



# 使用 CORS



designed by freepik

Estimated time:

50 min.

資訊工業策進會 Institute for Information Industry

【Key Points】：

## 學習目標

- 18-1: 什麼是 CORS
- 18-2: 設定及一般使用 CORS
- 18-3: 使用白名單



18-1

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

本模組將先提到什麼是同源政策  
再接著說明 CORS 如何提供跨域許可  
接下來說明 express 搭配 cors 模組的基礎用法  
並且用 Fiddler 監看 CORS 的過程  
最後說明如何以白名單過濾用戶端  
最後會有「自我評量」與「課後練習步驟(Lab)」，請各位依照說明進行練習。

### 【Key Points】：

什麼是 CORS  
設定及一般使用 CORS  
使用白名單

## 18-1：什麼是 CORS

- Same-origin Policy ( SOP ) 同源政策
- 什麼情況下符合同源政策
- JSONP的運作原理與流程
- 跨來源資源共用 ( CORS )



designed by freepik



designed by freepik

18-2

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

這一節著重於觀念與原理：

- Same-origin Policy ( SOP ) 同源政策
- 什麼情況下符合同源政策
- 什麼情況下違反同源政策
- JSONP的運作原理與流程
- 跨來源資源共用 ( CORS )

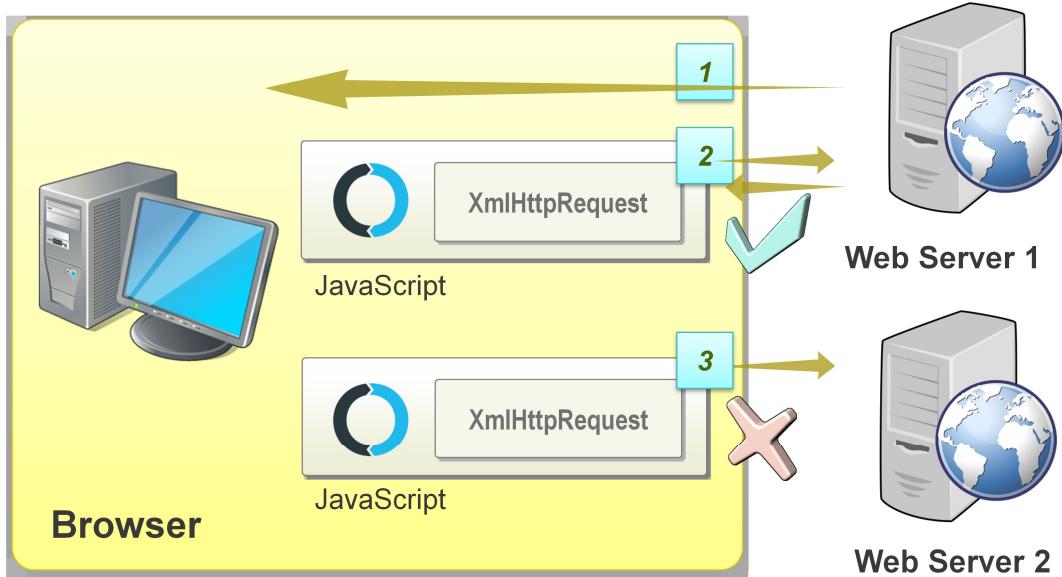
【Key Points】：

Same-origin Policy ( SOP ) 同源政策

JSONP

跨來源資源共用 ( CORS )

# Same-origin Policy ( SOP ) 同源政策



電腦、伺服器等素材圖片來源: <http://pngimg.com>

1-3

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

關於跨域與同源政策Same-origin Policy ( SOP ) 的更多說明:

- 基於安全理由，瀏覽器會限制跨域存取，以致於許多用戶端無法使用我們提供的Web API 服務。
- 如果瀏覽器從例如 WebServer1 取回網頁，網頁的 JavaScript 利用 XMLHttpRequest 物件存取網路功能時，只許呼叫原本的 WebServer1，不可跨到其他機器。(註)更詳細的規則，下一張投影片會講到。
- 即使程式設計師使用例如jQuery之類的函式庫，同樣也受此規則限制，畢竟，JavaScript的執行環境是瀏覽器。
- 事實上，HTTP Request 還是發得出來，遠端的伺服器也收得到。
- 但瀏覽器如果沒有得到Access-Control-Allow-Origin許可，會在最後一刻攔下資料而不交給我們的JavaScript程式，以致於感覺像是石沈大海

