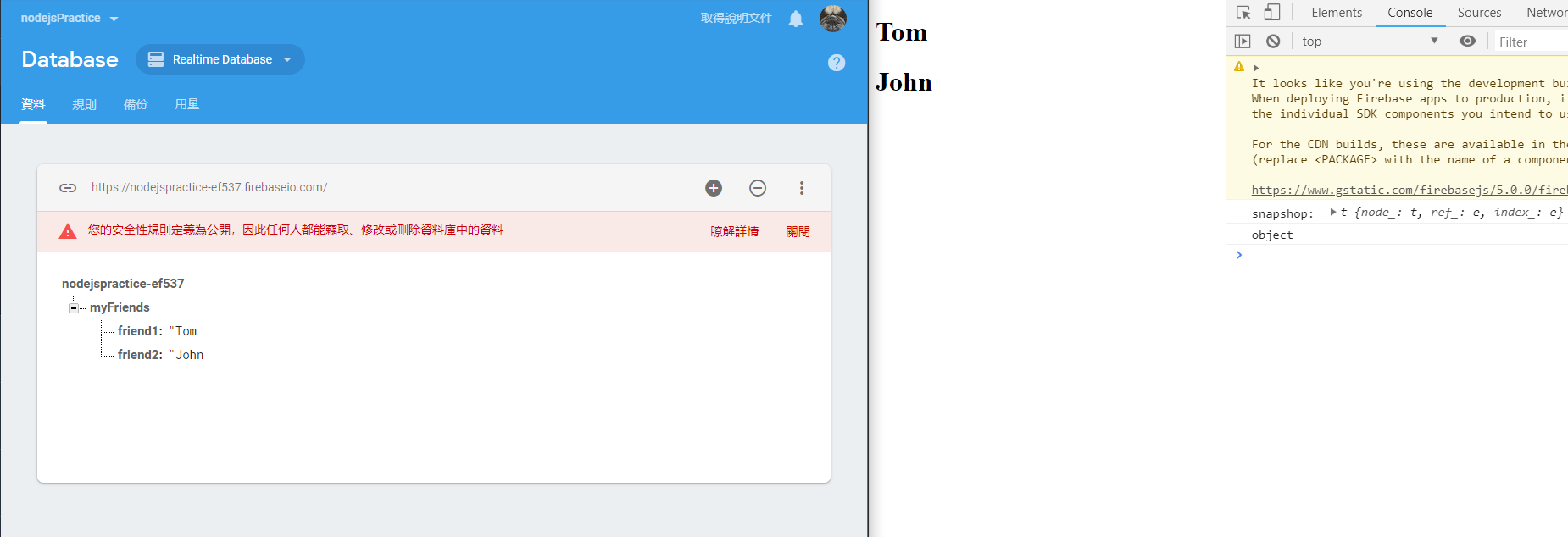




## 6-41 once - 顯示列表資料到網頁上

.once 抓取資料庫Obj格式

其實取剛剛6-40 抓取String一樣，只是回傳變成Obj解析時需要注意

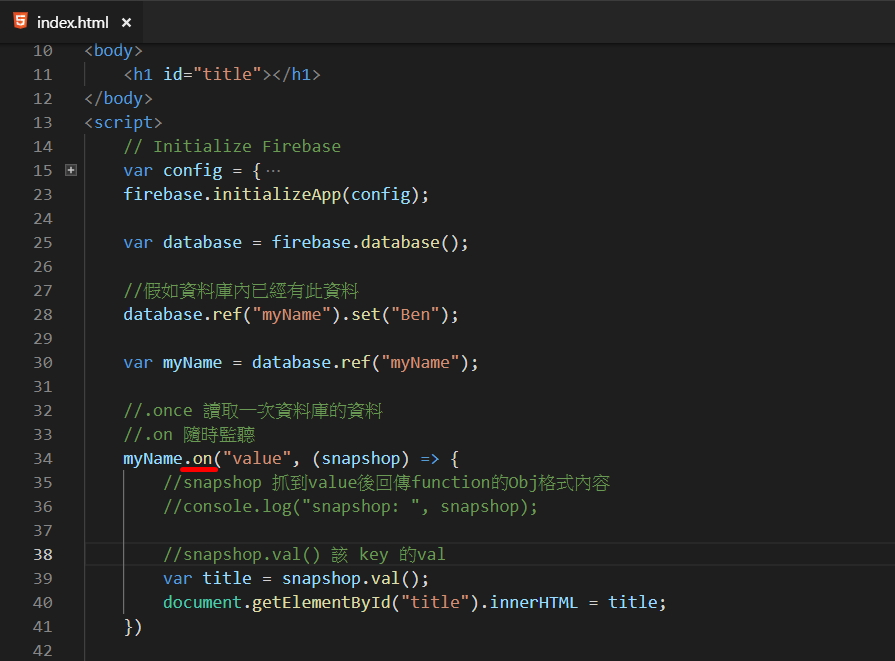


## 6-42 on - 資料即時呈現

.on與.once寫法一樣

只是呈現效果會持續監聽資料庫資料

如果有變更，會即時回傳。適合做例如聊天室這種功能。



## 6-43 firebase 非同步觀念

請參考影片老師說明

