



Web App 開發基礎

(HTML + CSS + JavaScript + TypeScript + Ionic + Angular + PHP + MySQL)

學習流程

Step 1: 網頁及APP開發基礎概念

Step 2: 網頁開發 (HTML+CSS+JavaScript)

Step 3: 使用Ionic製作手機APP畫面

Step 4: 使用Angular為Ionic開發完整的前端手機APP

Step 5: 以PHP及MySQL開發手機APP的後端

Step 6: 以Ajax駁通手機APP的前後兩端

Step 7: 完成前端處理後端所傳回的數據並更新畫面的機制

前端開發 - 課題

1	Web App及PWA簡介與HTML標籤語言
2	CSS樣式表與程式外觀設計
3	JavaScript程式編寫及應用
4	JSON及本機儲存
5	Ionic簡介及UI製作
6	Angular及Node.js與Ionic應用
7	App及頁面生命週期
8	NPM第三方插件應用
9	App警示元件、Modal及IonSlides範例
10	Tab-based頁面組、客製化元件及Ionic Service應用
11	練習答案

後端開發 - 課題

12	「前端與後端」概念
13	伺服器端配置方案
14	PHP、MySQL與PhpMyAdmin的關係
15	關聯式數據庫(Relational Database)概論
16	SQL語言精讀
17	PHP基礎概念
18	PHP存取數據庫
19	PHP與客戶端互動
20	項目實習-附近商場APP-伺服器端製作流程
21	前後端互動實作

前端開發

課題一：

Web App及PWA簡介與HTML標籤語言

甚麼是 Web APP



- ◎ Web App，即 Web Application（網路應用程式），是一種透過網路（如網際網路或內部網路）在瀏覽器上執行的應用程式。與傳統的桌面應用程式不同，Web App 不需要在使用者的電腦或行動裝置上安裝，只需透過網路連接到對應的伺服器即可使用。
- ◎ Web App 可用於各種應用場景，如電子商務（電子商店）、社交媒體平台、線上辦公室套件、郵件服務和教育平台等。隨著 Web 技術的發展，例如 HTML5、CSS3 和 JavaScript 的進步，Web App 的效能和功能已經越來越接近傳統的桌面應用程式。

Web APP 的好處

- ◎ 用途極廣：可被製作出網頁、手機應用程式甚至傳統的電腦軟件
- ◎ 跨平台：只需開發一個版本，iOS、Android、MacOS、Windows、甚至Linux/Unix皆可在不需加裝外掛的情況下完美執行
- ◎ 開發成本相對低：正因跨平台的特性，開發者只需製作一個版本便足夠，無需像傳統方法般為每個平台都開發一次(如以Swift為iOS開發一次，又以Java為Android開發一次，再以C#為Windows開發一次)
- ◎ 學習門檻低：HTML只屬於標籤語言，簡單易學；CSS更只是樣式表，容易理解；而且JavaScript及TypeScript強大而比舊版程式語言如Java、C#與Objective-C等的語法相對較不嚴謹且結構較為寬鬆，易於學習
- ◎ 次世代手機APP：PWA更被Apple和Google支援並可繞過App Store「上架」而無需經過審批，用戶可透過連結直接使用或安裝到手機裡待往後使用皆可

甚麼是PWA



漸進式網絡應用程式（英語：Progressive Web Apps，簡稱：PWA）是一種普通網頁或網站架構起來的網絡應用程式，但它可以以傳統應用程式或原生流動應用程式形式展示給用戶。這種應用程式形態視圖將目前最為現代化的瀏覽器提供的功能與流動裝置的體驗優勢相結合。

簡單地說，PWA就是寫得很像一隻手機APP的網頁。

PWA特點

- **Progressive**: 漸進式，提供每一位用戶做基本的瀏覽，若支援環境、可以提供更強大的功能。
- **Responsive**: 響應式的用戶介面，可以在不同裝置下作最佳化的顯示。
- **Connectivity independent**: 不依賴網路連接，透過 service workers 可以在低頻寬甚至是離線的環境下瀏覽網站。
- **App-like**: 讓網站可以具有像 APP 般的瀏覽速度等優點，提供更佳的用戶體驗。
- **Fresh**: 藉由 service worker 自動更新網站內容。
- **Safe**: 通過 HTTPS 來提供服務，確保資料安全性。
- **Discoverable**: 能夠執行 SEO 優化，讓使用者快速找到網站，增加網站經營成效。
- **Re-engageable**: 像是推播通知等特性，吸引使用者注意、主動和使用者互動，提升用戶回流率。
- **Installable**: 可以藉由 Add To Home，如同 App，會新增一個 icon，可以直接將網站加到手機桌面上做切換使用，不需要再透過 App Store 下載安裝。
- **Linkable**: 可以將網站、透過 URL 相互傳遞分享。

分享：如何完成一隻APP

1. 了解清楚客戶(或自己)究竟想做些甚麼、目的或價值

2. 列出一些重要功能並整理出APP有甚麼的部分或模組
(如：會員系統模組、付款及交易模組、產品瀏覽模組……)

3. 用紙筆簡單畫下大約有甚麼頁面，頁面中有甚麼元件和按鈕，
按下按鈕後會跳到哪一頁或發生甚麼事情……

4. 製作前端(一個只有UI畫面的Demo App)

5. 製作後端(設計數據庫及完成開發App)

如何接JOB ?

Free Hunter (香港)

<https://freehunter.hk>

Hello Toby (香港)

<https://www.hellotoby.com>

Freelancer (全球)

<https://freelancer.com>

Fiverr (全球)

<https://fiverr.com>

接JOB注意事項 (1)

合約清楚列明修改次數及範圍與交付條件，更可先收上期費用才開工，亦可增設Consultation Fee，並注意保養或維護限期

收到尾數後才開通網站的所有功能 /
設試用版，收到尾數後才開通正式版

申請SSL安全證書可能需要額外收費，
可於接Job前事先告知客戶，並向客戶收取額外申請費用

注意伺服器或網存的服務器(Server)存放的位置，一般而言，應選擇位於客戶及其客戶的地區附近

接JOB注意事項 (2)

一個Job應有至少一萬至數萬元(HKD)工酬，
「十零廿萬」其實都是「間間地」很常見的

APP類型大致有3種:

1. 靜態 (APP不需與伺服器互動)
2. 動態 (APP需與伺服器互動)
3. 動態而需即時通訊 (需設有Socket伺服器)

開發員上大致有3種:

1. 前台(客戶端)開發員或設計師
2. 後台(伺服器端)開發員
3. 全端開發員 (前台及後台都做)

網上下載資源

Shutterstock

<https://www.shutterstock.com>

iStock

<https://www.istockphoto.com/hk>

Adobe Stock

<https://stock.adobe.com>

Pixabay

<https://pixabay.com>

製作網站時需要到圖片/影片而客戶又未能提供，不需自己拍自己畫的，也未必需要付款找設計師攝影師幫忙，可到這些網站下載合適的插圖、相片或影片，甚至使用生成式AI服務協助製作多媒體資源。

網上資源注意事項

版權©，以Template來說，通常每個Project一個License

圖片/影片如果是“Editorial Use Only”則不能作商業用途

<https://support.shutterstock.com/s/article/Why-is-some-content-marked-Editorial-Use-Only>

有些版權©為Subscription-based，在訂閱期間可任下載任用，
但訂閱有效期過後所有項目都不得再繼續使用該些資源

購買及使用前睇清楚，免得自己或客戶往後給人追究，建議尊重知識產權，商業用途，應更加小心

寫APP所需硬件資源

iPhone/iPad + Android機

電腦

(如需開發iPhone/iPad App則必需要Mac)

伺服器

(靜態類APP不需要)

可網上租借

APP(與WEB)的兩大種類

靜態 (Static)

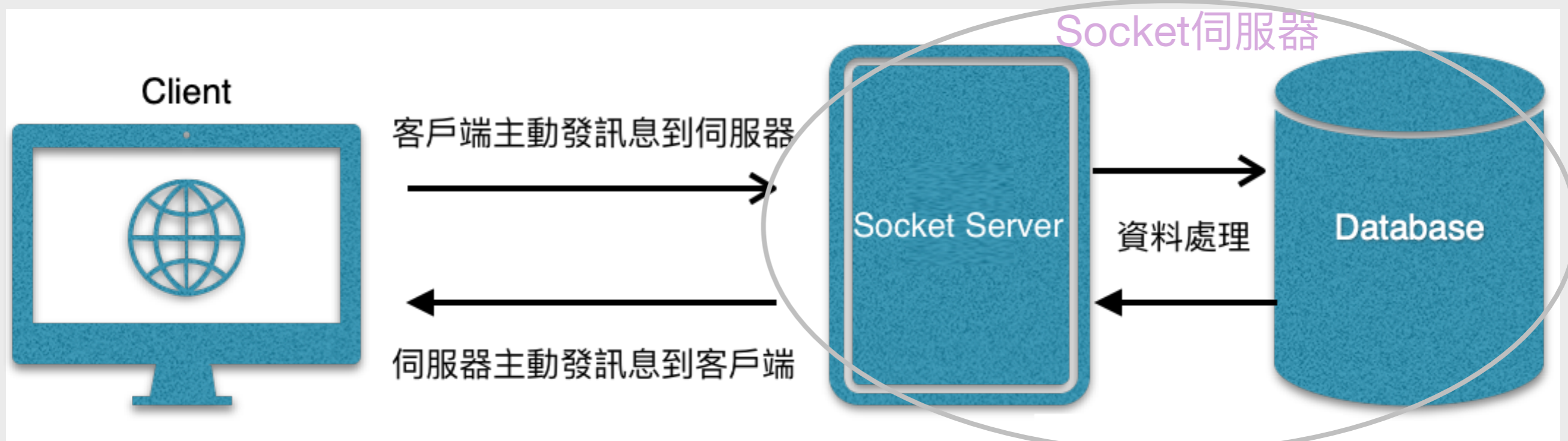
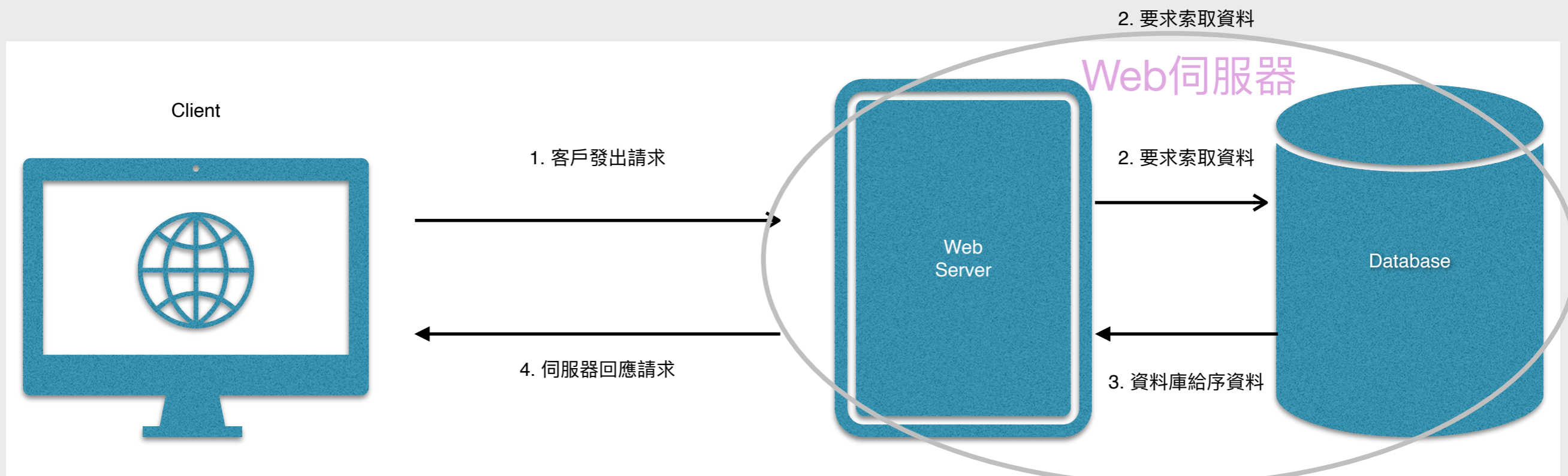
1. APP不需要互聯網也可完整地使用
2. 不需要設置伺服器也可完整地使用
3. (通常)APP單向地向用戶提供資訊
4. 例子：電子書、計算機APP、相簿APP、字典APP

動態 (Dynamic)

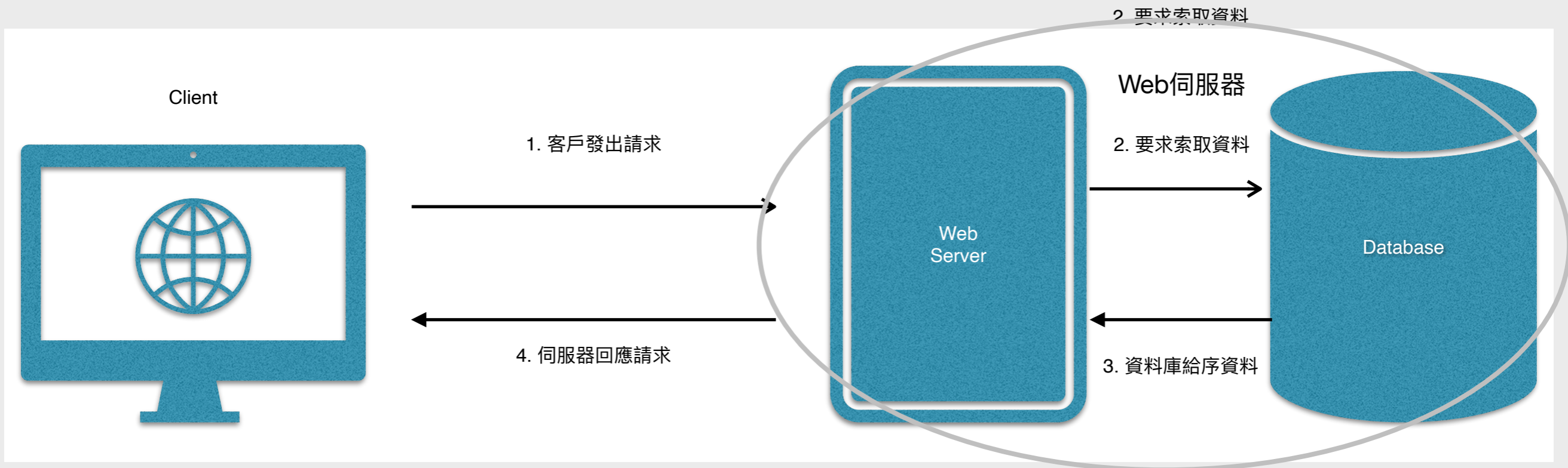
1. APP需要互聯網才能完整地使用
2. 需要設置伺服器才能完整地使
3. APP與用戶有雙向的互動
4. 例子：FB/IG、Wts/TG/WC、VIDA、印花優惠卷APP、巴士APP

動態靜態並不是指APP有沒有視覺上的觀感效果，
而是指APP是否需要與互聯網(自己的伺服器)溝通。

「前端/客戶端」與「後端/伺服器端」概念



WEB伺服器

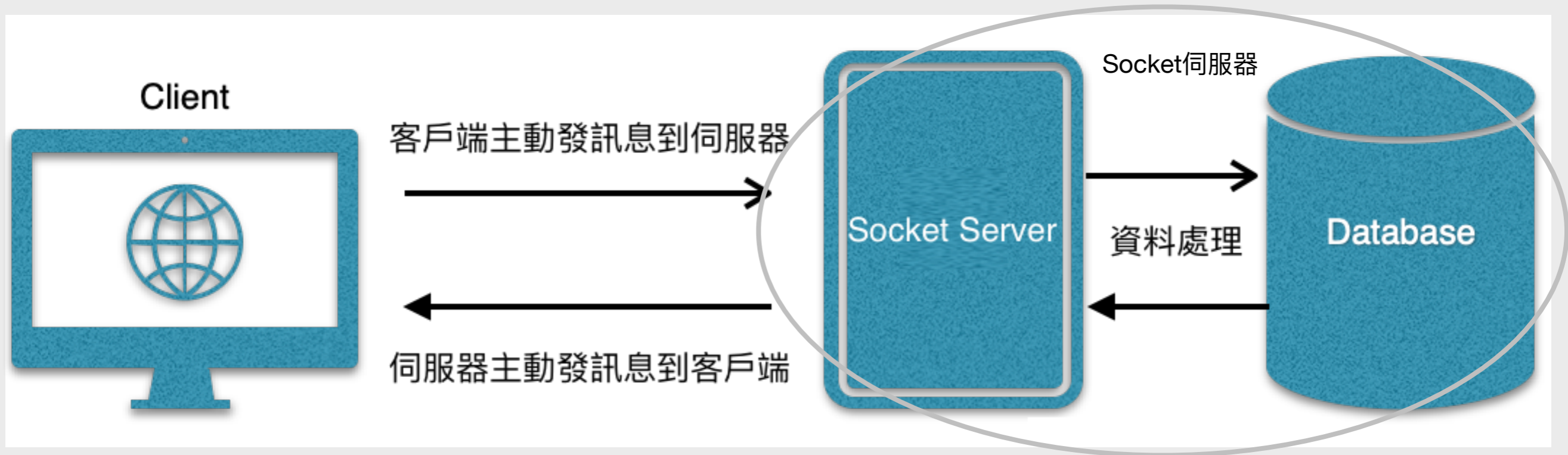


客戶主動發出「請求」來叫喚Server工作

Server並不會主動叫喚客戶端，只能在客戶發請求時作出回覆

仿如幫襯雪糕店，客人主動發出請求要雪糕，店員弄好後給予客人

Socket伺服器



客戶端或伺服器端皆可主動叫喚對方工作

適用於即時通訊、線上多人遊戲、推送通知功能

仿如郵局可以主動把郵件送給你，你也可以主動把郵件送到郵局

寫APP與寫GAME的分別

App

基於Template或客製開發、
使用網上寫apps軟件製作

編程: Native、Web、Hybrid
軟件: App Inventor、Thunkable

Game

使用Game Engines製作為主
(即建基於他方遊戲開發軟件而開發)

如: Unity、Unreal、Construct 3、
Game Marker、GameSalad

上載Web APP到互聯網

當製作好或希望在真實網絡世界中測試網站成品時，我們便需要上載網站到互聯網。以下為一般的流程：

購買Domain域名

購買Web Hosting網存空間或VPS虛擬私人伺服器

購買SSL安全證書

下載FTP軟件並上載網站到網存空間或伺服器

APP上架

iOS

Apple收取HKD780年費

若有賺(「付費App」及「App內購買」)，
Apple抽佣15% (年營收100萬美元以下的企業) ，
數碼產品需以「App內購買」形式售賣

Android

Google收取HKD200一次性費用

若有賺(「付費App」及「App內購買」)，
Google抽佣15% (年營收100萬美元以下的企業) ，
數碼產品需以「App內購買」形式售賣

Web

自找伺服器存放及運行前後台程式，以Web App或PWA型式「上架」

購買Domain域名

環速網存 Hosting Speed - 您的域名管理專家：提供全自動化網頁寄存 Web Hosting 及即時域名註冊 Domain Name Registration

環速網存7x24自動網域註冊平台
全港首創10分鐘內即時生效之網域登記服務!!

OFCA SBO LICENSE: 1397
.hk ACCREDITED REGISTRAR
15+ years caring company

主頁 公司資訊 網頁寄存 域名服務 電郵服務 雲端伺服器 Office365 其他服務 付款方式 即時申請 聯絡我們

客戶服務中心
Customer Service Center

登記電郵地址或登入名稱
密碼
登入 忘記密碼

我們的服務

- 網頁寄存服務
 - 多功能網頁寄存
 - DIY 網頁設計
 - WordPress 寄存
 - Cloud Apps 程式寄存
 - Node.js 寄存
 - Python 寄存
 - ASP.NET 寄存
- 域名服務
 - 註冊域名
 - .hk/.香港 域名專區
 - 轉移域名

多功能網頁寄存

環速網存特點

- > 自設自動化域名註冊及網頁寄存系統
- > 自設即時域名及網頁寄存管理平台
- > 由申請,付款至服務開啓只需10分鐘!

Hosting Speed Platform
Your domain work within 10 minutes!!

cPanel網存控制系統一小時特快安裝
網存控制系統最快十分鐘安裝完成!

- cPanel控制台Demo
- cPanel示範片段

五重垃圾電郵過濾方案
SpamAssassin垃圾郵件智能監別系統,有效減少垃圾電郵,而且用戶更能節省電郵空間及下載垃圾電郵的時間。

高速、安全、可靠設備
為確保達到99.95%/99.9%在線保證,所有網存主機均採用多台高階Quad Core Xeon伺服器高階處理器及RAID硬盤設定。

Hosting Speed

<https://hostingspeed.net>

HostGator

Call 24/7/365 at (866) 96-GATOR
Sign In → Live Chat →

Hosting PRO Hosting Domains Support Affiliates

Shared Hosting

Hosting for every website.

Unlimited storage, unmetered bandwidth, unbeatable hosting. This gator's got ya covered.

And we'll throw in a **free domain** for a year, too.

We Recommend

Hatchling Plan Now 60% off! Single website One-click WordPress installs Free WordPress/cPanel website transfer	Baby Plan Now 65% off! Unlimited websites One-click WordPress installs Free WordPress/cPanel website transfer	Business Plan Now 65% off! Unlimited websites One-click WordPress installs Free WordPress/cPanel website transfer
---	--	--

HostGator

<http://hostgator.com>

坊間一些平台會提供「一條龍」服務，讓客人可一次過同時購買Domain與網存空間，並附有SSL安全證書。

下載免費FTP軟件

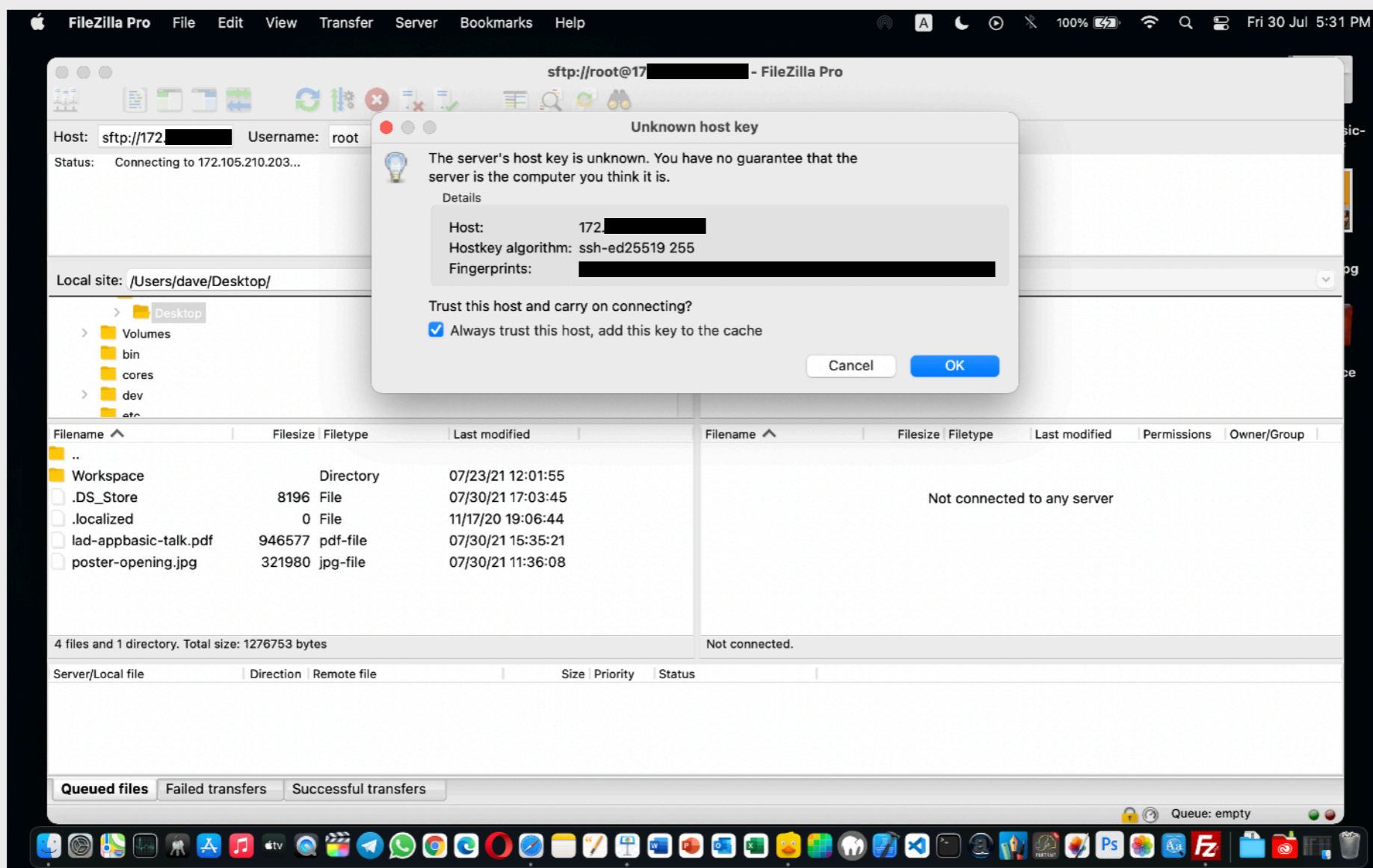
The screenshot shows the FileZilla project website. At the top, the FileZilla logo is displayed with the tagline 'The free FTP solution'. Below the logo is a navigation menu with categories like Home, FileZilla, FileZilla Server, Community, General, Development, and Other projects. The main content area includes an 'Overview' section with a welcome message, a 'Quick download links' section with buttons for 'Download FileZilla Client' and 'Download FileZilla Server', and a 'News' section with a recent update from 2021-07-28. A promotion banner for FileZilla Pro is also present.

FileZilla (兼容Mac與Windows電腦)

<https://filezilla-project.org>

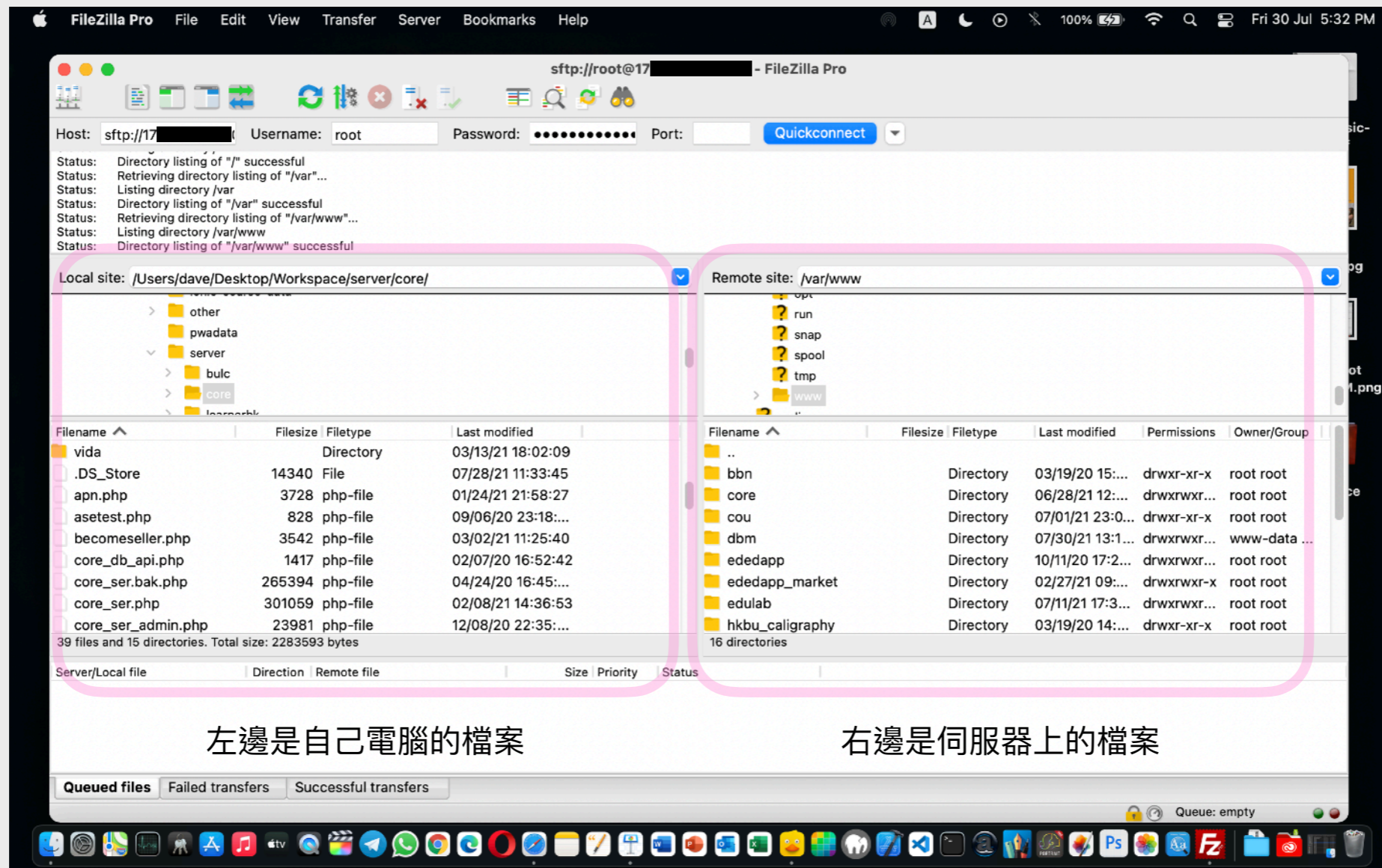
我們可透過FTP軟件來把網從自己的電腦上載到伺服器，在Mac電腦上，除了FileZilla，亦有付費的知名軟件Cyberduck可用。

下載免費FTP軟件



依照網存服務商所提供的資訊(即: IP地址或Domain域名、用戶名稱、用戶密碼、Port位)來連接到FTP或SFTP伺服器，如遇「Unknown host key」問題則點選「Always trust」並按下「OK」便可。

下載免費FTP軟件



左邊是自己電腦的檔案

右邊是伺服器上的檔案

左邊是自己電腦的檔案，右邊是伺服器上的檔案；如要從自己電腦上載檔案到伺服器則從左邊拉扯檔案或文件夾到右邊，反之亦然。當然亦可從電腦的文件夾中直接拉扯檔案到右邊以上載檔案，若下載則反之亦然。

商科術語 - MVP

Minimum Viable Product (最簡可行產品) 是指用最低成本來製作一隻最簡約而可行的產品，只有重要核心功能而沒有多餘功能；用作進行市場測試(「試水溫」)，更可用來吸引和尋找及投資者，或用來申請各類創業基金。

建議製作方法：

如資源有限，建議可製作WEB APP，並只製作UI(使用者界面)部分，並使用Demo Data

例如使用Ionic或Framework7製作可以「搵得」並「有樣有流程」，甚至可以用到最簡單核心功能的Demo APP或Prototype甚或MVP

開發工具

The image shows a Google search for 'vscode' and the Visual Studio Code download page. The search results show 'Visual Studio Code - Code Editor' as the top result. The download page is titled 'Download Visual Studio Code' and lists various operating systems and architectures.

Download Visual Studio Code
Free and built on open source. Integrated Git, debugging and extensions.

Windows
Windows 7, 8, 10

User Installer	64 bit	32 bit	ARM
System Installer	64 bit	32 bit	ARM
.zip	64 bit	32 bit	ARM

.deb
Debian, Ubuntu

.deb	64 bit	ARM	ARM 64
.rpm	64 bit	ARM	ARM 64
.tar.gz	64 bit	ARM	ARM 64

.rpm
Red Hat, Fedora, SUSE

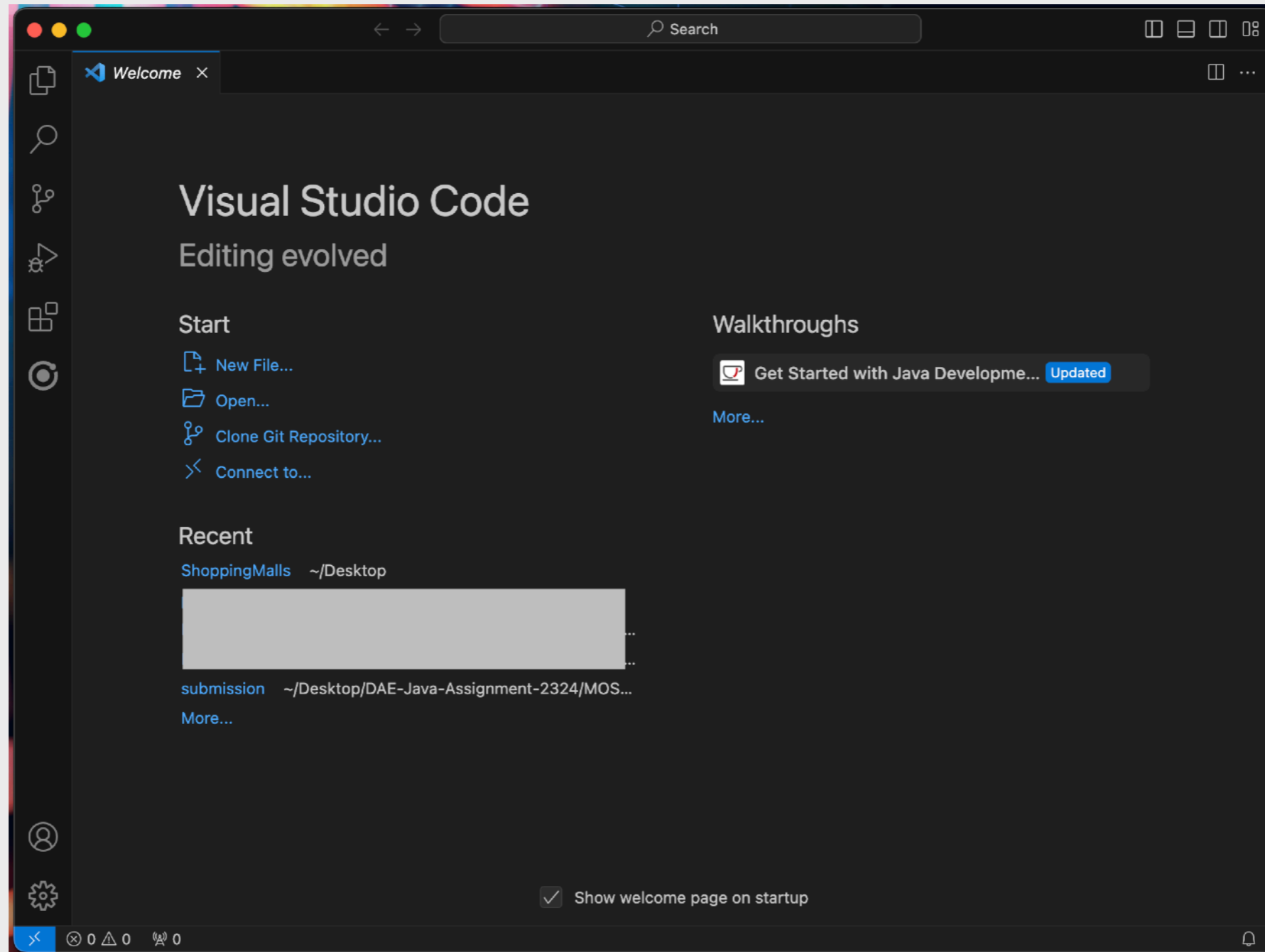
Mac
macOS 10.11+

.zip Universal Intel Chip Apple Silicon

Snap Store

搜尋「vscode」，並下載 Visual Studio Code

開發工具



Visual Studio Code (簡稱VS Code)，開啟後的畫面

APP開發方法



何謂原生APP？



Android



iOS

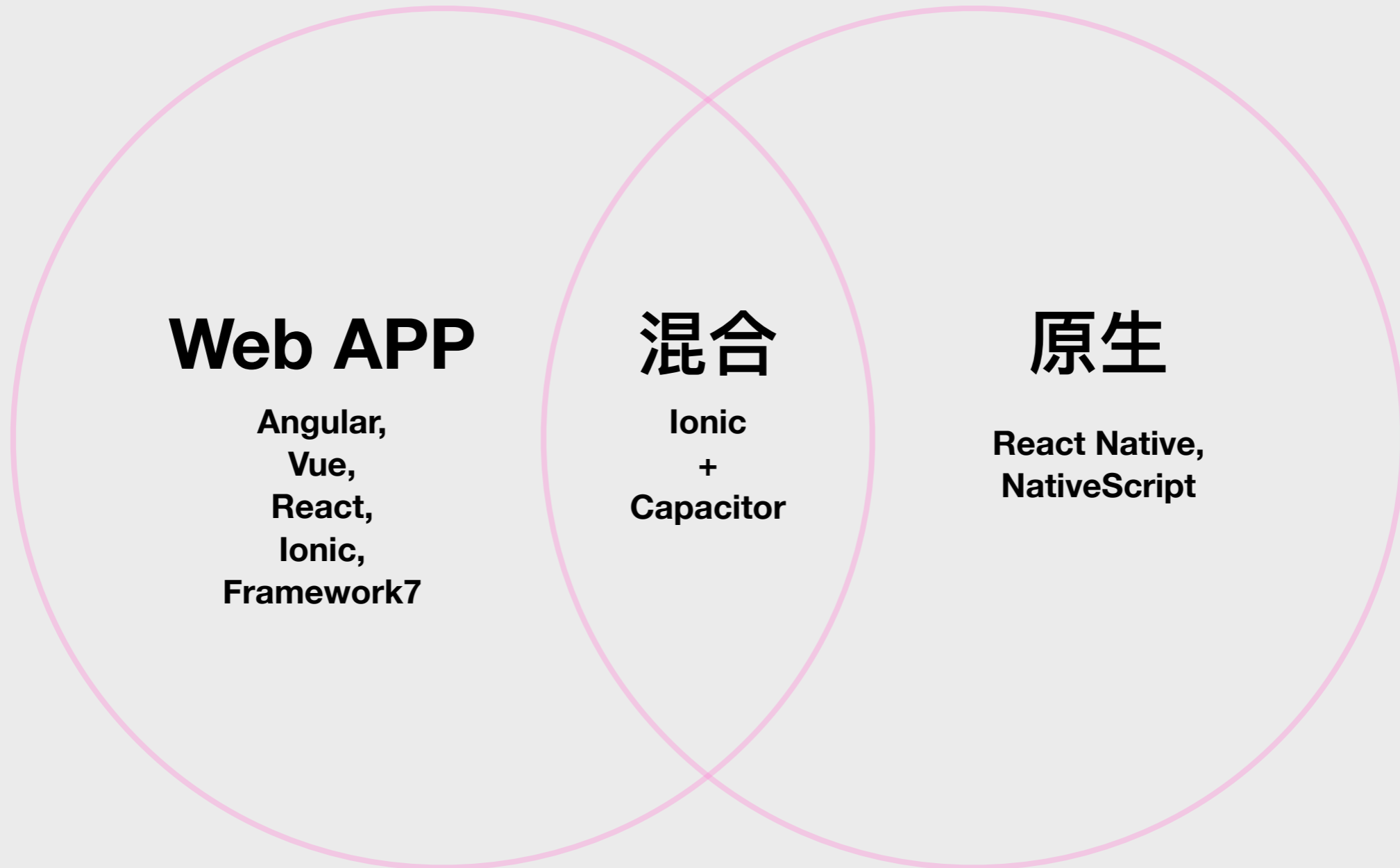
Native App (原生程式):

單純使用該平台原本的程式語言來進行編程，例如使用Swift或Objective-C或C++或C以編寫iOS程式(即iPhone與iPad)，Android則使用Java或Kotlin來編寫。好處是運行速度較快，壞處是每個系統皆要重新開發一次，而且花費的時間或人力成本較多。

近年亦興起以跨平台原生APP開發方法，使用特定的框架或套件來進行開發，該些框架或套件會把開發出來的專案轉換成Swift或Objective-C或C++或C來給予iOS裝置運行，以及轉換成Java或Kotlin來給予Android裝置運行。

常見的跨平台原生APP開發框架或套件有：React Native、Flutter與Native Script等。

APP開發框架



網頁開發基礎知識 (1)



HTML

一個素顏的網頁
(標籤語言)

沒有修飾，
最基礎的結構與外觀



CSS

替網頁整容及化妝
(階層式樣式表)

整容及化妝師，
用作修飾網頁的外觀，
也可為網頁化最初的妝



JavaScript

萬能操控專員
(程式語言)

補妝，控制整容及化妝師在
甚麼時候對網頁下手，甚至
更改網頁的結構

網頁開發基礎知識 (2)



HTML

是一種標記語言，我們使用它組織網頁裡的內容並給予定義，例如：定義段落、標題、資料表格，或是在頁面嵌入圖片和影片。



CSS

是一種樣式規則的語言，用來幫我們的 HTML 內容上套用樣式，例如：設置背景顏色、字型，或讓內容以多欄的方式呈現。



JavaScript

是一種腳本語言(Scripting Language)，它使你能夠動態的更新內容、控制多媒體、動畫.....等不同事情。

詳情可參閱：https://developer.mozilla.org/zh-TW/docs/Learn/JavaScript/First_steps/What_is_JavaScript

網頁開發基礎知識



HTML的外觀

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>HTML範例</title>
  </head>
  <body>
    <div>
      Hello，這是HTML的範例！
    </div>
  </body>
</html>
```

網頁開發基礎知識



CSS的外觀

```
a{
  color: green;
}

.img_TypeA{
  width: 50vw;
  height: 50vh;
  border: 5px solid rgb(50,211,6);
}

#div_main{
  width: 50vw;
  height: 50vh;
  border: 5px solid rgb(50,211,6);
}
```

網頁開發基礎知識



JavaScript的外觀

```
async function createAjax(url, method, data){  
  const response = await fetch(url, {  
    method, headers: {  
      'Accept': 'application/json',  
      'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',  
    }, body: new URLSearchParams(data)  
  });  
  return response.ok ? response.json():null;  
}
```

```
createAjax('ser.php', 'POST', {service:"test"})  
.then(res=>{  
  console.log(res);  
});
```

HTML網頁標籤語言

一個HTML5的網頁(文件)可以分成三個主要部分:

1. 文件類型(DOCTYPE)宣告
2. 網頁表頭
3. 網頁內容。

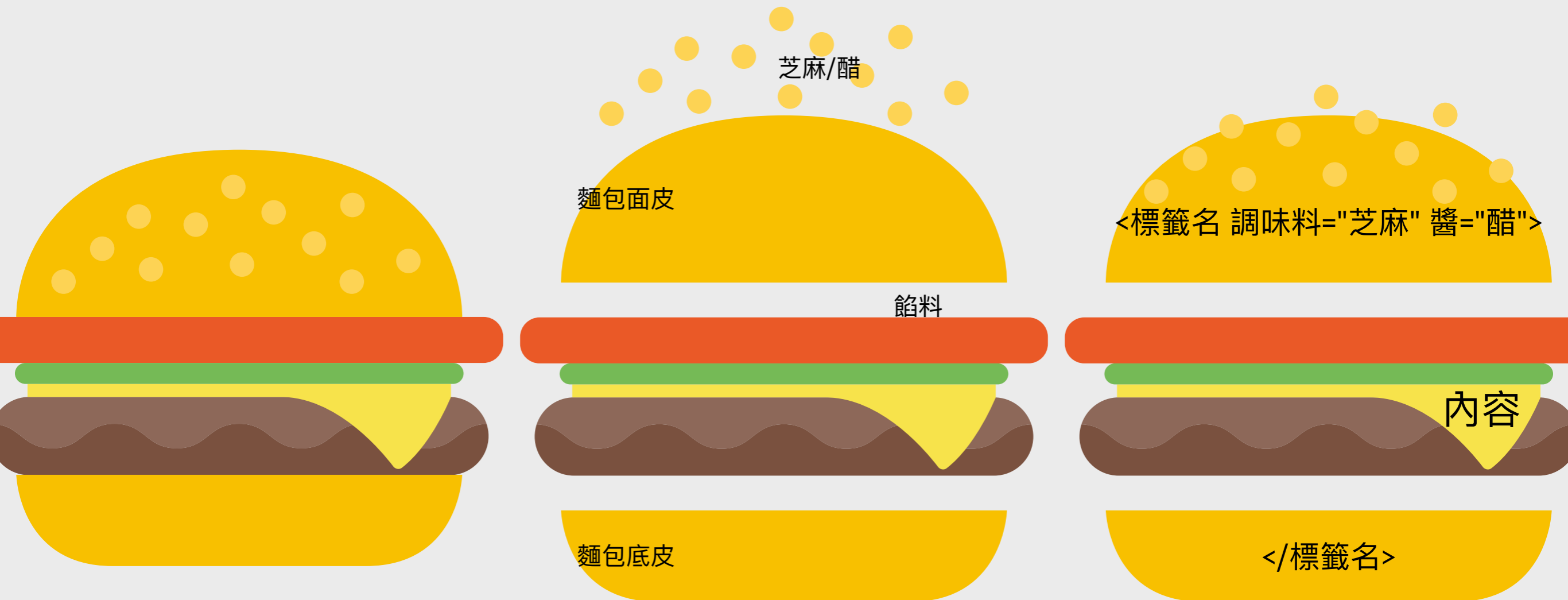
一般HTML5的網頁，我們可以使用`<!DOCTYPE HTML>`來宣告這個檔案是屬於HTML的類型。而使用`<head></head>`來表達網頁表頭，使用`<body></body>`內代表網頁內容。

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5     <title>HTML範例</title>
6   </head>
7   <body>
8     <div>Hello World</div>
9     <span>HTML</span>
10  </body>
11 </html>
```

HTML的大約外觀

HTML的寫法結構

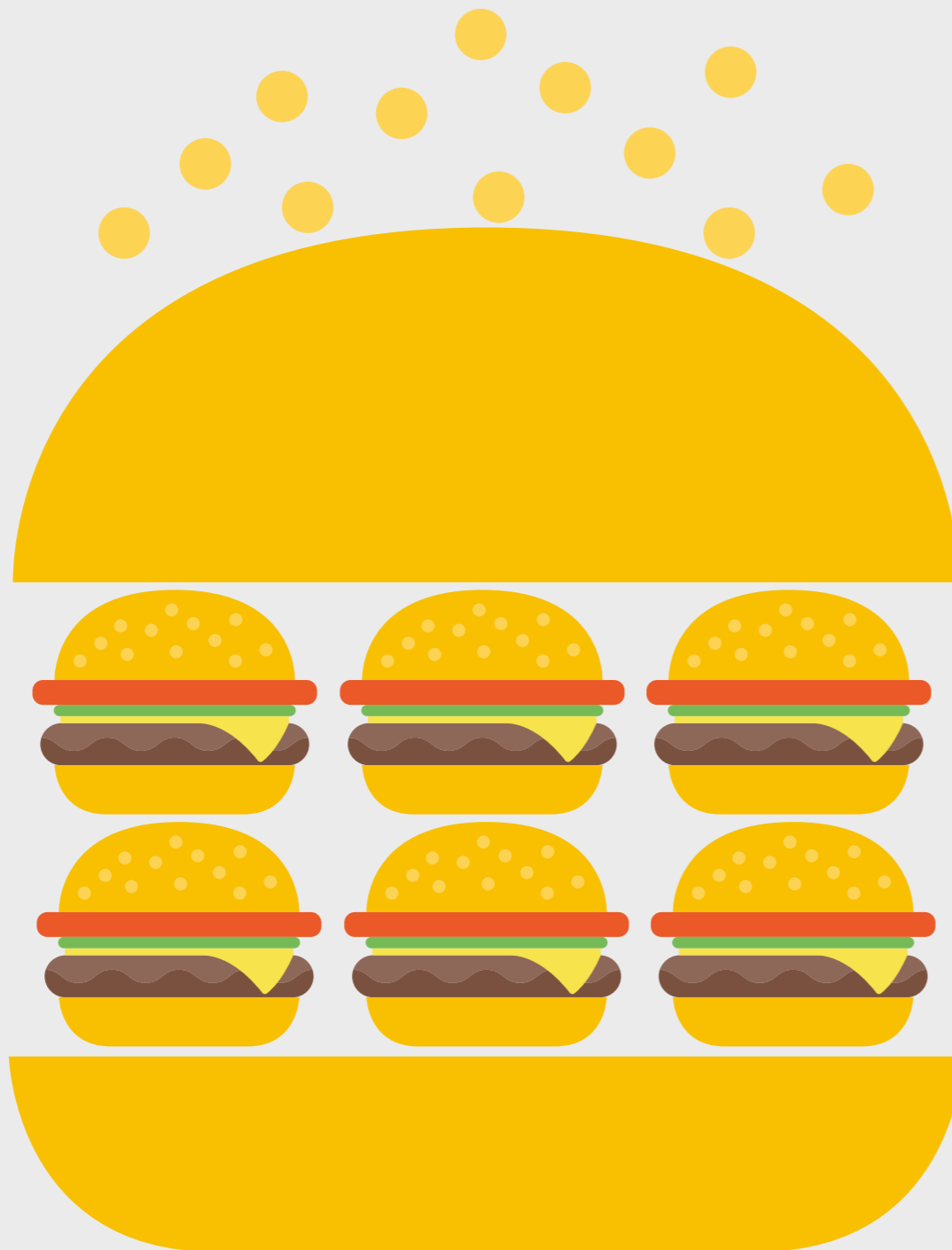
HTML是一種「標籤語言」，用來編寫一個網頁的「骨架」



漢堡包定理

HTML的寫法結構

漢堡包定理



漢堡飽中亦可以有漢堡飽

而漢堡包中的子漢堡包，
亦可以再冇漢堡包

如此類推.....

HTML網頁標籤語言

標籤(Tag)有始有終，有「開」便有「關」

所有標籤(Tag)和屬性(Attribute)皆為小寫(細階)

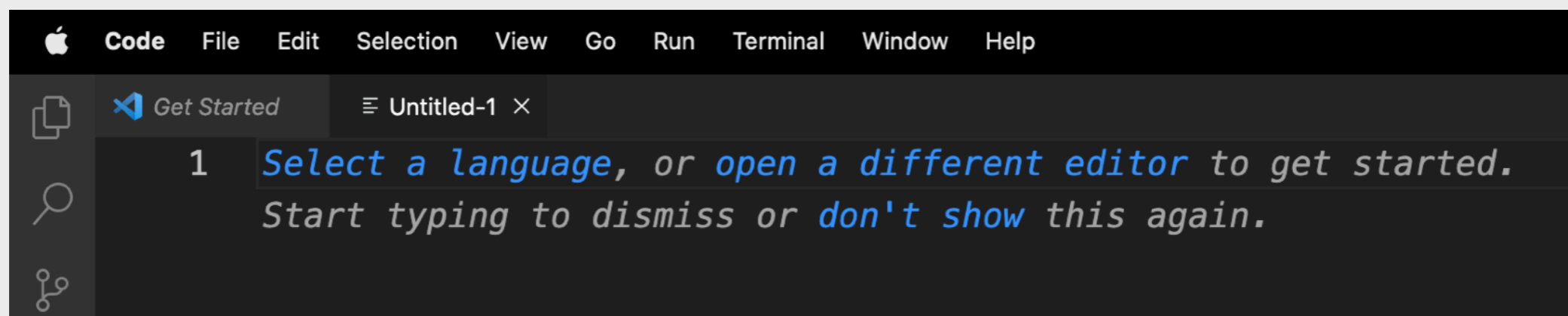
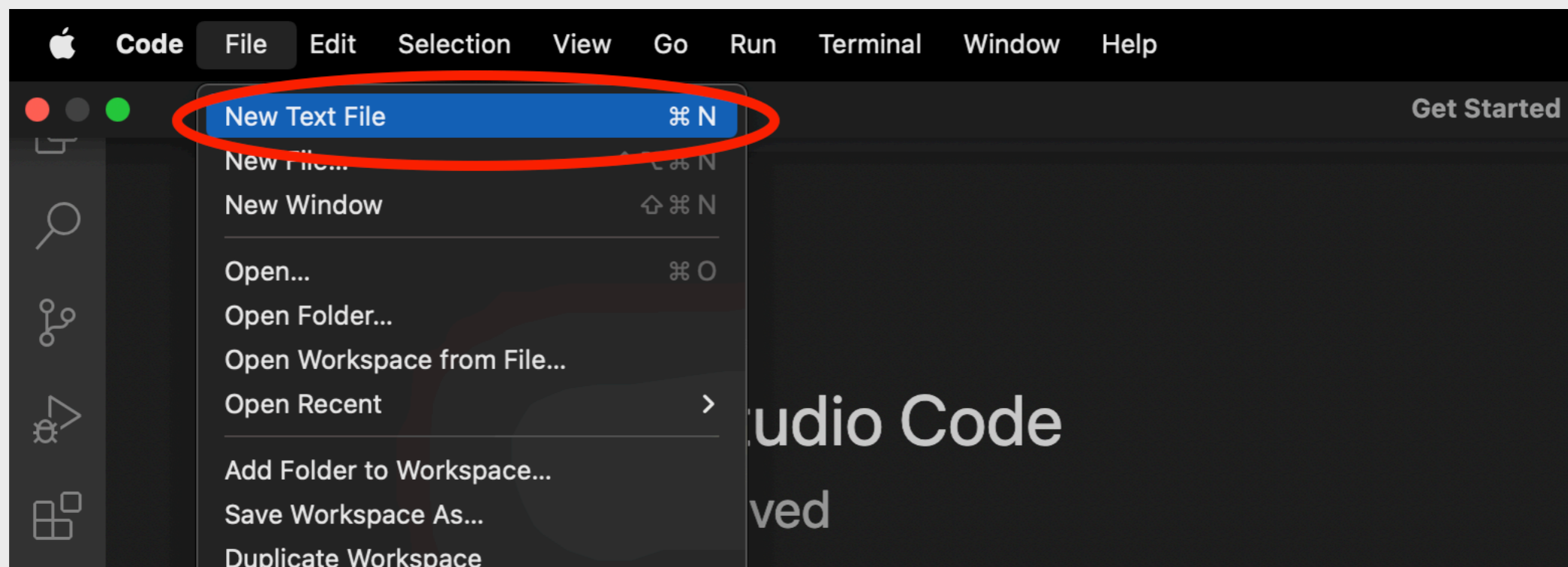
標籤元素的內容須寫於標籤元素內的「開始」與「終結」之內，如
`<title>Document Title</title>`之內

也有一些例外，如`
`、``、`<!DOCTYPE html>`等

為了對標籤元素進行更多的設定，我們便須知道甚麼是屬性(Attribute)
和屬性值(Attribute Value)

可透過`<!-- 註解內容 -->`為HTML內容寫下註解

建立一個HTML檔案 (1)



打開VSCode，於File選單中選New Text File，然後便會出現一個能夠輸入文字的空白頁面。

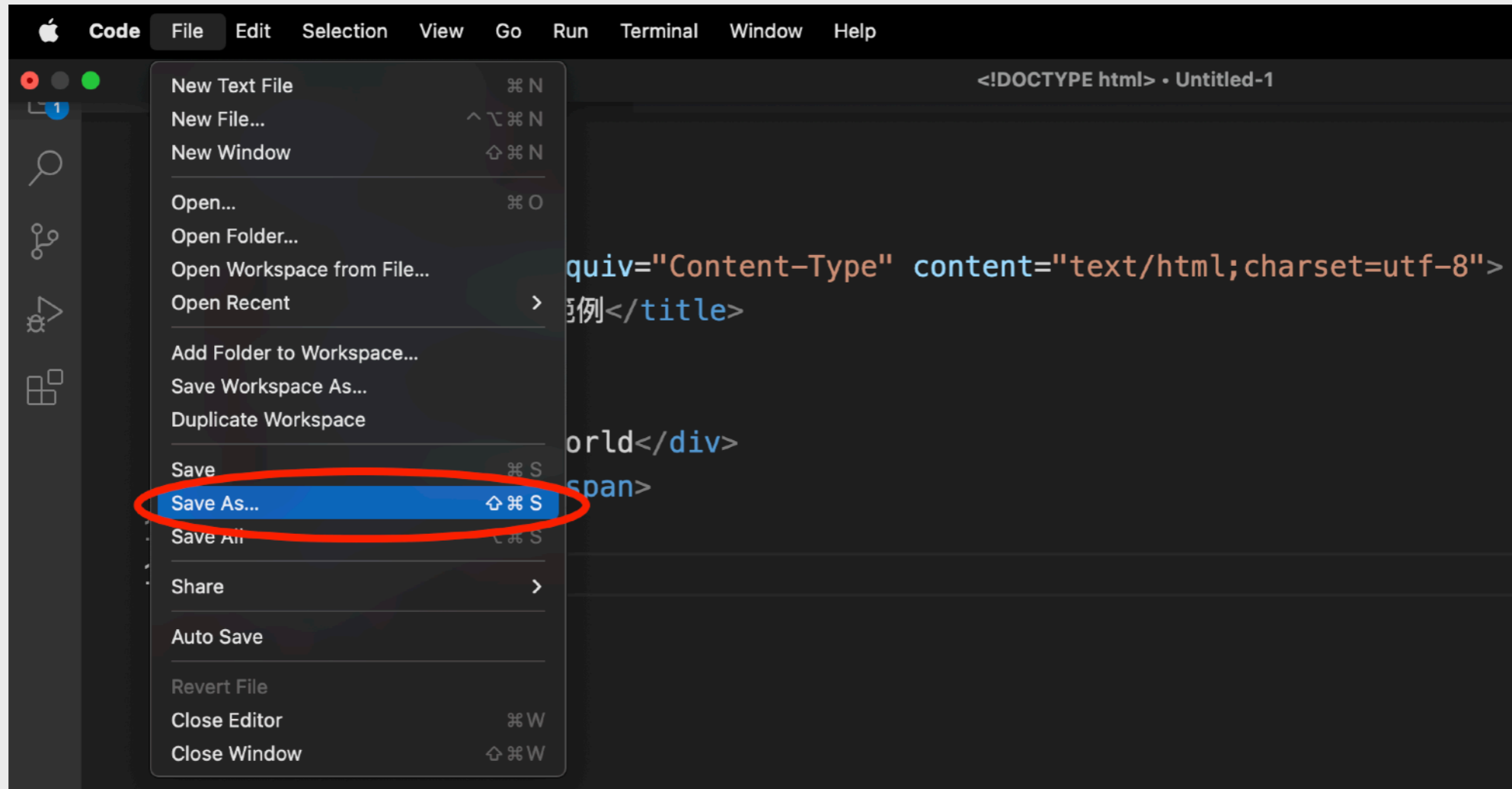
建立一個HTML檔案 (2)

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>HTML範例</title>
  </head>
  <body>
    <div>Hello World</div>
    <span>HTML</span>
  </body>
</html>
```

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5     <title>HTML範例</title>
6   </head>
7   <body>
8     <div>Hello World</div>
9     <span>HTML</span>
10  </body>
11 </html>
```

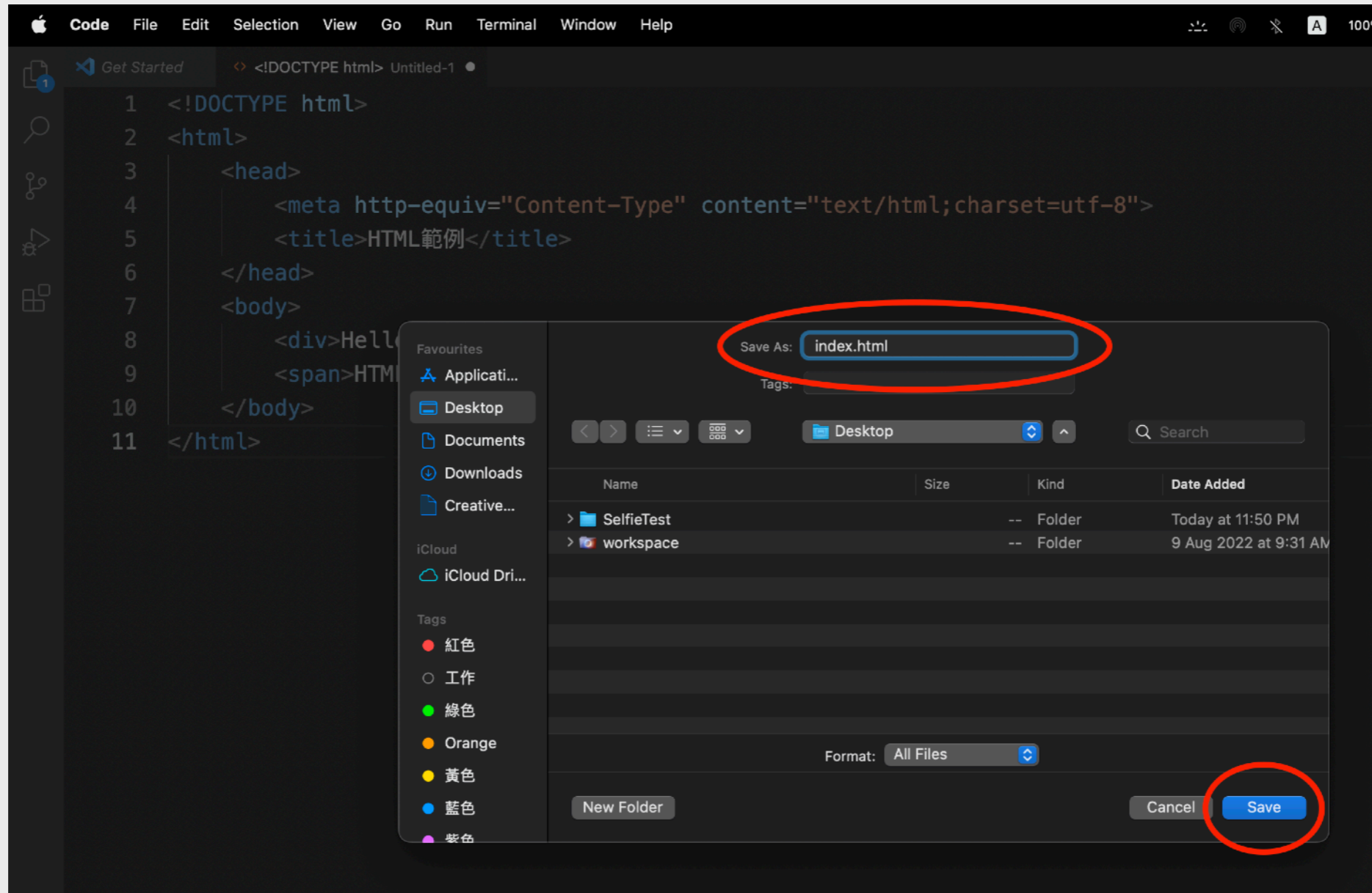
然後把內容輸入或複製貼上到該檔案中

建立一個HTML檔案 (3)



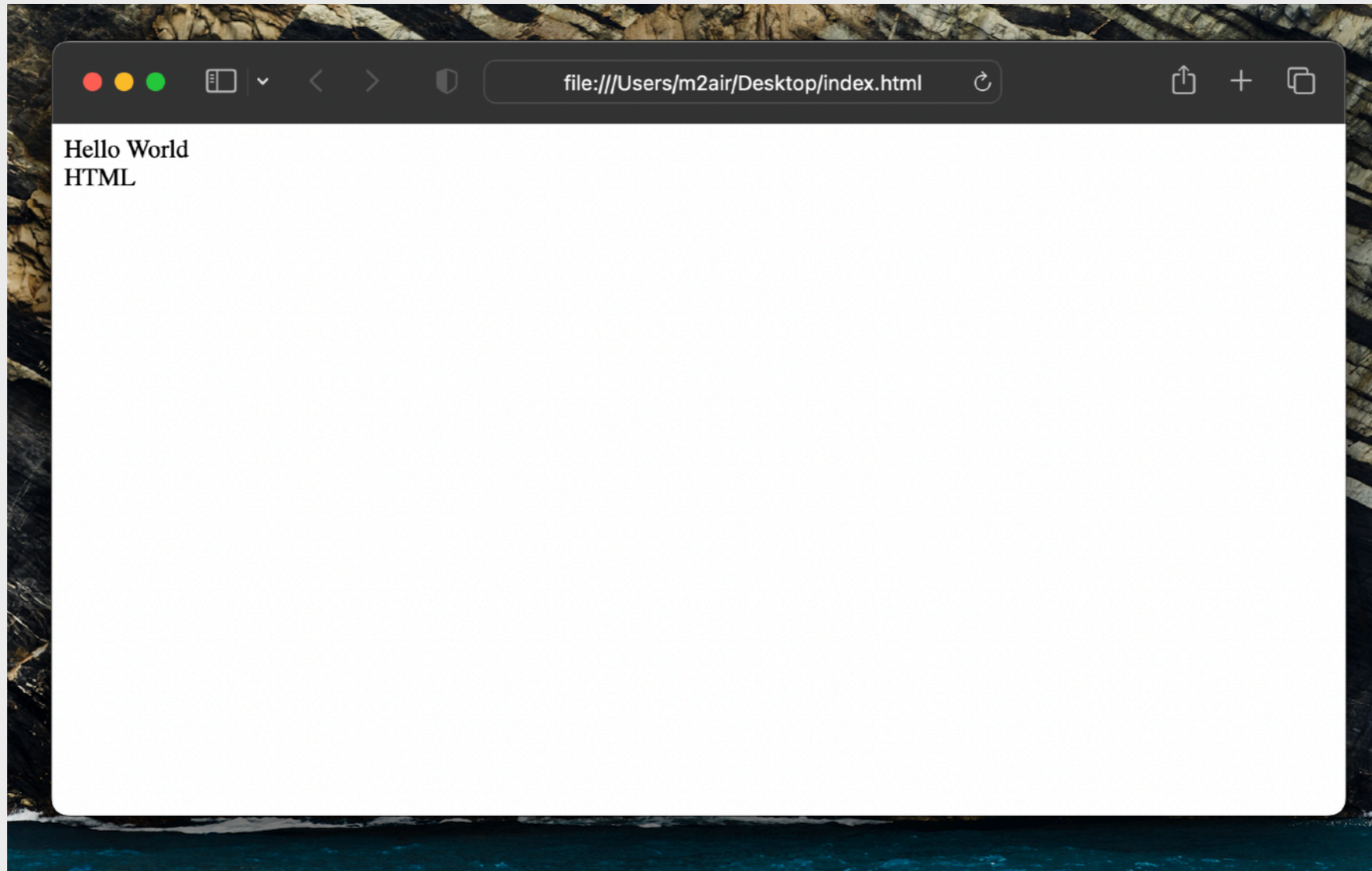
到File選單中按下「Save As...」以儲存檔案

建立一個HTML檔案 (4)



把文字檔命名為「index.html」並儲存到想存放的位置
(上述範例為Desktop桌面)

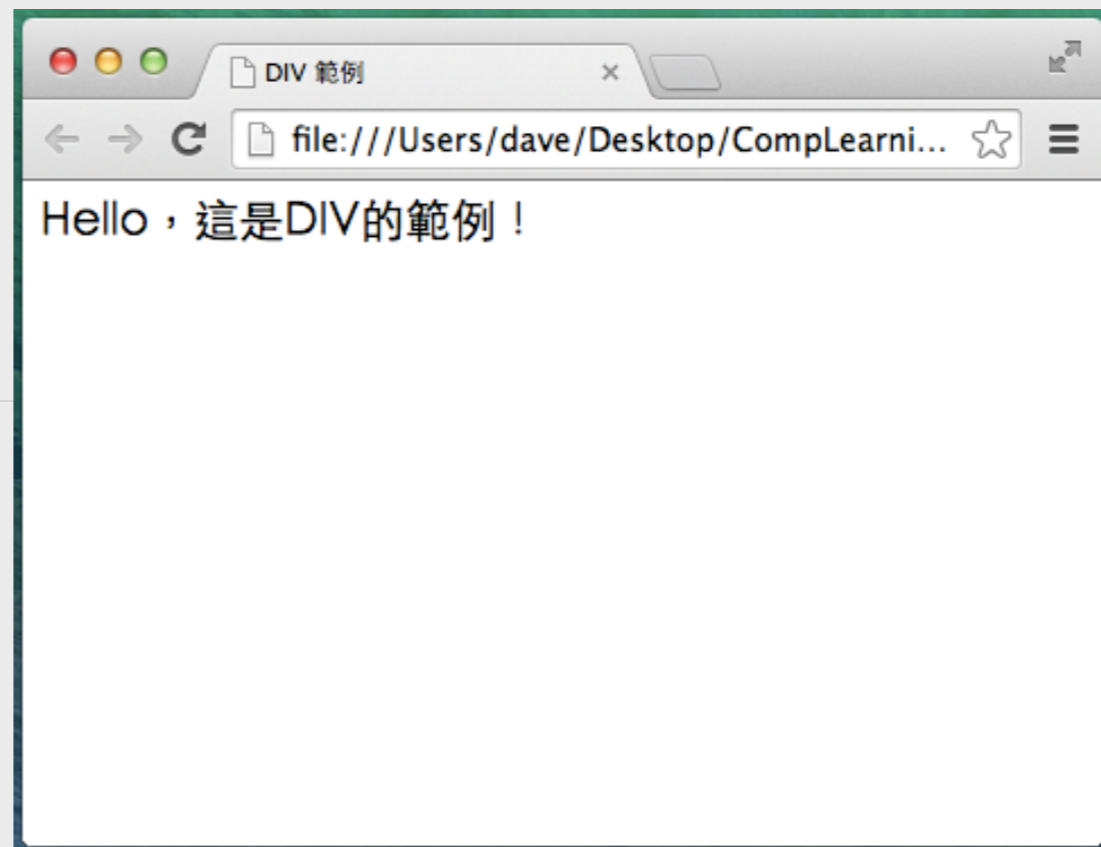
建立一個HTML檔案 (5)



然後使用瀏覽器把網頁檔(index.html)開啟，
則會成功看見網頁內容

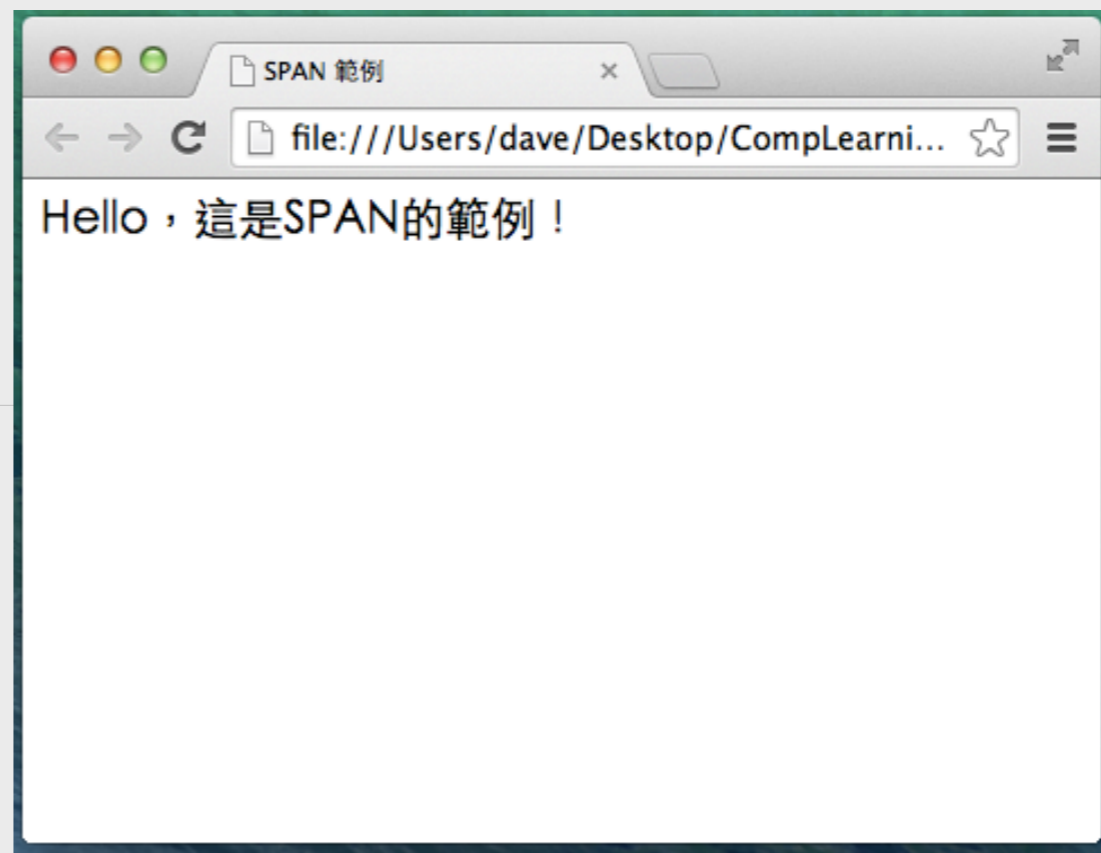
HTML精選標籤-DIV

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>DIV 範例</title>
</head>
<body>
<div>
Hello，這是DIV的範例！
</div>
</body>
</html>
```



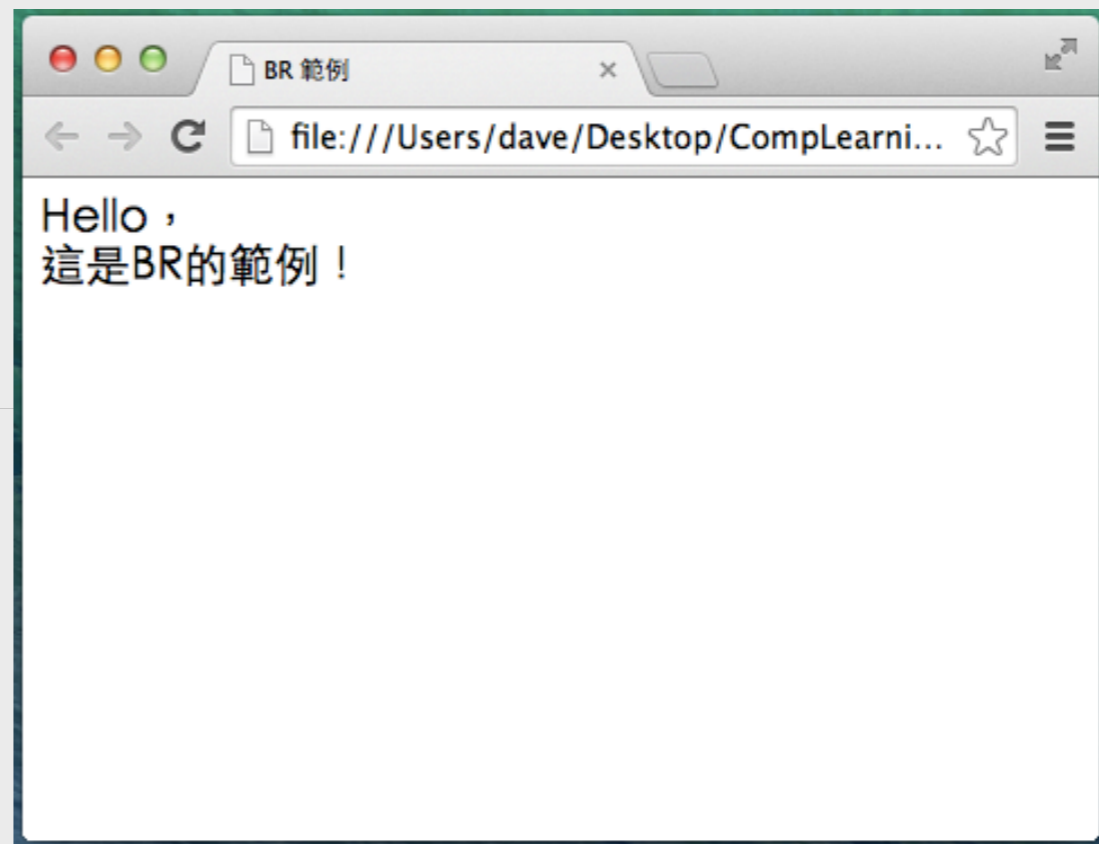
HTML精選標籤-SPAN

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>SPAN 範例</title>
</head>
<body>
<span>
Hello，這是SPAN的範例！
</span>
</body>
</html>
```



HTML精選標籤-BR

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>BR 範例</title>
</head>
<body>
Hello ,
<br>
這是BR的範例！
</body>
</html>
```



HTML精選標籤-A

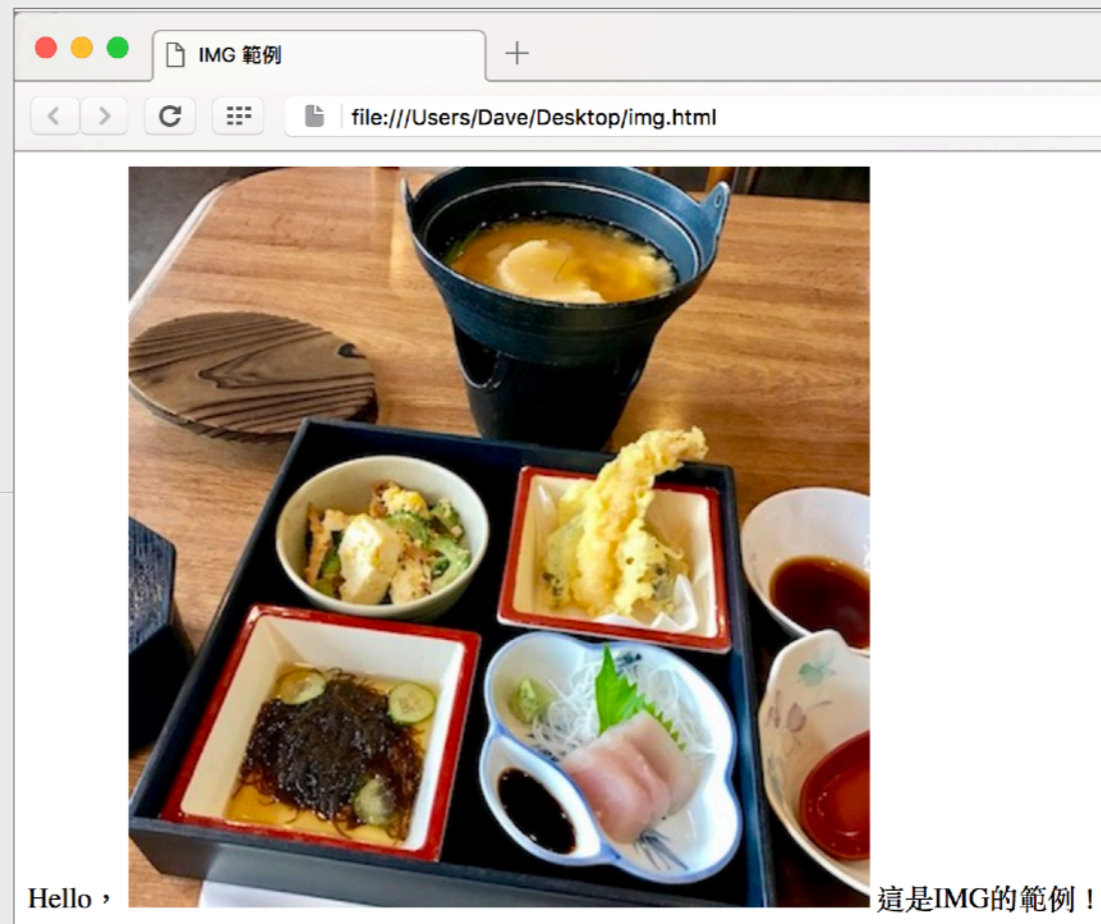
```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8">
<title>A 範例</title>
</head>
<body>
Hello ,
<a href="index.html">
這是A的範例！
</a>
</body>
</html>
```



HTML精選標籤-IMG

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>IMG 範例</title>
</head>
<body>
Hello ,

這是IMG的範例！
</body>
</html>
```



HTML精選標籤-VIDEO

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>VIDEO 範例</title>
</head>
<body>
<video width="320" height="240" controls>
<source src="video.mp4" type="video/mp4">
</video>
<br>
這是VIDEO的範例！
</body>
</html>
```



HTML精選標籤-TABLE

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>TABLE 範例</title>
```

```
<style>
table{
border: 1px solid black;
border-spacing: 0px;
}
td{
border: 1px solid black;
}
</style>
```

```
</head>
<body>
<table>
```

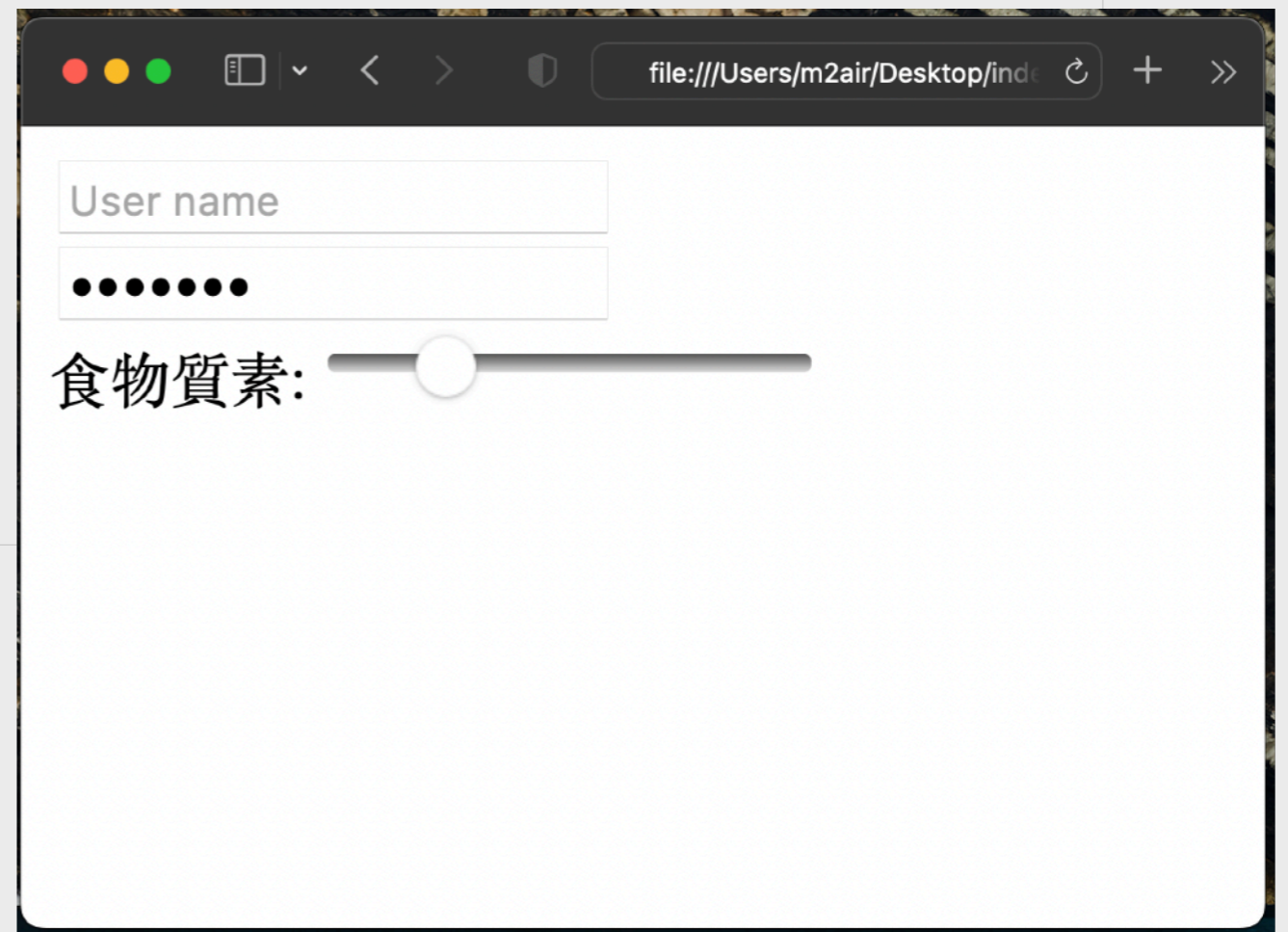
```
<tr><td>馮大文</td><td>Alex Fung</td><td>BComp, MEng, DEng</td></tr>
<tr><td>薩小欣</td><td>Lily Sa</td><td>BA (Hons), EdD, DBA</td></tr>
<tr><td>戴小明</td><td>Joe Tai</td><td>BSc, MPhil, ScD</td></tr>
<tr><td>杜安安</td><td>Eric To</td><td>BEng, MSc, MEd, EdD</td></tr>
<tr><td>穆大明</td><td>Ming Mu</td><td>BEcon, MPhil, MBA, PhD</td></tr>
<tr><td>鄒金主</td><td>Gold Chow</td><td>BArch (1st Hons), DEng</td></tr>
<tr><td>蔣奇</td><td>Kay Chiang</td><td>MCA, MSc, DCompSci</td></tr>
</table>
```

```
<br>
這是TABLE的範例！
</body>
</html>
```



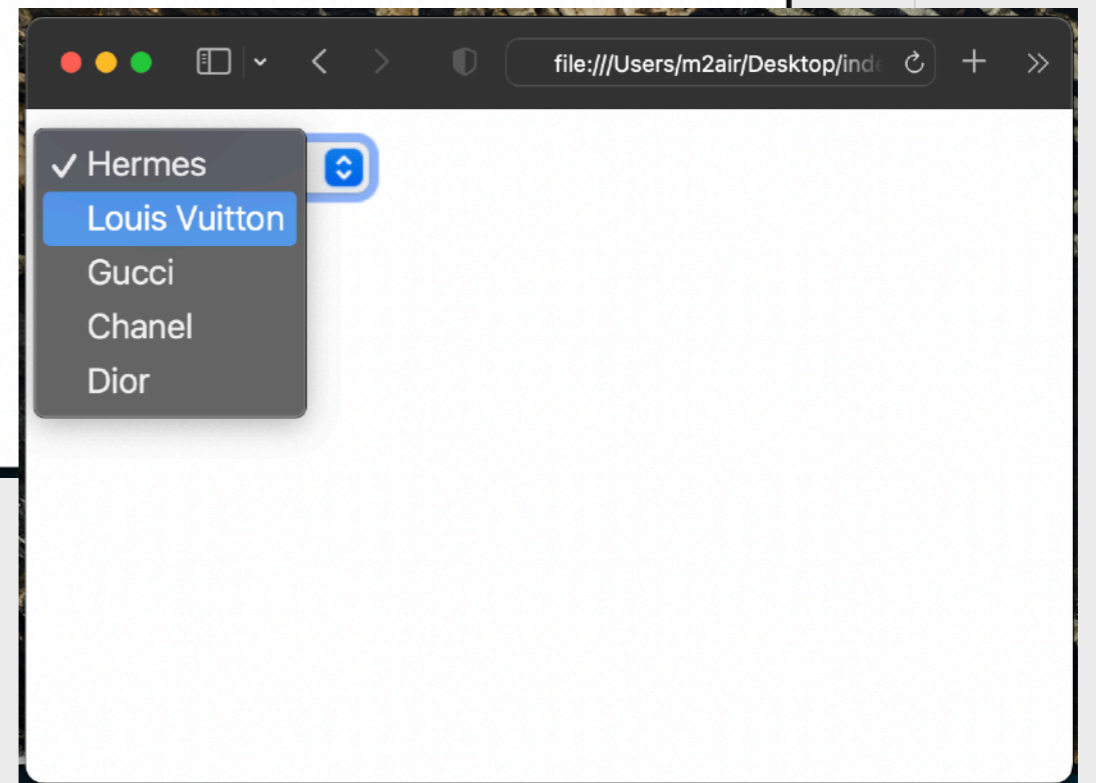
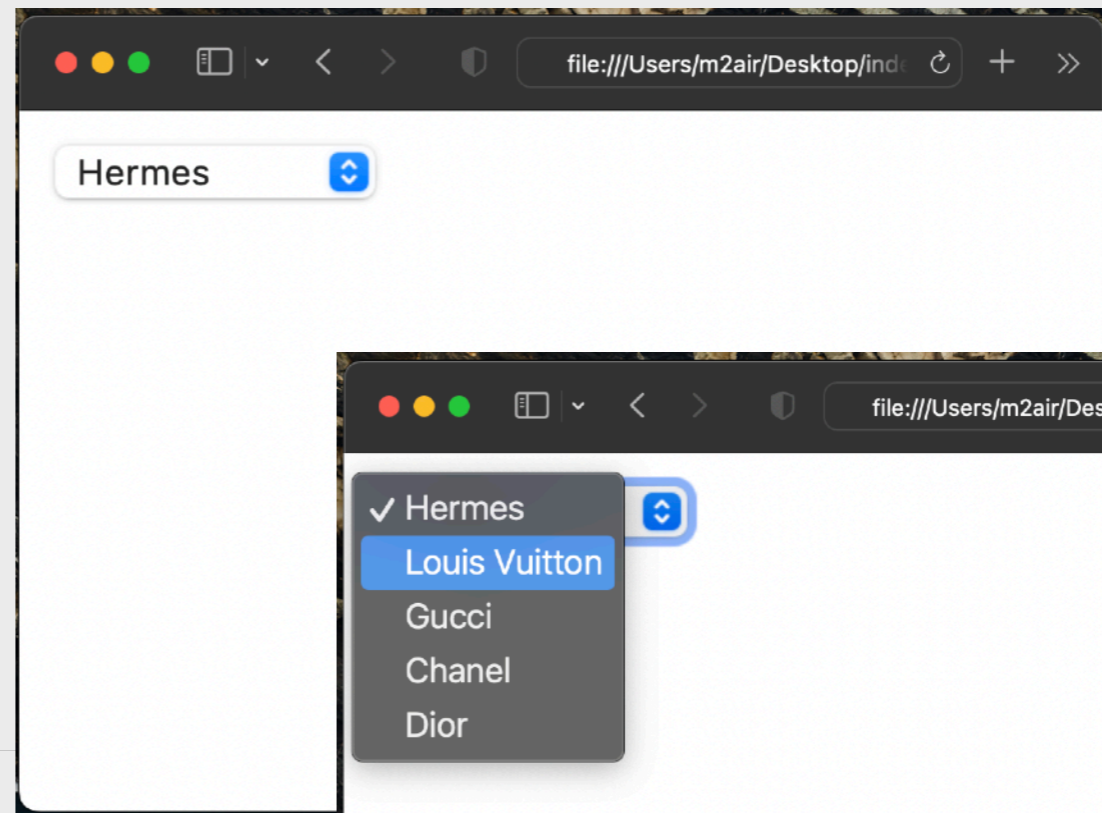
HTML精選標籤-INPUT

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>INPUT 範例</title>
</head>
<body>
<input placeholder="User name">
<br>
<input type="password" placeholder="Password">
<br>
食物質素: <input type="range" min="0" max="5">
</body>
</html>
```



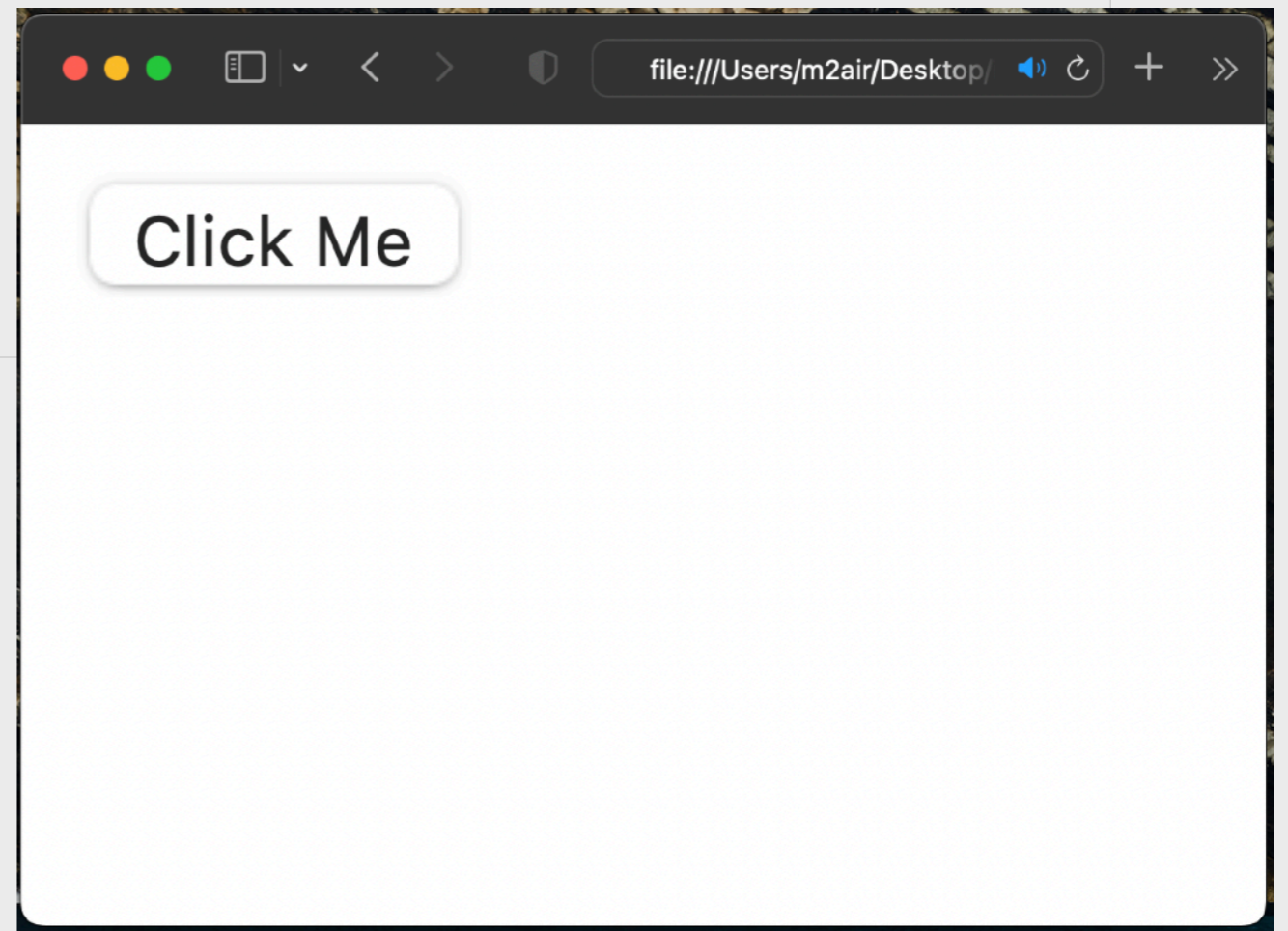
HTML精選標籤-SELECT

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>SELECT 範例</title>
</head>
<body>
<select id="brands">
  <option value="hermes">Hermes</option>
  <option value="lv">Louis Vuitton</option>
  <option value="gucci">Gucci</option>
  <option value="chanel">Chanel</option>
  <option value="dior">Dior</option>
</select>
</body>
</html>
```



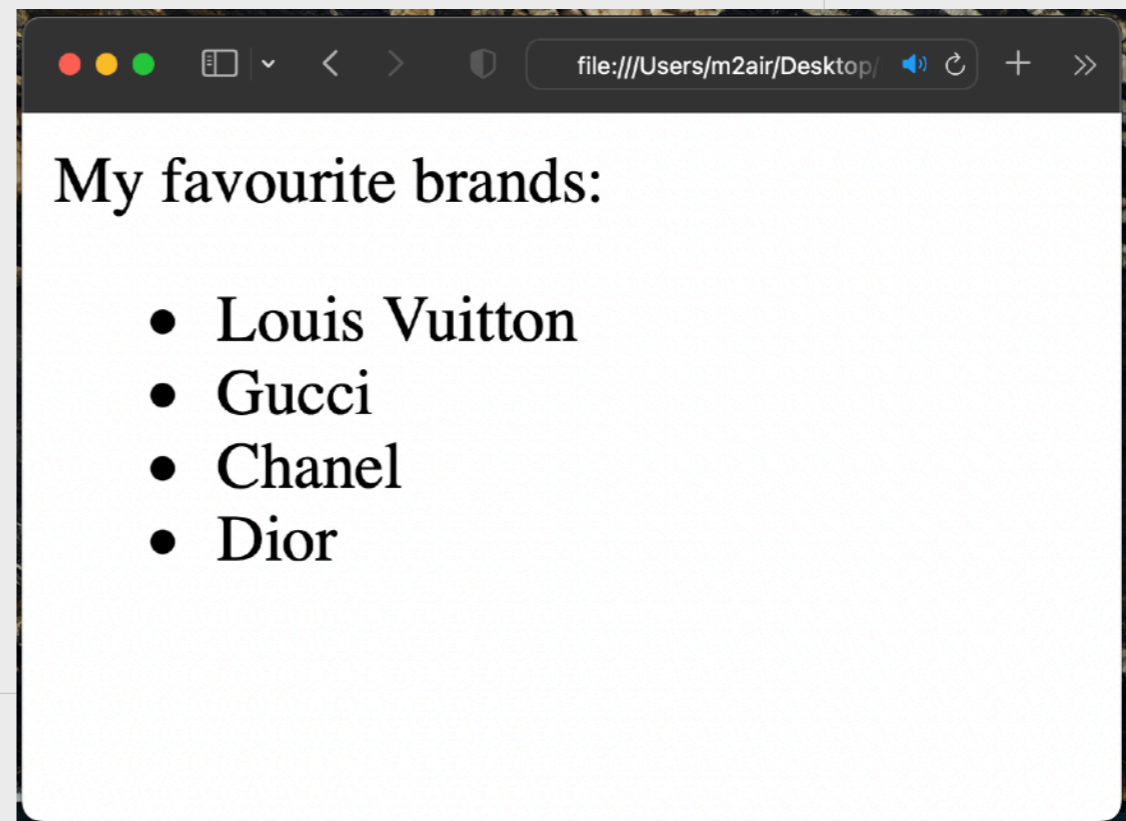
HTML精選標籤-BUTTON

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>BUTTON 範例</title>
</head>
<body>
<button onclick="alert('hello');">Click Me</button>
</body>
</html>
```



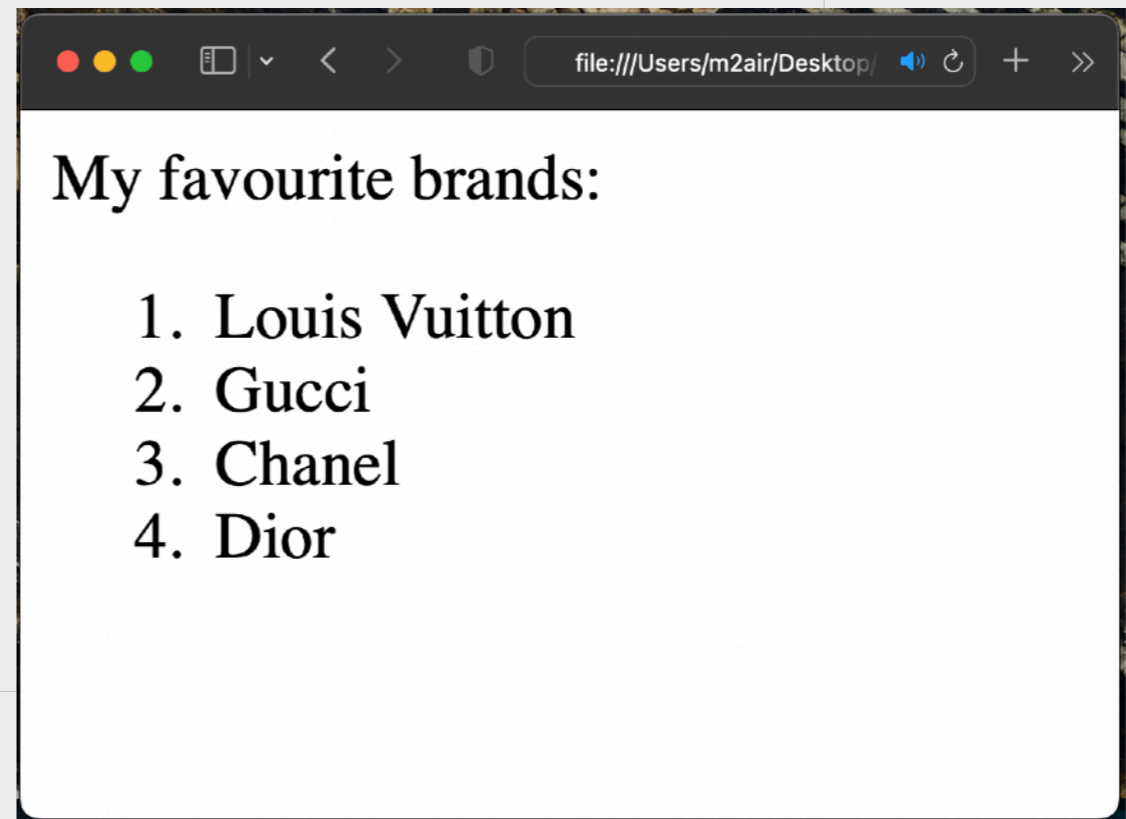
HTML精選標籤- UL & LI

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>UL與LI 範例</title>
</head>
<body>
My favourite brands:
<ul>
  <li>Louis Vuitton</li>
  <li>Gucci</li>
  <li>Chanel</li>
  <li>Dior</li>
</ul>
</body>
</html>
```