## 【Key Points】：

基於安全理由，瀏覽器會限制跨域存取

JavaScript 利用 XMLHttpRequest 物件，將受限於同源政策

瀏覽器如果沒有得到Access-Control-Allow-Origin許可，會在最後一刻攔下資料

# 什麼情況下符合同源政策

如果連到 <http://www.beauty.com/index.html> 取回網頁，  
網頁的JavaScript想連繫下列網址，結果分別是 --

網址	結果	說明
http://www.beauty.com/hello/Jeremy	○	不同資料夾/文件，沒問題
https://www.beauty.com/hello/Jeremy	✗	http 與 https 通訊協定不同
<a href="http://www2.beauty.com/hello/Jeremy">http://www2.beauty.com/hello/Jeremy</a>	✗	Host 機器名稱不同
http://www.beauty.com:81/hello/Jeremy	✗	埠號不同
http://user:pass@www.beauty.com/hello/Jeremy	○	不同使用者，可以

簡單地說，通訊協定、機器名稱、埠號三者都必須相同。

18-4



判斷是否同源的主要依據是HostName 機器名稱，[www.fcu.edu.tw](http://www.fcu.edu.tw) 與 www2.fcu.edu.tw 不同源。

其次是埠號是否相同

通訊協定也必須相同，http 與 https 將視為不同源

簡單地說，通訊協定、機器名稱、埠號三者都必須相同。

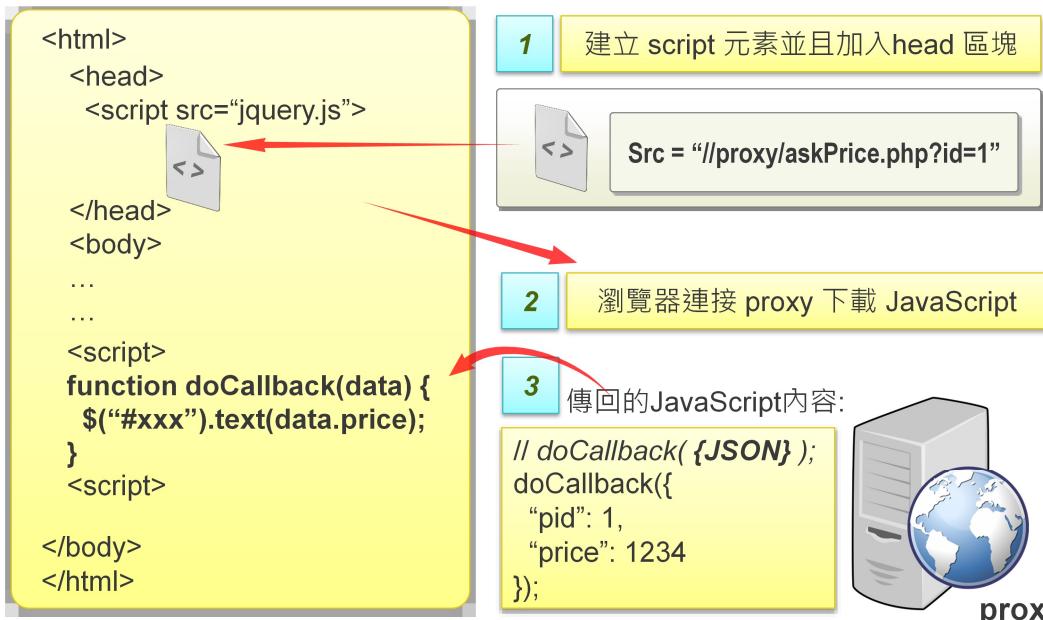
不同資料夾，沒關係，算同源。

不同使用者，也可以，算同源。

## 【Key Points】：

- HostName 機器名稱
- 埠號
- 通訊協定

# JSONP的運作原理與流程



電腦、伺服器等素材圖片來源: <http://pngimg.com>

1-5

III 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