<https://www.udemy.com/teachnodejs/learn/v4/t/lecture/8456118?start=217>

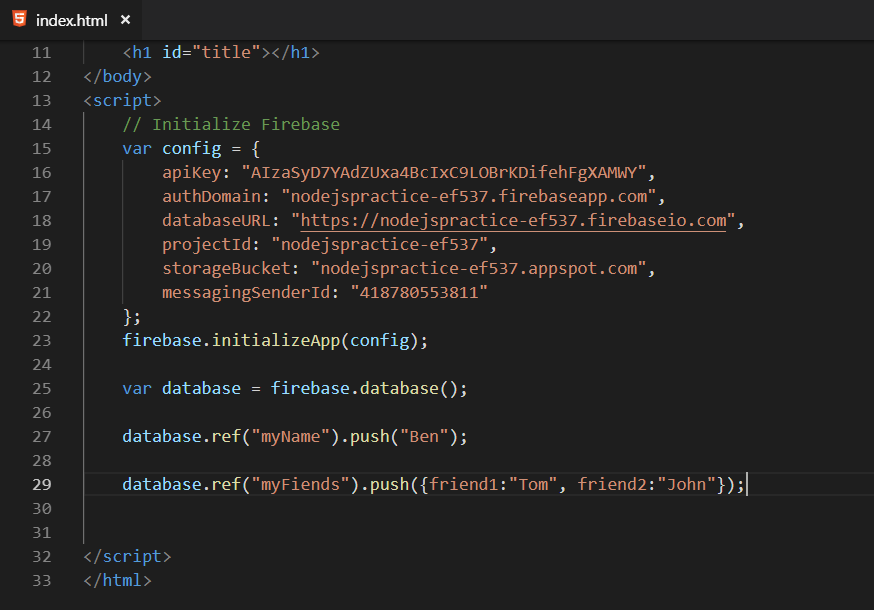
## 6-44 push- 新增資料

.push()跟.set()有點像

都是新增一筆資料至資料庫

.push()有一個特性是只會新增Obj

新增完後會有一個Key為亂數

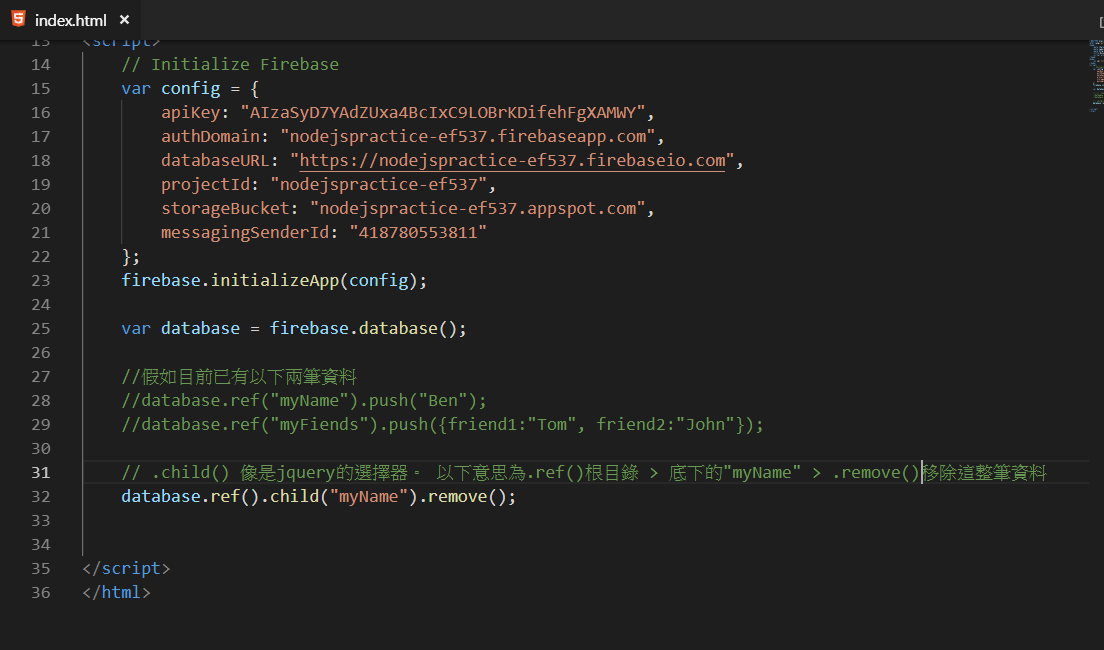


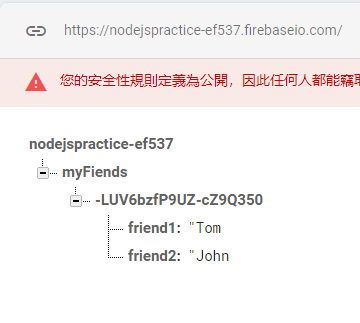


## 6-45 remove、child - 移除資料

.child() 就是jQuery的選擇器。指下一層