HTML精選標籤- OL & LI

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>OL與LI 範例</title>
</head>
<body>
My favourite brands:
<ol>
  <li>Louis Vuitton</li>
  <li>Gucci</li>
  <li>Chanel</li>
  <li>Dior</li>
</ol>
</body>
</html>
```



HTML Entities 字符實體

HTML中的預留字符必須被替換為字符實體。一些在鍵盤上找不到的字符也可以使用字符實體來替換。在HTML中，某些字符是預留的。在HTML中不能使用小於號 < 和大於號 >，這是因為瀏覽器會誤認為它們是標籤。如果希望正確地顯示預留字符，我們必須在HTML源代碼中使用字符實體 (character entities)。

符號	代表字串
<	<
>	>
&	&
"	"
'	'
¢	¢
£	£
¥	¥
€	€
©	©
®	®
[空格]	

HTML屬性介紹

- 所有 HTML 元素都可以有屬性
- 屬性提供有關元素的附加信息
- 屬性總是在開始標籤中指定
- 屬性通常以名稱/值對的形式出現，例如：`name="value"`
- 亦有一些例外，不以名稱/值對的形式出現，如 `required`、`readonly`、`selected`、`checked`、`multiple`、`disabled`、`autofocus`、`controls`、`autoplay` 等

HTML常用屬性

屬性	用於	用法及解釋
href	a	<code>W3Schools</code> 連結到指向的路徑，可指定絕對URL或相對URL
src	img, video, audio, source, script	<code></code> 顯示路徑目的的檔案，可指定絕對URL或相對URL
alt	img	<code></code> 如無法顯示圖像則顯示alt中的文字代替圖像，也方便視障人士使用屏幕閱讀器聽取圖像的意思
type	input	<code><input id="input_username" type="text"></input></code> 為input元素定義輸入種類
style	全部皆適用	<code><div style="color:red;">Red</div></code> 為元素加上 inline CSS 樣式
class	全部皆適用	<code>Class A SPANs</code> 為元素加上類別(Class)，可被JavaScript及CSS更易地使用
id	全部皆適用	<code><input id="input_username" type="text"></input></code> 為元素加上類別(Class)，可被JavaScript及CSS更易地使用

HTML練習1

```
<!DOCTYPE html>
<html>
  <head>
    <title>Welcome</title>
  </head>
  <body onload="bodyLoaded();" onclick=bodyClicked();>
    <TABLE>
      <tr>
        <td style="font-weight:bold;">Name</td>
        <td style="font-weight:bold;">Education</td>
      </tr>
      <tr>
        <td>John</td>
        <td>BSc, MEng, MBA, PhD</td>
      </tr>
      <tr>
        <td>Lily</td>
        <td>BSocSci, MA, MPhil, EdD</td>
      </tr>
      <tr>
        <td>Kay</td>
        <td>BComp, MSc, DEng, DBA</td>
      </tr-end>
    </table>
  </body>
</html>
```

以下有四個錯誤。

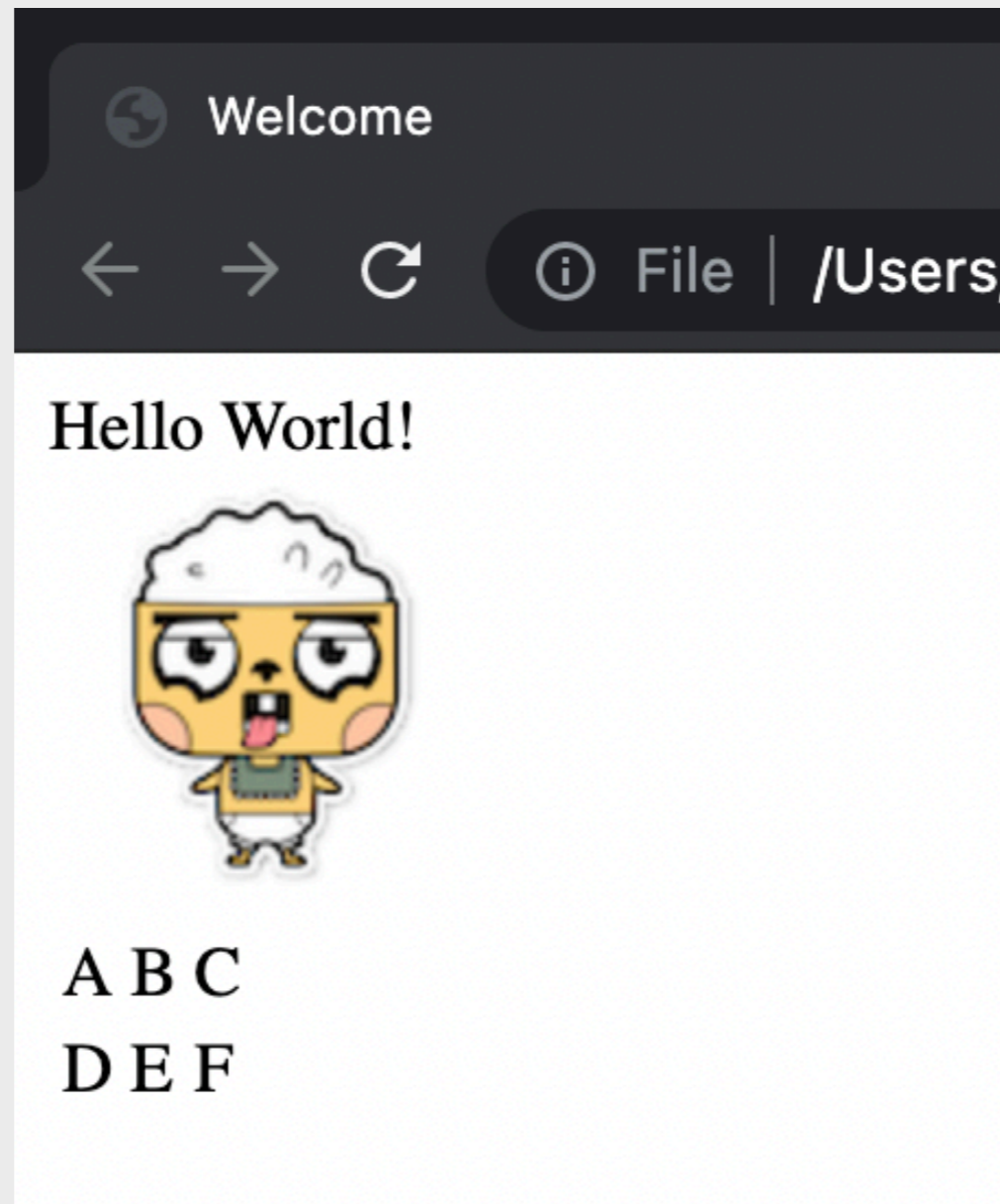
HTML練習2

請填寫漏空處。

```
< >
<html>
  <head>
    <title>Welcome</title>
  </head>
  <body>
    <div><span>Testing</span></ >
    < >
    < src="test.jpg" style="width:15vw;">
  </body>
</html>
```

HTML練習3

請根據以下圖片來編寫出HTML。(提示：圖片下方的ABCDEF為table，圖片檔名為test.jpg)



HTML表單與驗證 (1)

若需要用戶輸入資料並交給程式處理，則可用上表單(form)元件及各種輸入元件給予用戶「填表單」，例如註冊帳戶、填寫問卷、填寫申請表格等。

HTML 的 `<form>` 元素用於創建一個包含表單元素的區域，這些表單元素可以是文本欄位、選項、按鈕等。用戶可以在表單中輸入數據，然後將其提交到服務器以進行處理。

表單的基本用法包括：

- `action` 屬性：指定當表單提交時數據應該發送到的 URL。
- `method` 屬性：指定數據發送到服務器時使用的 HTTP 方法（通常是 GET 或 POST）。

表單控件：如 `<input>`、`<textarea>`、`<button>`、`<select>` 等，用於接收用戶輸入的數據。

HTML表單與驗證 (2)

以下是一個簡單的表單示例：

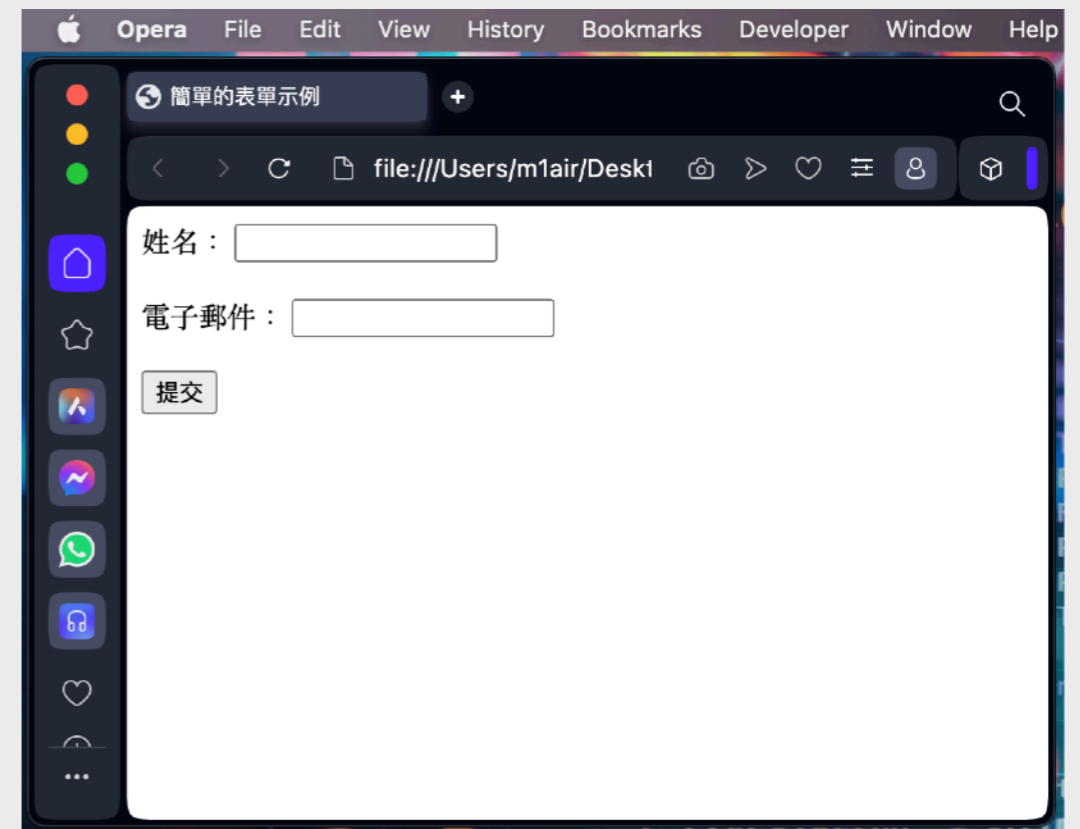
```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>簡單的表單示例</title>
</head>
<body>

<form action="submit_form.php" method="post">
  <label for="name">姓名：</label>
  <input type="text" id="name" name="name"><br><br>

  <label for="email">電子郵件：</label>
  <input type="email" id="email" name="email"><br><br>

  <input type="submit" value="提交">
</form>

</body>
</html>
```



HTML表單與驗證 (3)

在上頁的示例中：

表單的 `action` 屬性設為 `submit_form.php`，這意味著當用戶提交表單時，數據將被發送到同一目錄下的 `submit_form.php` 文件進行處理。（後端開發稍後會詳解）

`method="post"` 表示數據將通過 POST 方法發送，這種方法適合用於提交應該保密的數據（例如，密碼或個人信息）。

`<input>` 標籤用於創建不同類型的輸入控件，這裡有兩個輸入框：一個用於填寫姓名，另一個用於填寫電子郵件地址。`name` 屬性對每個輸入域指定一個名稱，該名稱將作為數據的鍵被發送到服務器。

`<label>` 標籤為輸入控件定義標籤，並通過 `for` 屬性與對應的輸入控件相關聯。

最後一個 `<input>` 是提交按鈕，當用戶單擊它時，表單將被發送到指定的 URL。

服務器端（在這個例子中是 `submit_form.php`）將處理傳入的表單數據，可以進行保存、發送郵件、驗證等操作。

HTML表單與驗證 (4)

本講義前些部分曾介紹過簡單的Input及Select元件，現在便詳細講述一下怎樣透過各種輸入元件及其屬性(attribute)與表單(Form)來詢問用戶輸入資料。

各種輸入元件:

input	textarea	select
-------	----------	--------

Input的種類:

checkbox	color	date	datetime-local
email	file	month	number
password	radio	range	tel
text	time	url	week

按鈕型Input:

reset	submit
-------	--------

各種輸入元件的特別屬性:

required	readonly	selected	checked
multiple	disabled	autofocus	

入學申請表：

名稱

稱謂

報讀原因

最高學歷 博士 碩士
 學士 大專
 中專 中學
 小學 其他

聯絡電話

聯絡電郵

優惠密碼

個人近照 no file selected

從何得知 廣告 親友
 搜尋 老師

HTML表單與驗證 (5)

placeholder屬性可用於顯示輸入提示，input與textarea皆可使用

type為radio及checkbox的input元件可加入name屬性來進行組合

type為radio及checkbox的input元件需加入value屬性來定義資料

各輸入元件皆可加入id屬性來方便JavaScript存取其值(value)

當送出表單時，如輸入不合規則，瀏覽器便會自動提示用戶，如：

The image displays three examples of browser validation messages overlaid on a form:

- Example 1:** A form with fields for "最高學歷" (Highest Education), "聯絡電話" (Contact Phone), and "聯絡電郵" (Contact Email). The "最高學歷" field has a message "Fill out this field" pointing to it.
- Example 2:** A form with a "最高學歷" field. A message "Select one of these options" points to the radio button options: 博士 (PhD), 碩士 (Master's), 學士 (Bachelor's), 大專 (Vocational), 中專 (Vocational), 中學 (High School), 小學 (Elementary), and 其他 (Other).
- Example 3:** A form with fields for "聯絡電話" (Contact Phone), "聯絡電郵" (Contact Email), "優惠密碼" (Promotional Password), and "個人近照" (Personal Photo). Messages include "Enter an email address" pointing to the email field, "Select a file" pointing to the file upload button, and "Choose File no file selected" pointing to the file input area.

HTML練習4

按照以下指示製作網頁：

1. 網頁標題title為「HTML表單與驗證」
2. 內容標題為「最喜愛精品品牌投票：」
3. 使用form元件包著所有輸入欄位
4. 除了兩個按鈕，其他全部都是必填欄位
5. 稱謂中把「小姐」預設為已選
6. 把稱謂的所有選擇設為組合「title」
7. 最愛品牌中把「Louis Vuitton」預設為已選
(其他為Gucci、Chanel、Hermes、Dior)
8. 本月購買量只接受不少於0的數字數值(min)
9. 本月購買量的預設值為「1」
10. 圖片分享接受多個檔案附載
11. 圖片分享只接受圖像檔「image/*」
12. Reset按鈕的值為「清空」
13. Submit按鈕的值為「送出」



HTML 表單與驗證

最喜愛精品品牌投票：

名稱

稱謂 小姐 女士 先生

最愛品牌

最近購買日期

本月購買量

最愛的色調

圖片分享 3 files

除了本教材，大家更可參考 https://www.w3schools.com/tags/tag_input.asp 來完成本練習

課題二：

CSS樣式表與程式外觀設計

CSS套用方式 (行內套用)

行內套用 (Inline)

直接套用到HTML的Element(元件)中，好處是直接寫到Element中，不需額外找地方編寫。

```
<span style="color:red;font-size:16pt;">This is font size 16.</span>
```

CSS套用方式 (嵌入套用)

嵌入套用 (Embed)

套用到HTML的head中，好處是統一所有CSS到head中方便修改和重覆使用。
注意：<style>元素內的內容是特別內容(即CSS樣式表)，並不使用HTML的寫法。

```
<head>
  <style>
    div{
      background-color: red;
    }
  </style>
</head>

<body>
  <div>
    背景顏色是紅色
  </div>
</body>
```

CSS套用方式 (外部連接套用)

外部連接套用 (External Link)

使用外部的CSS。通常用於統一所有HTML檔案的CSS。

```
<head>
```

```
<link rel="stylesheet" type="text/css" href="external-stylesheet.css">
```

```
</head>
```



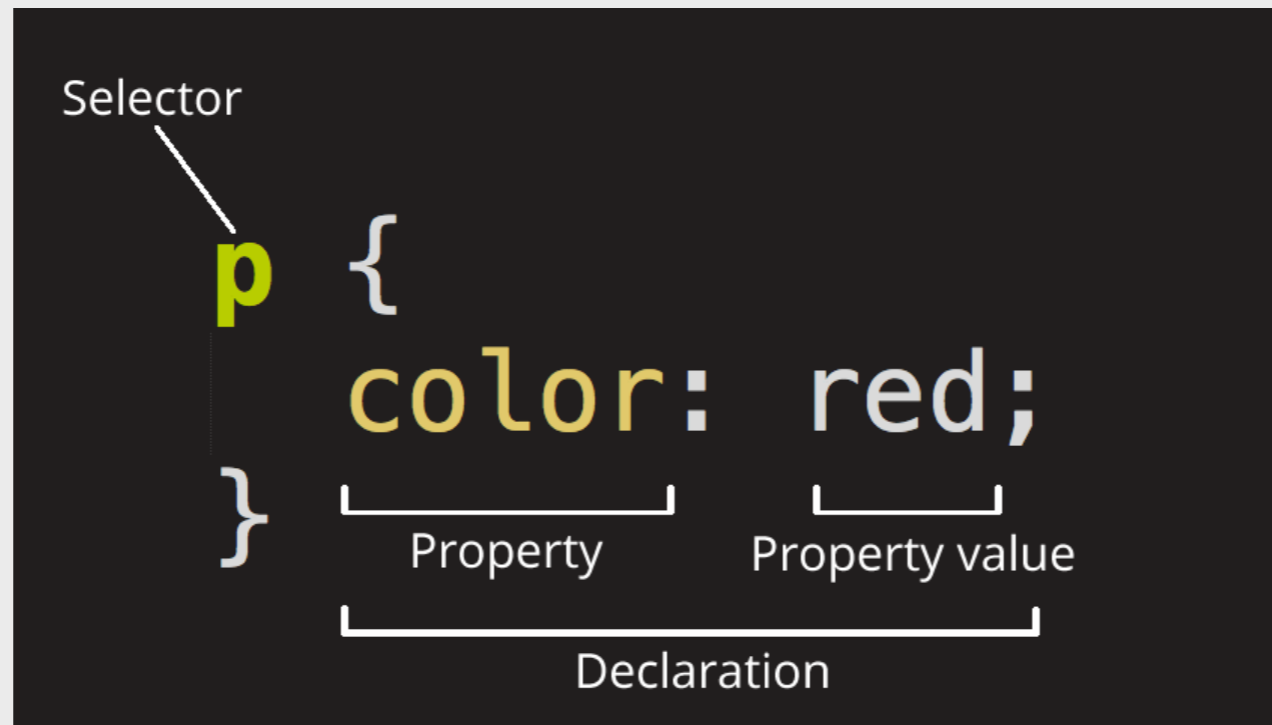
CSS常用單位

單位名稱	解說	例子
pt	Points，點，可使用小數點，1 pt = 1/72 inch	8pt / 13pt / 23.9pt
px	Pixels，像素，小數點無效，顯示屏上的一點	3px / 16px
vw / vh	顯示屏比例，可使用小數點，0為最少，100為最大 (View width / View height)	100vw / 39vh / 51.6vw / 66.1vh
%	相對於父元件(父標)的比例	7% / 33% / 56.2% / 2.1%
em	相對於每個子元素透過「倍數」乘以父元素的px值。	1em / 1.2em
rem	相對於每個子元素透過「百分比」乘以父元素的px值。	1rem / 1.2rem
calc	此函數允許在聲明CSS屬性值時執行一些計算。	width: calc(100% - 80px);

CSS常用Style(樣式)

樣式名稱	解說	例子
background-color	背景顏色	background-color: rgba(10, 20, 72, 0.3);
background-image	背景圖片	background-image: url('media/abc.jpg');
width	闊度	width: 100vw;
height	高度	height: 66%;
color	文字顏色	color: gold;
font-size	文字大小	font-size: 18pt;
font-family	文字字體	font-family: 'Helvetica';
padding	邊界留空	padding: 5px 6px;
display	顯示方式	display: none;
overflow-x	X軸捲動	overflow-x: hidden;
overflow-y	Y軸捲動	overflow-y: auto;
position	位置方式	position: absolute;
top	Y1定位	top: 0%;
left	X1定位	left: 13pt;
bottom	Y2定位	bottom: 3px;
right	X2定位	right: 9vh;
border	邊框	border: 2px solid rgb(10,10,10);
border-radius	邊框弧度	border-radius: 6pt;
text-align	水平排列	text-align: center;
vertical-align	垂直排列 (只適用於表格的td中)	vertical-align: middle;
border-spacing	邊框間距 (只適用於表格中)	border-spacing: 12pt;
text-decoration	文字底線	text-decoration:underline;

解析CSS Ruleset



整個架構我們稱為規則集(rule set)，或是簡稱為規則(rule)也可以。

選擇器(Selector)：在這個規則的最前頭為 HTML 的元素名。它將決定你 HTML 裡什麼元素將被你接下來的設定影響(在這個範例中,就是 段落元素 p)。若要改變欲影響的元素，只要更改選擇器就行了。

宣告(Declaration)：單一個規則，例如 color: red; 指定了這個元素的呈現樣貌。

屬性(Properties)：修改屬性是改變你 HTML 元素的一種方法。(在這範例中, color 是段落(p)元素的一種屬性。)在 CSS 中，你可以選擇哪些屬性用來影響rule。

屬性值(Property value)：屬性值 就是位於屬性右邊，在冒號(:)之後，從眾多的可能樣式選出一個給予屬性(範例中就是從眾多的 color 樣式中選出 red)。

歸納CSS樣式屬性

歸納 CSS 樣式屬性			
由大而小、 由外而內拆分 	層級	CSS 相關屬性	說明
	排版	(定位相關) position, top/right/bottom/left 等	影響區塊在畫面上的位置。
		(畫面流相關) display grid/flex、 block/inline-block 等。	影響區塊內的水平、垂直排版。
	區塊	width/height、background、box model、padding/margin等。	用於定義區塊本身的樣式呈現。
	內容	<ul style="list-style-type: none">font：字型、字體大小等等的樣式定義。字體排版 (line-height、text-indent 等)：有關行距、縮相關的結構向樣式定義。color: 字體顏色定義。	用於定義內容文案的樣式。
動畫	(動畫的參數定義) keyframes、animation、 transition 等。	可以用來定義動態整體脈絡中的樣式變化，包含動態需要相關的屬性定義、秒數。	

來源出自：<https://tw.alphacamp.co/blog/css-guide-box-model>

CSS 「化妝術」 - 方式

CSS有很多選擇「化妝」的對象，以下是4種極為常用的方法

1) 對所有同款元素(Tag)

正名: Element Type Selector

2) 對一個類別(Class)

正名: Class Selector

3) 對某一單獨元素

正名: ID Selector

4) 組合寫法

正名: Descendant Selector (組合選擇器)

CSS化妝術 - 對所有同款元素

Element Type Selector
(即所有同款的元素皆會化這個妝)

語法

```
元件名稱{  
  屬性: 設定值;  
}
```

實例(CSS)

```
div{  
  color: red;  
  background-color: gold;  
}
```

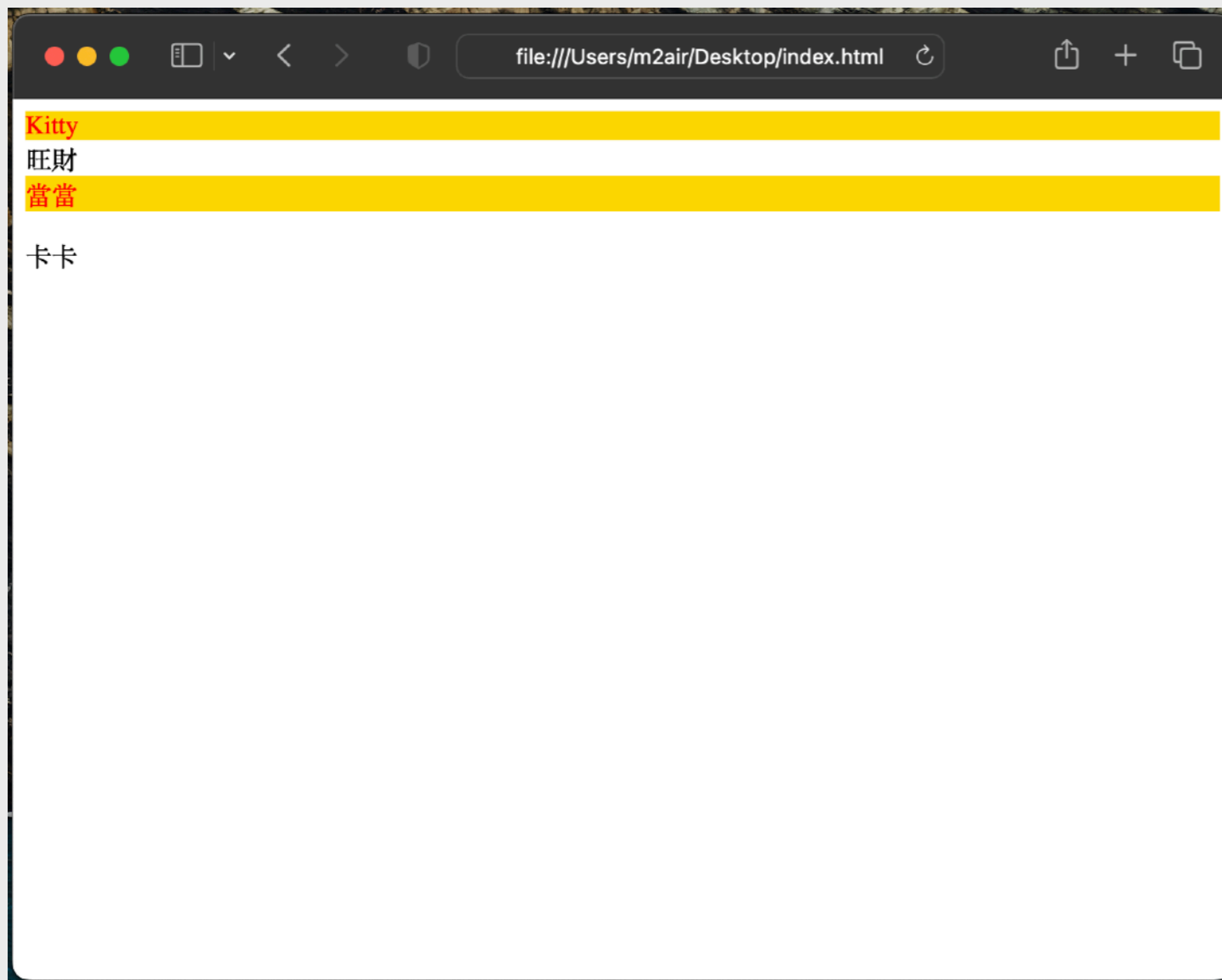
實例(HTML)

```
<div>Kitty</div>  
<span>旺財</span>  
<div>當當</div>  
<p>卡卡</p>
```

例子中，只有屬於 div 的元素才會化這個妝

CSS化妝術 - 對所有同款元素

Element Type Selector
(即所有同款的元素皆會化這個妝)



```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>CSS 範例</title>
  <style>
    div {
      color: red;
      background-color: gold;
    }
  </style>
</head>

<body>
  <div>Kitty</div>
  <span>旺財</span>
  <div>當當</div>
  <p>卡卡</p>
</body>

</html>
```

CSS化妝術 - 對一個類別

Class Selector

(即所有同類別的元素皆會化這個妝，用 . 號定義)

語法

```
.類別名稱{  
  屬性: 設定值;  
}
```

實例(CSS)

```
.Dog{  
  color: brown;  
  font-size: 20pt;  
}
```

實例(HTML)

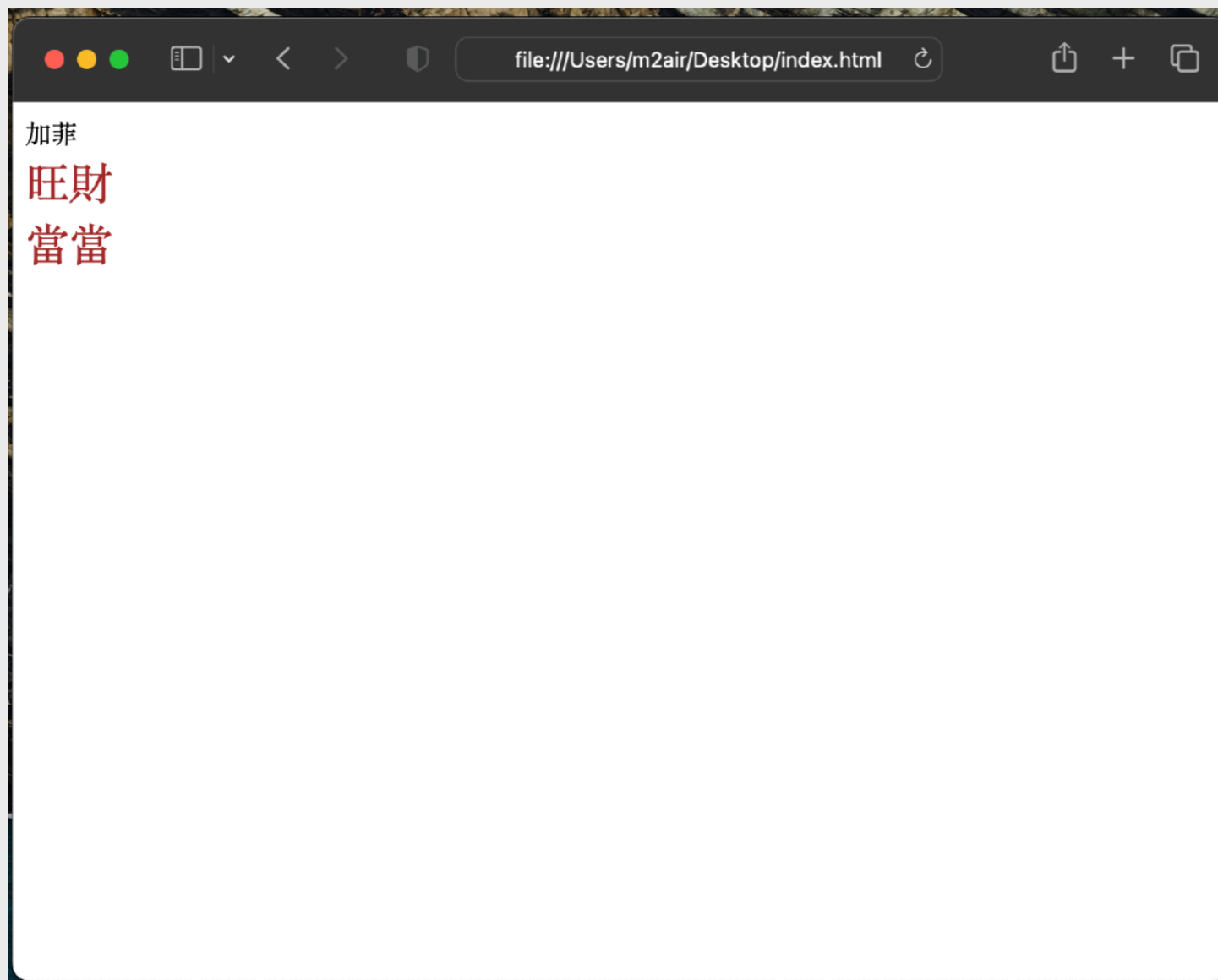
```
<div class="Cat">加菲</div>  
<div class="Dog">旺財</div>  
<div class="Dog">當當</div>
```

例子中，只有元素標識類別為 Dog 的那些元素才會化這個妝

CSS化妝術 - 對一個類別

Class Selector

(即所有同類別的元素皆會化這個妝，用 . 號定義)



```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>CSS 範例</title>
  <style>
    .Dog {
      color: brown;
      font-size: 20pt;
    }
  </style>
</head>

<body>
  <div class="Cat">加菲</div>
  <div class="Dog">旺財</div>
  <div class="Dog">當當</div>
</body>

</html>
```

CSS化妝術 - 對某一單獨元素

ID Selector

(只有一個指定元素會化這個妝，用 # 號定義)

語法

```
#元素標識{  
  屬性: 設定值;  
}
```

實例(CSS)

```
#img_01{  
  width: 275px;  
  border-radius: 15pt;  
  border: 2pt solid red;  
}  
#img_02 {  
  width: 160px;  
}
```

實例(HTML)

```
<div>Test</div>  
  

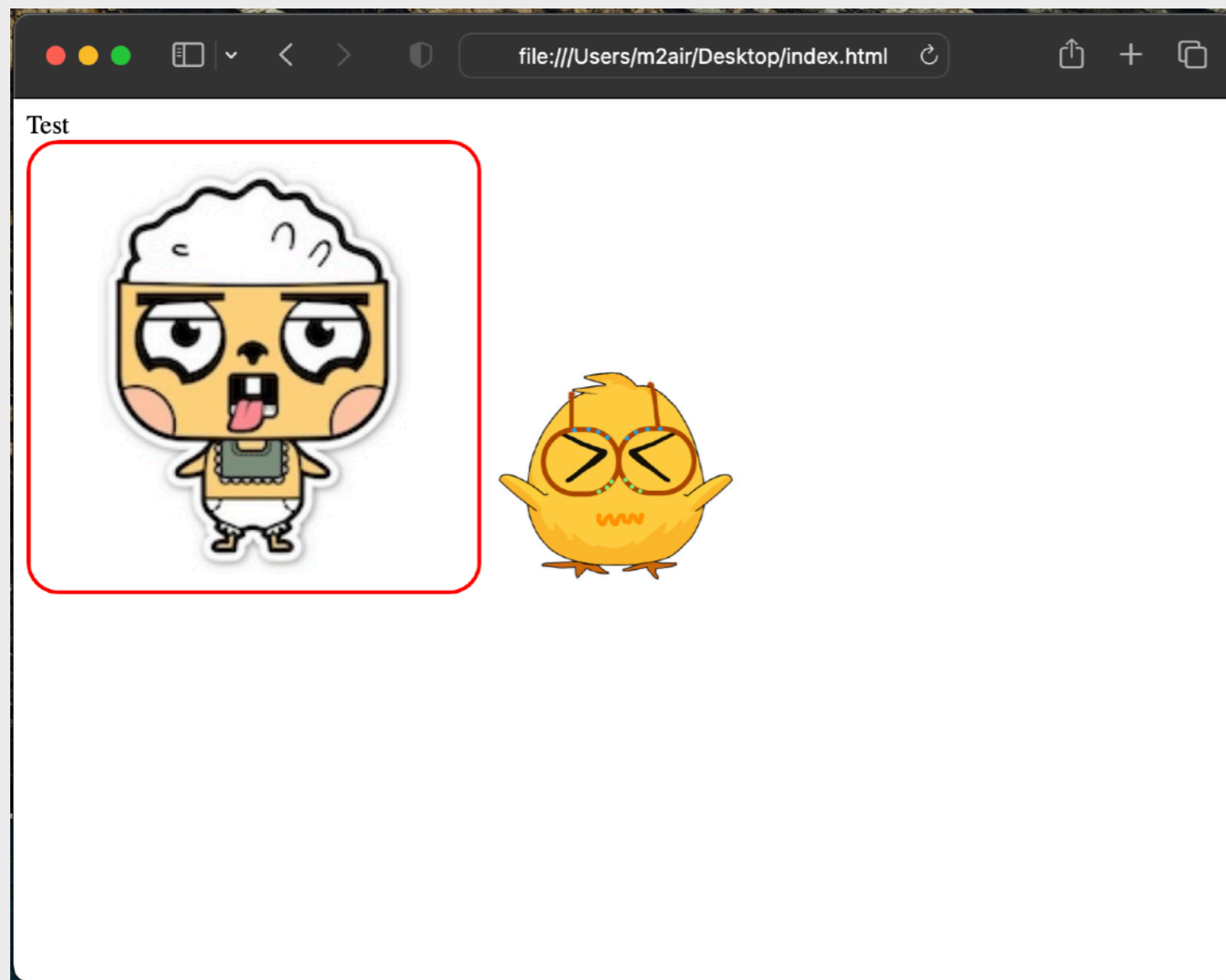
```

例子中，只有元素標識(「身分證」)為 img_01 的那個元素才會化這個妝

CSS化妝術 - 對某一單獨元素

ID Selector

(只有一個指定元素會化這個妝，用 # 號定義)



```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>CSS 範例</title>
  <style>
    #img_01 {
      width: 275px;
      border-radius: 15pt;
      border: 2pt solid red;
    }
    #img_02 {
      width: 160px;
    }
  </style>
</head>

<body>
  <div>Test</div>
  
  
</body>

</html>
```


CSS化妝術 - 組合寫法

Descendant Selector

語法

```
元素標識 (內的)元素標識{  
  屬性: 設定值;  
}
```

實例(CSS)

```
#div_01 img{  
  width: 275px;  
  border-radius: 15pt;  
  border: 2pt solid red;  
}
```

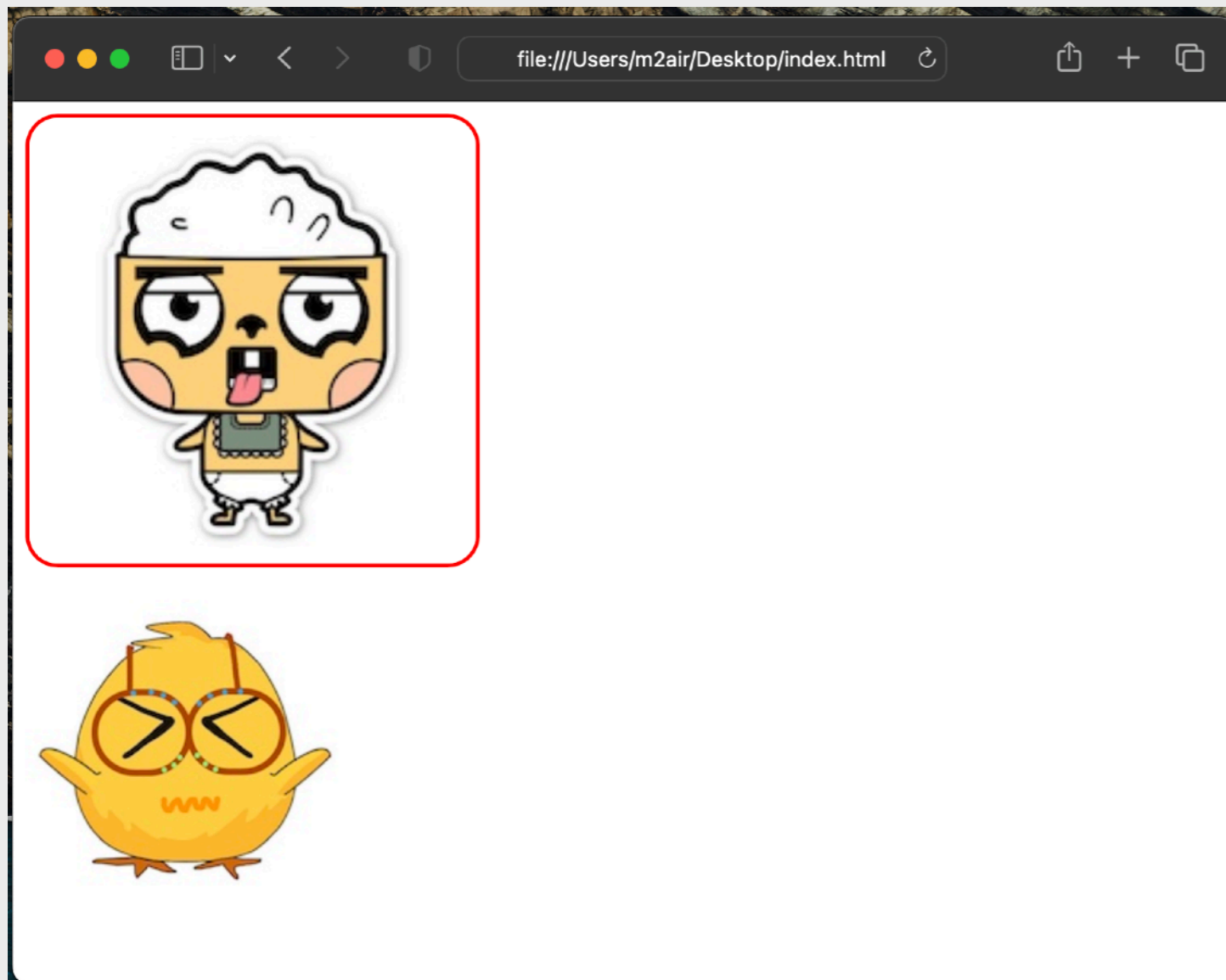
實例(HTML)

```
<div id="div_01" >  
  
</div>  

```

CSS化妝術 - 組合寫法

Descendant Selector



```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
  <title>CSS 範例</title>
  <style>
    #div_01 img {
      width: 275px;
      border-radius: 15pt;
      border: 2pt solid red;
    }
  </style>
</head>

<body>
  <div id="div_01">
    
  </div>
  
</body>

</html>
```

CSS化妝術 - 其他常用對象選擇方法

選擇器	示例	示例說明
element,element	div,p	選擇所有<div>元素和<p>元素
element>element	div>p	選擇所有父級是 <div> 元素的 <p> 元素
element+element	div+p	選擇所有緊跟在 <div> 元素之後的第一個 <p> 元素
[attribute]	[target]	選擇所有帶有target屬性元素
:hover	a:hover	選擇鼠標在鏈接上面時
:first-child	p:first-child	選擇所有父元素的首個子級 <p> 元素
:last-child	p:last-child	選擇所有父元素的最後一個子級 <p> 元素
:required	input:required	選擇所有必須填寫的 <input> 的元素
:valid	input:valid	選擇所有匹配輸入值為合法的 <input> 元素
:invalid	input:invalid	選擇所有匹配輸入值為非法的 <input> 元素
::selection	::selection	選擇所有被用戶選中(反白)的元素或部分

可查看更多CSS選擇器: <https://www.runoob.com/cssref/css-selectors.html>

CSS練習1

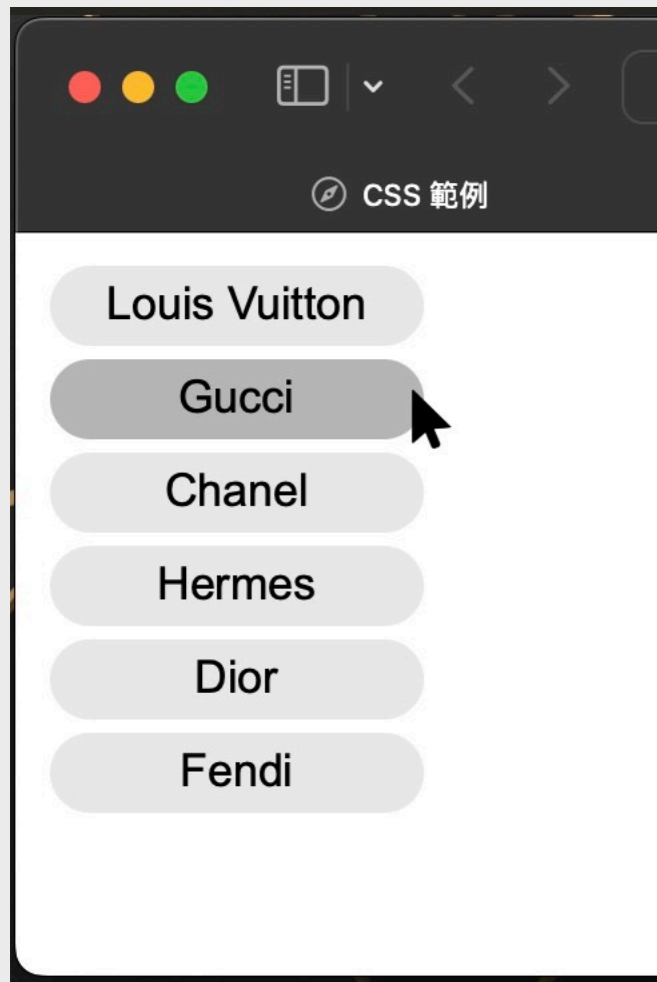
```
<!DOCTYPE html>
<html>
  <head>
    <title>Animals</title>
  </head>
  <body>
    <div class="MyPets" id="beeno">Dog</div>
    <div class="MyPets" id="kitty">Cat</div>
    <div class="OnlineAdopted">Penguin</div>
    <div>Panda</div>
    <div class="OnlineAdopted">Koala</div>
    <div id="ada">Monkey</div>
  </body>
</html>
```

1. 把所有的div的font-size設定為18pt
2. 把所有屬於MyPets類別的div的color設定為green
3. 把所有屬於OnlineAdopted類別的div的color設定為gold
4. 把id為beeno的div加上底線(text-decoration設定為underline)
5. 把id為kitty的div的背景顏色(background-color)設定為#73FDFF
6. 把body的字體粗度(font-weight)設定為粗體(bold)
7. 把所有屬於MyPets或OnlineAdopted類別的div的border設定為2px solid #922222及padding為12px
8. 把所有屬於MyPets或id為ada的div的margin-left設定為5.5vw

CSS化妝術 - :hover

透過加入: hover選擇器，令滑鼠指向特定元件時，發生特別效果

以本例子為例，當鼠標移到某「#table_1 td」時，便會轉變其底色為深灰色。



CSS:

```
body{
  font-family: Arial, Helvetica, sans-serif;
}
#table_1{
  width: 150px;
  border-spacing: 5px;
  font-size: 13pt;
  text-align: center;
}
#table_1 td{
  width: 100%;
  background-color: rgb(233,233,233);
  border-radius: 15px;
  padding: 5px;
}
#table_1 td:hover{
  background-color: rgb(183,183,183);
}
```

HTML:

```
<table id="table_1">
  <tr><td>Louis Vuitton</td></tr>
  <tr><td>Gucci</td></tr>
  <tr><td>Chanel</td></tr>
  <tr><td>Hermes</td></tr>
  <tr><td>Dior</td></tr>
  <tr><td>Fendi</td></tr>
</table>
```

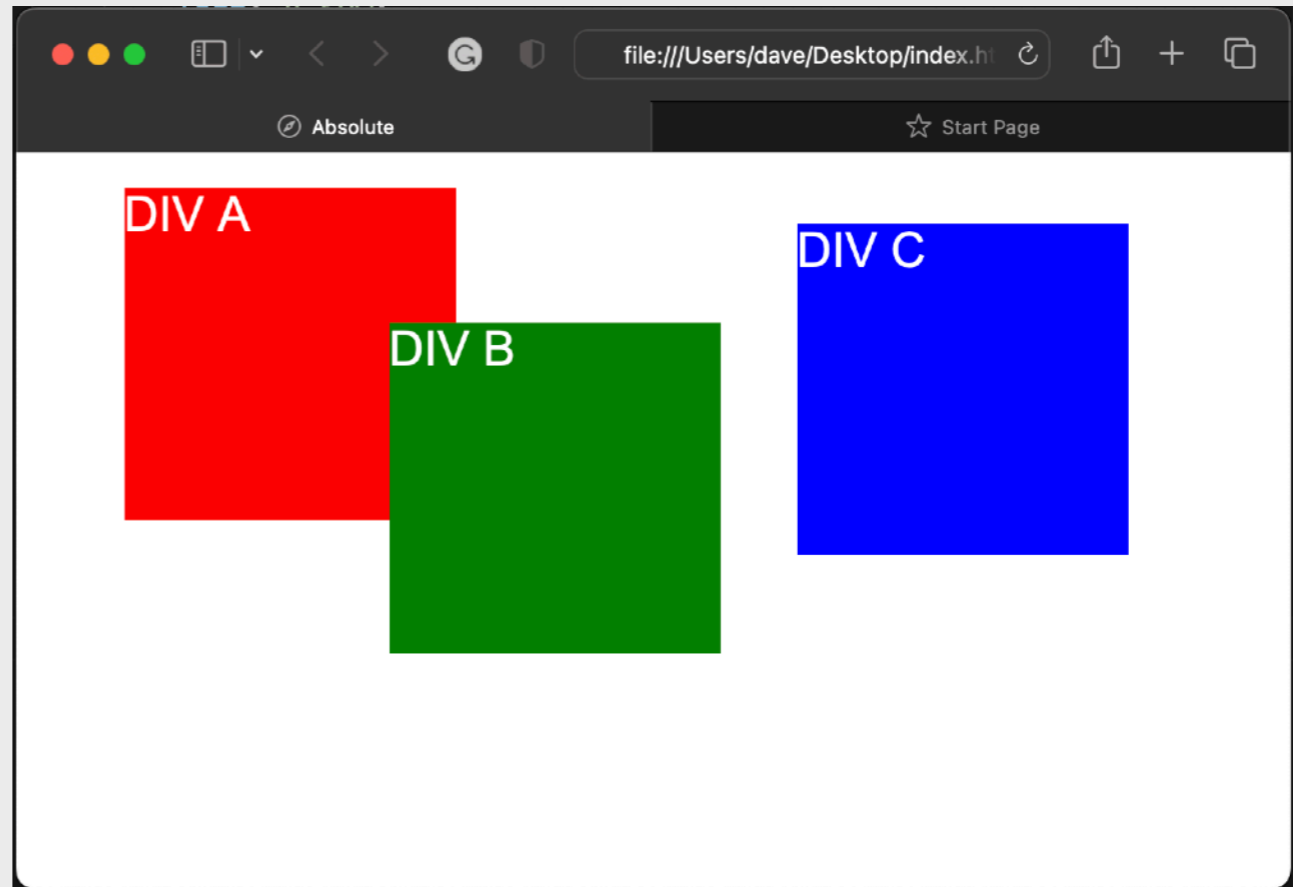
CSS Position的Absolute與Fixed

CSS:

```
div{
  width: 26vw;
  height: 26vw;
  color: white;
  font-size: 20pt;
  font-family: Arial, Helvetica, sans-serif;
}
#div_a{
  position: absolute;
  top: 5vh;
  left: 8.5vw;
  background-color: red;
}
#div_b{
  position: absolute;
  top: 23.2vh;
  left: 29.3vw;
  background-color: green;
}
#div_c{
  position: absolute;
  top: 9.8vh;
  left: 61.3vw;
  background-color: blue;
}
```

HTML:

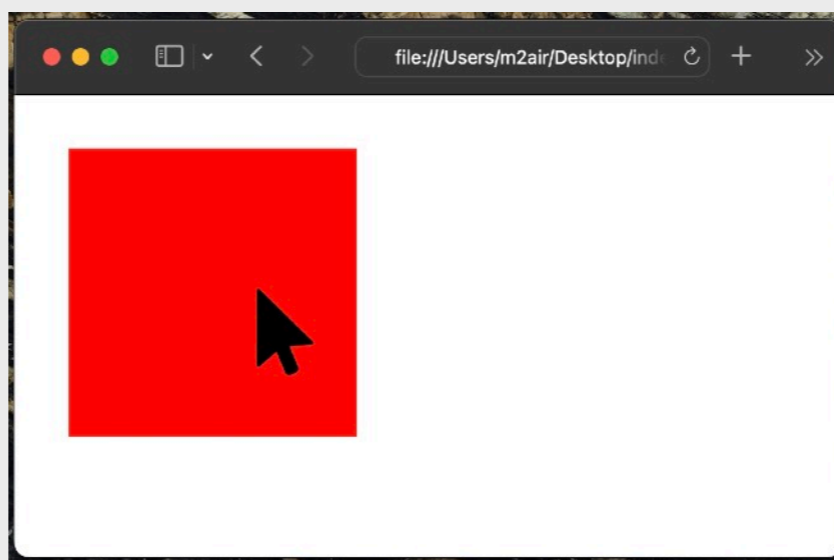
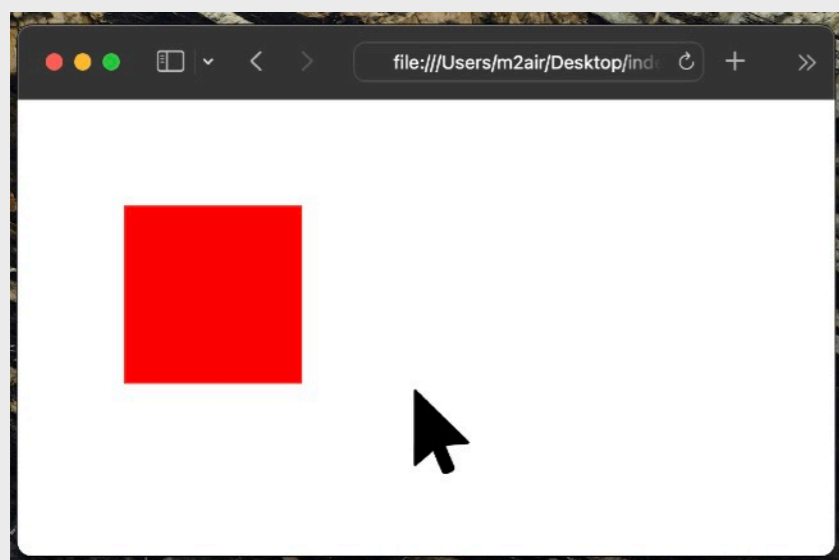
```
<div id="div_a">DIV A</div>
<div id="div_b">DIV B</div>
<div id="div_c">DIV C</div>
```



透過把position設定成為**absolute(絕對定位)**，便可以把HTML元件根據著我們給予的座標(「top或bottom」與「left或right」)而定位。當然，也別忘記設定元件的width和height。其實還有一種與absolute十分相似的，叫作**fixed(固定定位)**，其特性為不受卷動(scrolling)限制，即使卷動網頁，元件也會停留在同一位置，常見用途為網頁上的廣告。

CSS動畫 - transition (1)

Transition是CSS的其中一種動畫效果，通常用來對外觀上的轉換給予回饋，例如當鼠標指向元件時，轉變外觀以讓用戶辨別出鼠標已指向了元件。簡單說，例如在hover時會需要外觀上的轉換給予回饋，不管是顏色、形狀、大小等等，加上了transition，可以補足轉換中間的過程，形成補間動畫，看起來會更生動。以下例子為元件加入了對發生transform狀況時的transition進行方法，並在hover時更改其transform (scale)的樣式。



```
#div_a{
  position: absolute;
  top: 75px;
  left: 75px;
  width: 125px;
  height: 125px;
  background-color: red;
  transition: transform 0.2s ease;
}
#div_a:hover{
  transform: scale(1.6);
}
```

上圖正方形的是ID為div_1的元件，它設置了transition的屬性，並定義了當遇到transform有所改變時，並動作持續時間為0.2秒，並以ease(緩入中間快緩出)的動畫方式而進行轉變。

而div_1設置了:hover當鼠標指向它時，便會觸發了:hover選擇器，而:hover裡則定義了transform的屬性為放大成1.6倍(「scale(1.6)」)。

所以，當鼠標已指向元件時，便會改變了元件的大小(scale)，並產生了動畫效果。

CSS動畫 - transition (2)

透過transition語法，我們可設定其對應效果、持續時間及出現效果的方法。

Transition屬性的寫法：

transition: transition-property | transition-duration | transition-timing-function | transition-delay;

Transition出現效果的方法：

名稱	說明
linear	均速
ease	緩入中間快緩出
ease-in	緩入
ease-out	緩出
ease-in-out	緩入緩出(對比於ease較平緩)

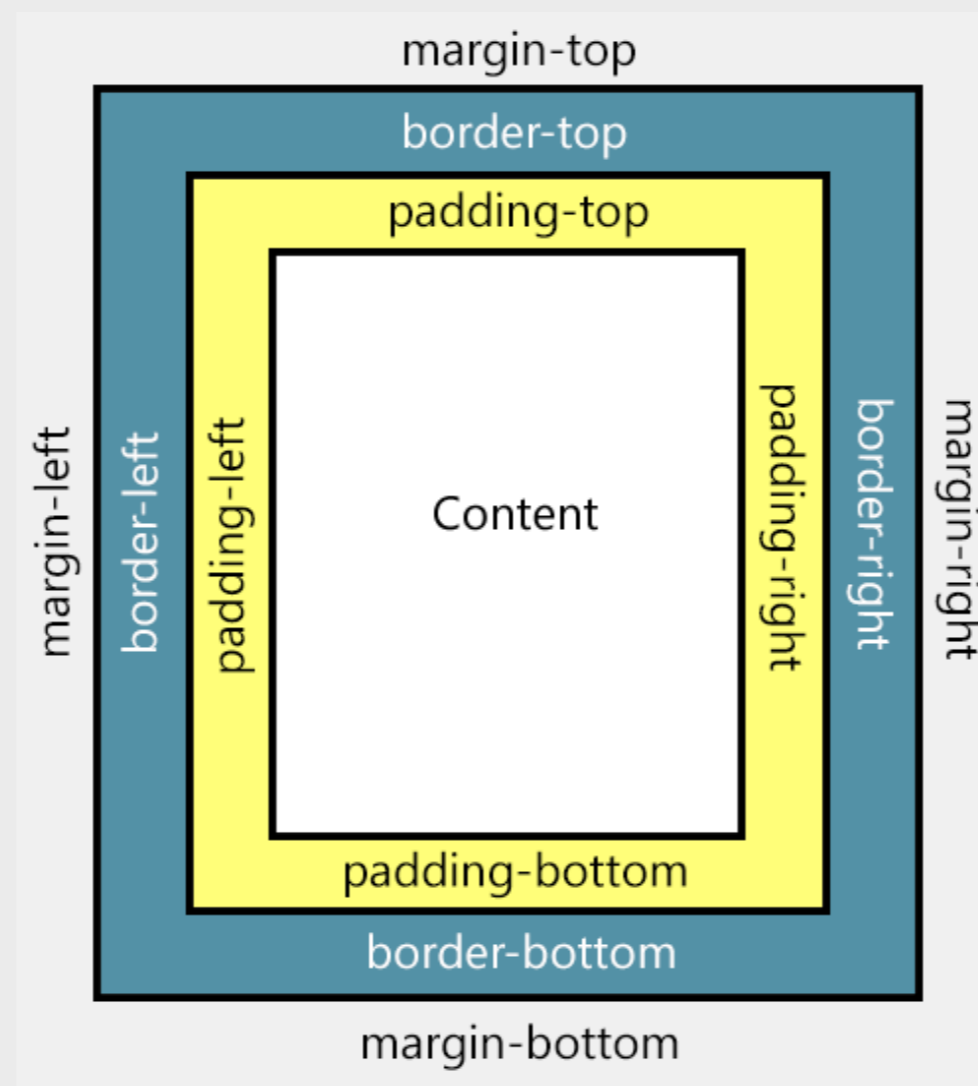
CSS中的transition屬性像background屬性一樣可以分開寫也可以縮寫，但也一樣有順序性，transition用到transition-duration及transition-delay，兩個屬性都是時間單位，因此容易搞混：

- 1.如果出現單寫：transition: width 1s的話，1s指的是transition-duration
- 2.寫了兩個時間單位數值：前面出現的是transition-duration，後出現是transition-delay

Transition預設對所有可套用屬性進行，但也可以針對單個或多個，如使用多個寫法，則使用逗號將不同屬性隔開即可，如：transition: width 1s, height 2s;

CSS Box Model (1)

CSS 排版有一個很重要的觀念：Box Model。它描述了元素之間的彼鄰關係，同時也左右了我們是否能夠成功透過 CSS，完成整個頁面的呈現。Box Model 的意思，直譯英翻中可以解釋成，方塊模組，也就是說，每一個元素我們都可視它為一個 Box（方塊）。一個 Box 由以下屬性組成：margin、padding、border、content。而一個 Box 的實際寬度（高度）是由 padding + border + width (height) 所組成，如下圖：



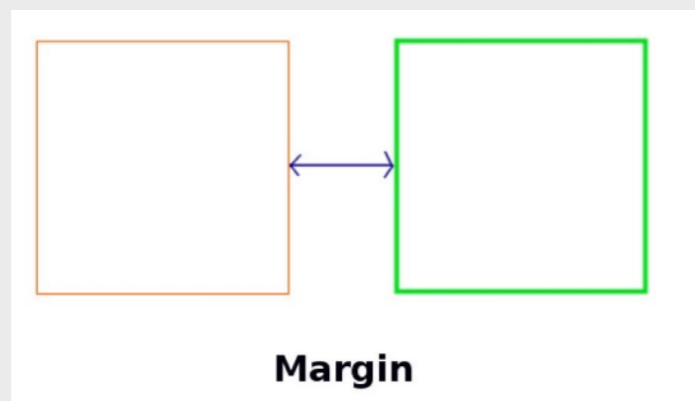
CSS Box Model (2)

margin (外間距)

margin 是負責調整元素與元素之間的邊界間距，屬於元素外部的邊界調整。

• 屬性介紹

- margin-top
設定元素與元素之間的上邊界間距。
- margin-right
設定元素與元素之間的右邊界間距。
- margin-bottom
設定元素與元素之間的下邊界間距。
- margin-left
設定元素與元素之間的左邊界間距。

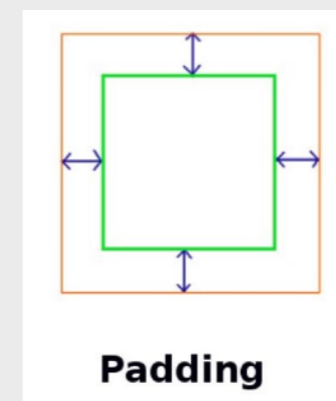


padding (內間距)

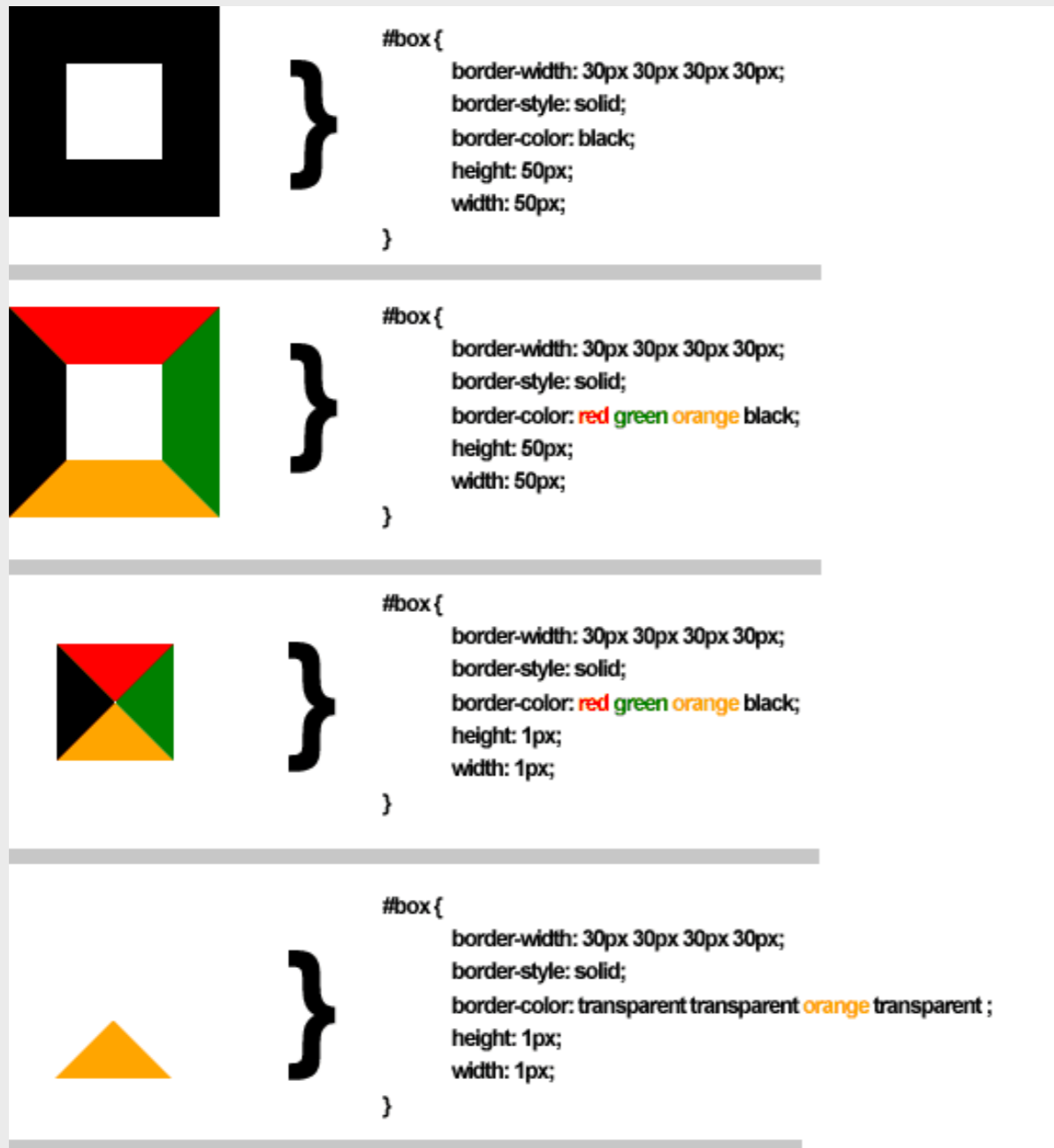
padding 是負責調整元素內所有內容與元素自身的邊界間距，屬於元素內部的邊界調整。

• 屬性介紹

- padding-top
設定元素內容與元素自身上邊界的間距。
- padding-right
設定元素內容與元素自身右邊界的間距。
- padding-bottom
設定元素內容與元素自身下邊界的間距。
- padding-left
設定元素內容與元素自身左邊界的間距。



CSS Box Model (3)



```
#box{  
  border-width: 30px 30px 30px 30px;  
  border-style: solid;  
  border-color: black;  
  height: 50px;  
  width: 50px;  
}
```

```
#box{  
  border-width: 30px 30px 30px 30px;  
  border-style: solid;  
  border-color: red green orange black;  
  height: 50px;  
  width: 50px;  
}
```

```
#box{  
  border-width: 30px 30px 30px 30px;  
  border-style: solid;  
  border-color: red green orange black;  
  height: 1px;  
  width: 1px;  
}
```

```
#box{  
  border-width: 30px 30px 30px 30px;  
  border-style: solid;  
  border-color: transparent transparent orange transparent;  
  height: 1px;  
  width: 1px;  
}
```

也可以將上述的屬性合併寫成 border 即可，代表四個方向的邊框都會套用一樣的樣式，寫法如下：

```
border: 邊框粗細 邊框顏色 邊框樣式;
```

內容出處：<https://ithelp.ithome.com.tw/articles/10205322>

border

border 是元素最外層的邊界，常見的元素外框設定就是此設定。

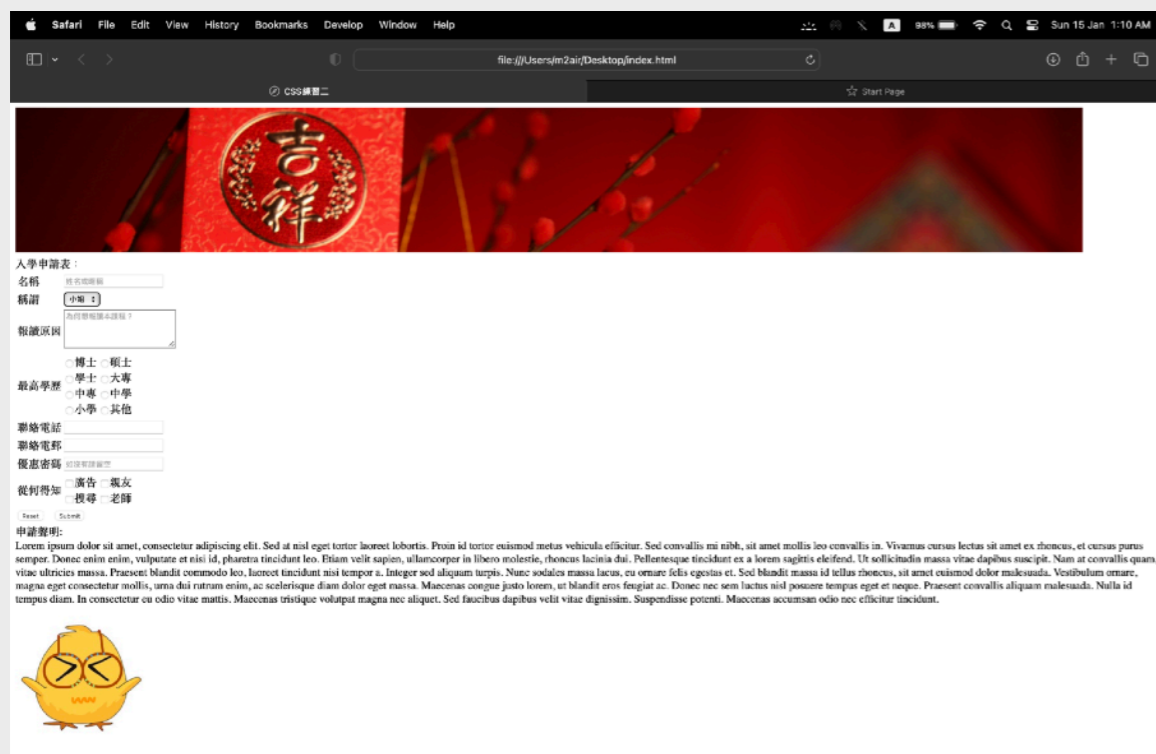
• 邊界樣式介紹

- border-width
設定邊框寬度大小。
- border-style
設定邊框樣式，常見的有 實線(solid)、虛線(dashed)。
- border-color
設定邊框顏色。

• 屬性介紹

- 底下屬性後方所接的值為上面提到的樣式，順序為 border-width、border-color、border-style，中間用空格隔開。
- border-top
設定元素自身的上邊框。
- border-right
設定元素自身的右邊框。
- border-bottom
設定元素自身的下邊框。
- border-left
設定元素自身的左邊框。

CSS練習2 (1)



變成
→



使用下兩頁的HTML碼去製作一個網頁，然後加入CSS以使到網頁更為美觀，以及擁有動畫效果。

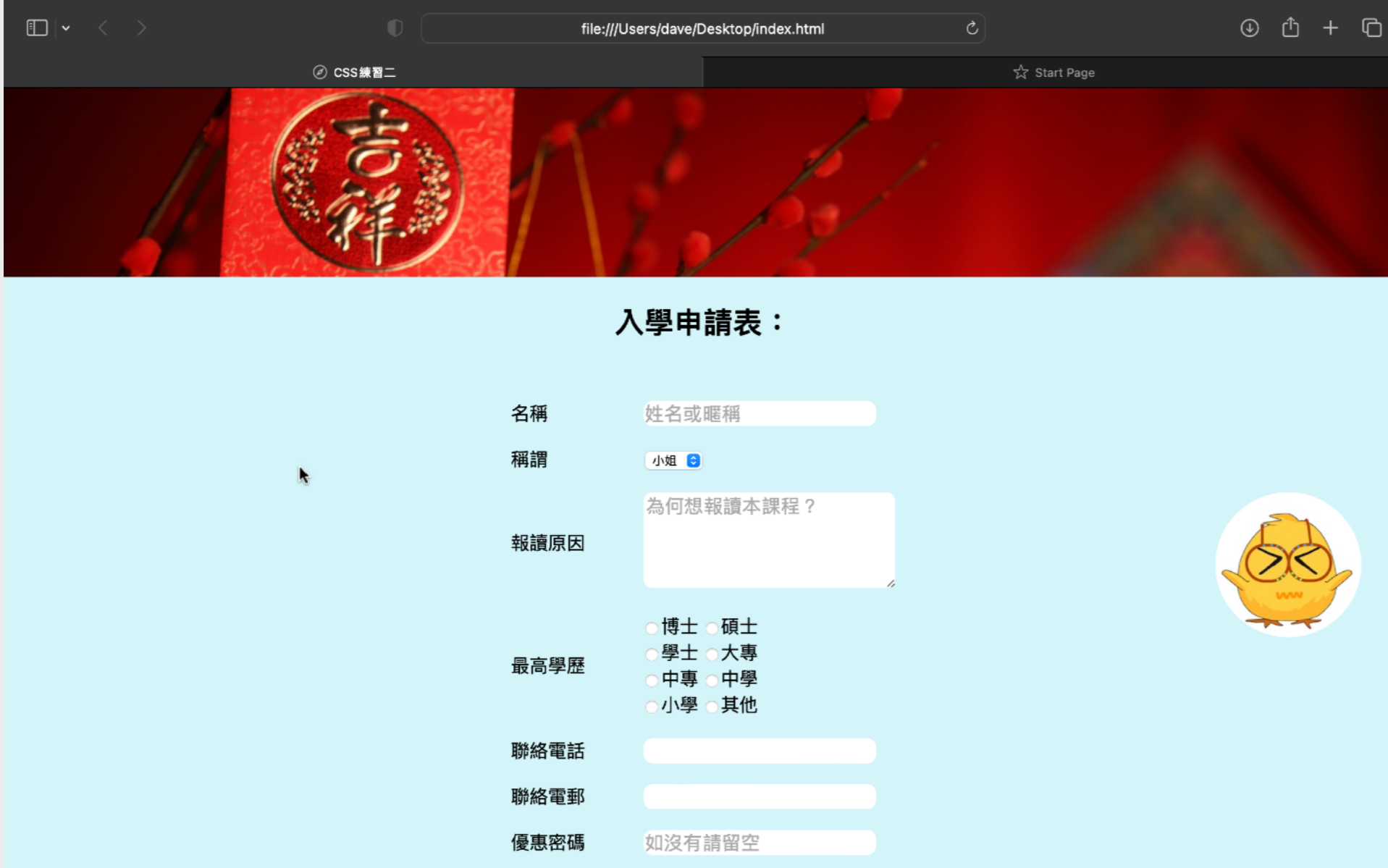
CSS練習2 (3)

繼:

```
<div id="div_message">
  <div>申請聲明:</div>
  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed at nisl eget tortor laoreet lobortis. Proin id tortor euismod
  metus vehicula efficitur. Sed convallis mi nibh, sit amet mollis leo convallis in. Vivamus cursus lectus sit amet ex rhoncus, et
  cursus purus semper. Donec enim enim, vulputate et nisi id, pharetra tincidunt leo. Etiam velit sapien, ullamcorper in libero
  molestie, rhoncus lacinia dui. Pellentesque tincidunt ex a lorem sagittis eleifend. Ut sollicitudin massa vitae dapibus suscipit.
  Nam at convallis quam, vitae ultricies massa. Praesent blandit commodo leo, laoreet tincidunt nisi tempor a. Integer sed
  aliquam turpis. Nunc sodales massa lacus, eu ornare felis egestas et. Sed blandit massa id tellus rhoncus, sit amet euismod
  dolor malesuada. Vestibulum ornare, magna eget consectetur mollis, urna dui rutrum enim, ac scelerisque diam dolor eget
  massa. Maecenas congue justo lorem, ut blandit eros feugiat ac. Donec nec sem luctus nisl posuere tempus eget et neque.
  Praesent convallis aliquam malesuada. Nulla id tempus diam. In consectetur eu odio vitae mattis. Maecenas tristique volutpat
  magna nec aliquet. Sed faucibus dapibus velit vitae dignissim. Suspendisse potenti. Maecenas accumsan odio nec efficitur
  tincidunt.
  </div>
  
</form>
</body>

</html>
```

CSS練習2 (4)



file:///Users/dave/Desktop/index.html

CSS練習二 Start Page

入學申請表：

名稱

稱謂


報讀原因

最高學歷 博士 碩士 學士 大專 中專 中學 小學 其他

聯絡電話

聯絡電郵

優惠密碼



注意：

1. 所有CSS皆添加在head元件裡的style元件中
2. 背景顏色為D3F5FA
3. 字體使用
Arial, Helvetica, sans-serif
4. body沒有任何外邊距
5. 各處的外邊距與內邊距
6. 「入學申請表：」的字體為粗體，大小為20pt
7. 輸入元件的邊框圓角
8. 輸入元件的字體大小為13pt
9. 輸入元件沒有邊框
10. 小雞為固定定位，
不受卷動(scrolling)限制
11. 小雞的邊框圓角為50%

CSS練習2 (5)

入學申請表：

名稱

稱謂

報讀原因

最高學歷 博士 碩士
 學士 大專
 中專 中學
 小學 其他

聯絡電話

聯絡電郵

優惠密碼



注意：

12.當鼠標指向小雞時，小雞便會以0.2秒持續時間和ease動畫效果來放大(scale)成1.3倍

CSS練習2 (6)

file:///Users/dave/Desktop/index.html

CSS練習二 Start Page

最高學歷 學士 大專
 中專 中學
 小學 其他

聯絡電話


聯絡電郵

優惠密碼

從何得知 廣告 親友
 搜尋 老師

申請聲明:

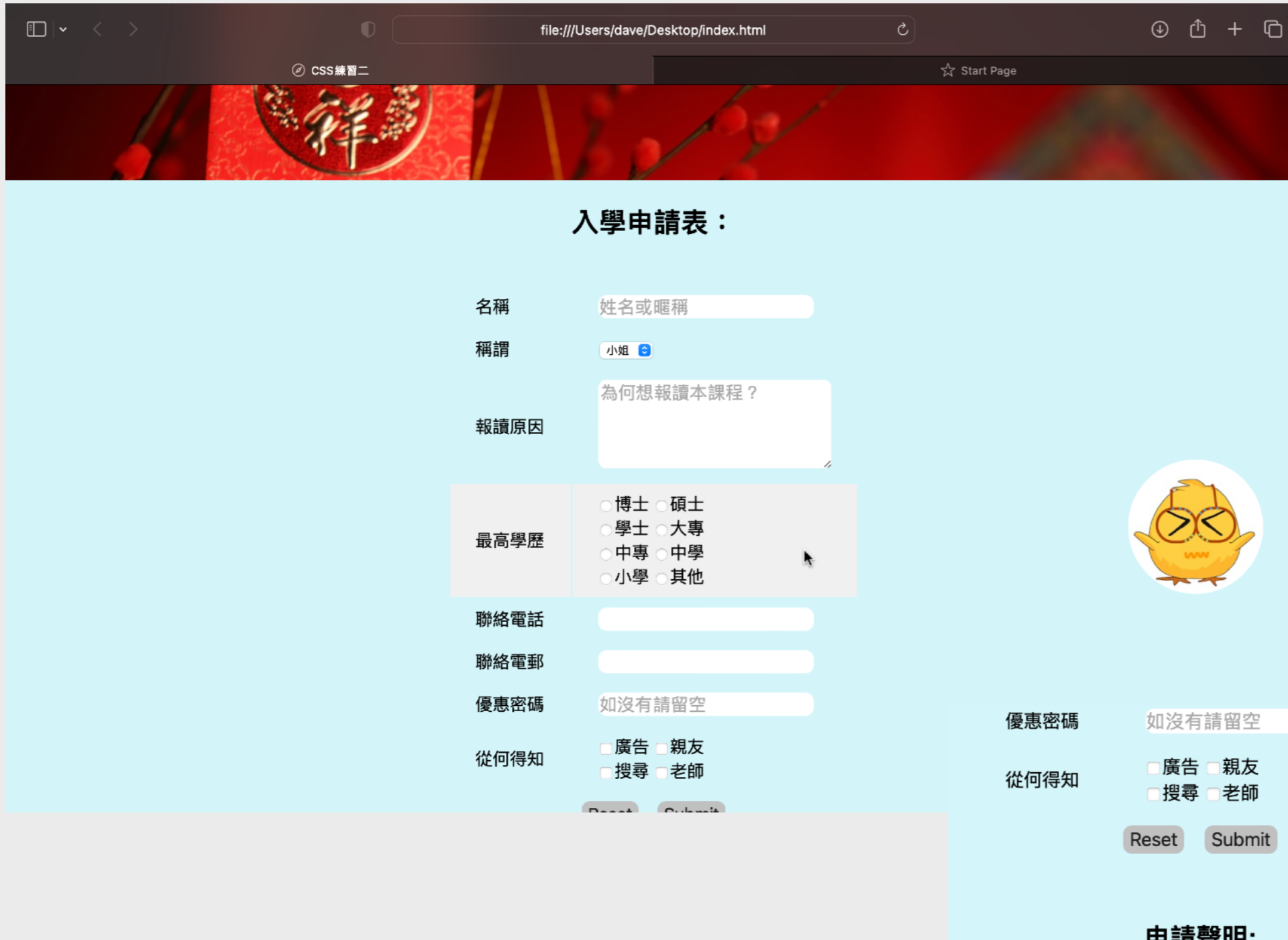
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed at nisl eget tortor laoreet lobortis. Proin id tortor euismod metus vehicula efficitur. Sed convallis mi nibh, sit amet mollis leo convallis in. Vivamus cursus lectus sit amet ex rhoncus, et cursus purus semper. Donec enim enim, vulputate et nisi id, pharetra tincidunt leo. Etiam velit sapien, ullamcorper in libero molestie, rhoncus lacinia dui. Pellentesque tincidunt ex a lorem sagittis eleifend. Ut sollicitudin massa vitae dapibus suscipit. Nam at convallis quam, vitae ultricies massa. Praesent blandit commodo leo, laoreet tincidunt nisi tempor a. Integer sed aliquam turpis. Nunc sodales massa lacus, eu ornare felis egestas et. Sed blandit massa id tellus rhoncus, sit amet euismod dolor malesuada. Vestibulum ornare, magna eget consectetur mollis, urna dui rutrum enim, ac scelerisque diam dolor eget massa. Maecenas congue justo lorem, ut blandit eros feugiat ac. Donec nec sem luctus nisl posuere tempus eget et neque. Praesent convallis aliquam malesuada. Nulla id tempus diam. In consectetur eu odio vitae mattis. Maecenas tristique volutpat magna nec aliquet. Sed faucibus dapibus velit vitae dignissim. Suspendisse potenti. Maecenas accumsan odio nec efficitur tincidunt.



注意：

13. 申請聲明的標題字體粗度
14. 申請聲明的闊度為50%
15. 申請聲明的字體大小為18pt
16. 申請聲明的文字為置中
17. 申請聲明的底部留有外邊距

CSS練習2 (7)



入學申請表：

名稱

稱謂

報讀原因

最高學歷 博士 碩士
 學士 大專
 中專 中學
 小學 其他

聯絡電話

聯絡電郵

優惠密碼

從何得知 廣告 親友
 搜尋 老師

優惠密碼

從何得知 廣告 親友
 搜尋 老師

由請聲明。

注意：

18.tr元件的背景顏色為透明
(transparent)

19.當鼠標指向每行tr元件時，便
會以0.3秒持續時間和ease動
畫效果來轉換背景顏色為
rgb(240, 240, 240)

20.把最底的tr內的td中置Reset
與Submit按鈕

以「手機模式」開啓網頁

The screenshot shows a web browser interface with several annotations:

- 1.** A red circle highlights the "Inspect" option in the context menu.
- 2.** A red circle highlights the "Elements" panel in the developer tools.
- 3.** A red circle highlights the "iPhone 6/7/8" device selection dropdown in the top left of the developer tools.

The page content includes a "Skip" button, an illustration of a smartphone, and the text "Welcome to the Ionic Super Starter". The developer tools show the HTML structure and CSS styles for the page.

開啓瀏覽器，右點滑鼠並選擇Inspect(檢查) -> 手機圖標 -> 選擇手機型號

響應式網頁設計 (1)

響應式網頁設計(Responsive Web Design)是一種讓網頁可以在不同大小、解析度的裝置螢幕下(例如桌上型電腦、筆電、手機、平板等)自動改變排版佈局的技術名稱。以下是一個簡單的響應式例子：

```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>CSS響應式</title>
  <style>
    body {
      margin: 0;
      font-family: Arial, Helvetica, sans-serif;
    }

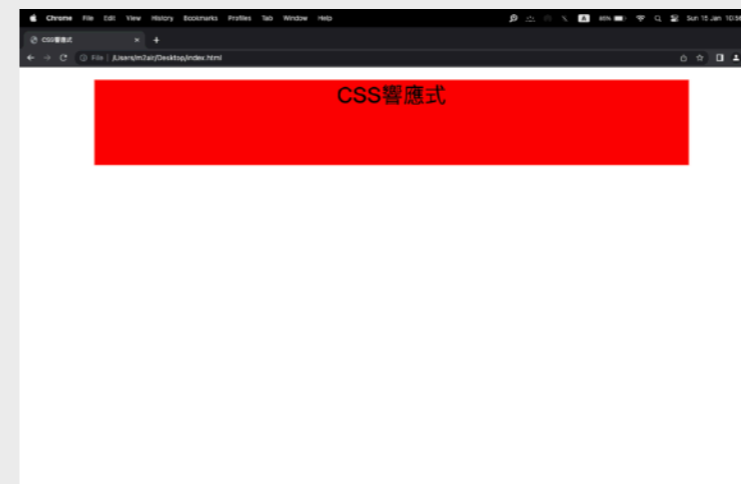
    .Div_1 {
      height: 20vh;
      box-sizing: border-box;
      margin: 3vh auto;
      text-align: center;
      font-size: 32pt;
      width: 80%;
      background-color: red;
    }

    /* 當裝置寬度在414px之內的樣式 */
    @media screen and (max-width: 414px) {
      .Div_1 {
        width: 96%;
        background-color: green;
      }
    }
  </style>
</head>
```

續：

```
<body>
  <div class="Div_1">CSS響應式</div>
</body>

</html>
```



桌面版



手機版

響應式網頁設計 (2)

為網頁加上兼容手機的HTML碼

```
<head>  
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">  
  <meta name="viewport" content="width=device-width, initial-scale=1">  
  <!-- 其他HTML -->  
</head>
```

加上以上這行 HTML 代碼到 `<head></head>` 內，是告訴瀏覽器如何調整網頁的顯示方式，以使其在不同設備和屏幕尺寸上呈現良好的效果。

具體來說，這個 viewport 設定包括兩個屬性：

- `width=device-width`：這個屬性指定網頁的寬度應該等於設備的寬度，也就是說，網頁的寬度應該根據設備的屏幕寬度來自動調整。
- `initial-scale=1`：這個屬性指定網頁的初始縮放比例，值為1表示不進行縮放，保持原始大小。

這樣的設定可以確保網頁在不同設備上以適當的尺寸呈現，提供更好的用戶體驗。例如，當你在行動設備上瀏覽這個網頁時，瀏覽器會根據設備的屏幕寬度自動調整網頁的版面，使其適應螢幕大小，並且不需要手動縮放或橫向滾動。這有助於提供更好的可讀性和瀏覽體驗。

需要注意的是，這個 viewport 設定通常用於響應式網頁設計（Responsive Web Design），讓網頁能夠適應不同的設備和屏幕尺寸。

響應式網頁設計 (3)

響應式網頁尺寸大小

雖然是讀取同一個網頁，但在CSS指令中要針對個別裝置明確寫出尺寸大小，以下為RWD常用的3個裝置解析度：

- 桌上型電腦：至少要1024px以上
- 平板電腦：720px~1024px
- 手機：320px~720px

設定媒體查詢

媒體查詢的設定可幫助網站判斷在什麼寬度、高度的裝置上要套用何種網頁模式，通常會以視區的寬度為主，畢竟這是最直接影響網頁呈現的指標。

常見的項目有這些：

- width：視區寬度
- height：視區高度
- device-width：裝置寬度
- device-height：裝置高度
- orientation：裝置寬高比
- aspect-ratio：視區寬高比
- color：視區顏色

響應式網頁尺寸大小

```
/* Extra small devices (phones, 600px and down) */
@media only screen and (max-width: 600px){ ... }
/* Small devices (portrait tablets and large phones, 600x and up) */
@media only screen and (min-width: 600px){ ... }
/* Medium devices (landscape tablets, 768px and up) */
@media only screen and (min-width: 768px){ ... }
/* Large devices (laptops/desktops, 992px and up) */
@media only screen and (min-width: 992px){ ... }
/* Extra large devices (large laptops and desktops, 1200px and up) */
@media only screen and (min-width: 1200px){ ... }
```

添加斷點

撰寫範例如下：

```
/* For desktop: */
.col-1 { width:50%; }
.col-2 { width:33.33%; }
.col-3 { width:25%; }

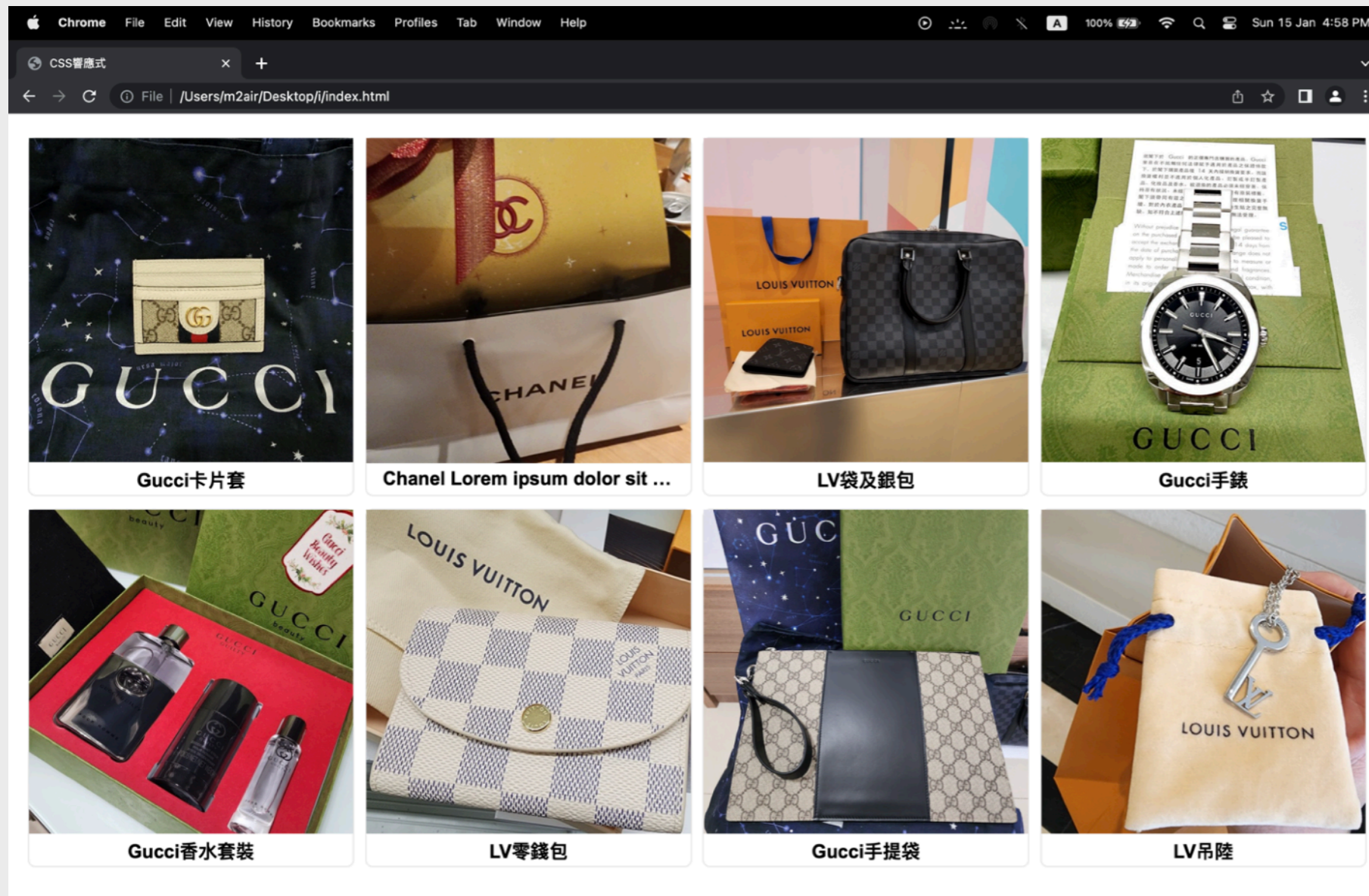
@media only screen and (max-width: 768px) {
  /* For mobile phones: */
  [class*="col-"] { width:100%; }
}
```

CSS練習3 (1)

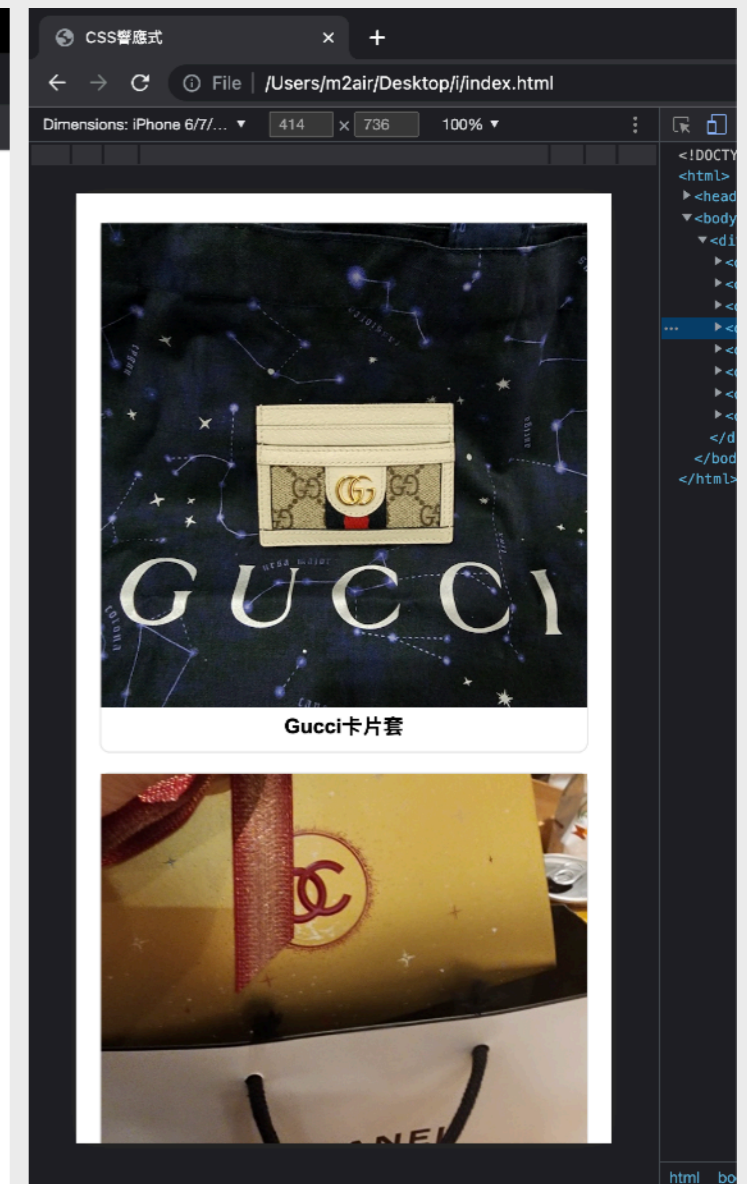
挑戰題，

以下是一個二手精品網店的產品列表，試參考以下附圖，製作一個相似並支援桌面版與手機版的響應式網頁。

圖片下載: <https://s3objectstorage-a.ap-south-1.linodeobjects.com/css-ex-3-assets.zip>



桌面版



手機版 (414x736)

CSS練習3 (2)

以下是可供參考的HTML碼

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>CSS響應式</title>
  <style></style>
</head>
<body>
  <div class="Div_Container">
    <div><div><div>Gucci卡片套</div></div></div>
    <div><div><div>Chanel Lorem ipsum dolor sit amet elit.</div></div></div>
    <div><div><div>LV袋及銀包</div></div></div>
    <div><div><div>Gucci手錶</div></div></div>
    <div><div><div>Gucci香水套裝</div></div></div>
    <div><div><div>LV零錢包</div></div></div>
    <div><div><div>Gucci手提袋</div></div></div>
    <div><div><div>LV吊陸</div></div></div>
  </div>
</body>
</html>
```


CSS練習3 (3)

添加以下CSS到head元件內的style元件裡。

```
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
}

.Div_Container{
  width: 98%;
  display: flex;
  flex-wrap: wrap;
  margin: 3vh auto;
}

.Div_Container>div {
  width: 24.5vw;
  min-height: 27vw;
}

.Div_Container>div div:first-child {
  width: 96%;
  height: 96%;
  border: 1px solid rgb(230,230,230);
  border-radius: 6pt;
  margin: 0 auto;
}

.Div_Container div div img {
  display: block;
  width: 100%;
}

.Div_Container>div div:first-child div{
  width: 90%;
  white-space: nowrap;
  text-overflow: ellipsis;
  text-align: center;
  overflow: hidden;
  margin: 0 auto;
  font-weight: bold;
  font-size: 15pt;
  padding: 4pt;
}
```

然後嘗試參考「響應式網頁設計 (1)」(前四頁)的例子來製作手機版CSS，max-width仍為414px。

「.Div_Container>div」的width為94.5vw，min-height為103vw，margin為0 auto。另外，

「.Div_Container>div div:first-child div」的font-size為12pt，padding為2pt。

更多CSS練習

https://www.w3schools.com/css/css_exercises.asp

更多練習可到W3School找到

課題三：

JavaScript程式編寫及應用

何謂編程

電腦程式簡介

- 利用電腦的語言來指令電腦工作
- 程式語言分有很多種，作用也有不同，但邏輯大同小異
- 程式語言的例子：C、C++、Objective-C、C#、Swift、Java、JavaScript、TypeScript、PHP、Visual Basic、Python、Erlang、Ruby.....
- 現代化的程式通常都是屬於物件導向程式(Object-oriented Programming)
- 可使用程式碼來開發電腦(包括手機及智能手錶)的軟件
- 包含修讀程式編寫的高等課程包括：Computer Science(計算機科學)、Information Technology(資訊科技)、Web Technologies(萬維網科技)、Software Engineering(軟件工程).....
- 熱門工作及應用範圍包括：Research(研究)、Software Engineering(軟件工程)、AI/Machine Learning(人工智能/機械學習)、Data Engineering(數據工程)
- 用於網頁上的程式語言為JavaScript或其衍生程式語言TypeScript

JavaScript簡介

JavaScript是一種物件導向程式，由Netscape Communications (網景通訊公司)於1995年12月創建。於起初的時候通常只用於客戶端，為網頁提供更多額外功能如控制HTML的元件及其CSS屬性。但到了現今的時代，作為一種程式語言，人們便開始利用它於伺服器端(如Node.js)以提供資料庫存取、HTTP請求處理與回覆、Socket程式等。

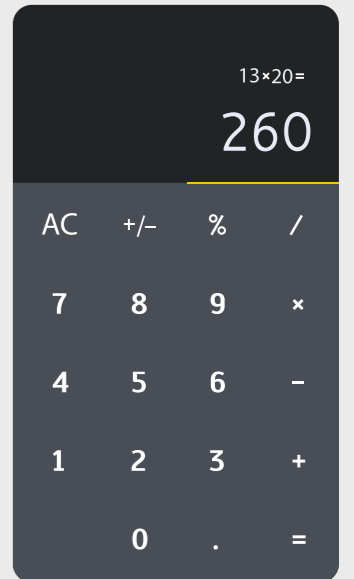
隨著HTML5與CSS3的誕生，一系列JavaScript的新功能亦隨之而出現，例如File APIs(瀏覽器檔案處理)、瀏覽器拖放功能、網絡Socket通訊、背景處理(多工處理的概念)、瀏覽器存儲功能。因此，之於HTML5、CSS3與JavaScript的萬維網技術得以重大的提升，令很多開發者都開始候用它來製作網絡應用、手機程式和電腦軟件。

JavaScript用處

以下是一些例子，讓程式新手理解JavaScript(簡稱JS)程式碼的用處：

- 計數機

- 透過JS擷取用戶輸入的資料(即數字和運算符)，再進行運算，最後顯示到畫面。



- 社交APP帖子點讚



Like



Like

- 當用戶(即用戶端)按下「讚」時，便會激發起JS工作，JS便向伺服器(Server)發出請求，請求伺服器幫助完成工作，請求中附著了用戶的User ID與帖子的ID。當伺服器完成工作並對用戶端作出了回應，用戶端的JS便接住伺服器給它的回應，然後再進行處理並更新畫面讓「讚」按鈕由空心變成實心。