同源政策有例外，圖片可以跨域抓回，JavaScript也可跨域載入。

JSONP (JSON Padding) 的基本想法是透過代理伺服器，間接進行網路存取。

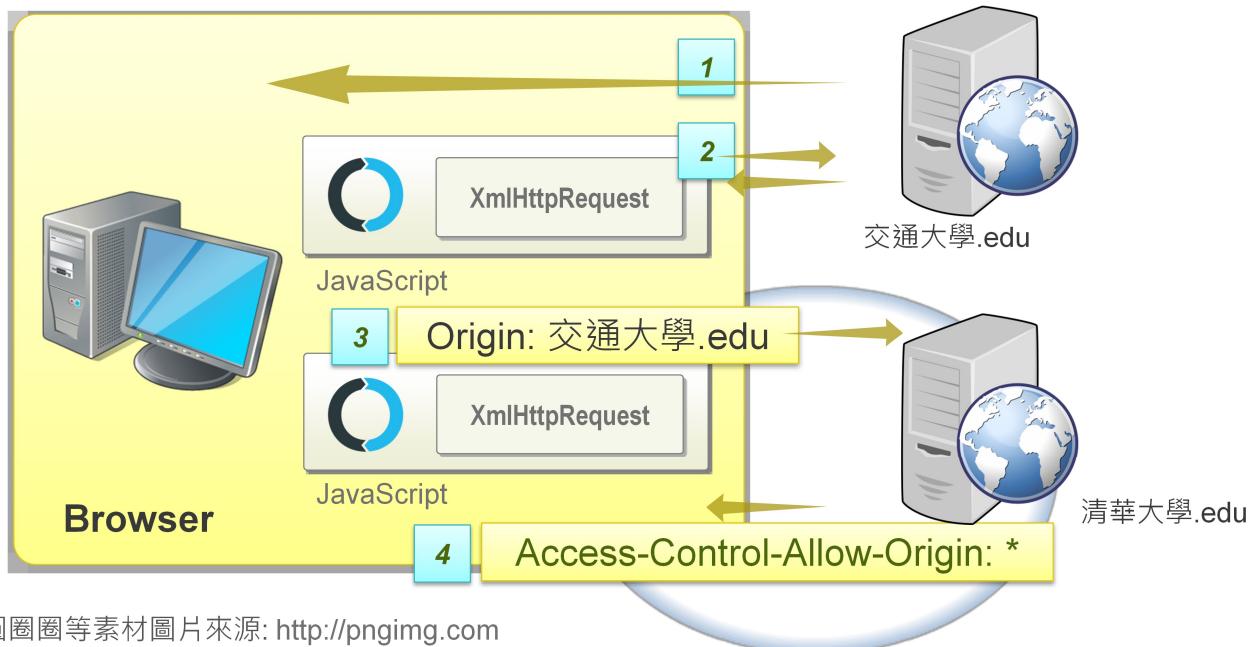
1. 有跨域需要時，動態建立 script 元素 ( script 的 src 屬性值指向代理伺服器的代理程式 )
2. 將動態建立的 script 元素加入 head 區塊
3. 瀏覽器因此嘗試從代理伺服器下載JavaScript，此舉等於等於執行遠端的代理程式
4. 遠端代理程式將結果包裝成函式呼叫，代抓的資料則是參數內容
5. 瀏覽器執行代理伺服器傳回的程式，在瀏覽器畫面輸出執行結果

遠端代理程式將結果包裝(padding)成函式呼叫，代抓的資料則是參數內容(通常是JSON)，這也是這招叫 JSONP 的原因。

## 【Key Points】：

基本想法是透過代理伺服器  
代理程式將結果包裝成函式呼叫  
瀏覽器執行代理伺服器傳回的程式

# 跨來源資源共用 ( CORS )



電腦、圓圈圖等素材圖片來源: <http://pngimg.com>

1-6

財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

- Same Origin Policy 限制 JavaScript 進行跨網域請求。
- JSONP 雖可跨域，但畢竟對大部分工程師來說，太曲折了。
- CORS 是 HTML5制定的新標準
- 瀏覽器發現會跨域時，發出的 HTTP 請求就會帶入 Origin header，例如：  
Origin: http://交通大學.edu
- 如果伺服端願意提供服務，則傳回的 HTTP header 會有：  
Access-Control-Allow-Origin: http://交通大學.edu  
或者  
Access-Control-Allow-Origin: \*
- 瀏覽器得到 Access-Control-Allow-Origin 等於是得到一個綠燈的放行訊號，就不攔截訊息回傳。