.remove() 刪除該筆資料



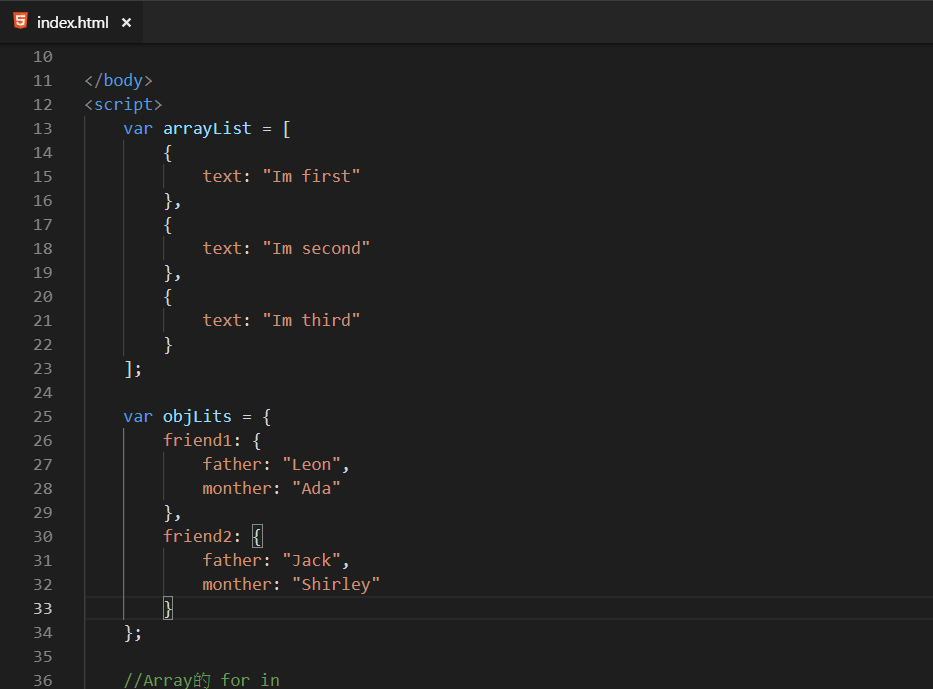


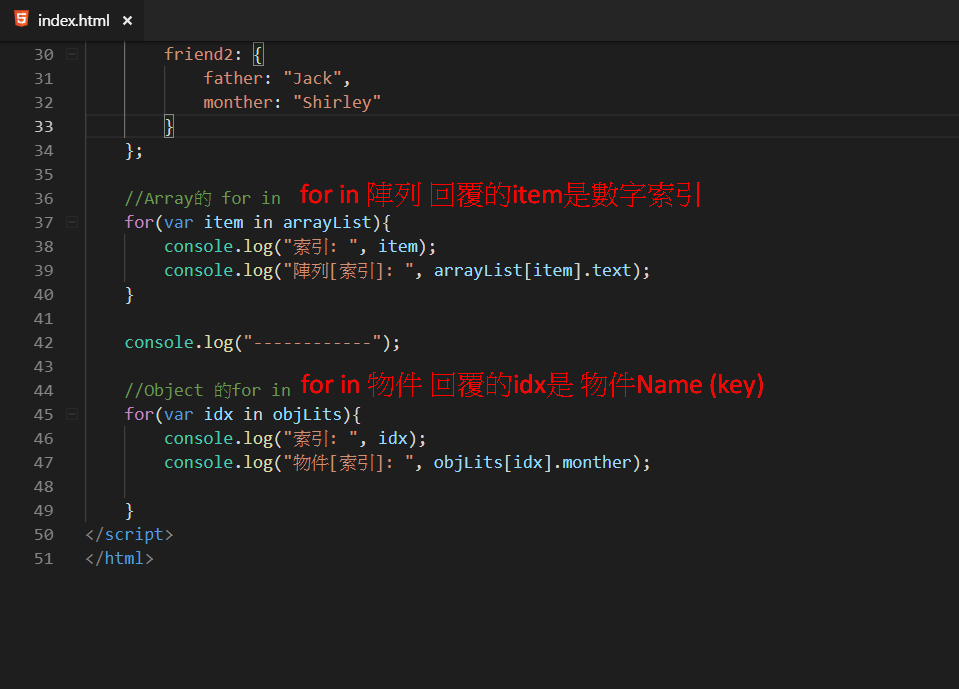
## 6-46 for-in 語法講解

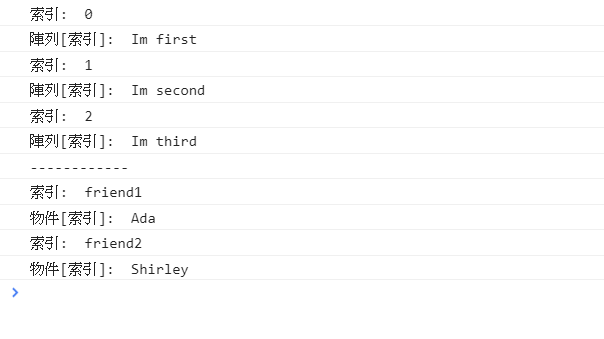
平常比較常使用for迴圈

這次解釋for in

索先規劃一個陣列、一個物件



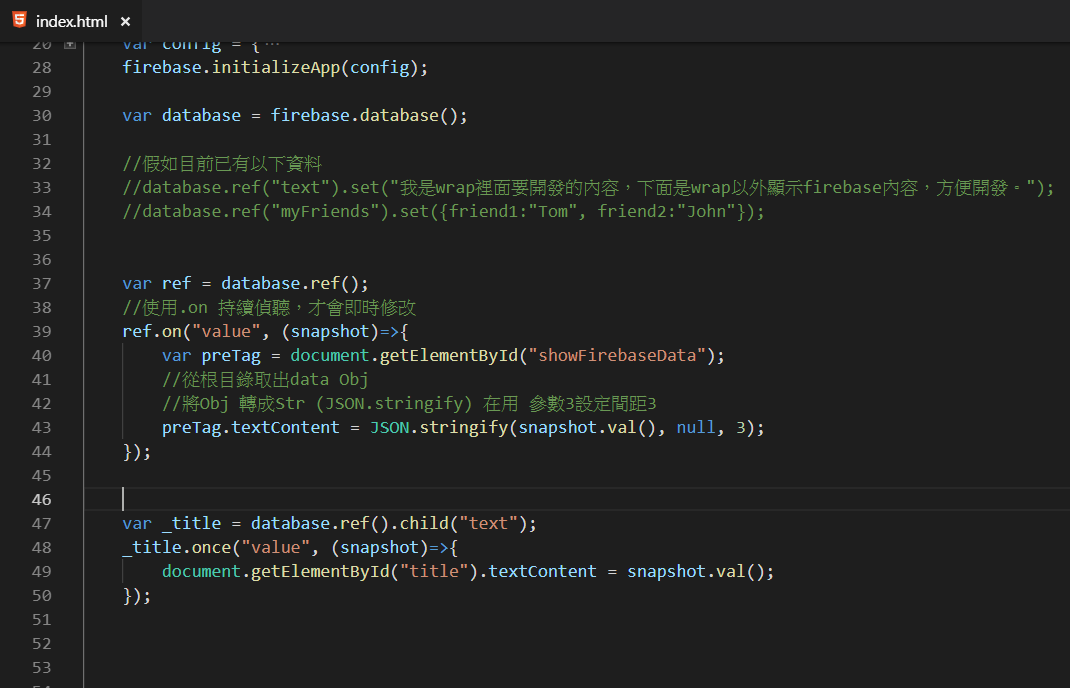


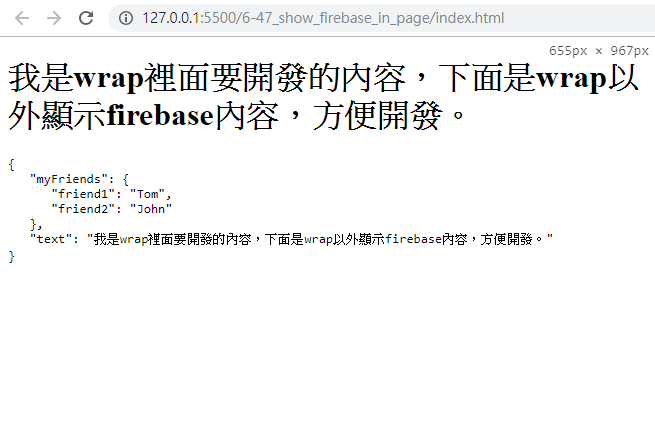