- 記事簿APP

- 透過JS擷取用戶輸入的資料，然後儲存到手機的儲存空間內。當用戶開啟舊筆記時，JS便找出存檔並顯示到畫面。

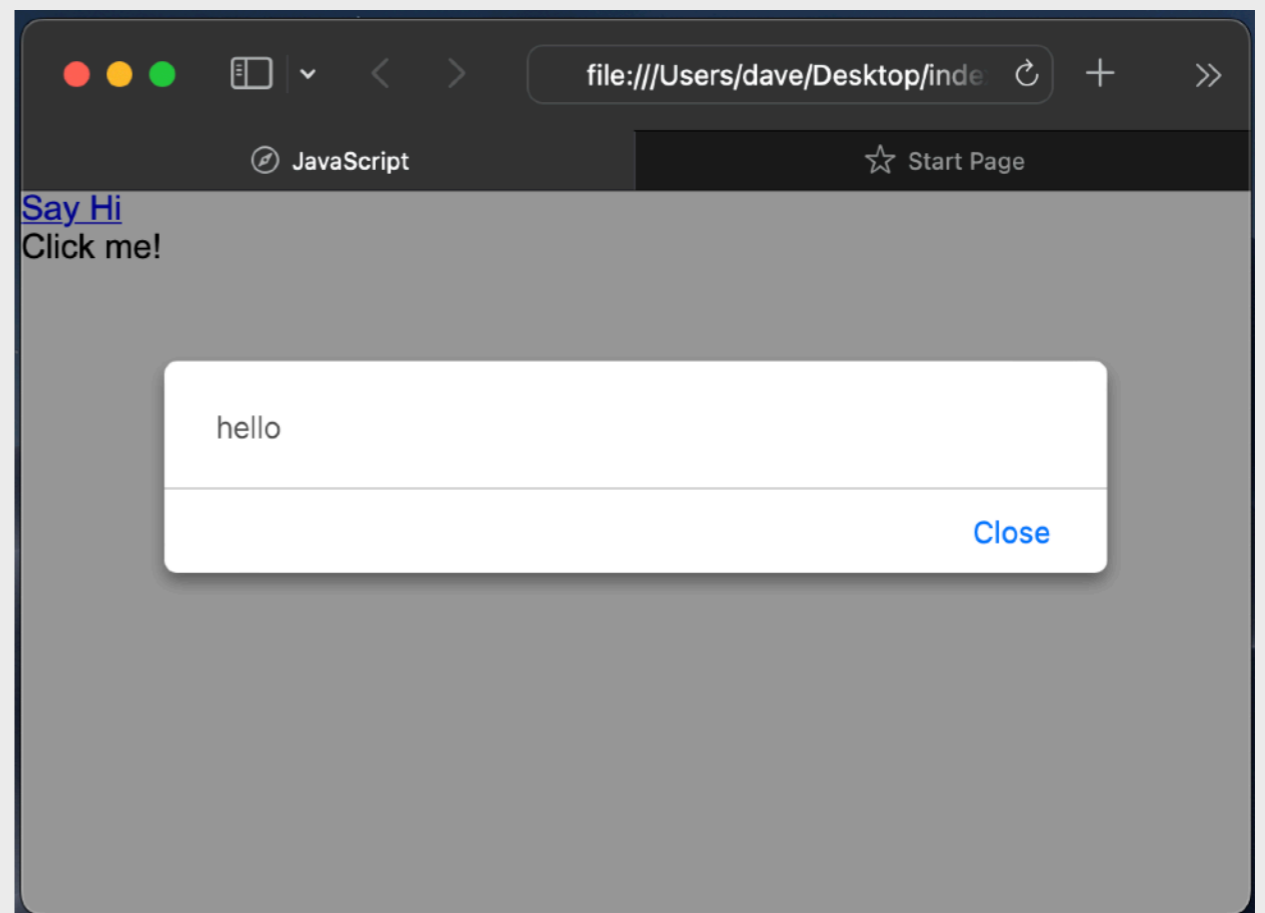
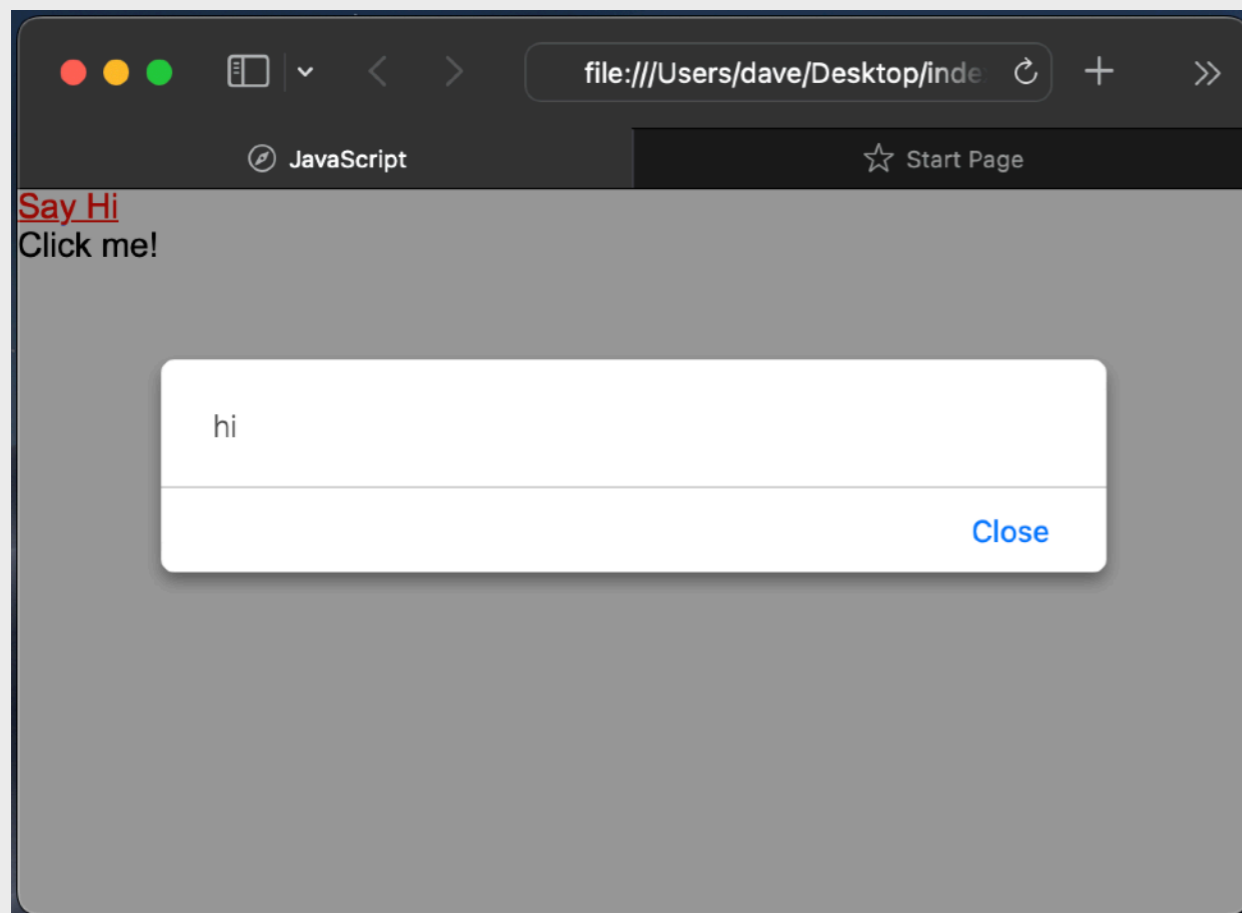


JavaScript套用方式 (行內套用)

行內套用 (Inline)

直接套用到HTML的Element(元件)中，好處是簡單直接地寫到元件中。

```
<a href="javascript:alert('hi');" >Say Hi</a>  
<div onclick="alert('hello');" >Click me!</div>
```



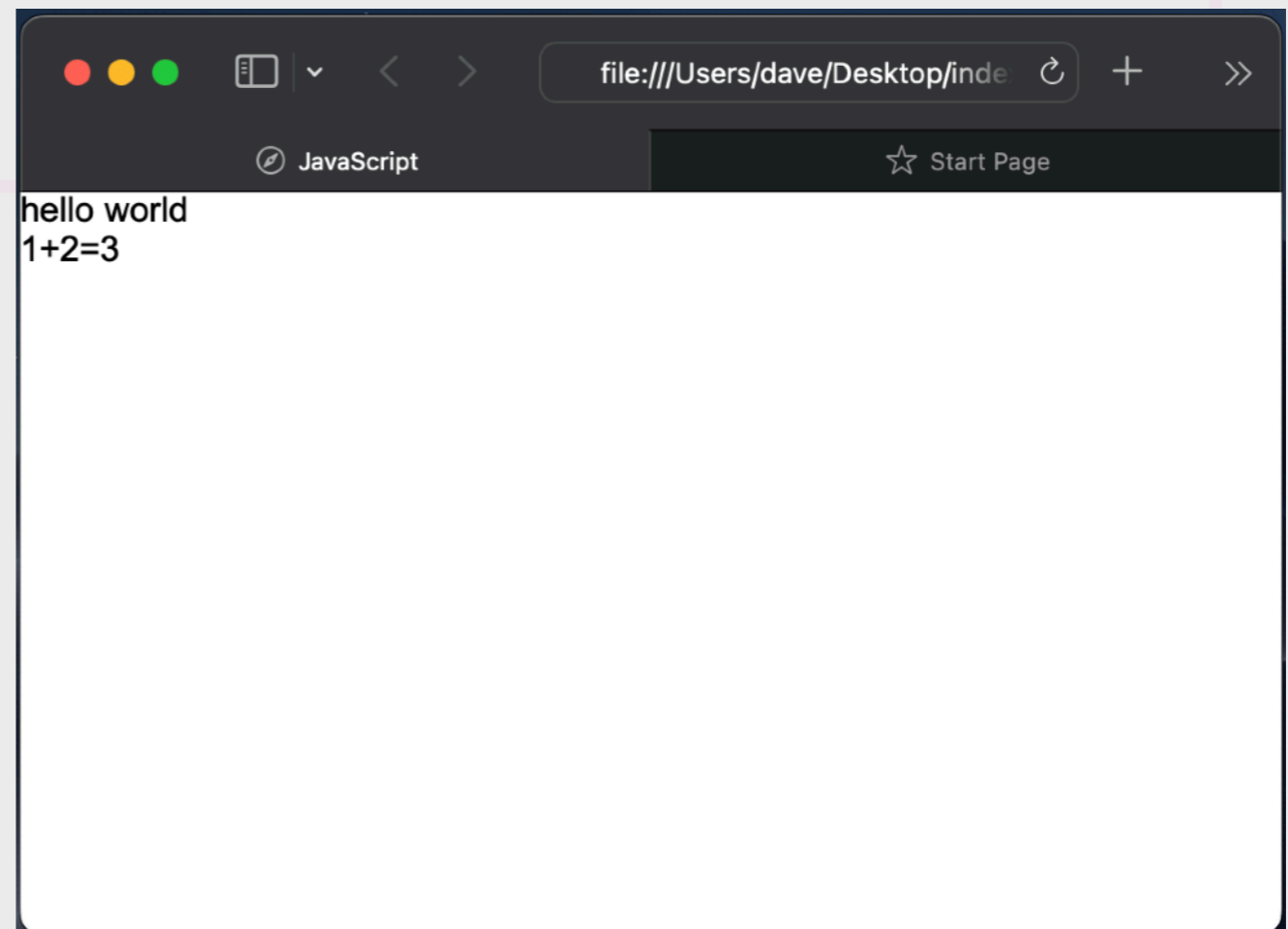
JavaScript套用方式 (嵌入套用)

嵌入套用 (Embed)

套用到HTML的文檔中。

注意：<script>元素內的內容是特別內容(即JavaScript程式碼)，並不使用HTML的寫法。

```
<script>  
  document.write('hello world<br>');  
  let a=1, b=2;  
  document.write('1+2='+ (a+b));  
</script>
```



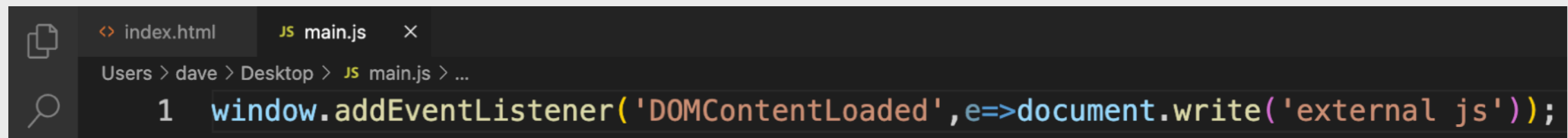
JavaScript套用方式 (外部連接套用)

外部連接套用 (External Link)

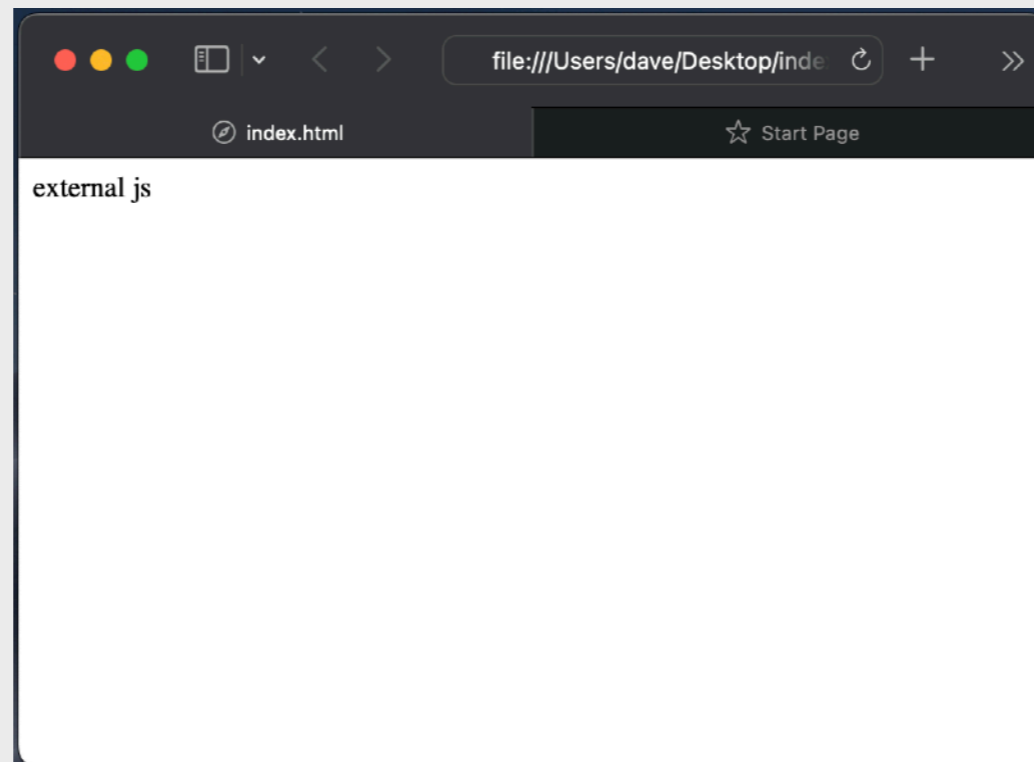
使用外部的js檔。通常用於載入第三方插件或較大型的JavaScript程式庫。

```
<head>  
  <script src="main.js"></script>  
</head>
```

main.js



```
index.html JS main.js x  
Users > dave > Desktop > JS main.js > ...  
1 window.addEventListener('DOMContentLoaded', e=>document.write('external js'));
```



JavaScript放在哪個元素之內？

通常使用外部連接套用(External Link)時，我們都會把JavaScript放到head元素內

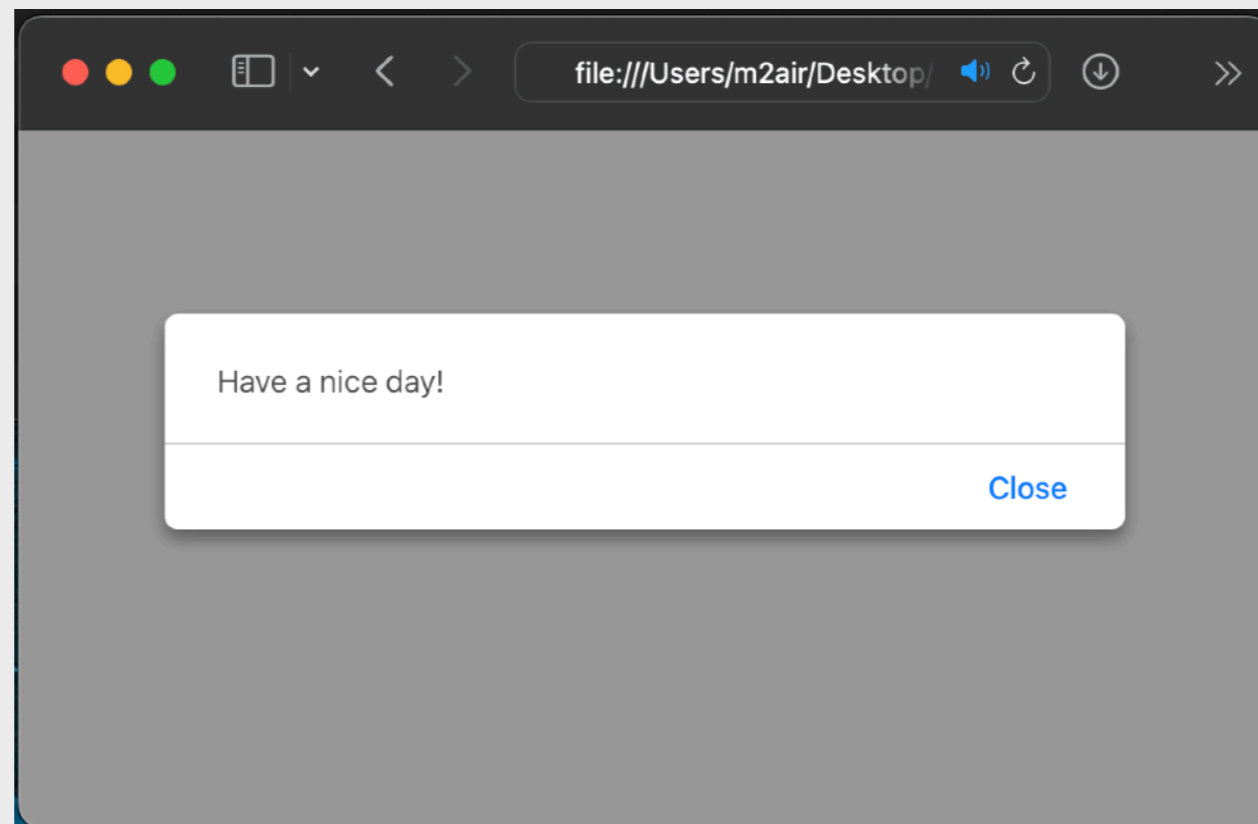
通常使用嵌入套用(Embed)時，我們都會把JavaScript放到body元素內，並建議放到body元素內的最底部

JavaScript練習1

試以嵌入套用方式把此句JavaScript放到以下網頁的body內：
`setTimeout(()=>alert('Have a nice day!'),1500);`

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
  <title>JavaScript練習1</title>
</head>
<body>
</body>
</html>
```

範例：



JavaScript的程式註解

程式註解是程式設計上很重要的部分，良好註解不但能夠輕易了解程式目的，在維護上也可以提供更多的資訊。JavaScript 程式註解是以 // 符號開始的列，如下所示：

```
// 大家好
```

```
let num1, num2; // 單行的註解
```

如要將大段文字作為註解，可用 /* */ 符號將文字包圍，如下所示：

```
/*
```

```
    這是一個 JavaScript 程式
```

```
    這也是一個多行的註解
```

```
*/
```

在程式的頂部，常見會有以下註解：

- 程式的作者 - 功能 - 日期

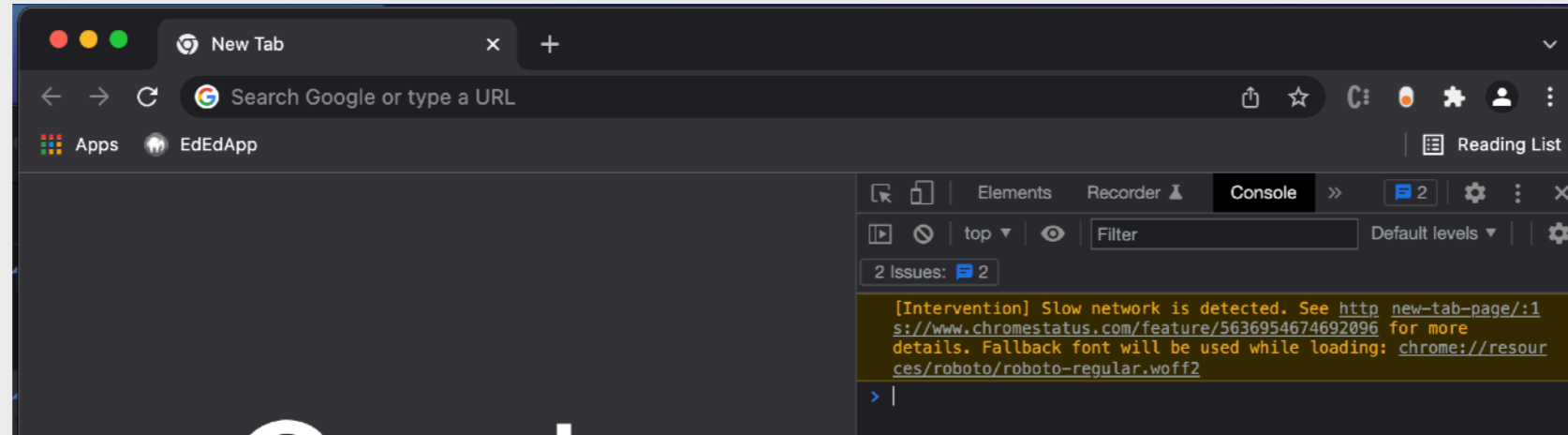
JavaScript注意事項

一點 . 有著「的」的意思，例如「amy.age」即是「amy『的』歲數」

分號 ; 的意思相似於「句號」，即表示完結一個指令

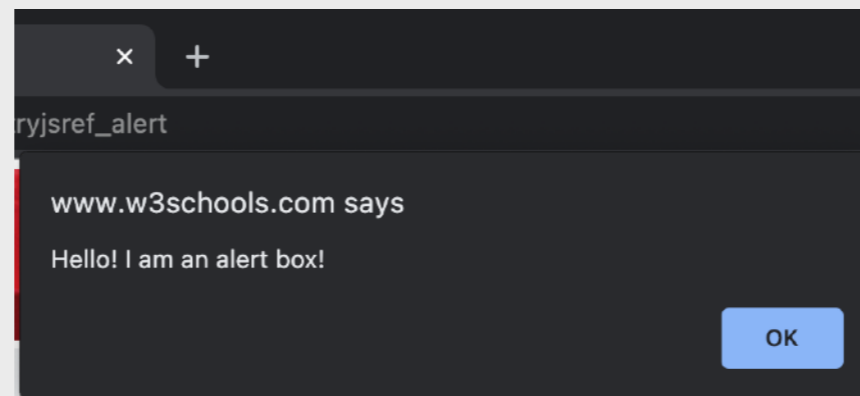
undefined、null、0、false、NaN、空字串的邏輯結果均為 false

JavaScript常用測試方法



`console.log(...)`

可開啓瀏覽器的Inspect模式並選擇Console來查看



`alert(...)`

在頁面彈出提示盒來顯示出測試結果

JavaScript改變HTML狀態

我們可透過 `document.getElementById` 來對某一HTML元件進行改變

當然，我們需要先為那個需要被改變的HTML元件加上一個 `id`

例如：`<div id="div_1">Hello</div>`

然後便可以 `document.getElementById("div_1")` 來取得刻元件

我們可使用 `.style` 來存取元素的CSS屬性，如果屬性名稱中含有破折號如 `background-color`，則需把各字串合併並把首字串後的所有字串的第一個字母變成大寫，如 `backgroundColor`

例如以 `document.getElementById("div_1").style.backgroundColor` 來讀取或改變 `div_1` 元件的CSS的背景顏色

例如以

`document.getElementById("div_1").style.backgroundColor="gold"`
來改變 `div_1` 元件的CSS的背景顏色為金色

怎樣暫存資訊？

怎樣在遊戲程式中儲存著玩家的得分？

血壓計怎樣記下使用者的上壓和下壓？

 試思考：

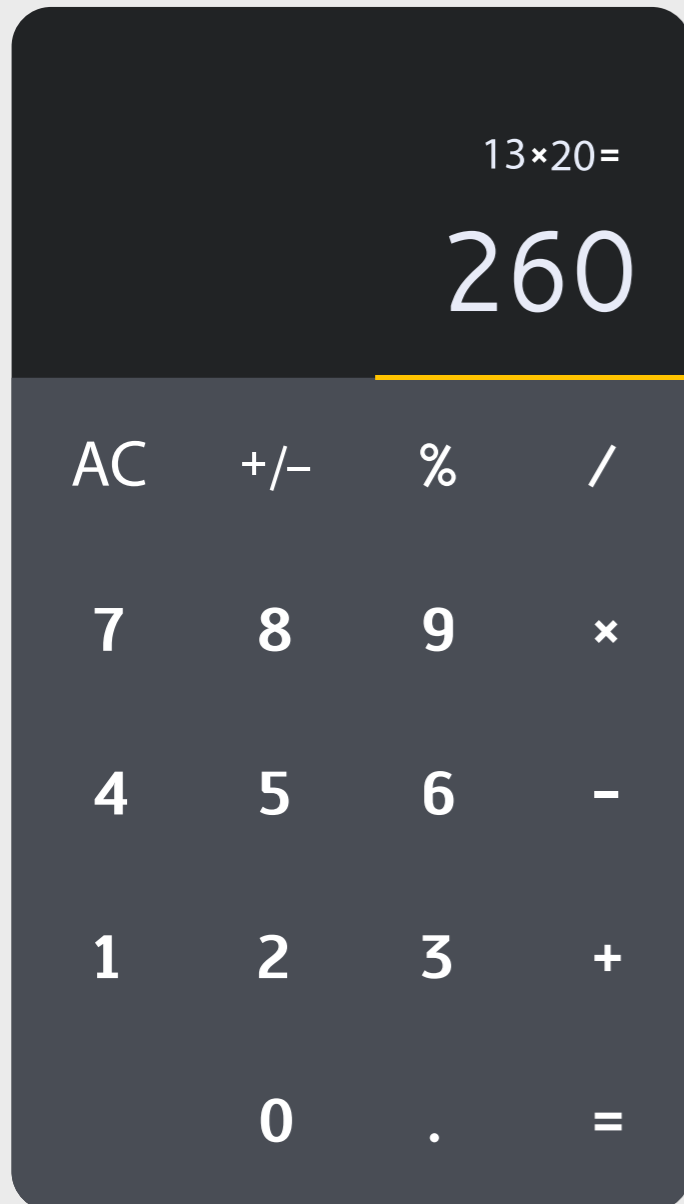
一個遊戲中通常會有甚麼變數/屬性？

Player Name?

戰鬥值?

Exp.?

JavaScript計數機例子的解讀



以計數機為例，當用戶輸入了第一個數值，然後按下運算符號了(即 $+ - \times \div$)，然後再輸入了第二個數值，我們要怎樣處理整個計算流程？

1. 把第一個數值記下，並把它命名為a
2. 把被按下運算符號記下，並把它命名為symbol
3. 把第二個數值記下，並把它命名為b
4. 當用戶按下等如號時，我們透過運算式計算，再把結果顯示

在上述的過程中，這些被記下來的數值和運算符號，其實就是稱之為「變數」！在程式語言的「變數」(Variables)可以視為是一個擁有名稱的盒子，能夠暫時儲存程式執行時所需的資料。

變數有不同種類？

- 日常生活中，原來變數都擁有著不同等類型
- 例如「年齡」和「公斤」則是數字類型變數，「玩家名稱 playerName」則是文字類型變數，「是否飢餓 isHungry」則是true/false類型變數.....

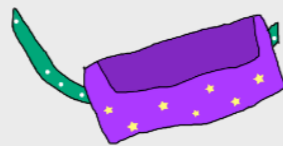
常見的變數型別(類型)有：

number (數字)、**boolean** (布林值，即true/false)、**string** (文字)



人類:

1. 身高 (height) - number
2. 體重 (weight) - number
3. 身上有否食物 (hasFood) - boolean



袋子

1. 名稱 (name) - string
2. 是否缺貨 (outOfStock) - boolean
3. 成本價 (cost) - number
4. 售價 (price) - number

如何製作變數？

簡單定義變數:

變數的定義字 變數名稱;

實例解釋:

變數 用戶歲數;

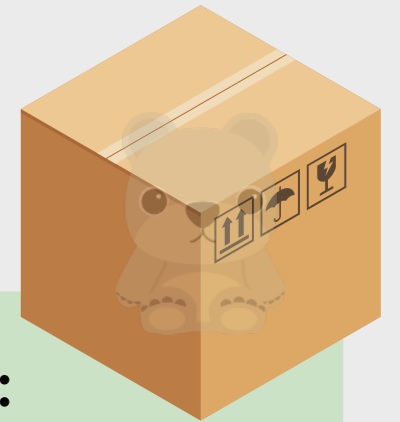
實例實作:

```
let userAge;
```

如只定義變數而不賦值，
就似是在工廠裡，先製造了「吉盒」
運貨盒，待產品製作完成後備用



let為定義變數所用，
const為定義常數所用。
常數即「不可變的變數」。



簡單定義常數及賦值:

常數的定義字 常數名稱=數值;

實例解釋:

常數 圓周率=3.1416;

實例實作:

```
const PI=3.1416;
```

簡單定義變數及賦值:

變數型態 變數名稱=數值;

實例實作:

```
let userAge=22;
```



單行定義變數

如果同一型別的變數，我們還可以於一行中一次過對它們定義及賦值，並以逗號分隔它們。

如只定義變數：

```
let a, b, c;  
let x, y, z;  
let playerFirstName, playerLastName;
```

如定義變數並對其賦值：

```
let a=1, b=2, c=3;  
let x=2.2, y=3.3, z=4.4;  
let playerFirstName="Emperor", playerLastName='Fung';
```

如夾雜定義與賦值：

```
let a=1, b, c, d=4;  
let w, x=2.2, y=3.3, z;  
let playerFirstName="Emperor", playerLastName;
```

變數的宣告-變數的初值

「指定敘述」 (Assignment Statement) 是在程式中存取指定變數的值，如果在宣告變數時沒有指定變數的初值，就可以使用指定敘述即“=”等號指定變數值或更改變數值，其語法如下所示：

變數 = 運算式;

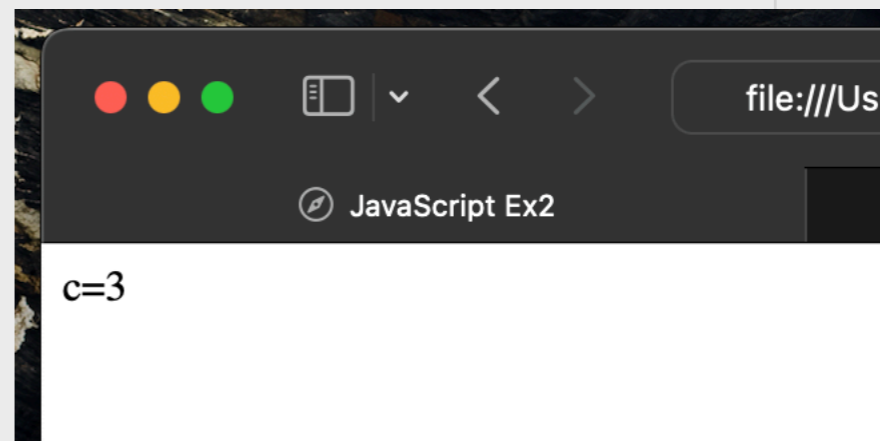
在指定敘述的左邊是變數名稱，右邊是「運算式」 (Expression)，運算式可以是運算子和運算元組成的任何運算式。

簡單的說，指定敘述是「將右邊運算式的運算結果指定給左邊的變數」。

JavaScript練習2

試填寫漏空處，以單行定義變數 a、b 和 c，並賦 a 值為 1，b 值為 2

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>JavaScript Ex2</title>
</head>
<body>
  <script>
    _____
    c=a+b;
    document.write('c='+c);
  </script>
</body>
</html>
```



變數命名法則

通常第1個英文字小寫之後為大寫，變數、程序/函式的命名可以使用不同英文字母大小寫的組合，如下表所示：

識別字種類	習慣的命名原則	範例
常數	使用英文大寫字母和底線"_"符號	MAX_SIZE、PI
變數	使用英文小寫字母開頭，如果是2個英文字組成，第2個之後的英文字以大寫開頭	size、userName

變數命名規則

JavaScript關鍵字(keywords)或物件(Object)名稱不可作為變數名稱

變數名稱不可以數字開頭

英文大小寫有分別，即是說 abc 及 ABC
是兩個不同的變數名稱

名稱不能含有符號(除了底線)或空白，
只能是英文字母、數字、底線和錢號

在宣告的有效範圍內必須唯一，只在其所屬的程式組裡有效

文字變數的內容需以被單引號或雙引號包著，如'hello'或"hello"

JavaScript關鍵字

我們不可完全地使用以下字眼作為變數的名稱，如不可以定義：

`const case=123;` ❌

但可以包含以下字眼於變數的名稱中，如：

`const _case=123;` ✅

`let className='Cat';` ✅

abstract	boolean	break	byte	case	catch	char
class	const	continue	debugger	default	delete	do
double	debugger	else	enum	export	extends	false
final	finally	float	for	function	goto	if
implements	import	in	instanceof	int	interface	long
native	new	null	package	private	protected	public
return	short	static	super	switch	synchronized	this
throw	throws	transient	true	try	typeof	var
void	volatile	while	with			

JavaScript變數名稱的範例



def, no_123, size1	合法名稱 ✓
Car, car, count, height, s1	合法名稱 ✓
123abc, 555xyz	不合法名稱，因為以數字開始 ✗
xyz#abc, bbb-ccc, Bc+de, o@o, x.y	不合法名稱，因為變數中有不容許的符號 ✗

JavaScript練習3

變數與函式命名及賦值使用練習

Q1. 定義 PI 為 π 常數，並賦予值為: 3.1416

Q2. 定義皇帝名稱(Emperor Name)為常數，並賦予值為: 康熙

Q3. 於同一行中定義變數 Dynasty Name、Emperors Count 及「是否有宰相」Has Prime Minister，並分別賦予值為: Qing、12 及 false

Q4. 定義變數人口數量(population)為常數並不賦值，會有甚麼結果？

程式為甚麼需要做運算？

試想想，假如你是一間涼茶鋪的東主，你需要賣涼茶，也需要執枱、清潔、沖茶……等。執枱、清潔和沖茶這些都是一些比較屬於「**執行性質**」的工作；但「賣涼茶」則是「**計算性質**」的工作，它需要涉及到計算售價和收錢並計算找續。哪麼我們怎樣透過程式來進行一些運算的工作呢？

在編程的世界裡，有兩種運算方法，就是算術運算(即是數學上的運算)和邏輯運算(即是判斷出結果是「是」或「否」(**true或false**))。在算術運算中，我們需要透過算術運算符(如+-*/)來進行運算。而在邏輯運算中，我們需要透過關係運算符(如>)甚或加上邏輯運算符(如!=)來進行運算。而計算結果的這種程式，則稱作為「運算式」。

運算式(Expressions)

運算式是可以求值以給出單個值的程式碼：variables(變數)、constants(常數)、operators(運算符)和parentheses(括號)
三種運算式：

- Arithmetic Expressions (算術運算式)
- Relational Expressions (關係運算式)
- Logical Expressions (邏輯運算式)

35

age

3 * number + 56

age / value + (number - age)

a > b

(26 + c != d) && (e <= f - 3)

Variable name
(變數名稱)

Assignment operator
(賦值運算符)

myAge = 運算式;

運算符號 (1)

算術運算符號:

算術符號	用途
+	相加
-	相減
*	相乘
/	相除
%	取餘數
=	變成為

關係運算符號:

關係符號	用途
==	判斷兩者是否相等
>	判斷前者是否大於後者
<	判斷前者是否少於後者
>=	判斷前者是否大於後者或兩者相等
<=	判斷前者是否少於後者或兩者相等
!=	判斷兩者是否不相等

邏輯運算符號:

邏輯符號	用途
&&	AND, 「以及」的意思
	OR, 「或者」的意思
!	NOT, 「相反」的意思

運算符號 (2)

算術運算符號(簡寫版):

符號	用途	例子
++	自身加一	i++;
--	自身減一	i--;
+=	自身加某值	a+=5;
-=	自身除某值	a-=5;
=	自身乘某值	a=5;
/=	自身除某值	a/=5;

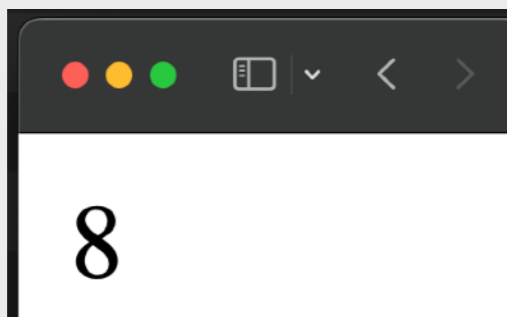
算術運算式的優先順序

運算式的求值遵循以下兩個規則：

- 始終優先評估最高優先級的運算符
- 如果運算符具有相同的優先級，則評估是從左到右
- 先乘除後加減，括號最優先

Operator type	Operator	Associates
grouping	<i>(expression)</i>	Left to right
unary	++, --, +, -	Right to left
cast	<i>(type)</i>	Right to left
multiplicative	*, /, %	Left to right
additive	+, -	Left to right
assignment	=, +=, -=, *=, /=, %=	Right to left

High priority	()
	++ --
	* / %
	+ -
Low priority	=



8

```
<body>
  <script>
    const num1 = (2 + 3 * 5 % 6) + 3;
    document.write(num1);
  </script>
</body>
```

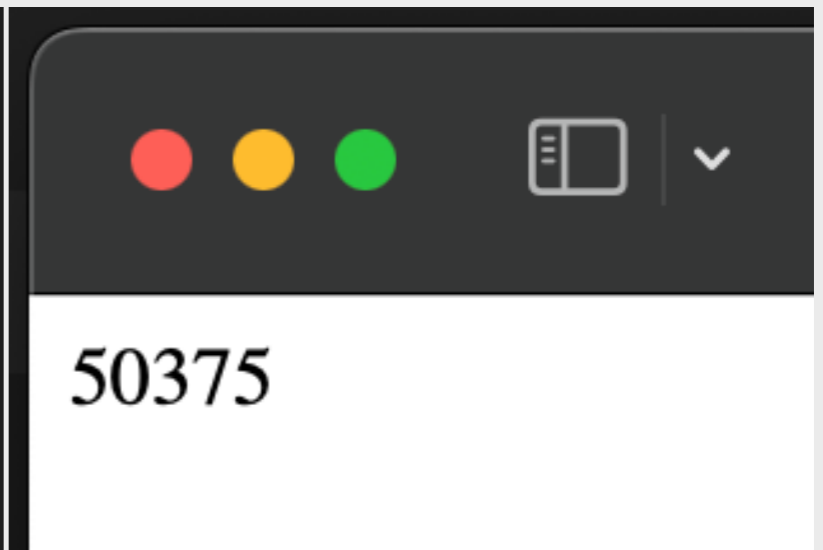
在這個例子中，
 $(2 + 3 * 5 \% 6) + 3$
的答案會是：8

算術運算式例子

假設存款\$50000，2個月後本利和(amount)會是甚麼？

```
const principal = 50000;  
const interestRate = 4.5/100;  
const year = 2/12;  
const amount = (principal * interestRate * year) + principal;  
document.write(amount);
```

```
<body>  
  <script>  
    const principal = 50000;  
    const interestRate = 4.5 / 100;  
    const year = 2 / 12;  
    const amount = (principal * interestRate * year) + principal;  
    document.write(amount);  
  </script>  
</body>
```



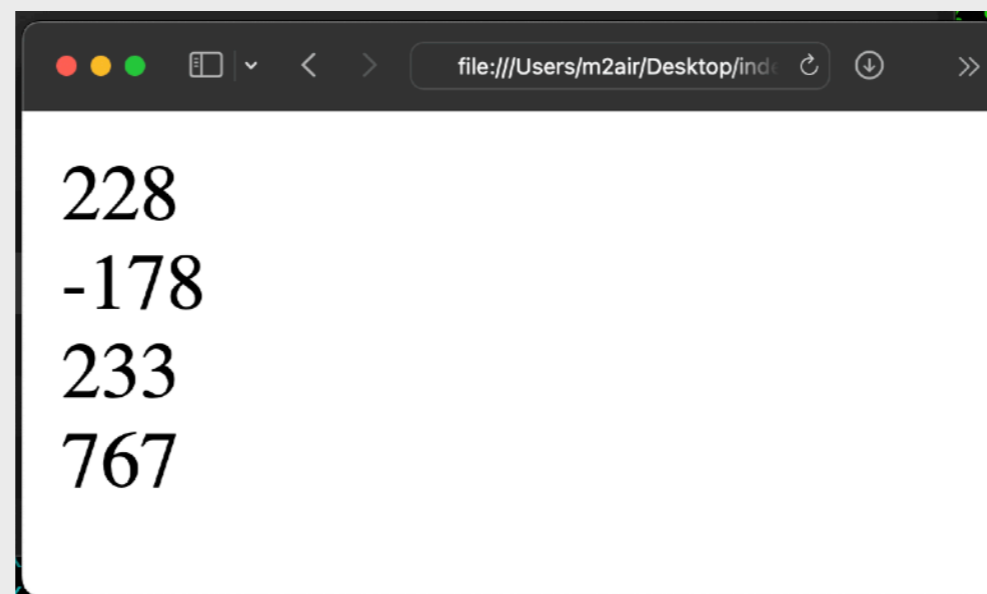
答案會是：50375

JavaScript練習4

以前面章節提及的涼茶鋪為例，「廿四味」售價為\$24，「靈芝茶」售價為\$33，「菊花茶」售價為\$18，「樽裝水」售價為\$5，然後：

1. 試為該四種產品之售價以變數(或常數)定義出來並賦值
2. 定義一個變數去記下總價格，初始值為 0
3. 一客人到訪，並一口氣喝下了 3 碗廿四味、2 碗靈芝茶和 5 碗菊花茶，然後該客人突然發現肚痛，並準備結帳去肚瀉，然而作為東主的你需要為其計算總價格
4. 透過 `document.write` 顯示出總價格
5. 該客人肚痛著並趕急的放下了\$50元在檯面後便匆忙離去了，你發現該客人付款不足，並透過 `document.write` 顯示出客人所欠的款項是多少
6. 該客人回來後購買多了 1 樽樽裝水給自己補充水份，你需要更新總價格的變數，試透過 `document.write` 顯示出新的總價格
7. 該客人終於找到\$1000元鈔票並給予你，試透過 `document.write` 顯示出所需找續的費用
8. 每行只顯示一個數值

結果：



```
228  
-178  
233  
767
```

提示：可使用
`document.write('
');`
來印出「換行」

JavaScript練習5

以下是一個把毫升轉換為升的程式，請填寫兩個漏空處。

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>JavaScript Ex2</title>
</head>
<body>
  輸入一個以ml(毫升)為單位的數字:<br>
  <input id="input_ml" type="_____ "><br>
  <button onclick="cal();">計算</button>
  <div id="div_ans"></div>
  <script>
    function cal(){
      let ans=document.getElementById('input_ml').value;
      ans=_____
      document.getElementById('div_ans').innerHTML='答案是：'+ans+'L';
    }
  </script>
</body>
</html>
```



關係運算式

- 關係運算式最終一定會評估為布林值即true(真)或false(假)：
- 注意：常見編程錯誤
 - 將相等運算符 == 與賦值運算符 = 混淆



運算子	說明	運算式範例
==	相等於	b == c, b 的值等於c 的值
!=	不等於	d != e, d 的值不等於e的值
>	大於	x > y, x 的數值大於y 的數值
<	小於	x < y, x 的數值小於y 的數值
>=	大於或等於	x >= y, x 的數值大於或等於y 的數值
<=	小於或等於	x <= y, x 的數值小於或等於y 的數值

邏輯運算式 (1)

試想想，如果要兩個或多個條件都成立才true的話，要怎樣做？
(e.g. 「便宜」又「好看」的手機我才買。)

試想想，如果有兩個或多個條件，只要其中一個成立便是true，哪要怎樣做？
(e.g. 「便宜」或「好看」的手機我都可以買。)

將算術類型、關係運算式的變數和常數與邏輯運算符組合

Logical And: **&&**

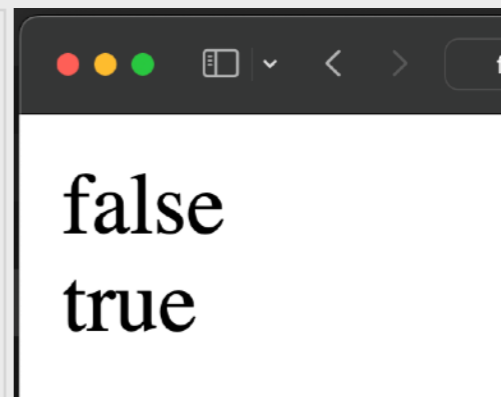
Logical Or: **||**

Logical unary NOT: **!**

評估為 **true** 是 或 **false** 否

例子：

```
<script>
  const value=200;
  document.write(value>1 && value<100);
  document.write('<br>');
  document.write((value>=1 || value==5) && value<300);
</script>
```



邏輯運算式 (2)

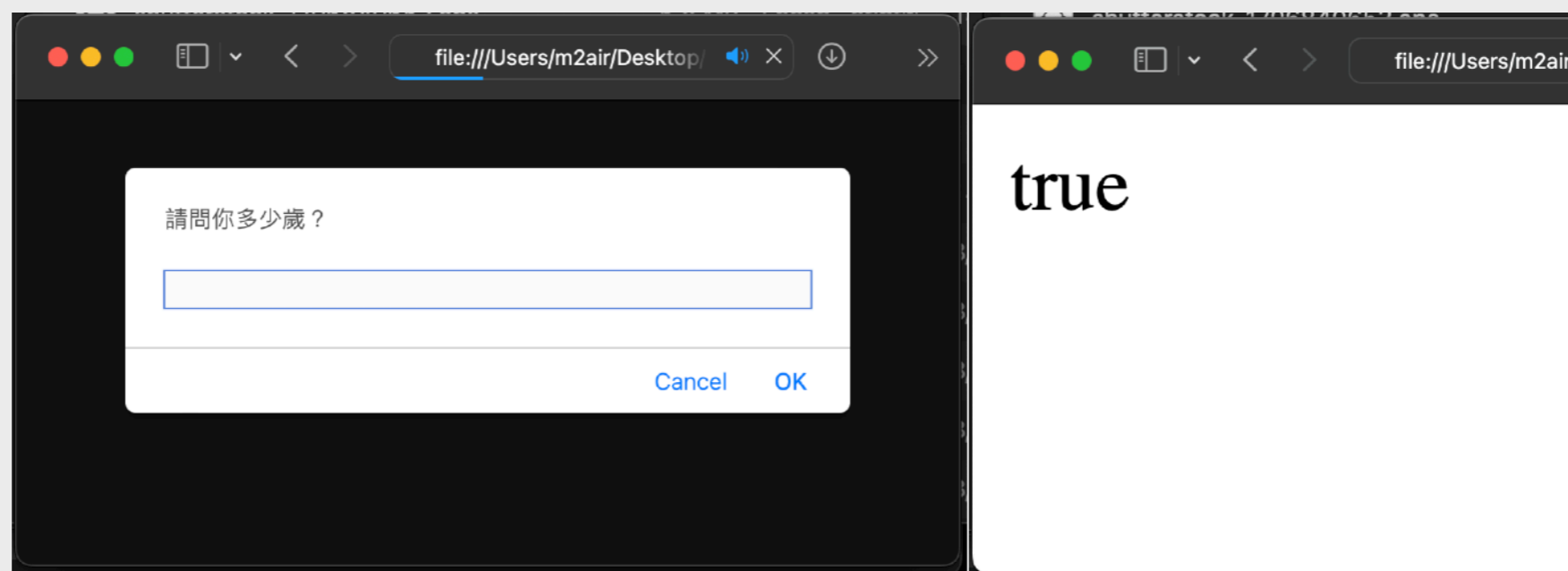
- “!”在電腦程式中表示NOT (否定)的意思，通常用於作邏輯判斷的布林值 (Boolean)。
 - 若 $b == \text{true}$ ，則 $!b$ 即等如 true 的相反，即 $!b$ 相等於 false
- “&&”即是邏輯中AND的意思，它必須符號兩邊的項均為 true 才會輸出 true 作為結果。
 - 即 $a == \text{true}$, $b == \text{false}$, 則 $a \&\& b == \text{false}$
 - a 與 b 有任何一個是 false 即會令 $a \&\& b == \text{false}$
 - 若 a 與 b 均是 true ，即會令 $a \&\& b == \text{true}$
- “||”即是邏輯中OR的意思，它必須符號兩邊的任何一項為 true ，即會輸出 true 作為結果。
 - $a == \text{true}$, $b == \text{false}$ ，則 $a || b == \text{true}$
 - $a == \text{false}$, $b == \text{true}$, 則 $a || b == \text{true}$
 - $a == \text{true}$, $b == \text{true}$, 則 $a || b == \text{true}$

JavaScript練習6

以前面章節提及的涼茶鋪為例，假若涼茶鋪實施了「小童及長者半價」的政策，即12歲或以下兒童及65歲或以上，試填下漏空處，以完全邏輯運算式。

```
<script>  
  let value=prompt('請問你多少歲?');  
  document.write(_____);  
</script>
```

結果：



怎樣在不同情況下做不同的任務？

試想想，如果涼茶鋪實施了「小童及長者半價」，在程式上是怎樣做判斷而做出對應的動作呢？

例如當用戶選擇了所需涼茶並輸入了歲數，我們可以怎樣憑着其年齡而決定給予正價還是半價？我們就會在稍後章節詳細介紹有關判斷的方法。

```
function calTotal() {
  const prdA=24, prdB=33, prdC=18;
  const quantityA=parseInt(document.getElementById('input_1').value);
  const quantityB=parseInt(document.getElementById('input_2').value);
  const quantityC=parseInt(document.getElementById('input_3').value);
  const quantityD=parseInt(document.getElementById('input_4').value);
  const age=parseInt(document.getElementById('input_5').value);
  let totalPrice=((quantityA * prdA) + (quantityB * prdB) + (quantityC * prdC));
  totalPrice/=2;
  document.getElementById('div_result').innerHTML='總價格為：'+totalPrice;
}
```

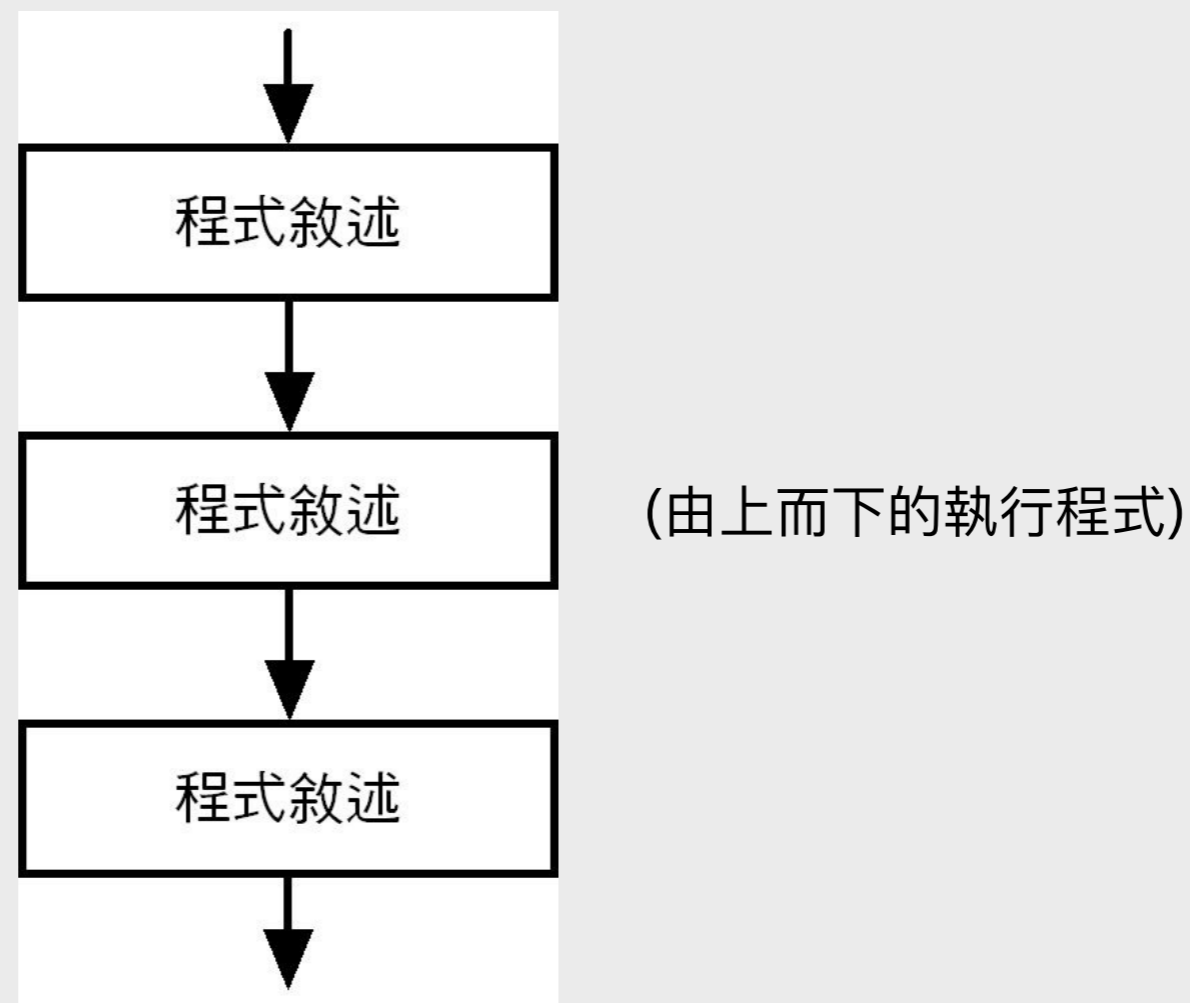
(黃格內)用甚麼方法能憑客戶年齡判斷出要否把總價格除二呢？

結構化程式設計

- 結構化程式由三種常見的流程控制來整合：
 - 循序結構 (Sequential structure)
 - 選擇結構 (Selection structure)
 - 重複結構 (Iteration structure)

循序結構

循序結構是程式預設的執行方式，也就是一個敘述接著一個敘述依序的執行，如下圖所示：



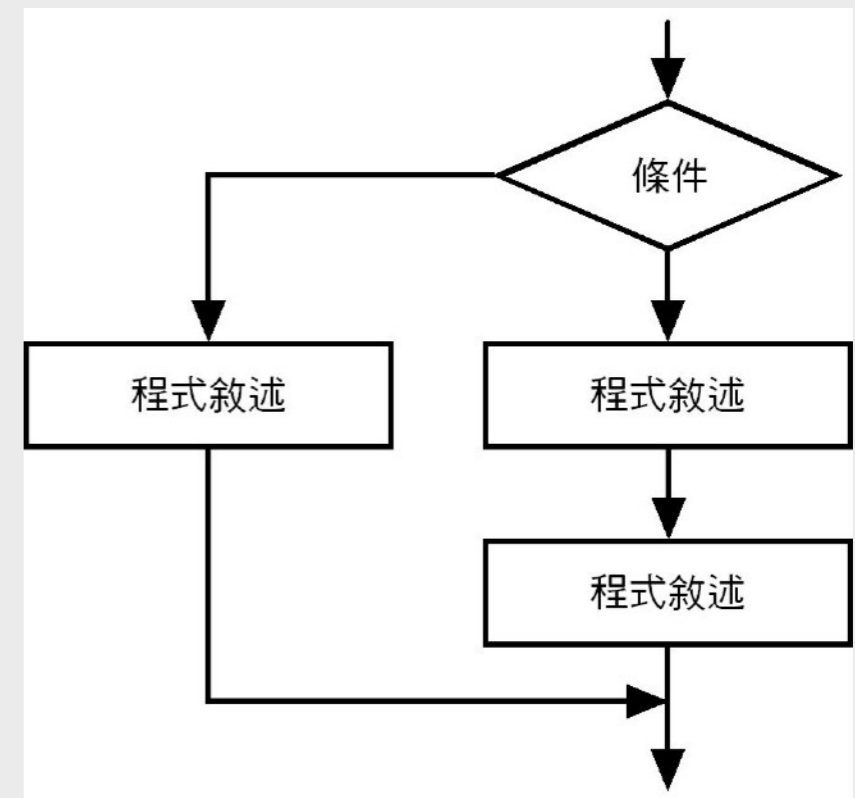
流程控制的基礎-說明

程式語言撰寫的程式碼大部分是一列指令接著一系列指令循序的執行，但是對於複雜的工作，為了達成預期的執行結果，我需要使用「選擇結構」(Selection Structure)來改變執行順序。即用來選擇在不同情況下做不同的任務。



流程控制的基礎-選擇結構 (1)

選擇結構是一種條件控制敘述，它是一個選擇題，可以分為單一選擇、二選一或多選一共三種。程式執行順序是依照關係運算式的條件，來決定執行哪一個區塊的程式碼。



流程控制的基礎-選擇結構 (2)

四種選擇陳述:

if (單選):

如果條件為真(true)，則執行操作

如果條件為假(false)，則跳過它

if ... else (二選一):

如果條件為真，則執行操作；否則，執行其他操作

if ... else if (多選一):

如果擁有排它情況，且多於兩個執行區塊，只能從中選一

switch (多選一):

根據運算式的值執行幾種動作之其中一種

多項選擇語句-在許多不同的動作中選擇其中一種

流程控制的基礎-選擇結構 (3)

涼茶鋪實施了「小童及長者半價」，就讓我們現在揭曉吧，我們可以憑着其年齡，然後透過 If 來判斷客人是否小童或長者，若是則給予半價優惠。

```
function calTotal() {  
  const prdA=24, prdB=33, prdC=18;  
  const quantityA=parseInt(document.getElementById('input_1').value);  
  const quantityB=parseInt(document.getElementById('input_2').value);  
  const quantityC=parseInt(document.getElementById('input_3').value);  
  const quantityD=parseInt(document.getElementById('input_4').value);  
  const age=parseInt(document.getElementById('input_5').value);  
  let totalPrice=((quantityA * prdA) + (quantityB * prdB) + (quantityC * prdC));  
  if(age<=12 || age>=65)totalPrice/=2;  
  document.getElementById('div_result').innerHTML='總價格為：'+totalPrice;  
}
```

單選 if (1)

如果條件成立，則執行操作

if即是「如果」的意思，即如果某一(些)條件成立時，則進行某一(些)動作。
在if後的開關細括號中包著一個布林值或能計算出布林值的運算式。

記得布林值是甚麼嗎？
布林值即是true或false

邏輯:

```
if(條件成立){  
    做某些程式;  
}
```

例一:

```
若(精力充沛){  
    打羽毛球;  
}
```

例二:

```
若(肚餓 && 夠錢){  
    吃自助餐;  
}
```

單選 if (2)

實例:

條件必須用括號()括起來

```
let grade=10;  
if(grade>=50){  
  document.write("Passed");  
}  
document.write("Bye");
```

- if 陳述{ }內的陳述一般會縮排，提高可讀性

Bye

```
let grade=60;  
if(grade>=50)  
  document.write("Passed");  
document.write("Bye");
```

- if 陳述{ }內只有一句陳述，可以省略大括號{ }

Passed
Bye

二選一 if else

如果條件成立，則執行操作；否則，執行其他操作

邏輯:

```
if(條件成立){  
    做某些程式;  
}else{  
    做某些程式;  
}
```

例一:

```
若(覺得精力充沛){  
    打羽毛球;  
}否則{  
    寫程式;  
}
```

例二:

```
若(極肚餓 && 錢多){  
    吃自助餐;  
}否則{  
    吃快餐;  
}
```

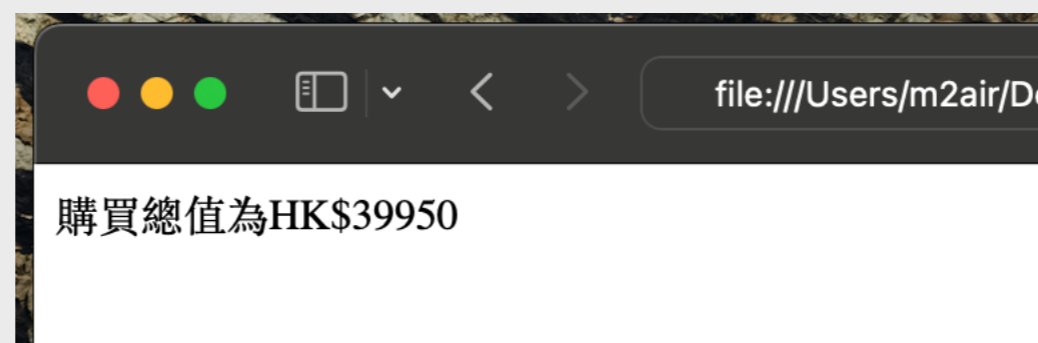
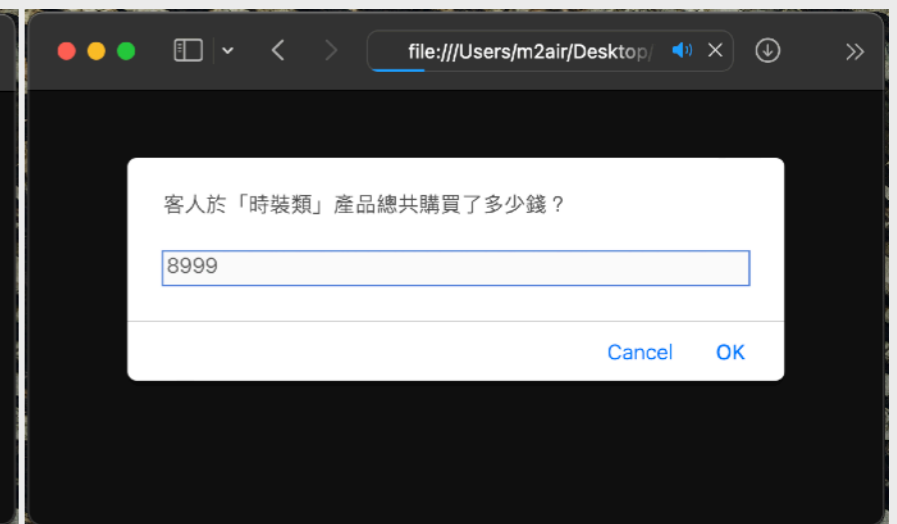
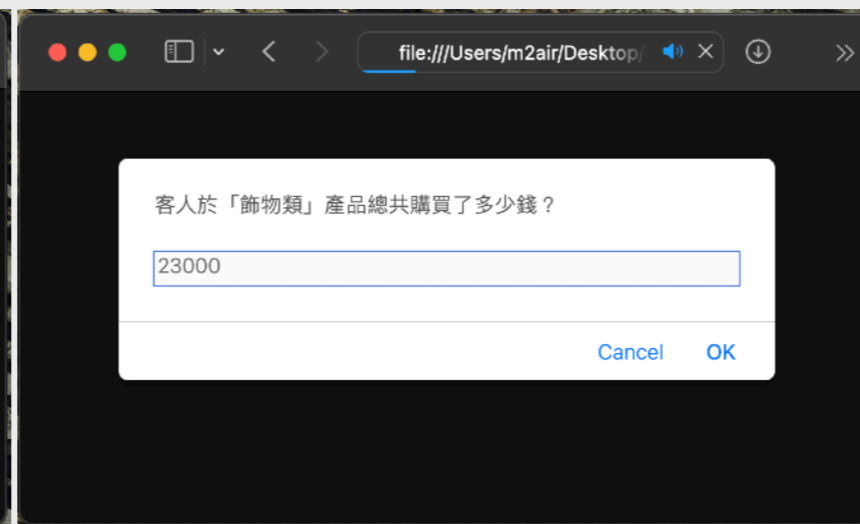
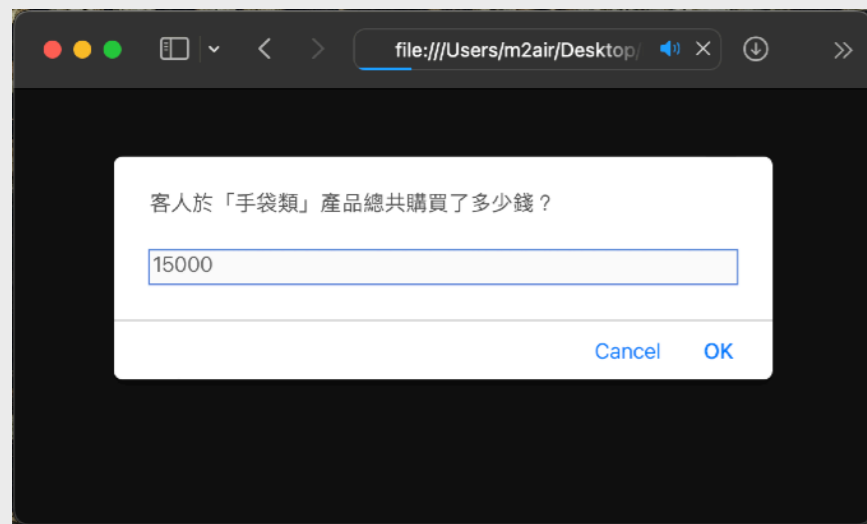
實例:

```
if(marks>=50 && attendance>=60)console.log('pass');  
else console.log('fail');
```

JavaScript練習7

為一所新開張的輕奢時尚精品店編寫一個簡單的程序，該店只售賣三類商品(手袋、飾物和時裝)，若顧客購買總值超過或等於\$25,000，則可享有85折，否則，購買總值減\$200。

1. 顯示歡迎語句，並透過prompt()要求用戶輸入所購買的手袋的總價格、所購買的飾物的總價格和所購買的時裝的總價格。同時把所輸入的數據透過parseFloat轉換成浮點數。
2. 判斷購買總值是否超過或等於25000，是則給予85折，否則減200。
3. 最後使用document.write以小數點無條件進位方式Math.ceil顯示購買總值參考



多選 — if else-if else

如果擁有排它情況，且多於兩個執行區塊，只能從中選一，就可以選擇用巢狀 if-else 敘述。將 if ... else 語句放在其他 if ... else 語句中

邏輯:

```
if(條件成立){  
    做某些程式;  
}else if(條件成立){  
    做某些程式;  
}else if(條件成立){  
    做某些程式;  
}else{  
    做某些程式;  
}
```

例一:

```
若(覺得精力充沛){  
    打羽毛球;  
}若以上不成立但(覺得精神尚可){  
    打乒乓球;  
}若以上不成立但(覺得精神一般){  
    玩美式桌球;  
}否則{  
    寫iPhone程式;  
}
```

例二:

```
若(肚餓){  
    吃自助餐;  
}否則{  
    寫程式;  
}
```

else if: 如果以上(都)唔中而中...

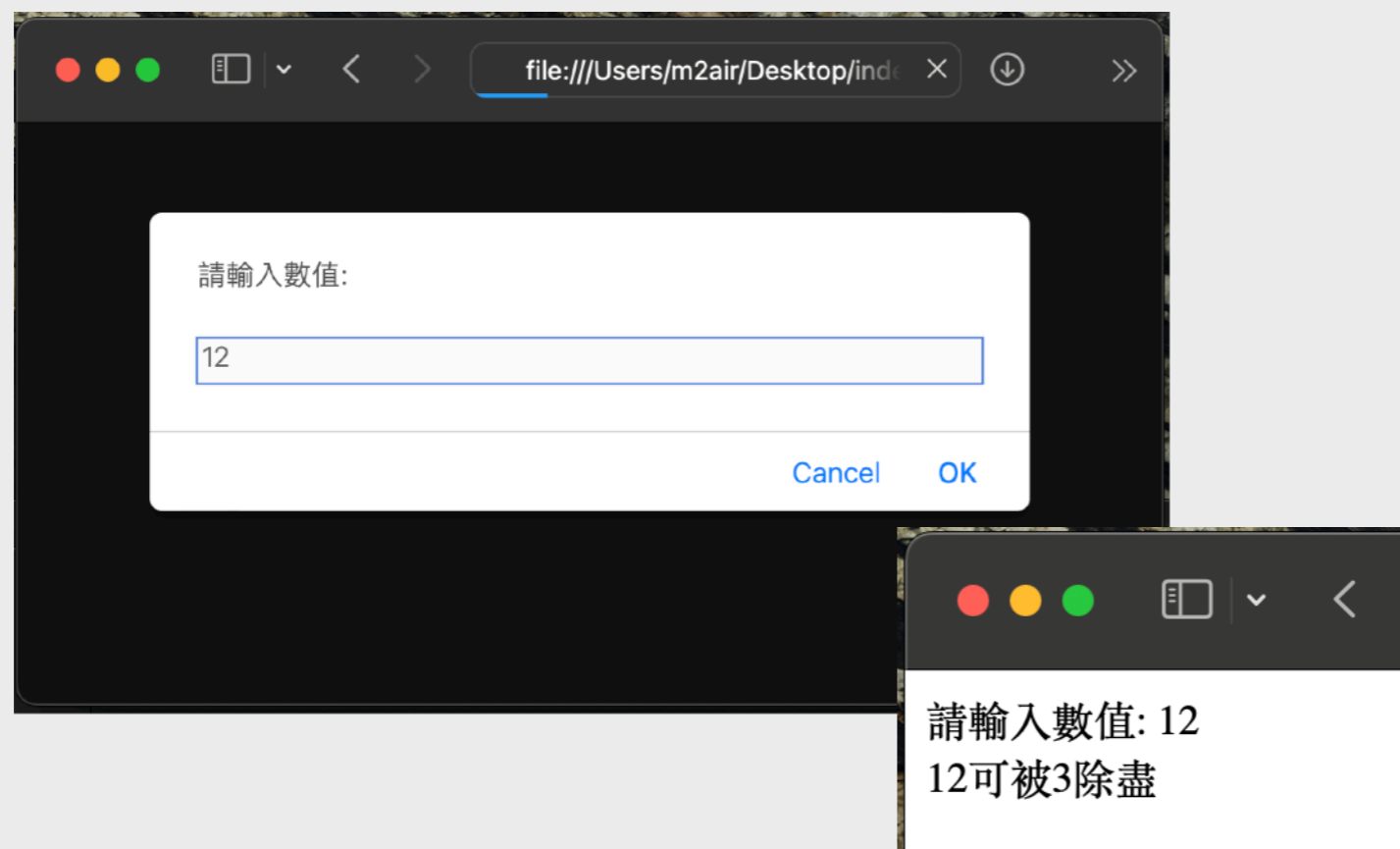
實例:

```
if(time>=12 && time<18)console.log('lunch or tea');  
else if(time>=18)console.log('dinner');  
else console.log('breakfast');
```

JavaScript練習8

參考以下程式，並作出修改，使程式可以讀取一個整數，然後：

- 如果該整數被5除盡，列印該數字加「可被5除盡」
- 如果該整數被3除盡，列印該數字加「可被3除盡」
- 如果該整數同時被3和5除盡，列印該數字加「可被5和3除盡」
- 餘下情況，列印該數字加「不可被5和3除盡」



更多例子:

請輸入數值: **40**
40可被5除盡

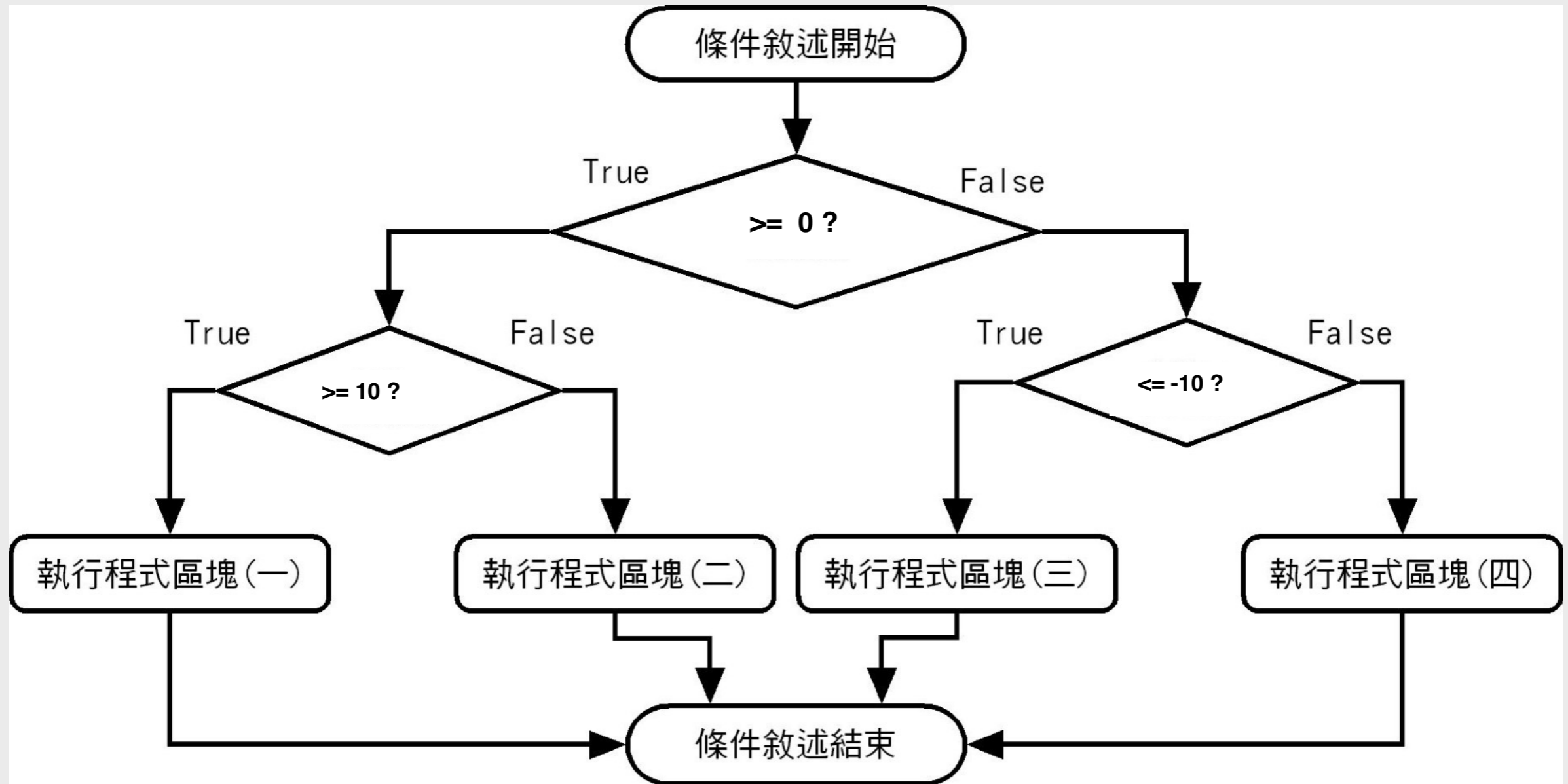
請輸入數值: **36**
36可被3除盡

請輸入數值: **30**
30可被5和3除盡

請輸入數值: **68**
68不可被5和3除盡

巢狀條件敘述：if 中有 if (1)

流程圖例子



巢狀條件敘述：if 中有 if (2)

程式例子

```
if (num >= 0) {  
    if (num >= 10) {  
        text1 = "The number is bigger than or equals to 10";  
    } else {  
        text2 = "The number is bigger than 0 but less than 10";  
    }  
} else {  
    if (num <= -10) {  
        text3 = "The number is smaller than or equals to -10";  
    } else {  
        text4 = "The number is smaller than 0 but bigger than -10";  
    }  
}
```

JavaScript練習9

承練習7，精品店新張期已過，試把原有的規則廢掉並套用以下新規則：

- 以「confirm」詢問客人是否VIP會員，若是，便詢問其會籍，會籍A者享7折，會籍B者享8折，會籍C者享9折，其他會籍一律享95折。
- 若非VIP會員，便詢問其是否首先購物，是者減200元。非者，問其是否第二次購物。
- 承上，若為第二次購物，減100元。非者，減50元。
- 最後亦如練習二一樣，以小數點無條件進位方式顯示購買總值。

```
file:///Users/m2air/Desktop/ind...
歡迎使用！
客人於「手袋類」產品總共購買了多少錢？ 15000
客人於「飾物類」產品總共購買了多少錢？ 2000
客人於「時裝類」產品總共購買了多少錢？ 34900
客人是VIP會員嗎？ false
客人是第一次購物嗎？ true
購買總值為HK$51700
```

```
file:///Users/m2air/Desktop/ind...
歡迎使用！
客人於「手袋類」產品總共購買了多少錢？ 4800
客人於「飾物類」產品總共購買了多少錢？ 52000
客人於「時裝類」產品總共購買了多少錢？ 19800
客人是VIP會員嗎？ true
請輸入客人的VIP種類: B
購買總值為HK$61280
```

總結 if else

- 三種選擇敘述:

單一選擇

- if 敘述 (statement):

- 如果條件為真(true)，則執行操作
- 如果條件為假(false)，則跳過它

二選一

- if...else 敘述 (statement):

- 如果條件為真，則執行操作；否則，便執行其他操作

多選一

- 巢狀 if...else 敘述 (statement):

- 根據運算式的值執行幾種動作之其中一種
- 多項選擇語句-在許多不同的動作中選擇其中一種

多選一 Switch Case (1)

Switch 多選一條件敘述比巢狀 if-else 多條件敘述來的清楚明白，程式碼也比較簡潔。它可以依照符合條件來執行不同區塊的程式碼，其語法如下所示：

```
switch (變數) {  
  case 值1:  
    程式區塊 1  
    break;  
  case 值2:  
    程式區塊 2  
    break;  
  default:  
    程式區塊 3  
}
```

- switch 敘述括號裡的變數，只能是number或string這兩種型態之一

- 只能是單一整數數值/字符/完串
- 無法判斷變數值是否大於或小於另一個值

- 當所有case都不為真時，可以使用默認default情況來執行任務

多選— Switch Case (2)

用來判斷在「甚麼情況」下去做「某些程式」。而比If Else則多了「連下面的case也都做」的概念。「default」在不需要時不用寫。

邏輯:

```
switch(變數名稱){  
  case 值一:  
    做某些程式;  
  case 值二:  
    做某些程式;  
  default:  
    若以上選擇皆不  
    中便做的某些程式;  
}
```

例一:

```
選擇(學歷){  
  情況 "博士":  
    適合教深造課程();  
  情況 "碩士":  
    適合教學士課程();  
  情況 "學士":  
    適合教副學位課程();  
  當以上都不是:  
    適合教興趣班();  
}
```

實例:

```
const eduLevel="master";  
switch (eduLevel) {  
  case "doctoral":  
    console.log("postgrad");  
  case "master":  
    console.log("undergrad");  
  case "bachelor":  
    console.log("sub-degree");  
  default:  
    console.log("interest-classes");  
}
```

若果我們把eduLevel賦值為master，因各個case內都不放break，故此master學歷便可以教授undergrad、sub-degree與interest-classes

多選一 Switch Case (3)

(加Break)

為Switch Case加上break號，令其變成為多選一的條件判斷式。

邏輯:

```
switch(變數名稱){  
    case 值一:  
        做某些程式;  
        break;  
    case 值二:  
        做某些程式;  
        break;  
    default:  
        若以上選擇皆不  
        中便做的某些程式;  
}
```

例一:

```
選擇(節日){  
    情況 "農曆新年":  
        吃蘿蔔糕();  
        break;  
    情況 "中秋節":  
        吃月餅();  
        break;  
    情況 "端午節":  
        吃粽();  
        break;  
    當以上都不是:  
        隨意吃();  
}
```

實例:

```
switch (festival) {  
    case "Lunar New Year":  
        eatTurnipCake();  
        break;  
    case "Mid-Autumn":  
        eatMooncake();  
        break;  
    case "Dragon Boat":  
        eatRiceDumpling();  
        break;  
    default:  
        freeToEat();  
}
```

多選— Switch Case (4)

sum是否相等於10?
false 否
跳至下一個case

sum是否相等於20?
true 是

執行case 20內的所有
敘述

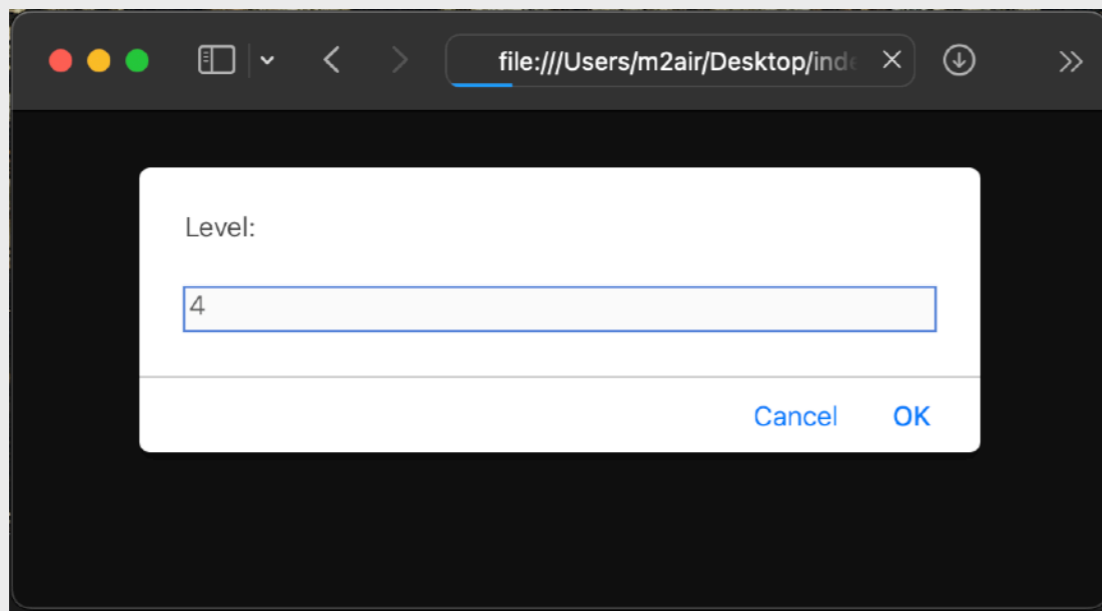
break (中斷):
跳離switch

```
const sum = 20;
switch(sum){
  case 10:
    document.write("當sum=10則進入此敘述");
    break;
  case 20:
    document.write("當sum=20則進入此敘述");
    break;
  case 30:
    document.write("當sum=30則進入此敘述");
    break;
  default:
    document.write("當sum的值不符合上述任合一個條件則進入此敘述");
}
document.write("Bye");
```

當sum=20則進入此敘述Bye

Case的範圍縮寫法

若能做到「範圍」的效果，參考右方例子，例如case 0與1的結果相同，case 2與3的結果相同，case 4與5的結果相同，我們便可使用「組合」的方法來編寫case內容。



```
const level = prompt('Level:');
let result;
switch(parseInt(level)){
  case 0:
  case 1:
    result="難吃";
    break;
  case 2:
  case 3:
    result="普通";
    break;
  case 4:
  case 5:
    result="好吃";
    break;
  default:
    result="未評分";
}
document.write("這次的食物是"+result);
```

條件判斷: For Loop (1)

- 允許我們多次執行語句或一系列語句
- 它們通常被稱為循環/迴圈
- 與條件陳述(conditional statement)一樣，由布林值(boolean)表達式控制
- JavaScript有三種重複語句：
 - for
 - while
 - do...while
- 程序員應該因應不同的情況選擇正確的迴圈語句

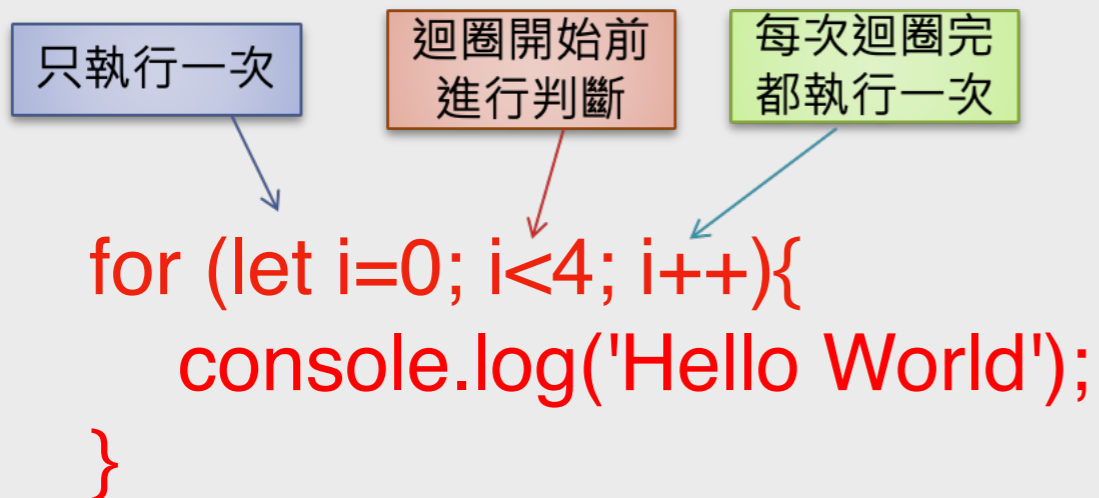
條件判斷: For Loop (2)

```
for(起始式; 條件式; 步進式){  
    程式語句;  
}
```

- **起始式** – 進入迴圈時，一開始執行的程式運算式，只執行一次。通常起始式，多為設定控制迴圈執行變數的起始值。
- **條件式** – 判斷是否重複執行迴圈的依據。如果條件式的結果為 true，則繼續執行迴圈；如果為 **false**，則跳離迴圈。
- **步進式** – 每經過一次迴圈，就會執行一次的運算式。通常步進式多為設定控制迴圈執行變數的遞增或遞減運算式。

條件判斷: For Loop (3)

例如：連續印出4行「Hello World」，如下所示：



i	輸出
0	Hello World
1	Hello World
2	Hello World
3	Hello World
4	

上述程式碼會重覆執行4次，變數 i 值依序為0、1、2、3和4，當變數 i 值為4時，由於判斷 $i < 4$ 為 false，所以會離開迴圈

條件判斷: For Loop (4)

例如：連續印出4行「Hello World」，如下所示：

版本1:

```
for (let i = 1; i <= 4; i++){  
    console.log('Hello World');  
}
```

版本2:

```
for (let i = 4; i >= 1; i--){  
    console.log('Hello World');  
}
```

版本3:

```
for (let i = 0; i < 4; i++){  
    console.log('Hello World');  
}
```

總結 For Loop

用途: 因應條件而進行指定次數的迴圈。

邏輯: `for(起始式; 條件式; 步進式){`
 做某些程式;
`}`

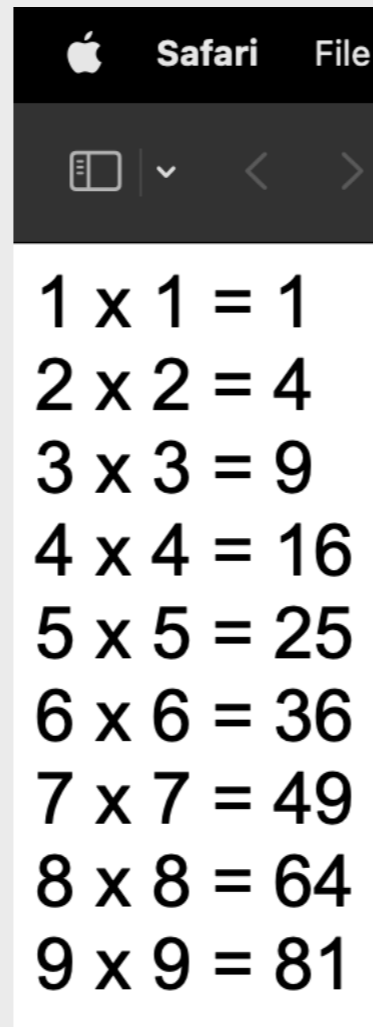
例一: 迴圈(開始步數=0; 步數少於3; 步數加上1){
 行一步;
`}`

例二: 迴圈(胃子飽滿度=0; 胃子飽滿度少於100; 胃子飽滿度加上1){
 吃一口飯;
`}`

實例: `for(let i=0; i<3; i++){`
 `console.log('walk');`
`}`

JavaScript練習10

連續印出1-9的平方：



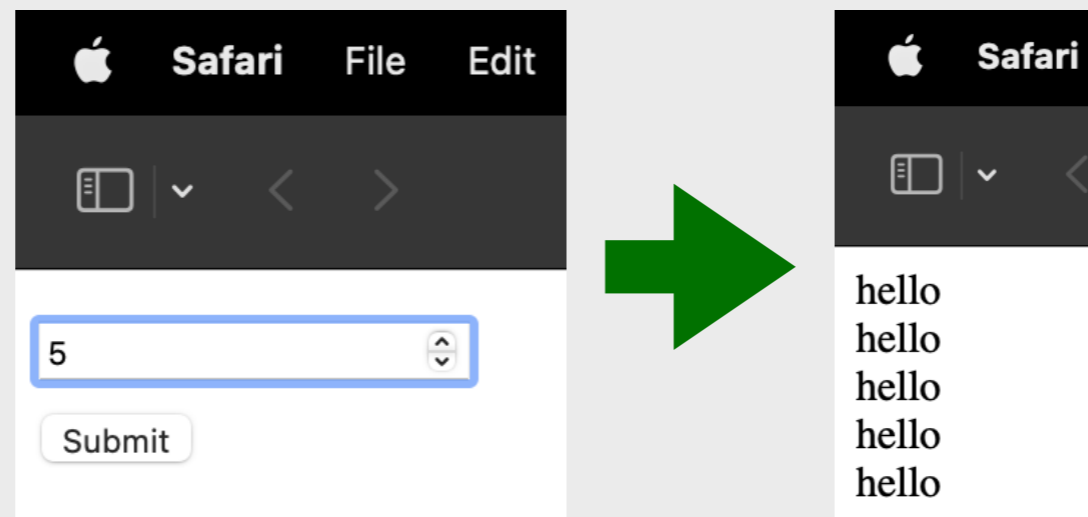
JavaScript練習11

利用 for-loop 搭配 if 條件判斷來設計一個 JavaScript 程式，將 1 到 10 中的偶數輸出。



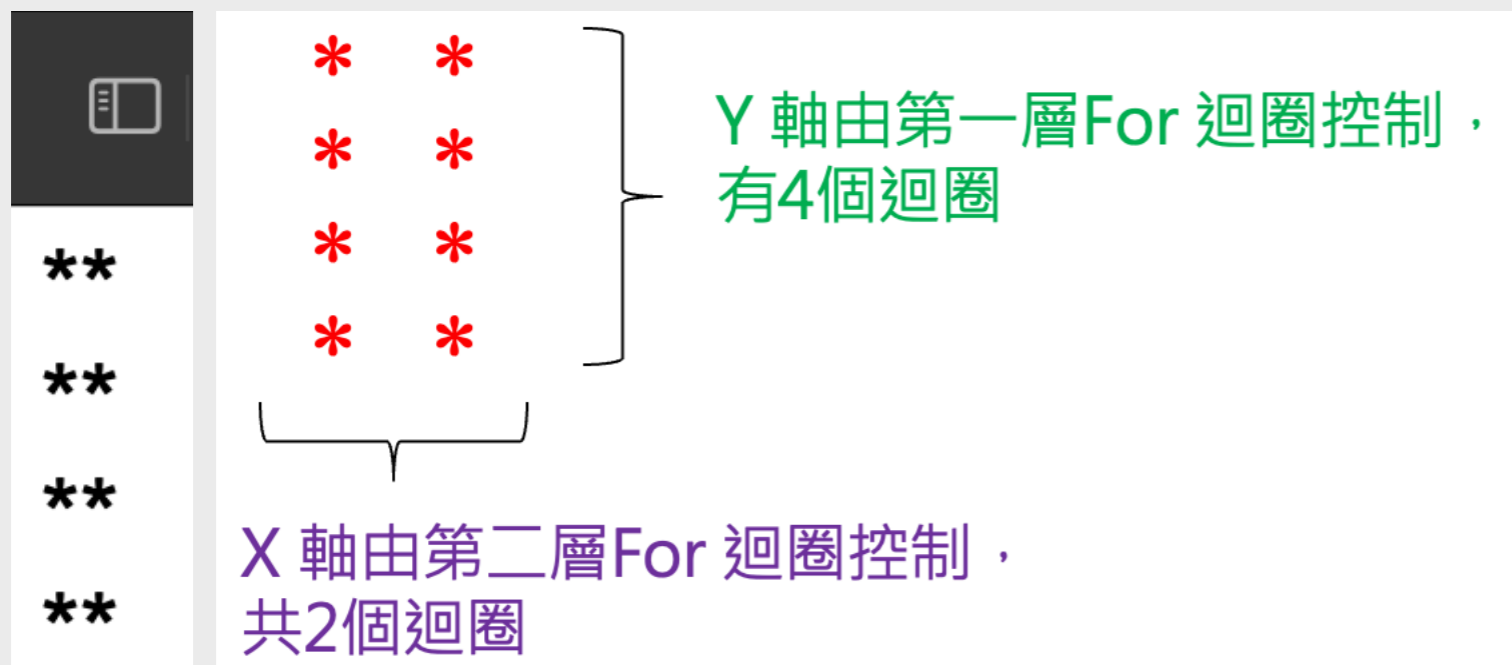
JavaScript練習12

程式根據用戶輸入執行次數而執行次數顯示hello



For 巢狀迴圈-說明

For巢狀迴圈能用作多次元空間圖形顯示。例如兩層的For迴圈即可畫出二維圖案，如下圖：



```
<script>
  for (let i=4; i>0; i--){
    for (let j=2; j>0; j--){
      document.write('*');
    }
    document.write('<br>'); // 換行
  }
</script>
```

JavaScript練習13

程式印出下圖乘數表，如下所示：

```
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25
6 12 18 24 30 36
```

Y 軸由第一層 For 迴圈控制，有6個迴圈

X 軸由第二層For 迴圈控制，迴圈數目根據當時Y 軸的迴圈數值而變更

JavaScript練習14

小龜🐢喜歡一邊行走一邊吃蛋糕🍰和月餅🥮。試以 for loop 和 if statement 並以 prompt 寫出一個詢問小龜的喜好。

1. 詢問小龜要行走多少步
2. 詢問小龜每步要吃多少次(件)蛋糕
3. 詢問小龜在吃完哪一步的哪一件蛋糕後要吃月餅
(a. 先詢問步數， b. 後詢問件數)
4. 顯示出每步吃蛋糕及(或)月餅的狀態

1. 小龜🐢要行走多少步？

Cancel OK

2. 小龜每步吃多少次蛋糕🍰？

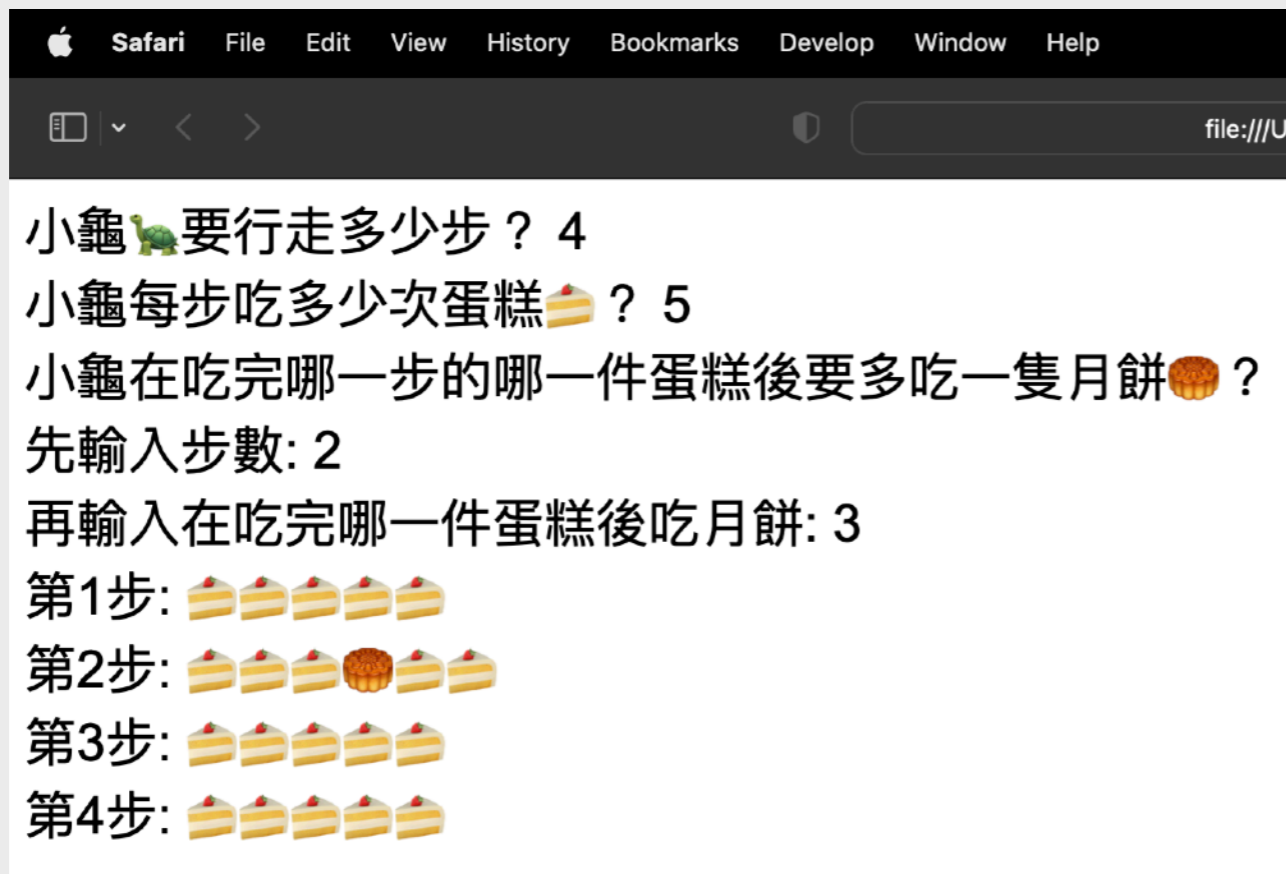
Cancel OK

3a. 小龜在吃完哪一步的哪一件蛋糕後要多吃一隻月餅🥮？
先輸入步數：

Cancel OK

3b. 再輸入在吃完哪一件蛋糕後吃月餅：

Cancel OK



條件判斷: While Loop (1)

用途: 因條件而進行迴圈。

邏輯: `while(條件成立){
 做某些程式;
}`

例一: `當(有力氣){
 跑步;
}`

例二: `當(胃子還沒爆開 && 還未夠時間){
 繼續吃自助餐;
 胃子飽滿度++;
 時間- -;
}`

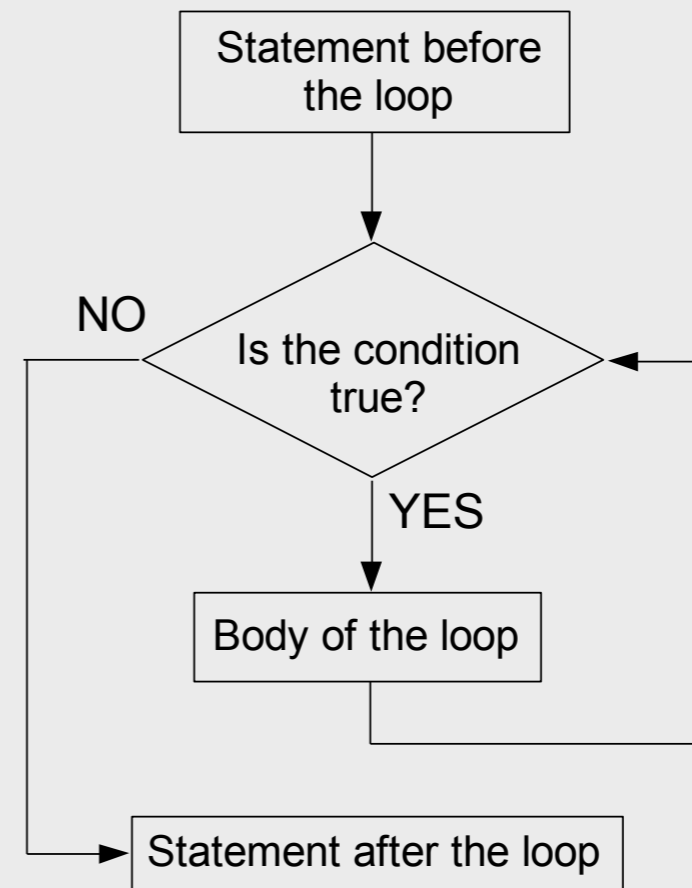
實例: `while(feelingGood){
 run(); energy--;
 if(energy<20)feelingGood=false;
}`

條件判斷: While Loop (2)

While 迴圈：

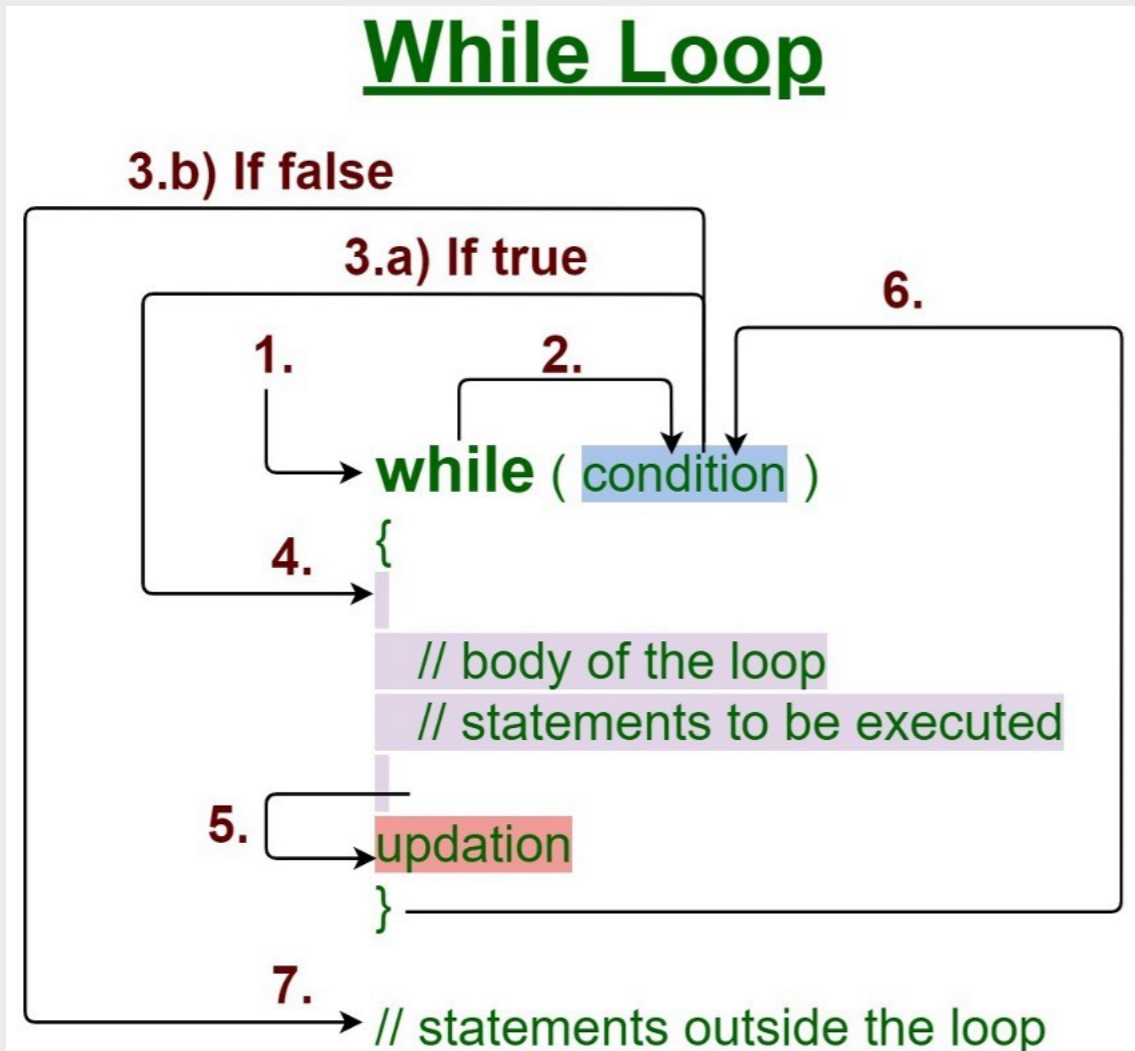
- 在條件為真時重複執行while迴圈主體內的程序。
- while迴圈內的主體可以是單個語句或塊。
- 最終，條件將變為假。此時，重複終止，並且執行重複語句之後的第一條語句。