## 【Key Points】：

Same Origin Policy 限制 JavaScript 進行跨網域請求  
CORS 是 HTML5制定的新標準  
強調一下 Access-Control-Allow-Origin 的作用。

## 18-2：設定及一般使用 CORS

- 安裝express與cors模組
- 掛載cors到express建立的Web Server
- 安裝與使用 Fiddler
- 檢視 HTTP 標頭內容



designed by freepik



designed by freepik

18-7

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

接下來，我們會學到：

- 安裝express與cors模組
- 掛載cors到express建立的Web Server
- 安裝 Fiddler
- 使用 Fiddler
- 以 Fiddler 檢視 HTTP 標頭內容

【Key Points】：

掛載cors到express建立的Web Server  
安裝與使用 Fiddler  
以 Fiddler檢視 HTTP 標頭內容

## 安裝express與cors模組

- Express 是 Node.js 界最流行的 Web 開發框架 / 模組
- cors 也是一個 Node.js 外掛模組，作用是搭配 Express 協同處理 HTTP 請求
- cors 主要功能是在回應給用戶端 HTTP 文件標題區加入：  
**Access-Control-Allow-Origin: \***
- 安裝指令：  
**npm install express**  
**npm install cors**

18-8



- cors 是一個 Node.js 外掛模組
- cors 主要功能是在回應給用戶端 HTTP 文件的標題區加入下列標頭：  
**Access-Control-Allow-Origin: \***
- 自動依據 HTTP 標題區的 Origin 的標頭資訊，判斷訪客的網域是否在白名單之列。
- 如果不在白名單之列，就不會輸出 Access-Control-Allow-Origin 標頭。
- 搭配 Express 協同處理 HTTP 請求，所以，Express 也必須安裝
  - npm install express
  - npm install cors

【Key Points】：

**npm install cors**

依據 HTTP 標題區的 Origin 的標頭資訊，判斷訪客的網域是否在白名單  
如果不在白名單之列，就不會輸出 Access-Control-Allow-Origin 標頭

# 掛載 cors 到 express 建立的 Web Server

1 引用並呼叫 express

2 引用並呼叫 cors()，  
將 cors 物件傳入 express 的  
use 方法

```
var express = require("express");
var app = express();
```

```
// 允許跨域使用本服務
var cors = require("cors");
app.use(cors());
```

```
app.listen(80);
```

```
app.get("/hello/:text",
  function (request, response) {
    response.send("Hello! " +
      request.params.text);
});
```

1-9



1. 引用 Express 模組

2. 呼叫 express() 建立 Web 應用程式

```
var express = require("express");
var app = express();
```

3. 引用 cors 模組

4. 呼叫 cors()

5. 將 cors() 傳回的物件帶入 express 的 use 方法

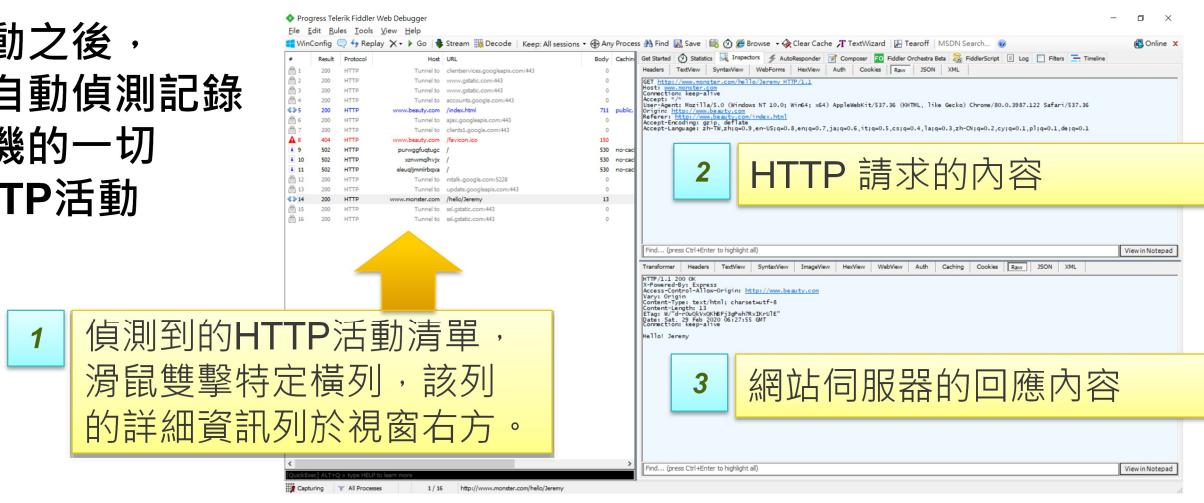
```
var cors = require("cors");
app.use(cors());
```