## 6-47 在網頁上即時瀏覽 firebase 資料

如果不想每次都打開Firebase查看目前資料庫狀態

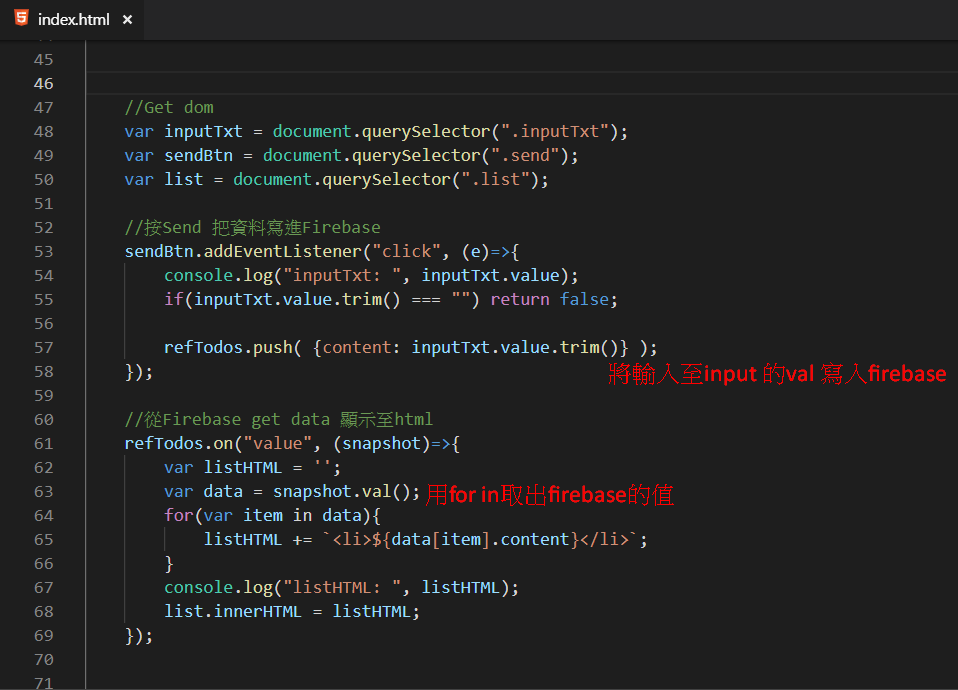
也可以直接顯示在頁面上，開發完畢後再刪除即可。





## 6-48 新增一個 todolist 吧！(上)

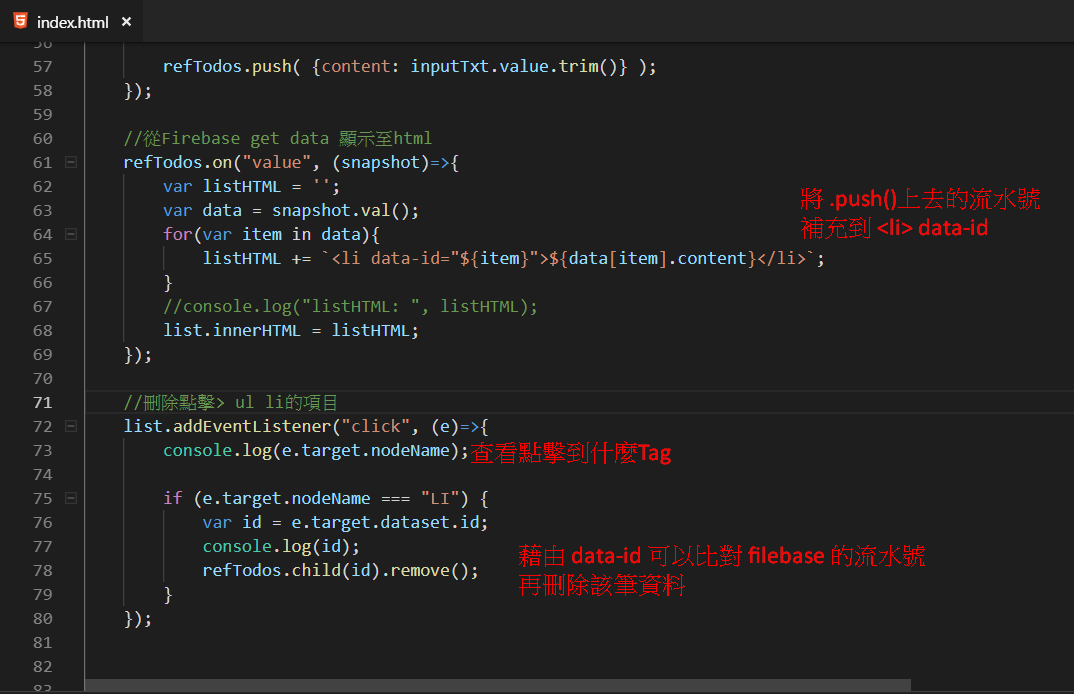
實作一個 <input> 輸入資料, <btn>送出後寫入firebase, 再顯示至前端頁面上。





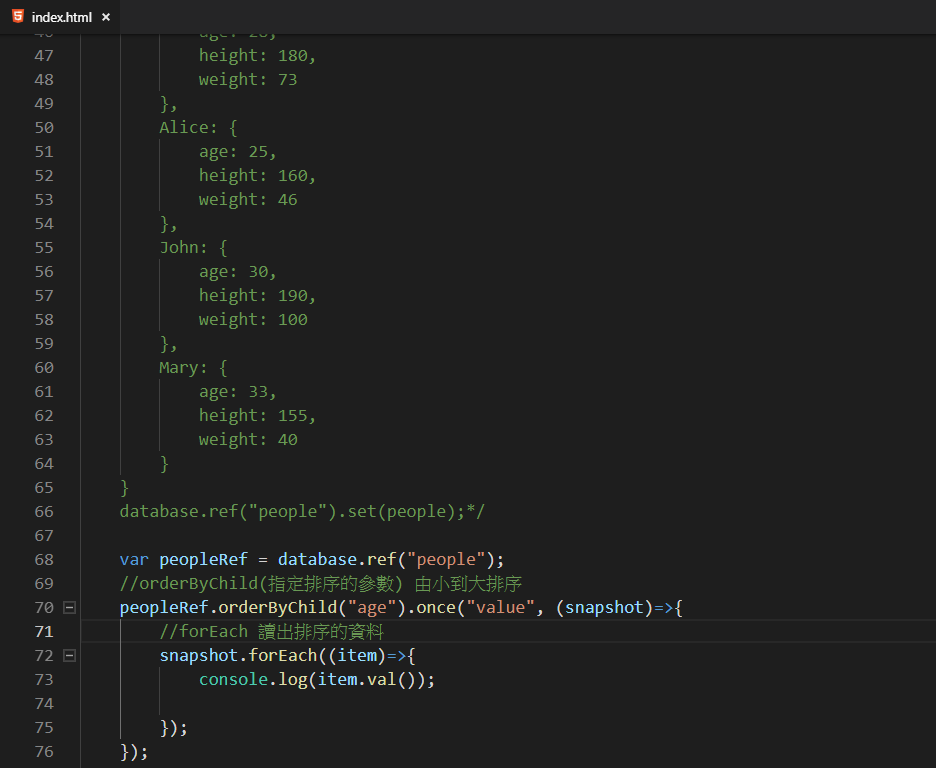