```
statementsBeforeLoop;  
while( condition ) {  
    statement_1;  
    ...  
    statement_n;  
}  
statementsAfterLoop;
```

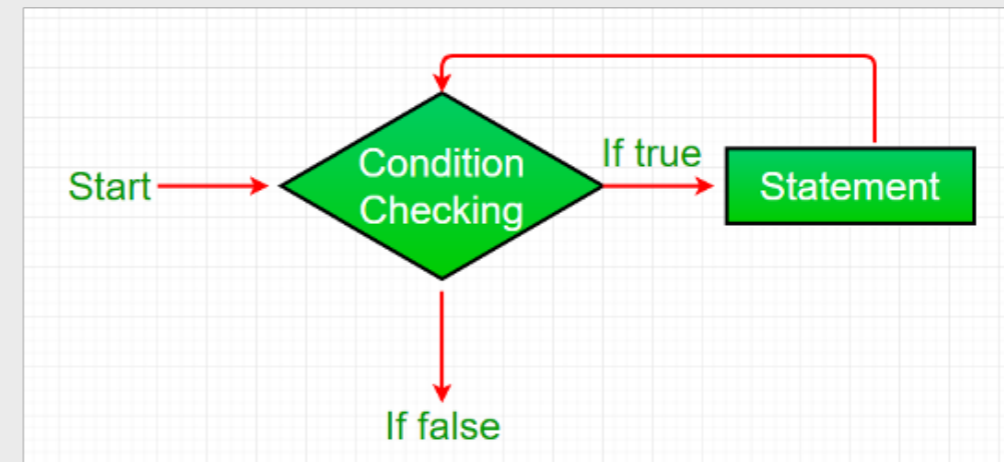


條件判斷: While Loop (3)

While Loop



```
statementsBeforeLoop;  
while( condition ) {  
    statement_1;  
    ...  
    statement_n;  
}  
statementsAfterLoop;
```



條件判斷: While Loop (4)

while迴圈 - 例子：

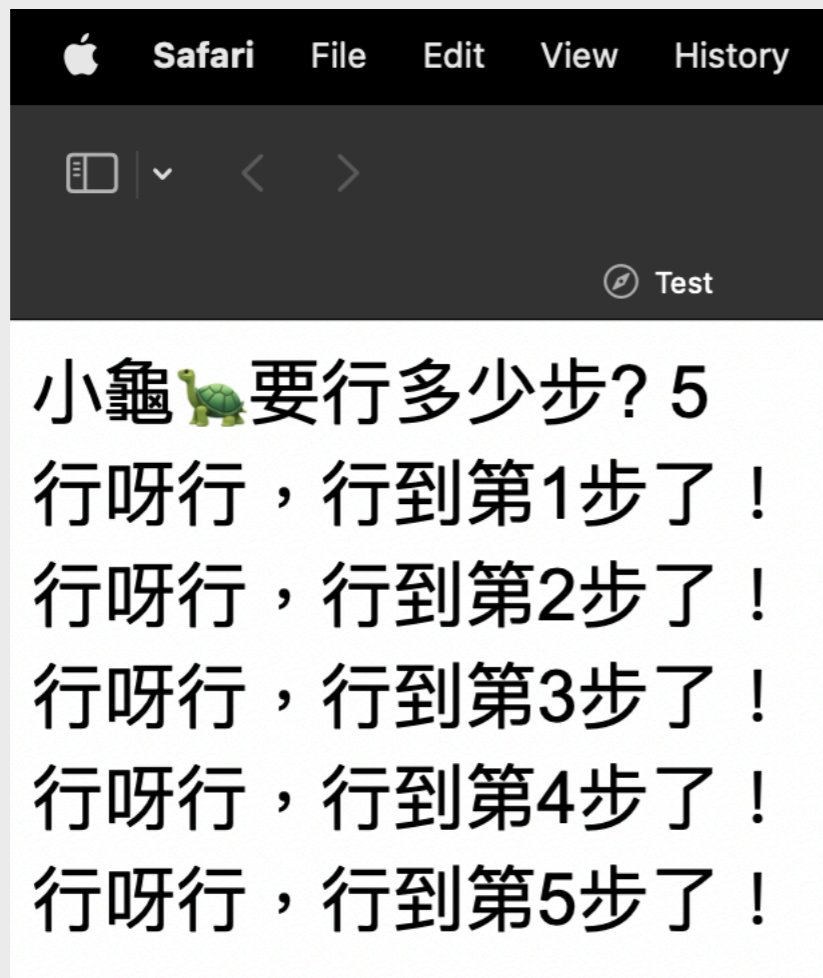
```
let i=1;  
while(i<=3){  
  document.write(i+'<br>');  
  i++;  
}
```

1
2
3

重複次數	變數	條件: $i \leq 3$	動作
1st	$i = 1$	$1 \leq 3 \rightarrow \text{true}$	1."1" is printed. 2. $i = 1 + 1 \rightarrow i$ is increased to 2
2nd	$i = 2$	$2 \leq 3 \rightarrow \text{true}$	1."2" is printed. 2. $i = 2 + 1 \rightarrow i$ is increased to 3
3rd	$i = 3$	$3 \leq 3 \rightarrow \text{true}$	1."3" is printed. 2. $i = 3 + 1 \rightarrow i$ is increased to 4
4th	$i = 4$	$4 \leq 3 \rightarrow \text{false}$	The loop is terminated.

JavaScript練習15

以 prompt 詢問小龜要步行多少步，然後使用 while loop 顯示出其步行步數



JavaScript練習16

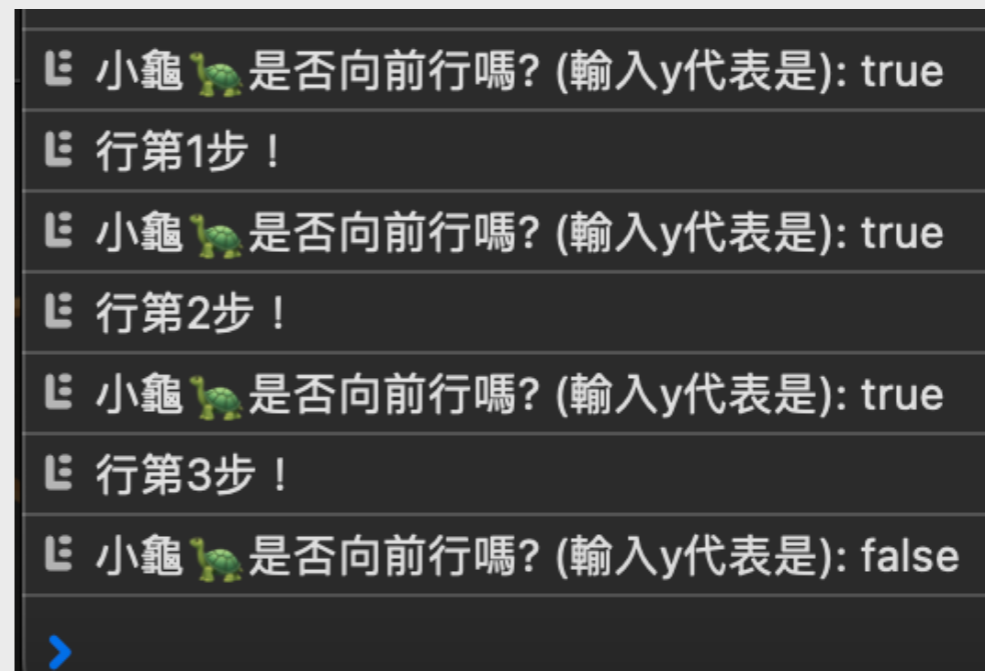
- 小龜喜歡一邊步行一邊作詩，試寫一個小程序，詢問「小龜🐢是否向前行嗎？(輸入y代表是)」，若輸入 y 即繼續迴圈(大小寫也接受)，其他一律停止迴圈。
- 若小龜步行到第 7 步時便會顯示「7步成詩了！」，然後使用 break 以停止步行。
- 提示：程式運用 **while loop** 與 **if statement**，當中使用了 **break** 來停止 while loop 的運行。

例子一：



```
× □ □ □ ⊕ | ! Elements
All Evaluations Errors Warnings Logs
小龜🐢是否向前行嗎? (輸入y代表是): true
行第1步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第2步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第3步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第4步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第5步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第6步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第7步!
7步成詩了!
```

例子二：



```
小龜🐢是否向前行嗎? (輸入y代表是): true
行第1步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第2步!
小龜🐢是否向前行嗎? (輸入y代表是): true
行第3步!
小龜🐢是否向前行嗎? (輸入y代表是): false
```

條件判斷: Do While Loop

用途: 先執行一次迴圈的內容，再因條件是否成立而繼續進行迴圈。

邏輯:

```
do{  
    做某些程式;  
}while(檢查條件)
```

例一:

```
做了先{  
    跑步; 力氣--;  
}然後才判斷是否(有力氣)
```

例二:

```
做了先{  
    拿一轉食物來吃; 胃部飽滿度++; 時限--;  
}然後才判斷是否(胃部還沒爆開 && 自助餐限時還未完結)
```

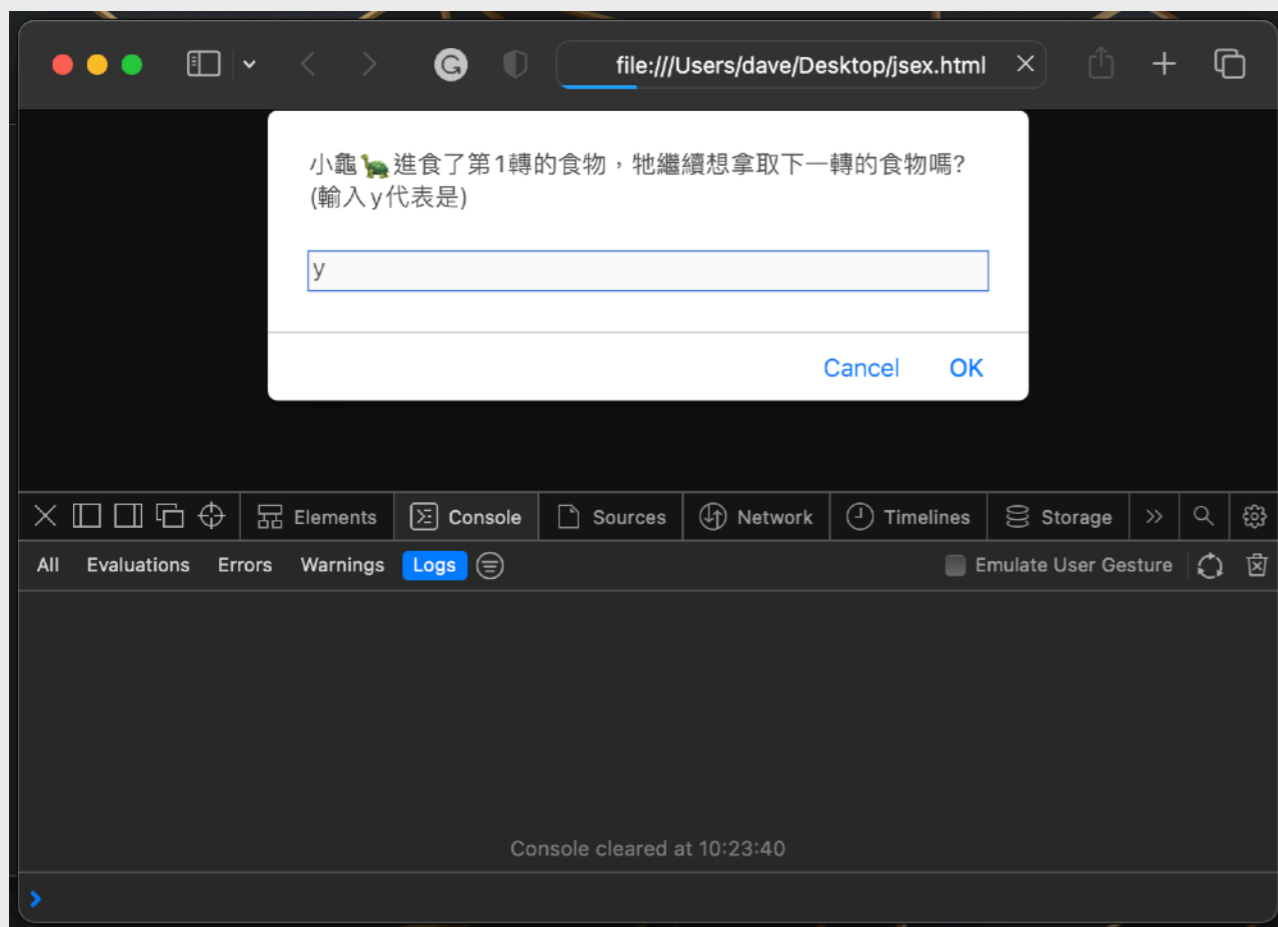
實例:

```
do {  
    eat();  
    hungry--;  
}  
while(hungry>=20);
```

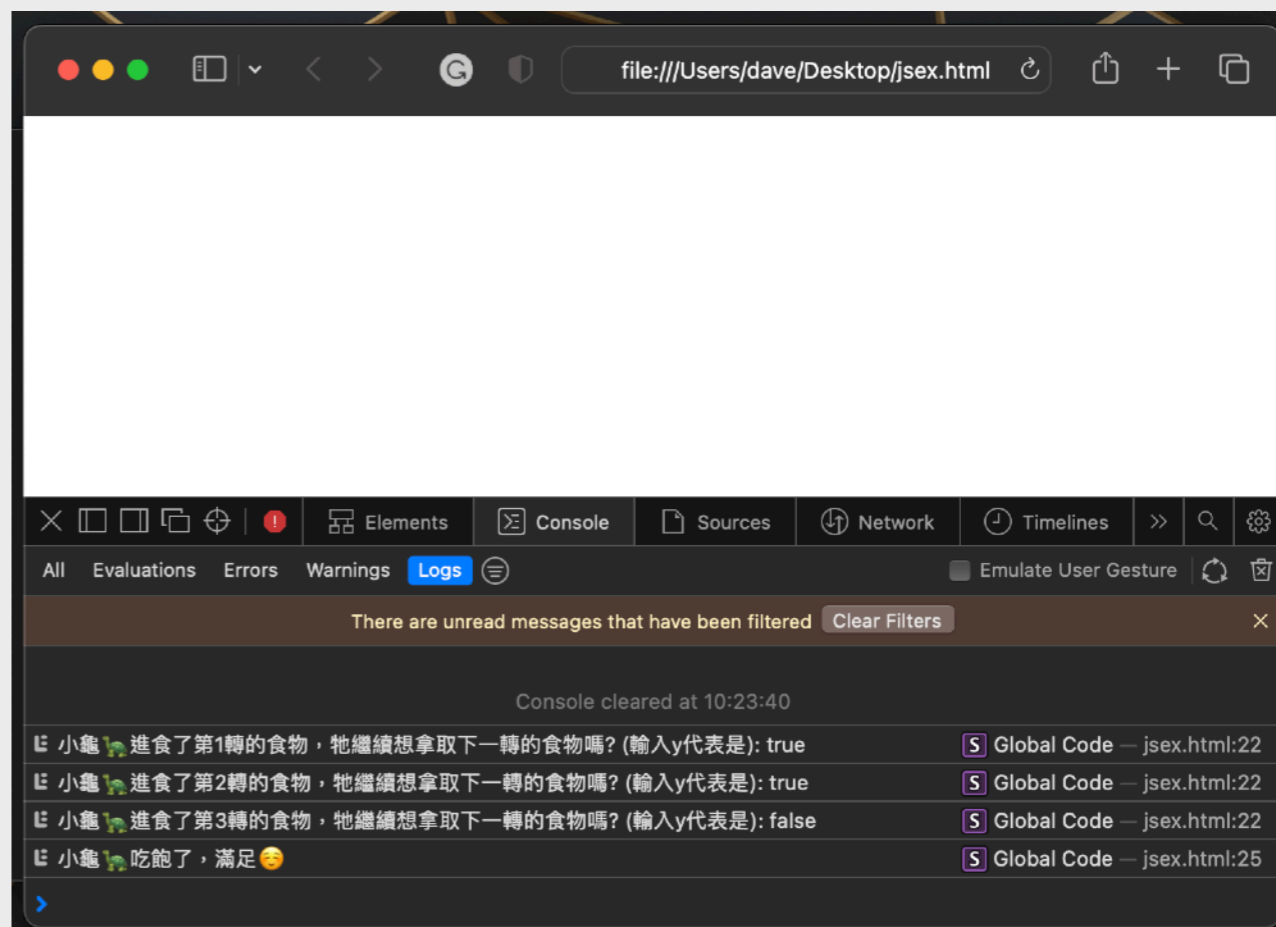
JavaScript練習17

- 小龜喜歡吃自助餐，試以 **do-while loop** 來顯示牠進食了的次數並詢問：「小龜 🐢 進食了第 i 轉的食物，牠繼續想拿取下一轉的食物嗎? (輸入y代表是)」
- 當用戶輸入「y」時，則繼續迴圈，否則停止迴圈
- 停止迴圈後顯示「小龜 🐢 吃飽了，滿足 😊」

例子一：



例子二：



Looping加Break

Break: 強制跳出整個迴圈Looping，適用於For、While及Do-While

類似Stop(停止)的意思

生活例子：

你在街上不斷(loop)尋找餐廳吃飯，當你找到了心怡的餐廳，你便會停止(break)再繼續尋找餐廳，所以即便完成了這個「尋找餐廳」的迴圈

程式例子：

```
for(let i=0; i<restaurants.length; i++){  
  if(goodToTry(restaurants[i])){  
    goInside(restaurants[i]);  
    break;  
  }  
}
```

Looping加Continue

Continue: 強制跳出是次迴圈Looping

類似Skip(略過)/「唔啱就Next」的意思

生活例子：

在網上尋找工作時，我們會先看職位名稱，若不感興趣便會略過而不按進去看詳細資料並繼續(continue)查看下一份工作

程式例子：

```
for(let i=0; i<jobs.length; i++){  
  if(!goodToKnowMore(jobs[i]))continue;  
  seeDetails(jobs[i]);  
}
```

Method(方法)或Function(功能) - 1

在物件導向程式的世界中，一個類別 (Class)就像是一藍圖(blueprint)，而藍圖裡列明了依據該藍圖而產生出來的實體以及藍圖自身的各項功能。

例如「智能貓 (SmartCat)」是一個類別(藍圖)，我們可以透過此藍圖來生產出智能貓。而智能貓廚藝精湛，懂得煮生菜即食麵，也懂得煲出人蔘杞子湯。故此，「煮生菜即食麵」和「煲人蔘杞子湯」便是所有智能貓皆擁有的「功能」，或稱作為「方法 (Method)」。

當然，智能貓並不只有兩個功能，它們還懂得做家務和維修電腦，所以「做家務」和「維修電腦」都是智能貓的方法。

Method(方法)或Function(功能) - 2

沒有輸入和輸出的Method實例：

```
function 自行找材料煮即食麵並放到客廳(){  
    // 工序內容...  
}
```

有輸入但沒有輸出的Method實例：

```
function 煮即食麵並放到客廳(材料){  
    // 工序內容...  
}
```

沒有輸入但有輸出的Method實例：

```
function 自行找材料煮即食麵(){  
    // 工序內容...  
    return 即食麵;  
}
```

有輸入也有輸出的Method實例：

```
function 煮即食麵(材料){  
    // 工序內容...  
    return 即食麵;  
}
```

Method(方法)或Function(功能) - 3

沒有輸入和輸出的功能程式碼實例：

```
function walk(){  
    /* ... */  
}
```

有輸入但沒有輸出的功能程式碼實例：

```
function walk(steps){  
    /* ... */  
}
```

沒有輸入但有輸出的功能程式碼實例：

```
function walk(){  
    /* ... */  
    return steps;  
}
```

有輸入也有輸出的功能程式碼實例：

```
function walk(steps){  
    /* ... */  
    return steps;  
}
```

在JavaScript中，我們以小寫「function」來定義一個功能(正名為「函式」)的程式組

在功能的名稱後，我們需要加入開關括號以傳入參數，如沒有參數需傳入，則漏空，即只寫下開關括號便可

如需傳入多個參數，使用逗號分隔便可

在JavaScript中建立Function

實體的Function:

```
function doSomething(){  
    // Do something  
}
```

把Function當作變數處理:

```
const fun={()=>{  
    // Do something  
}};
```

把Function儲存到JSON中:

```
const cat={"walk":()=>{  
    // Walk  
}};
```

或

```
const cat={};  
cat.walk={()=>{  
    // Walk  
}};
```

JavaScript ES6把Function當作Class(類別)中的Method處理:

```
class PersonES6{  
    constructor(name){  
        this.name=name;  
    }  
    showNameAge(){  
        console.log('Name: '+this.name);  
    }  
}
```

課題四： JSON及本機儲存

陣列(Array)與字典(Dictionary) - 1

在JavaScript裡，陣列與字典即統稱為：
JSON (JavaScript Object Notation)

兩者意思有點像「櫃桶」或「收納箱」的概念

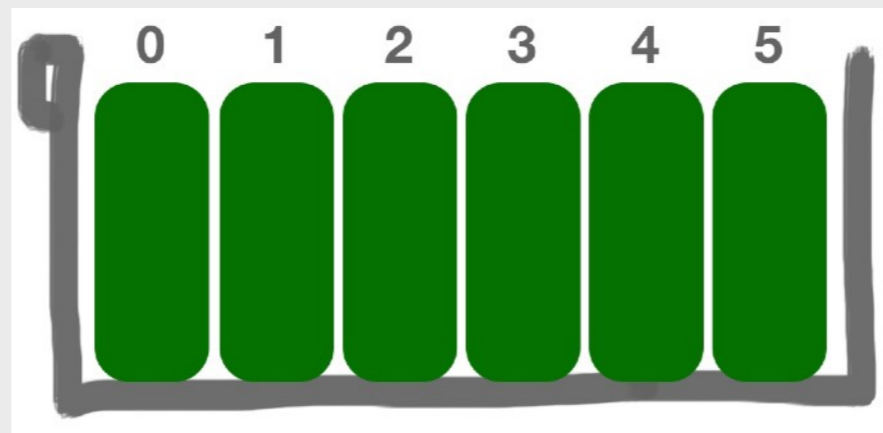
兩者都用來存放物件或變數

陣列(Array)

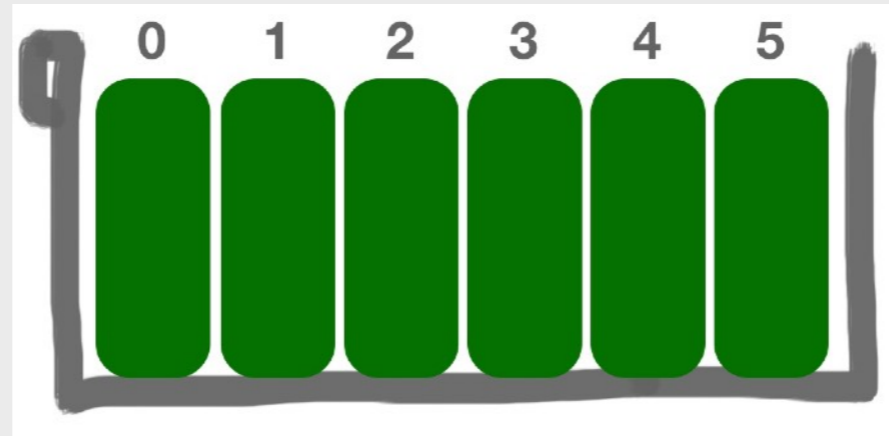
(索引)陣列即是一個列隊，就像排隊一樣，物件由0開始排起

(索引)陣列中的變數或物件並沒有自己的名稱

若要找出它們中的的其中一員，則要呼叫它的名次，
而它們的名次則是由0開始計起



陣列(Array)



上圖就是一個「索引陣列」(櫃桶)的例子，其陣列(櫃桶)中收納了6件物件，物件以「列隊」方式來儲存。

基於它們並沒有各自的名字，當我們要找出其中之一員時，便要呼叫它的名次。

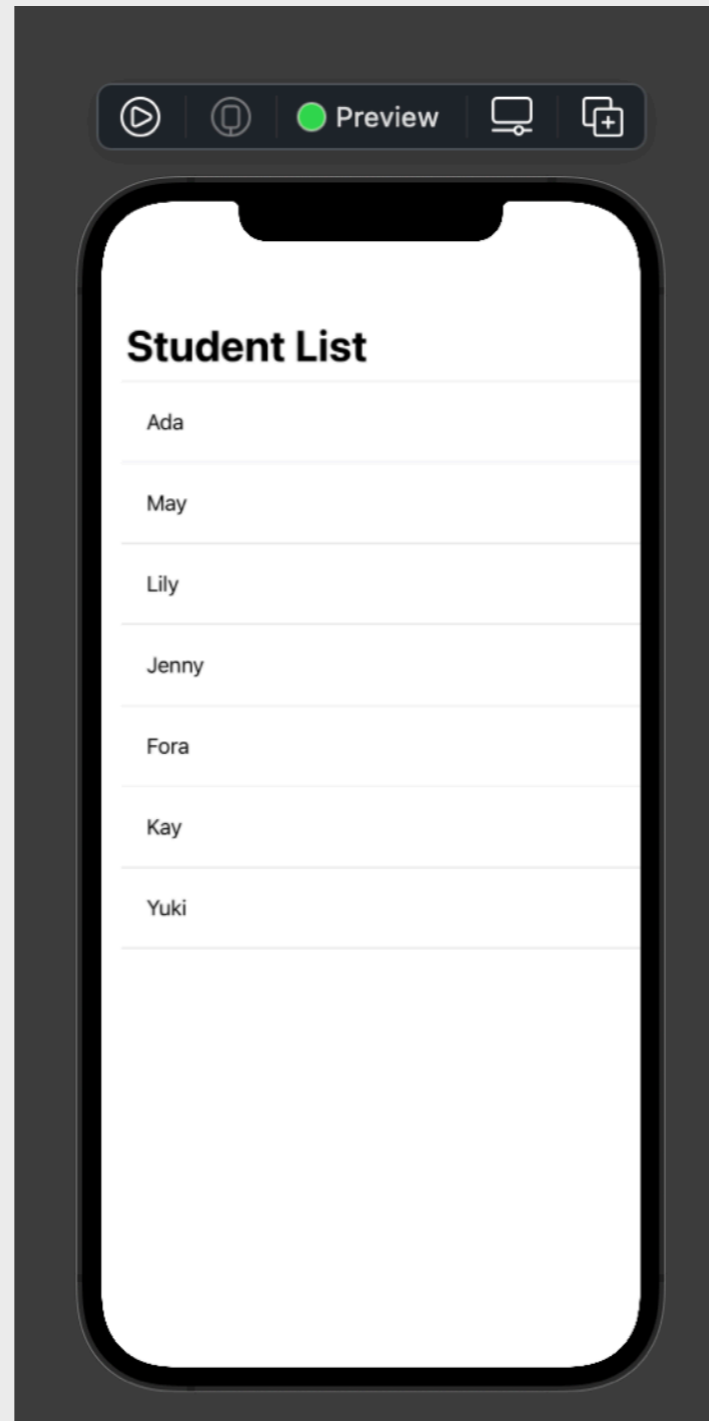
而由於它們的名次由0開始計起，例如我們需要找出「第一件」物件時，我們便需要呼叫出「0號」以找出「第一件」物件；當我們需要找出「第三件」物件時，我們便需要呼叫出「2號」以找出「第三件」物件。

索引陣列的寫法例子為：

```
let students=["Ada","May","Lily","Jenny","Fora","Kay","Yuki"];
```

陣列實作

```
let students=["Ada","May","Lily","Jenny","Fora","Kay","Yuki"];
```



字典(Dictionary)

字典以鍵值配對(key-value pair)型式而儲存物件或變數

每一個變數或物件皆有自己的名字

若要找出它們中的的其中一員，則要呼叫它的名字

與定義變數名稱一樣，建議使用英文字母、數字和底線，
不建議使用任何特殊符號

std_id: s001	name: Ada	gender: F	course: DEng
mode: full-time	age: 28	addr: xxxxxx	tel: yyyyyy

字典(Dictionary)

std_id: s001	name: Ada	gender: F	course: DEng
mode: full-time	age: 28	addr: xxxxxx	tel: yyyyyy

上圖就是一個「鍵值配對陣列」(櫃桶)的例子，其陣列(櫃桶)中收納了8件物件，物件並不以「列隊」方式來儲存。

基於它們皆擁有各自的名字，當要找出其中之一員時，呼叫該員的名字便可。

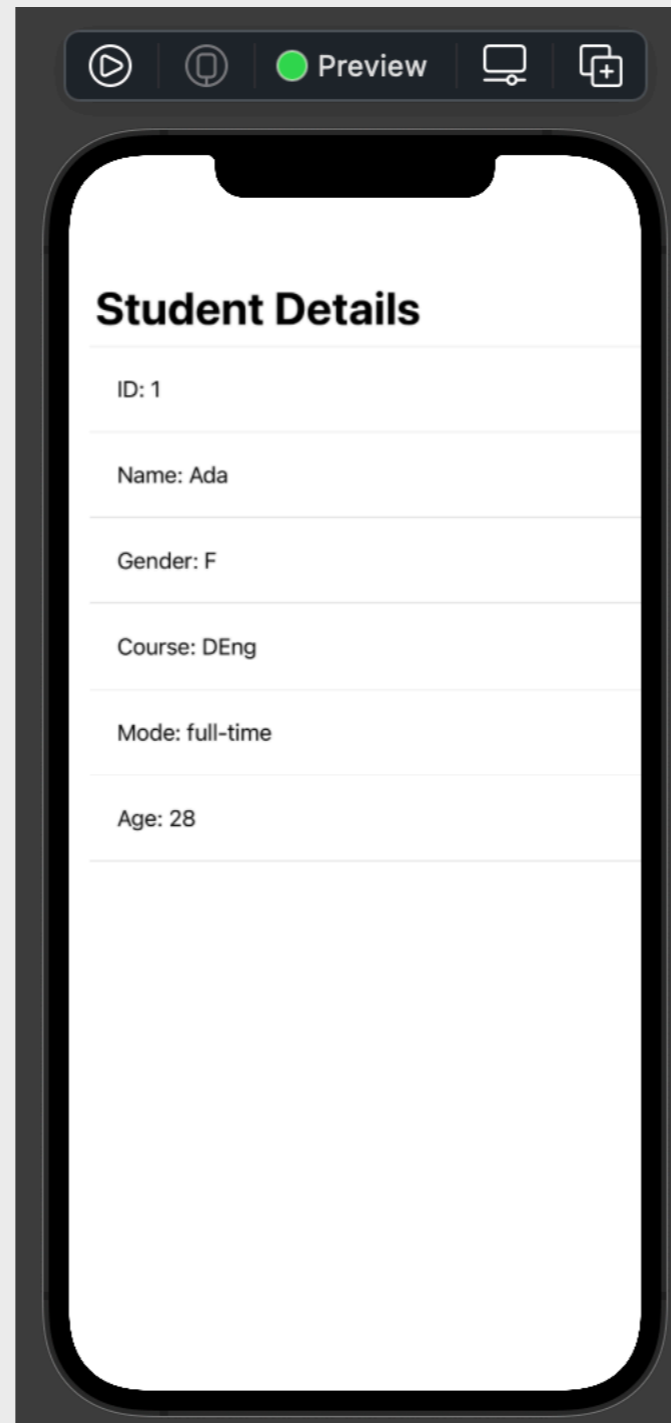
例如我們需要找出「name」物件時，我們呼叫「name」便可，若要找出「mode」物件，呼叫「mode」便可。

字典的寫法例子為：

```
let student={"std_id":1, "name":"Ada", "gender":"F",  
            "course":"DEng", "mode":"full-time", "age":28};
```

字典實作

```
let student={"std_id":1, "name":"Ada", "gender":"F", "course":"DEng", "mode":"full-time", "age":28};
```



陣列(Array)與字典(Dictionary) - 2

陣列中可以儲存無限個子陣列或子字典，
字典中亦可以儲存無限個子陣列或子字典
而子陣列或子字典中亦可儲存無限個子陣列或子字典，如此類推...

例子:

```
let foods=[  
  {"id":1, "name":"日本毛蟹", "photo":"01.jpg", "price":1080},  
  {"id":2, "name":"半島甜點", "photo":"02.jpg", "price":180},  
  {"id":3, "name":"龍蝦湯麵", "photo":"03.jpg", "price":280}  
];
```

```
let teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP203","level":2}]}  
];
```

陣列與字典實作

```
let foods=[  
  {"id":1, "name":"日本毛蟹", "photo":"01.jpg", "price":1080},  
  {"id":2, "name":"半島甜點", "photo":"02.jpg", "price":180},  
  {"id":3, "name":"龍蝦湯麵", "photo":"03.jpg", "price":280},  
  {"id":3, "name":"花鵬醉龍", "photo":"04.jpg", "price":380},  
  {"id":3, "name":"抹茶雪糕", "photo":"05.jpg", "price":60}  
];
```



陣列與字典: 後記

在Objective-C的世界裡，我們會使用NSArray或NSMutableArray作為陣列，使用NSDictionary或NSMutableDictionary作為字典。

在Swift的世界裡，Dictionary字典與Array陣列都是以 `[]` 來開頭和作結尾如：
`["name": "龍蝦湯麵", "photo": "03.jpg"]`

我們可會使用PHP在伺服器中處理陣列或字典，在PHP的世界裡，Array與Dictionary字典都是以 `[]` 來開頭和作結尾，並以 `=>` 來指派物件或變數如：
`["name" => "龍蝦湯麵", "photo" => "03.jpg"]`

本機儲存 (1)

若需要「永久」儲存變數或資訊到App中(不是儲存到雲端伺服器)，當用戶關閉了網站程式後而再次開啟該網站程式時都能存取到已儲存的變數或資訊，那麼，我們便需要使用到本機儲存(Local Storage)。然而，這種儲存模式是儲存在手機本機裡面而不是網上的任何地方，所以本機儲存不依賴網絡與互聯網(Internet)；而且只限擁有它的該個網站程式來存取，其他網站程式都不能存取，十分安全。本機儲存的壽命理論上是永久的，只有在網站程式主動要求，才能把已儲存的資料刪除。然而，在某些瀏覽器上，特別是手機或平板電腦上的瀏覽器，或是原生流動應用程式(Native Mobile App)內使用的嵌入式瀏覽器都是不可靠的，可能會因為各種原因(比如說退出App、網絡切換、內存不足等原因)而被清空。而且我們通常儲存的資料不會太多，目前業界基本上統一為上限5MB。一般來說，我們都只會使用它來儲存一些重要而簡短的資料如用戶User ID、程式個體ID、顯示語言、外觀色系等資訊。儲存、取出資料的函式為localStorage.setItem和localStorage.getItem。

使用**localStorage.setItem(鍵, 值)**函式，可以將資料寫進瀏覽器裡，setItem的第一個值是鍵的屬性名，第二個值就是相對應的值。

例子一：

```
let userID='u001';  
localStorage.setItem('userID',userID);
```

例子二 (用作儲存JSON)：

```
let userInfo={userID:'u001',userName:'Ada',lang:'tc',theme:'dark'};  
localStorage.setItem('userInfo',JSON.stringify(userInfo));
```

本機儲存 (2)

使用**localStorage.getItem(鍵名稱)**函式，取得瀏覽器Local Storage中所儲存的資料。

例子一：

```
let userID=localStorage.getItem('userID');  
if(userID!=null)console.log('The user ID is: '+userID);  
else console.log('Not found');
```

我們可透過localStorage.getItem來存取手機內置空間裡儲存著的資料。然後檢查它是否null值，並可因應存在與不存在要進行不同的程式。

例子二 (用作存取JSON)：

```
let userInfo=localStorage.getItem('userInfo');  
if(userInfo!=null){  
    userInfo=JSON.parse(userInfo);  
    console.log('The user ID is: '+userInfo.userID+', the user name is: '+userInfo.userName);  
}else console.log('Not found');
```

刪除資訊：

localStorage.clear() - 從網站程式中刪除所有已儲存並屬於它的資訊。

localStorage.removeItem(鍵) - 只刪除該鍵的資訊。

續課題三： JavaScript程式編寫及應用

條件判斷: For Each Loop

用途: 對應陣列變數續個進行處理的迴圈。

邏輯: `for(定義變數 於 陣列或鍵值對變數){
 做某些程式;
}`

例一: `續個迴圈(被抽中的細路 於 火星保衛隊){
 加冕派往參與保護火星大氣層的任務;
}`

實例: `const products=[
 {name:"鉛筆", price:3}, {name:"膠擦", price:2}
];
for(let product in products){
 console.log(product.name+' : HK$'+product.price+'
');
}`

條件判斷練習一

以下是一條陣列，名為「teachers」

```
teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},  
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},  
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4}]},  
];
```

Q1. 若要把teacher陣列中的物件逐個處理，該用哪種條件判斷式？

Q2. 把每個老師的名字(name)以 console.log 方式列出來。

Q3. For Loop 與 While Loop 有甚麼分別？

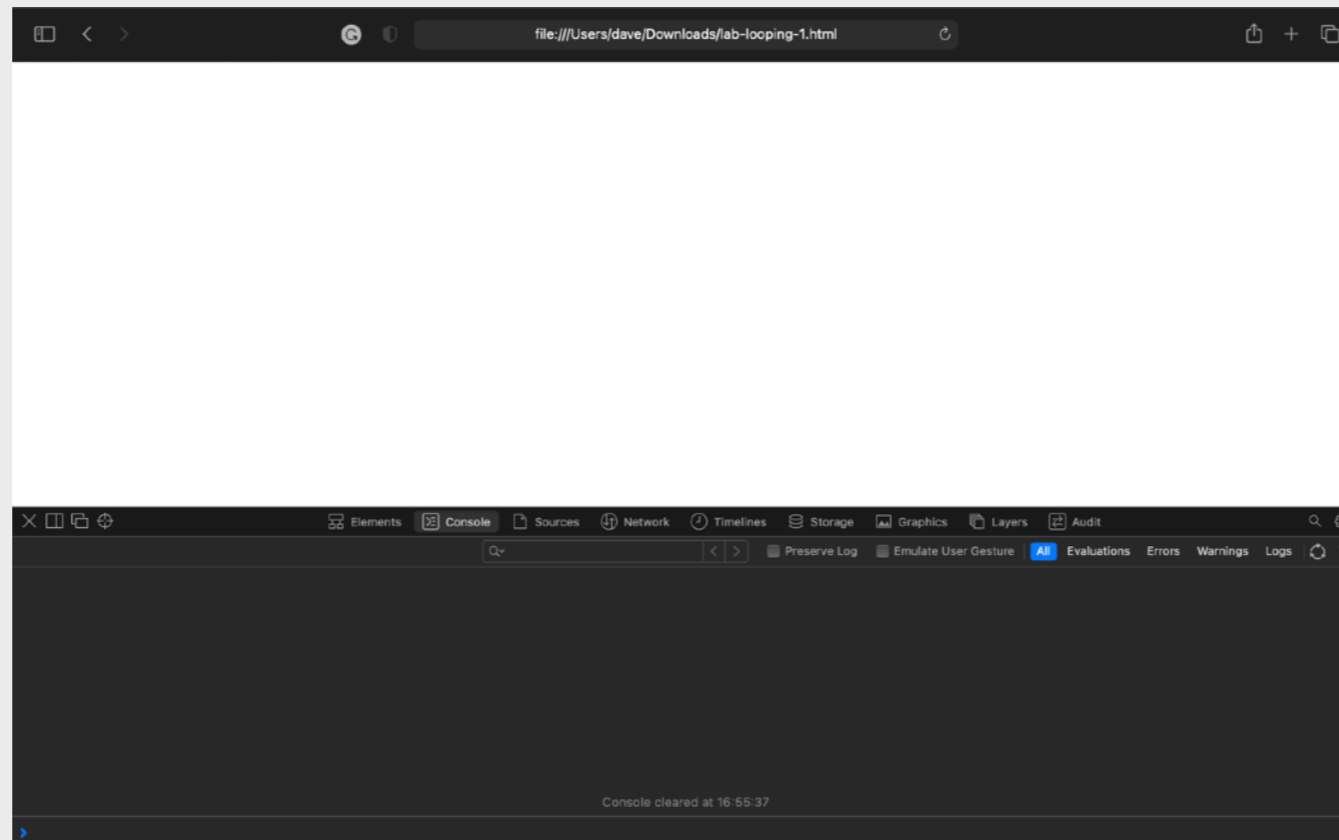
Q4. 把 id 大於 2 的老師名字列出來。

Q5. 運用 While Loop 把所有老師名字列出來。

Q6. (a) 把擁有BBA學歷的老師名字列出來。 (b) 把教授 Level 4 的老師名字列出來。

條件判斷練習一預備

```
lab-looping-1.html
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Test</title>
</head>
<body>
<script>
teachers=[
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4}]},
];
// Your code here...
</script>
</body>
</html>
```



製作一個HTML檔，加進基本的HTML標籤和 teachers 陣列，然後使用瀏覽器直接開啟，並打開除錯台(Console)觀看結果

條件判斷練習二

以下是一條鍵值配，名為「university」

```
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]}, schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}], presidentProfile:{"name":"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}}
```

Q1. 列出多於或等於 200 位職員的學院(school)的名字(name)

Q2. 列出B大樓(blkB)的座標，座標以逗號分隔

Q3. 使用 For Loop 把校長的學歷倒序列出來

Q4. 計出各學院員工人數的總和

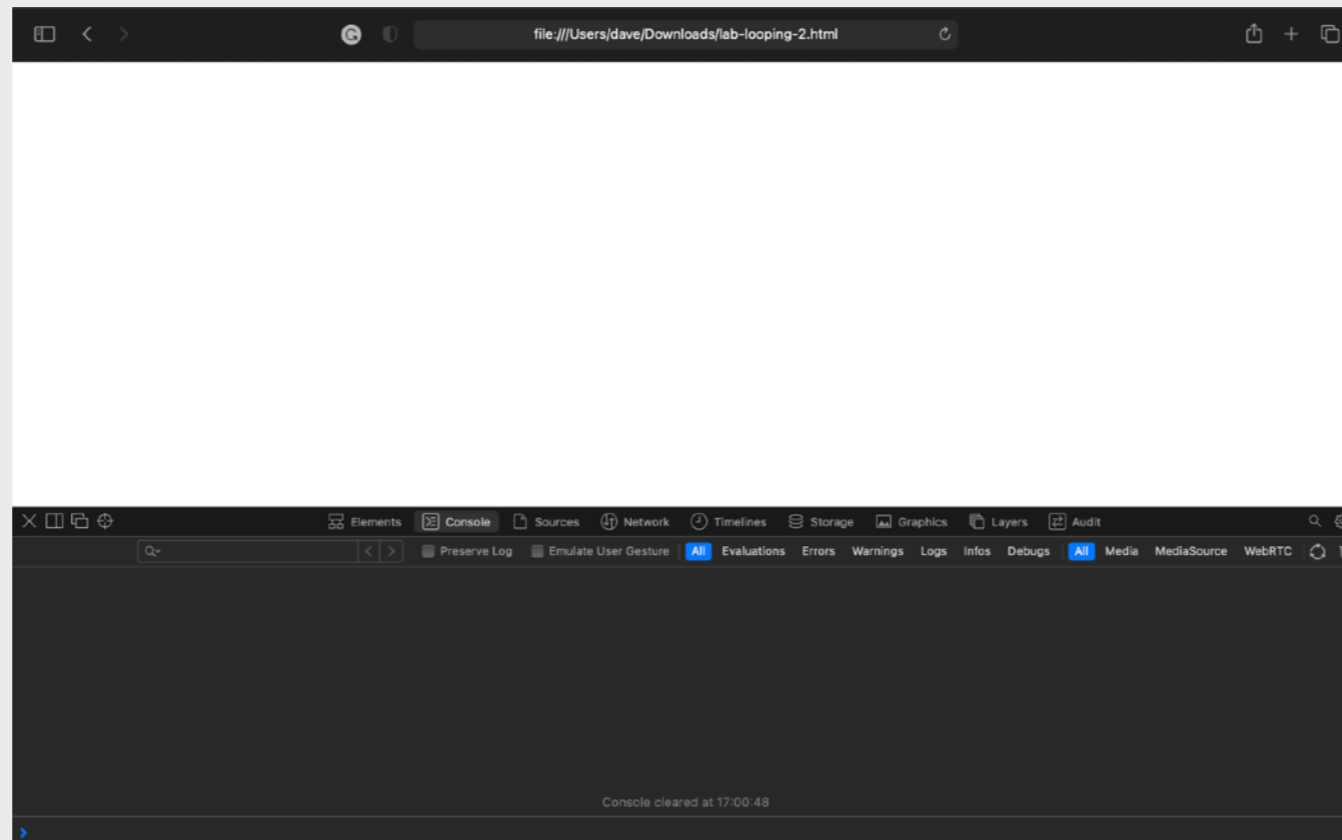
Q5 (挑戰題). 使用 charAt(0) 把校長首字元為「M」的學歷列出來

以上各題皆使用 console.log 列出便可

條件判斷練習二預備

```
lab-looping-2.html — Edited
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Test</title>
</head>
<body>
<script>
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]},
schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}],
presidentProfile:{"name":"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}
};

// Your code here...
</script>
</body>
</html>
```



製作一個HTML檔，加進基本的HTML標籤和 university 鍵值配，然後使用瀏覽器直接開啟，並打開除錯台(Console)觀看結果

JavaScript 計時器

JavaScript分兩種計時器，一種是setTimeout，另一種是setInterval。

setTimeout的意思就是延遲某毫秒後才執行某句或某些程式。例如：

```
window.onload=>{ setTimeout(=>{alert('Hello');}, 5000); }
```

這句很簡單，就是當window發生onload事件後，**等待5秒**，然後才執行alert顯示出Hello。

若果想在等待期間取消執行程式，則可先給setTimeout定義作變數，然後再透過clearTimeout來取消setTimeout：

```
let myTimeout=setTimeout(=>{alert('Hello');}, 5000);  
clearTimeout(myTimeout);
```

而setInterval的意思就是每某毫秒都執行某句或某些程式一次，直到Interval被clear。所以一定會先給setInterval定義作變數，然後再透過clearInterval來取消setInterval。例如：

```
window.onload=>{  
  let myInterval=setInterval(=>{console.log('interval');}, 2500);  
  setTimeout(=>{clearInterval(myInterval);}, 15000);  
}
```

這句很簡單，就是當window發生onload事件後，**每2.5秒**都每執行一次console.log('interval');，直到15秒後被clearInterval而取消執行。

JavaScript字串處理 - Length

length 字串長度

拿出字串的長度，即「幾多粒字」的意思。

用法：

```
string.length
```

範例：

```
var str="Hello";  
str.length; // 結果會是5。
```

JavaScript字串處理 - substr

substr() 提取字串中的文字
從字串中提取文字。

用法：

string.substr(始索引, 字串長度)

解釋：

string.substr(由第幾隻字開始, 幾多粒字)

範例：

```
var str="Hello World";
```

```
str.substr(2,3); // 結果會是「llo」。
```

```
str.substr(-3,3); // 結果會是「ld!」。
```

JavaScript字串處理 - indexOf

indexOf() 尋找字串

尋找字串中有沒有包含某段字串。

用法：

string.indexOf(搜尋值)

解釋：

string.indexOf(需要被尋找的字串)

範例：

```
var str="Hello World";
```

```
str.indexOf('ello'); // 結果會是1。
```

```
str.indexOf('W'); // 結果會是6。
```

```
str.indexOf('penguin'); // 因找不到，所以結果會是-1。
```

```
str.indexOf('o'); // 結果只會顯示第一個被找中的字串，即是4。
```

```
str.indexOf('w'); // 因大小寫為不同的字眼，所以找不到，即是-1。
```

JavaScript字串處理 - replace

replace() 字串取代
以新的字串取代舊的字串。

用法：

```
string.replace(新字串)
```

範例：

```
var str="Hello World";  
str.replace("World", "Holiday"); // 結果會是「Hello Holiday」。
```

JavaScript字串處理 - toLowerCase

toLowerCase() 小寫化
把字串轉換成全小寫。

用法：

```
string.toLowerCase();
```

範例：

```
var str="Hello World";  
str.toLowerCase(); // 結果會是「hello world」。
```

JavaScript字串處理 - toUpperCase

toUpperCase() 大寫化
把字串轉換成全大寫。

用法：

```
string.toUpperCase();
```

範例：

```
var str="Hello World";  
toUpperCase(); // 結果會是「HELLO WORLD」。
```


JavaScript字串處理 - split

split() 字串分拆

把字串分拆成多個字串的陣列。

用法：

```
string.split(分隔字串);
```

範例：

```
var str="I love penguin;I love koala;I love you";  
str.split(';');  
/*
```

結果會是：

```
I love penguin  
I love koala  
I love you  
*/
```

JavaScript字串處理 - trim

trim() 去除頭尾空白
把字串頭尾的空白字元去除。

用法：
string.trim();

範例：
var str=" Are you a genius? ";
str.trim(); // 結果會是「Are you a genius?」。

JavaScript字串處理 - toString

toString() 強制變成字串

把非字串類型的變數強制變成文字字串。

用法：

```
nonStringValue.toString();
```

範例：

```
var areYouRich=true;
```

```
areYouRich.toString(); // 結果會是「true」。
```

JavaScript Math 常用點數處理

Math.round() 四捨五入：

Math.round(3.14) // 3

Math.round(3.99) // 4

Math.round(-5.5) // -5

Math.floor() 無條件捨去：

Math.floor(3.14) // 3

Math.floor(8.99) // 8

Math.ceil() 無條件進位：

Math.ceil(3.14) // 4

Math.ceil(8.99) // 9

JavaScript Math 其他功能

<code>abs(x)</code>	傳回 x 的絕對值
<code>acos(x)</code>	傳回 x 的反餘弦值
<code>asin(x)</code>	傳回 x 的反正弦值
<code>atan(x)</code>	以介於 $-\pi/2$ 與 $\pi/2$ 弧度之間的數值來傳回 x 的反正切值
<code>atan2(y,x)</code>	傳回從 x 軸到點 (x,y) 的角度(介於 $-\pi/2$ 與 $\pi/2$ 弧度之間)
<code>cos(x)</code>	傳回數的餘弦
<code>exp(x)</code>	傳回 E^x 的指數
<code>log(x)</code>	傳回數的自然對數 (底為 e)
<code>max(x,y,z,...,n)</code>	傳回 $x,y,z,...,n$ 中的最高值
<code>min(x,y,z,...,n)</code>	傳回 $x,y,z,...,n$ 中的最低值
<code>pow(x,y)</code>	傳回 x 的 y 次冪
<code>random()</code>	傳回 $0 \sim 1$ 之間的隨機數
<code>sin(x)</code>	傳回數的正弦
<code>sqrt(x)</code>	傳回數的平方根
<code>tan(x)</code>	傳回角的正切

JavaScript Date 常用功能

<code>getDate()</code>	從 Date 對象傳回一個月中的某一天 (1 ~ 31)
<code>getDay()</code>	從 Date 對象傳回一周中的某一天 (0 ~ 6)
<code>getFullYear()</code>	從 Date 對象以四位數字傳回年份
<code>getHours()</code>	傳回 Date 對象的小時 (0 ~ 23)
<code>getMilliseconds()</code>	傳回 Date 對象的毫秒(0 ~ 999)
<code>getMinutes()</code>	傳回 Date 對象的分鐘 (0 ~ 59)
<code>getMonth()</code>	從 Date 對象傳回月份 (0 ~ 11)
<code>getSeconds()</code>	傳回 Date 對象的秒數 (0 ~ 59)
<code>toDateStrng()</code>	把 Date 對象的日期部分轉換為字符串
<code>toTimeString()</code>	把 Date 對象的時間部分轉換為字符串
<code>toJSON()</code>	以 JSON 數據格式傳回日期字符串

JavaScript Array 常用功能

<code>concat()</code>	合併多個陣列
<code>join()</code>	把數組的所有元素放入一個字符串
<code>isArray()</code>	判斷對像是否為陣列
<code>indexOf()</code>	搜索陣列中的元素，並傳回它所在的位置
<code>lastIndexOf()</code>	搜索陣列中的元素，並傳回它最後出現的位置
<code>pop()</code>	刪除陣列的最後一個元素
<code>push()</code>	向陣列的末尾添加一個或更多元素，並傳回新的長度
<code>unshift()</code>	向陣列的開頭添加一個或更多元素，並傳回新的長度
<code>reverse()</code>	反轉陣列的元素排列次序
<code>sort()</code>	對陣列的元素進行排序
<code>splice()</code>	從陣列中添加或刪除元素

JavaScript Document 常用參數及功能 – 1

(document.element)

addEventListener()	向指定元素添加事件監聽
appendChild()	為元素添加一個新的子元素
attributes	傳回一個元素的屬性陣列
childNodes	傳回元素的一個子節點的陣列
className	設置或傳回元素的class屬性
clientTop	表示一個元素的頂部邊框的寬度(像素)
clientLeft	表示一個元素的左邊框的寬度(像素)
clientHeight	在頁面上傳回內容的可視高度(像素)
clientWidth	在頁面上傳回內容的可視寬度(像素)
cloneNode()	克隆某個元素
firstChild	傳回元素的第一個子節點
focus()	設置文檔或元素獲取焦點
getAttribute()	傳回指定元素的屬性值

JavaScript Document 常用參數及功能 – 2

(document.element)

getElementsByTagName()	取得所有該類元素的元素實體
getElementsByClassName()	取得所有該類類別的元素實體
hasAttribute()	檢查元素是否包含了刻屬性
id	設置或者傳回元素的id
innerHTML	設置或者傳回元素的內容
insertBefore()	現有的子元素之前插入一個新的子元素
offsetHeight	元素的高度但不包含外邊距(margin)
offsetWidth	元素的寬度但不包含外邊距(margin)
offsetLeft	當前元素的相對水平位置(像素)
offsetTop	當前元素的相對垂直位置(像素)
parentNode	元素的父元素
querySelector()	傳回指定CSS選擇器元素的首個子元素
removeAttribute()	從元素中刪除指定的屬性
removeChild()	刪除一個子元素
removeEventListener()	移除事件監聽

JavaScript Document 常用參數及功能 – 3

(document.element)

replaceChild()	替換一個子元素
scrollHeight	傳回整個元素的高度(包括含滾動條的隱蔽的地方)
scrollLeft	傳回整個元素的闊度(包括含滾動條的隱蔽的地方)
scrollTop	傳回當前滾動的Y軸座標
scrollWidth	傳回當前滾動的X軸座標
setAttribute()	設置指定屬性並指定值
tagName	作為一個字符串傳回某個元素的標記名(大寫)

JavaScript Window 常用功能

alert()	顯示帶有一段消息和一個確認按鈕的警告框
atob()	解碼一個base-64編碼的字符串
btoa()	創建一個base-64編碼的字符串
blur()	把鍵盤焦點從頂層窗口移開
setInterval()	指定的周期(以毫秒計)來調用函數或計算表達式
setTimeout()	在指定的毫秒數後調用函數或計算表達式
clearInterval()	取消由setInterval()設置的timeout
clearTimeout()	取消由setTimeout()方法設置的timeout
close()	關閉瀏覽器窗口
confirm()	顯示帶有一段消息以及確認按鈕和取消按鈕的對話框
open()	打開一個新的瀏覽器窗口或查找一個已命名的窗口
prompt()	顯示可提示用戶輸入的對話框
scrollTo()	把內容滾動到指定的坐標
postMessage()	安全地實現跨源通信

JavaScript Navigator 常用參數

appName	傳回瀏覽器的代碼名
appVersion	傳回瀏覽器的名稱
cookieEnabled	傳回瀏覽器的平台和版本信息
platform	傳回指明瀏覽器中是否啟用cookie的布林值
userAgent	傳回運行瀏覽器的操作系統平台
	傳回由客戶機發送服務器的user-agent頭部的值

JavaScript History 常用功能

<code>back()</code>	加載history列表中的前一個URL
<code>forward()</code>	加載history列表中的下一個URL
<code>go()</code>	加載history列表中的某個具體頁面

JavaScript Location 常用參數

hash	傳回一個URL的錨部分
host	傳回一個URL的主機名和端口
hostname	傳回URL的主機名
href	傳回完整的URL
pathname	傳回的URL路徑名
port	傳回一個URL服務器使用的端口號
protocol	傳回一個URL協議
search	傳回一個URL的查詢部分

JavaScript Location 常用參數及功能

hash	傳回一個URL的錨部分
host	傳回一個URL的主機名和端口
hostname	傳回URL的主機名
href	傳回完整的URL
pathname	傳回的URL路徑名
port	傳回一個URL服務器使用的端口號
protocol	傳回一個URL協議
search	傳回一個URL的查詢部分
reload()	重新載入當前文檔
replace()	用新的文檔替換當前文檔

JavaScript 全局功能

`decodeURI()`

解碼某個編碼的URI

`decodeURIComponent()`

解碼一個編碼的URI組件

`encodeURIComponent()`

把字串編碼為URI

`encodeURIComponent()`

把字串編碼為URI組件

`escape()`

對字串進行編碼

`eval()`

把字串作為腳本代碼來執行

`isFinite()`

檢查某個值是否為有窮大的數

`isNaN()`

檢查某個值是否是數字

`Number()`

把對象的值轉換為數字

`parseFloat()`

解析一個字串並返回一個浮點數

`parseInt()`

解析一個字串並返回一個整數

`String()`

把值轉換為字串

`unescape()`

對由`escape()`編碼的字串進行解碼

JavaScript錯誤處理 - 1

用來處理例外錯誤，以免程式因發生錯誤而停止執行(「當機」)

當發生錯誤時，以更佳的方法來調解問題(「執手尾」)

JavaScript 會有這些類型的錯誤：

錯誤類型	說明
EvalError	eval() 執行錯誤
RangeError	一個數值超過允許範圍
ReferenceError	存取未宣告的變數
SyntaxError	程式語法錯誤
TypeError	型別錯誤
URIError	URI handling 相關函數例如 decodeURIComponent() 執行時錯誤

JavaScript錯誤處理 - 2

程式解說：

```
try {  
    // 預期可能會發生錯誤的程式碼  
} catch (e) {  
    // try 區塊有拋出錯誤時，則執行這裡的程式碼  
} finally {  
    // finally 區塊的程式碼一定會在最後被執行  
    // 你可以省略 finally 區塊  
}
```

程式例子：

```
try {  
    echo('hello');  
} catch (e) {  
    alert(e.name + ': ' + e.message);  
} finally {  
    alert('try catch 區塊結束');  
}
```

HTML和Windows物件和屬性

alert	all	anchor	anchors	area
assign	blur	button	checkbox	clearInterval
clearTimeout	clientInformation	close	closed	confirm
constructor	crypto	decodeURI	decodeURIComponent	defaultStatus
document	element	elements	embed	embeds
encodeURIComponent	encodeURIComponent	escape	event	fileUpload
focus	form	forms	frame	innerHeight
innerWidth	layer	layers	link	location
mimeType	navigate	navigator	frames	frameRate
hidden	history	image	images	offscreenBuffering
open	opener	option	outerHeight	outerWidth
packages	pageXOffset	pageYOffset	parent	parseFloat
parseInt	password	pkcs11	plugin	prompt
propertyIsEnum	radio	reset	screenX	screenY
scroll	secure	select	self	setInterval
setTimeout	status	submit	taint	text
textarea	top	unescape	untaint	window

HTML事件控制代碼

onblur	onclick	onerror	onfocus
onkeydown	onkeypress	onkeyup	onmouseover
onload	onmouseup	onmousedown	onsubmit

JavaScript Promise

非同步方式運行程式

```
const myPromise = new Promise((resolve, reject) => {
  if ((Math.random() * 100) < 90) {
    resolve('Resolved');
  }
  reject(new Error('Rejected'));
});

// 方法一：先定義後指派處理Resolve及Reject的功能
const onResolved = (resolvedValue)=>console.log(resolvedValue);
const onRejected = (error)=>console.log(error);
myPromise.then(onResolved, onRejected);

// 方法二：簡潔地直接指派處理Resolve及Reject的無名功能
myPromise.then((resolvedValue) => {
  console.log(resolvedValue);
}, (error) => {
  console.log(error);
});
```

JavaScript Ajax

Ajax(Asynchronous JavaScript and XML)，是指客戶端以JavaScript透過非同步方式向伺服器端發出請求(XMLHttpRequest)並取得回應來把客戶端進行局部的更新。

傳統方法：

```
function createAjax(_link, _method, _data, _endFunction) {
    const formData = new FormData();
    for (let i in _data) formData.append(i, _data[i]);
    const xhr = new XMLHttpRequest();
    xhr.open(_method, _link, true);
    xhr.onload = (e)=>{};
    xhr.onreadystatechange = ()=>{
        if ((xhr.readyState == 4) && (xhr.status == 200)) {
            try {
                if (_endFunction != null) {
                    _endFunction(eval('(' + xhr.responseText + ')'));
                }
            } catch (e) {}
        }
    };
    xhr.send(formData);
}

createAjax('ser.php', 'POST', {service:"test"}, res=>{
    console.log(res);
});
```

JavaScript ES6版本新方法：

```
async function createAjax(url, method, data){
    const response = await fetch(url, {
        method, headers: {
            'Accept': 'application/json',
            'Content-Type': 'application/x-www-
form-urlencoded;charset=UTF-8',
        }, body:new URLSearchParams(data)
    });
    return response.ok ? response.json():null;
}

createAjax('ser.php', 'POST', {service:"test"})
.then(res=>{
    console.log(res);
});
```

JavaScript的觸發事件

當使用者瀏覽網頁或JavaScript應用程式時，其實會觸發很多JavaScript的event(事件)，例如按下按鈕、在輸入欄位中打字等。而我們則可利用JavaScript來監測和處理事件的發生。以下是一些常用的事件：

事件名稱	觸發原因
click	當元素被點擊時
change	當Input(輸入)元素的值有所改變
touchstart	當手指觸到元素時
touchend	當手指離開所觸到的元素時
focus	當Input元素被點擊或取得焦點時
load	當元素載入完成時
keyup	當按下並放開鍵盤按鍵時
resize	當元素被改變大小時
scroll	當元素被捲動時
select	當文字被選取時
submit	當Form(表單)被送出時
beforeunload	當使用者關閉(或離開)網頁之前
unload	當使用者關閉(或離開)網頁之後
error	當元素發生錯誤時

JavaScript監測和處理事件 - 1

方法一：HTML Inline Attribute

```
<body>
<button onclick="buttonClicked();">Click me</button>
<script>
function buttonClicked(){
alert('Button clicked!');
}
</script>
</body>
```


JavaScript監測和處理事件 - 2

方法二：DOM Object Property

```
<body>
<button id="button_1">Click me</button>
<script>
function buttonClicked(){
alert('Button clicked!');
}
document.getElementById('button_1').onclick=buttonClicked;
</script>
</body>
```

或

```
<body>
<button id="button_1">Click me</button>
<script>
document.getElementById('button_1').onclick={()=>{
alert('Button clicked!');
}
}
</script>
</body>
```

如若移除事件處理函式，可使用「`ele.onclick=null;`」，當中的「`ele`」是指被處理的HTML元素如上述例子中的「`document.getElementById('button_1').onclick=null;`」。

JavaScript 監測和處理事件 - 3

方法三：EventListener

```
<body>
<button id="button_1">Click me</button>
<script>
document.getElementById('button_1').addEventListener('click', ()=>{
alert('The first call. ');
});
document.getElementById('button_1').addEventListener('click', ()=>{
console.log('The second call. ');
});
</script>
</body>
```

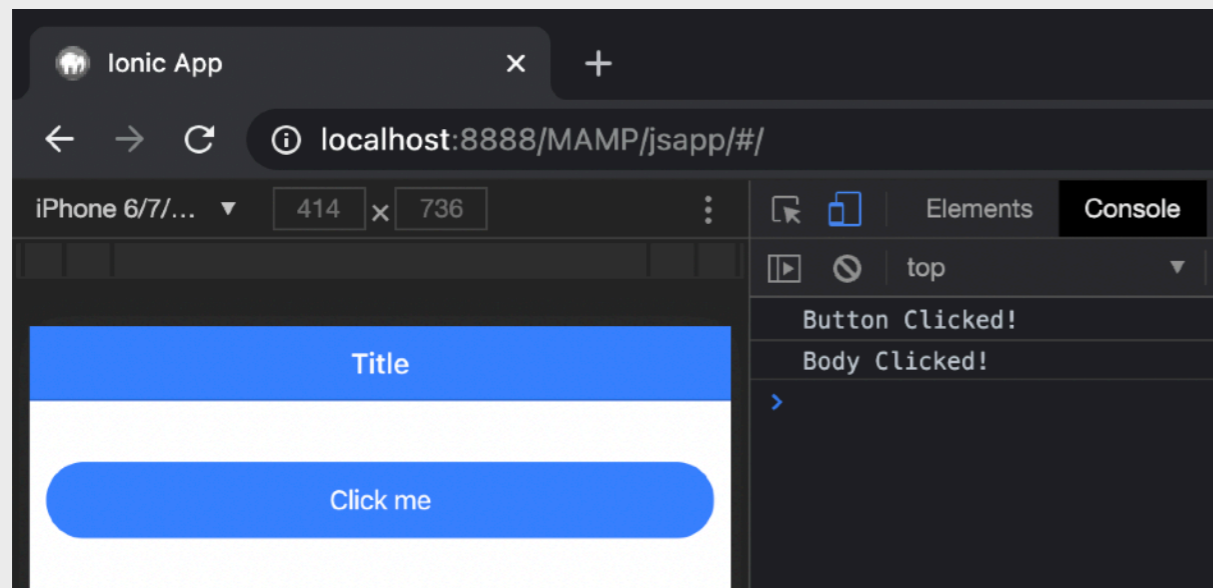
如若移除事件處理函數，可使用「removeEventListener」。

JavaScript 事件發生次序 (事件氣泡與事件捕捉) - 1

先看這段HTML及JavaScript程式碼：

```
<body>
<button id="button_1">Click me</button>
<script>
document.getElementById('button_1').addEventListener('click', ()=>{
console.log('Button Clicked!');
});
document.body.addEventListener('click', ()=>{
console.log('Body Clicked!');
});
</script>
</body>
```

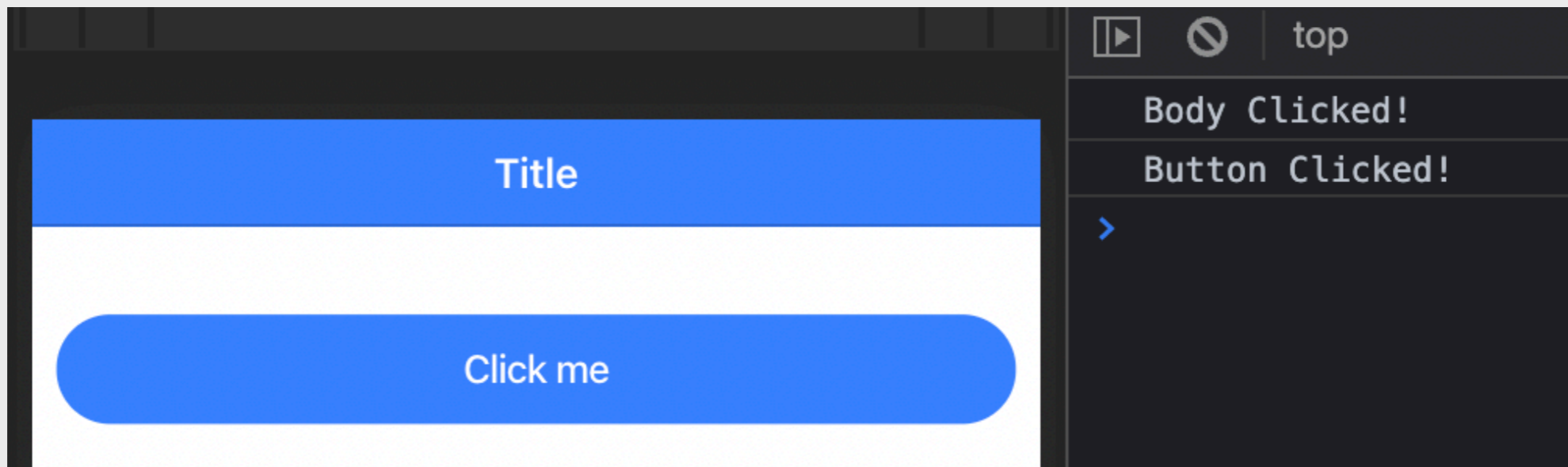
輸出結果：



JavaScript 事件發生次序 (事件氣泡與事件捕捉) - 2

在「addEventListener」和「removeEventListener」方法中，有第三個參數布林值「useCapture」用來指定事件發生的先後次序，若不輸入第三個參數，則自動預設為「false」，而「false」即是由子元素開始觸發事件，若「true」即是由父元素開始觸發事件，若我們希望把上述例子改成為先觸發「body」發生事件，則可把「document.body.addEventListener」這一段改成為：

```
document.body.addEventListener('click', ()=>{console.log('Body Clicked!');}, true);
```



即先會出現「Body Clicked!」然後再出現「Button Clicked!」。

JavaScript 事件物件

當事件被觸發時，事件並會攜帶著一個事件物件(Event Object)給予所處理事件的函式。而事件物件有試常用的屬性(property)：

屬性	說明
type	所發生的事件類型，如「click」
target	所被觸發事件的元素

event.stopPropagation()

當了解到「事件發生次序」後，此函式就是用來停止觸發後續的事件以執行事件處理函式。

event.preventDefault()

此函式是用來取消瀏覽器預設的行為。

教學參考

<https://www.fooish.com/javascript/dom/event.html>

JavaScript 欄位驗證 - 1

透過JavaScript，我們可以對input元素的數值進行驗證

例如當給予用戶填表單(form)時，
我們可以使用JavaScript配合HTML來驗證用戶有否出現格式上的錯誤

首先我們在HTML裡的input或textarea元素加上正確的屬性，例如只接受輸數字，type屬性便設定成number，若還要增加範圍限制，
則需設定其max及min值
當完成HTML元素設定後，便可使用JavaScript來進行驗證

JavaScript 欄位驗證 - 2

檢查必填input(欄位)

```
<body>
  <input type="text" id="input_name"><br>
  <input type="tel" id="input_tel"><br>
  <input type="email" id="input_email"><br>
  <button onclick="submitInputs();"></button>
</body>
<script>
function validateInputs(){
  const name=document.getElementById('input_name').value;
  const tel=document.getElementById('input_tel').value;
  const email=document.getElementById('input_email').value;
  if(name==null || name=="")return false;
  if(tel==null || tel=="")return false;
  if(email==null || email=="")return false;
  return true;
}
function submitInputs(){
  alert(validateInputs() ? '成功':'必須填寫所有欄位');
}
</script>
```

JavaScript 欄位驗證 - 3

使用checkValidity()方法來驗證格式有否錯誤

```
<body>
<input id="input_num" type="number" min="100" max="300" required>
<button onclick="validateInput();">驗證</button>
<p>如果輸入的數字小於100或大於300，會提示錯誤信息。</p>
<script>
function validateInput() {
    const num=document.getElementById('input_num');
    if(!num.checkValidity())alert(num.validationMessage);
    else alert('輸入正確');
}
</script>
</body>
```

詳情可參閱：<https://www.runoob.com/js/js-validation-api.html>

JavaScript Web Worker 多工處理 - 1

單工處理(單執行緒)是指同一時間只做同一件事，
多工處理(多執行緒)是指同一時間只做多於一件事

JavaScript以往都是單工處理，但隨著Web Worker的出現，
JavaScript則可以進行多工處理

在主工緒運行的同時，Web Worker工緒在後台運行，兩者互不干擾。

Web Worker較耗費資源，不應該過度使用，
而且一旦使用完畢，就應該關閉

詳情可參閱：<https://www.ruanyifeng.com/blog/2018/07/web-worker.html>

JavaScript Web Worker 多工處理 - 2

先創建worker.js為Web Worker的主邏輯程式檔案

worker.js

```
var i = 0;
function timedCount() {
  i = i + 1;
  postMessage(i);
  setTimeout("timedCount()",1000);
}
timedCount();
```

詳情可參閱：https://www.w3schools.com/html/html5_webworkers.asp

JavaScript Web Worker 多工處理 - 3

在普通的HTML中加上並運行Web Worker

worker.html

```
<!DOCTYPE html>
<html>
<body>
<p>運行秒數: <output id="result"></output></p>
<button onclick="startWorker()">啟動Worker</button>
<button onclick="stopWorker()">結束Worker</button>
<script>
var w;
function startWorker() {
  if(typeof(Worker)!=="undefined"){
    if(typeof(w)==="undefined")w=new Worker("workers.js");
    w.onmessage=event=>document.getElementById("result").innerHTML=event.data;
  }else document.getElementById("result").innerHTML="瀏覽器不支援Web Worker";
}
function stopWorker() {
  w.terminate();
  w=undefined;
}
</script>
</body>
</html>
```

課題五：

Ionic簡介及UI製作

Ionic簡介

Ionic是一個知名的流動應用程式開發框架，用於開發iOS及Android的應用程式或網路應用程式(Web App)，只需開發一個版本(一套程式碼)，即可使用於iOS及Android裝置上，不需為iOS及Android分別開發各自的程式碼，十分方便與省時。而使用Ionic作開發，我們通常也需要搭載其他的開發框架同時使用，而Ionic則支援React、Angular與Vue，當然也可以使用純JavaScript作開發。

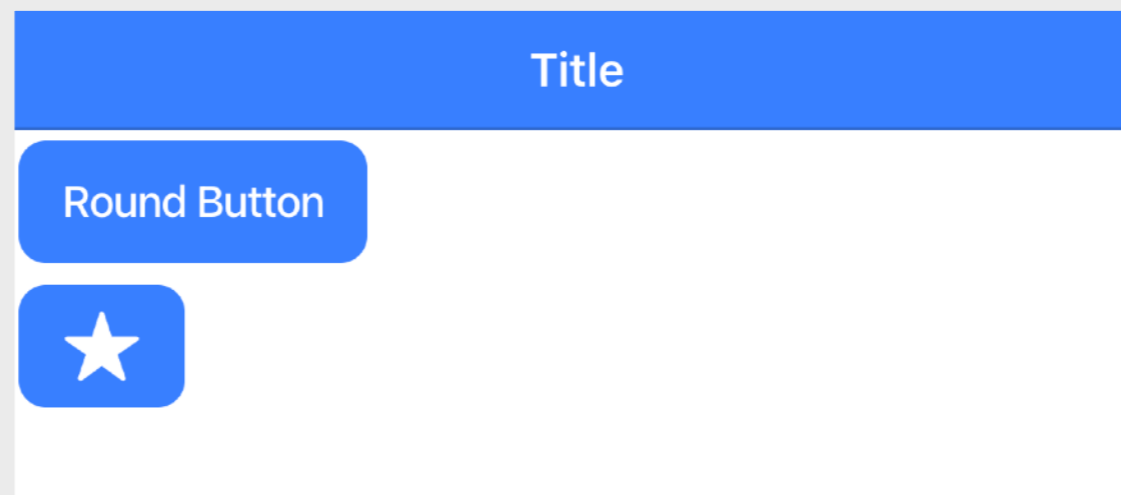
Ionic UI Components簡介與實作

Ionic提供多款的使用者界面元件(UI Components)及相關應用程式介面以供開發者製作出與iOS及Android相符的界面，開發者可參考Ionic的官網(<https://ionicframework.com/docs/components>)以使用其界面元件。以下是Ionic所擁有的界面元件：

Action Sheet	Alert	Badge	Button
Card	Checkbox	Chip	Content
Date & Time Pickers	Floating Action Button	Grid	Infinite Scroll
Icons	Input	Item	List
Menu	Modal	Navigation	Popover
Progress Indicators	Radio	Refresher	Reorder
Routing	Searchbar	Segment	Select
Slides	Tabs	Toast	Toggle
Toolbar	Avatar	Img	Thumbnail

ion-button

```
<ion-button shape="round">Round Button</ion-button>  
<br>  
<ion-button>  
  <ion-icon slot="icon-only" name="star"></ion-icon>  
</ion-button>
```



ion-card

```
<ion-card>
  
  <ion-card-header>
    <ion-card-subtitle>2021年5月</ion-card-
subtitle>
    <ion-card-title>APP開發比賽</ion-card-
title>
  </ion-card-header>
  <ion-card-content>
    比賽順利完成，並舉行了頒獎典禮，由聯合創辦人Dr.
Phone主持及頒獎，感謝三位冠亞季軍抽空參與。
  </ion-card-content>
</ion-card>
```



Ion-list, ion-item, ion-label, ion-checkbox

```
<ion-list>
  <ion-item>
    <ion-label>Default</ion-label>
    <ion-checkbox></ion-checkbox>
  </ion-item>
  <ion-item>
    <ion-label>Disabled</ion-label>
    <ion-checkbox disabled="true"></ion-checkbox>
  </ion-item>
  <ion-item>
    <ion-label>Checked</ion-label>
    <ion-checkbox checked="true"></ion-checkbox>
  </ion-item>
</ion-list>
```

Title	
Default	<input type="checkbox"/>
Disabled	<input type="checkbox"/>
Checked	<input checked="" type="checkbox"/>

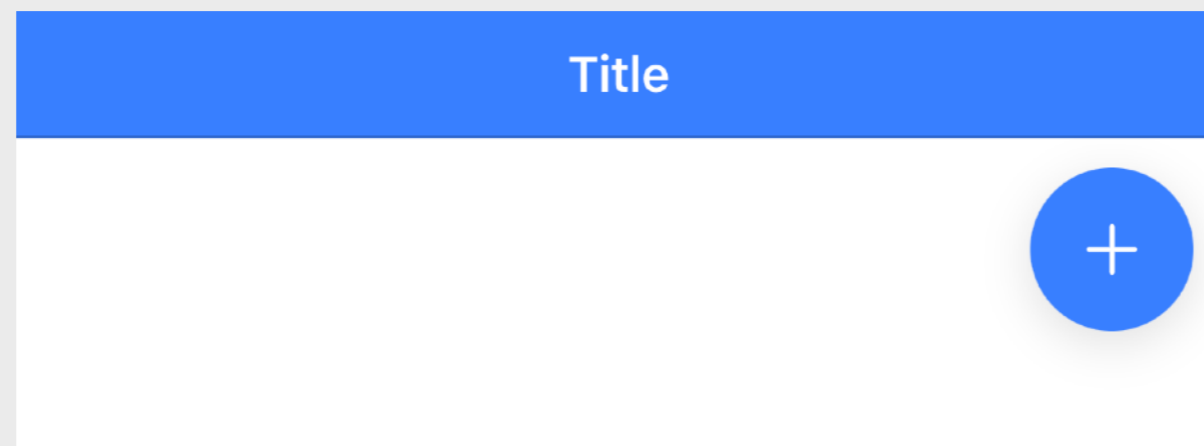
ion-datetime

```
<ion-list>
  <ion-item>
    <ion-label>MM DD YY</ion-label>
    <ion-datetime displayFormat="YY MM DD"
placeholder="Select
  Date"></ion-datetime>
  </ion-item>
</ion-list>
```

Title			
MM DD YY	Select Date		
Cancel	Done		
Feb	7		
Mar	8		
Apr	9		
May	10	2021	
Jun	11	2020	
Jul	12	2019	
Aug	13	2018	

ion-fab

```
<ion-fab vertical="top"  
horizontal="end" slot="fixed">  
  <ion-fab-button>  
    <ion-icon name="add"></ion-icon>  
  </ion-fab-button>  
</ion-fab>
```



ion-input

```
<ion-list>
  <ion-item>
    <ion-label
position="stacked">Input Sample</ion-
label>
    <ion-input placeholder="Type
something"></ion-input>
  </ion-item>
</ion-list>
```

Title

Input Sample

Type something

ion-segment

```
<ion-segment style="width:50%;margin:0 auto;">  
  <ion-segment-button value="ios">  
    <ion-label>iOS</ion-label>  
  </ion-segment-button>  
  <ion-segment-button value="android">  
    <ion-label>Android</ion-label>  
  </ion-segment-button>  
</ion-segment>
```

Title

iOS

Android

ion-radio

```
<ion-list>
  <ion-radio-group value="html">
    <ion-list-header>
      <ion-label>Which is not a programming language?</ion-label>
    </ion-list-header>
    <ion-item>
      <ion-label>TypeScript</ion-label>
      <ion-radio slot="start" value="ts"></ion-radio>
    </ion-item>
    <ion-item>
      <ion-label>Swift</ion-label>
      <ion-radio slot="start" value="swift"></ion-radio>
    </ion-item>
    <ion-item>
      <ion-label>HTML</ion-label>
      <ion-radio slot="start" value="html"></ion-radio>
    </ion-item>
    <ion-item>
      <ion-label>Objective-C</ion-label>
      <ion-radio slot="start" value="objc"></ion-radio>
    </ion-item>
    <ion-item>
      <ion-label>Erlang</ion-label>
      <ion-radio slot="start" value="erlang"></ion-radio>
    </ion-item>
  </ion-radio-group>
</ion-list>
```

Title
Which is not a programming language?
TypeScript
Swift
✓ HTML
Objective-C
Erlang

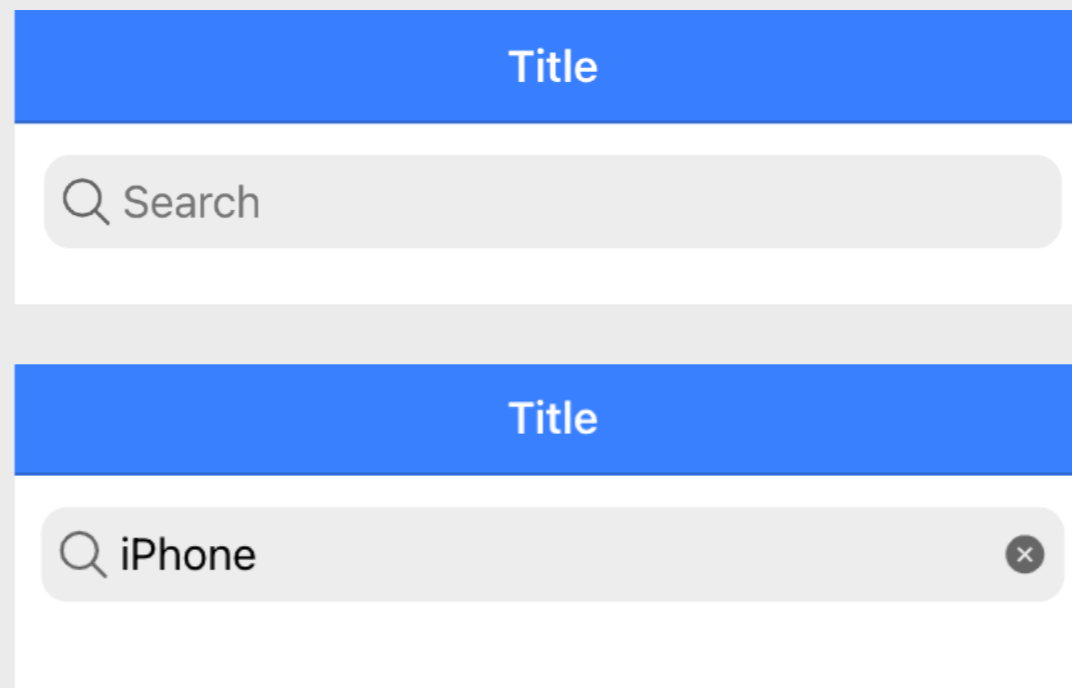
Ion-range

```
<ion-list>
  <ion-item>
    <ion-range color="danger" pin="true"></ion-range>
  </ion-item>
  <ion-item>
    <ion-range min="-200" max="200">
      <ion-label slot="start">-200</ion-label>
      <ion-label slot="end">200</ion-label>
    </ion-range>
  </ion-item>
  <ion-item>
    <ion-range min="20" max="80" step="2">
      <ion-icon size="small" slot="start" name="remove">
      </ion-icon>
      <ion-icon slot="end" name="add"></ion-icon>
    </ion-range>
  </ion-item>
</ion-list>
```

Title	
<input type="range"/>	
-200	200
-	+

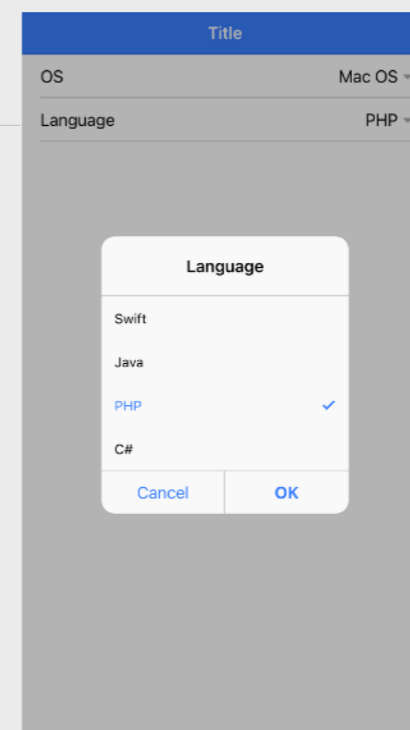
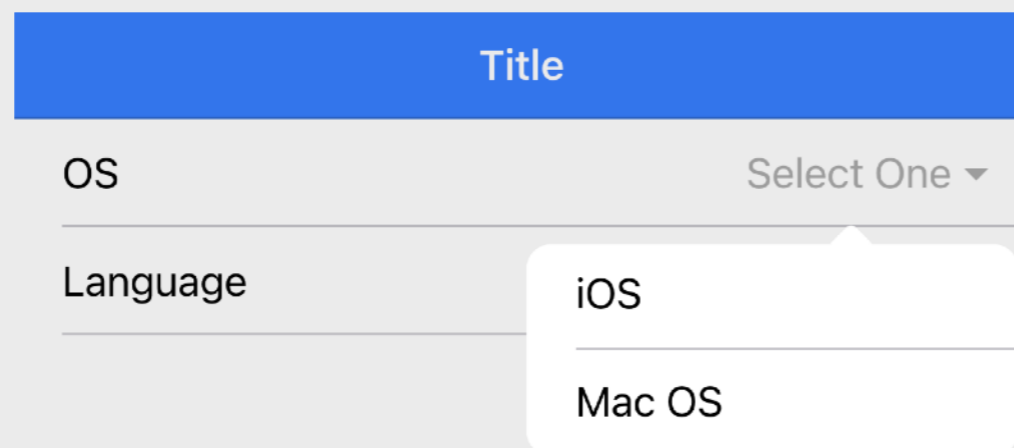
ion-searchbar

```
<ion-searchbar showCancelButton="always"  
placeholder="Search"></ion-searchbar>
```



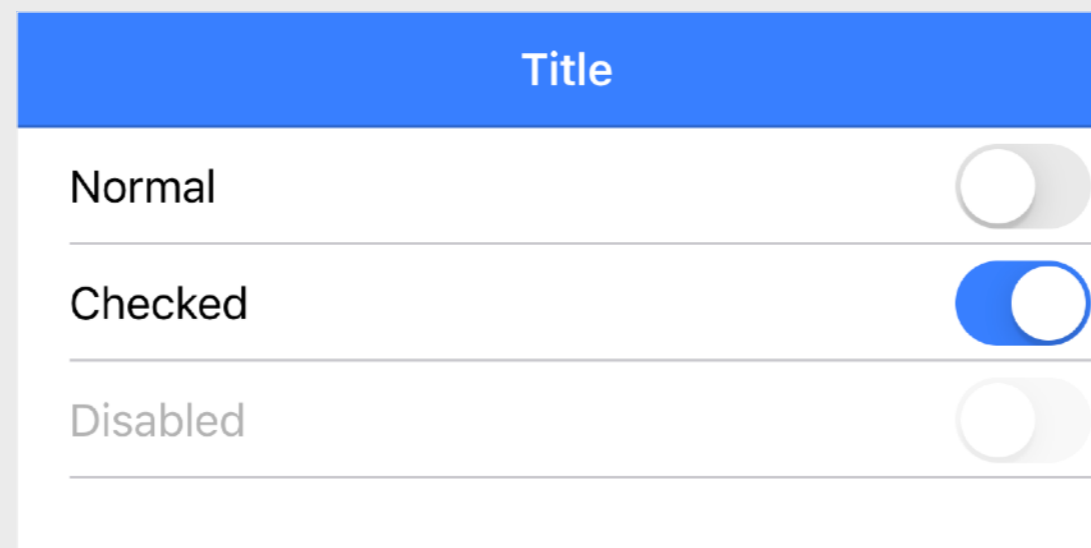
ion-select

```
<ion-item>
  <ion-label>OS</ion-label>
  <ion-select interface="popover" placeholder="Select One">
    <ion-select-option value="ios">iOS</ion-select-option>
    <ion-select-option value="macos">Mac OS</ion-select-option>
  </ion-select>
</ion-item>
<ion-item>
  <ion-label>Language</ion-label>
  <ion-select value="php" okText="Okay" cancelText="Dismiss">
    <ion-select-option value="swift">Swift</ion-select-option>
    <ion-select-option value="java">Java</ion-select-option>
    <ion-select-option value="php">PHP</ion-select-option>
    <ion-select-option value="csharp">C#</ion-select-option>
  </ion-select>
</ion-item>
```

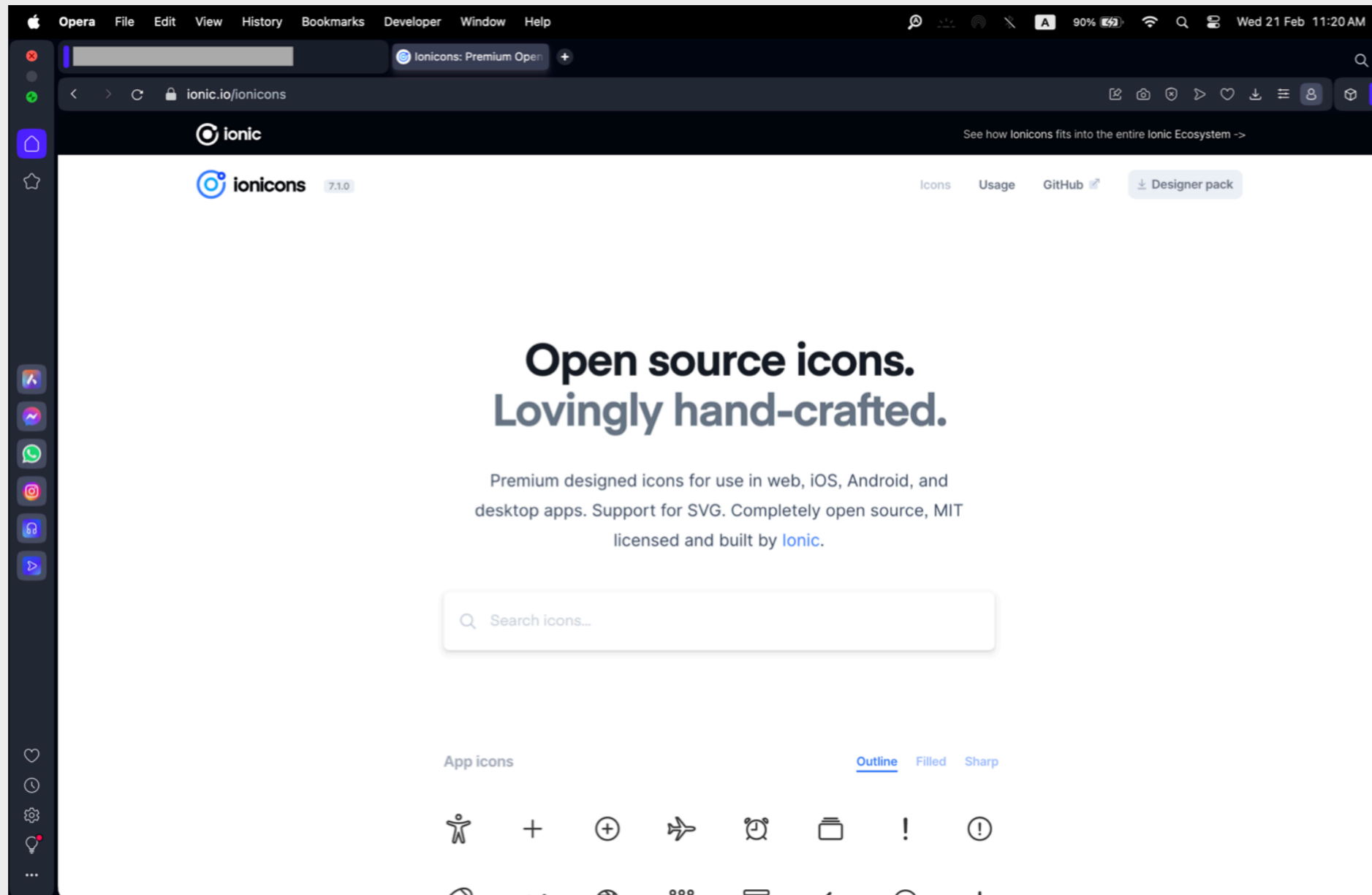


ion-toggle

```
<ion-list>  
  <ion-item>  
    <ion-label>Normal</ion-label>  
    <ion-toggle></ion-toggle>  
  </ion-item>  
  <ion-item>  
    <ion-label>Checked</ion-label>  
    <ion-toggle checked></ion-toggle>  
  </ion-item>  
  <ion-item>  
    <ion-label>Disabled</ion-label>  
    <ion-toggle disabled></ion-toggle>  
  </ion-item>  
</ion-list>
```



ion-icon (1)



<https://ionic.io/ionicons>

在Ionic Framework中，ion-icon 是一個用來顯示圖標的組件。Ionic 提供了很多內建的圖標，您可以很容易地在您的應用程式中使用它們。

ion-icon (2)

在Ionic Framework中，`ion-icon` 是一個用來顯示圖標的組件。Ionic 提供了很多內建的圖標，您可以很容易地在您的應用程式中使用它們。以下是一些基本的使用方法：

1. 引入圖標：

您可以直接在HTML文件中使用 `ion-icon` 標籤來引入圖標。Ionic 的圖標使用了 SVG 格式。

2. 指定圖標名稱：

使用 `name` 屬性來指定您想要顯示的圖標。例如，如果您想顯示一個“home”圖標，您可以這樣寫：

```
<ion-icon name="home"></ion-icon>
```

3. 設定大小：

您可以使用 `size` 屬性來設定圖標的大小。典型的值有 `small`、`large` 或者不設定來使用預設大小。

```
<ion-icon name="home" size="large"></ion-icon>
```

ion-icon (3)

4. 圖標顏色：

色彩可以通過CSS來控制，或者使用 `color` 屬性指定內建的顏色。

```
<ion-icon name="home" color="primary"></ion-icon>
```

5. 其他屬性：

`ion-icon` 還有其他屬性，比如 `src` 用於指定自定義 SVG 文件的路徑，或 `ios` 和 `md` 屬性用於指定不同平台下使用的圖標。