【Key Points】：

```
var express = require("express");
var app = express();
var cors = require("cors");
app.use(cors());
```

# 安裝與使用 Fiddler

- Fiddler 是一套HTTP監看與除錯工具
- 下載網址 <https://www.telerik.com/fiddler>
- 啟動之後，會自動偵測記錄本機的一切HTTP活動



18-10

財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

Fiddler 是一套HTTP監看與除錯工具

下載網址 <https://www.telerik.com/fiddler>，下載回來的安裝程式，依安裝程式預設值進行安裝即可。

啟動之後，會自動偵測記錄本機的一切HTTP活動，但 https 因為內容加密過，無法看到內容。

偵測到的HTTP活動清單會列於視窗左側，滑鼠雙擊特定橫列，該列的詳細資訊列於視窗右方，右上為HTTP請求的內容，網站伺服器的回應內容。

Raw 頁籤有最原始的內容，可看到Client/Server交換的原貌。

本身也是一個 HTTP Client 工具，例如 Composer 頁籤的功能，可對外模擬 HTTP 的 GET、POST、PUT、DELETE 等指令。

## 【Key Points】：

Fiddler 是一套HTTP監看與除錯工具

Raw 頁籤有最原始的內容，可看到Client/Server交換的原貌

本身也是一個 HTTP Client 工具

## 檢視 HTTP 標頭內容

- 假設瀏覽器連到 <http://www.beauty.com> 取回 index.html
- Index.html 的 JavaScript 以 AJAX 技術連繫 <http://www.monster.com> · Fiddler 偵測到的活動記錄摘要如下：

GET http://www.monster.com/hello/Jeremy HTTP/1.1

Host: [www.monster.com](http://www.monster.com)

Origin: <http://www.beauty.com>

Referer: <http://www.beauty.com/index.html>

HTTP/1.1 200 OK

X-Powered-By: Express

Access-Control-Allow-Origin: <http://www.beauty.com>

Content-Length: 13

Hello! Jeremy

18-11



- 假設瀏覽器連到 <http://www.beauty.com> 取回 index.html

- JavaScript 要跨域時，跨域的那次 HTTP 請求會被瀏覽器額外帶入 Origin 這個標頭，例如：  
Origin: <http://www.beauty.com>

- 如果伺服端願意提供服務，則傳回的 HTTP header 會有：

Access-Control-Allow-Origin: <http://www.beauty.com>

- 如果伺服端願意提供服務，也可簡單回應：

Access-Control-Allow-Origin: \*

- 星號代表所有的、全部的意思

### 【Key Points】：

跨域的那次 HTTP 請求會被瀏覽器額外帶入 Origin 這個標頭

如果伺服端願意提供服務，回應 Access-Control-Allow-Origin 標頭

上述例子以課後練習為例，如果時間來得及，當場以之示範一下如何用 Fiddler 偵測 HTTP 活動。

## 18-3：使用白名單

- 白名單與黑名單
- 如何利用**cors**設定白名單
- 複習增查修刪(CRUD)
- 設定Client端允許使用的HTTP方法



designed by freepik



designed by freepik

18-12

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

這一節學的是 cors 比較進階的用法：

- 會提到白名單與黑名單的思維模式有何不同
- 如何利用**cors**設定白名單
- 複習增查修刪(CRUD)
- 設定Client端允許使用的HTTP方法
- 不在允許之列的方法，不提供服務

### 【Key Points】：

白名單與黑名單的思維模式有何不同

如何利用**cors**設定白名單

設定Client端允許使用的HTTP方法

# 白名單與黑名單

- 檢驗資料正確與否的兩種哲學思維
- **黑名單**  
列入黑名單的確定是惡人，名單之外的皆是好人。
- **白名單**  
列名在白名單的人是好人，其他都是惡人。
- 資安領域經常採料敵從寬的態度，以白名單設計安全系統。