## 6-49 新增一個 todolist 吧！(下)

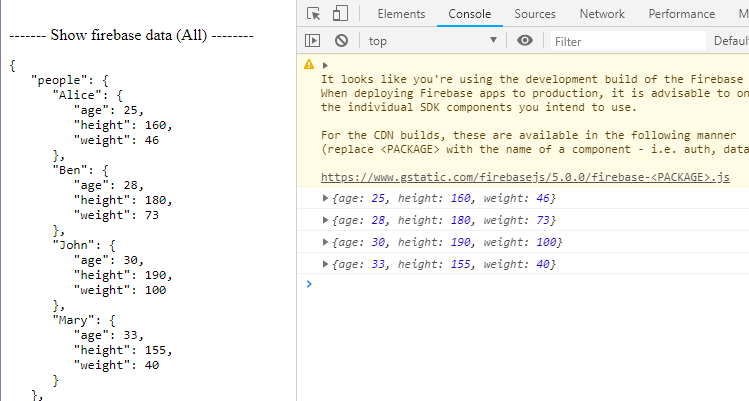
在新增一個點擊li 刪除該筆data



## 6-50 forEach、orderByChild - 資料排序篇

將資料排序，再使用forEach讀出排序後的資料





## 6-51 排序規則

排序規則官方文件:

<https://firebase.google.com/docs/database/admin/retrieve-data?hl=zh-cn#orderbychild>

orderByChild

對於指定的key的value方式是有一定的，由小到大序列為:

Null 或 跟本沒有這個key

False

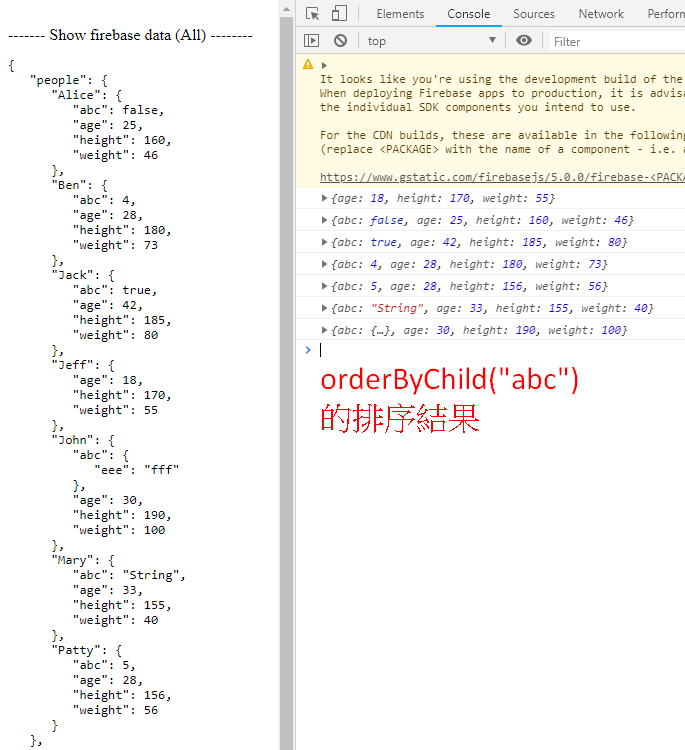
True

數字小

數字大

字串

Object



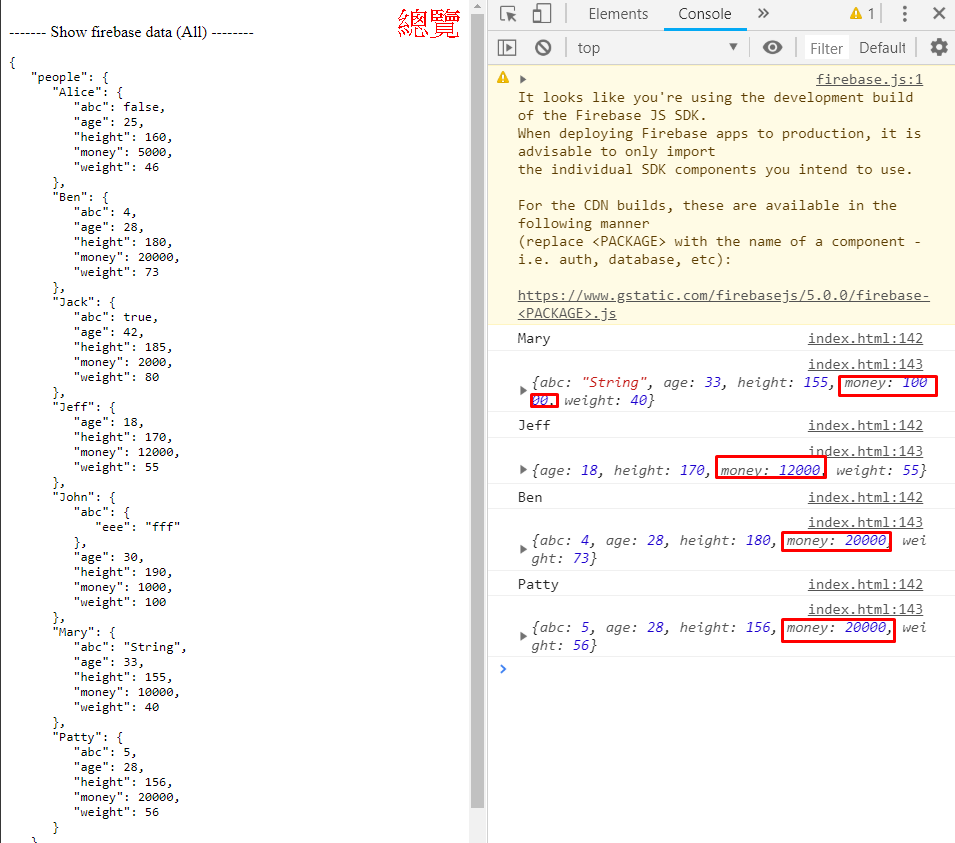
## 6-52 starAt、endAt 搜尋區間規則 – 1

startAt(int) 大於

endAt(int) 小於

equalTo(int & string) 指定值

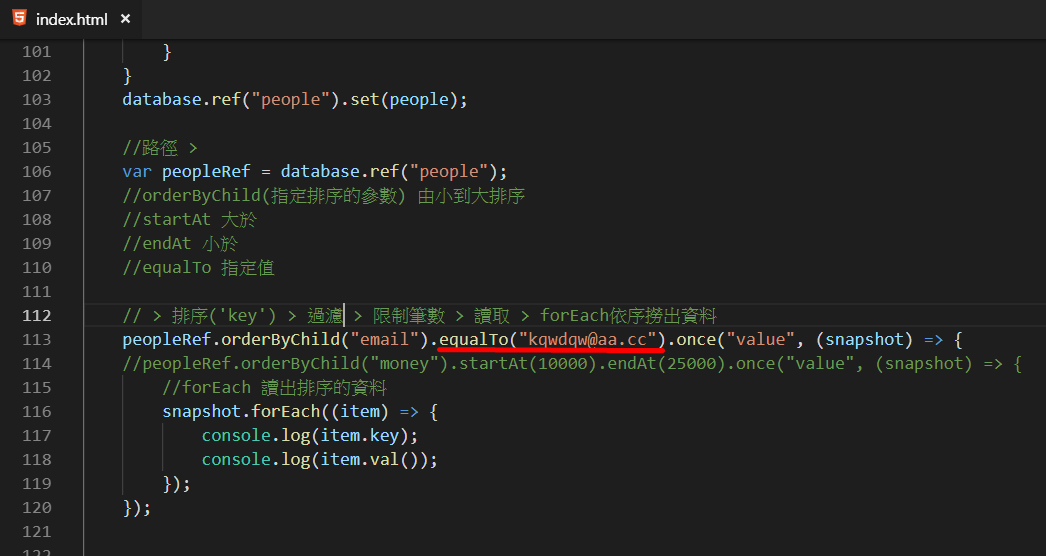


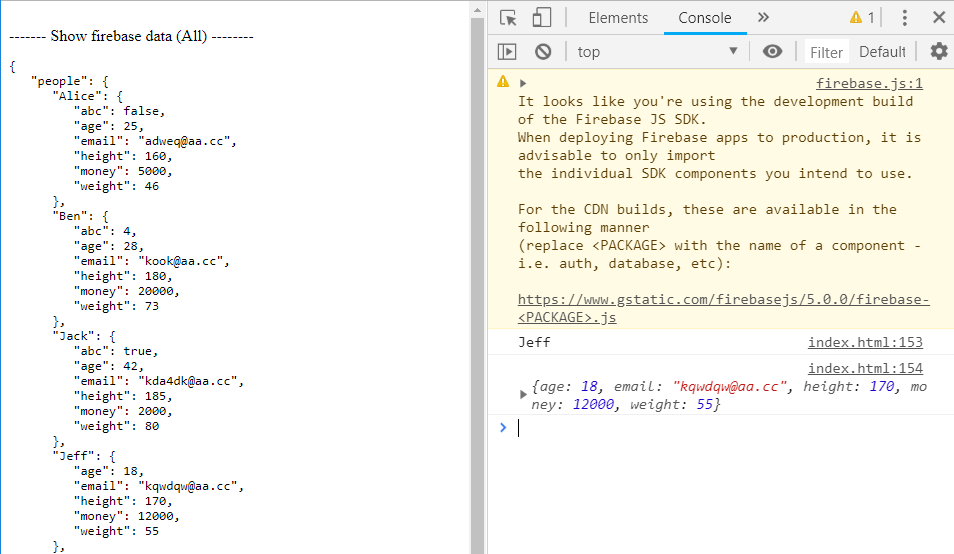