```
<ion-icon src="/path/to/external/file.svg"></ion-icon>
```

或者

```
<ion-icon ios="ios-add" md="md-add"></ion-icon>
```

在使用 `ion-icon` 時，確保您已經安裝了 Ionic 的圖標庫，或者透過 CDN 引入相關的 CSS 檔案。這樣，當您的應用加載時，圖標就會正確顯示。

Ionic實習1A

嘗試到 <https://s3objectstorage-a.ap-south-1.linodeobjects.com/ionic-labs-1.zip> 下載並解壓檔案，然後把資料夾「empty」複製並命名為「lab1」，然後嘗試在「index.html」內加入不同的元素如ion-button、ion-card、ion-list、ion-item、ion-input……等。最後使用瀏覽器並開啓「手機模式」連結到本地(本機)伺服器以「localhost」來開啓「lab1」。

Windows連結例子: <http://localhost/ionic-labs-1/lab1>

Mac連結例子: <http://localhost:8888/MAMP/ionic-labs-1/lab1>

Ionic實習1B

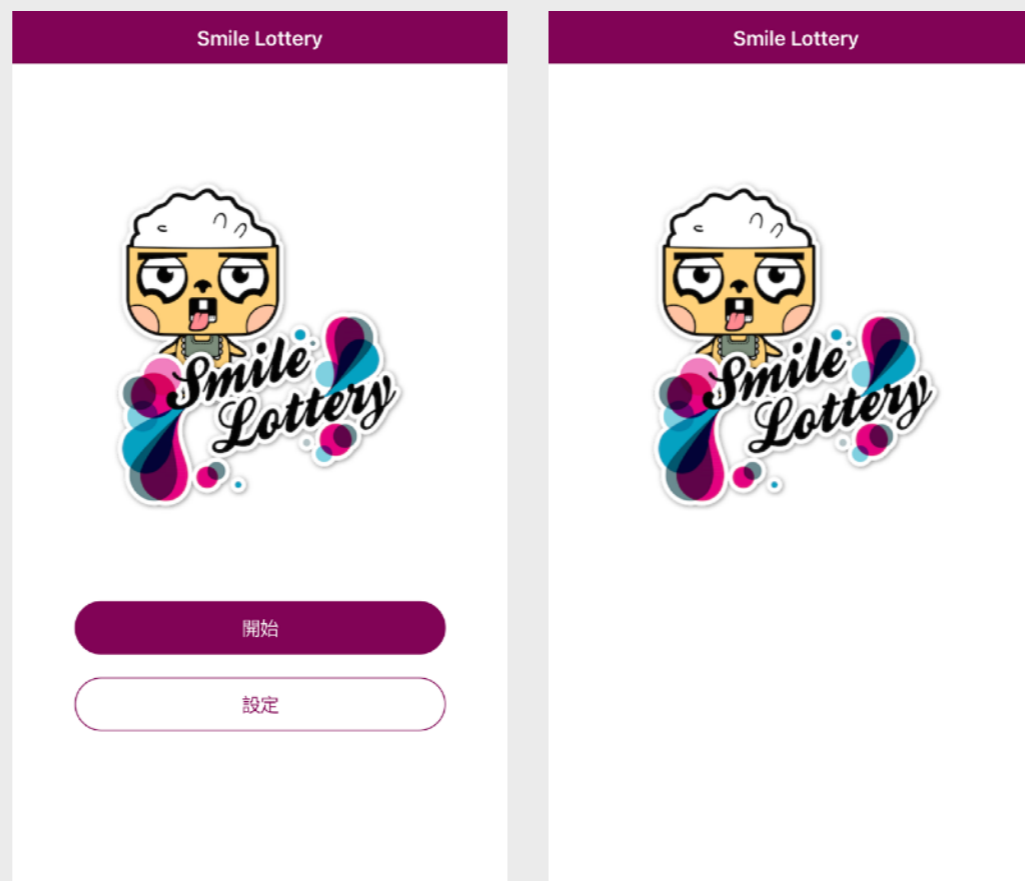
瀏覽「smilelottery」小遊戲，這是一個完整的製成品。試打開「index_lab.html」，這是一個有錯誤的版本，嘗試查看「TODO」並嘗試除錯及完成App作品。

製成品Windows連結例子: <http://localhost/ionic-labs-1/smilelottery>

製成品Mac連結例子: <http://localhost:8888/MAMP/ionic-labs-1/smilelottery>

錯誤版Windows連結例子: http://localhost/ionic-labs-1/smilelottery/index_lab.html

錯誤版Mac連結例子: http://localhost:8888/MAMP/ionic-labs-1/smilelottery/index_lab.html



Ionic實習1C

嘗試修改「selfintro」以完成「小明」的個人頁面。

Windows連結例子: <http://localhost/ionic-labs-1/selfintro>

Mac連結例子: <http://localhost:8888/MAMP/ionic-labs-1/selfintro>

答案Windows連結例子: http://localhost/ionic-labs-1/selfintro/index_ans.html

答案Mac連結例子: http://localhost:8888/MAMP/ionic-labs-1/selfintro/index_ans.html



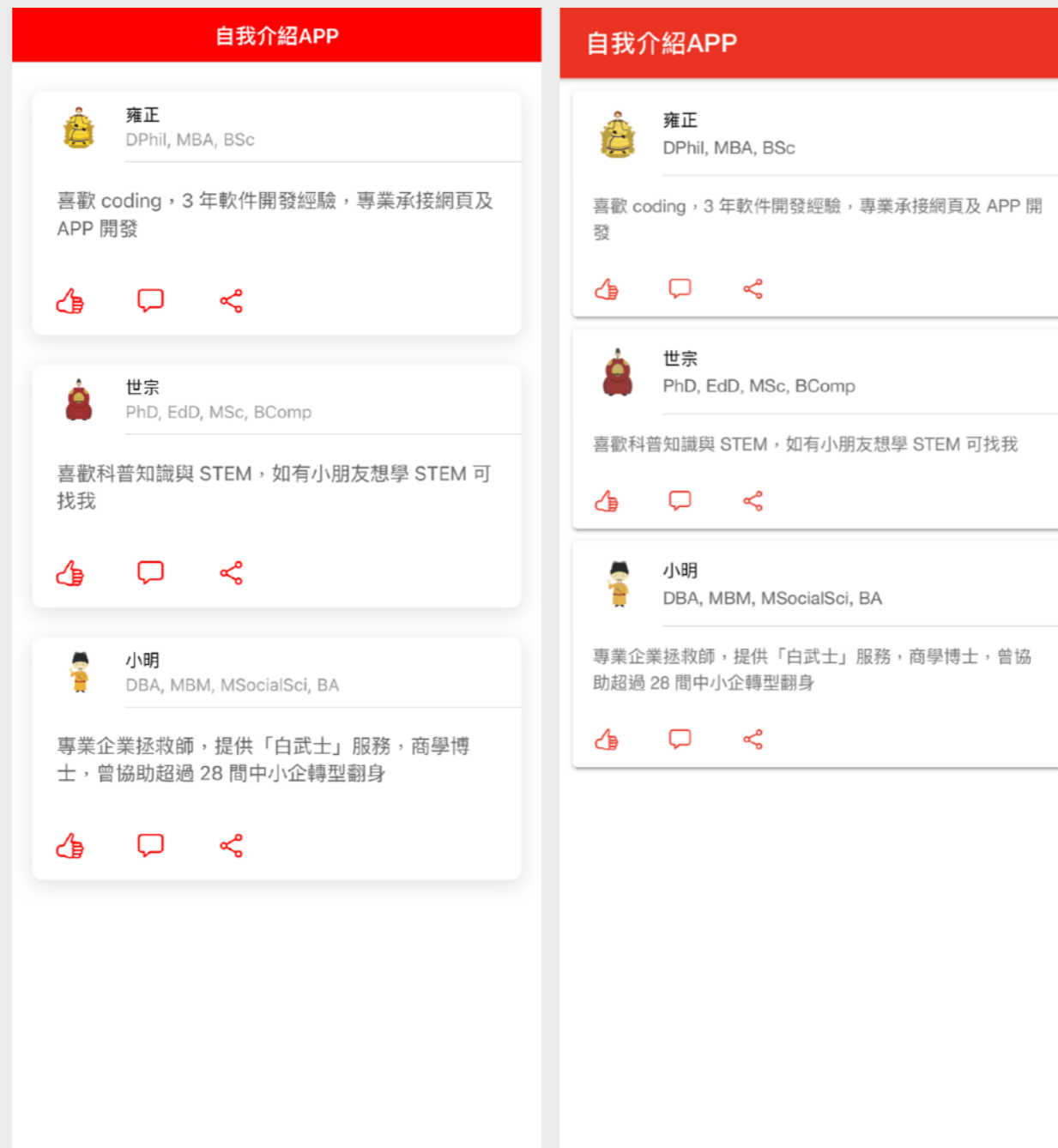
Ionic實習1D

在主頁為每位人物的ion-card (ion-card-content之後)加上「讚好」、「留言」及「分享」按鈕。可參考：

```
<ion-button fill="clear">
```

```
  <ion-icon name="thumbs-up-outline"></ion-icon>
```

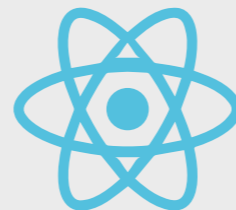
```
</ion-button>
```



課題六：

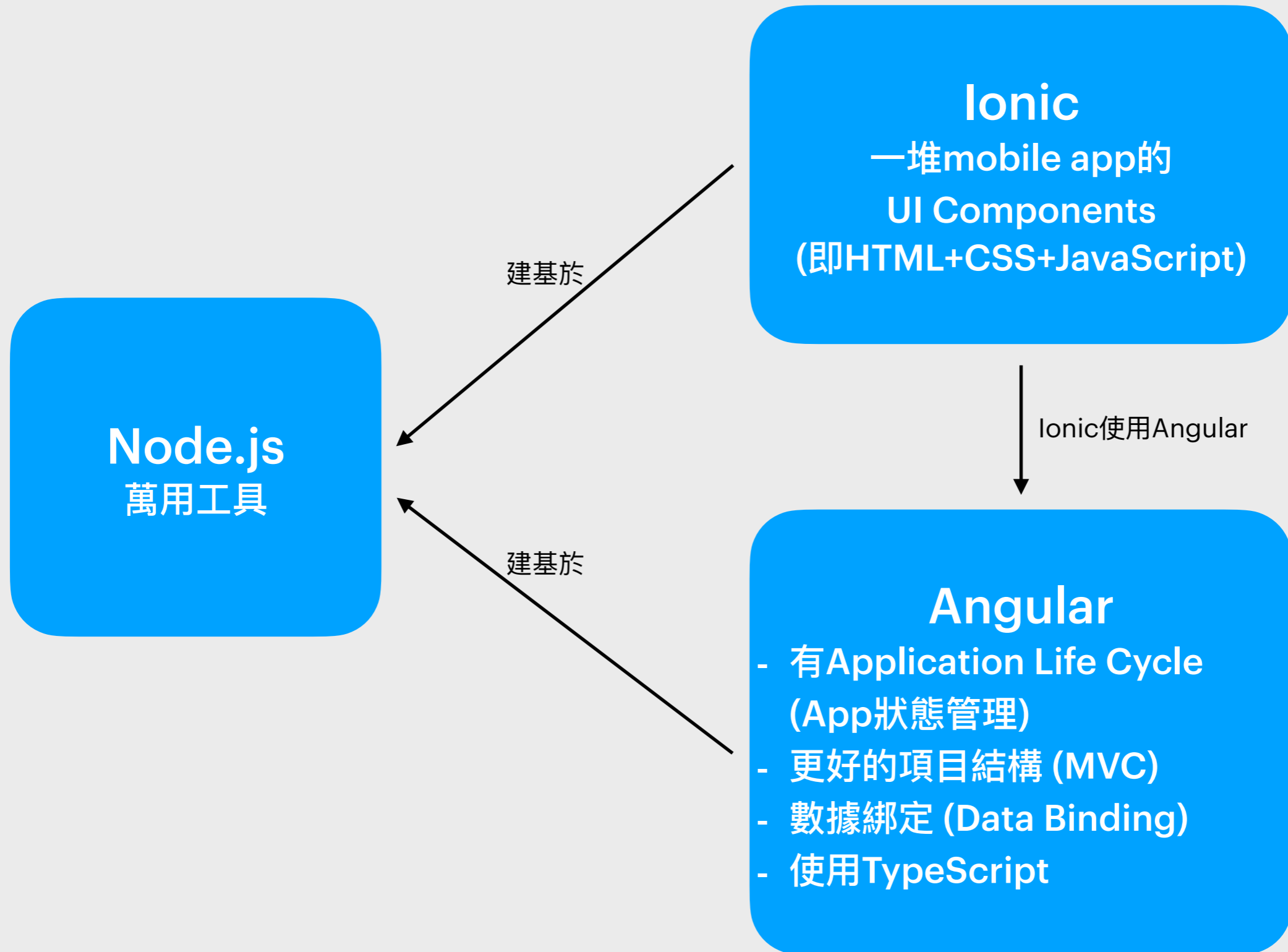
Angular及Node.js與Ionic應用

開發框架...



除了更用 Ionic + JavaScript/Angular 開發手機應用程式，
我們亦有其他選擇如：Framework7 + React, Onsen + Vue 等

Node.js、Ionic與Angular的關係



項目實習-附近商場APP-簡介

我們會以實習學習項目來學習Ionic及Angular，是次的實習就是製作出一隻「附近商場APP」App，本App分三頁：

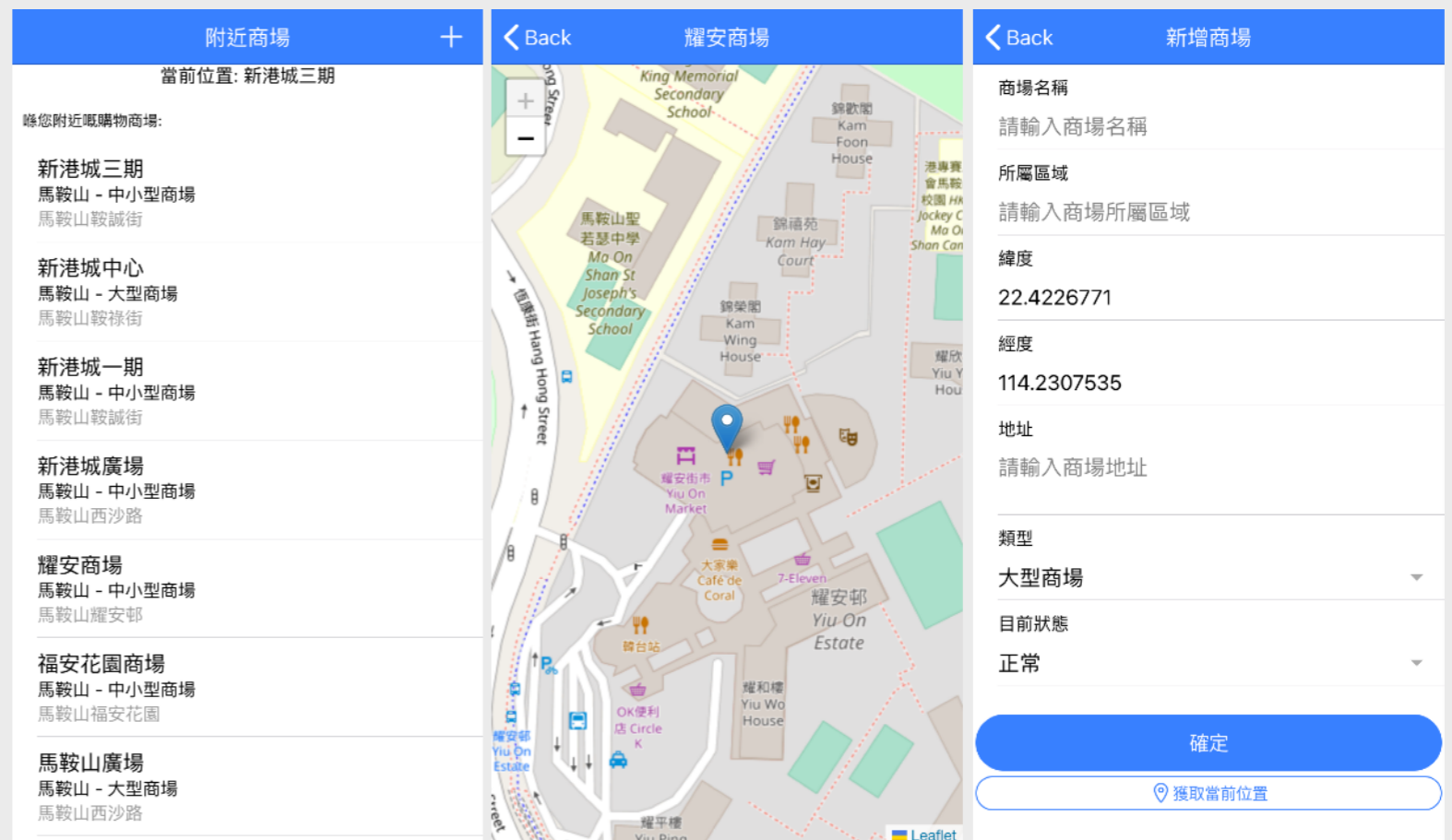
主頁 - 獲取用家所身處的地方之地理位置，並把商場由近至遠的方式排列出來，同時顯示商場的名稱、地區、類型及地址。

地圖頁 - 透過第三方程式庫Leaflet以地圖定位方式顯示出商場的位置。頁面標題並寫著商場的名稱。

新增商場頁 - 用戶可在此頁面填寫表單來新增商場到伺服器。填寫表單的同時，用戶還可要求程式獲取當前位置以方便填寫經緯度的資料。

完整版本程式碼下載：

<https://s3objectstorage-a.ap-south-1.linodeobjects.com/ionic-labs-2.zip>



電腦系統基本指令碼

因為Node.js只支援命令列介面(Command Line Interface, CLI)，所以若要透過Node.js來操作Ionic及Angular，我們需要學習一些基礎的指令碼去指令電腦工作。以下就是一些常的指令碼：

Windows	MacOS / Linux	說明	例子
cd	cd	進入到某目錄	cd ionicFolder/myApp
dir	ls	列出目前目錄的檔案	ls
mkdir	mkdir	建立新的目錄	mkdir myApp
copy	cp	複製檔案	cp -r myApp myApp2
move	mv	移動檔案或更改檔案名稱	mv -r myApp myGoodApp
del	rm	刪除檔案	rm -r myApp
control c	control c	停止運行運作中的指令	control c (不需按「Enter」)

安裝Node.js及Ionic (1)

如要用上選用Angular作開發框架的Ionic項目，我們首先需要安裝Node.js及Ionic，我們會有以下步驟：

1. 下載及安裝Node.js：

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

2. 【只限**Mac**用戶，Windows用戶不適用並請跳到下頁】，先執行：

```
sudo chown -R 用戶名稱 ~/.npm
```

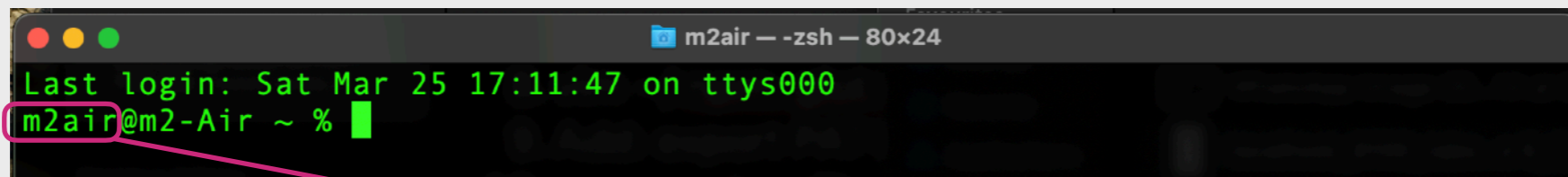
然後輸入Mac電腦的密碼(不會出現在螢幕上)然後鍵盤按return再執行：

```
sudo chown -R 用戶名稱 /usr/local/lib/node_modules
```

然後再執行：

```
sudo chown -R 用戶名稱 /usr/local/bin
```

讀者的Mac電腦用戶名稱，即終端機左邊@號之前的名稱，如：

A terminal window screenshot with a black background and green text. The title bar reads 'm2air - zsh - 80x24'. The prompt shows 'Last login: Sat Mar 25 17:11:47 on ttys000' followed by 'm2air@m2-Air ~ %'. The 'm2air' part of the prompt is circled in red, and a red arrow points from this circle to the text '例子中的用戶名為m2air' in the following block.

例子中的用戶名為**m2air**，即例子中便會執行：

```
sudo chown -R m2air ~/.npm 及 sudo chown -R m2air /usr/local/lib/node_modules 及  
sudo chown -R m2air /usr/local/bin
```

3. 然後打開終端機(即「命令提示字元」)下載及安裝NPM：

```
npm i -g npm@latest
```

安裝Node.js及Ionic (2)

4. 下載及安裝Angular :

```
npm i -g @angular/cli@latest
```

5. 下載及安裝ServiceWorker :

```
npm i -g @angular/service-worker@latest
```

6. 使用npm安裝Ionic :

```
npm i -g @ionic/cli@latest
```


建立Ionic Angular項目 (1)

要使用選用Angular作開發框架的Ionic項目，如已安裝好了Node.js及Ionic，並創建好了用來Ionic項目的目錄，我們有以下步驟：

1. 把終端機內的目錄位置設為用來Ionic項目的目錄 (使用終端機)

```
cd [DirectoryName]
```

2. 以Angular作開發框架來創建Ionic app (使用“blank”模板)

```
ionic start [AppName] blank --type=angular
```

3. 進入剛才所創建了的Ionic app的目錄

```
cd [AppName]
```

4. 加入匯出PWA的模組

```
ng add @angular/pwa
```

5. 運行該Ionic app

```
ionic serve
```

建立Ionic Angular項目 (2)

如被問及是否成為Ionic會員，亦可答「Y」或「N」

```
Initialized empty Git repository in /Users/dave/Desktop/myApp/.git/
```

```
Join the Ionic Community! ❤️
```

```
Connect with millions of developers on the Ionic Forum and get access to live events, news updates, and more.
```

```
? Create free Ionic account? (y/N) █
```

```
Connect with millions of developers on the Ionic Forum and get access to live events, news updates, and more.
```

```
? Create free Ionic account? (y/N) N█
```

```
143 },
144 "schematics": {
145   "@ionic/angular-toolkit:component": {
146     "styleext": "scss",
147     "spec": false
148   },
149   "@ionic/angular-toolkit:page": {
150     "styleext": "scss",
151     "spec": false
152   },
153   "@angular-eslint/schematics:application": {
```

到angular.json內的 `@ionic/angular-toolkit:component` 及 `@ionic/angular-toolkit:page` 內加上 `"spec": false`

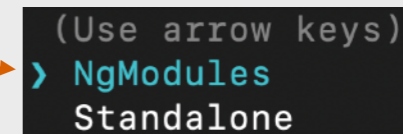
項目實習-附近商場APP-開創項目

那我們就一步一步的製作這隻App吧！

第一步，我們先開創一個新的Ionic Angular項目，在開創前，我們先以開啟終端機並以 `cd` 進入你想開創項目的資料夾(目錄)位置：`cd [DirectoryName]`

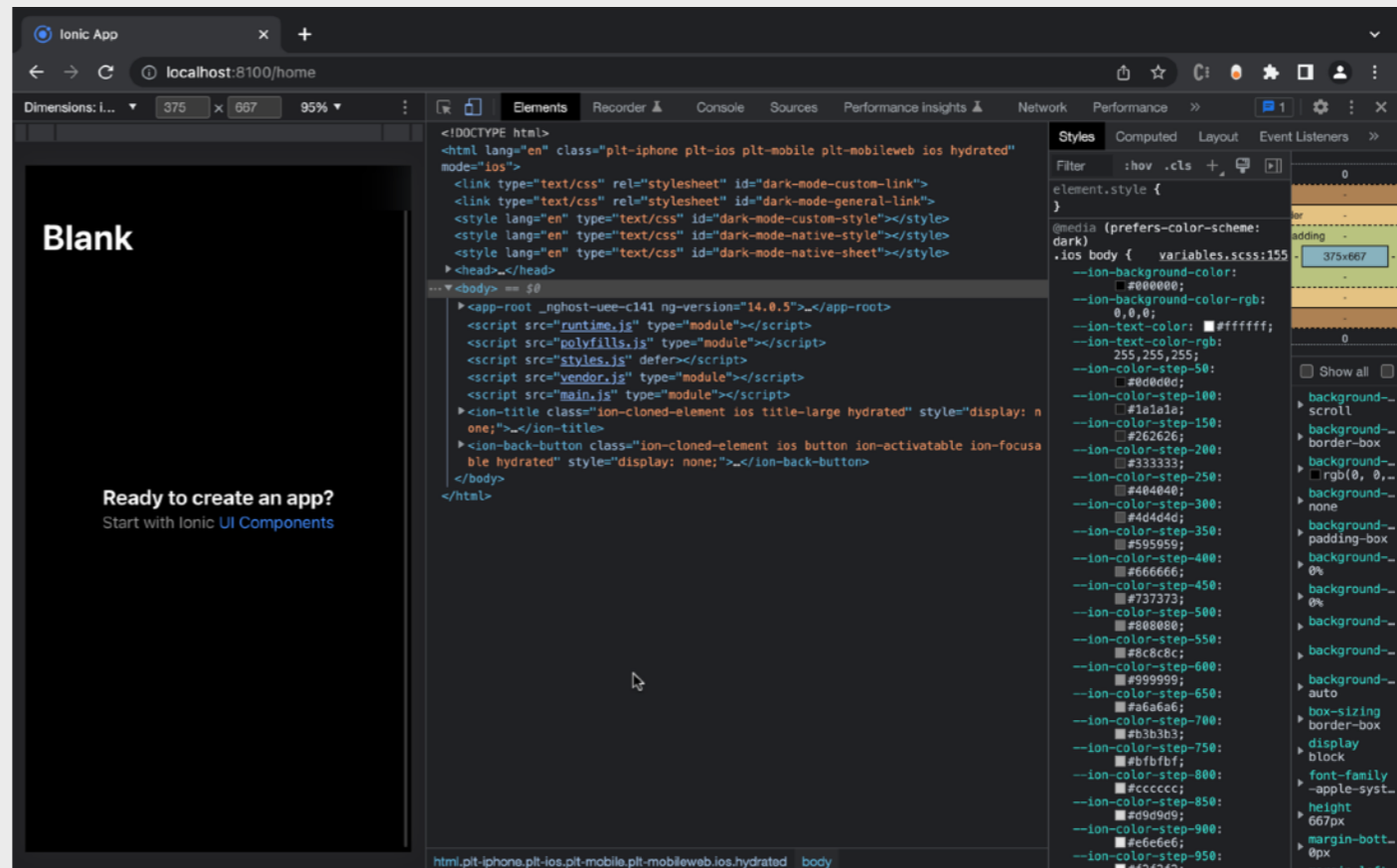
然後便開新Ionic Angular項目：`ionic start ShoppingMalls blank --type=angular`

然後選 **NgModules** (通常於鍵盤直接按Enter鍵即可)，
若果問及是否加進Capacitor或是否成為Ionic會員，答「N」即可。



然後把終端機進入 ShoppingMalls 資料夾裡：`cd ShoppingMalls`

開啟瀏覽器，
並調至手機模式



把TypeScript設定成預設默許Any類

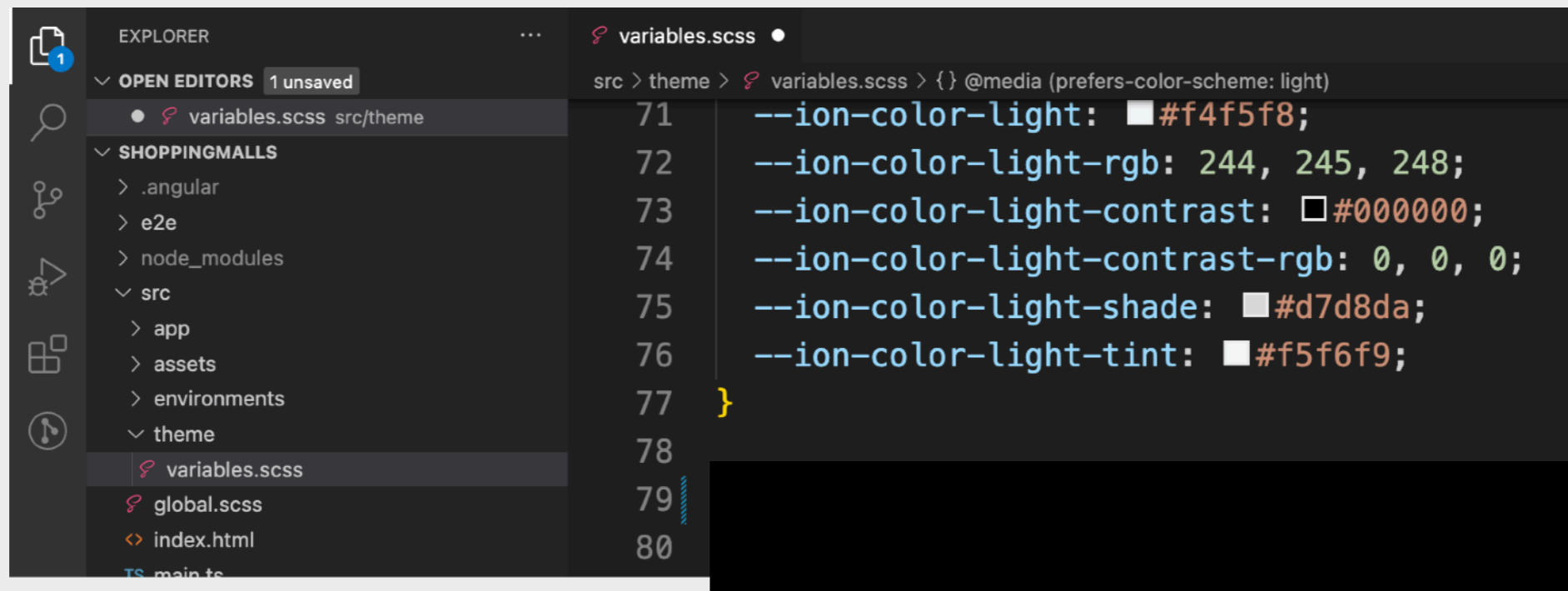
到 tsconfig.json，增加noImplicitAny並設為false，
亦把strict設為false

```
home.page.html TS home.page.ts TS tsconfig.json X TS app-routing.module.ts
TS tsconfig.json
1  /* To learn more about this file see: https://ang
2  {
3    "compileOnSave": false,
4    "compilerOptions": {
5      "baseUrl": "./",
6      "outDir": "./dist/out-tsc",
7      "forceConsistentCasingInFileNames": true,
8      "strict": false,
9      "noImplicitOverride": true,
10     "noPropertyAccessFromIndexSignature": true,
11     "noImplicitReturns": true,
12     "noImplicitAny": false,
13     "noFallthroughCasesInSwitch": true,
14     "sourceMap": true,
```

最後再運行App：`ionic serve`

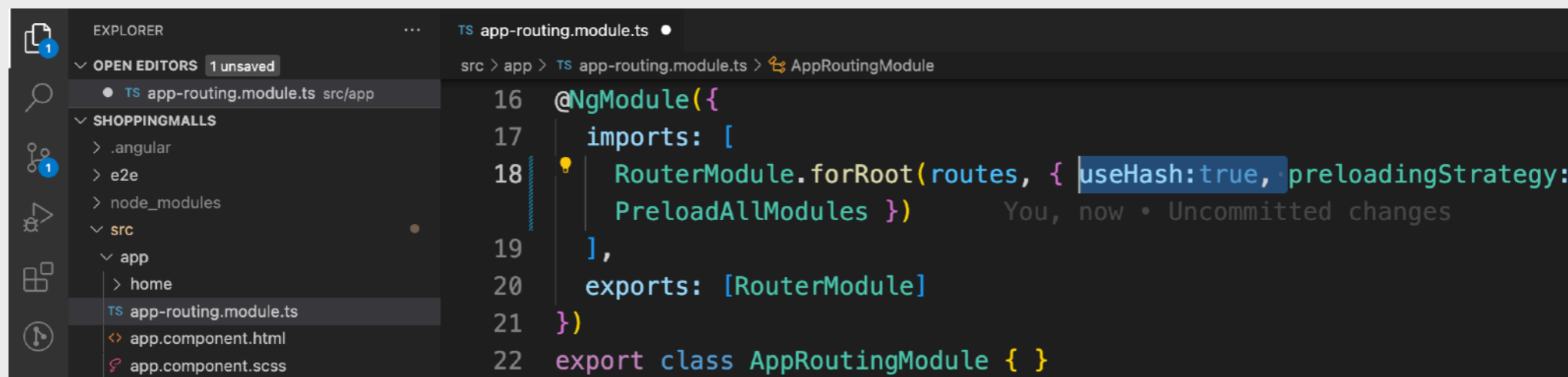
項目實習-附近商場APP-修改設置

接著我們到 `src/theme/variables.scss`，把 `light` theme 以後的CSS刪掉(即第78行開始向下一直刪除)，以令程式外觀主色調變成淺色風格，然後儲存它。



```
EXPLORER
  OPEN EDITORS 1 unsaved
    variables.scss src/theme
  SHOPPINGMALLS
    .angular
    e2e
    node_modules
    src
      app
      assets
      environments
      theme
        variables.scss
        global.scss
        index.html
        main.ts
  variables.scss
    src > theme > variables.scss > {} @media (prefers-color-scheme: light)
      71 --ion-color-light: #f4f5f8;
      72 --ion-color-light-rgb: 244, 245, 248;
      73 --ion-color-light-contrast: #000000;
      74 --ion-color-light-contrast-rgb: 0, 0, 0;
      75 --ion-color-light-shade: #d7d8da;
      76 --ion-color-light-tint: #f5f6f9;
      77 }
      78
      79
      80
```

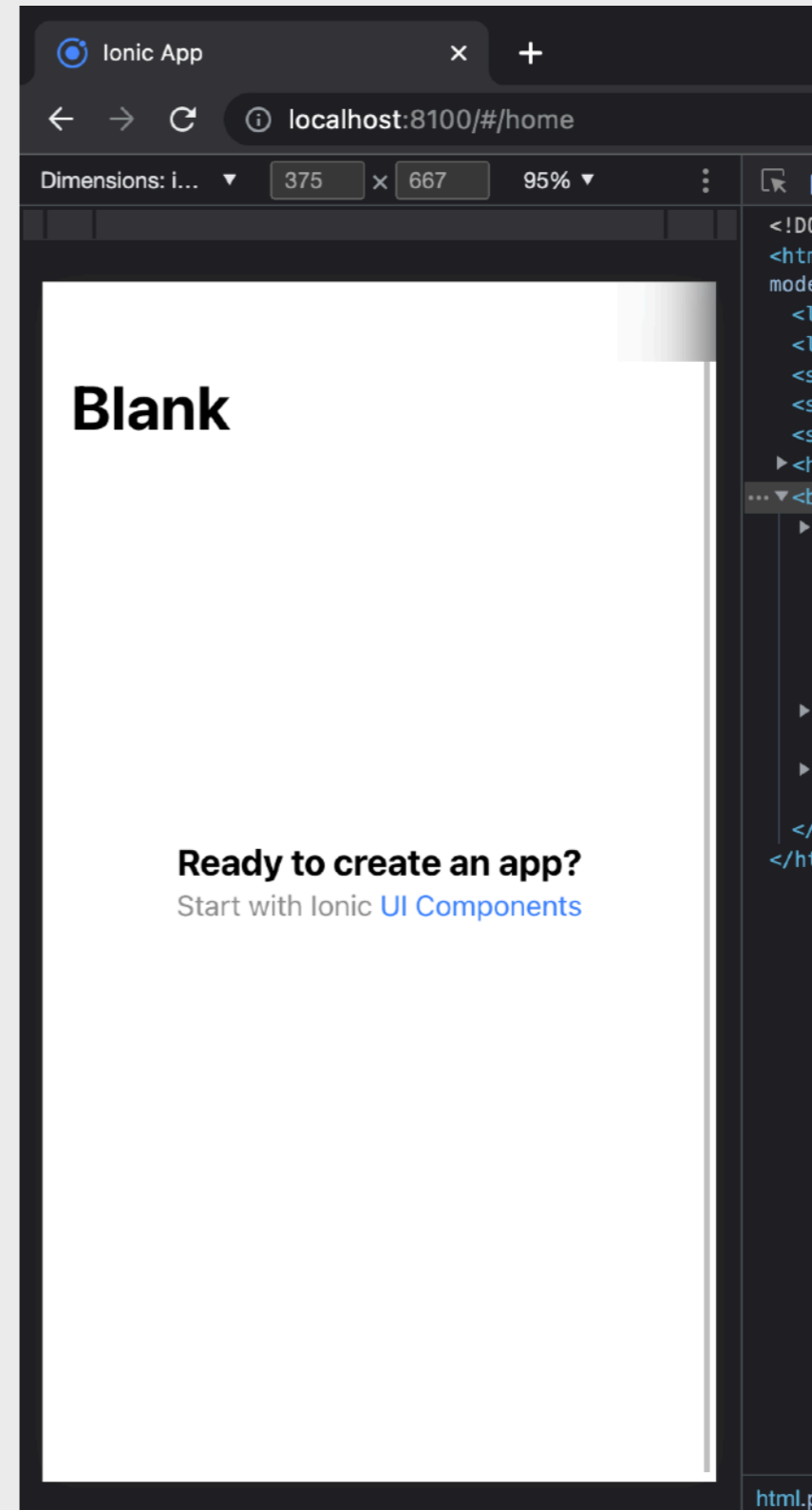
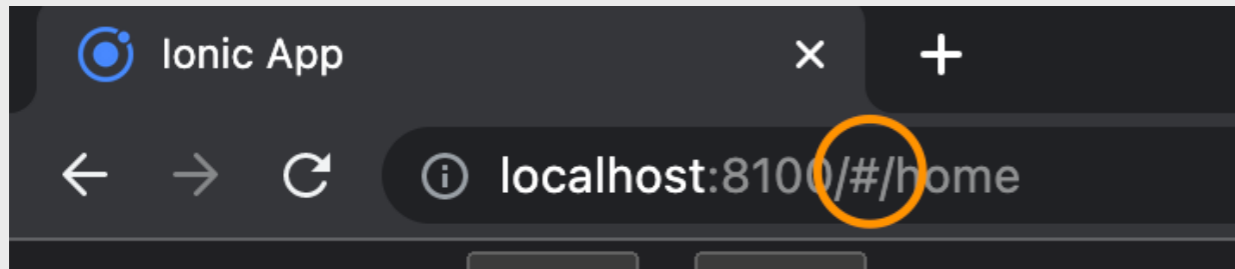
接著我們到 `src/app/app-routing.module.ts`，如下圖般在 `preloadingStrategy` 前加上：`useHash:true`，(約第18行)，以令App可以在「路徑型伺服器」(Path-based)上順利運作。



```
EXPLORER
  OPEN EDITORS 1 unsaved
    TS app-routing.module.ts src/app
  SHOPPINGMALLS
    .angular
    e2e
    node_modules
    src
      app
        home
        TS app-routing.module.ts
        app.component.html
        app.component.scss
  TS app-routing.module.ts
    src > app > TS app-routing.module.ts > AppRoutingModuleModule
      16 @NgModule({
      17   imports: [
      18     RouterModule.forRoot(routes, { useHash:true, preloadingStrategy:
      19       PreloadAllModules })
      20   ],
      21   exports: [RouterModule]
      22 })
      export class AppRoutingModuleModule { }
```

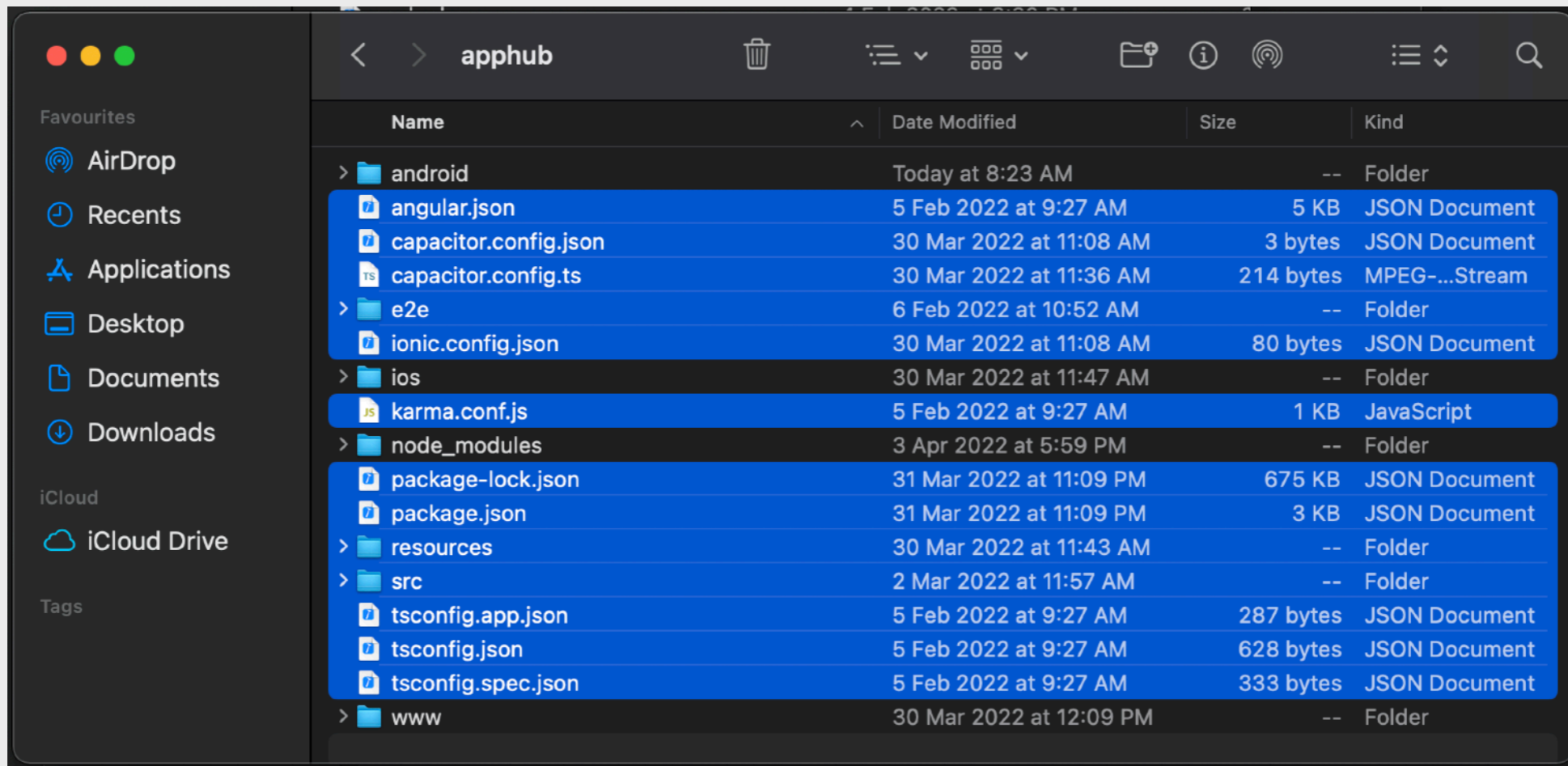
項目實習-附近商場APP-(續)修改設置

儲存後會在瀏覽器中看到網址中加多了#號，
而外觀則變成了淺色風格。

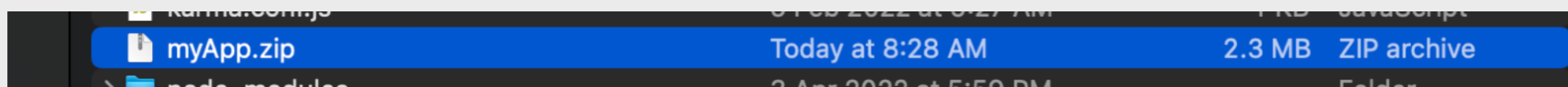


分享Ionic Angular項目

如要分享或備份App原始項目，我們可以把項目壓縮，但注意不需把「node_modules」、「www」及原生App目錄壓縮，否則壓縮檔會變得非常的大

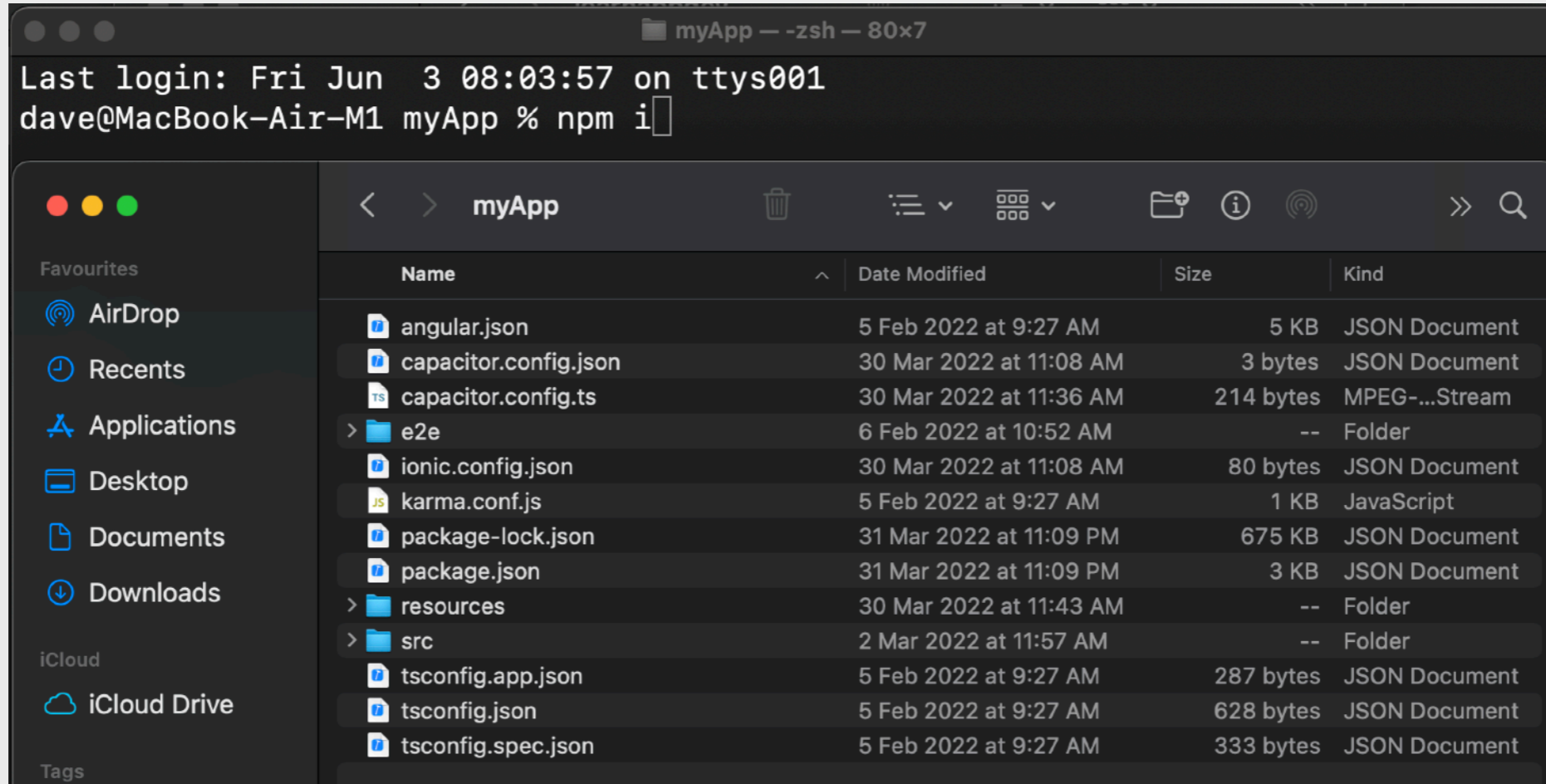


壓縮成壓縮檔，可更易地分享給別人或作備份

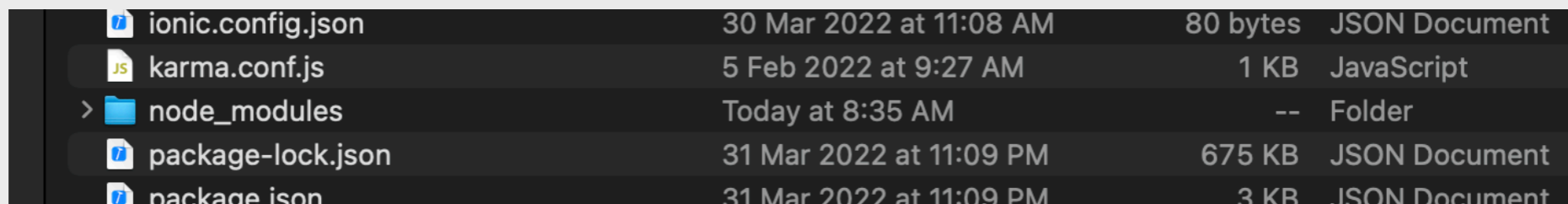


接手Ionic Angular項目

當接到App原始項目後，我們便可把它解壓縮，然後到終端機執行「`npm i`」指令以下載「node_modules」



「node_modules」 下載好了



Ionic Angular項目的結構

- 全域共用的SCSS - `src/app/global.scss`
- App標誌啟動畫面 - 於`resources`內
- 資產 - 於`src/assets`內
- 頁面(Pages)及服務檔(Services) - 於`src/app`內

Ionic Angular項目主要檔案類型

.ts - TypeScript 檔案

建基於JavaScript的一種物件導向程語言，
APP的客戶端邏輯運算就寫在這些檔案
(例子: tab1.page.ts)

.html - HTML 檔案

使用者界面及其佈局 (例子: tab1.page.html)

.scss - SCSS 檔案

套用CSS樣式到HTML元素
(例子: tab1.page.scss)

TypeScript簡介

Angular使用TypeScript作為其開發語言

物件導向程式語言 (OOP)

建基於JavaScript而開發出來並完全兼容JavaScript

TypeScript比JavaScript更適用於大型項目

改進了Class(類別)的概念與應用

Class(類別)中可為屬性及函式加上5種前綴來決定存取該屬性的權限：
public (預設值)、private、protected、static 及 readonly

函式內需使用**this**.來存取Class(類別)中所定義的變數及常數

物件導向程式簡介 (1)

擁有Class(類別)及物件(Object)的概念

Class(類別)就像一幅藍圖或設計圖

而根據藍圖生產出來的東西就叫作「物件(Object)」或「實體(Instance)」

實體被創建時會自動執行「constructor」函式，如子類別多載或複寫「constructor」時，「constructor」函式內的首句必需為「super();」

試舉一例，以下就是Car(車子)的Class(類別)，Car是車子的藍圖，藍圖中有一個文字屬性叫作「brand」，亦有一個函式叫做「constructor」：

```
class Car {  
  brand:string;  
  
  constructor() {  
    console.log('Car created');  
  }  
}
```

物件導向程式簡介 (2)

四大特性：

封裝(Encapsulation)、繼承(Inheritance)、多型(Polymorphism)、抽象(Abstraction)

封裝(Encapsulation)：

把具體執行步驟隱藏，即是把程式封裝成函式(程式組)，使程式更易的被呼叫。

假設旺財是一條狗，從前我們要叫牠去吠叫：

定義旺財
旺財.設置音調(5)
旺財.吸氣()
旺財.吐氣()

現在的做法(我們已把吠叫的程式封裝成函式「吠叫」)：

定義旺財是狗
旺財.吠叫()

物件導向程式簡介 (3)

繼承(Inheritance)：

一個類別可能會有「子類別」，子類別比原本的類別（稱為父類別）要更加具體化。

例如「狗」這個類別包含了「八哥」和「芝娃娃」。

又例如「電話」這個類別包含了「家居電話」、「手提電話」和「公眾電話」等。

子類別會繼承父類別的屬性和行為，例如「狗」類別擁有「吠叫」這個函式，「八哥」和「芝娃娃」類別皆繼承了「狗」類別，皆可以執行「吠叫」的函式。

當然，子類別正常應該都比父類別更加具體化，除了繼承父類別，子類別還應該會加入了一些自身獨特的函式和屬性。例如「芝娃娃」加入了一個特別的函式為「扮外星人」，這個函式則只有「芝娃娃」這個類別能執行，父類別(即「狗」類別)則沒有此函式，當然父類別亦不能執行此函式。

物件導向程式簡介 (4)

繼承(Inheritance)例子

```
class Phone {  
    brand: string;  
    makeCall(){ /* ... */ }  
    receiveCall(){ /* ... */ }  
}
```

```
class MobilePhone extends Phone {  
    battery: number;  
    launchApp(){ /* ... */ }  
}
```

物件導向程式簡介 (5)

多型(Polymorphism)：

一個類別或父子類別中相同名稱的函式。多型有兩種，分別是多載(Overloading)和複寫(Overriding)。

多載(Overloading)是指在同一個類別或父子類別中相同名稱而所接受傳入的參數數量或參數型態卻不同的函式(但JavaScript及TypeScript不接受參數型態不同)。例如：

```
吠叫(){ 設置音量(5); 吸氣(); 發音(); }
```

與

```
吠叫(音量){ 設置音量(音量); 吸氣(); 發音(); }
```

複寫(Overriding)是指複寫了父類別中的相同名稱而相同參數數量與參數型態的函式。例如父類別中的：

```
吠叫(){ 設置音量(5); 吸氣(); 發音(); }
```

被複寫成：

```
吠叫(){ 設置音量(10); 吸氣(); 露出憤怒的樣貌(); 發音(); }
```


物件導向程式簡介 (6)

抽象(Abstraction)：

抽象類別(Abstract Class)是不能直接實例化的類別，因為抽象類別沒有被實體化的意義。例如「動物」是一個很廣義的類別，「狗」、「企鵝」、「猩猩」、「人」都同屬於「動物」，但現實中我們卻不能創建一只「動物」，所以「動物」這個類別是很抽象的，只能被「繼承」。

而抽象函式(Abstract Method)則只是一個只有寫著名稱及其參數名稱的函式。例如「移動」是「動物」的一個函式，但現實中「蛇」、「魚」、「人」、「貓」的移動方式卻有著大大的不同，蛇的是用腹鱗及肌肉的縮放而移動，魚用游，人用雙腳，貓用雙手雙腳。雖然「所有動物皆可移動」，「移動」函式是必然存在的(「天生的」)，但此函式則是各有各的執行方法，因此「移動」便是一個「抽象函式」。

而抽象屬性(Abstract Property)則是必然存在的但各個「繼承類別」皆可會不同而成為了抽象的屬性或變數。例如「名稱」、「身分證號碼」等。

物件導向程式簡介 (7)

抽象(Abstraction)例子

```
abstract class Person {
  abstract name: string;

  display(): void{
    console.log(this.name);
  }
}

class Employee extends Person {
  name: string;
  empCode: number;

  constructor(name: string, code: number) {
    super();
    this.empCode = code;
    this.name = name;
  }
}
```

例子出自: <https://www.tutorialsteacher.com/typescript/abstract-class>

函式與屬性存取權限範圍

權限值	存取範圍	解釋
public	公域	可被任意存取
private	私域	只限自身Class(類別)可存取，Subclass(子類別)不可存取
protected	私域及子域	自身Class(類別)及Subclass(子類別)皆可存取

課題七： App及頁面生命週期

Ionic Platform程式生命週期

我們有時候需要在手機App啓動、暫停或恢復運作時處理一些事情，例如在App啓動時載入初始資料(如在記憶體裡擷取用戶的ID並從伺服器中載入用戶資料)，又例如在App進入背景狀態時儲存資料，又例如在App從背景狀態中恢復運作時更新資料……

在ts檔的頂端加入：

```
import { Platform } from '@ionic/angular';
```

在constructor加入變數定義：

```
constructor(public platform: Platform){ /* ... */ }
```

狀態	解釋	應用
ready	App開啓時被觸發	<code>this.platform.ready().then(()=>{ /* ... */ });</code>
pause	App進入背景狀態時被觸發	<code>this.platform.pause.subscribe(()=>{ /* ... */ });</code>
resume	App從背景中被開出來恢復運作時被觸發	<code>this.platform.resume.subscribe(()=>{ /* ... */ });</code>

Ionic 頁面生命週期

Ionic 除了在 Platform 中提供了 App 生命週期的觸發，Ionic 及 Angular 還提供了多個頁面 (Page) 生命週期的函式。

函式	解釋	應用
ngOnInit	頁面首次被開啓的剎那	<code>ngOnInit(){ /* ... */ }</code>
ngOnDestroy	頁面首次被掛掉的剎那	<code>ngOnDestroy(){ /* ... */ }</code>
ionViewWillEnter	頁面即將被開啓時	<code>ionViewWillEnter(){ /* ... */ }</code>
ionViewDidEnter	頁面已被開啓了的剎那	<code>ionViewDidEnter(){ /* ... */ }</code>
ionViewWillLeave	用戶將快離開頁面時	<code>ionViewWillLeave(){ /* ... */ }</code>
ionViewDidLeave	用戶已剛剛離開了頁面的剎那	<code>ionViewDidLeave(){ /* ... */ }</code>

續課題六：

Angular及Node.js與Ionic應用

Angular的ngIf和ngFor

透過Angular的ngIf和ngFor，我們將可以在HTML的元件中直接加入條件判斷的邏輯程式。

ngIf用來判斷HTML元件是否顯示，用法和一般的If Statement沒有甚麼分別，唯不用像在.ts檔裡般存取變數時加上「this.」的字眼。例如：

```
<ion-button *ngIf="!isMember" (click)="becomeMember()">成為會員</ion-button>
```

ngFor則是JavaScript中的For Each Loop Statement，基於一條陣列以迴圈方式而自動編寫HTML元件。例如：

```
<ion-list>  
  <ion-item *ngFor="let food of foods">  
    <ion-label>{{ food.name }}</ion-label>  
  </ion-item>  
</ion-list>
```

如要取得For Each Loop中的index，則可加上「index as i」即：

```
*ngFor="let food of foods;index as i"
```


Angular的ng-container

「ng-container」就是一個虛擬的HTML元件，它並不會被顯示出來，它主要給開發者把HTML元件分組或是把需要同一時間加入ngFor與ngIf的元件(因一個元件不能同時加入ngFor與ngIf)以一層虛擬元件包著來加入ngFor與ngIf。

例如我們不能這樣寫：

```
<ion-button *ngFor="let btn of buttons; index as i" *ngIf="btn.canLike" (click)="likePost(i)">  
  成為會員  
</ion-button>
```

但我們可透過「ng-container」來包著並使用「ngFor」渲染出各個IonButton元件，例如：

```
<ng-container *ngFor="let btn of buttons; index as i">  
  <ion-button *ngIf="btn.canLike" (click)="likePost(i)">  
    成為會員  
  </ion-button>  
</ng-container>
```

Angular的數據綁定-簡介

Angular透過數據綁定(Data Binding)，
把TypeScript (即App的邏輯運算程式)與HTML元件雙方自動互傳資料，
讓開發者不用手動更新HTML元件來顯示出已更新的資料

Angular有4種數據綁定，分別是：

插值(Interpolation)

屬性綁定(Property Binding)

事件綁定(Event Binding)

雙向綁定(Two-way Data Binding)

Angular的數據綁定-插值

插值(Interpolation)

在HTML元件中的內容加入：`{{value}}`

.ts檔

```
export class HomePage {  
  name = 'Eric';  
}
```

.html檔

```
<span>{{name}}</span>
```

項目實習-附近商場APP-插值實作

1. 接著我們到 `src/app/home/home.page.html`，把內容清理成只有簡潔的 `ion-header` 與 `ion-content`。

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Blank
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content>

</ion-content>
```

2. 然後在 `ion-content` 內加上：

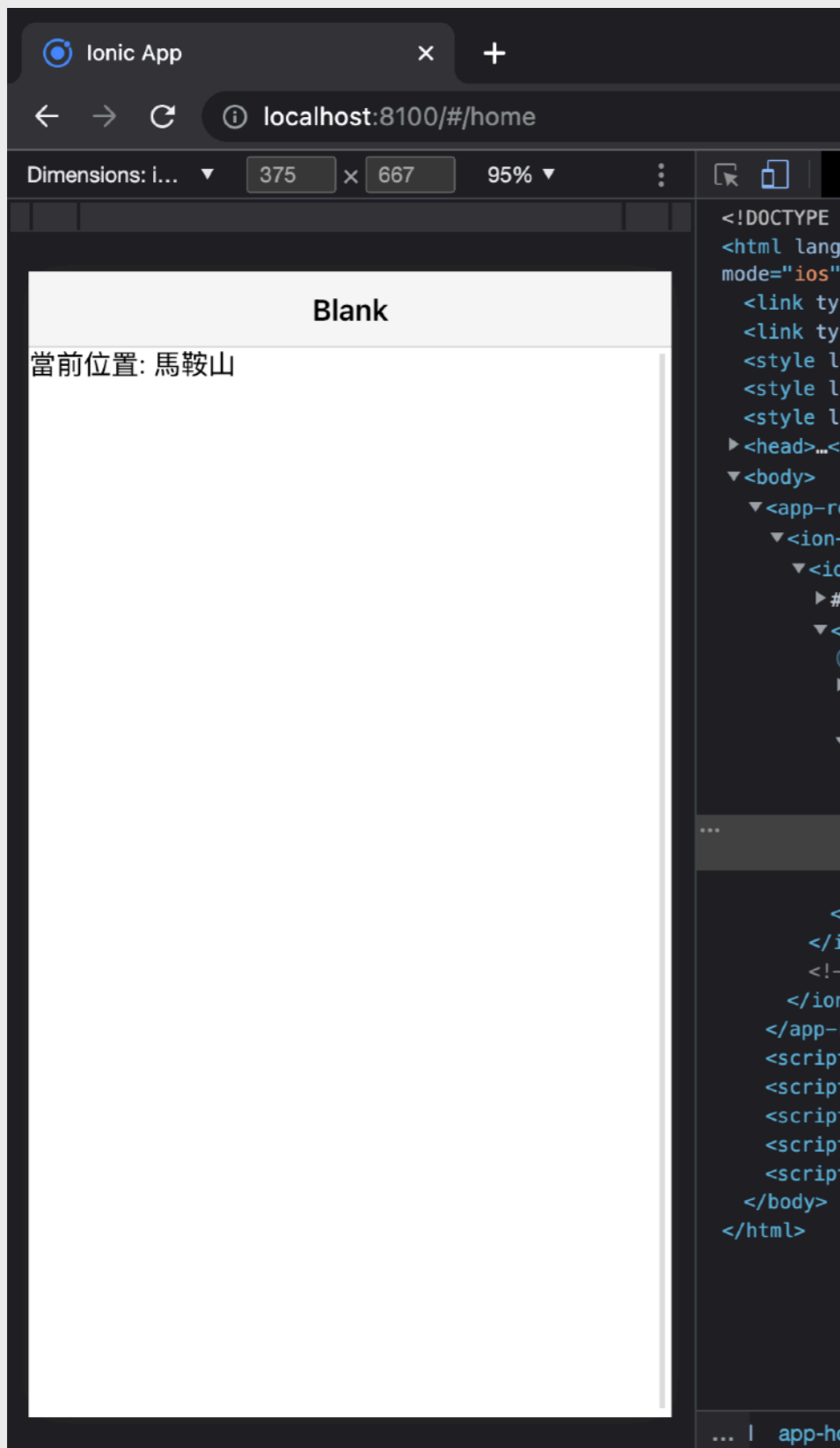
`<div class="Div_UserLocation">當前位置: {{ existingAddr }}</div>`
而這個 `existingAddr` 就是數據綁定中的插值。

```
10 <ion-content>
11
12   <div class="Div_UserLocation">當前位置: {{ existingAddr }}</div>
13
14 </ion-content>
```

3. 然後到 `src/app/home/home.page.ts` 中定義並賦予值「馬鞍山」：`existingAddr='馬鞍山'`；最後儲存兩個檔案。

```
10 export class HomePage {
11
12
13   existingAddr='馬鞍山';
14
15
16   constructor(){}
17
18
19 }
```

項目實習-附近商場APP-插值實作(續)



我們可以在App裡看到，使用者界面(HTML頁面)中顯示了TypeScript檔中的變數 `existingAddr`。

項目實習-附近商場APP-插值實作(續)

然後我們到 `src/app/home/home.page.ts` 加上JSON變數 `displayShoppingMalls`。

```
displayShoppingMalls=[
  {"raw_id":"1","mall_name":"康盛花園商場","mall_district":"將軍
  澳","mall_lat":"22.319778810368334","mall_lon":"114.25309928808232","mall_addr":"將軍澳寶琳北路1
  號康盛花園","mall_isbig":"0","mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
  {"raw_id":"2","mall_name":"翠林新城","mall_district":"將軍
  澳","mall_lat":"22.322606960496227","mall_lon":"114.24915413411578","mall_addr":"將軍澳翠林邨翠琳
  路11號","mall_isbig":"0","mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
  {"raw_id":"3","mall_name":"慧安商場","mall_district":"將軍
  澳","mall_lat":"22.324215110013157","mall_lon":"114.25415525501846","mall_addr":"將軍澳寶林毓雅里
  9號","mall_isbig":"0","mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
  {"raw_id":"4","mall_name":"慧星匯","mall_district":"將軍
  澳","mall_lat":"22.324247573426288","mall_lon":"114.25418206512319","mall_addr":"將軍澳寶林毓雅里
  9號慧安商場一樓","mall_isbig":"0","mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
  {"raw_id":"5","mall_name":"寶林商場","mall_district":"將軍
  澳","mall_lat":"22.325097983286835","mall_lon":"114.25616749930711","mall_addr":"將軍澳寶林寶林
  邨","mall_isbig":"0","mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"}
];
```

項目實習-附近商場APP-插值實作(續)

加上JSON變數 `displayShoppingMalls` 後如下圖所示。

若果在編輯器中有所走位，我們可在Windows電腦鍵盤按下 `Alt + Shift + F` 或在Mac電腦鍵盤按下 `option + shift + F` 以調節程式碼的位置。

```
10 export class HomePage {
11
12
13   existingAddr='馬鞍山';
14
15   displayShoppingMalls=[
16     {"raw_id":"1","mall_name":"康盛花園商場","mall_district":"將軍澳","mall_lat":"22.319778810368334",
17     "mall_lon":"114.25309928808232","mall_addr":"將軍澳寶琳北路1號康盛花園","mall_isbig":"0",
18     "mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
19     {"raw_id":"2","mall_name":"翠林新城","mall_district":"將軍澳","mall_lat":"22.322606960496227",
20     "mall_lon":"114.24915413411578","mall_addr":"將軍澳翠林邨翠琳路11號","mall_isbig":"0",
21     "mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
22     {"raw_id":"3","mall_name":"慧安商場","mall_district":"將軍澳","mall_lat":"22.324215110013157",
23     "mall_lon":"114.25415525501846","mall_addr":"將軍澳寶林毓雅里9號","mall_isbig":"0","mall_status":"normal",
24     "created_timestamp":"2021-04-25 19:43:05"},
25     {"raw_id":"4","mall_name":"慧星匯","mall_district":"將軍澳","mall_lat":"22.324247573426288",
26     "mall_lon":"114.25418206512319","mall_addr":"將軍澳寶林毓雅里9號慧安商場一樓","mall_isbig":"0",
27     "mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
28     {"raw_id":"5","mall_name":"寶林商場","mall_district":"將軍澳","mall_lat":"22.325097983286835",
29     "mall_lon":"114.25616749930711","mall_addr":"將軍澳寶林寶林邨","mall_isbig":"0","mall_status":"normal",
30     "created_timestamp":"2021-04-25 19:43:05"}
31   ];
```

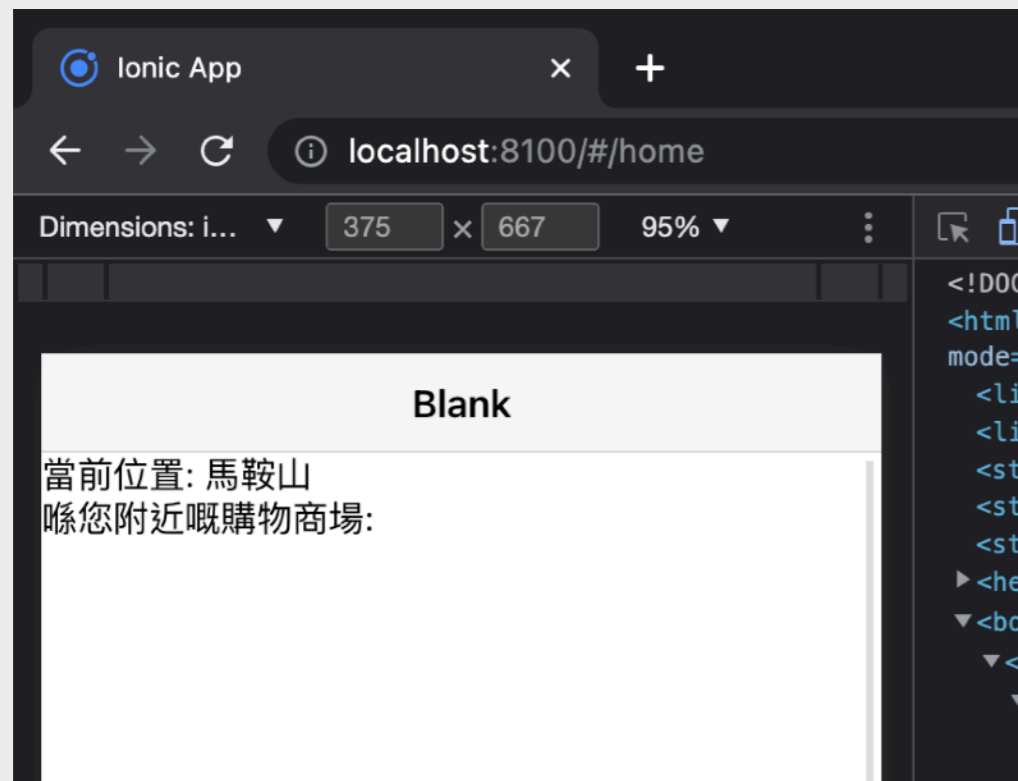
You, 4 minutes ago • Uncommitted changes

項目實習-附近商場APP-ngIf實作

然後我們到 `src/app/home/home.page.html` 中的 `ion-content` 內加上顯示標題。
請注意，我們加上了 `ngIf` 判斷式到 HTML 元件當中。

```
<div class="Div_LeftSmallTitle" *ngIf="displayShoppingMalls.length>0">睇您附近嘅購物商場:</div>
```

```
10 <ion-content>
11
12   <div class="Div_UserLocation">當前位置: {{ existingAddr }}</div>
13
14   <div class="Div_LeftSmallTitle" *ngIf="displayShoppingMalls.length>0">睇您附近嘅購物
    商場:</div>
15
16 </ion-content>
```

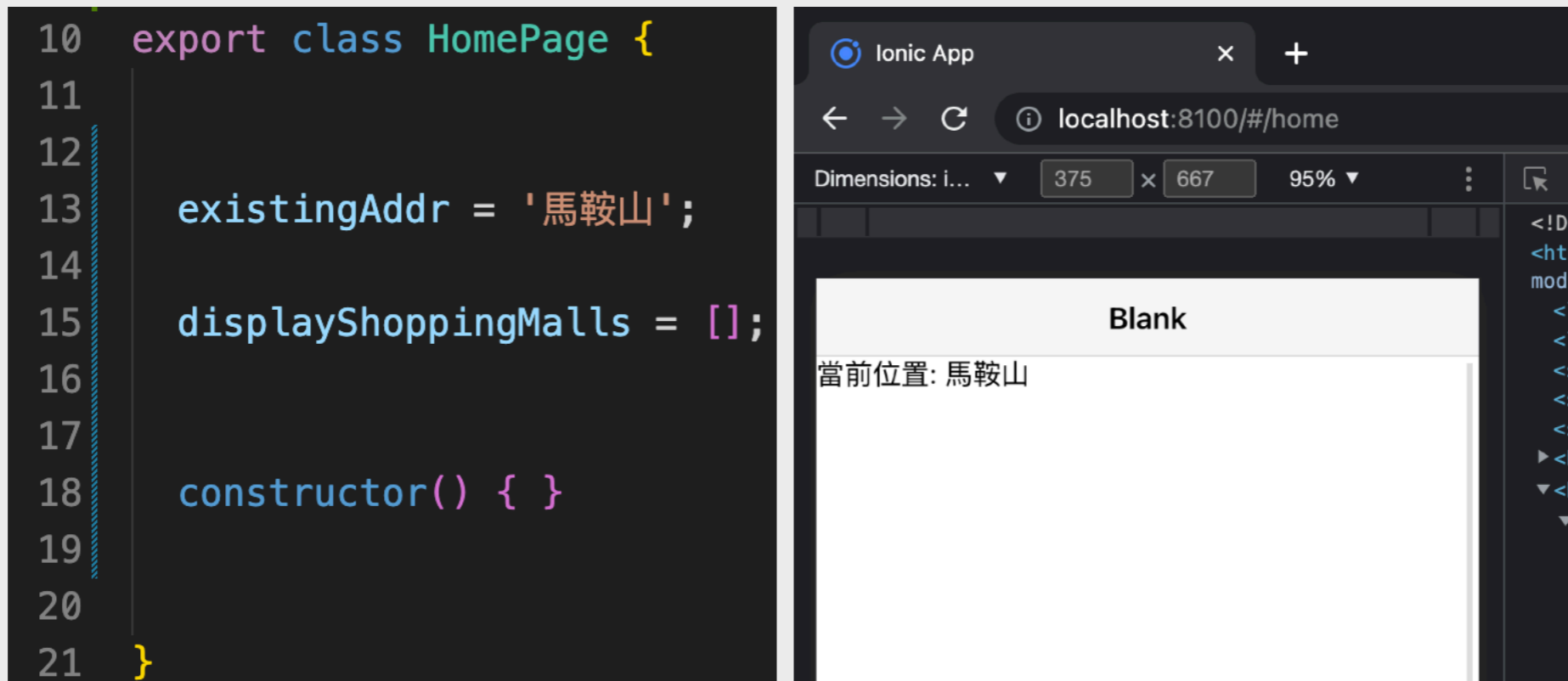


當儲存好檔案後，我們會看見標題出現了。
哪麼為甚麼標題會出現呢？

因為 `ngIf` 中的 `displayShoppingMalls` 的 `length` (長度) 大於 `0`，所以便會被顯示出來。

項目實習-附近商場APP-ngIf實作(續)

1. 若果我們暫時性地把 `displayShoppingMalls` 清空為 `[]`，那麼我們便可以看到 `ngIf` 中因為 `displayShoppingMalls.length` 不是大於 0，所以標題文字便不會被顯示。



2. 最後，我們把 `displayShoppingMalls` 還原並儲存。

```
10 export class HomePage {
11
12
13   existingAddr='馬鞍山';
14
15   displayShoppingMalls=[
16     {"raw_id":"1","mall_name":"康盛花園商場","mall_district":"將軍澳","mall_lat":"22.319778810368334",
17     "mall_lon":"114.25309928808232","mall_addr":"將軍澳寶琳北路1號康盛花園","mall_isbig":"0",
18     "mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
19     {"raw_id":"2","mall_name":"翠林新城","mall_district":"將軍澳","mall_lat":"22.322606960496227",
20     "mall_lon":"114.24915413411578","mall_addr":"將軍澳翠林邨翠琳路11號","mall_isbig":"0",
21     "mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
22     {"raw_id":"3","mall_name":"慧安商場","mall_district":"將軍澳","mall_lat":"22.324215110013157",
23     "mall_lon":"114.25415525501846","mall_addr":"將軍澳寶林靚雅里9號","mall_isbig":"0","mall_status":"normal",
24     "created_timestamp":"2021-04-25 19:43:05"},
25     {"raw_id":"4","mall_name":"慧星匯","mall_district":"將軍澳","mall_lat":"22.324247573426288",
26     "mall_lon":"114.25418206512319","mall_addr":"將軍澳寶林靚雅里9號慧安商場一樓","mall_isbig":"0",
27     "mall_status":"normal","created_timestamp":"2021-04-25 19:43:05"},
28     {"raw_id":"5","mall_name":"寶林商場","mall_district":"將軍澳","mall_lat":"22.325097983286835",
29     "mall_lon":"114.25616749930711","mall_addr":"將軍澳寶林靚雅里","mall_isbig":"0","mall_status":"normal",
30     "created_timestamp":"2021-04-25 19:43:05"}
31 ];
```

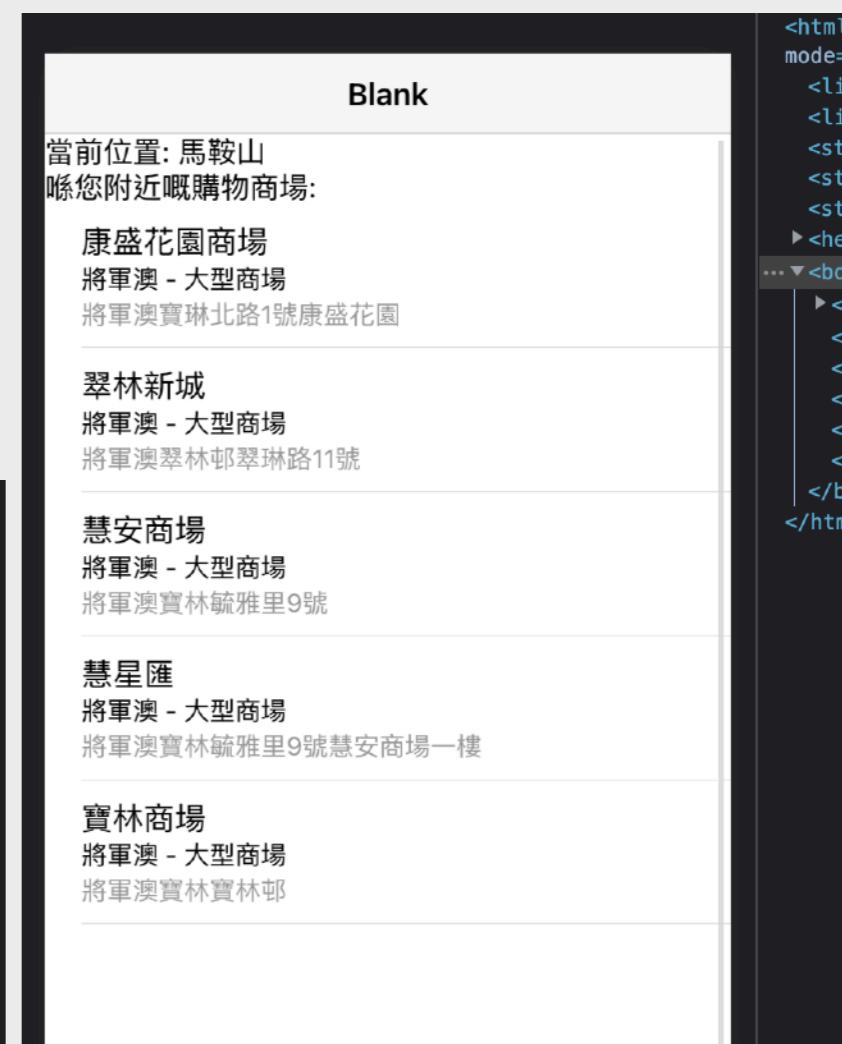
(如上上頁所示)

項目實習-附近商場APP-ngFor與插值實作

製作一個 `ion-list` 並使用 `ngFor` 迴圈出 `displayShoppingMalls` 的資料，同時在 `ion-label` 寫下 `h2`、`h3` 及 `p` 元素並使用插值來顯示資料。注意於 `h3` 中，我們使用了 `if else statement` 的簡寫來顯示出商場的類型。

```
<ion-list>
  <ion-item *ngFor="let m of displayShoppingMalls">
    <ion-label>
      <h2>{{ m.mall_name }}</h2>
      <h3>{{ m.mall_district }} - {{ m.mall_isbig ? '大':'中小' }}型商場</h3>
      <p>{{ m.mall_addr }}</p>
    </ion-label>
  </ion-item>
</ion-list>
```

```
12 <div class="Div_UserLocation">當前位置: {{ existingAddr }}</div>
13
14 <div class="Div_LeftSmallTitle" *ngIf="displayShoppingMalls.length>0">喺您附近嘅購物
15 商場:</div>
16
17 <ion-list>
18   <ion-item *ngFor="let m of displayShoppingMalls">
19     <ion-label>
20       <h2>{{ m.mall_name }}</h2>
21       <h3>{{ m.mall_district }} - {{ m.mall_isbig ? '大':'中小' }}型商場</h3>
22       <p>{{ m.mall_addr }}</p>
23     </ion-label>
24   </ion-item>
25 </ion-list>
26 </ion-content>
```



儲存後便會看見畫面顯示出了商場列表。

項目實習-附近商場APP-CSS實作(1)

最後透過CSS在 `src/app/home/home.page.scss`
把畫面弄得美觀一點。

```
.Div_EmptyRecord{  
  width: 100%;  
  margin-top: 15pt;  
  text-align: center;  
  color: var(--ion-color-primary);  
}
```

```
.Div_UserLocation{  
  width: 100%;  
  text-align: center;  
  font-size: 11pt;  
}
```

```
.Div_LeftSmallTitle{  
  width: 100%;  
  text-align: left;  
  font-size: 9pt;  
  padding-left: 6pt;  
  margin-top: 15.5pt;  
  margin-bottom: 8pt;  
}
```

```
.Table_ShopList{  
  width: 100%;  
  border: none;  
  border-spacing: 0;  
  padding: 0;  
}  
.Table_ShopList td{  
  border: none;  
  border-bottom: 1px solid rgb(200,200,200);  
  border-spacing: 0;  
  padding: 0;  
}  
.Td_SL_Icon{  
  width: 20%;  
}  
.Td_SL_Content{  
  width: 80%;  
}  
.IonImg_SL_Icon{  
  display: block;  
  width: 100%;  
}  
.Div_SL_ShopName{  
  padding-left: 6.5pt;  
  font-weight: bold;  
  font-size: 13.5pt;  
}  
.Div_SL_ShopDesc{  
  padding-left: 6.5pt;  
  font-size: 10pt;  
}
```



完成後的結果

Angular的數據綁定-屬性(1)

屬性綁定(Property Binding)

在HTML元件中的屬性加入：`[property]="value"`

例子一：資源連結

.ts檔

```
export class HomePage {  
  imgSrc = 'pic01.jpg';  
}
```

.html檔

```
<ion-img [src]="imgSrc"></ion-img>
```

例子二：CSS Class連結

.ts檔

```
export class HomePage {  
  eleCssClass = 'ClassA';  
}
```

.html檔

```
<div [ngClass]="eleCssClass"></div>
```

Angular的數據綁定-屬性(2)

屬性綁定(Property Binding)

例子三：CSS Inline Style連結

.ts檔

```
export class HomePage {  
  isSpecial = true;  
}
```

.html檔

```
<div [style.color]="isSpecial ? 'red':'green'"></div>
```

Angular的數據綁定-事件(1)

事件綁定(Event Binding)

在HTML元件中的屬性加入：`(event)="functionName()"`

.ts檔

```
export class HomePage {  
  buttonClicked(){  
    alert('clicked');  
  }  
}
```

.html檔

```
<ion-button (click)="buttonClicked();">Click Me</ion-button>
```

Angular的數據綁定-事件(2)

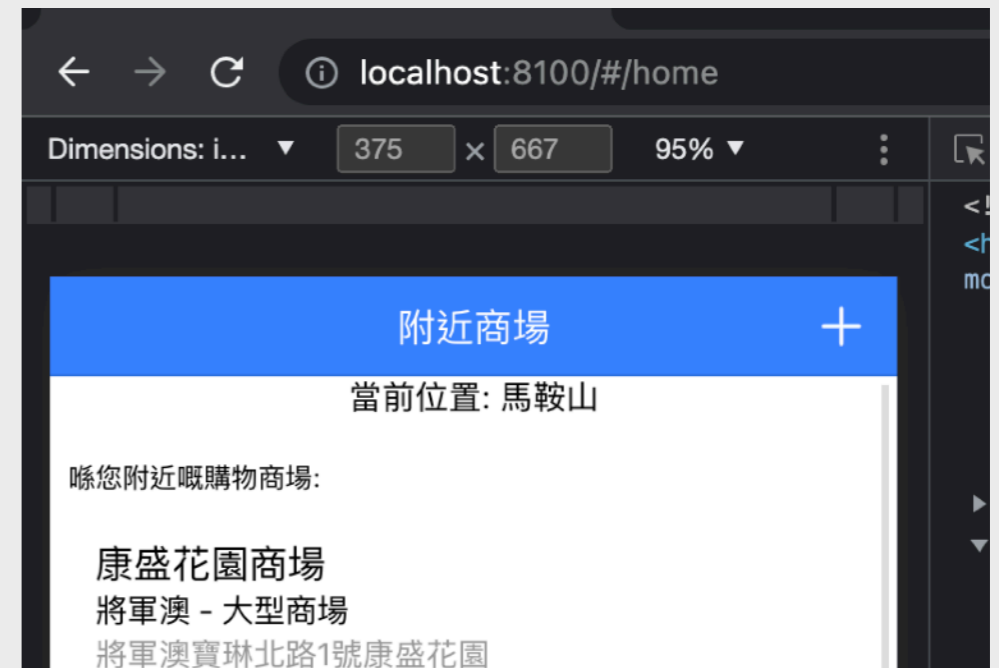
Angular Event (事件) 種類

事件	表達	例子	HTML元件事件觸發時機
Click 點擊	(click)	(click)="eleClicked()"	元件被點擊時觸發事件
Double Click 雙點擊	(dblclick)	(dblclick)="eleDbClicked()"	元件被雙點擊時觸發事件
Submit 被送出	(submit)	(submit)="formSubmitted()"	應用到Form(表單)元件，當表單被送出時觸發事件
Blur 被失焦時	(blur)	(blur)="inputBlurred()"	應用到Input(輸入)元件，當輸入被失焦時觸發事件
Focus 被聚焦時	(focus)	(focus)="inputFocused()"	應用到Input(輸入)元件，當輸入被聚焦時觸發事件
Scroll 被捲動時	(scroll)	(scroll)="eleScrolled()"	元件被捲動時觸發事件
Key Up 放鍵時	(keyup)	(keyup)="inputKeyUpped()"	應用到Input(輸入)元件，當鍵盤的鍵被放開時的那刻時觸發事件
Key Press 完成按鍵時	(keypress)	(keypress)="inputKeyPressed()"	應用到Input(輸入)元件，當鍵盤中可以輸出文字符號的按鍵被完成按鍵時 (按下並放開)
Key Down 按鍵時	(keydown)	(keydown)="inputKeyDowned()"	應用到Input(輸入)元件，鍵盤的鍵被按下時的那刻

項目實習-附近商場APP-事件綁定(1)

首先我們在 `home.page.html` 中換上以下的 `header` :

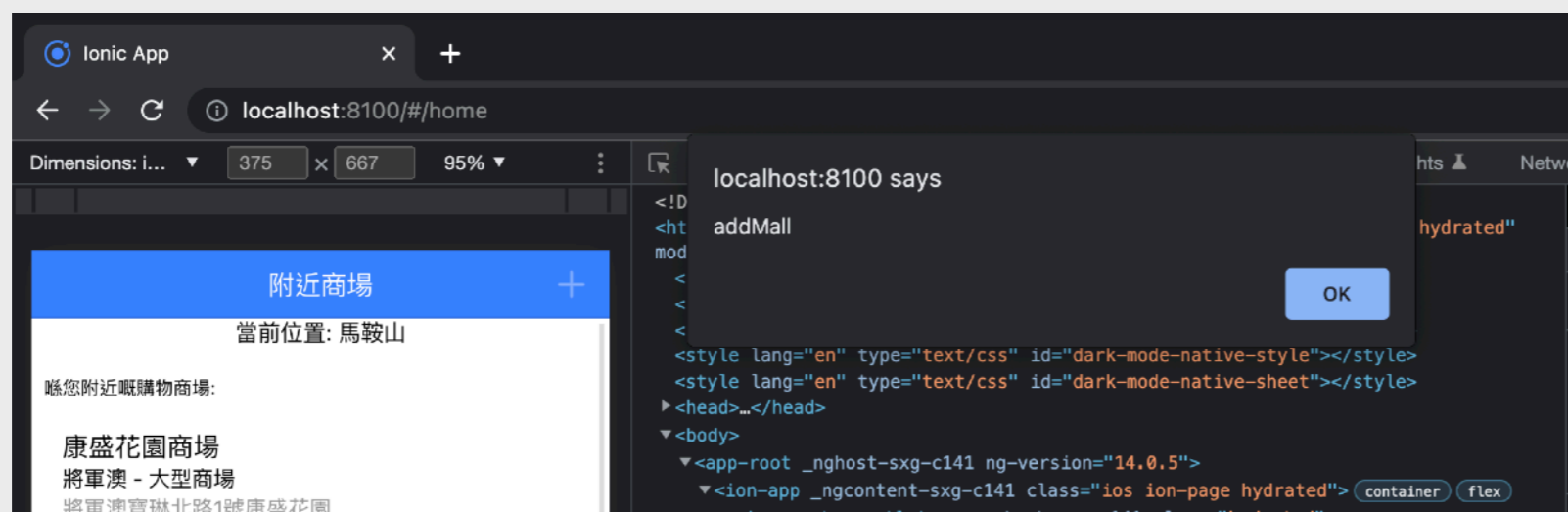
```
<ion-header>
  <ion-toolbar color="primary">
    <ion-title>附近商場</ion-title>
    <ion-buttons slot="end">
      <ion-button>
        <ion-icon slot="icon-only" name="add"></ion-icon>
      </ion-button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>
```



完成後的結果

然後我們在 `<ion-button>` 裡加入 `click` 事件綁定: `<ion-button (click)="addMall();">`

再到 `home.page.ts` 檔裡加入 `addMall()` 函式到 `HomePage` 的 `Class` 內:



```
addMall(){
  alert('addMall');
}
```

完成後當按下 + 號的結果

項目實習-附近商場APP-事件綁定(2)

然後我們在 `home.page.html` 也為 `ion-item` 加上 `click` 事件：

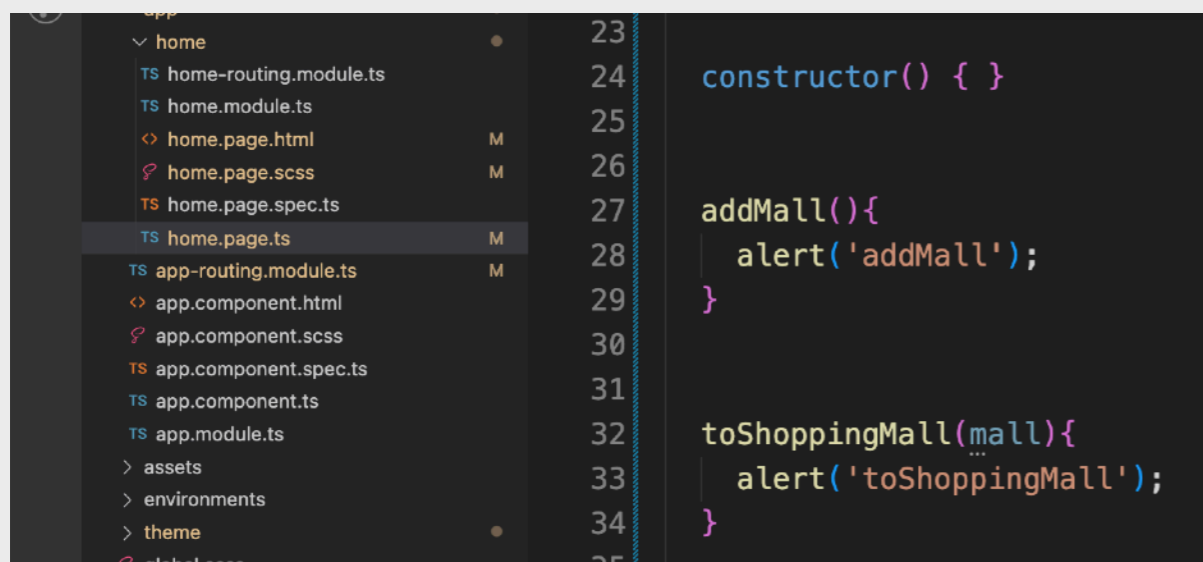
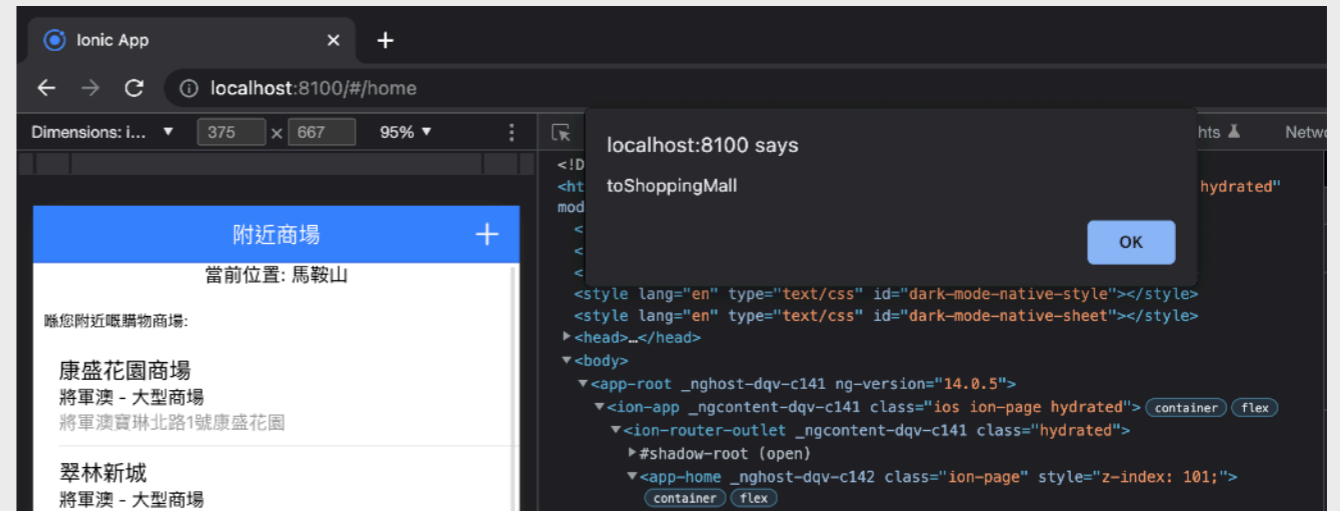
```
<ion-item *ngFor="let m of displayShoppingMalls" (click)="toShoppingMall(m);">
```

"noImplicitAny": false

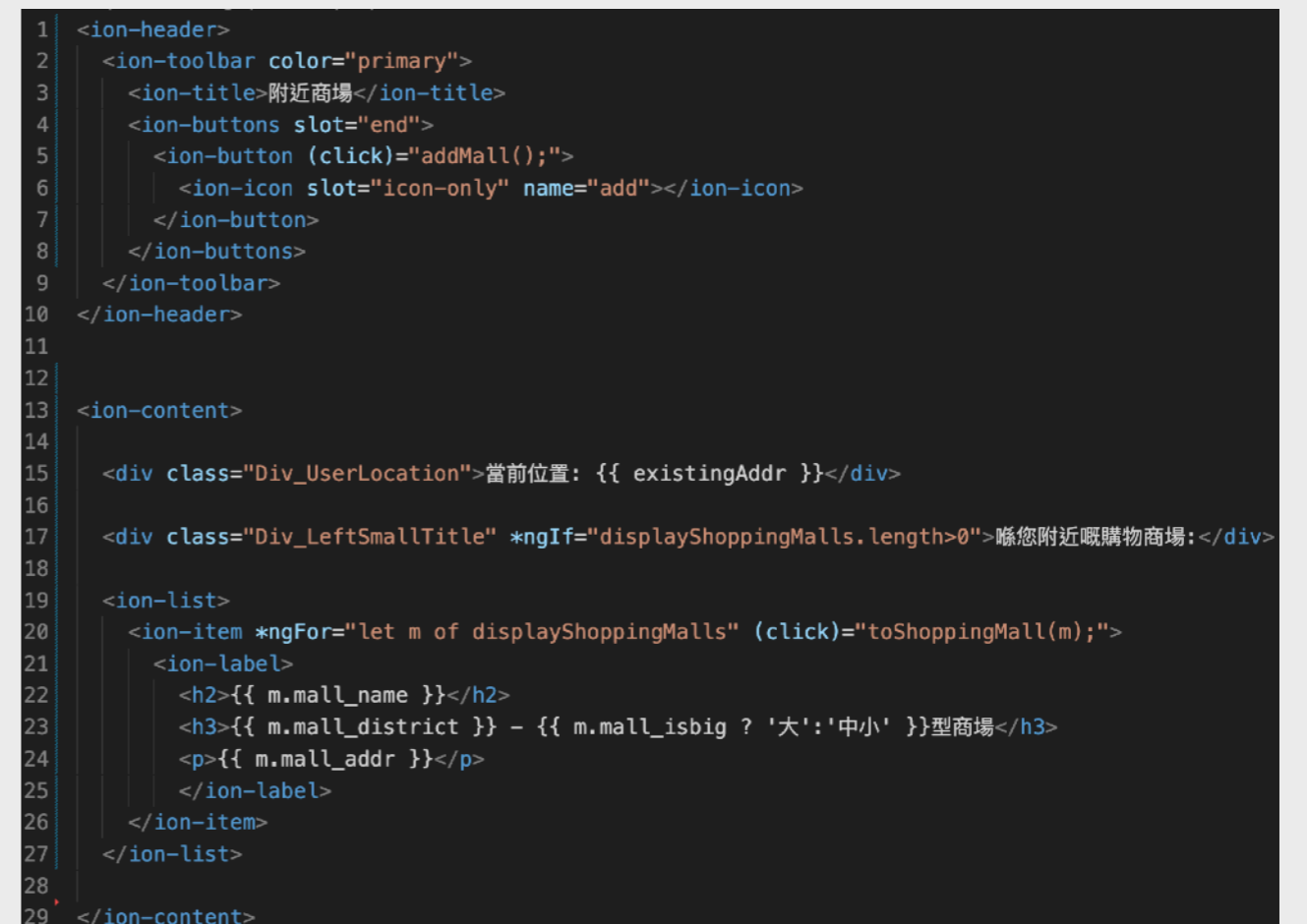
也同樣到TS檔加上 `toShoppingMall` 函式：

```
toShoppingMall(mall){  
  alert('toShoppingMall');  
}
```

當按下其中一個商場時，都會彈出提示框



HTML檔與TS檔此時的版面



Angular的數據綁定-雙向

雙向綁定(Two-way Data Binding)

在HTML元件中的屬性加入：`[(ngModel)]="value"`

雙向綁定只可用於可以讓使用者互動的 HTML 元素，如：

`<input>`、`<select>`、`<textarea>`等

當HTML輸入元素的數值發生變化時，TypeScript裡的變數也會被更被，反之亦然；使用前需先在.ts檔裡的頂端載入(import) FormsModule

.ts檔

```
export class HomePage {  
  loginName = 'eric';  
}
```

.html檔

```
<ion-input [(ngModel)]="loginName"></ion-input>
```

新增商場

商場名稱
請輸入商場名稱

所屬區域
請輸入商場所屬區域

緯度
0

經度
0

地址
請輸入商場地址

類型
大型商場

目前狀態
正常

確定

Angular換頁機制

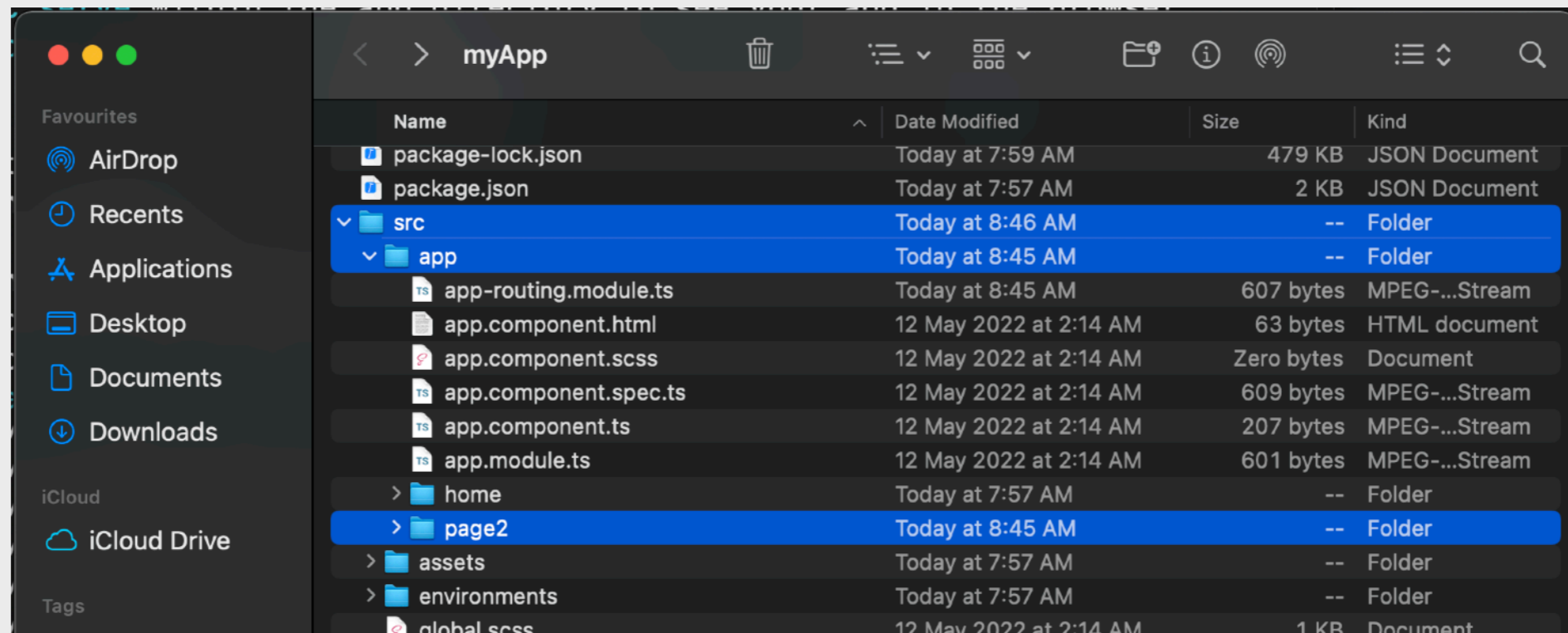
1. 到「app-routing.module.ts」檢查「routes」陣列中有沒有你所要跳進的頁面的路徑資訊。正常來說，頁面的路徑資訊會在終端機執行「ionic generate page ...」時自動被加進這個儲存路徑資訊的檔案
2. 如路徑資訊有誤或未被加進「app-routing.module.ts」，便需手動加入類似這樣的JSON：
`{path:'pagepath', loadChildren: './page2/page2.module#Page2Module'}`
(注意需修改JSON中的資訊以迎合你的頁面路徑)
3. 到需要觸發換頁的「.page.ts」，在其頂端載入「Router」即：
`import { Router } from '@angular/router';`
4. 然後在其「constructor」中定義「Router」物件如：
`constructor(public router:Router){ /* ... */ }`
5. 然後在其頁中的換頁觸發部分中加入換頁程式如：
`this.router.navigateByUrl('page2');`
(注意需修改路徑參數)
6. 如果要在換頁時把文字參數傳進下一頁，需於「app-routes.ts」中的「path:'page2」新增斜號及參數名稱(可多於一個參數)如：`path:'page2/:name/:age'`
然後到會被換進的頁面的「.page.ts」裡的頂端載入「ActivatedRoute」即：
`import { ActivatedRoute } from '@angular/router';`
並於其「constructor」中定義「ActivatedRoute」物件如：`public activatedRoute:ActivatedRoute`
再加入拿取傳入參數的程式如：`this.userid=this.activatedRoute.snapshot paramMap.get('userid');`
(注意需修改參數的名稱，如果參數是數字資料，則需使用「Number()」或「parseInt()」包著它)

Angular新增頁面及換頁範例 (1)

我們到終端機執行開新頁面的指令(範例中把頁面稱之為「Page2」) : `ionic generate page Page2`

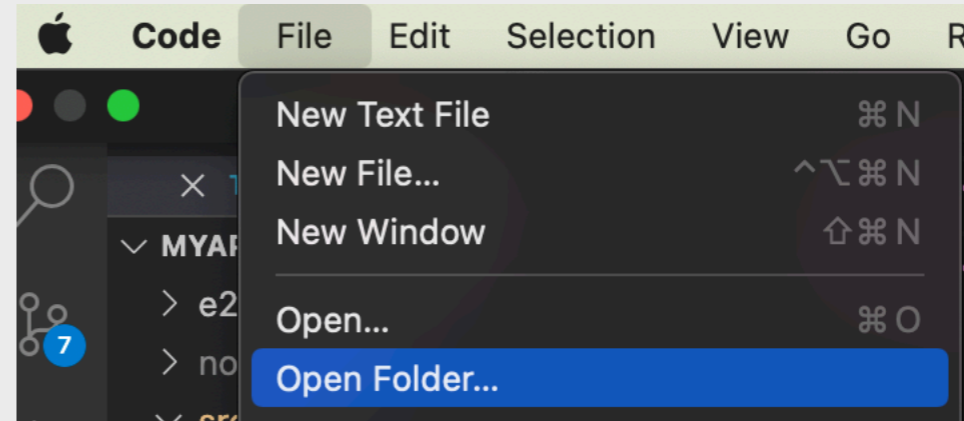
```
dave@MacBook-Air-M1 myApp % cd myApp
dave@MacBook-Air-M1 myApp % ionic generate page Page2
> ng generate page Page2 --project=app
```

「Page2」被創建了在「src/app」的目錄裡



Angular新增頁面及換頁範例 (2)

打開VS Code並使用「File -> Open Folder」
來開啓Ionic Angular項目

A screenshot of the Visual Studio Code editor showing the 'app-routing.module.ts' file. The file content is as follows:

```
1 import { NgModule } from '@angular/core';
2 import { PreloadAllModules, RouterModule, Routes } from '@angular/router';
3
4 const routes: Routes = [
5   {
6     path: 'home',
7     loadChildren: () => import('./home/home.module').then( m => m.HomePageModule ),
8   },
9   {
10    path: '',
11    redirectTo: 'home',
12    pathMatch: 'full'
13  },
14  {
15    path: 'page2',
16    loadChildren: () => import('./page2/page2.module').then( m => m.Page2Module )
17  },
18 ];
19
20 @NgModule({
21   imports: [
22     RouterModule.forRoot(routes, { preloadingStrategy: PreloadAllModules })
23   ],
24   exports: [RouterModule]
```

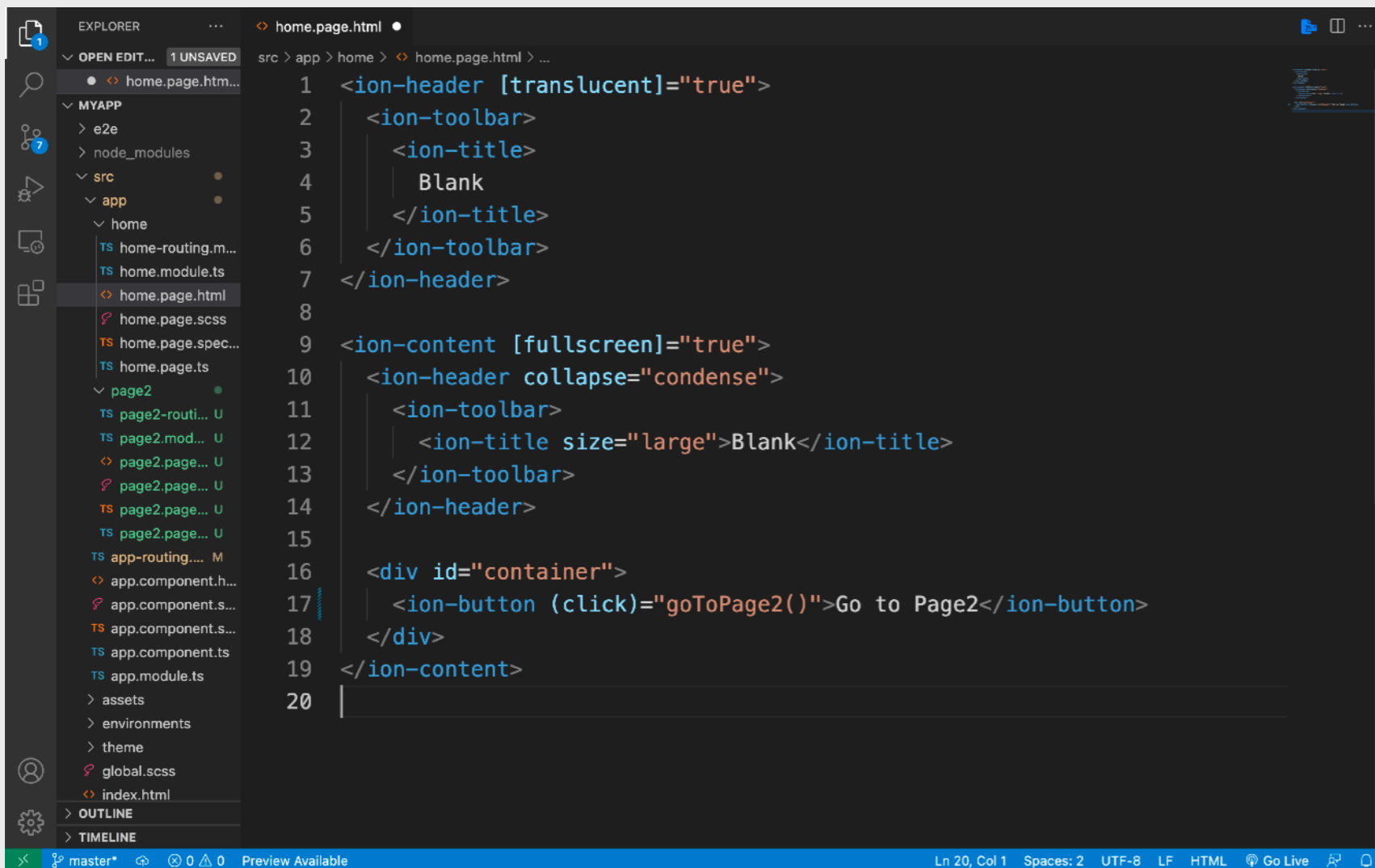
The Explorer sidebar on the left shows the project structure, with 'page2' folder expanded under 'src > app'. The status bar at the bottom indicates 'Ln 1, Col 1 Spaces: 2 UTF-8 LF TypeScript Go Live'.

到「app-routing.module.ts」會看見
「Page2」的路徑已被自動加進到
「routes」的JSON裡

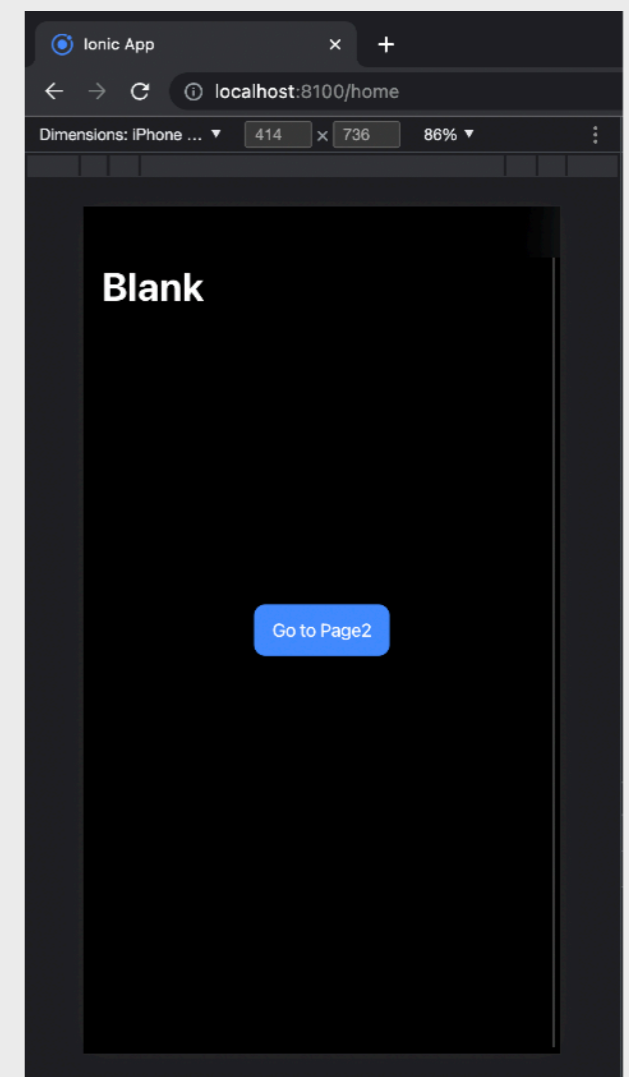
Angular新增頁面及換頁範例 (3)

到「Home」頁面的HTML檔加進「ion-button」並綁定點擊(click)事件為「goToPage2()」，然後到終端機執行指令以使Ionic運行：`ionic serve`

開啓瀏覽器並開啓
<http://localhost:8100>



```
1 <ion-header [translucent]="true">
2   <ion-toolbar>
3     <ion-title>
4       Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content [fullscreen]="true">
10  <ion-header collapse="condense">
11    <ion-toolbar>
12      <ion-title size="large">Blank</ion-title>
13    </ion-toolbar>
14  </ion-header>
15
16  <div id="container">
17    <ion-button (click)="goToPage2()">Go to Page2</ion-button>
18  </div>
19 </ion-content>
20
```



Angular新增頁面及換頁範例 (4)

到「Home」頁面的TS檔加進「import」語句：

```
import { Router } from '@angular/router';
```

並到「constructor」定義「Router」：`public router:Router`

再在「HomePage」的「Class」內編寫「goToPage2()」函式，

函式內寫下：`this.router.navigateByUrl('page2');`

```
src > app > home > TS home.page.ts > HomePage > goToPage2
1  import { Component } from '@angular/core';
2  import { Router } from '@angular/router';
3
4  @Component({
5    selector: 'app-home',
6    templateUrl: 'home.page.html',
7    styleUrls: ['home.page.scss'],
8  })
9  export class HomePage {
10
11    constructor(public router:Router) {}
12
13    goToPage2(){
14      this.router.navigateByUrl('page2');
15    }
16
17 }
```

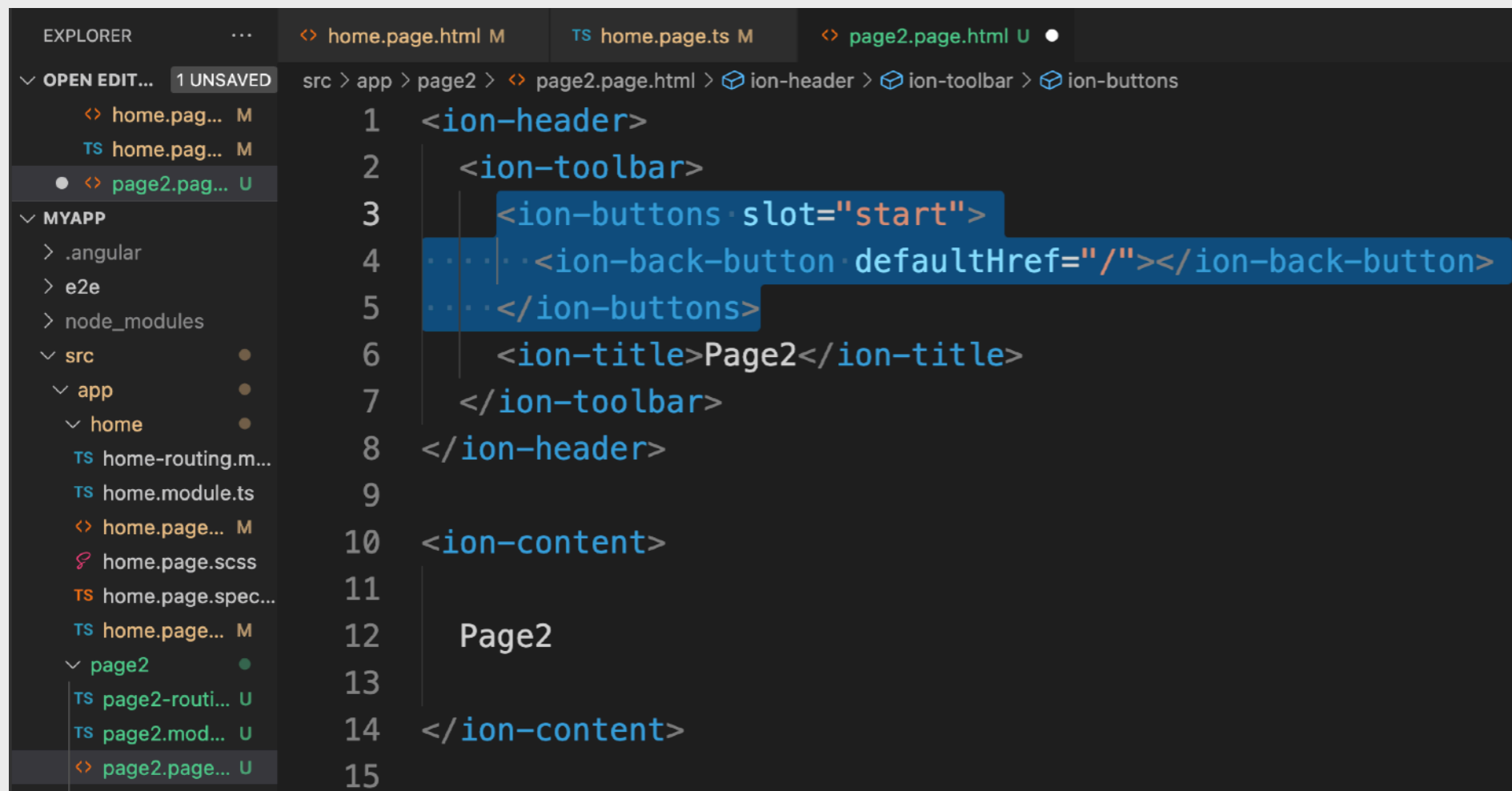
Angular新增頁面及換頁範例 (5)

到「Page2」頁面的HTML檔裡加進返回鍵(Back Button)：

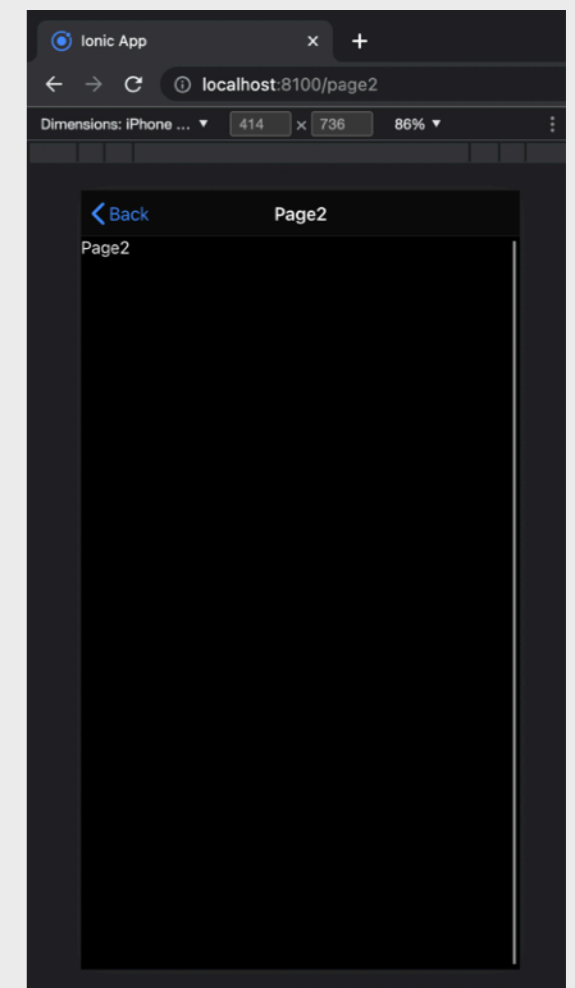
```
<ion-buttons slot="start">  
  <ion-back-button defaultHref="/"></ion-back-button>  
</ion-buttons>
```

也可在頁面內容「ion-content」內加上「Page2」字樣

儲存檔案後到瀏覽器查看，
換頁效果大功告成！



```
1 <ion-header>  
2   <ion-toolbar>  
3     <ion-buttons slot="start">  
4       <ion-back-button defaultHref="/"></ion-back-button>  
5     </ion-buttons>  
6     <ion-title>Page2</ion-title>  
7   </ion-toolbar>  
8 </ion-header>  
9  
10 <ion-content>  
11  
12   Page2  
13 </ion-content>  
14  
15
```



項目實習-附近商場APP-新增頁面(1)

首先，我們到終端機在鍵盤按下 **Ctrl + C** 以停止 Ionic 程式運作。

然後輸入以下指令以新增 AddMall 頁面：

```
ionic generate page AddMall
```

再輸入以下指令以新增 Mall 頁面：

```
ionic generate page Mall
```

```
[ng]
[ng] Build at: 2022-07-13T04:14:52.927Z - Hash: ffa6737e36912ffc - Time: 293ms
[ng] ✓ Compiled successfully.
^C
Daves-MacBook-Pro:ShoppingMalls dave$ ionic generate page AddMall
> ng generate page AddMall --project=app
The 'path' option in '/Users/dave/Desktop/Workspace/ionic-lab/ShoppingMalls/node_modules/@ionic/angular-toolkit/page/schema.json' is using deprecated behaviour. 'workingDirectory' smart default provider should be used instead.
CREATE src/app/add-mall/add-mall-routing.module.ts (352 bytes)
CREATE src/app/add-mall/add-mall.module.ts (481 bytes)
CREATE src/app/add-mall/add-mall.page.scss (0 bytes)
CREATE src/app/add-mall/add-mall.page.html (126 bytes)
CREATE src/app/add-mall/add-mall.page.spec.ts (669 bytes)
CREATE src/app/add-mall/add-mall.page.ts (263 bytes)
UPDATE src/app/app-routing.module.ts (632 bytes)
[OK] Generated page!

-----

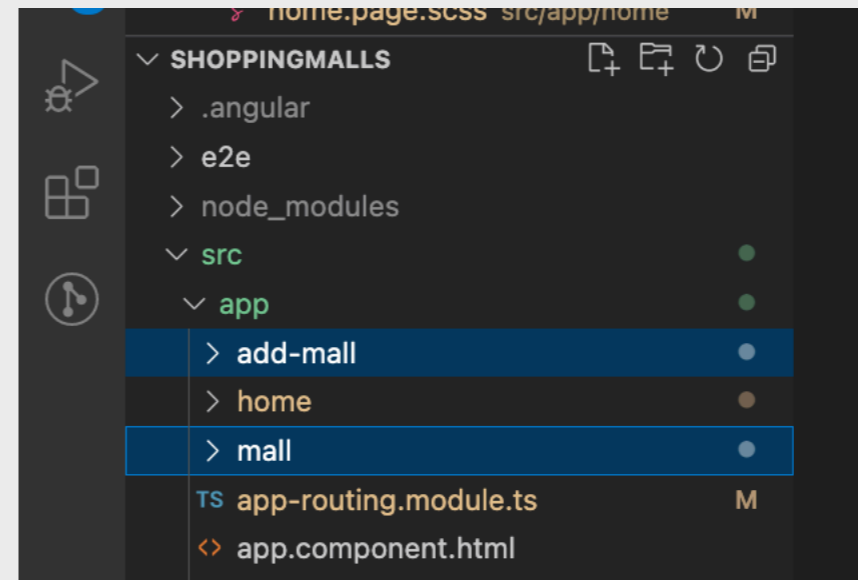
Ionic CLI update available: 6.5.0 → 6.20.1
Run npm i -g @ionic/cli to update

-----

Daves-MacBook-Pro:ShoppingMalls dave$ ionic generate page Mall
> ng generate page Mall --project=app
The 'path' option in '/Users/dave/Desktop/Workspace/ionic-lab/ShoppingMalls/node_modules/@ionic/angular-toolkit/page/schema.json' is using deprecated behaviour. 'workingDirectory' smart default provider should be used instead.
CREATE src/app/mall/mall-routing.module.ts (339 bytes)
CREATE src/app/mall/mall.module.ts (458 bytes)
CREATE src/app/mall/mall.page.scss (0 bytes)
CREATE src/app/mall/mall.page.html (123 bytes)
CREATE src/app/mall/mall.page.spec.ts (647 bytes)
CREATE src/app/mall/mall.page.ts (248 bytes)
UPDATE src/app/app-routing.module.ts (741 bytes)
[OK] Generated page!
Daves-MacBook-Pro:ShoppingMalls dave$ █
```

項目實習-附近商場APP-新增頁面(2)

然後我們會看見兩個頁面都已經成功版創建了到 Ionic 項目裡。



```
<ion-header>
  <ion-toolbar color="primary">
    <ion-buttons slot="start">
      <ion-back-button defaultHref="/"></ion-back-button>
    </ion-buttons>
    <ion-title>新增商場</ion-title>
  </ion-toolbar>
</ion-header>
```

然後到 [add-mall.page.html](#) 把 ion-header 換成左邊的 HTML 碼。

項目實習-附近商場APP-新增頁面(3)

然後再輸入 `ionic serve` 指令到終端機以運行 Ionic App。

再到「Home」頁面的TS檔加進「import」語句：

```
import { Router } from '@angular/router';
```

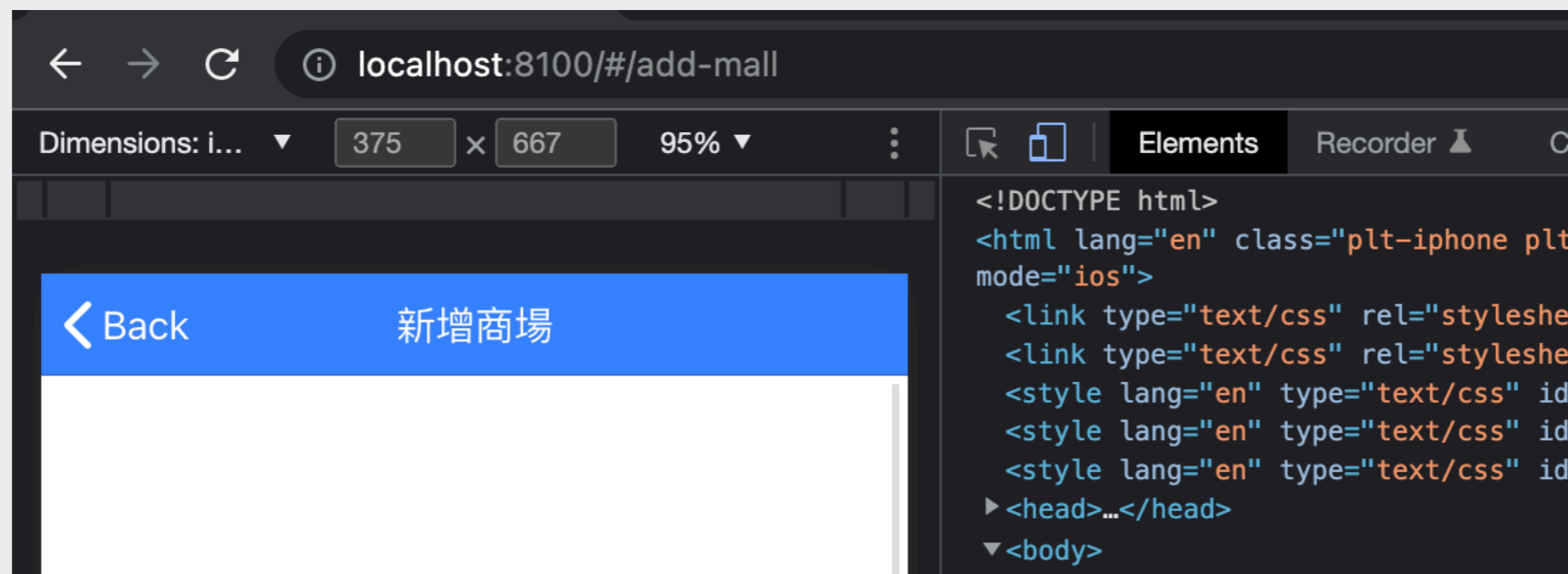
並到「constructor」定義「Router」：`public router:Router`

再在「HomePage」的「Class」內編寫「`addMall()`」函式，

函式內寫下：`this.router.navigateByUrl('add-mall');`

```
1 import { Component } from '@angular/core';
2 import { Router } from '@angular/router';
3
```

```
25 constructor(
26   public router:Router,
27 ){}
28
29
30 addMall(){
31   this.router.navigateByUrl('add-mall');
32 }
```



我們可看見當按下右上角 + 號時，便跳頁到「新增商場」頁

項目實習-附近商場APP-雙向數據綁定(1)

```
<ion-list>
  <ion-item>
    <ion-label position="stacked">商場名稱</ion-label>
    <ion-input [(ngModel)]="mallName" placeholder="請輸入商場名稱"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label position="stacked">所屬區域</ion-label>
    <ion-input [(ngModel)]="mallDistrict" placeholder="請輸入商場所屬區域"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label position="stacked">緯度</ion-label>
    <ion-input [(ngModel)]="mallLat" type="number" placeholder="請輸入商場緯度"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label position="stacked">經度</ion-label>
    <ion-input [(ngModel)]="mallLon" type="number" placeholder="請輸入商場經度"></ion-input>
  </ion-item>
  <ion-item>
    <ion-label position="stacked">地址</ion-label>
    <ion-textarea [(ngModel)]="mallAddr" placeholder="請輸入商場地址"></ion-textarea>
  </ion-item>
  <ion-item>
    <ion-label position="stacked">類型</ion-label>
    <ion-select [(ngModel)]="mallType" interface="popover">
      <ion-select-option value="big">大型商場</ion-select-option>
      <ion-select-option value="middlesmall">中小型商場</ion-select-option>
    </ion-select>
  </ion-item>
  <ion-item>
    <ion-label position="stacked">目前狀態</ion-label>
    <ion-select [(ngModel)]="mallStatus">
      <ion-select-option *ngFor="let ms of mallStatusList" [value]="ms.value">{{ ms.displayname }}</ion-select-option>
    </ion-select>
  </ion-item>
</ion-list>
```

在 `add-mall.page.html` 中的 `ion-content` 內加入 `ion-list`，而 list 中加入各項 `ion-input`，並加上 `[(ngModel)]` 以連結TS檔中的變數。

而 `ngModel` 的特色就是雙向綁定，若用戶輸入了或改變了 `input` 的資料，TS檔內的變數便會隨之而更新，反之，若TS檔內的變數更改了，`input` 的資料便會被自己更新。

項目實習-附近商場APP-雙向數據綁定(2)

然後到 `add-mall.page.ts` 裡

加進下方的變數：

```
mallName="";
mallDistrict="";
mallLat=0;
mallLon=0;
mallAddr="";
mallType='big';
mallStatus='normal';
gettingCurrentLocation=false;
mallStatusList=[
  {displayname:"正常",value:"normal"},
  {displayname:"興建中",value:"preparing"},
  {displayname:"已遷移",value:"moved"},
  {displayname:"暫時關閉",value:"tmp-closed"},
  {displayname:"已關閉",value:"closed"}
];
```

```
8 export class AddMallPage implements OnInit {
9
10   mallName='';
11   mallDistrict='';
12   mallLat=0;
13   mallLon=0;
14   mallAddr='';
15   mallType='big';
16   mallStatus='normal';
17   gettingCurrentLocation=false;
18   mallStatusList=[
19     {displayname:"正常",value:"normal"},
20     {displayname:"興建中",value:"preparing"},
21     {displayname:"已遷移",value:"moved"},
22     {displayname:"暫時關閉",value:"tmp-closed"},
23     {displayname:"已關閉",value:"closed"}
24   ];
25
26   constructor() { }
```

項目實習-附近商場APP-雙向數據綁定(3)

然後到 [add-mall.page.html](#) 底下加進下方的HTML碼：

```
<br>
<ion-button shape="round" expand="block" (click)="addMall();">確定</ion-button>
<ion-button shape="round" expand="block" size="small"
fill="outline" (click)="getCurrentLocation();">
  <ion-icon name="location-outline"></ion-icon>
  獲取當前位置
</ion-button>

<ng-container *ngIf="gettingCurrentLocation">
  <br>
  <div style="text-align:center;font-size:10pt;">定位中...</div>
</ng-container>
```

項目實習-附近商場APP-雙向數據綁定(4)

然後到 `add-mall.page.ts` 裡加進下方的函式：

```
getCurrentLocation(){
  this.gettingCurrentLocation=true;
  navigator.geolocation.getCurrentPosition(position=>{
    this.mallLat=position.coords.latitude;
    this.mallLon=position.coords.longitude;
    this.gettingCurrentLocation=false;
  });
}
```

```
addMall(){
  const data={
    mall_name: this.mallName,
    mall_district: this.mallDistrict,
    mall_lat: this.mallLat,
    mall_lon: this.mallLon,
    mall_addr: this.mallAddr,
    mall_isbig: this.mallType==='big',
    mall_status: this.mallStatus
  };
  console.log('add_mall',data);
}
```



完成後的結果

項目實習-附近商場APP-雙向數據綁定(5)

The screenshot shows two parts of the application. On the left is a mobile app form titled '新增商場' (Add Mall) with a 'Back' button. The form fields are: '商場名稱' (Mall Name) with value '康熙皇商場', '所屬區域' (District) with value '西貢', and '緯度' (Latitude) with value '22.4228598'. On the right is the browser's developer console showing the following log output:

```
[webpack-dev-server] Live Reloading enabled.
Angular is running in development mode. Call enableProdMode() to enable production mode.
add_mall
{mall_name: '康熙皇商場', mall_district: '西貢', mall_lat: 22.4228598, mall_lon: 114.2307659, mall_addr: '西灣河新運路1號', ...}
```

完成後，我們會發現，若果點按App下方的「獲取當前位置」按鈕，畫面上的緯度和經度便會自動更新，這就是因為經緯度的 input 連結上了TS檔裡的 `mallLat` 和 `mallLon` 變數，當TS檔裡更新了它們時，畫面中的經緯度 input 並會隨之而自動更新。

最後，當我們按下「確定」按鈕以送出表單時，因我們在 `addMall` 函式中寫下了 `console.log` 出我們所定義了的變數，在 log 中，我們可以看到，TS檔裡的變數已被自動更新了為我們在畫面中所輸入的資料。這個就是 Angular 「雙向數據綁定」的威力了！

This screenshot shows the same '新增商場' form after the '獲取當前位置' (Get Current Location) button has been pressed. The '緯度' (Latitude) field is now '22.4228598' and the '經度' (Longitude) field is '114.2307659'. The '地址' (Address) field is '西灣河新運路1號'. The '類型' (Type) dropdown is set to '中小型商場' and the '目前狀態' (Current Status) dropdown is set to '正常'. At the bottom, there is a blue '確定' (Confirm) button and a white button with a location icon and the text '獲取當前位置'.

項目實習-附近商場APP-頁面附參數轉跳(1)

然後我們便製作 Mall 頁面。因為我們需要傳遞商場的名稱、緯度和經度到 Mall 頁面才能夠在 Mall 頁面的標題顯示商場名稱以及在地圖中顯示出商場的位置，所以我們在跳頁時並要傳遞這三個變數(參數)到 Mall 頁面。

首先，我們到 `src/app/app-routing.module.ts` 裡修改 Mall 頁的 path 並定義三個參數為：`title`、`lat` 與 `lon`。即 `path: 'mall/:title/:lat/:lon'`，如下圖：

```
17     AddMallPageModule)
18     },
19     {
20       path: 'mall/:title/:lat/:lon',
21       loadChildren: () => import('./mall/mall.module').then( m => m.MallPageModule)
22     },
23   ];
```

項目實習-附近商場APP-頁面附參數轉跳(2)

然後到會被換進的頁面的 `mall.page.ts` 裡的頂端載入 `ActivatedRoute` 即：

```
import { ActivatedRoute } from '@angular/router';
```

並於其 `constructor` 中定義 `ActivatedRoute` 物件即：

```
public route:ActivatedRoute
```

接著定義商場名稱(建議賦初始值為「地圖」)、緯度和經度，即：

```
pageTitle='地圖';
```

```
lat=0;
```

```
lon=0;
```

再加入拿取傳入參數的程式即：

```
this.pageTitle=this.route.snapshot.paramMap.get('title');
```

```
this.lat=Number(this.route.snapshot.paramMap.get('lat'));
```

```
this.lon=Number(this.route.snapshot.paramMap.get('lon'));
```

因為緯度和經度參數是數字資料，所以需使用 `Number()` 或 `parseInt()` 包著

實際做法請參考下頁之圖示

項目實習-附近商場APP-頁面附參數轉跳(3)

```
... TS app-routing.module.ts M TS mall.page.ts U TS home.page.ts M <> add-mall.page.html U TS add-mall.page.ts U
src > app > mall > TS mall.page.ts > MallPage
1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute } from '@angular/router';
3
4 @Component({
5   selector: 'app-mall',
6   templateUrl: './mall.page.html',
7   styleUrls: ['./mall.page.scss'],
8 })
9 export class MallPage implements OnInit {
10
11   pageTitle='地圖';
12   lat=0;
13   lon=0;
14
15   constructor(public route:ActivatedRoute){
16     this.pageTitle=this.route.snapshot.paramMap.get('title');
17     this.lat=Number(this.route.snapshot.paramMap.get('lat'));
18     this.lon=Number(this.route.snapshot.paramMap.get('lon'));
19   }
20
21   ngOnInit(){}
22
23 }
```

完成後的 mall.page.ts 程式碼

項目實習-附近商場APP-頁面附參數轉跳(4)

最後到 `home.page.ts` 修改其 `toShoppingMall` 函式為：

```
this.router.navigateByUrl('mall/'+mall.mall_name+'/'+mall.mall_lat+'/'+mall.mall_lon);
```

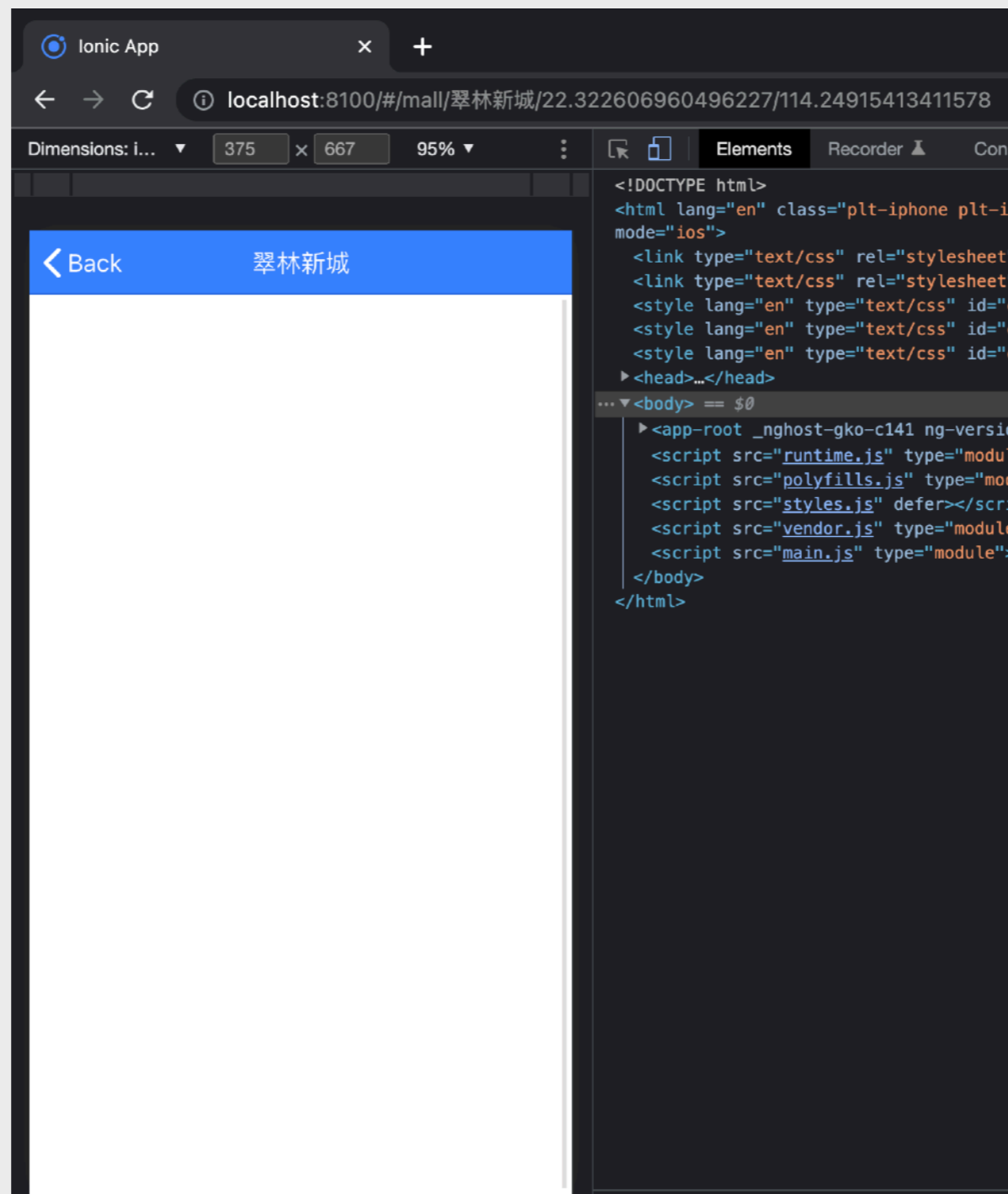
```
35 | toShoppingMall(mall){  
36 |     this.router.navigateByUrl('mall/'+mall.mall_name+'/'+mall.mall_lat+'/'+mall.mall_lon);  
37 | }
```

```
<ion-header>  
  <ion-toolbar color="primary">  
    <ion-buttons slot="start">  
      <ion-back-button defaultHref="/"></ion-back-button>  
    </ion-buttons>  
    <ion-title>{{ pageTitle }}</ion-title>  
  </ion-toolbar>  
</ion-header>
```

然後到 `mall.page.html` 把 `ion-header` 換成左邊的 HTML 碼。

項目實習-附近商場APP-頁面附參數轉跳(5)

完成後，便會看見在主頁的商場列表中按下商場名稱便成功跳頁，而瀏覽器的網址尾端則寫著商場名稱、緯度和經度，而 App 的左上角亦顯示了 Back 按鈕。



課題八： NPM第三方插件應用

項目實習-附近商場APP-加入NPM插件

到重要的重頭戲了！就是加入地圖來顯示商場的位置。在這裡，我們便需要用上了基於 Node.js 而開發的第三方插件，我們需要透過 **NPM** (Node Package Manager) 來下載及安裝插件。是次我們並用上了 **Leaflet** (<https://www.npmjs.com/package/leaflet>)，它是一個免費開源的 JavaScript 程式庫讓開發者加入地圖到流動裝置應用程式。

首先我們先終止運行 Ionic App (於終端機中以鍵盤按下 **Ctrl + C**)，然後輸入：

```
npm i leaflet
```

以下載及安裝 Leaflet 插件。

```
Daves-MacBook-Pro:ShoppingMalls dave$ npm i leaflet
added 1 package, removed 1 package, and audited 1305 packages in 2s

164 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
Daves-MacBook-Pro:ShoppingMalls dave$
```

因 Leaflet 需要我們加入其 CSS 檔到 index.html 的 head 元件中，所以我們便加入：

```
<link rel="stylesheet" href="https://unpkg.com/leaflet@1.8.0/dist/leaflet.css"/>
```

到 index.html 的 head 元件中如下頁之圖示。

項目實習-附近商場APP-使用Leaflet地圖(1)

```
... TS app-routing.module.ts M TS mall.page.ts U TS home.page.ts M < mall.page.html U < index.html M • < add-mall.page.html U TS add-mall.page.ts U < home.page.html M home.page.scss M
src > < index.html > ...
M You, 7 seconds ago | 1 author (You)
U
M 1 <!DOCTYPE html>
U
M 2 <html lang="en">
M
M 3
U
U 4 <head>
U
M 5 <meta charset="utf-8" />
M
M 6 <title>Ionic App</title>
U
U 7
U
U 8 <base href="/" />
U
U 9
U
U 10 <meta name="color-scheme" content="light dark" />
M
M 11 <meta name="viewport" content="viewport-fit=cover, width=device-width, initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0,
U
U 12 <meta name="format-detection" content="telephone=no" />
M
M 13 <meta name="msapplication-tap-highlight" content="no" />
M
M 14
U
U 15 <link rel="icon" type="image/png" href="assets/icon/favicon.png" />
U
U 16
U
U 17 <!-- add to homescreen for ios -->
U
U 18 <meta name="apple-mobile-web-app-capable" content="yes" />
U
U 19 <meta name="apple-mobile-web-app-status-bar-style" content="black" />
M
M 20
U
U 21 <link rel="stylesheet" href="https://unpkg.com/leaflet@1.8.0/dist/leaflet.css"/>
U
U 22 </head>
M
M 23
U
U 24 <body>
U
U 25 <app-root></app-root>
U
U 26 </body>
M
M 27
U
U 28 </html>
```

放在 index.html 的 head 元件中

項目實習-附近商場APP-使用Leaflet地圖(2)

1. 因 Leaflet 要求我們加進一個 div 用來裝著它所提供的地圖顯示畫面，所以我們到 `mall.page.html` 的 `ion-content` 裡加入：

```
<div id="div_map"></div>
```

```
src > app > mall > <> mall.page.html > ...
1 <ion-header>
2   <ion-toolbar color="primary">
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>{{ pageTitle }}</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <div id="div_map"></div>
12
13
14 </ion-content>
```

2. 到 `mall.page.scss` 加進 `#div_map` 的 CSS：

```
#div_map{
  width: 100%;
  height: 100%;
}
```

```
src > app > mall > mall.page.scss > #div_r
1 #div_map{
2   width: 100%;
3   height: 100%;
4 }
```

項目實習-附近商場APP-使用Leaflet地圖(3)

最後到 `mall.page.ts` 加上 `loadmap()` 函式，然後再加上 `ionViewDidEnter` 函式並於其中載入 `loadmap()` 函式。
`ionViewDidEnter` 函式為 Ionic 頁面生命週期的其中一個預設函式，會在頁面當已被開啓了的剎那間運行。

而在 `mall.page.ts` 頂端亦需要加上載入 Leaflet 的程式庫：

```
import leaflet from 'leaflet';
```

```
ionViewDidEnter(){
  this.loadmap();
}

loadmap(){
  const map=leaflet.map("div_map").fitWorld();
  leaflet.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attributions:'Map data © <a href="https://openstreetmap.org">OpenStreetMap</a> contributors',
    maxZoom:18, minZoom:8
  }).addTo(map);
  map.setView([this.lat,this.lon], 18);
  const markerGroup=leaflet.featureGroup();
  markerGroup.addLayer(leaflet.marker([this.lat,this.lon]));
  map.addLayer(markerGroup);
}
```

項目實習-附近商場APP-使用Leaflet地圖(4)

```
TS add-mall.page.ts U TS mall.page.ts U X home.page.html M home.page.scss M
src > app > mall > TS mall.page.ts > MallPage > ngOnInit
1 import { Component, OnInit } from '@angular/core';
2 import { ActivatedRoute } from '@angular/router';
3 import leaflet from 'leaflet';
4
5 @Component({
6   selector: 'app-mall',
7   templateUrl: './mall.page.html',
8   styleUrls: ['./mall.page.scss'],
9 })
10 export class MallPage implements OnInit {
11
12   pageTitle='地圖';
13   lat=0;
14   lon=0;
15
16   constructor(public route:ActivatedRoute){
17     this.pageTitle=this.route.snapshot.paramMap.get('title');
18     this.lat=Number(this.route.snapshot.paramMap.get('lat'));
19     this.lon=Number(this.route.snapshot.paramMap.get('lon'));
20   }
21
22   ngOnInit(){}
23
24   ionViewDidEnter(){
25     this.loadmap();
26   }
27
28   loadmap(){
29     const map=leaflet.map("div_map").fitWorld();
30     leaflet.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
31       attributions:'Map data © <a href="https://openstreetmap.org">OpenStreetMap</a> contributors',
32       maxZoom:18, minZoom:8
33     }).addTo(map);
34     map.setView([this.lat,this.lon], 18);
35     const markerGroup=leaflet.featureGroup();
36     markerGroup.addLayer(leaflet.marker([this.lat,this.lon]));
37     map.addLayer(markerGroup);
38   }
39
40 }
```

完成後的
mall.page.ts
檔程式碼

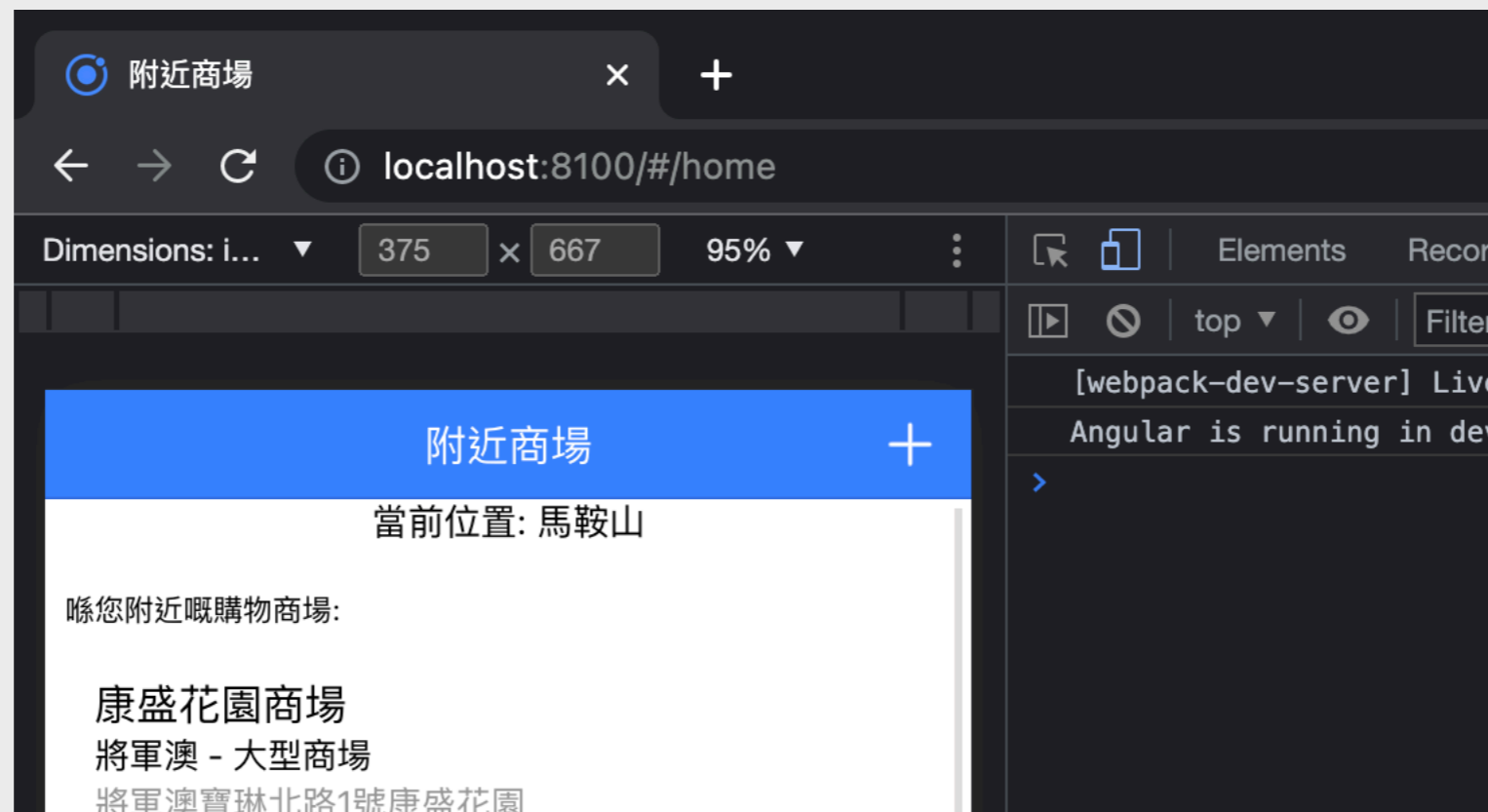
續課題六： Angular及Node.js與Ionic應用

項目實習-附近商場APP-瀏覽器標題

最後到 index.html 的 head 元件裡更改 title 元件的內容為「附近商場」。

```
TS add-mall.page.ts U  <> index.html M X  <> home.page.html
src > <> index.html > html > head > meta
    You, 3 minutes ago | 1 author (You)
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="utf-8" />
6      <title>附近商場</title>
```

於瀏覽器中手動刷新(Refresh)頁面，便可看見頁面標題已被更新了。



項目實習-附近商場APP-完成客戶端

恭喜！終於完成了客戶端頁面！


附近商場 +

當前位置: 馬鞍山

睇您附近嘅購物商場:

- 康盛花園商場**
將軍澳 - 大型商場
將軍澳寶琳北路1號康盛花園
- 翠林新城**
將軍澳 - 大型商場
將軍澳翠林邨翠琳路11號
- 慧安商場**
將軍澳 - 大型商場
將軍澳寶林毓雅里9號
- 慧星匯**
將軍澳 - 大型商場
將軍澳寶林毓雅里9號慧安商場一樓
- 寶林商場**
將軍澳 - 大型商場
將軍澳寶林寶林邨

← Back 慧星匯



Map showing the location of 慧星匯 (Well On Shopping Arcade) in Ma鞍山, surrounded by residential blocks and landmarks like Po Tai House, Po Ning House, Po Lam Estate, and Po Lam Market.

← Back 新增商場

商場名稱
請輸入商場名稱

所屬區域
請輸入商場所屬區域

緯度
0

經度
0

地址
請輸入商場地址

類型
大型商場

目前狀態
正常

確定

📍 獲取當前位置

Ionic Service (公用服務) (1)

當Ionic的項目做到有一定的規模時，我們便可能會出現一些很多頁面都需要呼叫執行的函式，或是多個頁面都需要共同存取的變數，這個時候，我們便可以使用到「Service」來給各個版面共同執行函式和存取變數。

到終端機執行指令以創建「Service」(在本範例中命名為「GlobalService」)：`ionic generate service GlobalService`
如Ionic程式正在運行中，我們則需先在終端機執行指令以停止運行Ionic：`control c`

```
[dave@MacBook-Air-M1 myApp % ionic generate service GlobalService
> ng generate service GlobalService --project=app
CREATE src/app/global-service.service.spec.ts (393 bytes)
CREATE src/app/global-service.service.ts (142 bytes)
[OK] Generated service!
dave@MacBook-Air-M1 myApp %
```

```
<> home.page.html M    TS home.page.ts M    <> page2.page.html U    TS page2.page.ts U    TS global-service.service.ts U x
src > app > TS global-service.service.ts > GlobalServiceService > sayHello
1  import { Injectable } from '@angular/core';
2
3  @Injectable({
4    providedIn: 'root'
5  })
6  export class GlobalServiceService {
7
8    constructor() { }
9
10 public sayHello(){
11   alert('Hello');
12 }
13 }
14
```

然後到剛創建好的「global-service.service.ts」撰寫「sayHello()」的函式：

```
public sayHello(){
  alert('Hello');
}
```

Ionic Service (公用服務) (2)

```
src > app > home > home.page.html > ion-content > div#container > br
1 <ion-header [translucent]="true">
2   <ion-toolbar>
3     <ion-title>
4       Blank
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content [fullscreen]="true">
10  <ion-header collapse="condense">
11    <ion-toolbar>
12      <ion-title size="large">Blank</ion-title>
13    </ion-toolbar>
14  </ion-header>
15
16  <div id="container">
17    <ion-button (click)="goToPage2()">Go to Page2</ion-button>
18    <br>
19    <ion-button (click)="callGlobalToSayHello()">Say Hello</ion-button>
20  </div>
21 </ion-content>
22
```

再到「home.page.html」加入按下會呼叫名為「callGlobalToSayHello()」的函式

同樣地，到「page2.page.html」加入按下會呼叫名為「callGlobalToSayHello()」的函式

```
src > app > page2 > page2.page.html > ion-content
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Page2</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11
12   Page2
13   <br>
14   <ion-button (click)="callGlobalToSayHello()">Say Hello</ion-button>
15
16 </ion-content>
```


Ionic Service (公用服務) (3)

然後在「home.page.ts」和「page2.page.ts」分別加入「import statement」即：

```
import { GlobalServiceService } from '../global-service.service';
```

再於「constructor」裡定義「globalService」即：

```
public globalService:GlobalServiceService
```

最後加上「callGlobalToSayHello」函式即：

```
callGlobalToSayHello(){this.globalService.sayHello();}
```

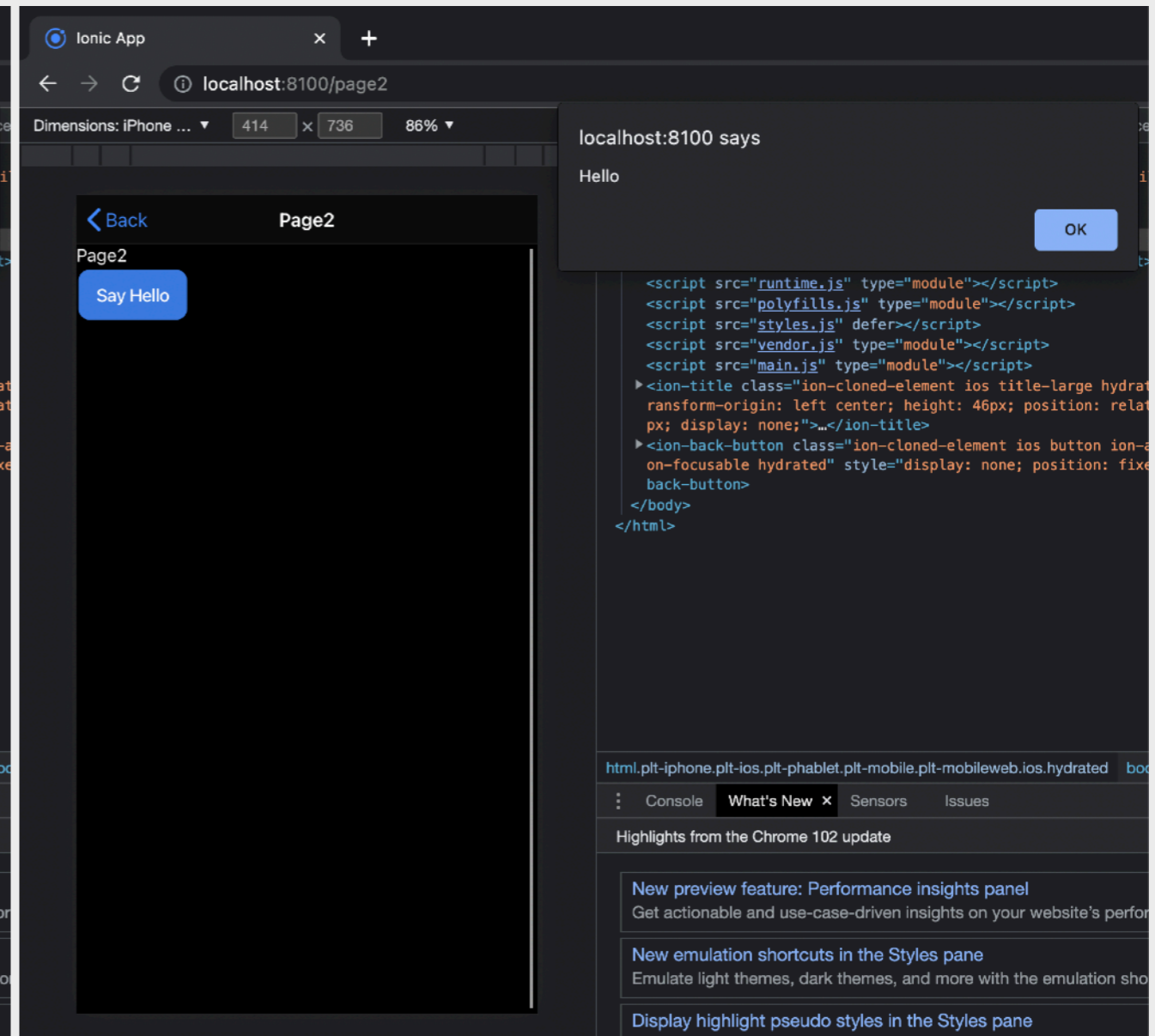
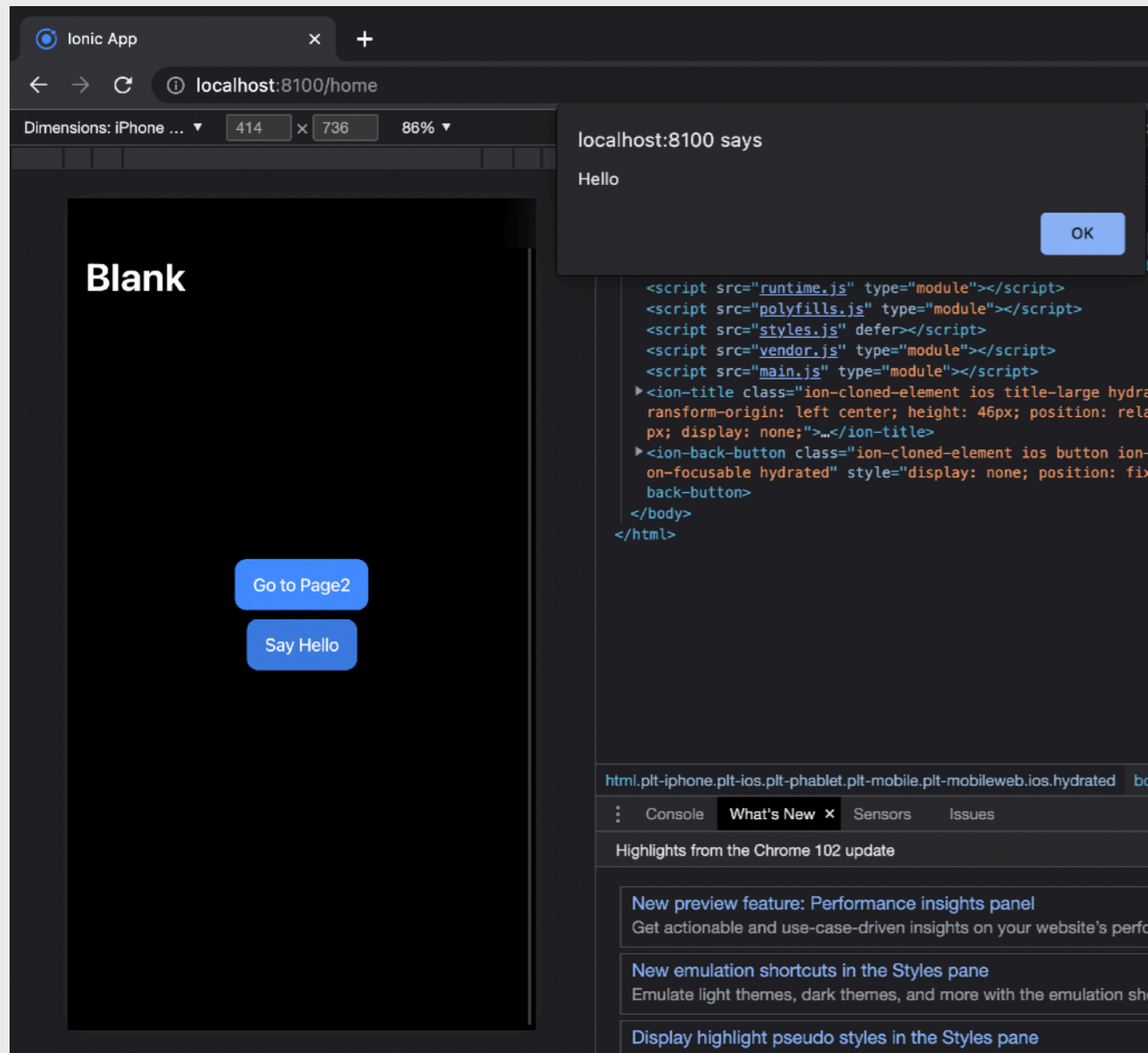
```
home.page.html M TS home.page.ts M X page2.page.html U TS page2.page.ts U TS global-service.service.ts U
c > app > home > TS home.page.ts > HomePage
1 import { Component } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { GlobalServiceService } from '../global-service.service';
4
5 @Component({
6   selector: 'app-home',
7   templateUrl: 'home.page.html',
8   styleUrls: ['home.page.scss'],
9 })
10 export class HomePage {
11
12   constructor(public router:Router, public globalService:GlobalServiceService) {}
13
14   goToPage2(){
15     this.router.navigateByUrl('page2');
16   }
17
18   callGlobalToSayHello(){
19     this.globalService.sayHello();
20   }
21
22 }
```

```
home.page.html M TS home.page.ts M page2.page.html U TS page2.page.ts U X TS global-service.service.ts U
c > app > page2 > TS page2.page.ts > Page2Page > constructor
1 import { Component, OnInit } from '@angular/core';
2 import { GlobalServiceService } from '../global-service.service';
3
4 @Component({
5   selector: 'app-page2',
6   templateUrl: './page2.page.html',
7   styleUrls: ['./page2.page.scss'],
8 })
9 export class Page2Page implements OnInit {
10
11   constructor(public globalService:GlobalServiceService) { }
12
13   ngOnInit() {
14   }
15
16   callGlobalToSayHello(){
17     this.globalService.sayHello();
18   }
19
20 }
```

Ionic Service (公用服務) (4)

最後，我們到終端機執行(如需)：`ionic serve`

當我們在「HomePage」或「Page2」按下「Say Hello」按鈕時，瀏覽器都會執行公用服務(即「GlobalService」)的函式「callGlobalToSayHello()」，即出現「Hello Alert」，大功告成！



Ajax應用範例

要與伺服器溝通以取得線上資料或發送資料到線上，我們便需要應用到「發出請求(Request)」並「接受回覆(Response)」，Ajax便可以滿足這樣的需求，在

```
ml M TS home.page.ts M <> page2.page.html U TS page2.page.ts U TS global-service.service.ts U TS app.module.ts M
> app > TS global-service.service.ts > ...
1 import { Injectable } from '@angular/core';
2
3 @Injectable({
4   providedIn: 'root'
5 })
6 export class GlobalServiceService {
7
8   constructor() { }
9
10  public sayHello(){
11    alert('Hello');
12  }
13
14  public async createAjax(url, method, data){
15    const response = await fetch(url, {
16      method, headers:{
17        'Accept':'application/json',
18        'Content-Type':'application/x-www-form-urlencoded;charset=UTF-8',
19      }, body:new URLSearchParams(data)
20    });
21    return response.ok ? response.json():null;
22  }
23 }
```

把Ajax的程式函式寫到TS檔案(如「公用服務」檔)：

```
public async createAjax(url, method, data){
  const response = await fetch(url, {
    method, headers:{
      'Accept':'application/json',
      'Content-Type':'application/x-www-form-urlencoded;charset=UTF-8',
    }, body:new URLSearchParams(data)
  });
  return response.ok ? response.json():null;
}
```

執行範例：

```
createAjax('xxx.php','POST',{userID:123})
.then(response=>{ /* ... */ });
```

課題九：

App警告元件、Modal及IonSlides範例

警示元件範例 - alertController (1)



①載入

```
import {AlertController} from '@ionic/angular';
```

②實體化

```
constructor(public alertController:AlertController){}
```

③調用

```
public showAlert(){
```

```
  this.alertController.create({ 只使用header或只使用  
    header: "恭喜您",          message或兩者都使用皆可以。
```

```
    message: "您中獎了！您是本程式的第10位用戶，因此贏  
    得了十萬港元特別獎賞，請填下您的網上銀行資料，我們會在  
    兩個工作天內存款到您的戶口。請相信我們不是騙徒！",
```

```
    buttons: [
```

```
      {text:"取消", role:"cancel"},
```

```
      {text:"領取"}  
    ]
```

```
  }
```

```
  }).then(a=>a.present());
```

```
}
```

可自由增加或減少buttons的數量，
如果是「取消」行為的話則可設定
role為cancel。

警示元件範例 - UIAlertController (2)

Tab 1

Show Alert

恭喜您

您中獎了！您是本程式的第10位用戶，因此贏得了十萬港元特別獎賞，請填下您的網上銀行資料，我們會在兩個工作天內存款到您的戶口。請相信我們不是騙徒！

請輸入戶口號碼

請輸入戶口密碼

請輸入驗證密碼 (如有)

取消

Tab 1

Show Alert

恭喜您

您中獎了！您是本程式的第10位用戶，因此贏得了十萬港元特別獎賞，請填下您的網上銀行資料，我們會在兩個工作天內存款到您的戶口。請相信我們不是騙徒！

12345678

.....

....

取消

領取



Tab 1



Tab 2



Tab 1



Tab 2



Tab 3

①載入 及 ②實體化

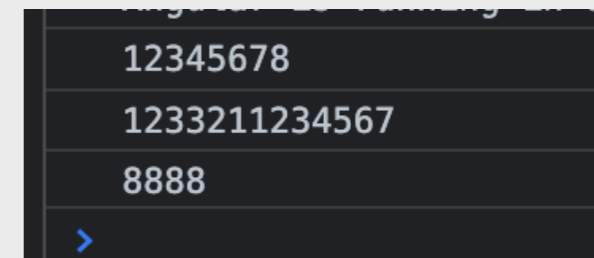
與【警示元件範例 - UIAlertController (1)】相同

③調用

```
public showAlert(){
    this.alertController.create({
        header: "恭喜您",
        message: "您中獎了！您是本程式的第10位用戶，因此贏得了十萬港元特別獎賞，請填下您的網上銀行資料，我們會在兩個工作天內存款到您的戶口。請相信我們不是騙徒！";
        inputs: [
            {name:"acno", type:"text", placeholder:"請輸入戶口號碼"},
            {name:"acpw1", type:"password", placeholder:"請輸入戶口密碼"},
            {name:"acpw2", type:"password", placeholder:"請輸入驗證密碼 (如有)"}
        ],
        buttons: [
            {text:"取消", role:"cancel"},
            {text:"領取", handler:alertData=>{
                console.log(alertData.acno);
                console.log(alertData.acpw1);
                console.log(alertData.acpw2);
            }}
        ]
    }).then(a=>a.present());
}
```

inputs的數量可自由增加或減少。當然Alert美觀上不該有太多的input吧。

handler為按下該button後要執行的事情。



警示元件範例 - ToastController

Tab 1

Show Toast

火星其實並沒有火

tab 1

tab 2

tab 3

①載入

```
import {ToastController} from '@ionic/angular';
```

②實體化

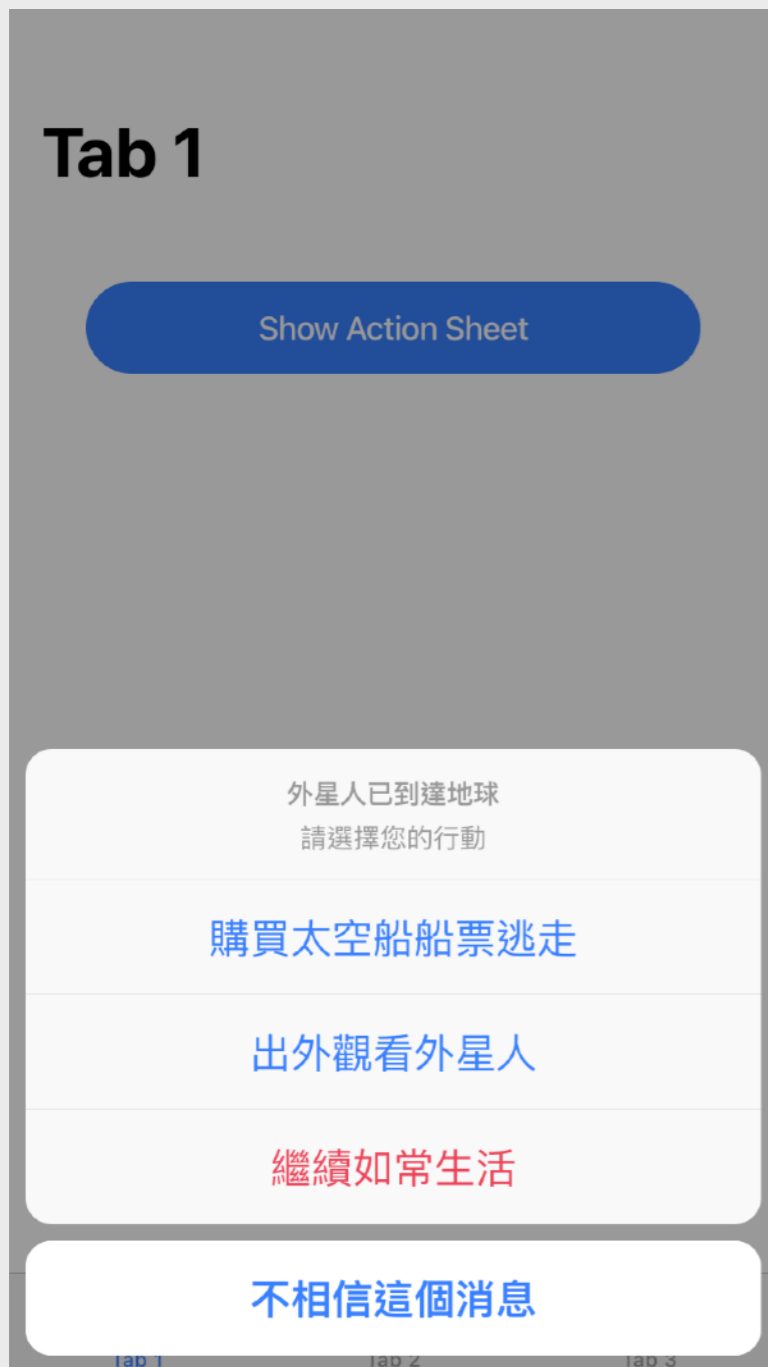
```
constructor(public toastController:ToastController){}
```

③調用

```
public showToast(){  
  this.toastController.create({  
    message: "火星其實並沒有火",  
    position: "bottom",  
    duration: 2000  
  }).then(t=>t.present());  
}
```

Toast使用message來顯示訊息，position為顯示的位置，可設定為top、middle和bottom，duration為顯示時間(以毫秒作單位)。

警示元件範例 - ActionSheetController



①載入

```
import {ActionSheetController} from '@ionic/angular';
```

②實體化

```
constructor(public actSheet:ActionSheetController){}
```

③調用

```
public showActionSheet(){  
  this.actSheet.create({  
    header: "外星人已到達地球",  
    subHeader: "請選擇您的行動",  
    buttons: [  
      {text:"購買太空船船票逃走", handler:()=>{console.log('a');}},  
      {text:"出外觀看外星人", handler:()=>{console.log('b');}},  
      {text:"繼續如常生活", role:"destructive", handler:()=>{console.log('c');}},  
      {text:"不相信這個消息", role:"cancel", handler:()=>{console.log('d');}}  
    ]  
  }).then(a=>a.present());  
}
```

可自由增加或減少buttons的數量，如果是「取消」行為的話則可設定role為cancel，如果是「刪除」行為的話則可設定role為destructive。

handler為按下該button後要執行的事情。

彈出式頁面範例 - Modal (1)



我們可使用Modal來製作出彈出式全頁頁面。多數用於顯示或處理比較簡單的資料及事情。

彈出式頁面範例 - Modal (2)

TypeScript檔案

```
import { Component, ViewChild } from '@angular/core';
import { IonModal } from '@ionic/angular';
import { OverlayEventDetail } from '@ionic/core/components';

@Component({
  selector: 'app-tab1',
  templateUrl: 'tab1.page.html',
  styleUrls: ['tab1.page.scss']
})
export class Tab1Page {

  @ViewChild(IonModal) modal: IonModal;

  message = '開個Modal出來看看一個秘密🤔';
  name: string;

  cancel() {
    this.modal.dismiss(null, 'cancel');
  }

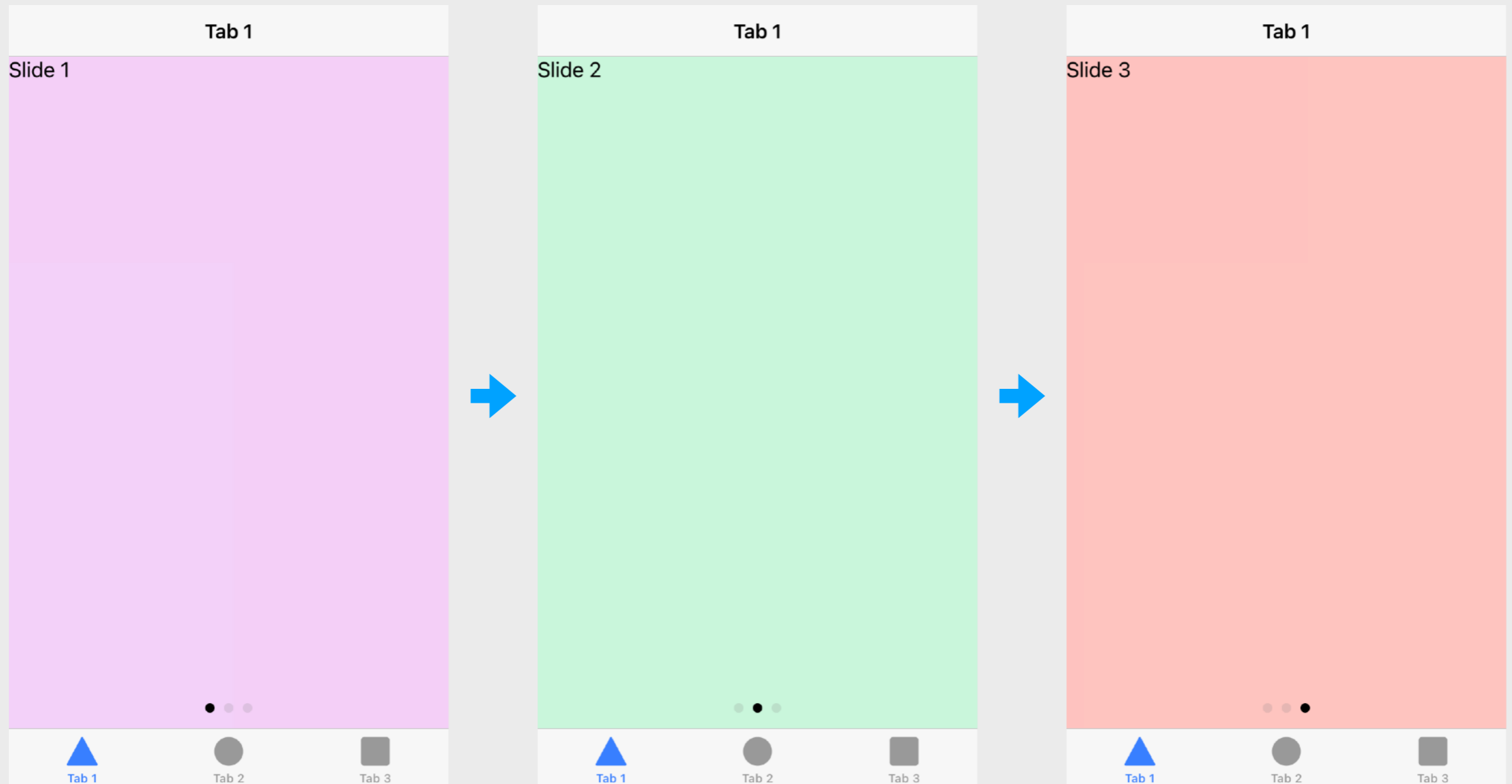
  confirm() {
    this.modal.dismiss(this.name, 'confirm');
  }

  onWillDismiss(event: Event) {
    const ev = event as CustomEvent<OverlayEventDetail<string>>;
    if (ev.detail.role === 'confirm') {
      this.message = ev.detail.data+'是個聰明人🤪';
    }
  }
}
```

HTML檔案

```
<ion-header>
  <ion-toolbar>
    <ion-title>Modal範例</ion-title>
  </ion-toolbar>
</ion-header>
<ion-content class="ion-padding">
  <ion-button id="open-modal" expand="block">打開Modal</ion-button>
  <p>{{ message }}</p>
  <ion-modal trigger="open-modal" (willDismiss)="onWillDismiss($event)">
    <ng-template>
      <ion-header>
        <ion-toolbar>
          <ion-buttons slot="start">
            <ion-button (click)="cancel()">閉關</ion-button>
          </ion-buttons>
          <ion-title>🔒 輸入一個人名以觀看秘密</ion-title>
          <ion-buttons slot="end">
            <ion-button (click)="confirm()" [strong]="true">確定</ion-button>
          </ion-buttons>
        </ion-toolbar>
      </ion-header>
      <ion-content class="ion-padding">
        <ion-item>
          <ion-label position="stacked">請輸入人名</ion-label>
          <ion-input type="text" placeholder="一個人名" [(ngModel)]="name"></ion-input>
        </ion-item>
      </ion-content>
    </ng-template>
  </ion-modal>
</ion-content>
```

IonSlides 組件橫向滑動頁面範例 (1)



我們可使用IonSlides來製作出橫向滑動頁面。

IonSlides 組件橫向滑動頁面範例 (2)

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Tab 1
    </ion-title>
  </ion-toolbar>
</ion-header>
```

```
<ion-content>
```

```
<ion-slides #tab1Slides pager="true">
```

```
<ion-slide id="slide1">
  <div>Slide 1</div>
</ion-slide>
```

```
<ion-slide id="slide2">
  <div>Slide 2</div>
</ion-slide>
```

```
<ion-slide id="slide3">
  <div>Slide 3</div>
</ion-slide>
```

```
</ion-slides>
```

```
</ion-content>
```

HTML檔

若false則不顯示pager

可以#號來定義Slides的名稱，並在TS裡把其變成為TS物件來調用

```
import { Component, ViewChild } from '@angular/core';
import { IonSlides } from '@ionic/angular';
```

```
@Component({
  selector: 'app-tab1',
  templateUrl: 'tab1.page.html',
  styleUrls: ['tab1.page.scss']
})
```

```
export class Tab1Page {
```

```
  @ViewChild(IonSlides,{static:false}) tab1Slides: IonSlides;
```

```
  constructor(){}
}
```

TypeScript檔

可自行設定slides的height

pager於不作為當前頁面時的顏色

pager於作為當前頁面時的顏色

```
ion-slides{
  width: 100%;
  height: 100%;
  overflow-y: hidden;
  --bullet-background: gray;
  --bullet-background-active: black;
}
ion-slide{
  display: block;
  width: 100%;
  height: 100%;
  overflow-y: auto;
  text-align: left;
}
```

```
#slide1{
  background-color: #F4D0F8;
}
#slide2{
  background-color: #C9F6DB;
}
#slide3{
  background-color: #FEC3BF;
}
```

CSS檔

亦可呼叫Slides物件進行強制跳頁：
this.homeSlides.slideTo(頁面數碼);

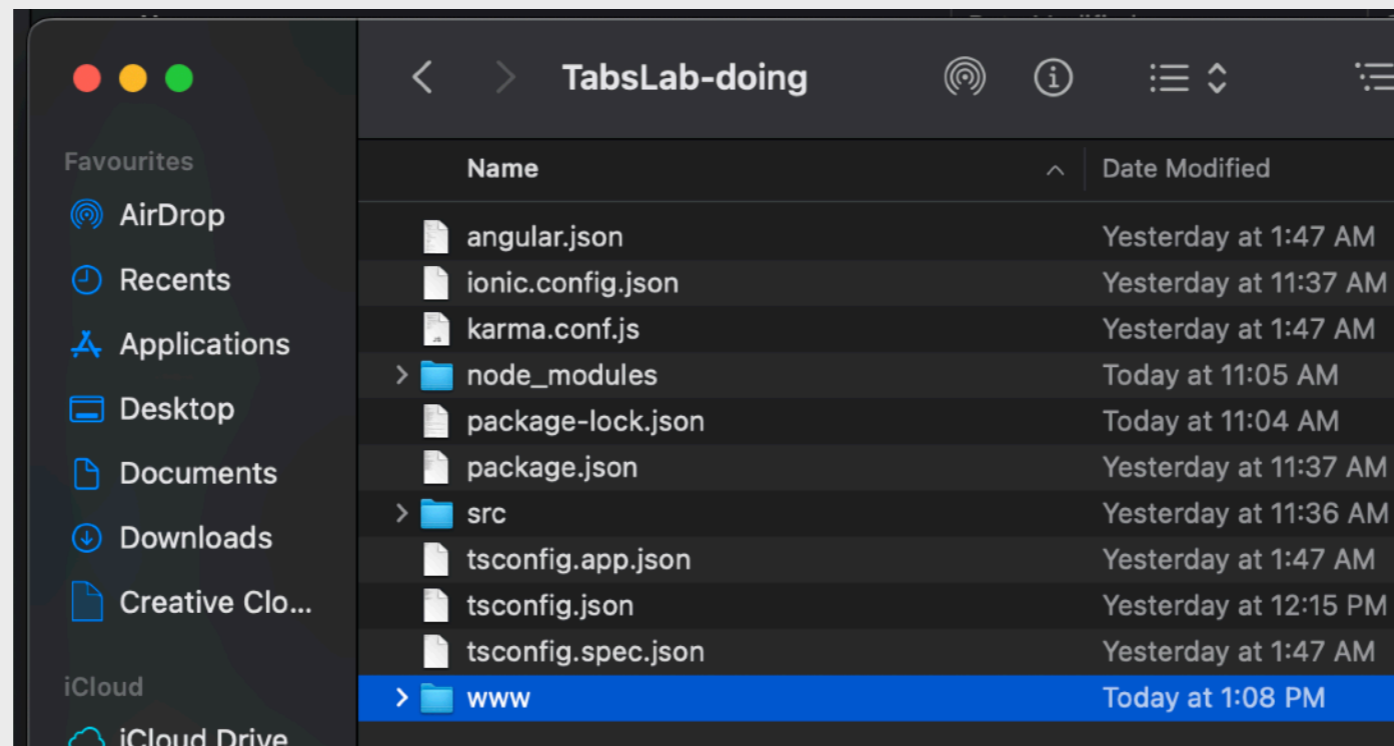
匯出Ionic專案成可運行的程式 (1)

大家有否察覺到，我們一直以來都是透過 ionic serve 來運行程式呢？哪麼怎樣把程式「上架」分享給別人使用呢？難道要叫別人走到Mac或Windows或Linux電腦然後安裝Node.js+Ionic+Angular再在終端機裡ionic serve來運行程式？

當然不會這麼愚蠢吧，其實我們可透過在終端機裡執行：

`ionic build --release --prod --aot`

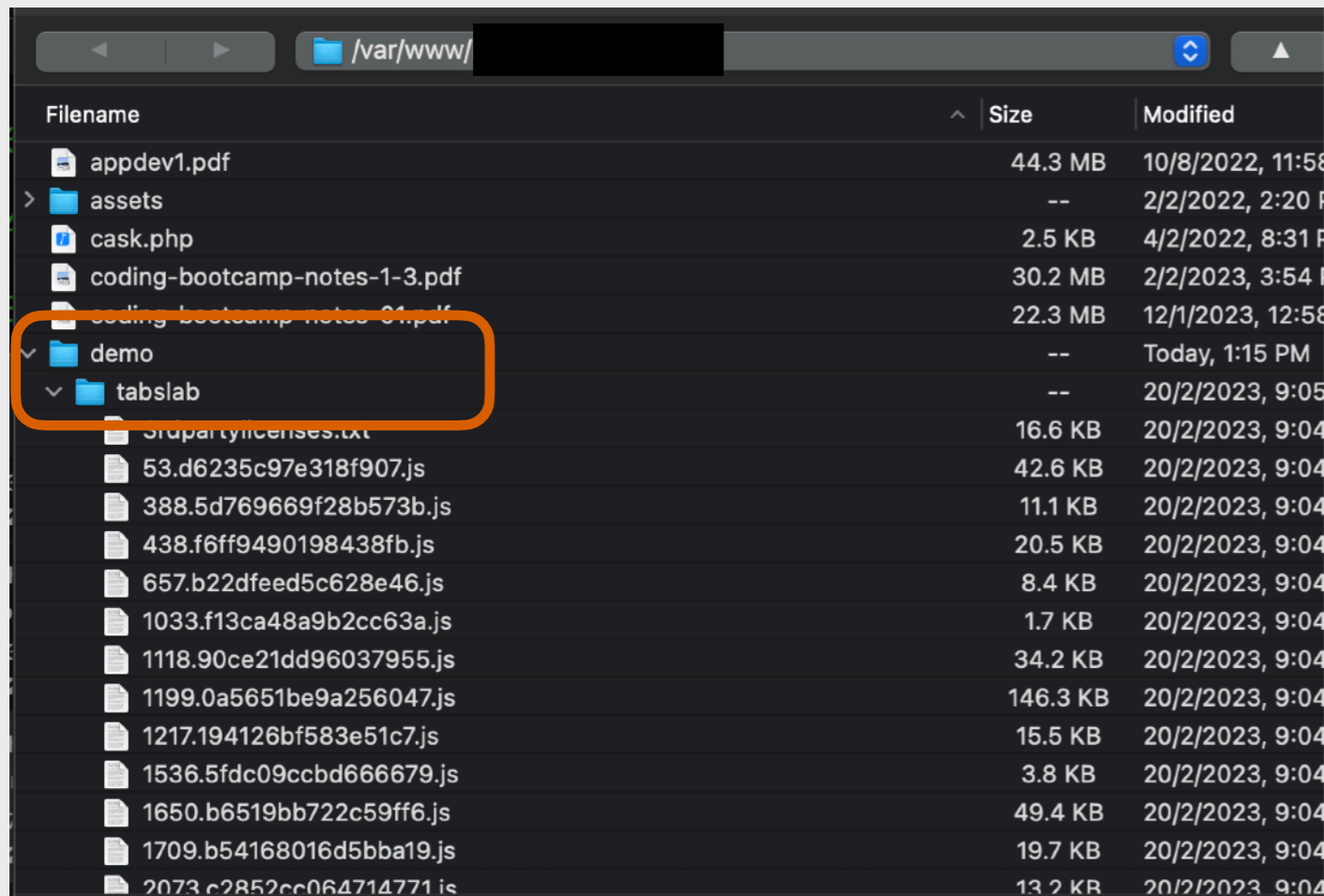
來把Ionic專案匯出(Export)成網站，然後再放上自架或租借回來的伺服器。



匯出後會出現多一個名為www的資料夾，
這就是網站成品。

匯出Ionic專案成可運行的程式 (2)

```
index.html x
www > index.html > html
1 <!DOCTYPE html><html lang="en"><head>
2   <meta charset="utf-8">
3   <title>TabsLab</title>
4
5   <base href="/demo/tabslab/">
6
7   <meta name="color-scheme" content="light dark">
8   <meta name="viewport" content="viewport-fit=cover, width=device-wid
9   <meta name="format-detection" content="telephone=no">
10  <meta name="msapplication-tap-highlight" content="no">
```



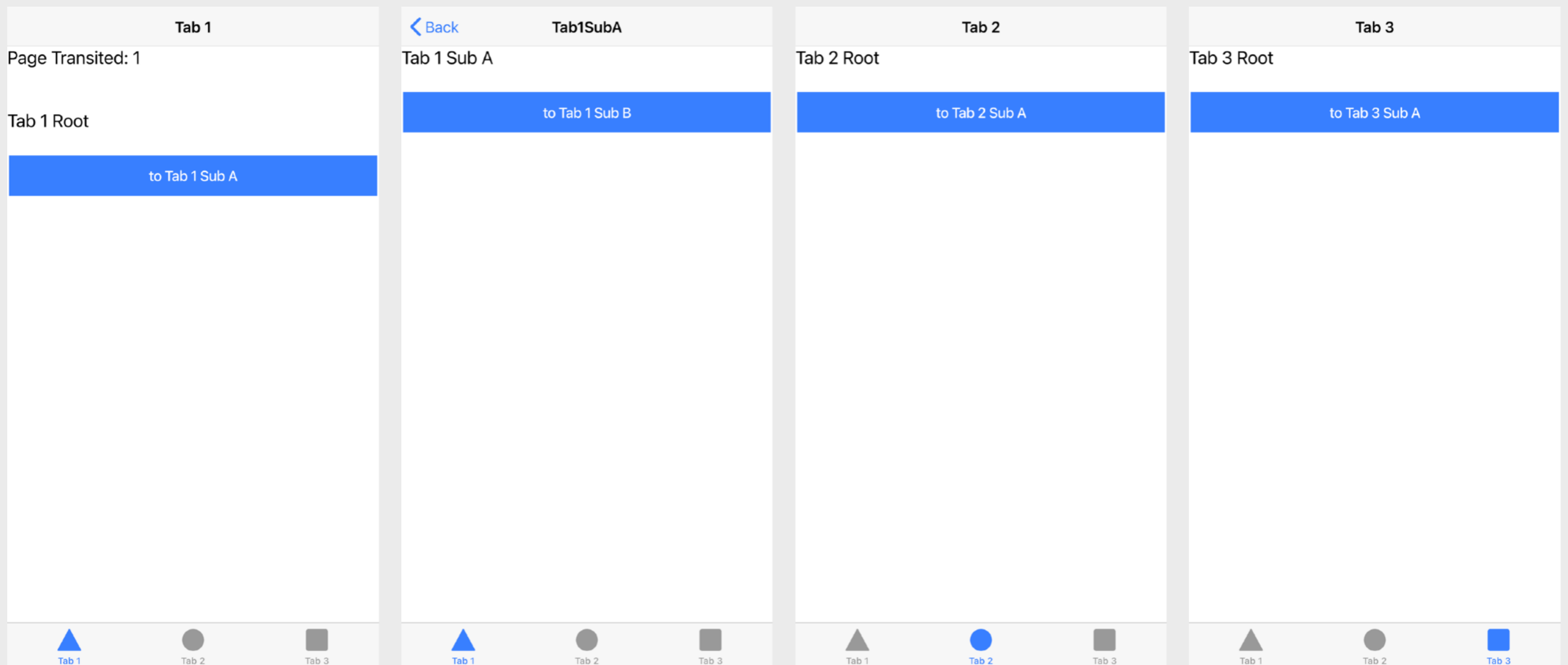
若放上以路徑來作導引的伺服器(如 PHP、JSP、ASP等)，我們則需注意，若伺服器中，儲存著我們剛匯出來的程式的資料夾並不是root(即主網域的那一層)，那麼我們則需到index.html中修改base元素，把其href屬性設定為程式所被放置的路徑。然而在本例子中，筆者把本身的www改了為tabslab，而且置於根網域的demo資料夾之內，所以index.html中base元素的href被設為 </demo/tabslab/>。

課題十：

Tab-based頁面組、客製化元件及
Ionic Service應用

Tab-based 頁面結構及換頁 (1)

Tab-base的意思就是把不同的頁面組(或頁面)，以水平選單的方式供用戶選擇使用甚麼的頁面組。一般而言，頁面組選擇器都會放到程式的底端部分，並常稱作為TabBar。



以上是Tab-based介面的例子

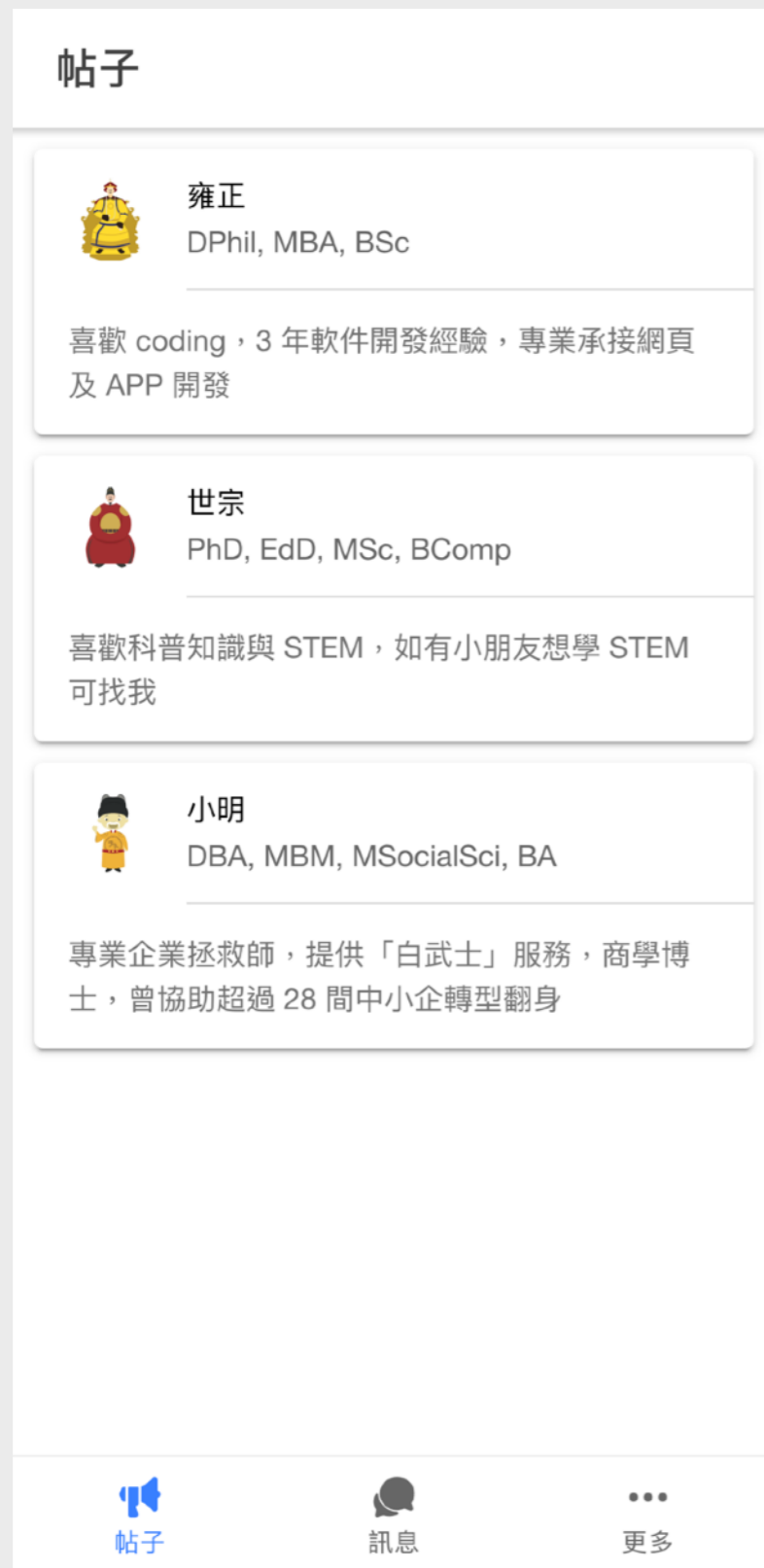
Tab-based 頁面結構及換頁 (2)



為了理解Tab-based的結構，我們先釐清一個重要的概念，就是父頁面和子頁面組。
在跳頁時，我們需要分別清楚是想把整個父頁面轉到另一頁面(在父頁面中進行換頁)還是在子頁面組中進行換頁。

TabBar

Tab-based 頁面結構及換頁 (3)



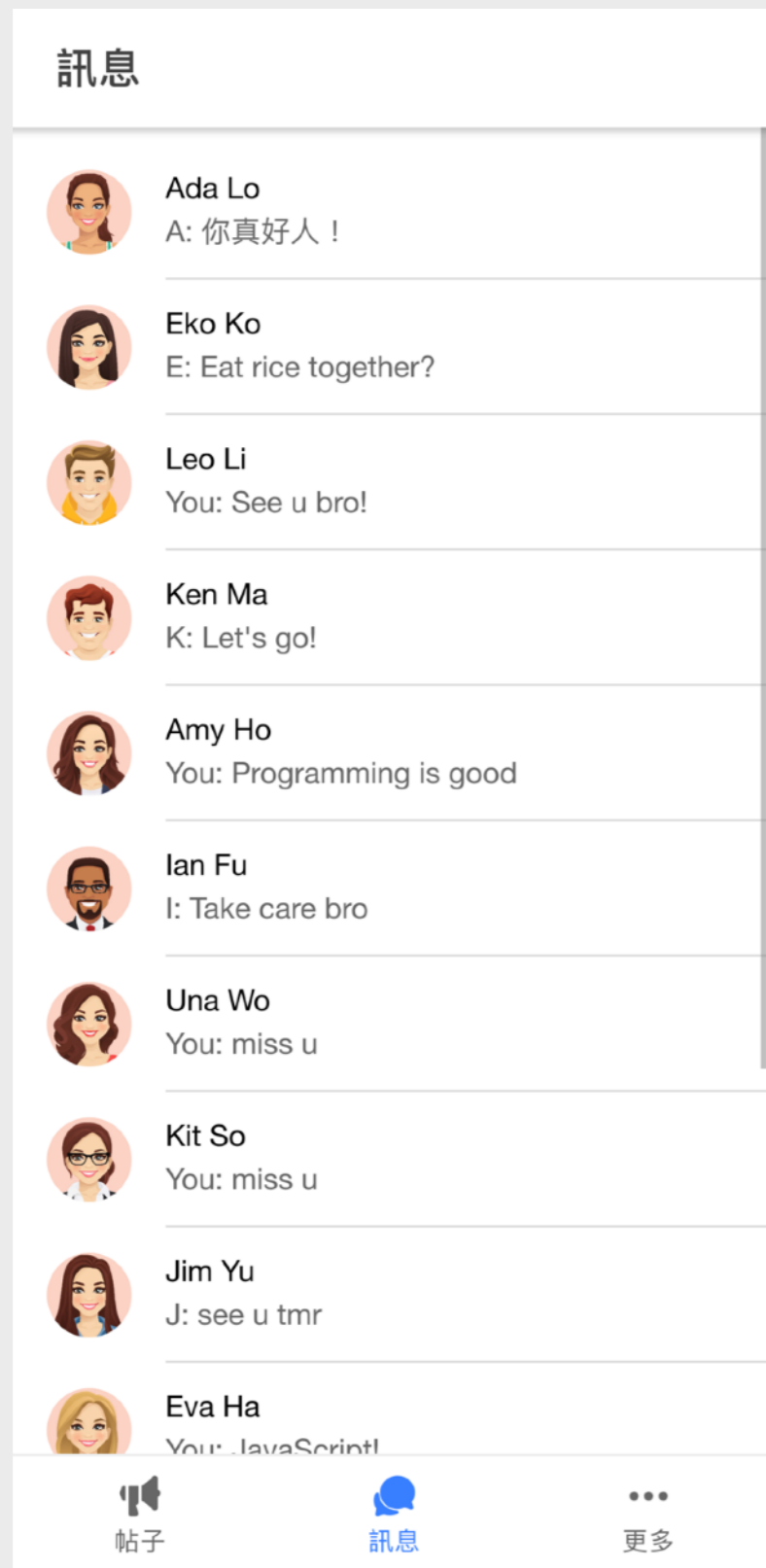
例子一

子頁面組內跳頁
→

因在子頁面組內進行跳頁，所以父頁面不受任何影響，TabBar也保留在父頁面底下



Tab-based 頁面結構及換頁 (4)

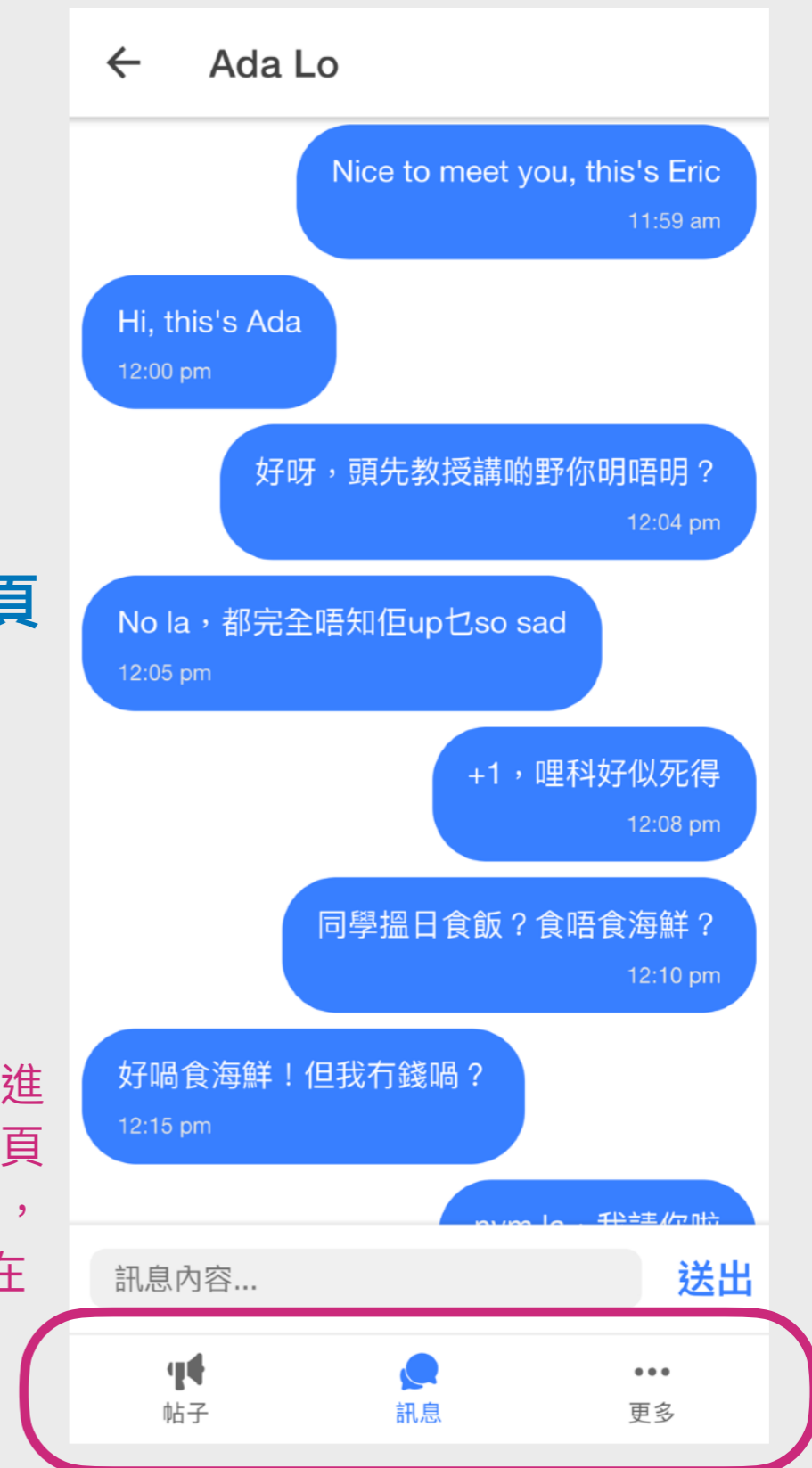


例子二

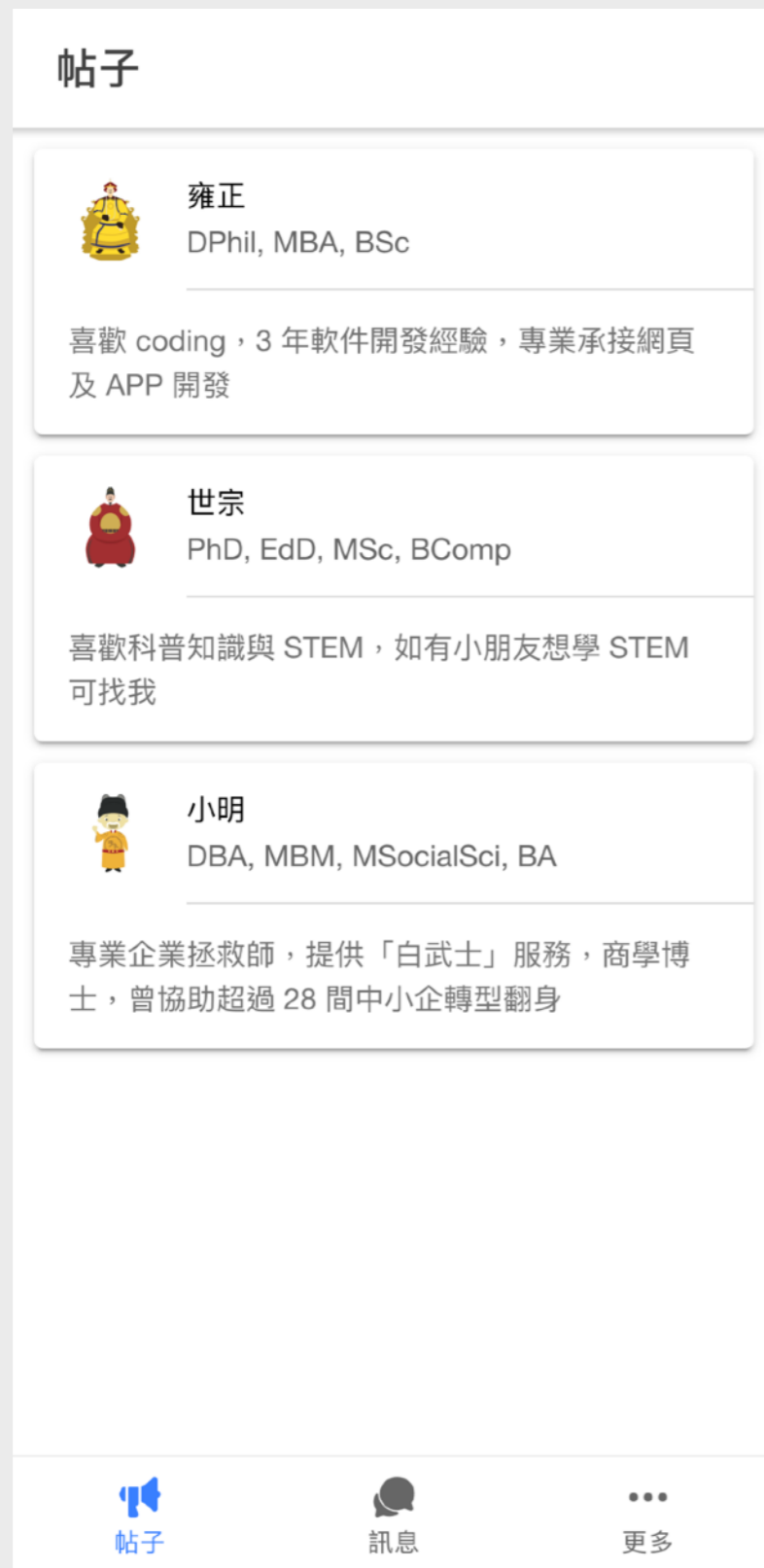
子頁面組內跳頁



因在子頁面組內進行跳頁，所以父頁面不受任何影響，TabBar也保留在父頁面底下



Tab-based 頁面結構及換頁 (4)



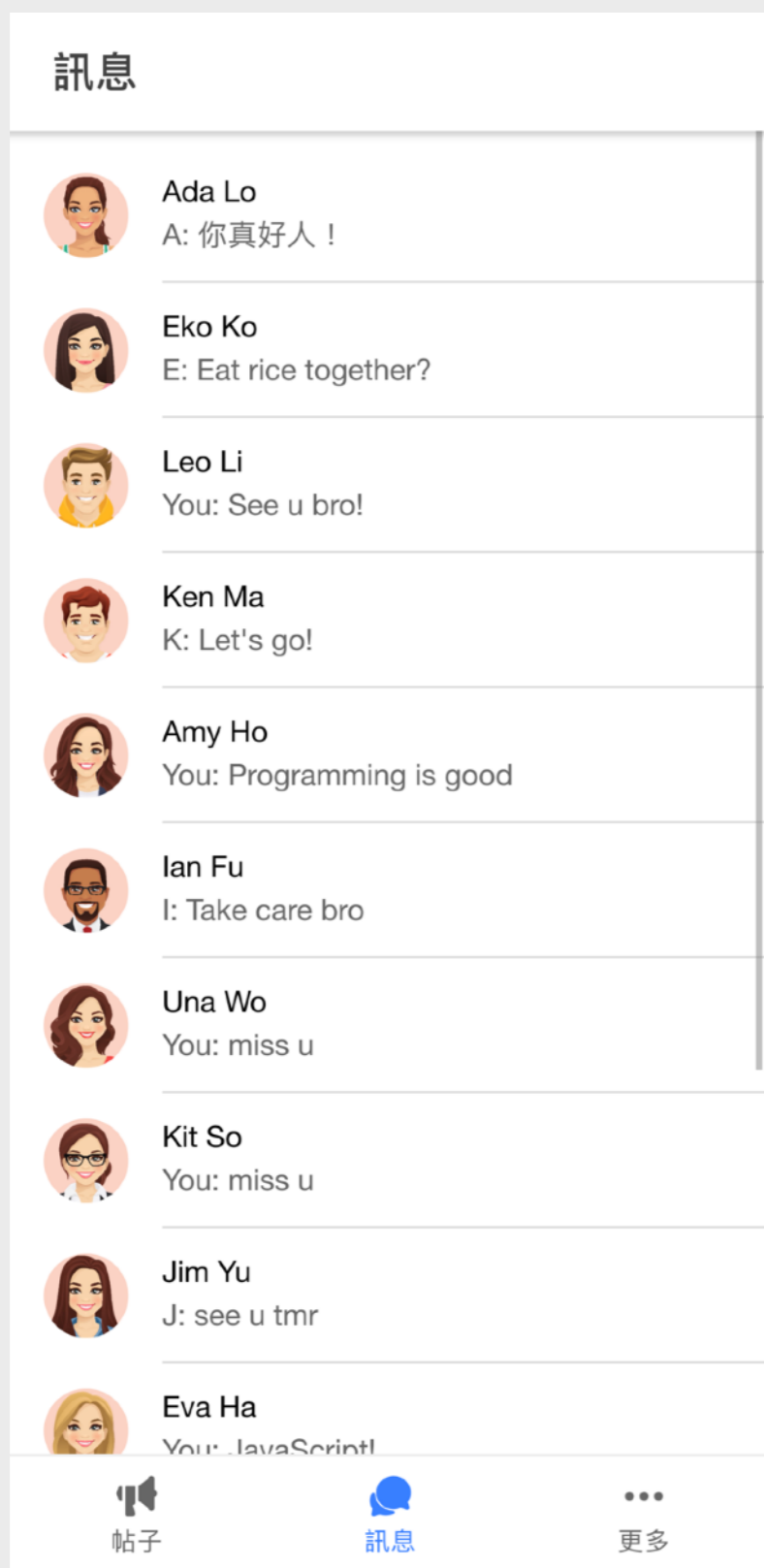
例子三

整個父頁面跳頁
→

因整個父頁面已跳到個人頁，所以 TabBar 不會出現

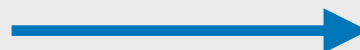


Tab-based 頁面結構及換頁 (4)



例子四

整個父頁面跳頁



因整個父頁面已跳到聊天頁，所以 TabBar 不會出現

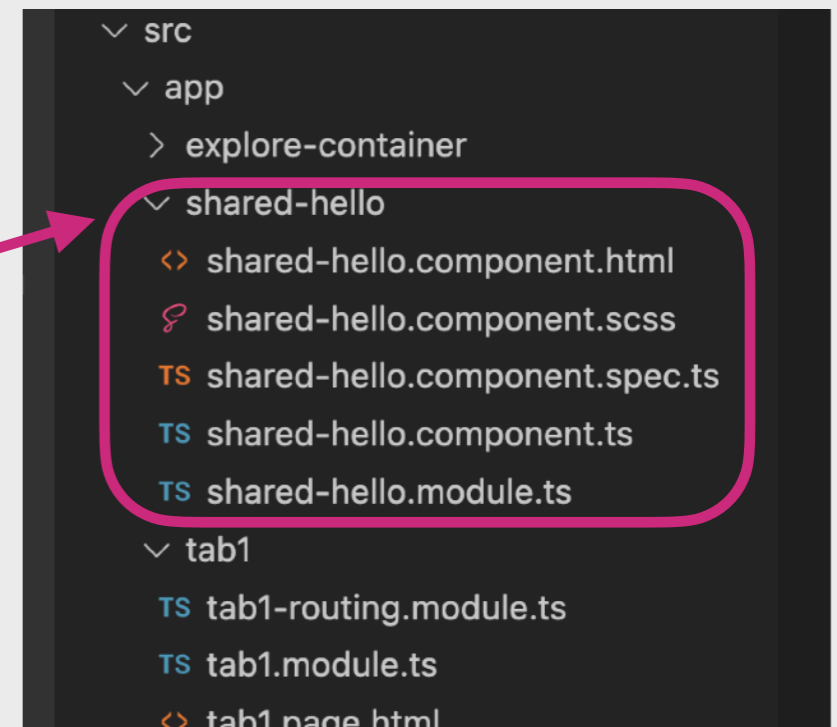


Custom Component 客製化元件 (1)

稍後，我們就把Tab-based介面、Custom Component與早前提及過的Ionic Service一起製作出來。哪甚麼是Custom Component客製化元件呢？

顧名思義，Custom Component客製化元件就是讓我們自創新的介面元件，並以**獨立的模組儲存著**，然後再以自定義的Tag名應用到一個或以上的HTML裡，例如：

```
<app-shared-hello pageName="Tab 1 Sub B"></app-shared-hello>
```



```
<ion-content>
  <app-shared-hello pageName="Tab1 Sub B"></app-shared-hello>
</ion-content>
```

哪為甚麼要這樣做呢？試想想，如果App裡有很多的頁面，但當中有不同的頁面都需要使用到同一堆HTML碼加CSS樣式加TypeScript程式的**組合**，而該組合亦比較沈長或複雜，那麼我們便要很多餘地把它重覆擺放在不同的頁面，很浪費時間；而且每次要更改它的話都要到每個頁面裡作出更改，非常不方便。所以，我們便引進了「把**組合**抽出來做」的**模組化**概念，當某頁面需要使用它時便可加插**模組Tag**到該頁面的HTML中，更可**透過attribute來傳遞參數(資料)到該模組**中供其使用。

建立 Tab-based 程式 (1)

現在，我們就把Tab-based介面、Custom Component與早前提及過的Ionic Service一起製作出來。首先開啟終端機並創建一個Tab-based的Ionic+Angular程式：

```
ionic start TabsLab tabs --type=angular
```

如被問及是否加進Capacitor的問題，若你想開發原生iOS及Android App則鍵盤按下「Y」鍵後再「Enter」，否則答「N」則可

```
dave@DD-M1Mac Desktop % ionic start TabsLab tabs --type=angular
✓ Preparing directory ./TabsLab in 1.32ms
✓ Downloading and extracting tabs starter in 1.05s
? Integrate your new app with Capacitor to target native iOS and Android? (y/N)
N
```

如被問及是否成為Ionic會員，亦可答「Y」或「N」

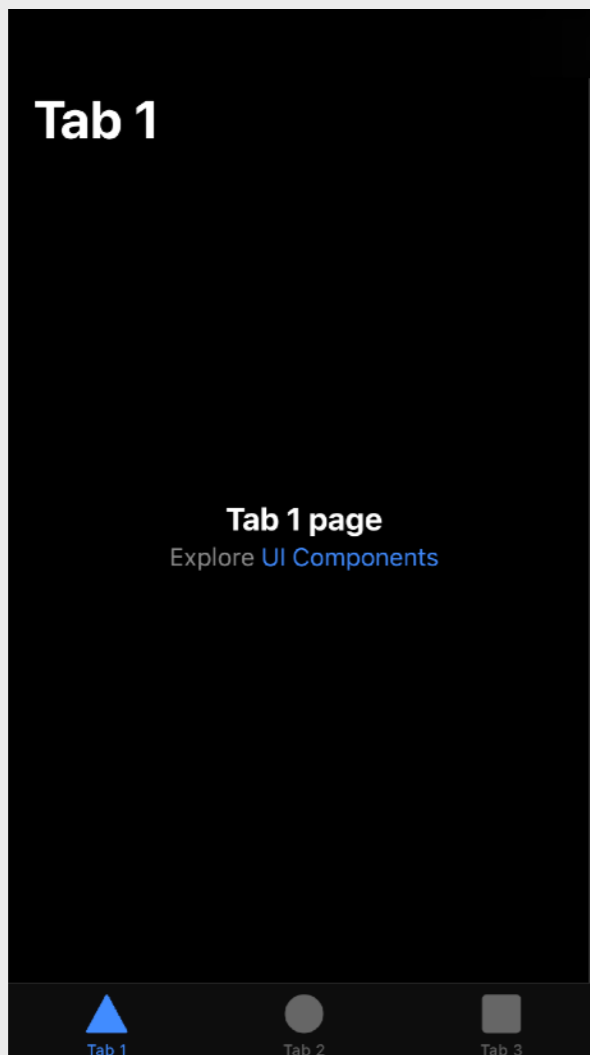
```
Join the Ionic Community! ❤️
Connect with millions of developers on the Ionic Forum and get access to live
events, news updates, and more.
? Create free Ionic account? (y/N) N
```

建立 Tab-based 程式 (2)

注意，我們需先cd到剛剛創建好的Ionic Project即TabsLab才可運程式，否則會出現錯誤。

```
dave@DD-M1Mac Desktop % ionic serve
[ERROR] Sorry! ionic serve can only be run in an Ionic project directory.