## 6-53 starAt、endAt 搜尋區間規則 – 2

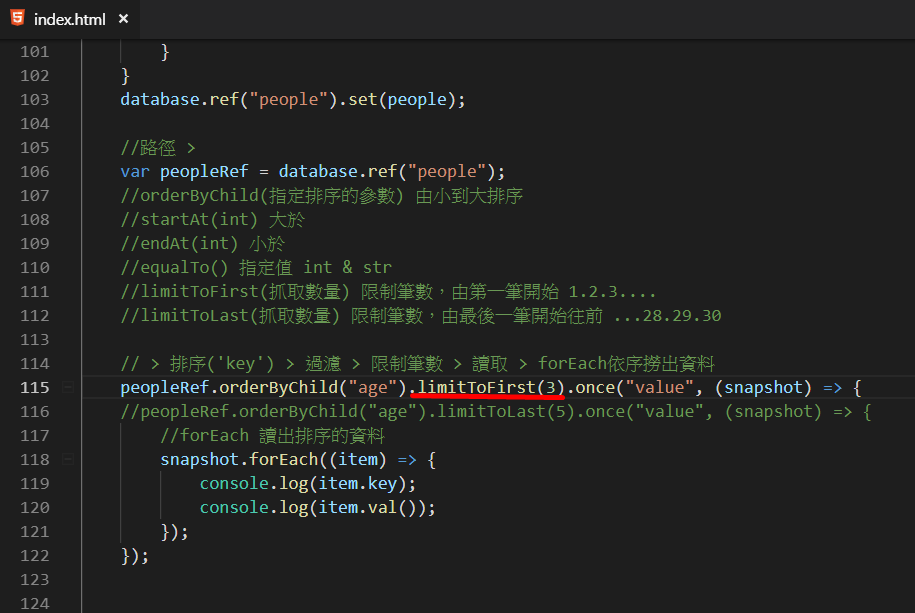
同6-52

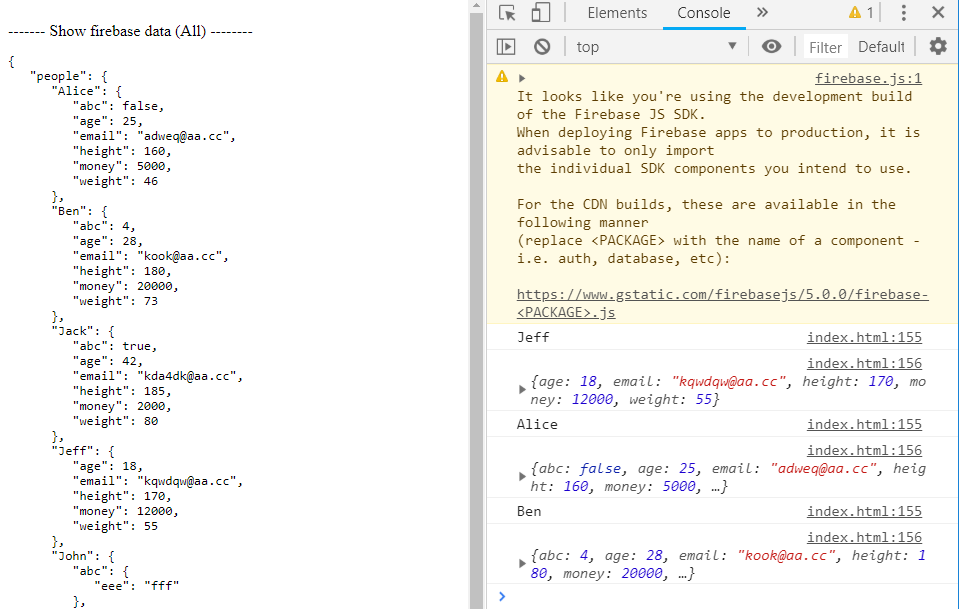
使用 equalTo(int & string) 指定值 撈出string資料





## 6-54 limit 限制筆數

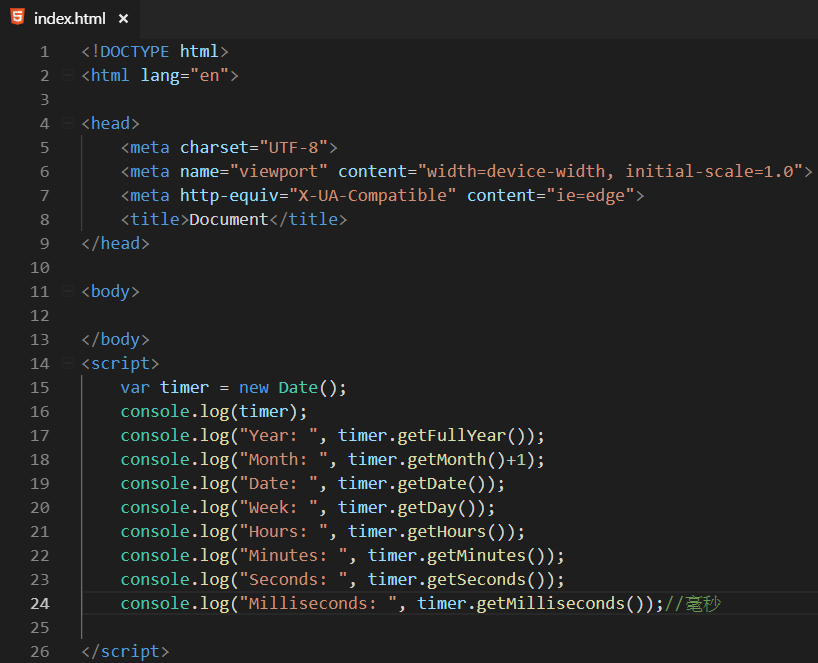


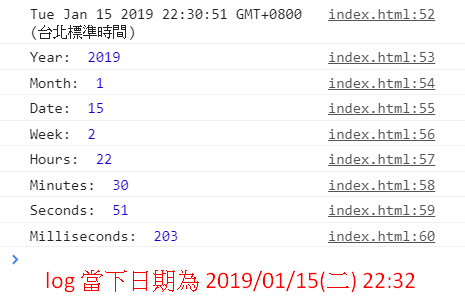