If this is a project you'd like to integrate with Ionic, run ionic init.
dave@DD-M1Mac Desktop % cd TabsLab
dave@DD-M1Mac TabsLab % ionic serve
> ng run app:serve --host=localhost --port=8101
```



程式成功運行了，我們可看見底下附送了3個Tab。由於本例子中使用的電腦被調較成dark mode，所以程式以dark mode模式顯示出來。

建立 Tab-based 程式 (3)

```
variables.scss x
src > theme > variables.scss > ...
70 /** light **/
71 --ion-color-light: #f4f5f8;
72 --ion-color-light-rgb: 244, 245, 248;
73 --ion-color-light-contrast: #000000;
74 --ion-color-light-contrast-rgb: 0, 0, 0;
75 --ion-color-light-shade: #d7d8da;
76 --ion-color-light-tint: #f5f6f9;
77 }
78
79 @media (prefers-color-scheme: dark) {
80   /*
81   * Dark Colors
82   * -----
83   */
84
85   body {
86     --ion-color-primary: #428cff;
```



```
variables.scss M x
src > theme > variables.scss > :root
70 /** light **/
71 --ion-color-light: #f4f5f8;
72 --ion-color-light-rgb: 244, 245, 248;
73 --ion-color-light-contrast: #000000;
74 --ion-color-light-contrast-rgb: 0, 0, 0;
75 --ion-color-light-shade: #d7d8da;
76 --ion-color-light-tint: #f5f6f9;
77 }
```

為了讓程式內容更易的被看見，所以我們到variables.scss
把第79行或以下的dark theme CSS刪掉。

```
variables.scss M  tsconfig.json
tsconfig.json > {} compilerOptions
1  /* To learn more about this file see: https://
2  {
3    "compileOnSave": false,
4    "compilerOptions": {
5      "baseUrl": "./",
6      "outDir": "./dist/out-tsc",
7      "forceConsistentCasingInFileNames": true,
8      "strict": false,
9      "noImplicitAny": false,
10     "noImplicitOverride": true,
```

按 **control + c** 以停止運行程式，然後到
tsconfig.json，增加noImplicitAny並設為
false，亦把strict設為false，如未曾安裝
tslint則再在終端機內輸入：

npm i -g tslint

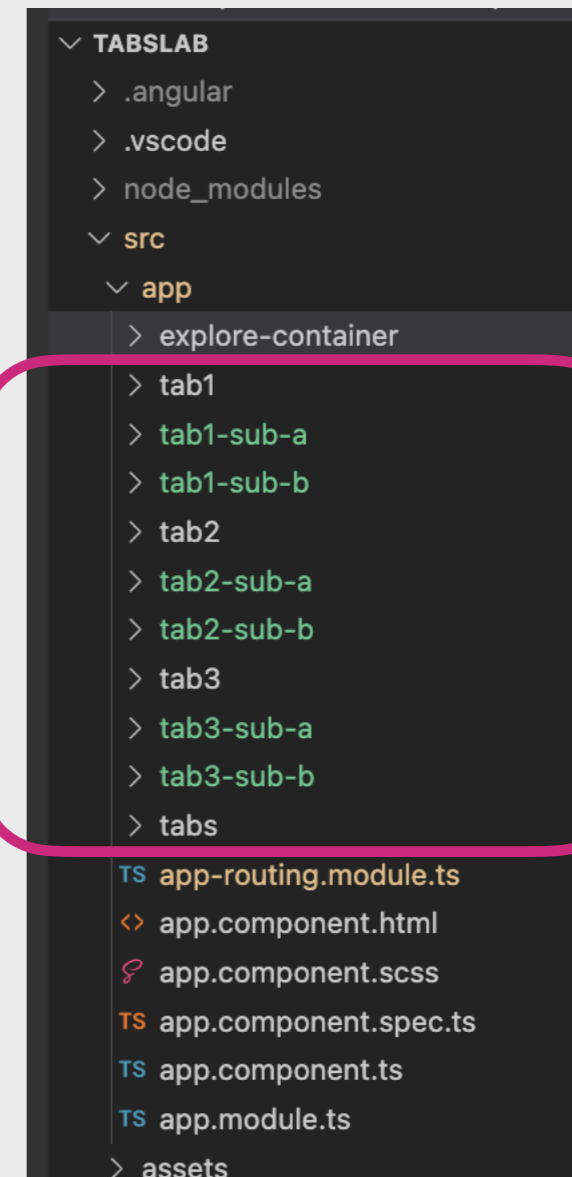
建立 Tab-based 程式 (4)

然後為Tab1、Tab2及Tab3各建立兩個子頁面待稍後章節使用。
(共6個頁面會被建立)

```
ionic g page Tab1SubA
ionic g page Tab1SubB
ionic g page Tab2SubA
ionic g page Tab2SubB
ionic g page Tab3SubA
ionic g page Tab3SubB
```

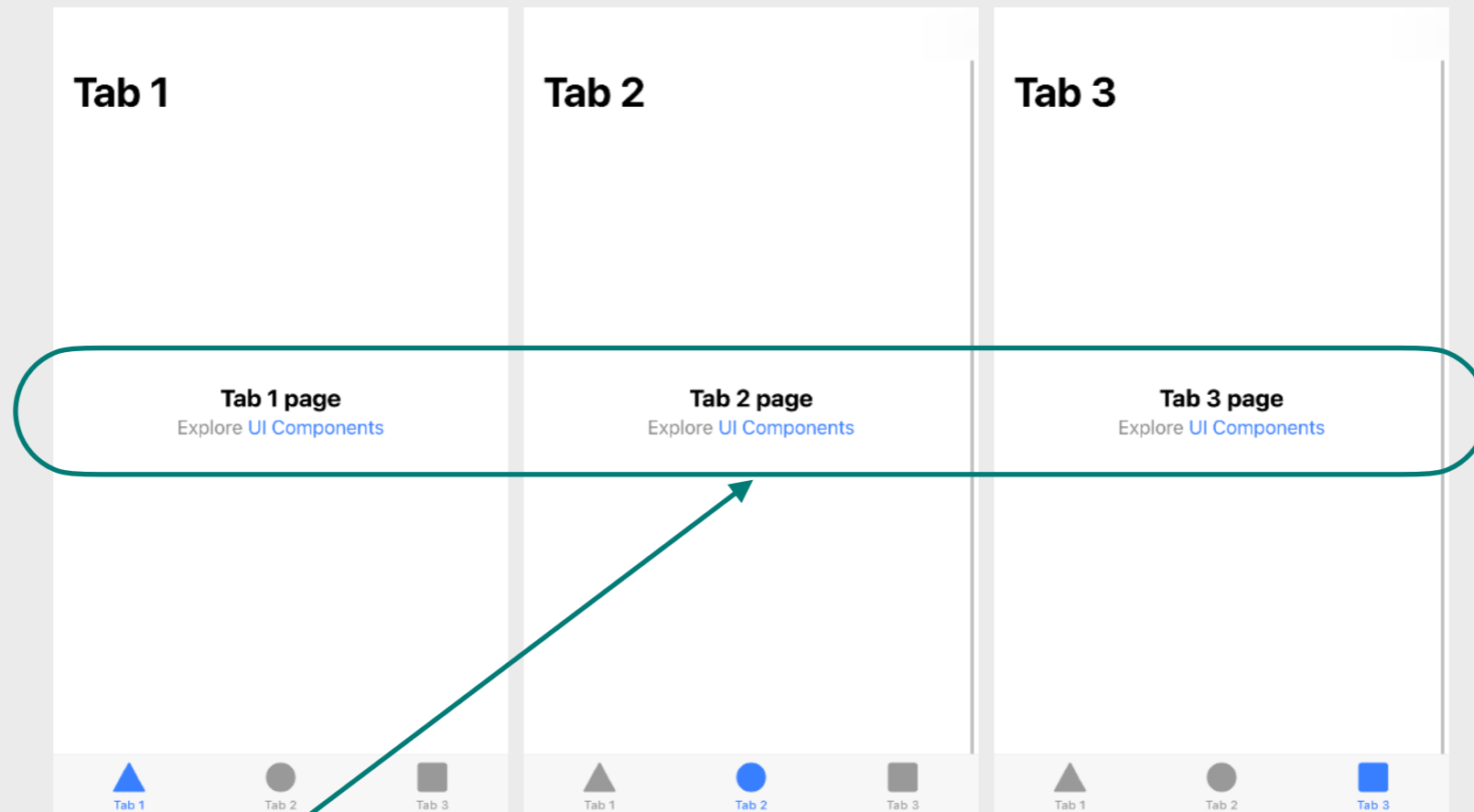
g是generate的簡寫，
Ionic接受以g代替長寫

最後再運行程式：`ionic serve`



在VS Code裡看見成功建立了6個頁面

Custom Component 客製化元件 (2)



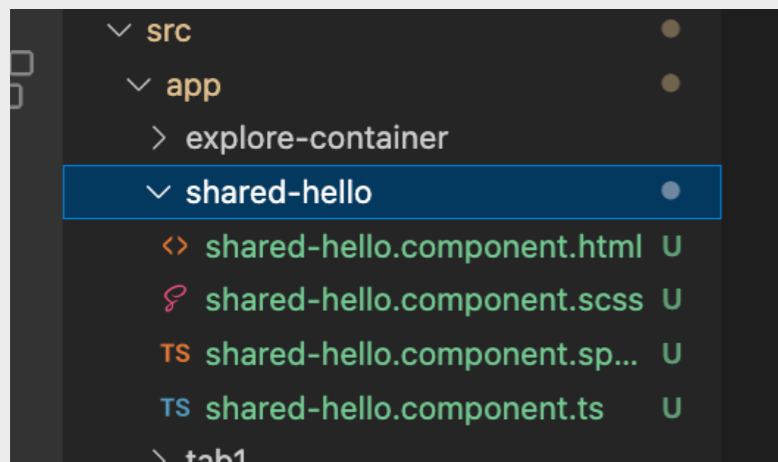
有沒有留意到，三個Tab畫面中的正中央，在其HTML檔中，都是有一個寫著`<app-explore-container>`的Tag呢？其實，這就是Custom Component，把一堆HTML碼加CSS樣式加TypeScript程式組合起來成為一個獨立的部件(即Component的意思)，令不同的頁面都可以輕易用到它，不需重覆寫很多次(即不需在不同頁面重覆寫出來)。

Custom Component 客製化元件 (3)

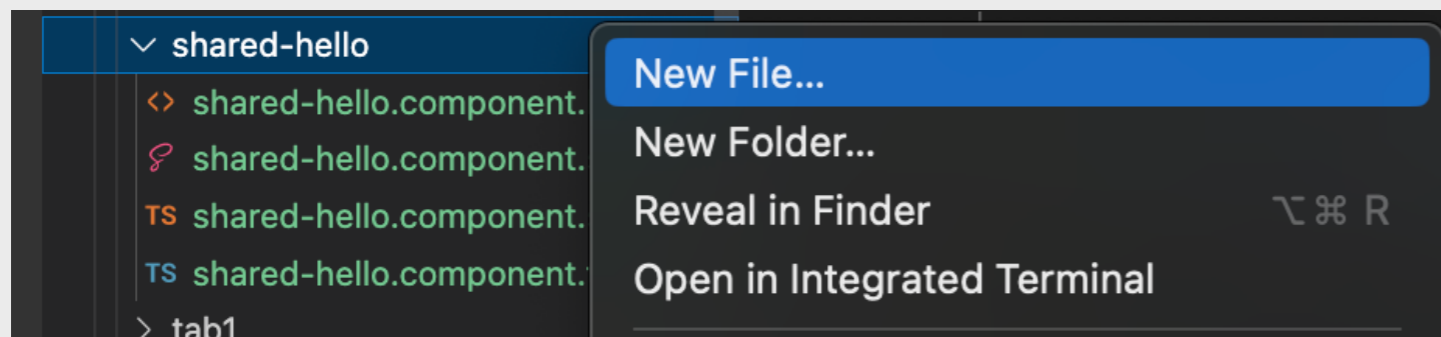
剛才一口氣的建立了6個頁面，計入本來有的3個Tab頁面，即是共有9個頁面。大概應該也猜到吧，我們應該就是想到各頁面都放置同一個的Component。因此，我們現在便需要建立一個Component，在本例子中，我們就稱該Component為SharedHello：

ionic g component SharedHello

然後我們會看見成功新增了一個SharedHello資料夾並包含著4個SharedHello相關的檔案：



然後右擊SharedHello的資料夾選建立新檔案，並命名為shared-hello.module.ts



Custom Component 客製化元件 (4)

參考ExploreContainer Component中的explore-container.module.ts，把該檔的程式碼抄到shared-hello.module.ts並換上SharedHello Component對應的名字，即：

```
import { NgModule } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';

import { IonicModule } from '@ionic/angular';

import { SharedHelloComponent } from './shared-hello.component';

@NgModule({
  imports: [ CommonModule, FormsModule, IonicModule ],
  declarations: [ SharedHelloComponent ],
  exports: [ SharedHelloComponent ]
})
export class SharedHelloComponentModule {}
```

Custom Component 客製化元件 (5)

到shared-hello.component.html看看，它附送了一小段的HTML：

```
src > app > shared-hello > <> shared-hello.component.html > ...
1  <p>
2  |   shared-hello works!
3  </p>
4
```

現在我們就想嘗試在Tab1、Tab2及Tab3運用SharedHello，首先我們到tab1.module.ts裡載入SharedHello，即：

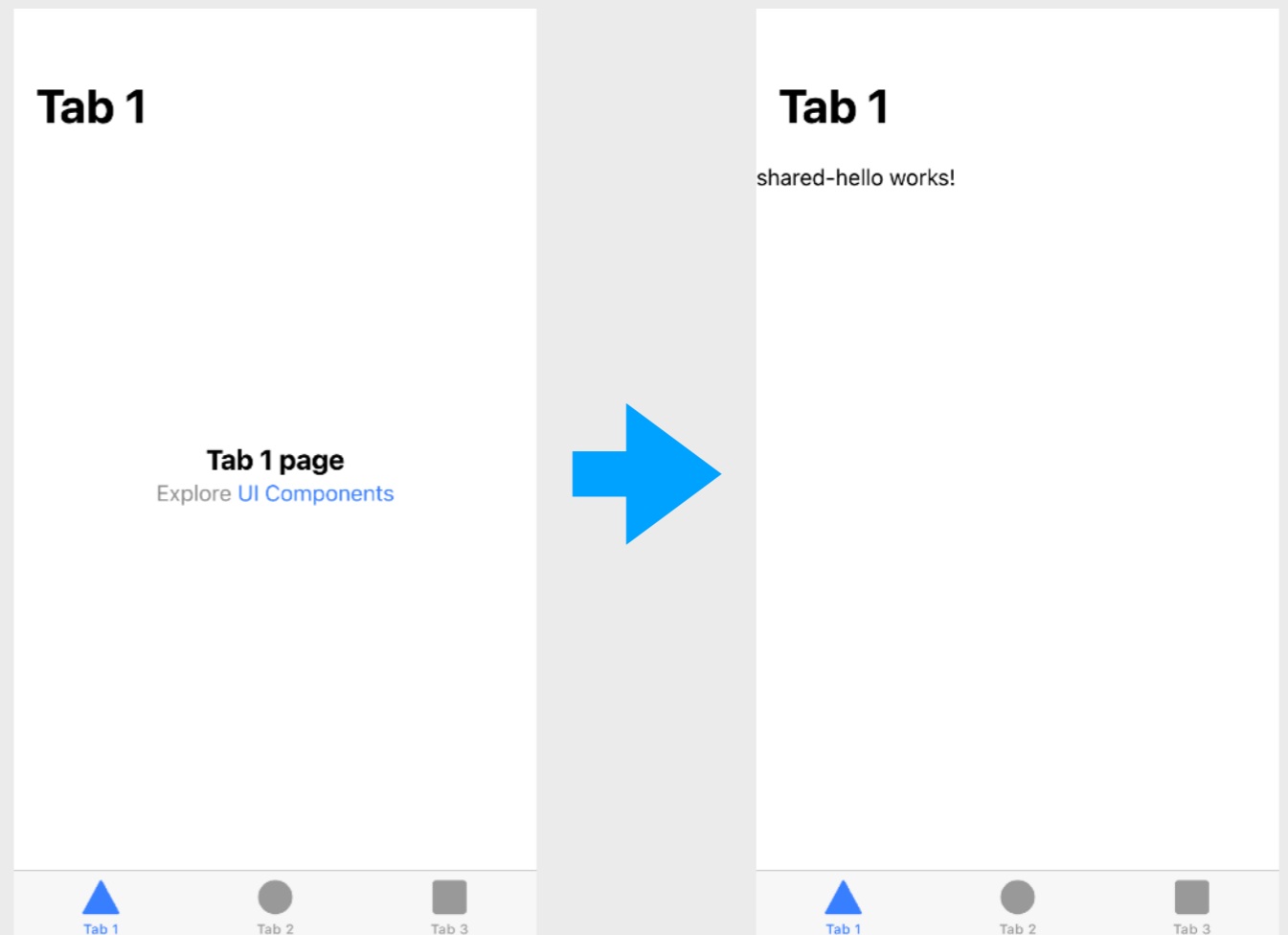
`import { SharedHelloComponentModule } from '../shared-hello/shared-hello.module';`
然後再在@NgModule區域內的imports陣列裡加上SharedHelloComponentModule

```
variables.scss M  tsconfig.json M  tab1.page.html  TS tab1.module.ts M  TS shared-hello.module.ts U
src > app > tab1 > TS tab1.module.ts > Tab1PageModule
1  import { IonicModule } from '@ionic/angular';
2  import { NgModule } from '@angular/core';
3  import { CommonModule } from '@angular/common';
4  import { FormsModule } from '@angular/forms';
5  import { Tab1Page } from './tab1.page';
6  import { ExploreContainerComponentModule } from '../explore-container/explore-container.module';
7
8  import { Tab1PageRoutingModule } from './tab1-routing.module';
9
10 import { SharedHelloComponentModule } from '../shared-hello/shared-hello.module';
11
12 @NgModule({
13   imports: [
14     IonicModule,
15     CommonModule,
16     FormsModule,
17     ExploreContainerComponentModule,
18     Tab1PageRoutingModule,
19     SharedHelloComponentModule
20   ],
21   declarations: [Tab1Page]
22 })
23 export class Tab1PageModule {}
24
```

Custom Component 客製化元件 (6)

我們到終端機以ionic serve運行程式，然後到Tab1的HTML檔把app-explore-container換成app-shared-hello並把name參數刪掉，即：

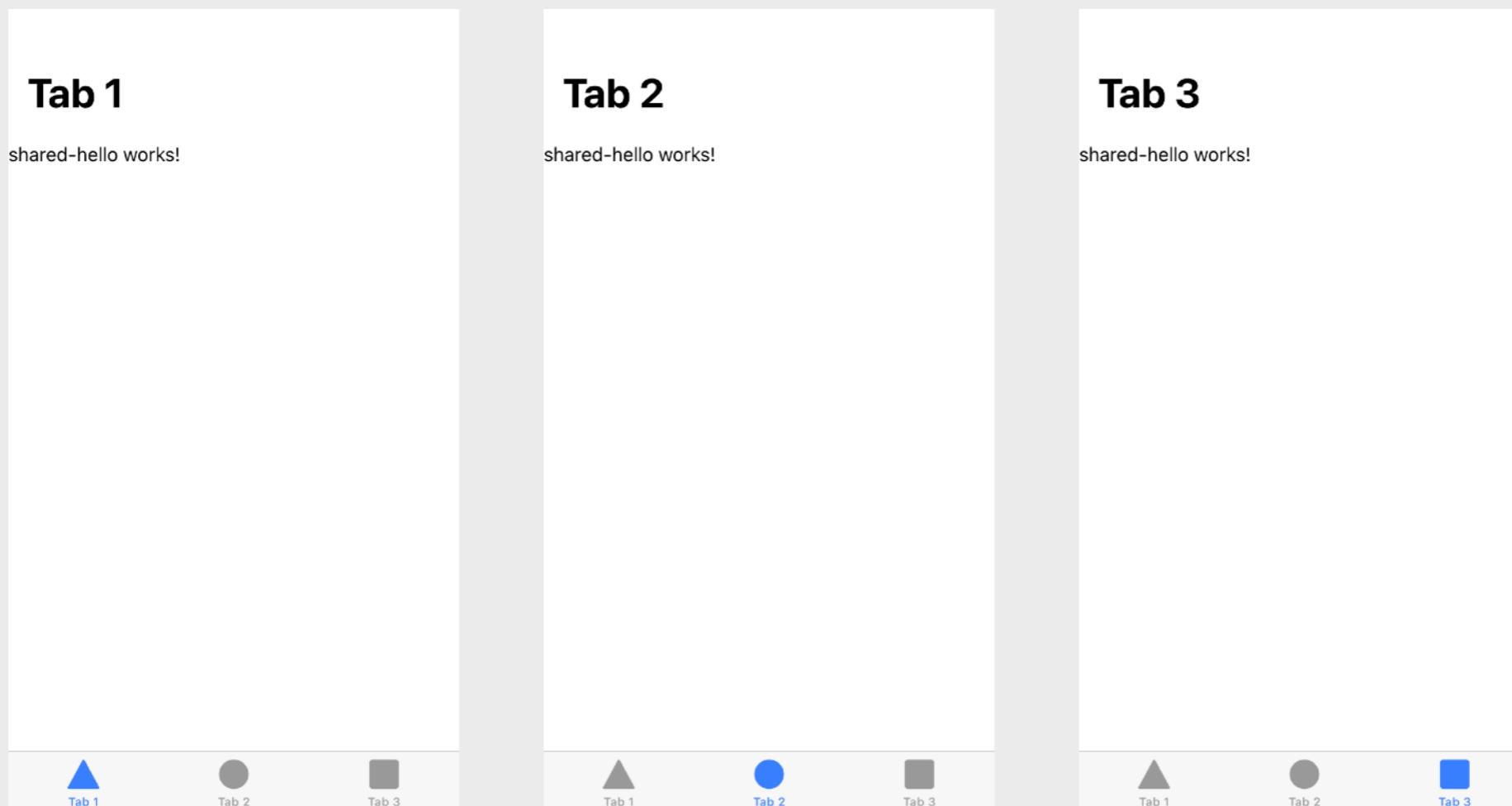
```
9 <ion-content [fullscreen]="true">
10   <ion-header collapse="condense">
11     <ion-toolbar>
12       <ion-title size="large">Tab 1</ion-title>
13     </ion-toolbar>
14   </ion-header>
15
16   <app-shared-hello></app-shared-hello>
17 </ion-content>
```



然後便會看到Tab1出現了「shared-hello works!」字句。注意所有**Custom Component**都會是以**app-**開頭，如**app-explore-container**和**app-shared-hello**。

Custom Component 客製化元件 (7)

同樣地，我們把Tab2及Tab3的app-explore-container換成app-shared-hello(記得需先到兩頁各自的module.ts檔裡載入SharedHello)，然後我們便看到3頁都同樣使用了SharedHello。



Custom Component 客製化元件 (8)

其實從explore-container也看得出，Custom Component是可以「吃」參數的，我們是可以透過HTML attribute來傳遞文字參數甚至是傳遞TypeScript裡的變數作參數到Component的。要令Component接受參數，我們則需要到其Component的TypeScript檔裡載入 **Input**。

```
TS shared-hello.component.ts •  
src > app > shared-hello > TS shared-hello.component.ts > SharedHelloComponent > toPagePath  
1 import { Component, OnInit, Input } from '@angular/core';
```

```
TS shared-hello.component.ts •  
src > app > shared-hello > TS shared-hello.component.ts > ...  
1 import { Component, OnInit, Input } from '@angular/core';  
2  
3 @Component({  
4   selector: 'app-shared-hello',  
5   templateUrl: './shared-hello.component.html',  
6   styleUrls: ['./shared-hello.component.scss'],  
7 })  
8 export class SharedHelloComponent implements OnInit {  
9  
10  @Input() pageName?: string;  
11  @Input() toPageName?: string;  
12  @Input() toPagePath?: string;  
13  
14  constructor() { }
```

並在Class中以 **@input()** 開頭來定義變數：

@Input() pageName?: string;

@Input() toPageName?: string;

@Input() toPagePath?: string;

Custom Component 客製化元件 (9)

設定好Input後，我們便把以下HTML代碼貼上SharedHello的HTML檔：

```
{{ pageName }}  
<ng-container *ngIf="toPageName">  
  <br><br>  
  <ion-button expand="full" (click)="toPage();">to {{ toPageName }}</ion-button>  
</ng-container>
```

TS shared-hello.component.ts

<> shared-hello.component.html ×

src > app > shared-hello > <> shared-hello.component.html > ng-container

```
1  {{ pageName }}  
2  <ng-container *ngIf="toPageName">  
3    <br><br>  
4    <ion-button expand="full" (click)="toPage();">to {{ toPageName }}</ion-button>  
5  </ng-container>
```

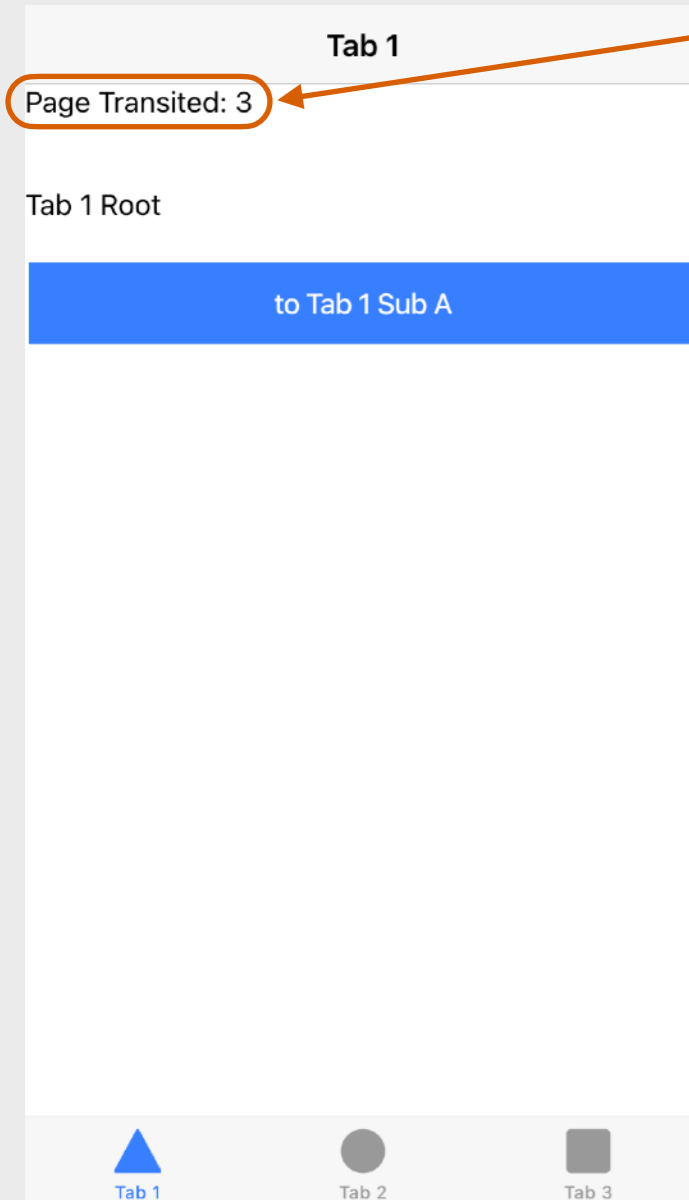
Custom Component 客製化元件 (10)

然後加上`toPage`函式，很顯然，此函式就是想進行跳頁。不需多說，前些的章節已經講述了跳頁的方法，但是我們會留在稍後透過Ionic Service來處理跳頁。

```
src > app > shared-hello > TS shared-hello.component.ts > ...
1  import { Component, OnInit, Input } from '@angular/core';
2
3  @Component({
4    selector: 'app-shared-hello',
5    templateUrl: './shared-hello.component.html',
6    styleUrls: ['./shared-hello.component.scss'],
7  })
8  export class SharedHelloComponent implements OnInit {
9
10     @Input() pageName?: string;
11     @Input() toPageName?: string;
12     @Input() toPagePath?: string;
13
14     constructor() { }
15
16     ngOnInit() {}
17
18     toPage(){}
19
20 }
21
```

Tab-based 實例中的 Ionic Service (1)

在本程式中，我們希望紀錄低跳頁次數，並在Tab1裡顯示，如右圖所示：



因此，我們需要建立到Ionic Service，並透過它來儲存著紀錄跳頁次數的變數，在本例子中我們把該變數命名為pageTransited。

首先我們在終端機裡停止運程式並執行 `ionic g service Global` 以創建出Ionic Service，然後在 `global.service.ts` 裡定義變數：
`public pageTransited=0;`

接著到上頁提到的跳頁部分，我們先載入Router即：

```
import { Router } from '@angular/router';
```

接著在constructor的開關小括號中定義Router即：

```
public router:Router
```

再加入toPage並於其中使用Router而跳頁即：

```
toPage(page){  
  this.router.navigateByUrl(page);  
}
```

Tab-based 實例中的 Ionic Service (2)

完成後的 global.service.ts

```
src > app > TS global.service.ts > ...
 1  import { Injectable } from '@angular/core';
 2  import { Router } from '@angular/router';
 3
 4  @Injectable({
 5    providedIn: 'root'
 6  })
 7  export class GlobalService {
 8
 9    public pageTransited=0;
10
11    constructor(public router:Router){}
12
13    toPage(page){
14      this.router.navigateByUrl(page);
15    }
16
17  }
```

Custom Component 客製化元件 (11)

然後到shared-hello.component.ts，先載入GlobalService：

```
import { GlobalService } from '../global.service';
```

然後創建GlobalService實體：

```
public gs:GlobalService
```

最後便完成toPage函式：

```
toPage(){  
    this.gs.pageTransited++;  
    this.gs.toPage(this.toPagePath);  
}
```

```
src > app > shared-hello > TS shared-hello.component.ts > ...  
1  import { Component, OnInit, Input } from '@angular/core';  
2  import { GlobalService } from '../global.service';  
3  
4  @Component({  
5      selector: 'app-shared-hello',  
6      templateUrl: './shared-hello.component.html',  
7      styleUrls: ['./shared-hello.component.scss'],  
8  })  
9  export class SharedHelloComponent implements OnInit {  
10  
11      @Input() pageName?: string;  
12      @Input() toPageName?: string;  
13      @Input() toPagePath?: string;  
14  
15      constructor(public gs:GlobalService) { }  
16  
17      ngOnInit() {}  
18  
19      toPage(){  
20          this.gs.pageTransited++;  
21          this.gs.toPage(this.toPagePath);  
22      }  
23  
24  }
```

Tab-based 頁面組內進行跳頁 (1)

要在Tab-based頁面組內進行跳頁，我們則需要使用tabs專屬的routing path (路徑)。首先，我們先到tabs資料夾內的[tabs-routing.module.ts](#)。然後再加上tabs內的子頁面路徑，因為我們在開始時創建了新的6個頁面，所以我們便加入6項子頁面路徑到routes陣列中的第0項鍵值配(key-value-pair)物件的children陣列中。而因為是在Tab-based頁面組內進行跳頁，所以路徑需要以根Tab名為開頭。(如下頁所示)

Tab1SubA :

```
{path:'tab1/tab1-sub-a', loadChildren:()=>import('../tab1-sub-a/tab1-sub-a.module').then(m=>m.Tab1SubAPageModule)}
```

Tab1SubB :

```
{path:'tab1/tab1-sub-b', loadChildren:()=>import('../tab1-sub-b/tab1-sub-b.module').then(m=>m.Tab1SubBPageModule)}
```

Tab2SubA :

```
{path:'tab2/tab2-sub-a', loadChildren:()=>import('../tab2-sub-a/tab2-sub-a.module').then(m=>m.Tab2SubAPageModule)}
```

Tab2SubB :

```
{path:'tab2/tab2-sub-b', loadChildren:()=>import('../tab2-sub-b/tab2-sub-b.module').then(m=>m.Tab2SubBPageModule)}
```

Tab3SubA :

```
{path:'tab3/tab3-sub-a', loadChildren:()=>import('../tab3-sub-a/tab3-sub-a.module').then(m=>m.Tab3SubAPageModule)}
```

Tab3SubB :

```
{path:'tab3/tab3-sub-b', loadChildren:()=>import('../tab3-sub-b/tab3-sub-b.module').then(m=>m.Tab3SubBPageModule)}
```

Tab-based 頁面組內進行跳頁 (2)

```
5  const routes: Routes = [  
6    {  
7      path: 'tabs',  
8      component: TabsPage,  
9      children: [  
10     {path: 'tab1', loadChildren: () => import('../tab1/tab1.module').then(m => m.Tab1PageModule)},  
11     {path: 'tab1/tab1-sub-a', loadChildren: () => import('../tab1-sub-a/tab1-sub-a.module').then(m => m.Tab1SubAPageModule)},  
12     {path: 'tab1/tab1-sub-b', loadChildren: () => import('../tab1-sub-b/tab1-sub-b.module').then(m => m.Tab1SubBPageModule)},  
13     {path: 'tab2', loadChildren: () => import('../tab2/tab2.module').then(m => m.Tab2PageModule)},  
14     {path: 'tab2/tab2-sub-a', loadChildren: () => import('../tab2-sub-a/tab2-sub-a.module').then(m => m.Tab2SubAPageModule)},  
15     {path: 'tab2/tab2-sub-b', loadChildren: () => import('../tab2-sub-b/tab2-sub-b.module').then(m => m.Tab2SubBPageModule)},  
16     {path: 'tab3', loadChildren: () => import('../tab3/tab3.module').then(m => m.Tab3PageModule)},  
17     {path: 'tab3/tab3-sub-a', loadChildren: () => import('../tab3-sub-a/tab3-sub-a.module').then(m => m.Tab3SubAPageModule)},  
18     {path: 'tab3/tab3-sub-b', loadChildren: () => import('../tab3-sub-b/tab3-sub-b.module').then(m => m.Tab3SubBPageModule)},  
19     {path: '', redirectTo: '/tabs/tab1', pathMatch: 'full'}  
20   ]  
21 },  
22 {  
23   path: '',  
24   redirectTo: '/tabs/tab1',  
25   pathMatch: 'full'  
26 }  
27 ];
```

(為方便截圖，本例子以單行編寫各routing path。)

Tab-based 頁面組內進行跳頁 (3)

然而，Tab1需要在畫面裡顯示跳頁次數，而紀錄跳頁次數的變數則定義了在Global Service中，故此，我們需在Tab1的TypeScript檔裡載入Global Service：

```
import { GlobalService } from '../global.service';
```

並在constructor開關細括號中再將其實體化：

```
constructor(public gs:GlobalService){}
```

```
src > app > tab1 > TS tab1.page.ts > ...
1  import { Component } from '@angular/core';
2  import { GlobalService } from '../global.service';
3
4  @Component({
5    selector: 'app-tab1',
6    templateUrl: 'tab1.page.html',
7    styleUrls: ['tab1.page.scss']
8  })
9  export class Tab1Page {
10
11    constructor(public gs:GlobalService){}
12
13 }
```

並加插以下HTML到ion-content內以顯示gs(那隻Global Service實體)中的pageTransited：

```
<div>Page Transited: {{ gs.pageTransited }}</div><br><br>
```

```
9  <ion-content>
10
11    <div>Page Transited: {{ gs.pageTransited }}</div><br><br>
12
13 </ion-content>
```

Tab-based 頁面組內進行跳頁 (4)

來到最後的一個環節，就是把SharedHello放到各頁面。首先我們到每一個屬於Tab的頁面(即Tab1、Tab1SubA、Tab1SubB、Tab2、Tab2SubA、Tab2SubB、Tab3、Tab3SubA及Tab3SubB)的HTML檔加插SharedHello。在加插之前，就讓我們把和頁面的HTML簡化一下，只保留較傳統ion-header和ion-content。

首先是Tab1、Tab2及Tab3的版本：

```
<ion-header>  
  <ion-toolbar>  
    <ion-title>  
      Tab 1  
    </ion-title>  
  </ion-toolbar>  
</ion-header>  
  
<ion-content>  
</ion-content>
```

注意需自行修改**粗色紅色**的Tab號數，屬於**Tab2**的頁面便寫**2**，屬於**Tab3**的頁面便寫**3**。

Tab-based 頁面組內進行跳頁 (5)

然後是Tab1SubA、Tab1SubB、Tab2SubA、Tab2SubB、Tab3SubA及Tab3SubB的版本：

```
<ion-header>  
  <ion-toolbar>  
    <ion-buttons slot="start">  
      <ion-back-button defaultHref="/"></ion-back-button>  
    </ion-buttons>  
    <ion-title>Tab1SubA</ion-title>  
  </ion-toolbar>  
</ion-header>  
  
<ion-content>  
</ion-content>
```

注意需自行修改粗色紅色的Tab號數，屬於**Tab2**的頁面便寫**2**，屬於**Tab3**的頁面便寫**3**，SubA頁面則寫**A**，SubB頁面則寫**B**。

Tab-based 頁面組內進行跳頁 (6)

然後參照Tab1或Tab2或Tab3的.module.ts，到其他每一個屬於Tab的頁面(即Tab1SubA、Tab1SubB、Tab2SubA、Tab2SubB、Tab3SubA及Tab3SubB)的.module.ts檔載入SharedHelloComponentModule：

```
import { SharedHelloComponentModule } from '../shared-hello/shared-hello.module';
```

然後再在@NgModule區域內的imports陣列裡加上SharedHelloComponentModule

以Tab1SubA為例：

```
src > app > tab1-sub-a > TS tab1-sub-a.module.ts > ...
1  import { NgModule } from '@angular/core';
2  import { CommonModule } from '@angular/common';
3  import { FormsModule } from '@angular/forms';
4
5  import { IonicModule } from '@ionic/angular';
6
7  import { Tab1SubAPageRoutingModule } from './tab1-sub-a-routing.module';
8
9  import { Tab1SubAPage } from './tab1-sub-a.page';
10
11 import { SharedHelloComponentModule } from '../shared-hello/shared-hello.module';
12
13 @NgModule({
14   imports: [
15     CommonModule,
16     FormsModule,
17     IonicModule,
18     Tab1SubAPageRoutingModule,
19     SharedHelloComponentModule
20   ],
21   declarations: [Tab1SubAPage]
22 })
23 export class Tab1SubAPageModule {}
24
```

Tab-based 頁面組內進行跳頁 (7)

最後就是在各HTML檔加插SharedHello。

首先是Tab1、Tab2及Tab3的版本：

```
<app-shared-hello pageName="Tab 1 Root" toPageName="Tab 1 Sub A"
toPagePath="tabs/tab1/tab1-sub-a"></app-shared-hello>
```

然後是Tab1SubA、Tab2SubA及Tab3SubA的版本：

```
<app-shared-hello pageName="Tab 1 Sub A" toPageName="Tab 1 Sub B"
toPagePath="tabs/tab1/tab1-sub-b"></app-shared-hello>
```

最後是Tab1SubB、Tab2SubB及Tab3SubB的版本：

```
<app-shared-hello pageName="Tab1 Sub B"></app-shared-hello>
```

注意需自行修改**粗色紅色**的Tab號數，屬於**Tab2**的頁面便寫**2**，屬於**Tab3**的頁面便寫**3**。大家亦可看到，toPagePath是用來決定跳到甚麼頁面的參數，而因為SubB的頁面是沒有跳頁功能的，所以便不需要toPagePath。亦因為是在Tab-based頁面組內進行跳頁，所以路徑需要以**Tab-based**頁面組名稱加斜號(即「tabs/」)開頭。

Tab-based 頁面組 HTML 檔 (1)

tab1.html

```
tab1.page.html x
src > app > tab1 > tab1.page.html > ...
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Tab 1
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content>
10
11   <div>Page Transited: {{ gs.pageTransited }}</div><br><br>
12   <app-shared-hello pageName="Tab 1 Root" toPageName="Tab 1 Sub A" toPagePath="tabs/tab1/tab1-sub-a"></app-shared-hello>
13
14 </ion-content>
15
```

tab2.html

```
tab1.page.html x tab2.page.html x
src > app > tab2 > tab2.page.html > ...
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Tab 2
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content>
10
11   <app-shared-hello pageName="Tab 2 Root" toPageName="Tab 2 Sub A" toPagePath="tabs/tab2/tab2-sub-a"></app-shared-hello>
12
13 </ion-content>
14
```

tab3.html

```
tab1.page.html x tab2.page.html x tab3.page.html x
src > app > tab3 > tab3.page.html > ...
1 <ion-header>
2   <ion-toolbar>
3     <ion-title>
4       Tab 3
5     </ion-title>
6   </ion-toolbar>
7 </ion-header>
8
9 <ion-content>
10
11   <app-shared-hello pageName="Tab 3 Root" toPageName="Tab 3 Sub A" toPagePath="tabs/tab3/tab3-sub-a"></app-shared-hello>
12
13 </ion-content>
14
```

Tab-based 頁面組 HTML 檔 (2)

tab1-sub-a.html

```
tab1-sub-a.page.html x
src > app > tab1-sub-a > tab1-sub-a.page.html > ion-content
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Tab1SubA</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <app-shared-hello pageName="Tab 1 Sub A" toPageName="Tab 1 Sub B" toPagePath="tabs/tab1/tab1-sub-b"></app-shared-hello>
12 </ion-content>
```

tab2-sub-a.html

```
tab2-sub-a.page.html x
src > app > tab2-sub-a > tab2-sub-a.page.html > ion-content > app-shared-hello
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Tab2SubA</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <app-shared-hello pageName="Tab 2 Sub A" toPageName="Tab 2 Sub B" toPagePath="tabs/tab2/tab2-sub-b"></app-shared-hello>
12 </ion-content>
```

tab3-sub-a.html

```
tab3-sub-a.page.html x
src > app > tab3-sub-a > tab3-sub-a.page.html > ion-content > app-shared-hello
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Tab3SubA</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <app-shared-hello pageName="Tab 3 Sub A" toPageName="Tab 3 Sub B" toPagePath="tabs/tab3/tab3-sub-b"></app-shared-hello>
12 </ion-content>
```

Tab-based 頁面組 HTML 檔 (3)

tab1-sub-b.html

```
tab1-sub-b.page.html X
src > app > tab1-sub-b > tab1-sub-b.page.html > ion-content > app-shared-hello
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Tab1SubB</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <app-shared-hello pageName="Tab1 Sub B"></app-shared-hello>
12 </ion-content>
```

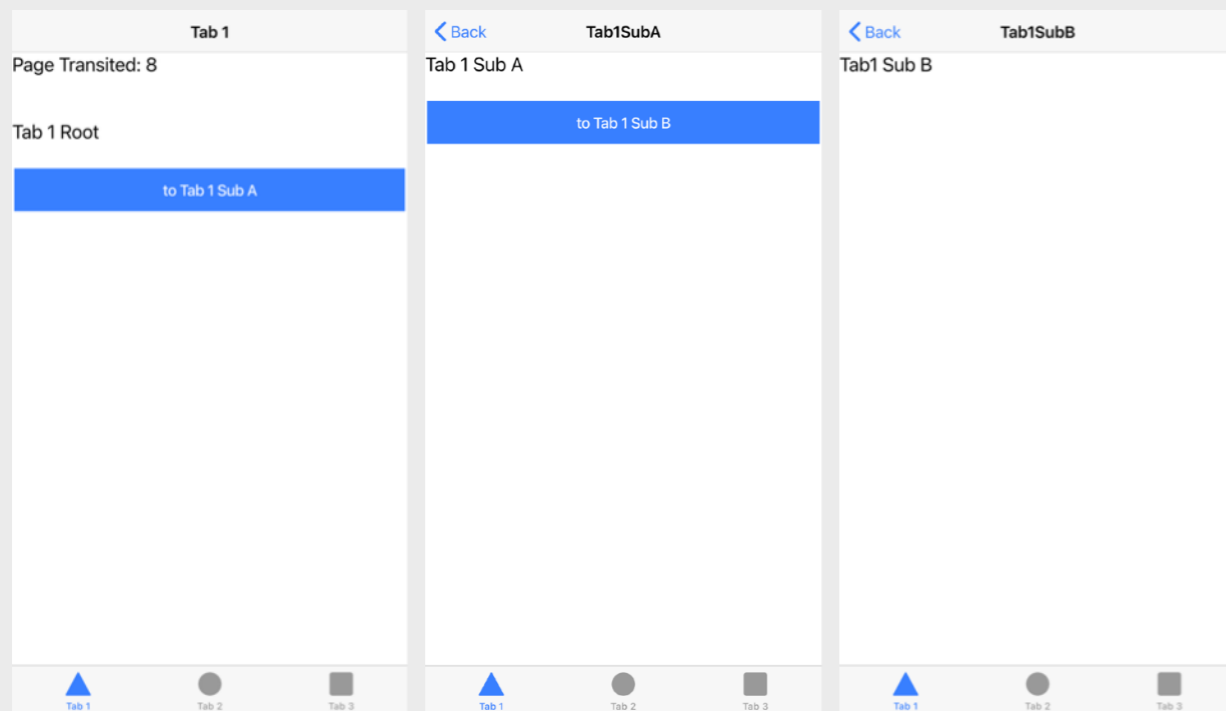
tab2-sub-b.html

```
tab2-sub-b.page.html X
src > app > tab2-sub-b > tab2-sub-b.page.html > ion-content
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Tab2SubB</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <app-shared-hello pageName="Tab2 Sub B"></app-shared-hello>
12 </ion-content>
```

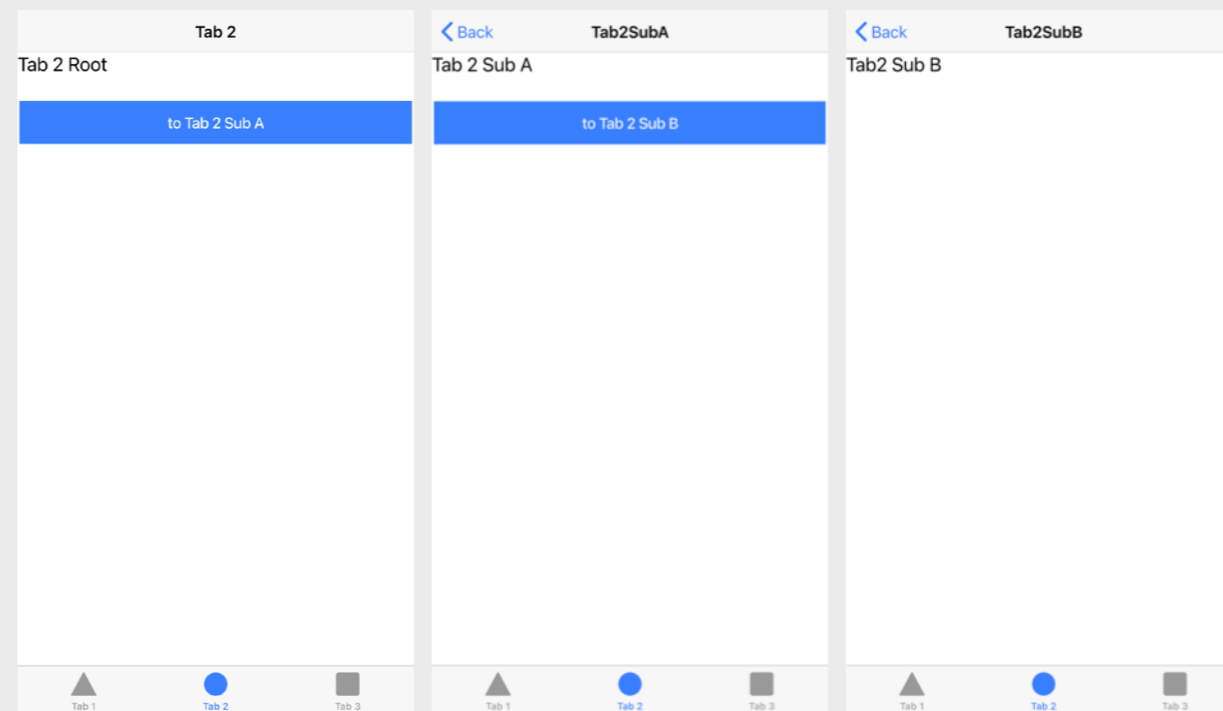
tab3-sub-b.html

```
tab3-sub-b.page.html X
src > app > tab3-sub-b > tab3-sub-b.page.html > ion-content > app-shared-hello
1 <ion-header>
2   <ion-toolbar>
3     <ion-buttons slot="start">
4       <ion-back-button defaultHref="/"></ion-back-button>
5     </ion-buttons>
6     <ion-title>Tab3SubB</ion-title>
7   </ion-toolbar>
8 </ion-header>
9
10 <ion-content>
11   <app-shared-hello pageName="Tab3 Sub B"></app-shared-hello>
12 </ion-content>
```

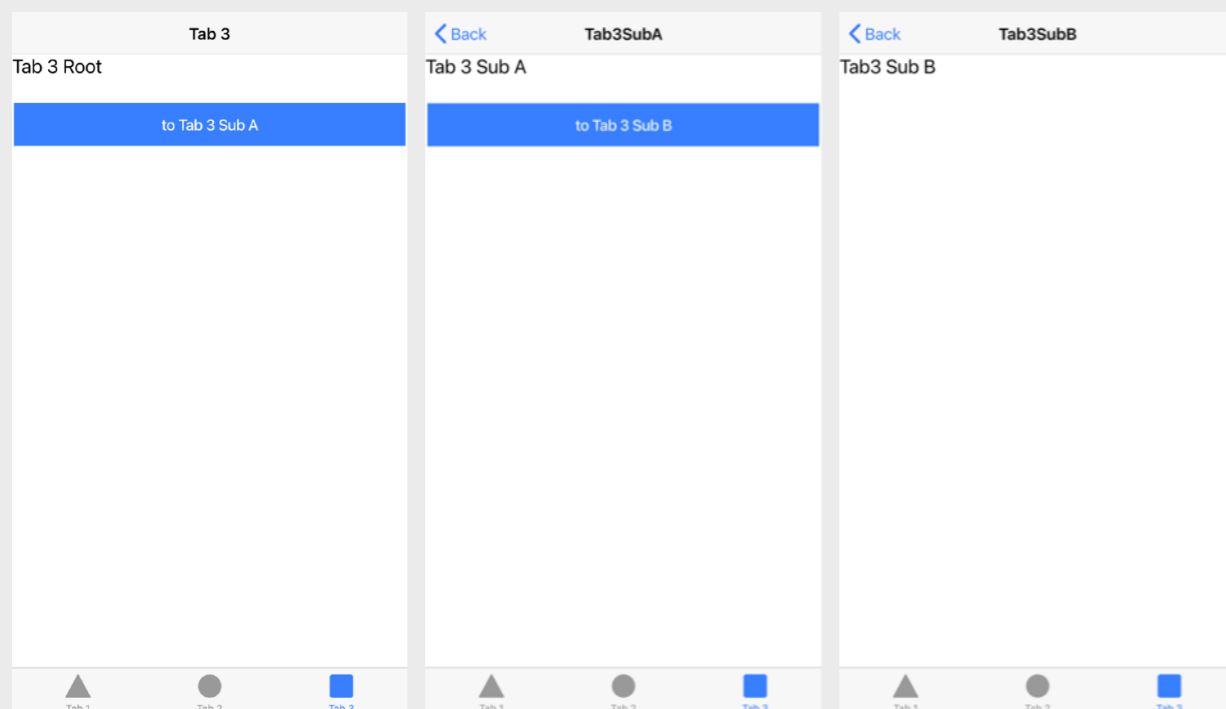

Tab-based 程式大功告成



Tab1頁面組



Tab2頁面組



Tab3頁面組

完整版本程式碼下載：

<https://s3objectstorage-a.ap-south-1.linodeobjects.com/TabsLab-Final.zip>

練習答案

```
<!DOCTYPE html>
<html>
  <head>
    <title>Welcome<title>
  </head>
  <body onload="bodyLoaded();" onclick=bodyClicked();>
    <TABLE>
      <tr>
        <td style="font-weight:bold;">Name</td>
        <td style="font-weight:bold;">Education</td>
      </tr>
      <tr>
        <td>John</td>
        <td>BSc, MEng, MBA, PhD</td>
      </tr>
      <tr>
        <td>Lily</td>
        <td>BSocSci, MA, MPhil, EdD</td>
      </tr>
      <tr>
        <td>Kay</td>
        <td>BComp, MSc, DEng, DBA</td>
      </tr-end>
    </table>
  </body>
</html>
```

HTML練習1

答案

以下有四個錯誤：

1. 關<title>需要斜號
2. Tag屬性值需要雙引號
3. TABLE不應大寫
4. </tr-end>應寫成</tr>

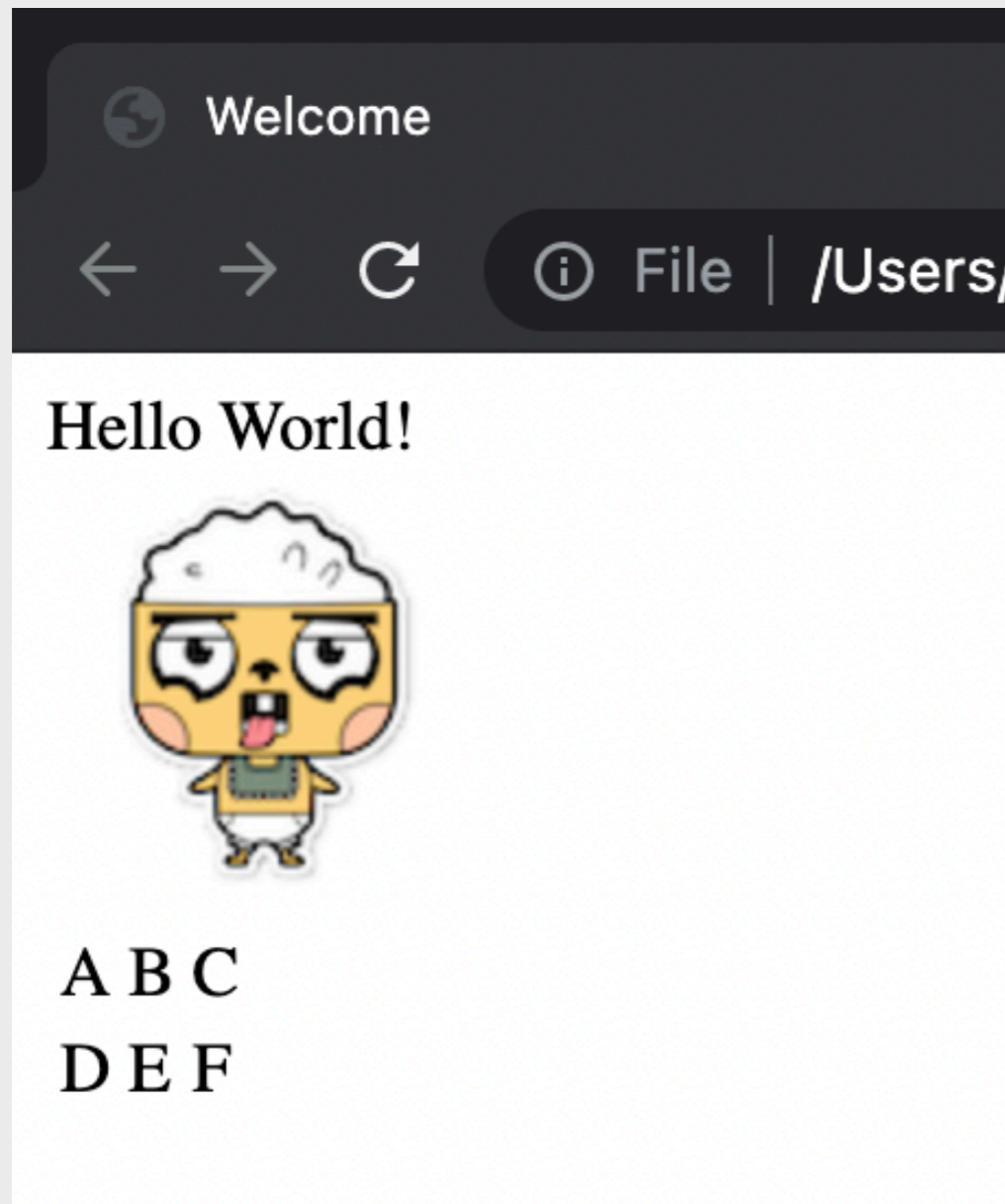
HTML練習2答案

請填寫漏空處。

```
<!DOCTYPE html>
<html>
  <head>
    <title>Welcome</title>
  </head>
  <body>
    <div><span>Testing</span></div>
    <br>
    
  </body>
</html>
```

HTML練習3答案

請根據以下圖片來編寫出HTML。(提示：圖片下方的ABCDEF為table)



```
<!DOCTYPE html>
<html>
  <head>
    <title>Welcome</title>
  </head>
  <body>
    Hello World!
    <br>
    
    <table>
      <tr><td>A</td><td>B</td><td>C</td></tr>
      <tr><td>D</td><td>E</td><td>F</td></tr>
    </table>
  </body>
</html>
```


CSS練習1答案

```
<!DOCTYPE html>
<html>
  <head>
    <title>Animals</title>
    <style>
      div{font-size:18pt;}
      .MyPets{color:green;}
      .OnlineAdopted{color:gold;}
      #beeno{text-decoration:underline;}
      #kitty{background-color:#73FDFF;}
      body{font-weight:bold;}
      .MyPets, .OnlineAdopted{
        border: 2px solid #922222;
        padding: 12px;
      }
      .MyPets, #ada{margin-left:5.5vw;}
    </style>
  </head>
  <body>
    <div class="MyPets" id="beeno">Dog</div>
    <div class="MyPets" id="kitty">Cat</div>
    <div class="OnlineAdopted">Penguin</div>
    <div>Panda</div>
    <div class="OnlineAdopted">Koala</div>
    <div id="ada">Monkey</div>
  </body>
</html>
```

1. 把所有的div的font-size設定為18pt
2. 把所有屬於MyPets類別的div的color設定為green
3. 把所有屬於OnlineAdopted類別的div的color設定為gold
4. 把id為beeno的div加上底線(text-decoration設定為underline)
5. 把id為kitty的div的背景顏色(background-color)設定為#73FDFF
6. 把body的字體粗度(font-weight)設定為粗體(bold)
7. 把所有屬於MyPets或OnlineAdopted類別的div的border設定為2px solid #922222及padding為12px
8. 把所有屬於MyPets或id為ada的div的margin-left設定為5.5vw

CSS練習2答案

```
body {
  margin: 0;
  font-family: Arial, Helvetica, sans-serif;
  background-color: #D3F5FA;
}

#img_banner {
  width: 100%;
}

#div_title {
  margin-top: 2.5vh;
  font-weight: bold;
  font-size: 20pt;
  text-align: center;
}

#table_form {
  padding: 0;
  margin: 0 auto;
  margin-top: 6vh;
  font-size: 13pt;
}

#table_form tr {
  background-color: transparent;
  transition: background-color 0.3s ease;
}

#table_form tr:hover {
  background-color: rgb(240, 240, 240);
}

#table_form td {
  padding: 6pt 2vw;
}
```

```
#table_form input,
#table_form textarea {
  border-radius: 6pt;
  border: 0;
  font-size: 13pt;
}

#table_form tr:last-child td {
  text-align: center;
}

#div_message{
  width: 50%;
  font-size: 12pt;
  margin: 6vh auto;
  text-align: justify;
}

#div_message div{
  font-weight: bold;
  font-size: 18pt;
  text-align: center;
  margin-bottom: 1vh;
}

#img_ad{
  position: fixed;
  width: 100pt;
  right: 3vw;
  bottom: 30vh;
  transition: transform 0.2s ease;
  border-radius: 50%;
}

#img_ad:hover{
  transform: scale(1.3);
}
```


CSS練習3答案

```
/* 當裝置寬度在414px之內的樣式 */  
@media screen and (max-width: 414px) {  
  .Div_Container>div {  
    width: 94.5vw;  
    min-height: 103vw;  
    margin: 0 auto;  
  }  
  .Div_Container>div div:first-child div {  
    font-size: 12pt;  
    padding: 2pt;  
  }  
}
```

JavaScript練習1答案

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>JavaScript練習1</title>
</head>
<body>
  <script>
    setTimeout(()=>alert('Have a nice day!'),1500);
  </script>
</body>
</html>
```

JavaScript練習2答案

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>JavaScript Ex2</title>
</head>
<body>
  <script>
    let a=1, b=2, c;
    c=a+b;
    document.write('c='+c);
  </script>
</body>
</html>
```

JavaScript練習3答案

1.

```
const PI=3.1416;
```

2.

```
const emperorName='康熙';
```

3.

```
let dynastyName='Qing', emperorsCount=12, hasPrimeMinister=false;
```

4.

Error

JavaScript練習4答案

```
const prdA=24, prdB=33, prdC=18, prdD=5;
let totalPrice=0;
totalPrice=(3*prdA + 2*prdB + 5*prdC);
document.write(totalPrice);
document.write('<br>');
document.write(50-totalPrice);
totalPrice+=prdD;
document.write('<br>');
document.write(totalPrice);
document.write('<br>');
document.write(1000-totalPrice);
```

JavaScript練習5答案

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>JavaScript Ex2</title>
</head>
<body>
  輸入一個以ml(毫升)為單位的數字:<br>
  <input id="input_ml" type="number"><br>
  <button onclick="cal();">計算</button>
  <div id="div_ans"></div>
  <script>
    function cal(){
      let ans=document.getElementById('input_ml').value;
      ans=ans/1000;
      document.getElementById('div_ans').innerHTML='答案是：'+ans+'L';
    }
  </script>
</body>
</html>
```

JavaScript練習6答案

```
<script>  
  let value=prompt('請問你多少歲?');  
  document.write(value<=12 || value>=65);  
</script>
```

JavaScript練習7答案

```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <title>JavaScript Ex7</title>
</head>

<body>
  <script>
    const num1=parseFloat(prompt('客人於「手袋類」產品總共購買了多少錢?'));
    const num2=parseFloat(prompt('客人於「飾物類」產品總共購買了多少錢?'));
    const num3=parseFloat(prompt('客人於「時裝類」產品總共購買了多少錢?'));
    let total=num1+num2+num3;
    if(total>=25000)total*=0.85;
    else total-=200;
    document.write('購買總值為HK$'+Math.ceil(total));
  </script>
</body>

</html>
```


JavaScript練習8答案

```
<!DOCTYPE html>
<html>

<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
  <title>JavaScript Ex8</title>
</head>

<body>
  <script>
    const num=prompt('請輸入數值:');
    document.write('請輸入數值: '+num+'<br>');
    if(num%5==0 && num%3==0)document.write(num+"可被5和3除盡");
    else if(num%5==0)document.write(num+"可被5除盡");
    else if(num%3==0)document.write(num+"可被3除盡");
    else document.write(num+"不可被5和3除盡");
  </script>
</body>

</html>
```

JavaScript練習9答案

```
document.write('歡迎使用！<br>');
const num1 = prompt('客人於「手袋類」產品總共購買了多少錢？');
document.write('客人於「手袋類」產品總共購買了多少錢？'+num1+'<br>');
const num2 = prompt('客人於「飾物類」產品總共購買了多少錢？');
document.write('客人於「飾物類」產品總共購買了多少錢？'+num2+'<br>');
const num3 = prompt('客人於「時裝類」產品總共購買了多少錢？');
document.write('客人於「時裝類」產品總共購買了多少錢？'+num3+'<br>');
let total = parseInt(num1) + parseInt(num2) + parseInt(num3);

const isVIP = confirm('客人是VIP會員嗎？');
document.write('客人是VIP會員嗎？'+isVIP+'<br>');

if (isVIP) {
    const vipType=prompt('請輸入客人的VIP種類:').toUpperCase();
    document.write('請輸入客人的VIP種類: '+vipType+'<br>');
    if (vipType=='C') total *= 0.9;
    else if (vipType=='B') total *= 0.8;
    else if (vipType=='A') total *= 0.7;
    else total *= 0.95;
} else {
    const firstBuy = confirm('客人是第一次購物嗎？');
    document.write('客人是第一次購物嗎？'+firstBuy+'<br>');
    if (firstBuy) total -= 200;
    else {
        const secondBuy = confirm('客人是第二次購物嗎？');
        document.write('客人是第二次購物嗎？'+secondBuy+'<br>');
        if (secondBuy) total -= 100;
        else {
            if (total >= 30000) total -= 50;
        }
    }
}

document.write('購買總值為HK$' + Math.ceil(total));
```

JavaScript練習10答案

```
<body>
  <script>
    for(let i=1; i<=9; i++){
      document.write(i+' x '+i+' = '+i*i+'<br>');
    }
  </script>
</body>
```

JavaScript練習11答案

```
<body>
  <script>
    for(let i=1; i<=10; i++){
      if(i%2==0)document.write('1到10中的偶數有：'+i+'<br>');
    }
  </script>
</body>
```

JavaScript練習12答案

```
<body>
  <input id="input_1" type="number"><br>
  <button onclick="tryCal()">Submit</button>
  <script>
    function tryCal(){
      const times=parseInt(document.getElementById('input_1').value);
      for(let i=0; i<times; i++){
        document.write('hello<br>');
      }
    }
  </script>
</body>
```

JavaScript練習13答案

```
<body>
  <script>
    for (let i=1; i<=6; i++){
      for (let j=1; j<=i; j++){
        document.write(i*j+' ');
      }
      document.write('<br>');
    }
  </script>
</body>
```

JavaScript練習14答案

```
<body>
  <script>
    const steps=parseInt(prompt('小龜🐢要行走多少步?'));
    document.write('小龜🐢要行走多少步? '+steps+'<br>');
    const cakes=parseInt(prompt('小龜每步吃多少次蛋糕🍰?'));
    document.write('小龜每步吃多少次蛋糕🍰? '+cakes+'<br>');
    const selectedStep=parseInt(prompt('小龜在吃完哪一步的哪一件蛋糕後要多吃一隻月餅🥮? \n先輸入步數:'));
    document.write('小龜在吃完哪一步的哪一件蛋糕後要多吃一隻月餅🥮? <br>先輸入步數: '+selectedStep+'<br>');
    const selectedCake=parseInt(prompt('再輸入在吃完哪一件蛋糕後吃月餅:'));
    document.write('再輸入在吃完哪一件蛋糕後吃月餅: '+selectedCake+'<br>');
    for(let i=1; i<=steps; i++){
      document.write('第'+i+'步: ');
      for(let j=1; j<=cakes; j++){
        document.write('🍰');
        if(selectedStep==i && j==selectedCake)document.write('🥮');
      }
      document.write('<br>');
    }
  </script>
</body>
```

JavaScript練習15答案

```
<body>
  <script>
    const endStep=parseInt(prompt('小龜🐢要行多少步?'));
    document.write('小龜🐢要行多少步? '+endStep+'<br>');
    let counter=0;
    while(counter<endStep){
      counter++;
      document.write('行呀行，行到第'+counter+'步了！<br>');
    }
  </script>
</body>
```


JavaScript練習16答案

```
<body>
  <script>
    const quest='小龜🐢是否向前行嗎? (輸入y代表是)';
    let steps=0;
    let walk=(prompt(quest).toLowerCase()=='y');
    console.log(quest+' : '+walk);
    while(walk){
      steps++;
      console.log('行第'+steps+'步 ! ');
      if(steps==7){
        console.log('7步成詩了 ! ');
        break;
      }
      walk=(prompt(quest).toLowerCase()=='y');
      console.log(quest+' : '+walk);
    }
  </script>
</body>
```

JavaScript練習17答案

```
<body>
  <script>
    let i=1;
    let hasNextRound=false;
    do{
      const quest='小龜🐢進食了第'+i+'轉的食物，牠繼續
想拿取下一轉的食物嗎？(輸入y代表是)';
      hasNextRound=(prompt(quest).toLowerCase()=='y');
      console.log(quest+'：'+hasNextRound);
      i++;
    }while(hasNextRound);
    console.log('小龜🐢吃飽了，滿足😊');
  </script>
</body>
```

條件判斷練習一答案 (1)

以下是一條陣列，名為「teachers」

```
teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},  
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},  
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4]}}  
];
```

Q1. 若要把teacher陣列中的物件逐個處理，該用哪種條件判斷式？

該以 For Loop 處理，但用 While Loop 亦可以

條件判斷練習一答案 (2)

以下是一條陣列，名為「teachers」

```
teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},  
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},  
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4]}}  
];
```

Q2. 把每個老師的名字(name)以 console.log 方式列出來。

```
for(let i=0; i<teachers.length; i++){  
  console.log(teachers[i].name);  
}
```

條件判斷練習一答案 (3)

以下是一條陣列，名為「teachers」

```
teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},  
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},  
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4]}}  
];
```

Q3. For Loop 與 While Loop 有甚麼分別？

For Loop 明確知道要執行迴圈多少次，
While Loop 不確定要執行迴圈多少次

For Loop 需於條件定義中定義索引初始值、條件判斷式及索引在結束單次迴圈後變動的方式。而 While Loop 在結束單次迴圈後的變動方式則需在迴圈內定義

條件判斷練習一答案 (4)

以下是一條陣列，名為「teachers」

```
teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},  
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},  
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4}]},  
];
```

Q4. 把 id 大於 2 的老師名字列出來。

```
for(let i=0; i<teachers.length; i++){  
  if(teachers[i].id>2){  
    console.log(teachers[i].name);  
  }  
}
```

條件判斷練習一答案 (5)

以下是一條陣列，名為「teachers」

```
teachers=[
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4}]}
];
```

Q5. 運用 While Loop 把所有老師名字列出來。

```
let i=0;
while(i<teachers.length){
  console.log(teachers[i].name);
  i++;
}
```

條件判斷練習一答案 (6)

以下是一條陣列，名為「teachers」

```
teachers=[  
  {"id":1, "name":"Ada", "edu":["BEng","MSc"], "courses":[{"code":"COMP101","level":1}]},  
  {"id":2, "name":"Ben", "edu":["BSc","MComp","DEng"], "courses":[{"code":"COMP403","level":4}]},  
  {"id":3, "name":"Kay", "edu":["BBA","MBA","DBA"], "courses":[{"code":"BUS301","level":3}]},  
  {"id":4, "name":"Leo", "edu":["BBA","MMkt","PhD"], "courses":[{"code":"MKT402","level":4]}}  
];
```

Q6. (a) 把擁有BBA學歷的老師名字列出來。 (b) 把教授 Level 4 的老師名字列出來。

```
(a)   for(let i=0; i<teachers.length; i++){  
      for(let k=0; k<teachers[i].edu.length; k++){  
        if(teachers[i].edu[k]=='BBA')console.log(teachers[i].name);  
      }  
    }
```

```
(b)   for(let i=0; i<teachers.length; i++){  
      for(let k=0; k<teachers[i].courses.length; k++){  
        if(teachers[i].courses[k].level==4)console.log(teachers[i].name);  
      }  
    }
```


條件判斷練習二答案 (1)

以下是一條鍵值配，名為「university」

```
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]},  
schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}],  
presidentProfile:{"name":"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}  
};
```

Q1. 列出多於或等於 200 位職員的學院(school)的名字(name)

```
for(let i=0; i<university.schools.length; i++){  
  if(university.schools[i].staffCount>=200){  
    console.log(university.schools[i].name);  
  }  
}
```

條件判斷練習二答案 (2)

以下是一條鍵值配，名為「university」

```
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]}, schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}], presidentProfile:{"name":"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}};
```

Q2. 列出B大樓(blkB)的座標，座標以逗號分隔

```
for(let blk in university.buildings){  
  if(blk=='blkB')console.log(university.buildings[blk][0]+' '+university.buildings[blk][1]);  
}
```

或

```
for(let blk in university.buildings){  
  if(blk=='blkB'){  
    let output="";  
    for(let i=0; i<university.buildings[blk].length; i++){  
      if(i==0)output=university.buildings[blk][i];  
      else output=output+' '+university.buildings[blk][i];  
    }  
    console.log(output);  
  }  
}
```

條件判斷練習二答案 (3)

以下是一條鍵值配，名為「university」

```
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]}, schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}], presidentProfile:{"name":"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}};
```

Q3. 使用 For Loop 把校長的學歷倒序列出來

```
for(let i=university.presidentProfile.edu.length-1; i>=0; i--){  
  console.log(university.presidentProfile.edu[i]);  
}
```

條件判斷練習二答案 (4)

以下是一條鍵值配，名為「university」

```
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]}, schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}], presidentProfile:{"name":"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}};
```

Q4. 計出各學院員工人數的總和

```
let totalHeadCount=0;
for(let i=0; i<university.schools.length; i++){
  totalHeadCount+=university.schools[i].staffCount;
}
console.log(totalHeadCount);
```

條件判斷練習二答案 (5)

以下是一條鍵值配，名為「university」

```
university={"name":"App Dev University", buildings:{blkA:[22.280878,114.044674],blkB:[22.292451,114.043912]}, schools:[{"name":"Science","staffCount":200}, {"name":"Arts","staffCount":166}, {"name":"Business","staffCount":323}], presidentProfile:{name:"Dr. Jo","edu":["BSc","MCompSci","MEd","DEng"]}};
```

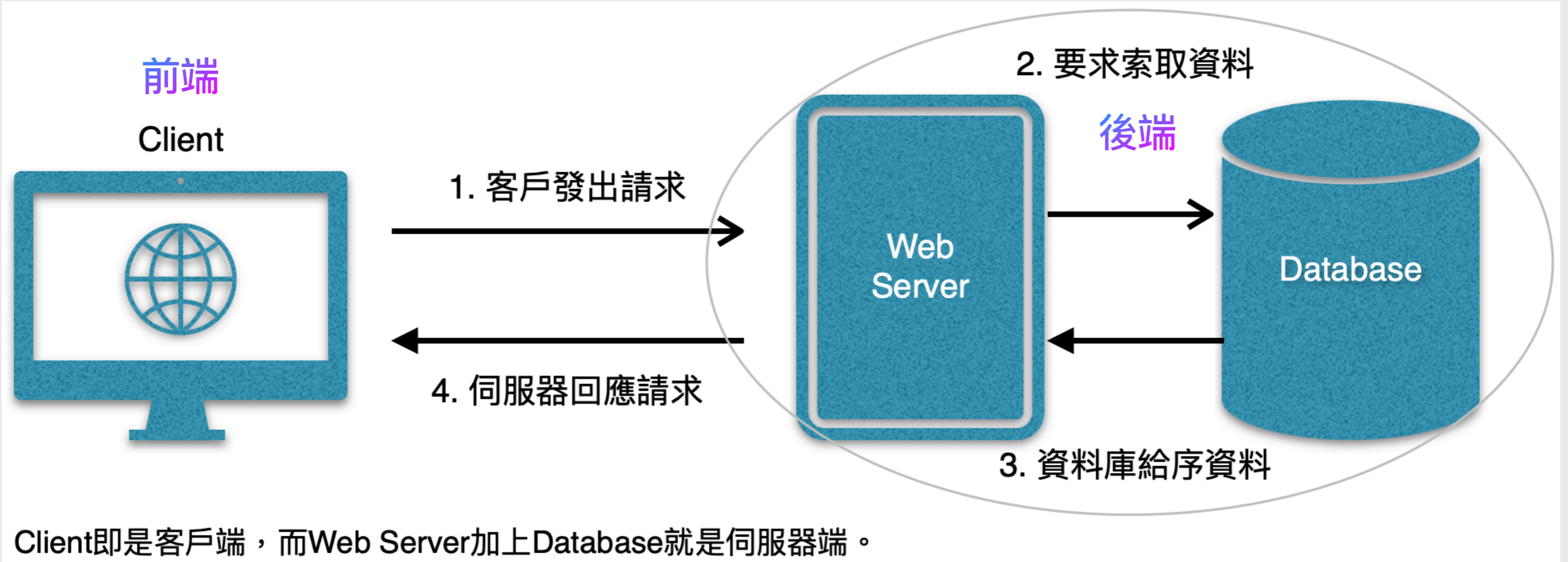
Q5 (挑戰題). 使用 `charAt(0)` 把校長首字元為「M」的學歷列出來

```
for(let i=0; i<university.presidentProfile.edu.length; i++){  
  if(university.presidentProfile.edu[i].charAt(0)=='M')console.log(university.presidentProfile.edu[i]);  
}
```

後端開發

「前端與後端」概念

溫故知新



主從式系統(Client-Server System)的意思是使用者(前端/客戶端)透過網路來連接到伺服器(後端/伺服器端)並獲取所需服務。例如使用社交媒體時，用戶透過電腦或流動裝置登上社交媒體的網站從而獲得資訊(如朋友的發文)，又或是對朋友的留言「讚好」；這都是需要使用者使用程式連接到伺服器讓伺服器完成指定任務並從伺服器獲得所需的回覆。而正確來說，這個過程就是：

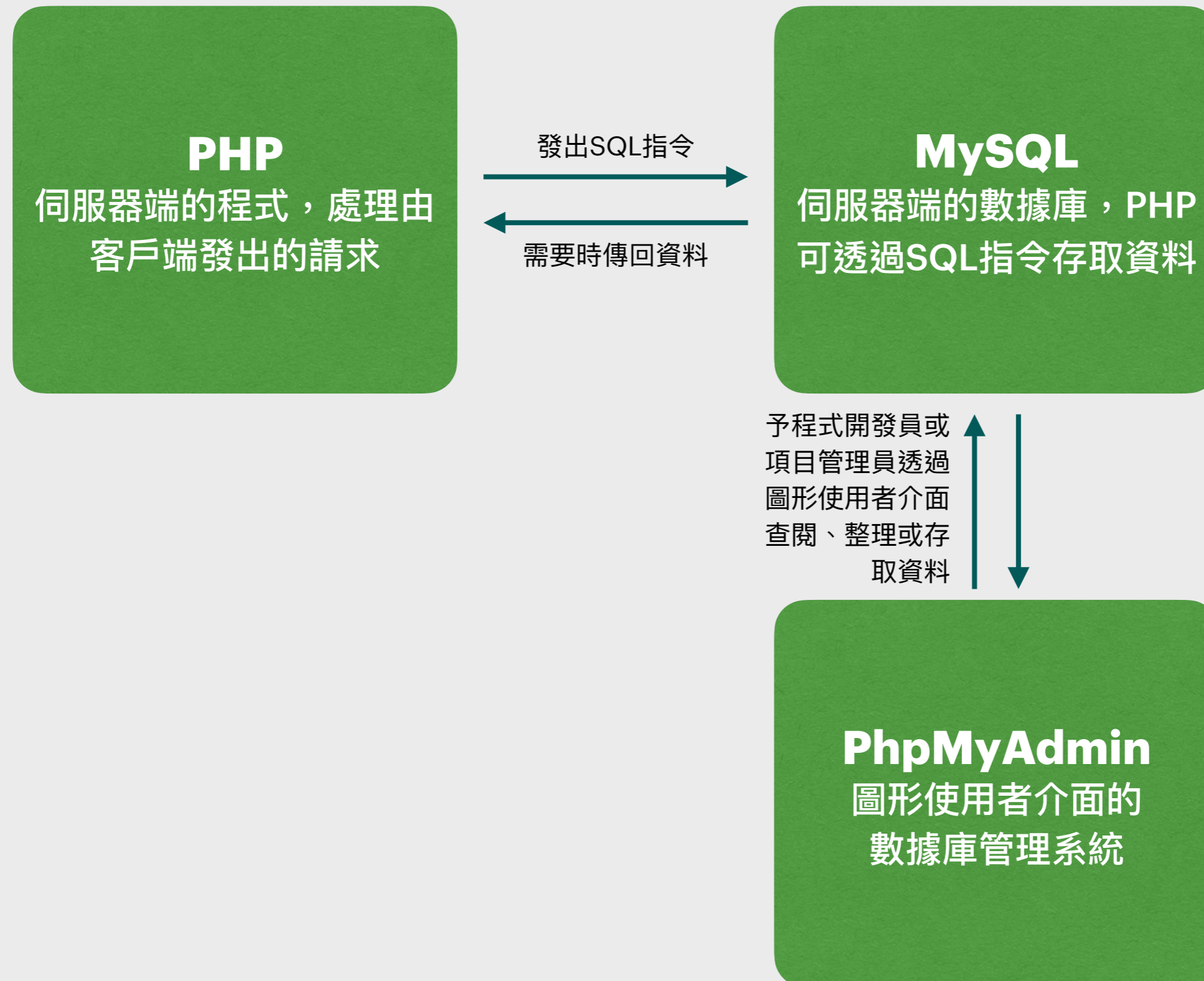
客戶向伺服器發出請求 -> 伺服器進行邏輯運算(及存取資料庫) -> 伺服器回覆客戶

伺服器端配置方案

網站種類	所需程式語言	普遍配用的數據庫	普遍配用的伺服器	簡介
PHP	PHP	MySQL, PostgreSQL	Windows: AppServ, WAMP; Mac: MAMP; Linux: LAMP	架站相對容易，PHP程式語言亦比較易學，而且免費，功能齊備，適合開發中小型系統
JSP及Serverlet	JSP Script, Java	Oracle Database, MySQL, PostgreSQL	Tomcat, GlassFish	免費，但開發門檻較高，需對Java、物件導向程式概念及MVC概念有一定認識；適合開發中型大型系統
ASP.Net	C# (主流) 或 Visual Basic	Microsoft SQL Server	Windows .NET Server	分有免費和不同的收費版本，但只能運行於Windows伺服器，而且開發門檻較高，需對C#、物件導向程式概念及MVC概念有一定認識；適合開發中型大型系統
Node.js	JavaScript	MongoDB, SQLite, MySQL, PostgreSQL	Node.js	免費，但因為Node.js並非單純的網站伺服器，開發門檻高，需對JavaScript、物件導向程式有基礎認識

而建造及開發主從式手機程式或網站，在選擇伺服器端則有很多種的方案，比較簡單和普遍的就是架設PHP、JSP或ASP.Net的網站。

PHP、MySQL與PhpMyAdmin的關係 (1)



PHP、MySQL與PhpMyAdmin的關係 (2)

PHP 簡介：

PHP (Hypertext Preprocessor) 是一種開源的服務器端腳本語言，廣泛用於網頁開發。由於其語法簡單、學習曲線低，使得它成為初學者入門伺服器端編程的首選語言之一。

PHP 的主要特點包括：

- 開源性：PHP 是開源的，意味著它免費提供給用戶，並允許用戶查看、修改和擴展源代碼。
- 跨平台：PHP 可以在多種作業系統上運行，如 Windows、Linux 和 macOS。
- 嵌入性：PHP 代碼可以嵌入到 HTML 代碼中，通過服務器預處理後生成動態內容。
- 資料庫支持：PHP 支持多種資料庫連接，其中最常見的是 MySQL。

MySQL 簡介：

MySQL 是一種流行的關聯式資料庫管理系統 (RDBMS)，以其穩定性、可靠性和高性能而聞名。它是基於 SQL (Structured Query Language) 的系統，用於管理和操作結構化數據。MySQL 的特性包括：

- 開源性：MySQL 提供開源版本，社群可以使用和修改。
- 廣泛的應用：從小型個人項目到大型企業級應用，MySQL 都能夠勝任。
- 支持標準SQL：它使用標準的 SQL 數據語言形式。
- 性能：MySQL 提供了高性能的數據存儲和檢索機制。

PHP、MySQL與PhpMyAdmin的關係 (3)

PHPMyAdmin 簡介：

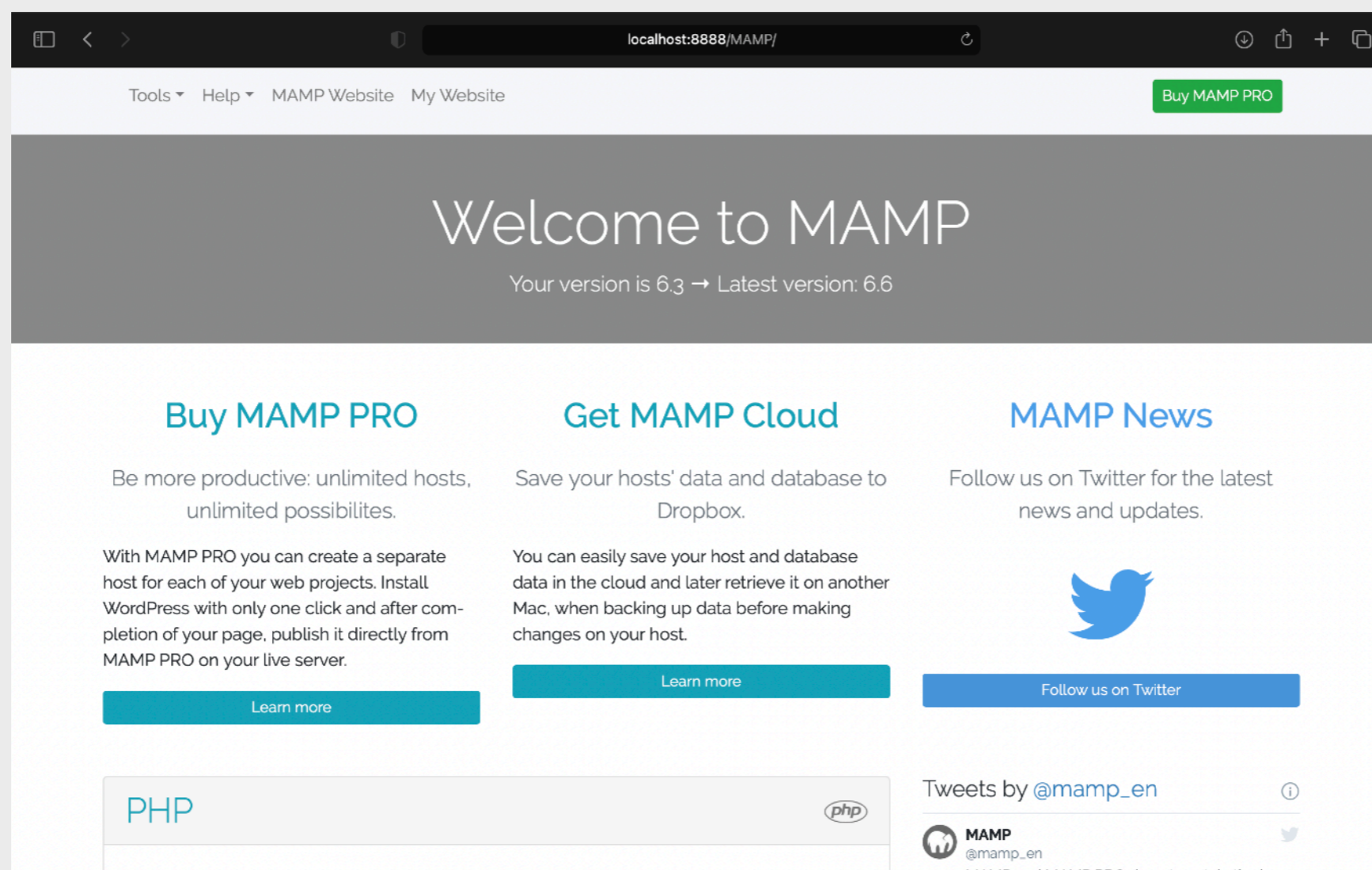
PHPMyAdmin 是一個以 PHP 為基礎的開源工具，用於通過網頁介面管理 MySQL 數據庫。它使得用戶可以輕鬆地執行各種數據庫操作，無需深入了解 SQL 語法。