## 6-55 new Date

紀錄到資料路的資訊，最好都要附註上時間資訊

先用原生js抓取時間





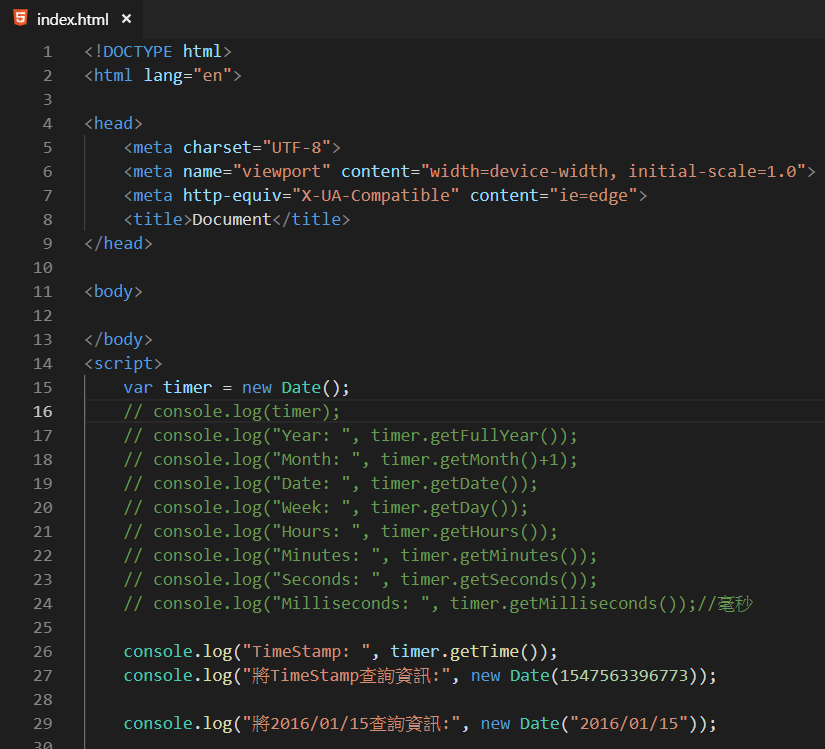
## 6-56 timeStamp 時間戳記

timestamp 為時間戳記，全世界統一的數字

可參考 <https://zh.wikipedia.org/wiki/UNIX%E6%97%B6%E9%97%B4>

從協調世界時1970年1月1日0時0分0秒起至現在的總秒數

有了時間戳記就可以使用js內建語法查詢該時間資訊或比較時間或當流水號





## 6-57 搜尋時間區間