PHPMyAdmin 的主要功能包括：

- 使用者友好的界面：通過網頁界面，用戶可以直觀地操作數據庫，如建立資料庫、執行 SQL 查詢等。
- 豐富的功能：提供資料庫管理的全套工具，包括資料庫導入導出、備份恢復、權限管理等。
- 多語言支持：支持多種語言，讓不同語言的使用者都能使用。

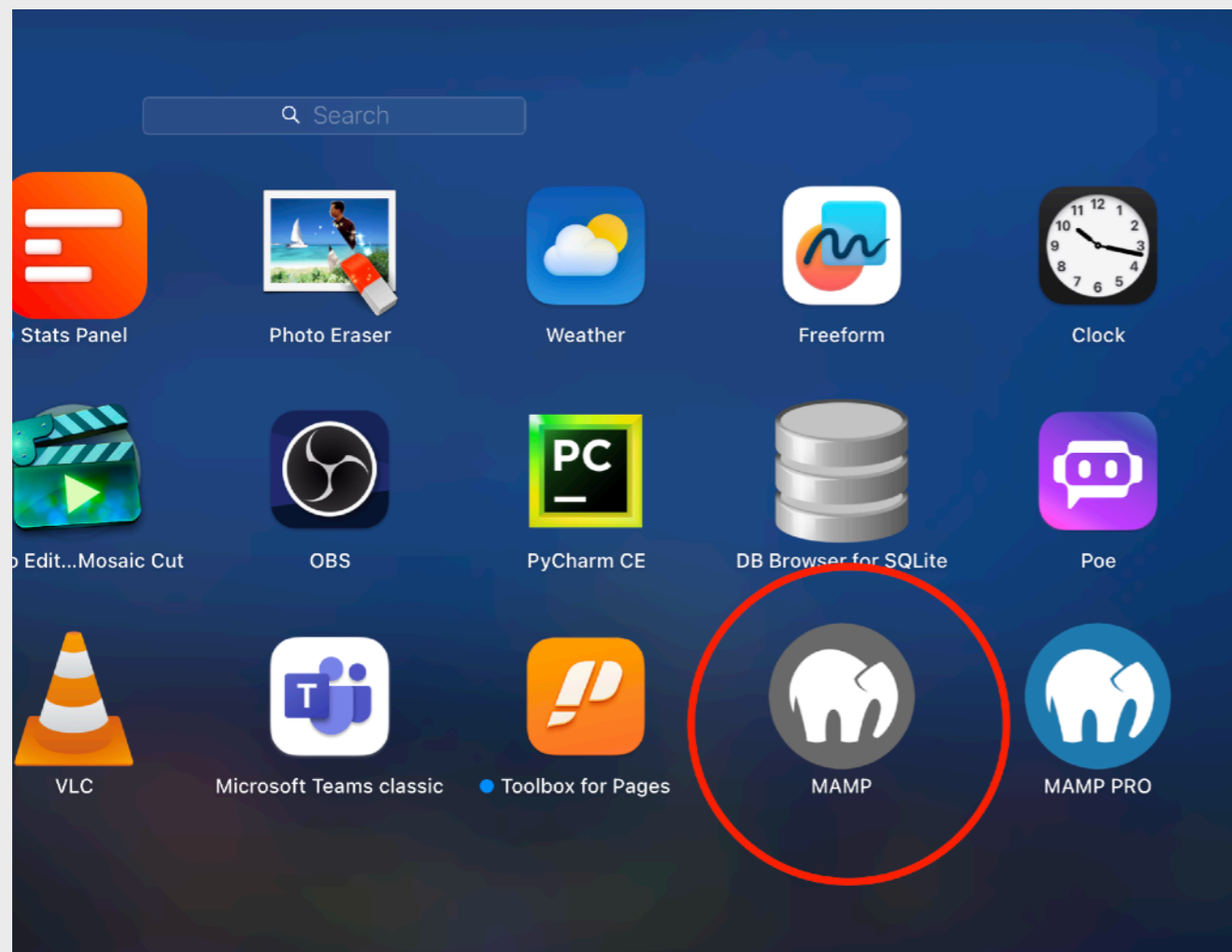
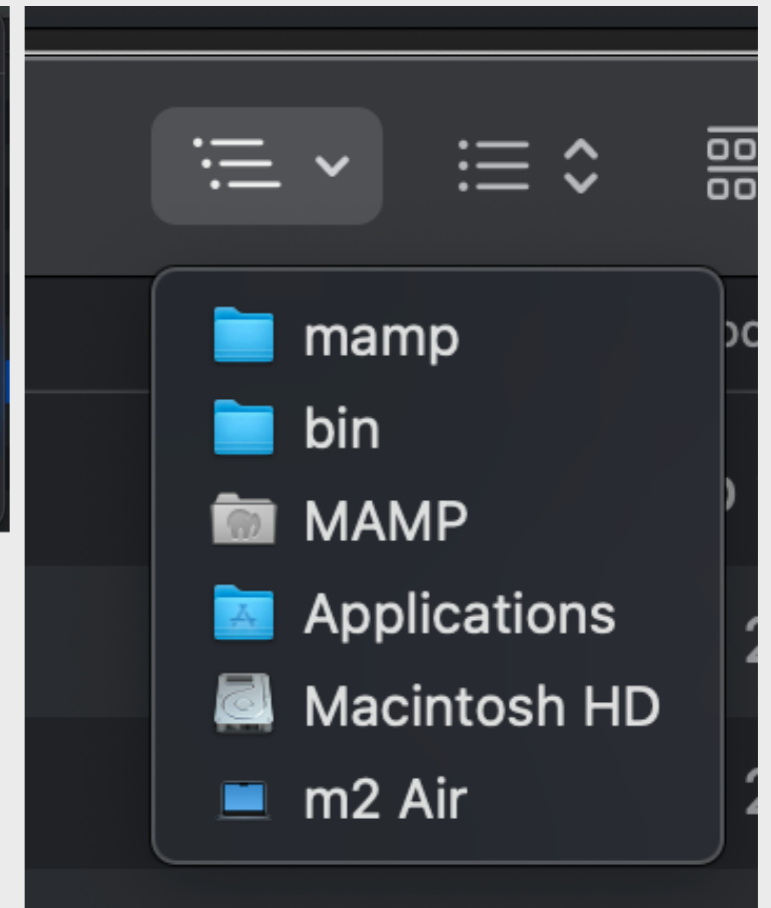
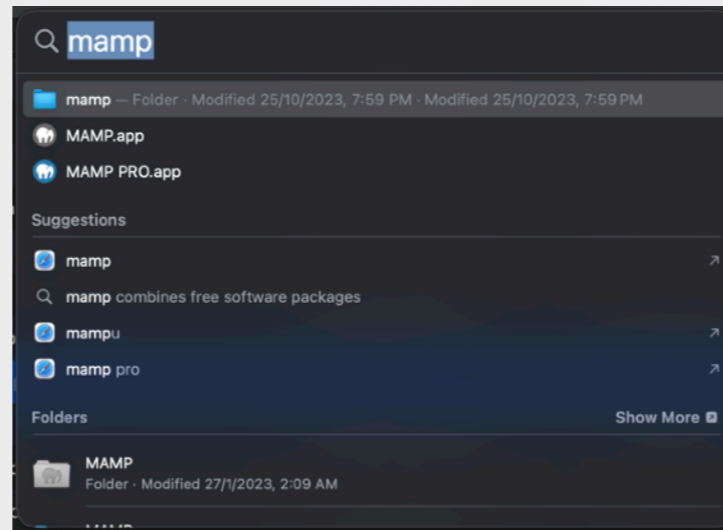
PHP、MySQL 和 PHPMyAdmin 的組合是開發現代動態網站和網路應用的強大工具。開發者可以利用 PHP 來處理前端與後端的交互，利用 MySQL 存儲數據，並通過 PHPMyAdmin 來簡化數據庫的管理工作。這樣的組合不僅提高了開發效率，同時也降低了學習和使用複雜數據庫系統的門檻。

於Mac電腦上，我們可以下載並安裝比較知名的MAMP安裝包，基本功能是免費的，包含了Apache伺服軟件、MySQL數據庫和phpMyAdmin圖像使用者界面的數據庫管理網頁系統，相信其免費版已足夠初學者或個人開發者使用，官方網站為: <https://www.mamp.info/en/> 安裝後開啟瀏覽器並跟據Port位連結到 <http://localhost:8888/MAMP/> 測試，如成功則會看到下圖畫面。



在Mac上安裝伺服器以運行Web App

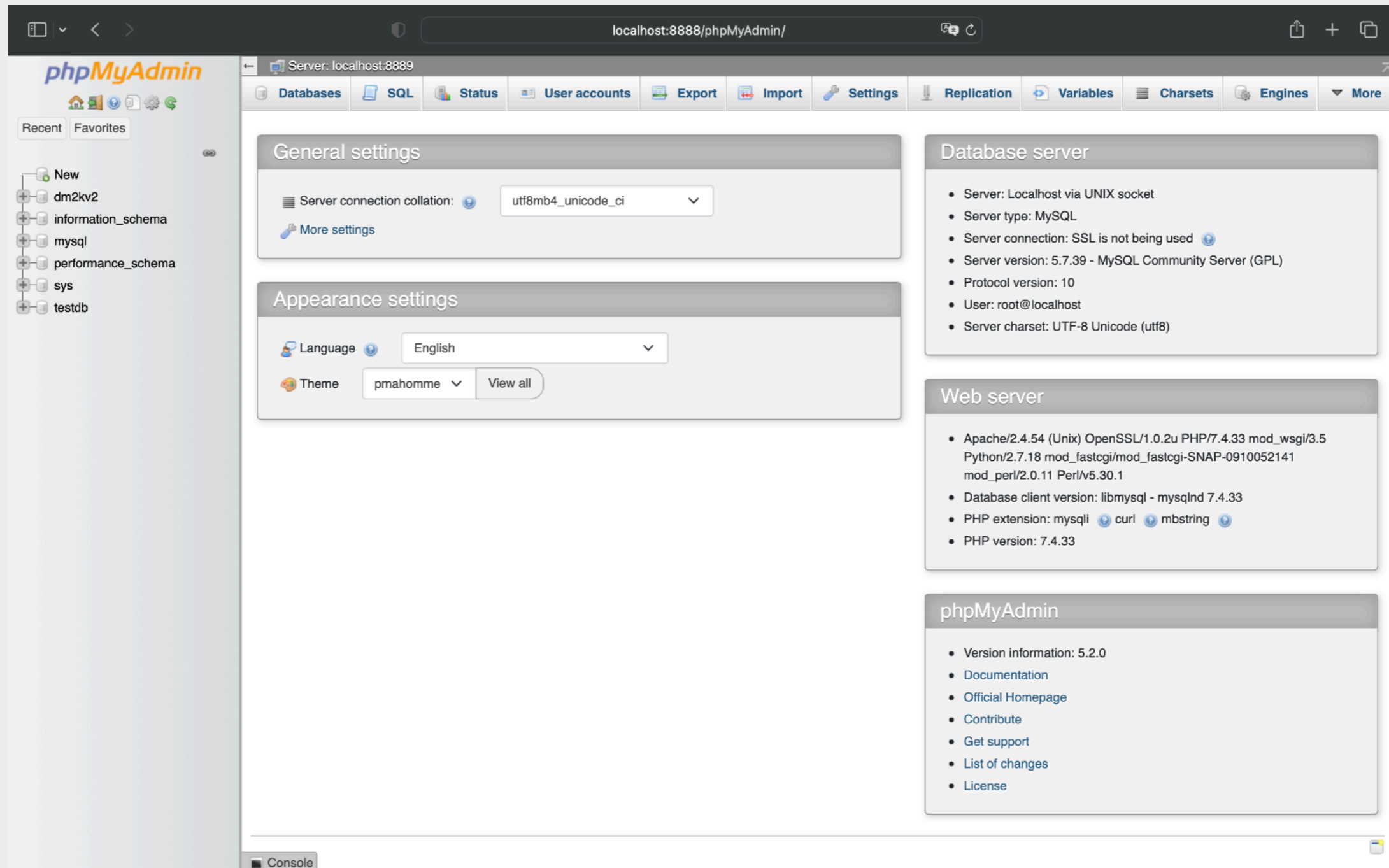
MAMP的網站檔案則通常儲存在上圖的文件夾，你可直接搜尋它出來



開啟灰色圖示的MAMP，
請不要開錯MAMP PRO

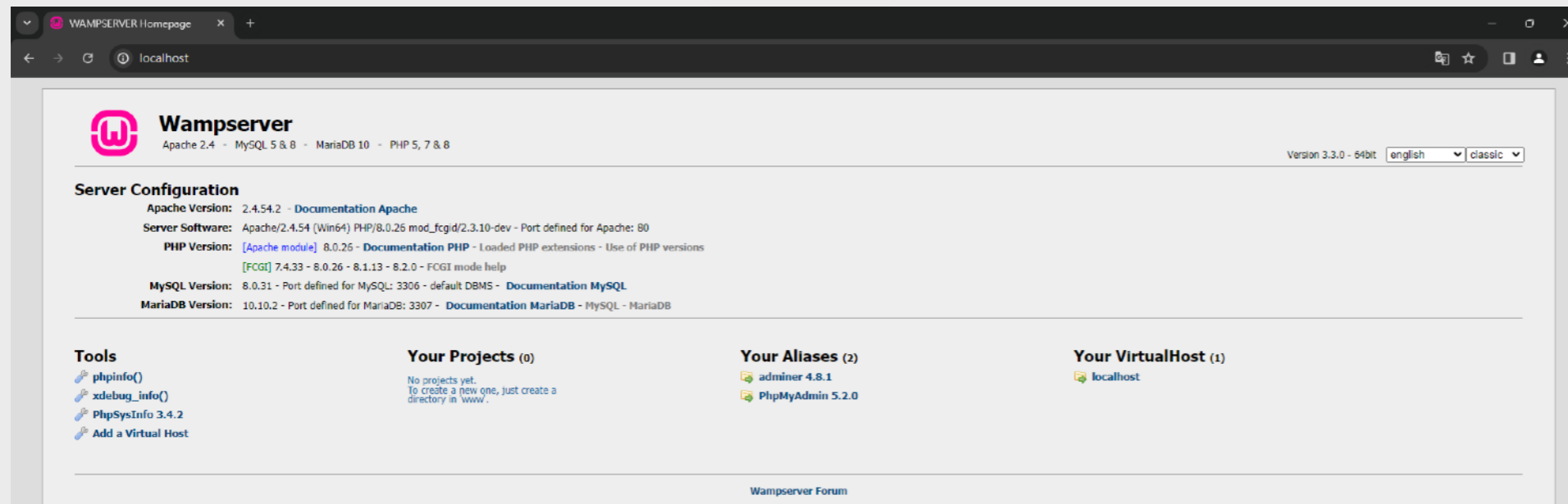
在Mac上安裝伺服器以運行Web App

然後輸入 <http://localhost:8888/phpmyadmin> 便可以連到PHPMyAdmin的版面。一般來說，如問及使用者名稱及密碼，則各輸入「root」為使用者名稱和密碼則可以登入。

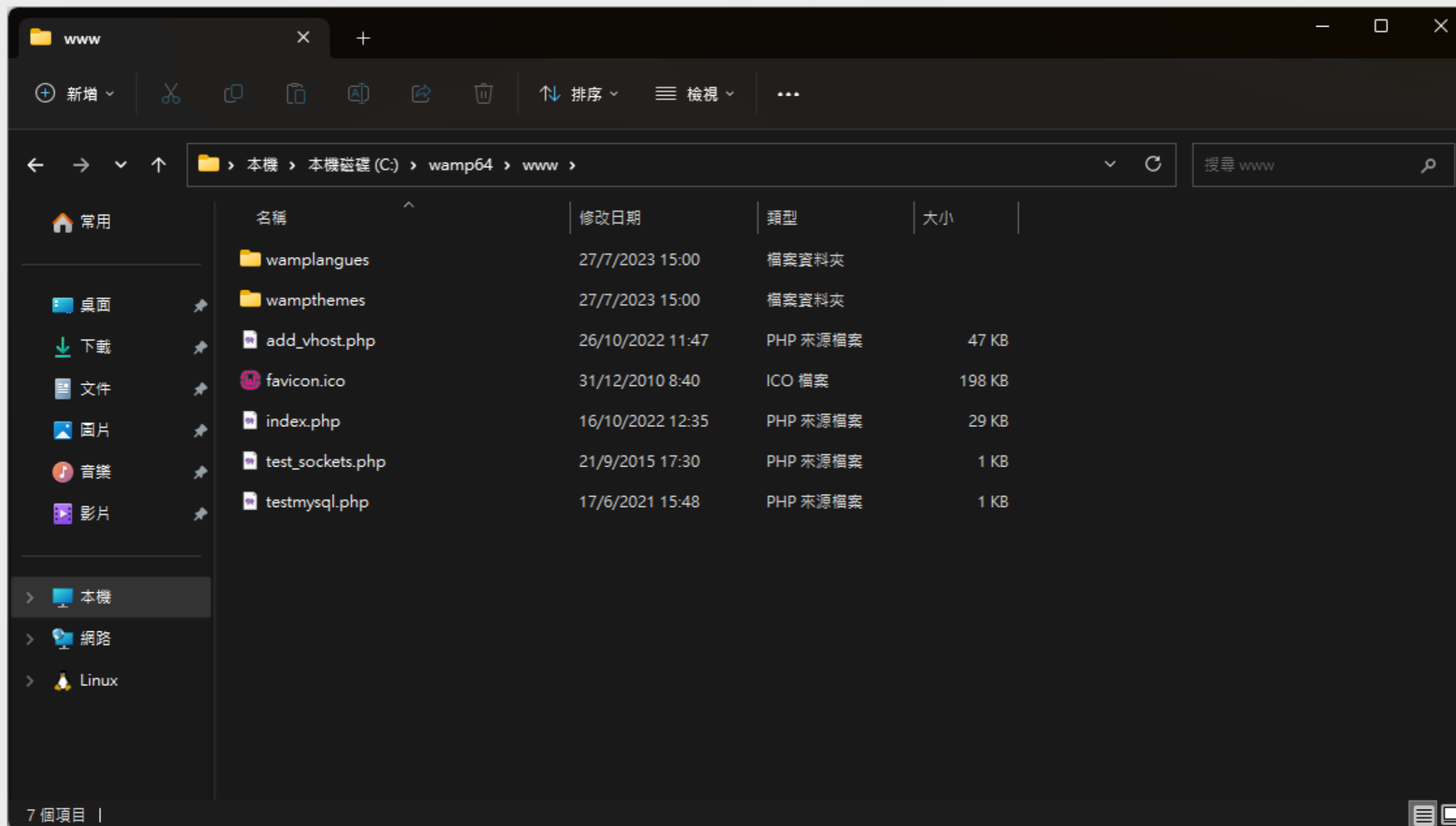


在Mac上安裝伺服器以運行Web App

於Windows電腦上，我們可以下載並安裝比較知名的WAMP安裝包，這是免費的，包含了Apache伺服軟件、MySQL數據庫和phpMyAdmin圖像使用者界面的數據庫管理網頁系統，相信已足夠初學者或個人開發者使用，官方網站為: <https://www.wampserver.com/> 安裝後開啟瀏覽器並跟據Port位連結到 <http://localhost/> 測試(如Port數值是80便不需特別於localhost後輸入:80)，如成功則會看到下圖畫面。



在Windows上安裝伺服器以運行Web App



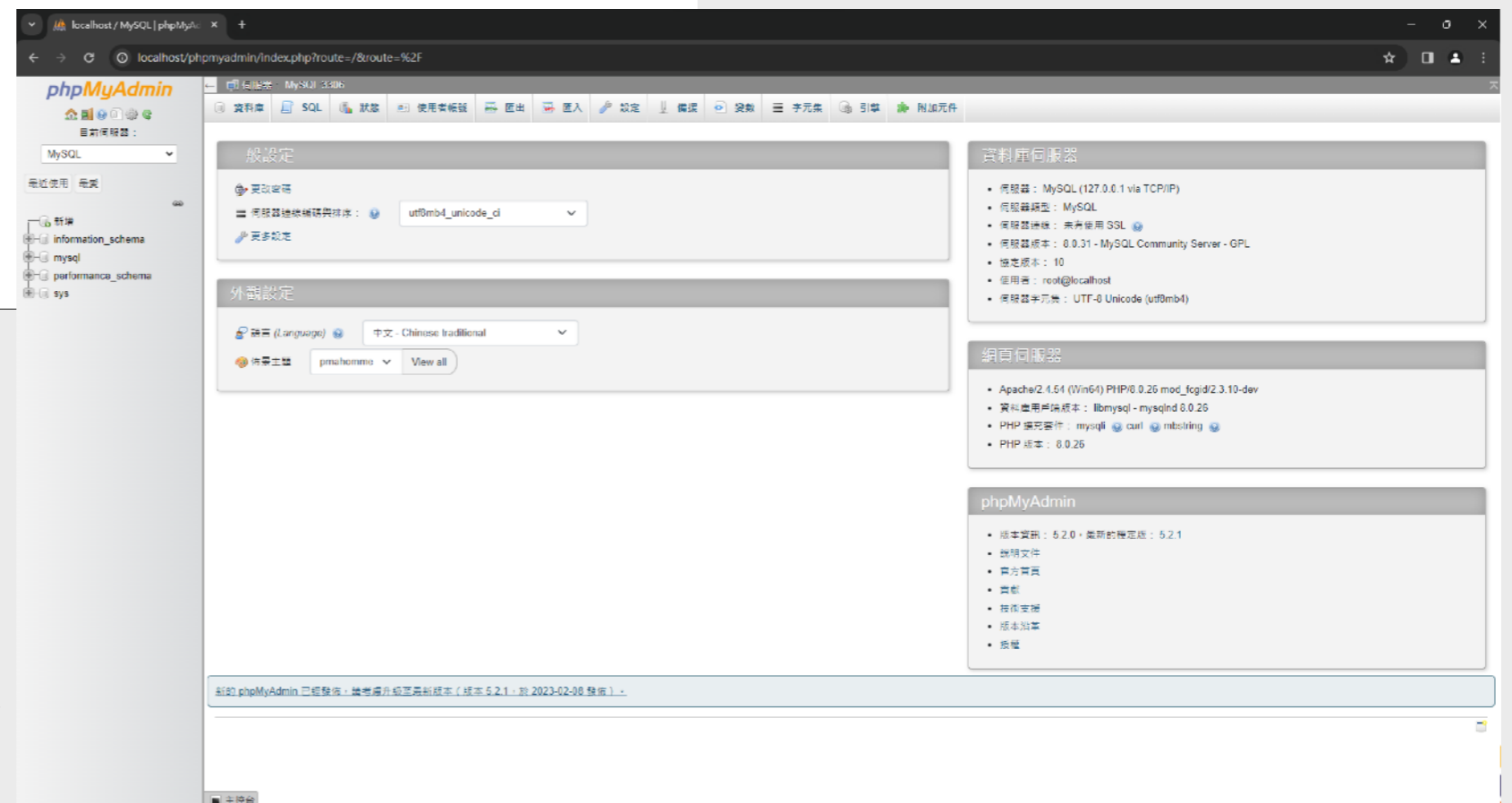
WAMP的網站檔案則通常儲存在上圖的文件夾

在Windows上安裝伺服器以運行Web App

然後輸入 <http://localhost/phpmyadmin> 便可以連到PHPMyAdmin的版面。一般來說，使用者名稱輸入「root」而密碼留空則可以登入。



登入前



登入後

在Windows上安裝伺服器以運行Web App

關聯式數據庫(Relational Database)概論(1)

一般來說，若果我們需要有系統地儲存資料，我們便會使用到數據庫，而數據庫分有很多種，當中比較主流的就是關聯式數據庫。

關聯式數據庫 (Relational Database) 是一種廣泛使用的數據庫，它基於關聯模型，由數學家 Edgar F. Codd 在1970年代初期提出。在關聯式數據庫中，所有的數據都存儲在稱為表 (tables) 的結構中，這些表是由行 (rows) 和列 (columns) 組成的集合。

以下是一些關聯式數據庫的關鍵特徵：

- 表(Table)：表是數據的集合，其中每一行代表一個數據記錄，每一列代表記錄中的一個字段（如名稱、地址等）。
- 行(Row)：表中的一行對應於單個記錄，也被稱為元組 (tuple) 或實體 (entity)。
- 列(Column)：表中的一列包含了具有相同數據類型的的信息，例如所有記錄的姓名或地址。
- 主鍵(Primary Key)：每個表中的一個或多個字段，用於唯一識別表中的每一行。沒有兩行具有相同的主鍵值。
- 外鍵(Foreign Key)：一個表中的字段，它是另一個表的主鍵的引用，用於建立兩個表之間的關聯。
- 關聯(Relation)：表之間的關聯，可以通過主鍵和外鍵的關聯來實現。
- 查詢(Query)：使用結構化查詢語言 (SQL) 等查詢語言，可以檢索、更新、管理和操作數據庫中的數據。

關聯式數據庫(Relational Database)概論(2)

- 完整性約束：數據庫的規則，用於維護數據的準確性和一致性，例如，確保電子郵件地址的格式正確或年齡字段中不包含負數。

關聯式數據庫管理系統（RDBMS）是用於創建、維護和管理關聯式數據庫的軟件。知名的RDBMS包括MySQL、Oracle、Microsoft SQL Server和PostgreSQL等。這些系統通常提供了數據庫設計、數據存儲、數據檢索、安全性管理和其他數據庫管理相關的功能。

例子：

university_db

students

student_id	student_name_tc	student_name_en	doctoral_advisor	research_area
1	袁郁美	Mei Yuen	5	Digital Marketing & KOL
2	甄恩恩	Yan Yan	2	Management for a Private University
3	陸續來	Ken Luk	6	Architecture Engineering

professors

professor_id	professor_name_tc	professor_name_en	professor_education
1	馮大文	Alex Fung	BComp, MEng, DEng
2	薩小欣	Lily Sa	BA (Hons), EdD, DBA
3	戴小明	Joe Tai	BSc, MPhil, ScD
4	杜安安	Eric To	BEng, MSc, MEd, EdD
5	穆大明	Ming Mu	BEcon, MPhil, MBA, PhD
6	鄒金主	Gold Chow	BArch (1st Hons), PhD
7	蔣奇	Kay Chiang	MCA, MSc, DCompSci

關聯式數據庫(Relational Database)概論(3)

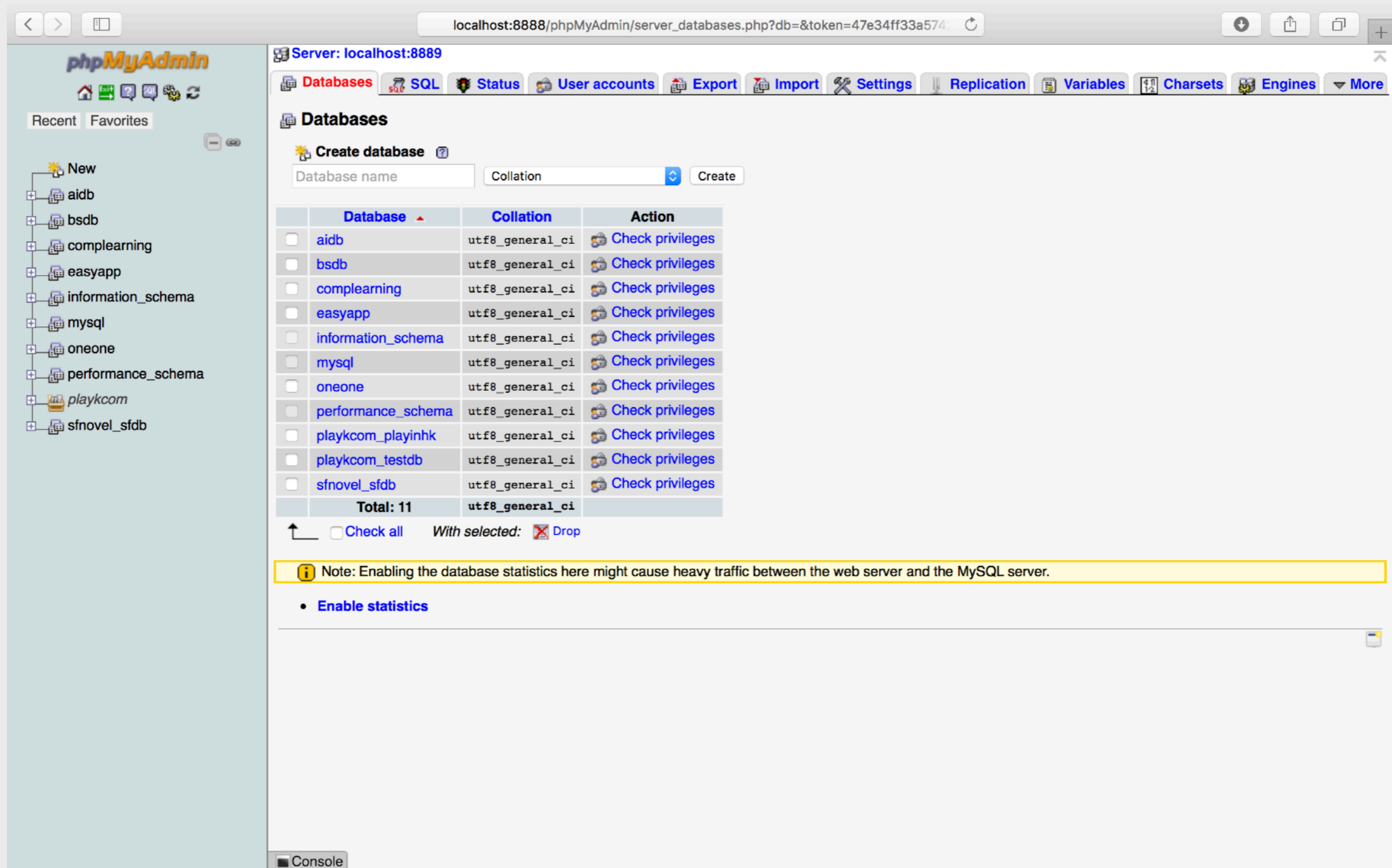
假如上頁例子是一間大學的數據庫，有兩個Table(列表)，其中一個名為students的Table儲存了學生的資料，另一名為professors的Table則儲存了教授的資料，我們可以看到，在students中student_id就是獨一無二和不能重覆使用的資訊，符合唯一性的條件，憑這個值便能找到指定的一個學生資料，是用來識別一位學生的，而我們則稱這個鍵為「Primary Key(主鍵)」；而professors中的Primary Key則是professor_id。而大家也可看到，students中的doctoral_advisor其實是代表某學生的博士論文教導師，可憑這鍵而在professors中找到了一位教授，而這個鍵我們則稱之為「Foreign Key(外鍵)」。當然，在數據庫的學問中，我們還有很多不同的Key(鍵)，如Super Key、Candidate Key和Alternate Key等。

對於關聯式數據庫，不同公司或不同團體都各自開發了大同小異的數據庫系統，例如主流免費的：
MySQL、PostgreSQL、SQLite (多應用於流動裝置)
公司擁有的：Microsoft SQL Server、Oracle Database

這些數據庫對於初學者而言其實都大同小異的，而要存取這些數據庫，我們便一般要編寫SQL(Structured Query Language)語言來進行存取。

然而，在本教材中，我們選用了免費且開源的數據庫系統MySQL作例子。

關聯式數據庫(Relational Database)概論(4)

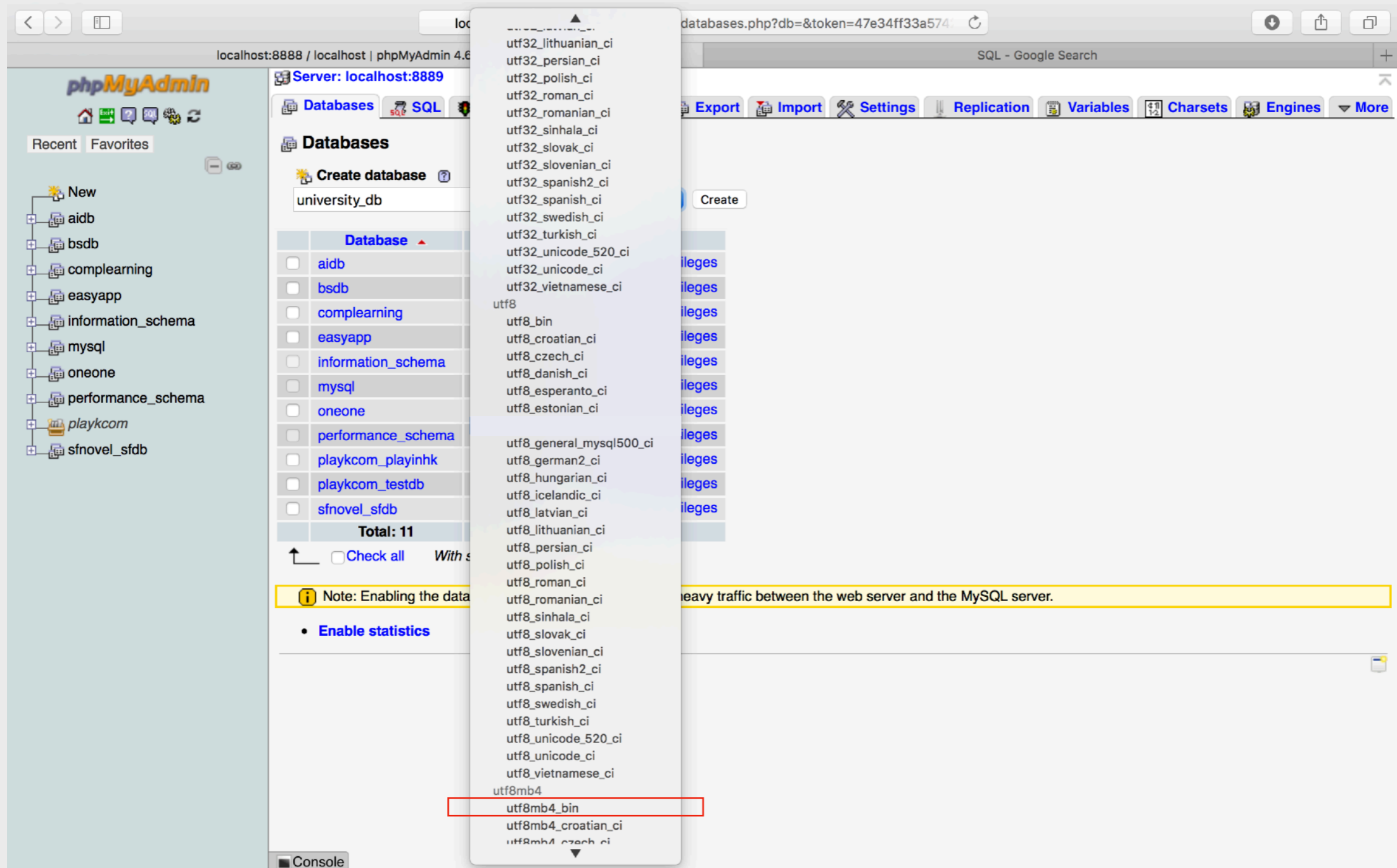


在學習SQL現在大家便先到phpMyAdmin自行試試建立上述的university_db數據庫來對數據庫管理系統有一個初步的認識。

Windows連結例子: <http://localhost/phpmyadmin>

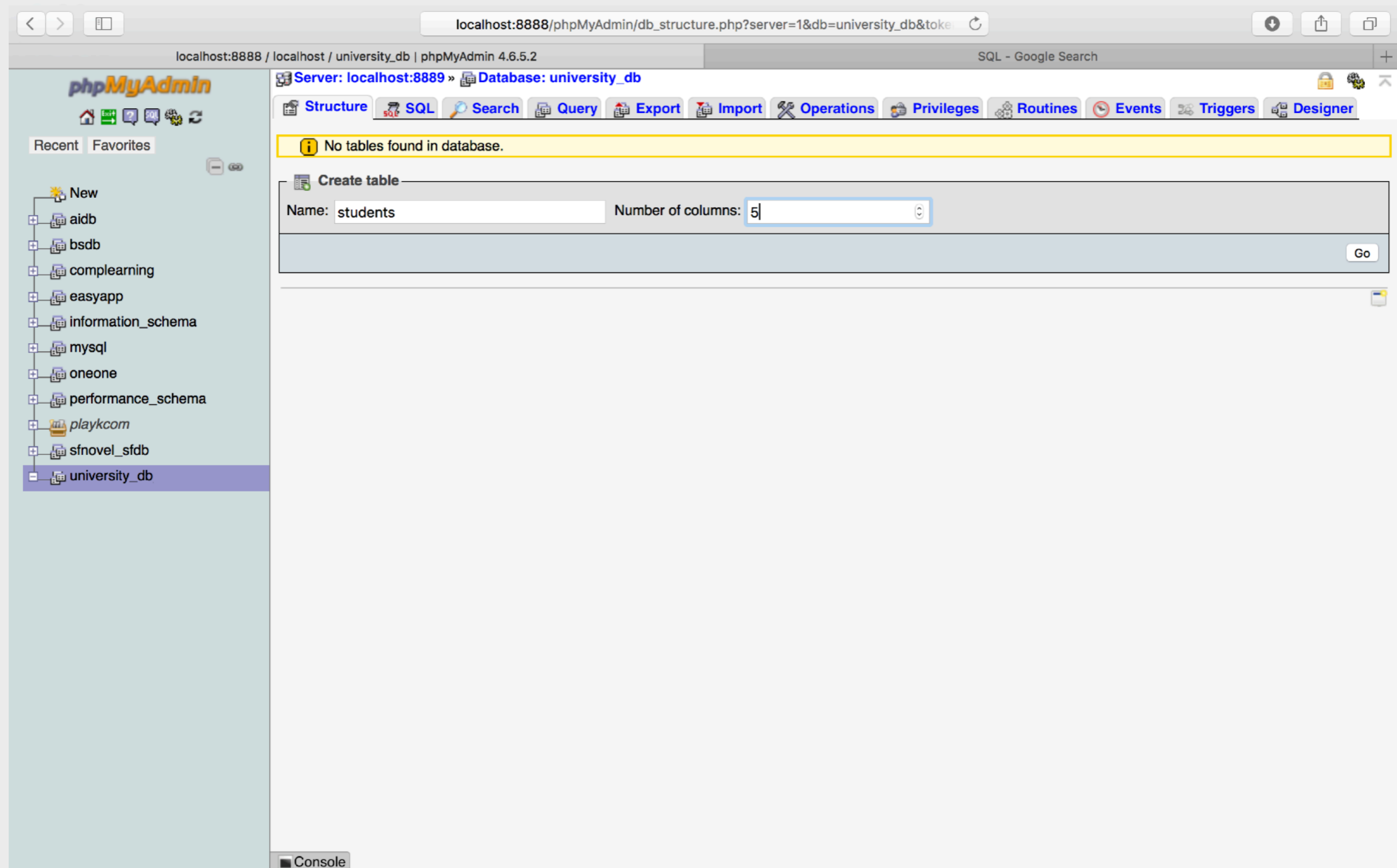
Mac連結例子: <http://localhost:8888/phpmyadmin>

關聯式數據庫(Relational Database)概論(4)



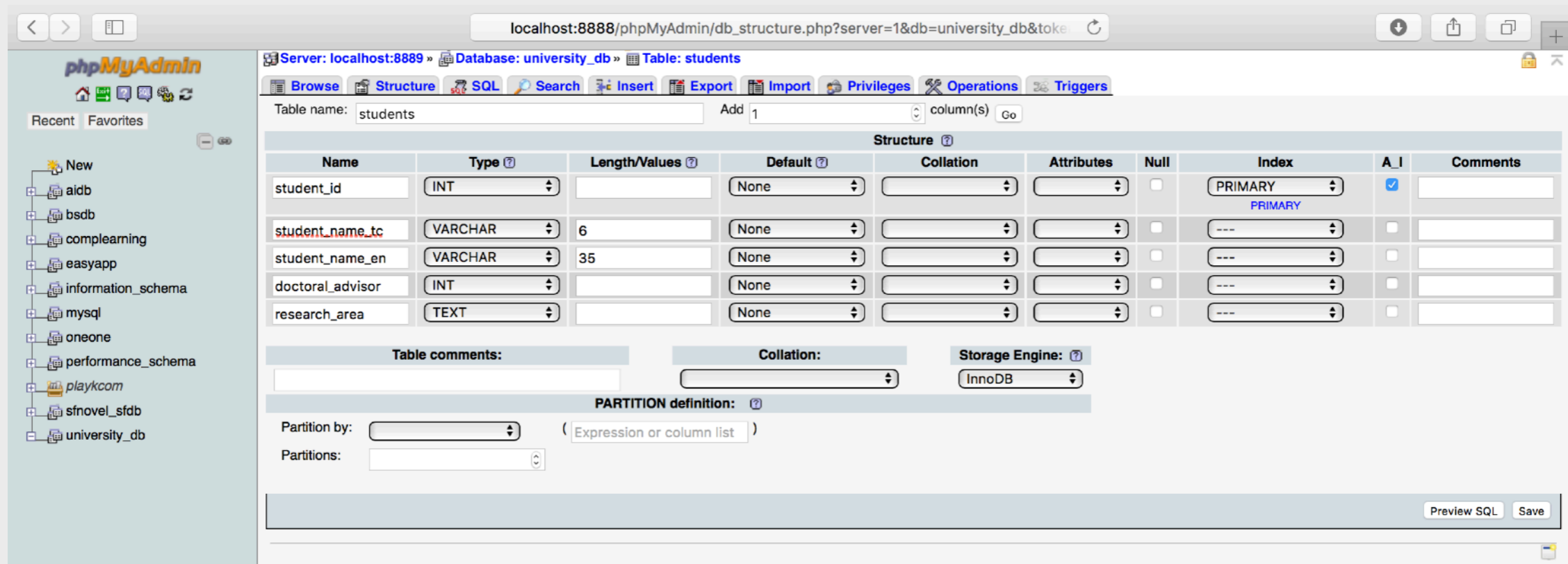
建立university_db數據庫時把Collation設定為utf8mb4_bin

關聯式數據庫(Relational Database)概論(5)



名稱輸入students，列數輸入5

關聯式數據庫(Relational Database)概論(6)



輸入Table的結構資料，因為student_id是Primary Key，所以需要在Index中選擇PRIMARY，也為了方便日後的資料輸入，所以別選A_I (Auto Increment)，即新數值自動加一(新增學生時，新學生的student_id會自動加一)。

常用的Type(種類)：

INT - 整數

DOUBLE - 點數

VARCHAR - 限制數量的文字，較TEXT節省空間

TEXT - 沒有限制數量的文字

DATE - 日期

DATETIME - 日期與時間

ENUM - 文字選擇 (如：兒童票/成人票/長者或傷殘人士票/員工家屬票)

關聯式數據庫(Relational Database)概論(7)

Server: localhost:8889 » Database: university_db » Table: students

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	student_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	student_name_tc	varchar(6)	utf8_general_ci		No	None			Change Drop More
3	student_name_en	varchar(35)	utf8_general_ci		No	None			Change Drop More
4	doctoral_advisor	int(11)			No	None			Change Drop More
5	research_area	text	utf8_general_ci		No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index

Print Propose table structure Move columns Improve table structure

Add 1 column(s) after research_area Go

+ Indexes

Partitions ?

No partitioning defined!

Partition table

Information

Table comments:

Space usage		Row statistics	
Data	16 KiB	Format	Compact
Index	0 B	Collation	utf8_general_ci
Total	16 KiB	Next autoindex	1
		Creation	Jul 21, 2017 at 12:11 PM

Console

按Save後便可看到Table已被建立。

關聯式數據庫(Relational Database)概論(8)

Server: localhost:8889 » Database: university_db » Table: professors

Table name: professors Add 1 column(s) Go

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I	Comments
professor_id	INT		None			<input type="checkbox"/>	PRIMARY PRIMARY	<input checked="" type="checkbox"/>	
professor_name_tc	VARCHAR	6	None			<input type="checkbox"/>	---	<input type="checkbox"/>	
professor_name_en	VARCHAR	35	None			<input type="checkbox"/>	---	<input type="checkbox"/>	
professor_education	TEXT		None			<input type="checkbox"/>	---	<input type="checkbox"/>	

Table comments: Collation: Storage Engine: InnoDB

PARTITION definition: Partition by: (Expression or column list) Partitions:

Preview SQL Save

試試使用同樣方法來建立professors Table

localhost:8888/phpMyAdmin/db_structure.php?token=47e34ff33a574231d5881e

Server: localhost:8889 » Database: university_db » Table: professors

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	professor_id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	professor_name_tc	varchar(6)	utf8_general_ci		No	None			Change Drop More
3	professor_name_en	varchar(35)	utf8_general_ci		No	None			Change Drop More
4	professor_education	text	utf8_general_ci		No	None			Change Drop More

Check all With selected: Browse Change Drop Primary Unique Index

Print Propose table structure Move columns Improve table structure

Add 1 column(s) after professor_education Go

+ Indexes

Partitions No partitioning defined! Partition table

Information

professors Table已被建立。

關聯式數據庫(Relational Database)概論(9)

Server: localhost:8889 » Database: university_db » Table: professors

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Column	Type	Function	Null	Value
professor_id	int(11)			
professor_name_tc	varchar(6)			馮大文
professor_name_en	varchar(35)			Alex Fung
professor_education	text			BComp, MEng, DEng

Ignore

Column	Type	Function	Null	Value
professor_id	int(11)			
professor_name_tc	varchar(6)			薩小欣
professor_name_en	varchar(35)			Lily Sa
professor_education	text			BA (Hons), EdD, DBA

Insert as new row and then Go back to previous page

Preview SQL Reset

Continue insertion with 2 rows

Console

可以嘗試使用phpMyAdmin新增或修改資料

SQL語言精讀 (1)

對於關聯式數據庫來說，我們一般都會編寫SQL(Structured Query Language)語言來進行存取。以下就是一些對初學者精選的語句。

Create Table (建立新列表)

```
CREATE TABLE 表格名 (鍵名稱 鍵種類 其他設定, 鍵名稱 鍵種類 其他設定);
```

例子：

```
CREATE TABLE professors  
(professor_id INT NOT NULL AUTO_INCREMENT,  
professor_name_tc VARCHAR(6) NOT NULL,  
professor_name_en VARCHAR(35) NOT NULL,  
professor_education TEXT NOT NULL,  
PRIMARY KEY (professor_id));
```

```
CREATE TABLE students  
(student_id INT NOT NULL AUTO_INCREMENT,  
student_name_tc VARCHAR(6) NOT NULL,  
student_name_en VARCHAR(35) NOT NULL,  
doctoral_advisor INT NOT NULL,  
research_area TEXT NOT NULL,  
PRIMARY KEY (student_id),  
FOREIGN KEY (doctoral_advisor) REFERENCES  
professors(professor_id));
```

SQL語言精讀 (2)

大家可試在phpMyAdmin隨意建立另一數據庫然後到SQL編輯頁面中輸入並執行SQL。

The screenshot shows the phpMyAdmin interface with the SQL editor open for the 'university2_db' database. The query editor contains the following SQL code:

```
1 CREATE TABLE professors
2 (professor_id INT NOT NULL AUTO_INCREMENT,
3 professor_name_tc VARCHAR(6) NOT NULL,
4 professor_name_en VARCHAR(35) NOT NULL,
5 professor_education TEXT NOT NULL,
6 PRIMARY KEY (professor_id));
7
8 CREATE TABLE students
9 (student_id INT NOT NULL AUTO_INCREMENT,
10 student_name_tc VARCHAR(6) NOT NULL,
11 student_name_en VARCHAR(35) NOT NULL,
12 doctoral_advisor INT NOT NULL,
13 research_area TEXT NOT NULL,
14 PRIMARY KEY (student_id),
15 FOREIGN KEY (doctoral_advisor) REFERENCES professors(professor_id));
```

Below the query editor, there are several options and a 'Go' button:

- Buttons: Clear, Format, Get auto-saved query
- Checkbox: Bind parameters ?
- Options: [Delimiter ;] Show this query here again Retain query box Rollback when finished Enable foreign key checks
- Button: Go (highlighted with a red circle)

SQL語言精讀 (3)

Insert Into (加入新資料)

```
INSERT INTO 表格名 (鍵名稱, 鍵名稱) VALUES (數值, 數值);
```

例子：

```
INSERT INTO professors
(professor_name_tc, professor_name_en, professor_education)
VALUES
('馮大文', 'Alex Fung', 'BComp, MEng, DEng'),
('薩小欣', 'Lily Sa', 'BA (Hons), EdD, DBA'),
('戴小明', 'Joe Tai', 'BSc, MPhil, ScD'),
('杜安安', 'Eric To', 'BEng, MSc, MEd, EdD'),
('穆大明', 'Ming Mu', 'BEcon, MPhil, MBA, PhD'),
('鄒金主', 'Gold Chow', 'BArch (1st Hons), PhD'),
('蔣奇', 'Kay Chiang', 'MCA, MSc, DCompSci');

INSERT INTO students
(student_name_tc, student_name_en, doctoral_advisor, research_area)
VALUES
('袁郁美', 'Mei Yuen', 5, 'Digital Marketing & KOL'),
('甄恩恩', 'Yan Yan', 2, 'Management for a Private University'),
('陸續來', 'Ken Luk', 6, 'Architecture Engineering');
```

SQL語言精讀 (4)

Select ... From (查看資料)

Select ... From (查看資料)

SELECT 鍵, 鍵, 鍵 FROM 表格名 WHERE 條件;

例子：

```
SELECT * FROM professors;
```

```
SELECT * FROM professors WHERE professor_id=2;
```

```
SELECT * FROM professors WHERE professor_name_en='Lily Sa';
```

```
SELECT * FROM professors WHERE professor_education LIKE '%PhD%';
```

```
SELECT professor_name_tc, professor_education FROM professors WHERE professor_id=5;
```

SQL語言精讀 (5)

Update (更新資料)

```
UPDATE 表格名 SET 鍵=數值 WHERE 條件;
```

例子：

```
UPDATE professors SET professor_name_en='Iris Sa' WHERE professor_id=2;
```

Delete From (刪除資料)

```
DELETE FROM 表格名 WHERE 條件;
```

例子：

```
DELETE FROM students WHERE student_id=3;
```

當然，數據庫原理和SQL都是一門高程度的學問，大家如想深入研究，建議進修相關的高等課程。

SQL語法教學申延閱讀

SQL語法教學 - 1Keydata: <http://www.1keydata.com/tw/sql/sql.html>

w3schools: <https://www.w3schools.com/sql/default.asp>

PHP基礎概念 (1)

程式碼區塊

以 `<?php` 作程式碼區塊的開端及以 `?>` 作程式碼區塊的結尾

```
<?php  
    // 程式碼寫在這裡  
?>
```

程式註解

```
<?php  
/*  
    多行註解  
    多行註解  
*/  
  
// 單行註解  
?>
```

PHP基礎概念 (2)

文字輸出

```
<?php
    echo 'abc';           // 輸出文字

    $str='abc';          // 定義變數
    echo $str;           // 輸出變數
    echo $str.'def123';  // 輸出文字及變數
?>
```

定義變數

PHP 中的變量以 \$ 符號開頭，不需要類型宣告。

```
$penguin='企鵝';
$num=123;
$bool=false;
$float=3.14;
$obj=new Object(); // 宣告物件
$arr=array();      // 陣列或鍵值配初始化(長寫版)
$arr=[];           // 陣列或鍵值配初始化(短寫版)
```

字串

字串可使用單引號或雙引號。

```
<?php
    $singleQuot = '單引號';
    $doubleQuot = "雙引號";
?>
```

PHP基礎概念 (3)

字串合併

在 PHP 中合併字串主要有兩種方式：

1. 使用「點」運算符 `.`
2. 使用大括號 `{}` 包含的字串變數

使用點號 `.` 合併字串

點號運算符用於將兩個或多個字串連接起來，創建一個新的字串。

```
<?php
$str1 = "Hello, ";
$str2 = "PHP!";
$combinedStr = $str1.$str2; // 合併兩個字串
echo $combinedStr; // 輸出 "Hello, PHP!"
?>
```

你也可以連接更多的字串或變數：

```
<?php
$str3 = " Have a nice day!";
$finalStr = $combinedStr.$str3;
echo $finalStr; // 輸出 "Hello, PHP! Have a nice day!"
?>
```

PHP基礎概念 (4)

使用大括號 {} 合併字串

亦可以在雙引號中使用大括號 {} 來明確指定要解析的變數。

```
<?php
$name = "John";
$greeting = "Hello, {$name}!"; // 使用大括號
echo $greeting; // 輸出 "Hello, John!"
?>
```

PHP基礎概念 (5)

PHP 與 HTML 結合

PHP 可以在 HTML 文件中以直接插入代碼的方式來增加網頁的動態功能。但請記得存檔時要把該 HTML 文件儲存成 PHP 檔案（即副檔名為 .php）。

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>PHP Webpage</title>
  </head>
  <body>
    <h1>Welcome to my Webpage</h1>
    <div>Some text.</div>
    <div><?php echo 'Some dynamic text from PHP.'; ?></div>
  </body>
</html>
```

PHP基礎概念 (6)

例外處理

```
try{  
    // 要嘗試執行的程式碼  
}catch (Exception $e){  
    // 得出錯誤時的處理機制  
}
```

迴圈

```
for($i=0; $i<count($arr); $i++){ /* ... */ } // For迴圈  
foreach($array as $key=>$value){ /* ... */ } // 鍵值配的Foreach迴圈  
foreach($array as $value){ /* ... */ } // 陣列的Foreach迴圈  
while($i<=10){ /* ... */ } // While迴圈  
do{ /* ... */ }while($i<=10) // Do While迴圈
```

判斷式

```
<?php  
if($number==10){  
    echo "The number is 10.";  
}elseif($number<10){  
    echo "The number is less than 10.";  
}else{  
    echo "The number is greater than 10.";  
}  
?>
```

PHP基礎概念 (7)

陣列

PHP 中有三種類型的陣列：索引陣列、關聯陣列、多維陣列。有兩種寫法，可使用 **array()** 或直接以 **[]** 來寫出。

```
<?php
```

```
// 索引陣列
```

```
$colors = array("Red", "Green", "Blue"); // 寫法一
```

```
$colors = ["Red", "Green", "Blue"]; // 寫法二
```

```
// 關聯陣列
```

```
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43"); // 寫法一
```

```
$age = ["Peter"=>"35", "Ben"=>"37", "Joe"=>"43"]; // 寫法二
```

```
// 多維陣列
```

```
$persons = array( // 寫法一
```

```
    "John" => array("Age"=>25, "Hair"=>"brown"),
```

```
    "Mary" => array("Age"=>27, "Hair"=>"black")
```

```
);
```

```
$persons = [ // 寫法二
```

```
    "John" => ["Age"=>25, "Hair"=>"brown"],
```

```
    "Mary" => ["Age"=>27, "Hair"=>"black"]
```

```
];
```

```
?>
```

PHP基礎概念 (8)

常用陣列處理函數

<code>count(\$arr)</code>	// 取得\$arr陣列所包含的變數數量
<code>asort(\$arr)</code>	// 將\$arr根據value(值)而排序
<code>ksort(\$arr)</code>	// 將\$arr根據key(鍵)而排序
<code>in_array('Ada', \$students)</code>	// 判斷\$students陣列中，「Ada」是否存在
<code>array_key_exists('name', \$student)</code>	// 判斷\$student鍵值配中，「name」鍵是否存在

常用字串處理

<code>substr('abcdefg', 1, 3)</code>	// 字串擷取
<code>str_replace('world', 'Ada', 'Hello world')</code>	// 字串取代
<code>strlen('abcdefg')</code>	// 字串長度
<code>mb_strlen('一二三')</code>	// 字串長度(中文)
<code>trim(\$str)</code>	// 清除字串前後空白
<code>strtolower(\$str)</code>	// 把字串轉換為小寫
<code>strtoupper(\$str)</code>	// 把字串轉換為大寫
<code>ucfirst(\$str)</code>	// 把首字字母轉成大寫
<code>json_encode(\$str)</code>	// 把文字以JSON方式顯示

PHP基礎概念 (9)

呼叫及定義函式

```
<?php  
  
// 定義函式  
function echoStr($str){  
    echo $str;  
}  
  
// 呼叫函式  
echoStr('abc');  
?>
```

類別(Class)應用

```
require_once('car.php'); // 先載入類別檔案  
$c=new Car();           // 宣告類別  
$c->startRun();         // 呼叫類別函式
```

PHP基礎概念 (10)

PHP GET、POST 與 SESSION 使用

在網頁開發中，GET 和 POST 是兩種常見的 HTTP 請求方法，用於從用戶端向伺服器傳送資料。SESSION 則是一種用於保持用戶狀態的機制。以下是對這三者使用方法的介紹。

第一節：GET 請求

GET 請求通常用於請求服務器發送特定資源，並且可以將查詢字符串（名稱/值對）附加到 URL（網址）中。

GET 請求的特點：

資料可在 URL 中看到，並適合請求非敏感資料，而且有長度限制（視乎不同的瀏覽器），但能被書籤(Bookmark)。例如：<https://www.google.com/search?client=opera&q=php>

在 PHP 中接收 GET 數據：

```
<?php
// welcome_get.php
$name = $_GET['name']; // 使用 $_GET['參數名稱']
echo "歡迎 " . $name;
?>
```

PHP基礎概念 (11)

第二節：POST 請求

POST 請求通常用於向伺服器提交要被處理的數據，例如提交表單或發出 Ajax。

POST 請求的特點：

數據不會顯示在 URL 中，所以適合傳輸敏感資料，而且沒有大小限制，但不能被書籤。

在 PHP 中接收 POST 數據：

```
<?php
// welcome_post.php
$name = $_POST['name']; // 使用 $_POST['參數名稱']
echo "歡迎 " . $name;
?>
```

PHP基礎概念 (12)

第三節：SESSION

SESSION 為伺服器端的暫存記憶公域變數。亦有著「會話」的意思，在 PHP 程式設計中 SESSION 代表伺服器與客戶端之間的「會話」，意思是伺服器與客戶端不斷的交流。

3.1 啟動 SESSION：

在 PHP 腳本開頭使用 `session_start()` 函數。

```
<?php  
// 啟動新的或繼續已有的會話  
session_start();  
?>
```

3.2 註冊 SESSION 變量：

將數據存儲到 SESSION 變量中。

```
<?php  
// 存儲 session 數據  
$_SESSION["favcolor"] = "green";  
$_SESSION["favanimal"] = "cat";  
?>
```

PHP基礎概念 (13)

3.3 使用 SESSION 變量：

在應用的其他頁面上使用 SESSION 變量。

```
<?php
// 開始會話
session_start();

// 訪問 session 數據
echo "我最喜歡的顏色是 " . $_SESSION["favcolor"] . " 。 <br>";
echo "我最喜歡的動物是 " . $_SESSION["favanimal"] . " 。”;
?>
```

3.4 銷毀 SESSION：

完成使用後，可刪除某些 SESSION 變量或完全銷毀會話。

```
<?php
// 刪除所有的 SESSION 變量
session_unset();

// 銷毀一個會話
session_destroy();
?>
```

通過這種方式，可以有效地管理用戶在網站中的登錄狀態和其他重要信息。

PHP基礎概念 (14)

PHP 的設置選項：

在 PHP 中，**php.ini** 文件是一個配置文件，它為 PHP 運行時的環境提供了設置選項。這個文件包含了許多指令，用於控制 PHP 的功能，包括資源限制（如記憶體使用量）、文件上傳的處理、錯誤報告等級和日誌記錄等。

當 PHP 啟動時（無論是作為伺服器模組還是命令行介面），它會讀取這個文件並應用設定的配置。這些設置可以在運行時使用 PHP 函數 `ini_set()` 進行修改，但有些設置僅能在 `php.ini` 文件中設定，無法在腳本中動態更改。

`php.ini` 文件通常位於以下位置之一：

- 對於 Windows 服務器，它可能位於 PHP 安裝目錄中，如 `C:\php\php.ini`。
- 對於 Linux/Unix 服務器，它通常位於 `/etc/php/` 或 `/etc/php5/`、`/etc/php7/` 等目錄中，具體取決於 PHP 版本。
- 對於共享主機，`php.ini` 文件可能無法直接訪問，這種情況下，你可能需要通過主機提供的控制面板或其他工具來修改配置。

PHP基礎概念 (15)

php.ini 中的一些常見設置包括：

- memory_limit - 腳本可以消耗的最大記憶體量。
- upload_max_filesize 和 post_max_size - 分別控制 PHP 可以處理的最大上傳文件大小和 POST 數據大小。
- display_errors - 指定是否在瀏覽器中顯示錯誤。
- error_reporting - 定義 PHP 報告哪些類型的錯誤。
- date.timezone - 設定用於所有日期時間功能的默認時區。
- extension - 啟用或禁用 PHP 擴展。

要查看當前配置和位置，你可以創建一個包含以下代碼的 PHP 文件，只有一句 `phpinfo();` 即可，然後通過瀏覽器訪問它：

```
<?php phpinfo(); ?>
```

這將輸出包括 php.ini 配置文件路徑在內的所有當前 PHP 配置資訊。更改 php.ini 文件後，你通常需要重新啟動 Web 服務器（如 Apache、Nginx 或 PHP-FPM）以使更改生效。

PHP基礎概念 (16)

總結 PHP 的 GET、POST 與 SESSION 使用：

GET 和 POST 是基於 HTTP 協議用於客戶端和服務器之間通信的兩種主要方法。SESSION 是一種伺服器端的存儲機制，用於跨多個頁面請求保持用戶數據。

當使用 GET 方法時，資料會附加在 URL 中，適合非敏感信息的傳輸。而 POST 方法不會在 URL 中顯示數據，適合傳輸敏感信息。SESSION 則被用來保存用戶的狀態信息，這在實現用戶登錄功能時非常重要。

希望這篇章能夠幫助你了解 PHP 中 GET、POST 和 SESSION 的基本用法。在實際開發中，你將會頻繁使用到這些知識。

PHP 存取 MySQL 數據庫 (1)

PHP 可使用 PDO (PHP Data Objects) 來存取 MySQL 數據庫。PDO 是一個數據庫訪問層，提供了一個統一的方法來訪問多種數據庫。使用 PDO 操作 MySQL 可以提高數據庫操作的安全性，並且可以輕鬆地切換到其他支持 PDO 的數據庫系統。

透過添加以下函式到 PHP 檔，使用 SQL 的指令式語句 (如 INSERT INTO、UPDATE、DELETE FROM 等)，便可對 MySQL 數據庫執行指令 (請自行更改使用帳號和密碼，即「`username`」與「`password`」)：

```
function executeCommandToDB($_sql, $_variables, $_getLastInsertId=false) {
    $db=new PDO('mysql:host=localhost;dbname=db1;charset=utf8mb4', 'username', 'password');
    $db->exec("SET SESSION time_zone='+8:00'");
    $sth=$db->prepare($_sql);
    $count=0;
    if($_variables!=null)$count=$sth->execute($_variables);
    else $count=$sth->execute();
    if($_getLastInsertId)$last_id=$db->lastInsertId();
    $db=null;
    if(!$_getLastInsertId && $count>0)return true;
    elseif($_getLastInsertId && $count>0)return $last_id;
    return false;
}
```

PHP 操作 MySQL 數據庫 (2)

函式參數

- `$_sql`: SQL 語句字符串，這是你想要執行的 SQL 命令。
- `$_variables`: 一個關聯數組，包含 SQL 語句中要綁定的參數。
- `$_getLastInsertId`: 一個布爾值，用於指示是否需要返回最新插入行的 ID。

函式操作流程

1. 創建 PDO 實例:

使用 `new PDO()` 創建一個新的數據庫連接對象。連接至名為 `db1` 的數據庫，並設置字符集為 `utf8mb4` 以支持 Unicode 字符。

2. 設置時區:

通過 `exec()` 方法對數據庫會話進行時區設定，這裡將時區設為 `+8:00`，適用於香港。

3. 準備 SQL 語句:

調用 `$db->prepare($_sql)` 預處理 SQL 語句。這有助於防止 SQL 注入攻擊，因為它允許參數化查詢。

4. 執行 SQL 語句:

如果提供了 `$_variables` 參數，則使用 `execute($_variables)` 帶參數執行 SQL 語句。如果沒有提供，則直接調用 `execute()`。

5. 獲取插入 ID:

如果 `$_getLastInsertId` 為 `true`，使用 `$db->lastInsertId()` 獲取最後插入的行 ID。

PHP 操作 MySQL 數據庫 (3)

6. 關閉數據庫連接:

將 `$db` 設為 `null` 來關閉數據庫連接。

7. 返回結果:

- 如果 `$_getLastInsertId` 為 `false` 而且 `$count` 大於 0，表示 SQL 命令成功執行，則返回 `true`。
- 如果 `$_getLastInsertId` 為 `true` 且 `$count` 大於 0，表示成功執行且需要返回插入 ID，則返回 `$last_id`。
- 如果以上條件都不符合，表示 SQL 命令執行失敗，則返回 `false`。

範例一：

```
$cus_id=executeCommandToDB("INSERT INTO customers (cus_name, cus_age) VALUES (:cname, :cage)", ['cus_name'=>$_POST['cname'],'cus_age'=>$_POST['cage']], true);
```

範例二：

```
executeCommandToDB("UPDATE books SET bprice=0", null);
```

PHP 存取 MySQL 數據庫 (4)

透過添加以下函式到 PHP 檔，使用 SQL 的 SELECT 語句，便可對 MySQL 數據庫執行指令來取得數據庫中的資料（請自行更改使用帳號和密碼，即「username」與「password」）：

```
function retrieveFromDB($_sql, $_variables) {  
    $db=new PDO('mysql:host=localhost;dbname=db1;charset=utf8mb4', 'username', 'password');  
    $db->exec("SET SESSION time_zone='+8:00'");  
    $sth=$db->prepare($_sql);  
    if($_variables!=null)$sth->execute($_variables);  
    else $sth->execute();  
    $result=$sth->fetchAll(PDO::FETCH_ASSOC);  
    $db=null;  
    return $result;  
}
```

函式參數

- \$_sql: 要執行的 SQL 語句。
- \$_variables: 執行預處理語句時綁定的變量數組，預設為 null。

PHP 存取 MySQL 數據庫 (5)

函式操作流程

1. 建立 PDO 連接

創建新的 PDO 連接以與 MySQL 數據庫進行通訊，連接信息包含在全局變量 `$paras` 中。

2. 設定時區

通過 `exec()` 方法對數據庫會話進行時區設定，這裡將時區設為 `+8:00`，適用於香港。

3. 準備 SQL 語句

利用 PDO 的 `prepare` 方法，預處理 SQL 語句，這有助於防範 SQL 注入攻擊。

4. 執行 SQL 語句

根據是否有變量數組提供，使用 `execute` 方法執行 SQL 語句。如果有變量需要綁定，則將其作為參數傳遞。

5. 獲取查詢結果

使用 `fetchAll` 方法以關聯數組格式獲取所有查詢結果。

6. 結束連接

清除數據庫連接對象來結束連接。

7. 返回結果

以陣列型式傳回多行結果。

PHP 存取 MySQL 數據庫 (6)

範例一：

```
$cus=retrieveFromDB("SELECT * FROM customers WHERE cusid=?", [$_POST['cusid']]);  
$cus=count($cus)>0 ? $cus[0]:null;
```

範例二：

```
$books=retrieveFromDB("SELECT * FROM books", null);
```

範例三：

```
$books=retrieveFromDB("SELECT * FROM books WHERE bookid=:bookid",  
['bookid'=>$_POST['bookid']]);
```

PHP與客戶端互動範例 (1)

ser.php

```
<?php
session_start();
header('Content-Type: application/json; charset=utf-8');
header('Access-Control-Allow-Origin: *');
if(function_exists('date_default_timezone_set'))date_default_timezone_set('Asia/Hong_Kong');
if(empty($_POST['service'])){
    http_response_code(400);
    echo "Missing the 'service'.";
    return;
}
$service=$_POST['service'];
include('dbapi.php'); // 載入數據庫存取函式的PHP檔案

if($service=='test'){
    echoSuccessful($ser,'core everything good');
}
elseif($service=='testdb'){
    $res=retrieveFromDB("SELECT * FROM shopping_malls LIMIT 0,10", null);
    echo json_encode($res, JSON_UNESCAPED_UNICODE);
    // 上句把PHP陣列或鍵值配以JSON方式顯示出來
}
?>
```

PHP與客戶端互動範例 (2)

這個章節將一步步解析上頁的 PHP Web Server 程式碼，這段程式碼設計用於處理 HTTP 請求，並根據請求參數返回相應的 JSON 格式數據。

程式碼結構，該程式碼可分為以下幾個主要部分：

1. 初始設定
2. 請求處理
3. 數據庫連接
4. 服務響應

接下來我們將逐一深入探討每一部份。

1. 初始設定

```
session_start();  
header('Content-Type: application/json; charset=utf-8');  
header('Access-Control-Allow-Origin: *');  
if(function_exists('date_default_timezone_set')) date_default_timezone_set('Asia/Hong_Kong');
```

- `session_start()`：啟動一個新的會話或者恢復當前會話。
- `header('Content-Type: application/json; charset=utf-8')`：設定內容類型為 JSON，並指定編碼為 UTF-8，確保客戶端知道返回的內容是 JSON 格式。
- `header('Access-Control-Allow-Origin: *')`：允許任何源對該服務發起跨域請求。
- `date_default_timezone_set('Asia/Hong_Kong')`：如果 PHP 配置允許，將預設時區設定為香港時間。

PHP與客戶端互動範例 (3)

2. 請求處理

```
if(empty($_POST['service'])){  
    http_response_code(400);  
    echo "Missing the 'service'.";  
    return;  
}  
$service = $_POST['service'];
```

- 檢查 POST 請求中是否包含 `service` 欄位。
- 如果沒有，將 HTTP 響應狀態碼設為 `400`（壞的請求），並返回錯誤信息 "Missing the 'service'."。
- 如果 `service` 存在，將其值賦給變量 `\$service`。

3. 數據庫連接

```
include('dbapi.php');
```

- 包含 (include) 一個名為 `dbconnect.php` 的文件，這個文件應該包含數據庫連接和操作的函式或代碼。

PHP與客戶端互動範例 (4)

4. 服務響應

```
if($service == 'test'){
    echoSuccessful($ser,'core everything good');
}
elseif($service == 'testdb'){
    $res = retrieveFromDB("SELECT * FROM shopping_malls LIMIT 0,10", null);
    echo json_encode($res, JSON_UNESCAPED_UNICODE);
}
```

- 根據`\$service`的值，判斷應該執行哪一種服務。
- 如果`\$service`是`'test'`，則呼叫一個名為`echoSuccessful`的函式（此函式在代碼中未定義，可能在其他文件中）。
- 如果`\$service`是`'testdb'`，則使用`retrieveFromDB`函式（可能在`dbconnect.php`文件中定義）從數據庫中檢索購物商場的列表，且限制結果為前 10 條記錄。
- 最後，使用`json_encode`將查詢結果轉換為 JSON 格式並返回，`JSON_UNESCAPED_UNICODE`選項確保多字節的 Unicode 字符不會被轉義。

PHP與客戶端互動範例 (5)

dbapi.php - 這是 ser.php 透過 `include('dbapi.php');` 來所使用的 PHP 檔案

```
<?php

function executeCommandToDB($_sql, $_variables, $_getLastInsertId=false) {
    $db=new PDO('mysql:host=localhost;dbname=db1;charset=utf8mb4', 'username', 'password');
    $db->exec("SET SESSION time_zone='+8:00'");
    $sth=$db->prepare($_sql);
    $count=0;
    if($_variables!=null)$count=$sth->execute($_variables);
    else $count=$sth->execute();
    if($_getLastInsertId)$last_id=$db->lastInsertId();
    $db=null;
    if(!$_getLastInsertId && $count>0)return true;
    elseif($_getLastInsertId && $count>0)return $last_id;
    return false;
}

function retrieveFromDB($_sql, $_variables) {
    $db=new PDO('mysql:host=localhost;dbname=db1;charset=utf8mb4', 'username', 'password');
    $db->exec("SET SESSION time_zone='+8:00'");
    $sth=$db->prepare($_sql);
    if($_variables!=null)$sth->execute($_variables);
    else $sth->execute();
    $result=$sth->fetchAll(PDO::FETCH_ASSOC);
    $db=null;
    return $result;
}

function retrieveFromDBOneRow($_sql, $_variables) {
    $result=retrieveFromDB($_sql, $_variables);
    if(count($result)==0)return null;
    return $result[0];
}

?>
```

項目實習-附近商場APP-伺服器端製作流程

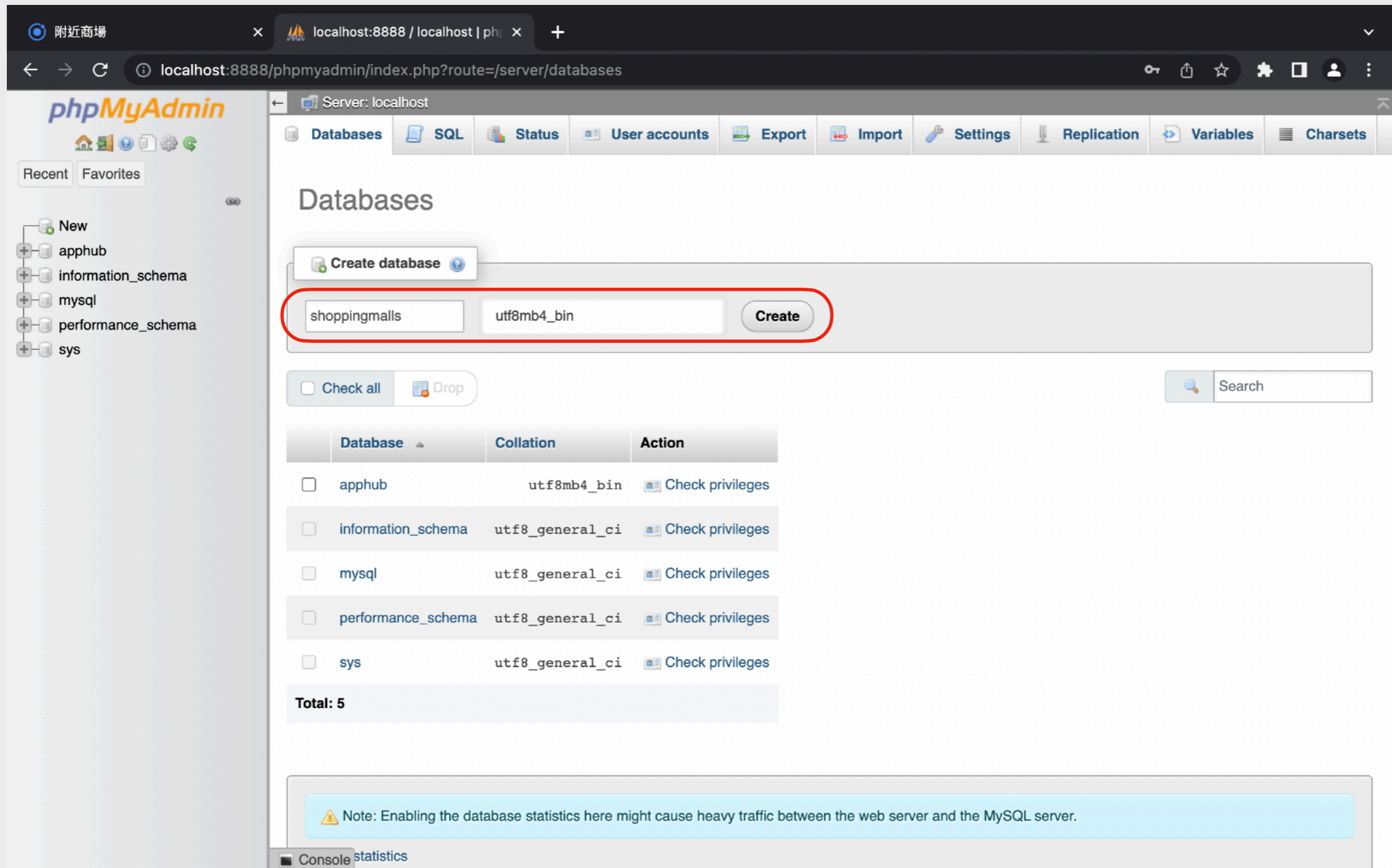
1. 啟動伺服器程式 (即Windows電腦的AppServ或Mac電腦的MAMP)
2. 創建數據庫(Database)並匯入教材中已事先預備好的SQL檔案
3. 把教材中已事先預備好的PHP檔案放到伺服器程式的資料夾裡
4. 設定 db_api.php 中的數據庫密碼
5. 到客戶端(即 Ionic App)中加入本筆記中已事先預備好的程式碼

項目實習-附近商場APP-啟動伺服器程式



如果使用Mac電腦，首先開啟MAMP，
並確保已啟動(Start)了MAMP伺服器
若是Windows電腦的話，
則確保已啟動了AppServ或WAMP
伺服器。

項目實習-附近商場APP-創建數據庫(1)



The screenshot shows the phpMyAdmin interface for a MySQL server. The 'Databases' tab is active, and the 'Create database' form is visible. The form contains two input fields: the first is labeled 'shoppingmalls' and the second is labeled 'utf8mb4_bin'. A red circle highlights the 'Create' button next to the second field. Below the form is a table listing existing databases on the server.

Database	Collation	Action
<input type="checkbox"/> apphub	utf8mb4_bin	<input type="checkbox"/> Check privileges
<input type="checkbox"/> information_schema	utf8_general_ci	<input type="checkbox"/> Check privileges
<input type="checkbox"/> mysql	utf8_general_ci	<input type="checkbox"/> Check privileges
<input type="checkbox"/> performance_schema	utf8_general_ci	<input type="checkbox"/> Check privileges
<input type="checkbox"/> sys	utf8_general_ci	<input type="checkbox"/> Check privileges

Total: 5

Note: Enabling the database statistics here might cause heavy traffic between the web server and the MySQL server.

首先到 PHPMYAdmin 並在 Database 分頁中 Create Database 框中輸入 **shoppingmalls**，並選擇 **utf8mb4_bin** 作編碼，然後按 Create。

項目實習-附近商場APP-創建數據庫(2)

The screenshot shows the phpMyAdmin interface for the 'shoppingmalls' database. The 'Import' tab is active, and the 'File to import' section is highlighted with a red circle. The file 'shoppingmalls.sql' is selected, and the 'Go' button at the bottom right is also circled in red. A file explorer window is overlaid on the interface, showing the directory structure of the project.

Folder/File	Modified	Type	Size	Type
ionic-labs-2	Today at 2:06 PM	Folder	--	Folder
ShoppingMallsApp	Today at 2:05 PM	Folder	--	Folder
shoppingmallserver	11 Jul 2022 at 1:22 PM	Folder	--	Folder
db_api.php	6 Jul 2022 at 12:50 PM	PHP Script	1 KB	PHP Script
ser.php	11 Jul 2022 at 1:19 PM	PHP Script	2 KB	PHP Script
serapi.php	18 Jun 2021 at 3:27 PM	PHP Script	10 KB	PHP Script
shoppingmalls.sql	11 Jul 2022 at 1:23 PM	TextDocument	113 KB	TextDocument

進入 shoppingmalls 並選擇 Import 分頁，然後按「選擇檔案」來匯入教材中已預備好的 shoppingmalls.sql (於 shoppingmallserver 資料夾內)，最後按右下方的 Go。

項目實習-附近商場APP-創建數據庫(3)

The screenshot shows the phpMyAdmin interface for a MySQL database named 'shoppingmalls'. The left sidebar shows the database structure with 'shopping_malls' selected. The main panel displays the following messages and SQL queries:

- Import has been successfully finished, 5 queries executed. (shoppingmalls.sql)**
- MySQL returned an empty result set (i.e. zero rows). (Query took 0.0313 seconds.)**
The SQL query shown is:

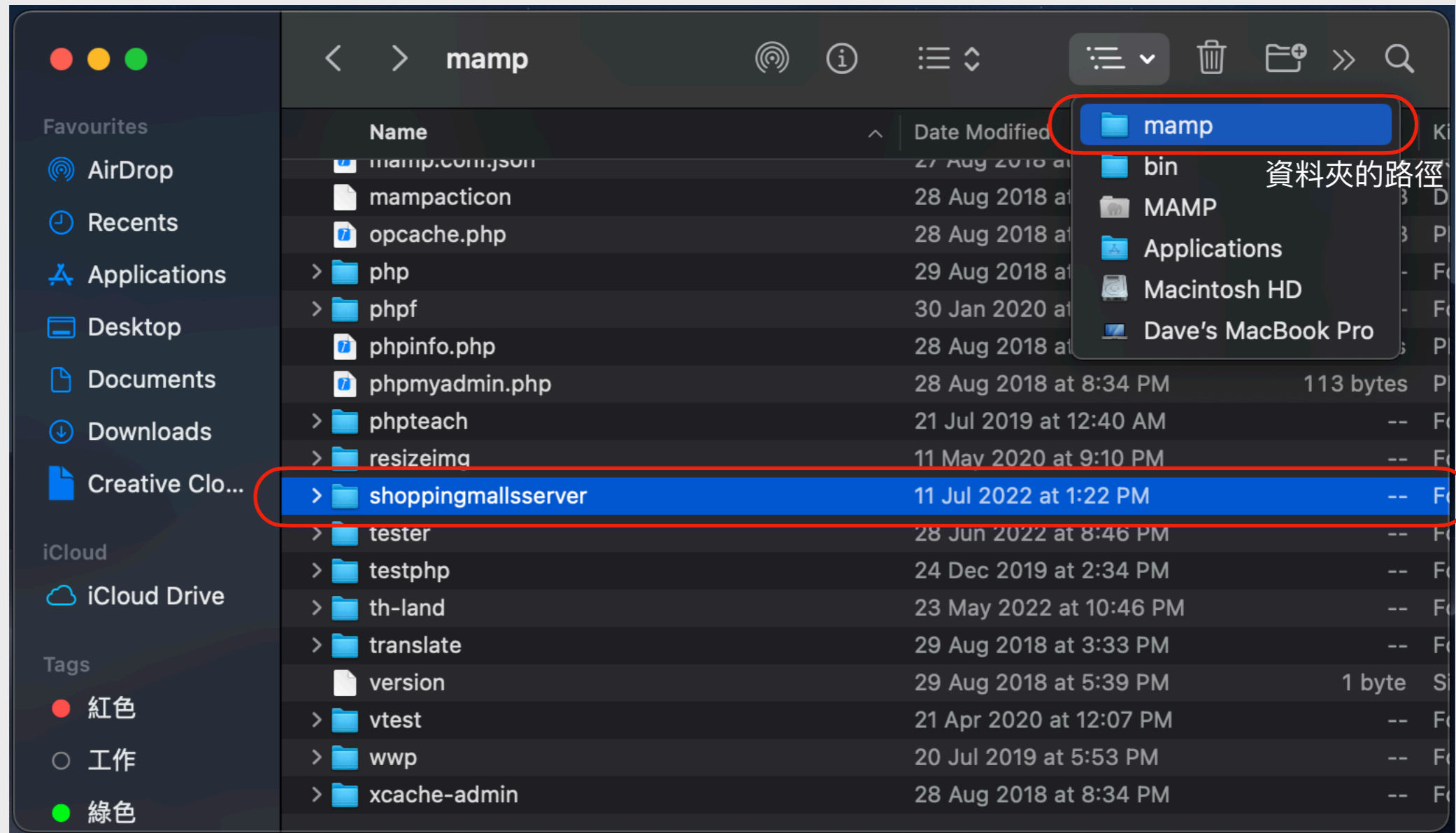
```
CREATE TABLE shopping_malls ( raw_id int(11) NOT NULL, mall_name varchar(35) COLLATE utf8mb4_bin NOT NULL, mall_district varchar(15) COLLATE utf8mb4_bin DEFAULT NULL, mall_lat double NOT NULL DEFAULT '0', mall_lon double NOT NULL DEFAULT '0', mall_addr varchar(60) COLLATE utf8mb4_bin DEFAULT NULL, mall_isbig tinyint(1) NOT NULL DEFAULT '0', mall_status enum('normal','preparing','moved','tmp-closed','closed') COLLATE utf8mb4_bin NOT NULL DEFAULT 'normal', created_timestamp timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin;
```
- 464 rows inserted. (Query took 0.0069 seconds.)**
The SQL query shown is:

```
INSERT INTO shopping_malls (raw_id, mall_name, mall_district, mall_lat, mall_lon, mall_addr, mall_isbig, mall_status, created_timestamp) VALUES (1, '康盛花園商場', '將軍澳', 22.319778810368334, 114.25309928808232, '將軍澳寶琳北路1號康盛花園', 0, 'normal', '2021-04-25 11:43:05'), (2, '翠林新城', '將軍澳', 22.322606960496227, 114.24915413411578, '將軍澳翠林邨翠琳路11號', 0, 'normal', '2021-04-25 11:43:05'), (3, '慧安商場', '將軍澳', 22.324215110013157, 114.25415525501846, '將軍澳寶林毓雅里9號', 0, 'normal', '2021-04-25 11:43:05'), (4, '慧星匯', '將軍澳', 22.324247573426288, 114.25418206512319, '將軍澳寶林毓雅里9號慧安商場一樓', 0, 'normal', '2021-04-25 11:43:05'), (5, '寶林商場', '將軍澳', 22.325097983286835, 114.25616749930711, '將軍澳寶林寶林邨', 0,
```
- 351 rows inserted. (Query took 0.0027 seconds.)**
The SQL query shown is:

```
INSERT INTO shopping_malls (raw_id, mall_name, mall_district, mall_lat, mall_lon, mall_addr, mall_isbig, mall_status, created_timestamp) VALUES (465, '又一廣場', '九龍塘/又一村', 22.331313582100066, 114.17271942595822, '海棠路21-31號', 0, 'normal', '2021-04-25 11:43:07'), (466, '華麗購物廣場', '深水埗', 22.330735454990272, 114.16489875294434, '深水埗南昌街', 0, 'normal', '2021-04-25 11:43:07'), (467, '西九龍中心', '深水埗', 22.33111765794281, 114.15980825479394, '深水埗欽州街37K', 1, 'normal', '2021-04-25 11:43:07'), (468, '黃金電腦商場', '深水埗', 22.332031885307075, 114.16224602595837, '深水埗福華街黃金大廈B/F-G/F', 1, 'normal', '2021-04-25 11:43:07'), (469, '高登電腦中心', '深水埗', 22.332031885307075, 114.16224602595837, '深水埗福華街黃金大廈1/F', 1,
```

成功匯入，不同的 Table 已被建立起來。

項目實習-附近商場APP-移到伺服器



現在，我們就把教材中的整個 shoppingmallserver 資料夾搬或複製到 MAMP 或 AppServ 或 WAMP 的伺服器端資料夾內。

項目實習-附近商場APP-數據庫連接設定

```
Code File Edit Selection View Go Run Terminal Window Help
db_api.php — shoppingmallserver
EXPLORER
OPEN EDITORS
db_api.php
SHOPPINGMALLSERVER
db_api.php
ser.php
serapi.php
shoppingmalls.sql
db_api.php
1 <?php
2
3 function executeCommandToDB($_sql, $_variables, $_getLastInsertId=false) {
4     $db=new PDO('mysql:host=localhost;dbname=shoppingmalls;charset=utf8mb4',
5     'root', 'root'); 到 db_api.php 確定數據庫的密碼正確無誤
6     $db->exec("SET CHARACTER SET utf8mb4");
7     $db->exec("SET NAMES utf8mb4");
8     $db->exec("SET SESSION time_zone='+8:00'");
9     $sth=$db->prepare($_sql);
10    //$sth->execute("set names utf8mb4");
11    $count=0;
12    if($_variables!=null) {
13        $count=$sth->execute($_variables);
14    }
15    else $count=$sth->execute();
16    if($_getLastInsertId)$last_id=$db->lastInsertId();
17    $db=null;
18    if(!$getLastInsertId && $count>0)return true;
19    elseif($_getLastInsertId && $count>0)return $last_id;
20    return false;
21 }
22 function retrieveFromDB($_sql, $_variables) {
23     $db=new PDO('mysql:host=localhost;dbname=shoppingmalls;charset=utf8mb4',
24     'root', 'root'); 到 db_api.php 確定數據庫的密碼正確無誤
25     $db->exec("SET CHARACTER SET utf8mb4");
```

Ln 1, Col 1 Spaces: 4 UTF-8 LF PHP Go Live

前後端互動實作

項目實習-附近商場APP-加入程式碼(1)

載入AlertController及NavController到add-mall.page.ts的頂端

```
import { AlertController, NavController } from '@ionic/angular';
```

```
TS home.page.ts TS add-mall.page.ts X
src > app > add-mall > TS add-mall.page.ts > AddMallPage
1 import { Component, OnInit } from '@angular/core';
2 import { AlertController, NavController } from '@ionic/angular';
3
```

然後到constructor的開關小括號中間加定義兩個Controller物件變數

```
constructor(
  public alertController:AlertController,
  public navCtrl:NavController
){}
```

```
32 constructor(
33   public alertController:AlertController,
34   public navCtrl:NavController
35 ){}
```

項目實習-附近商場APP-加入程式碼(2)

然後到add-mall.page.ts的底端加入Ajax程式碼

```
async createAjax(url, data, method='POST'){
  const response = await fetch(url, {
    method:method.toUpperCase(), headers: {
      'Accept': 'application/json',
      'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',
    }, body:new URLSearchParams(data)
  });
  return response.ok ? response.json():null;
}
```

```
68   async createAjax(url, data, method='POST'){
69     const response = await fetch(url, {
70       method:method.toUpperCase(), headers: {
71         'Accept': 'application/json',
72         'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',
73       }, body:new URLSearchParams(data)
74     });
75     return response.ok ? response.json():null;
76   }
```

項目實習-附近商場APP-加入程式碼(3)

在add-mall.page.ts的addMall函式的底端加入呼叫Ajax函式及通知用戶的程式碼

```
const serPath='http://localhost:8888/MAMP/shoppingmallserver/ser.php';
this.createAjax(serPath,data).then(response=>{           (請留意serPath在MAMP和AppServ是不同的)
  if(response.succeed){
    this.alertController.create({
      header: "成功",
      message: "已成功新增商場",
      buttons: [{text:"知道", role:"cancel", handler:()=>{
        this.navController.pop();
      }},]
    }).then(a=>{a.present();});
  }
});
```

```
43  addMall() {
44    const data = {
45      mall_name: this.mallName,
46      mall_district: this.mallDistrict,
47      mall_lat: this.mallLat,
48      mall_lon: this.mallLon,
49      mall_addr: this.mallAddr,
50      mall_isbig: this.mallType == 'big',
51      mall_status: this.mallStatus
52    };
53    console.log('add_mall', data);
54    const serPath='http://localhost:8888/MAMP/shoppingmallserver/ser.php';
55    this.createAjax(serPath,data).then(response=>{
56      if(response.succeed){
57        this.alertController.create({
58          header: "成功",
59          message: "已成功新增商場",
60          buttons: [{text:"知道", role:"cancel", handler:()=>{
61            this.navController.pop();
62          }},]
63        }).then(a=>{a.present();});
64      }
65    });
66  }
```

項目實習-附近商場APP-加入程式碼(4)

在add-mall.page.ts的addMall函式的data變數中加上一組鍵值：

`service: 'add_mall',`

```
43  addMall() {
44    const data = {
45      service: 'add_mall',
46      mall_name: this.mallName,
47      mall_district: this.mallDistrict,
48      mall_lat: this.mallLat,
49      mall_lon: this.mallLon,
50      mall_addr: this.mallAddr,
51      mall_isbig: this.mallType == 'big',
52      mall_status: this.mallStatus
53    };
54    console.log('add_mall', data);
```

項目實習-附近商場APP-加入程式碼(5)

載入LoadingController到home.page.ts的頂端

```
import { LoadingController } from '@ionic/angular';
```

```
TS home.page.ts M • TS add-mall.page.ts U
src > app > home > TS home.page.ts > HomePage
1 import { Component } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { LoadingController } from '@ionic/angular';
4
```

然後到constructor的開關小括號中間加定義LoadingController物件變數

```
public loadingController:LoadingController
```

```
22 constructor(
23   public router:Router,
24   public loadingController:LoadingController
25 ) { }
```


項目實習-附近商場APP-加入程式碼(6)

把home.page.ts原先定義了的變數換成下列變數：

```
lat=0;  
lon=0;  
existingAddr="";  
loadMallsLimit=25;  
nearbyShoppingMalls=[];  
displayShoppingMalls=[];
```

```
10 export class HomePage {  
11  
12     lat=0;  
13     lon=0;  
14     existingAddr="";  
15     loadMallsLimit=25;  
16     nearbyShoppingMalls=[];  
17     displayShoppingMalls=[];  
18  
19     constructor(  

```

項目實習-附近商場APP-加入程式碼(7)

在home.page.ts加入getLocation函式

```
getLocation(){  
  if(navigator.geolocation){  
    const loading=this.loadingController.create({  
      message: '定位中...'  
    }).then(l=>{  
      l.present();  
      navigator.geolocation.getCurrentPosition(position=>{  
        this.lat=position.coords.latitude;  
        this.lon=position.coords.longitude;  
        this.loadShoppingMalls();  
        this.getAddress();  
        l.dismiss();  
      });  
    });  
  }  
}
```

```
32 getLocation(){  
33   if(navigator.geolocation){  
34     const loading=this.loadingController.create({  
35       message: '定位中...'  
36     }).then(l=>{  
37       l.present();  
38       navigator.geolocation.getCurrentPosition(position=>{  
39         this.lat=position.coords.latitude;  
40         this.lon=position.coords.longitude;  
41         this.loadShoppingMalls();  
42         this.getAddress();  
43         l.dismiss();  
44       });  
45     });  
46   }  
47 }
```

項目實習-附近商場APP-加入程式碼(8)

在home.page.ts加入loadShoppingMalls函式

```
loadShoppingMalls(){  
    (請留意serPath在MAMP和AppServ是不同的)  
    const serPath='http://localhost:8888/MAMP/shoppingmallserver/ser.php';  
    this.createAjax(serPath,{service:'get_nearby_malls',lat:this.lat,lon:this.lon}).then(response=>{  
        if(response.succeed){  
            this.nearbyShoppingMalls=response.msg.nearbyShoppingMalls;  
            this.displayShoppingMalls=[];  
            for(let i=0; i<this.loadMallsLimit; i++){  
                this.displayShoppingMalls.push(this.nearbyShoppingMalls[i]);  
            }  
        }  
    });  
}
```

```
51 loadShoppingMalls(){  
52     const serPath='http://localhost:8888/MAMP/shoppingmallserver/ser.php';  
53     this.createAjax(serPath,{service:'get_nearby_malls',lat:this.lat,lon:this.lon}).then(response=>{  
54         if(response.succeed){  
55             this.nearbyShoppingMalls=response.msg.nearbyShoppingMalls;  
56             this.displayShoppingMalls=[];  
57             for(let i=0; i<this.loadMallsLimit; i++){  
58                 this.displayShoppingMalls.push(this.nearbyShoppingMalls[i]);  
59             }  
60         }  
61     });  
62 }
```

項目實習-附近商場APP-加入程式碼(9)

在home.page.ts加入tryLoadShoppingMalls函式

```
tryLoadShoppingMalls(infiniteScroll){
  setTimeout(()=>{
    const l=this.displayShoppingMalls.length+this.loadMallsLimit;
    for(let i=this.displayShoppingMalls.length; i<l; i++){
      this.displayShoppingMalls.push(this.nearbyShoppingMalls[i]);
    }
    infiniteScroll.target.complete();
  },300);
}
```

```
62 tryLoadShoppingMalls(infiniteScroll){
63   setTimeout(()=>{
64     const l=this.displayShoppingMalls.length+this.loadMallsLimit;
65     for(let i=this.displayShoppingMalls.length; i<l; i++){
66       this.displayShoppingMalls.push(this.nearbyShoppingMalls[i]);
67     }
68     infiniteScroll.target.complete();
69   },300);
70 }
```

項目實習-附近商場APP-加入程式碼(10)

在home.page.ts加入getAddress函式

```
getAddress(){
  const serPath='https://nominatim.openstreetmap.org/reverse?
format=json&lat='+this.lat+'&lon='+this.lon+'&zoom=18&addressdetails=1&accept-language=zh-
hant';
  this.createAjax(serPath,{}).then(latlngData=>{
    this.existingAddr=this.getOsmAddress(latlngData);
    console.log(this.existingAddr);
  });
}
```

```
72 getAddress(){
73   const serPath='https://nominatim.openstreetmap.org/reverse?format=json&lat='+this.lat
74   + '&lon='+this.lon+'&zoom=18&addressdetails=1&accept-language=zh-hant';
75   this.createAjax(serPath,{}).then(latlngData=>{
76     this.existingAddr=this.getOsmAddress(latlngData);
77     console.log(this.existingAddr);
78   });
79 }
```

項目實習-附近商場APP-加入程式碼(11)

在home.page.ts加入getOsmAddress函式

```
getOsmAddress(addrData) {  
  let addr = addrData.display_name;  
  if (addrData.display_name.indexOf(',') !== -1) {  
    const addrs = addrData.display_name.split(', ');  
    addr = addrs[0];  
    if (addr.indexOf('第') !== -1 && addr.indexOf('座') !== -1) {  
      addr = addrs[1];  
      if (addrData.address.suburb !== null) {  
        addr = addrData.address.suburb + addr;  
      }  
    }  
  }  
  if (addr.indexOf('House') !== -1) {  
    addr = addrs[1];  
    if (addrData.address.suburb !== null) {  
      addr = addrData.address.suburb + addr;  
    }  
  }  
  if (addr.indexOf('Tower') !== -1) {  
    addr = addrs[1];  
    if (addrData.address.suburb !== null) {  
      addr = addrData.address.suburb + addr;  
    }  
  }  
  return addr;  
}
```

```
81  getOsmAddress(addrData) {  
82    let addr = addrData.display_name;  
83    if (addrData.display_name.indexOf(',') !== -1) {  
84      const addrs = addrData.display_name.split(', ');  
85      addr = addrs[0];  
86      if (addr.indexOf('第') !== -1 && addr.indexOf('座') !== -1) {  
87        addr = addrs[1];  
88        if (addrData.address.suburb !== null) {  
89          addr = addrData.address.suburb + addr;  
90        }  
91      }  
92      if (addr.indexOf('House') !== -1) {  
93        addr = addrs[1];  
94        if (addrData.address.suburb !== null) {  
95          addr = addrData.address.suburb + addr;  
96        }  
97      }  
98      if (addr.indexOf('Tower') !== -1) {  
99        addr = addrs[1];  
100       if (addrData.address.suburb !== null) {  
101         addr = addrData.address.suburb + addr;  
102       }  
103     }  
104   }  
105   return addr;  
106 }  
107
```

項目實習-附近商場APP-加入程式碼(12)

在home.page.ts加入createAjax函式

```
async createAjax(url, data, method='POST'){
  const response = await fetch(url, {
    method:method.toUpperCase(), headers: {
      'Accept': 'application/json',
      'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',
    }, body:new URLSearchParams(data)
  });
  return response.ok ? response.json():null;
}
```

```
108  async createAjax(url, data, method='POST'){
109      const response = await fetch(url, {
110          method:method.toUpperCase(), headers: {
111              'Accept': 'application/json',
112              'Content-Type': 'application/x-www-form-urlencoded;charset=UTF-8',
113          }, body:new URLSearchParams(data)
114      });
115      return response.ok ? response.json():null;
116  }
```

項目實習-附近商場APP-加入程式碼(13)

在home.page.ts的constructor加入 `this.getLocation();` 以呼叫獲取地理位置的函式

```
19     constructor(  
20         public router:Router,  
21         public loadingController:LoadingController  
22     ) {  
23         this.getLocation();  
24     }
```

在home.page.html底端加進滑動載入更多商場的HTML元件

```
<ion-infinite-scroll (ionInfinite)="tryLoadShoppingMalls($event);">  
  <ion-infinite-scroll-content></ion-infinite-scroll-content>  
</ion-infinite-scroll>
```

```
28     <ion-infinite-scroll (ionInfinite)="tryLoadShoppingMalls($event);">  
29         <ion-infinite-scroll-content></ion-infinite-scroll-content>  
30     </ion-infinite-scroll>  
31  
32 </ion-content>
```


項目實習-附近商場APP-大功告成

The screenshot displays a web browser window with the following elements:

- Browser Interface:** Chrome browser, address bar shows `localhost:8100/#/home`, dimensions set to iPhone SE (375x667).
- App UI:** A mobile app titled "附近商場" (Nearby Malls) with a blue header. The main content shows the current location as "愉田苑商場" and a list of nearby malls, including "愉翠商場", "愉田苑商場", "置富第一城", "都會廣場", "置富第一城·樂薈", "富豪花園商場", and "花園城三期".
- Development Console:** Shows messages from Webpack Dev Server: "[webpack-dev-server] Live Reloading enabled." and "Angular is running in development mode. Call enableProdMode() to enable production mode." It also lists source files like `index.js:551`, `core.mjs:25563`, and `home.page.ts:79`.

「附近商場APP」大功告成！



THANKS