18-13



檢驗資料正確與否的兩種哲學思維。

## 黑名單

列入黑名單的確定是惡人，名單之外的皆是好人。

例如：岳不群是惡人。

## 白名單

列名在白名單的人是好人，其他都是惡人。

例如：某位女孩在成長的過程，被教育說只有把拔是好人，其餘的男人沒一個是好東西。

資安領域經常採料敵從寬的態度，以白名單方式設計安全系統。

白名單的作法比黑名單安全。

## 【Key Points】：

兩種哲學思維

黑名單與白名單有何不同

白名單比較安全

# 如何利用cors設定白名單

- 建立一個物件，該物件需有一個名為 origin 的屬性，屬性值為陣列
- 陣列的每一個元素各是一道「信得過」的網址

```
const corsOptions = {  
    origin: [  
        'http://www.beauty.com',  
        'http://localhost',  
    ]};
```

- 掛載到 express 伺服器的寫法：

```
app.use( cors( corsOptions ) );
```

18-14



建立內含白名單陣列的選項物件：

```
const corsOptions = {  
    origin: [  
        'http://www.beauty.com',  
        'http://localhost',  
    ]};
```

選項物件的origin 屬性值是一個陣列，陣列的各個元素都是字串，每一個字串對應一個白名單內的網址

呼叫 cors() 時，傳入上述的選項物件，

將 cors() 傳回的物件帶入 express 的 use 方法。

```
app.use( cors( corsOptions ) );
```

## 【Key Points】：

選項物件有 origin 屬性

origin 屬性值是陣列，內容為白名單網址

以 express 的 use 方法掛入 cors

# 增查修刪(CRUD)

- CRUD 為 Create、Read、Update、Delete (增查修刪)的簡寫
- Web API 透過下列四個 HTTP 方法提供「增查修刪」功能：
  - Create → POST
  - Read → GET
  - Update → PUT
  - Delete → DELETE
- 可以設定 Web API 只提供哪些功能

18-15



CRUD 為 Create、Read、Update、Delete (增查修刪)的簡寫

Web API 透過下列四個 HTTP 方法提供「增查修刪」功能：

Create → POST

Read → GET

Update → PUT

Delete → DELETE

我們可以設定 Web API 只提供哪些功能。

【Key Points】：

CRUD 為 Create、Read、Update、Delete 的簡寫

Web API 透過 POST, GET, PUT, DELETE 方法提供「增查修刪」功能

我們可以設定 Web API 只提供哪些功能

## 限定HTTP方法清單

- 除了網址白名單，也能限定用戶端只能使用Web API哪些方法：

```
const corsOptions = {
  origin: [
    'http://www.beauty.com',
    'http://localhost',
  ],
  methods: 'GET, PUT, POST,DELETE'
};
app.use(cors(corsOptions));
```

18-16



Web API 透過 POST、GET、PUT、DELETE 四個 HTTP 方法提供「增查修刪」功能

當我們設計 Web API 服務時，除了以網址白名單限制來源，也能限定用戶端只能使用哪些方法。

作法時，在建立 cors 選項物件時，指定 methods 屬性，屬性值為字串型態，將有意同意的各個 HTTP 方法以逗號分隔。例如：

```
const corsOptions = {
  origin: [
    'http://www.beauty.com',
    'http://localhost',
  ],
  methods: 'GET, PUT, POST,DELETE'
};
app.use(cors(corsOptions));
```

【Key Points】：

POST、GET、PUT、DELETE 四大 HTTP 方法

透過 methods 屬性進行功能過濾

methods 屬性值為字串型態

# Summary〈精華回顧〉

- Same-origin Policy ( SOP ) 同源政策
- 什麼情況下符合同源政策
- 跨來源資源共用 ( CORS ) 的兩個重要 HTTP 標頭:
  - Origin
  - Access-Control-Allow-Origin
- cors模組如何協助產生 Access-Control-Allow-Origin
- cors如何依據 Origin 配合白名單過濾訪客
- 設計 Web API 服務時，除了以網址白名單限制來源，也能限定Web API 服務只提供增查修刪的哪些功能。



18-17

 財團法人資訊工業策進會  
INSTITUTE FOR INFORMATION INDUSTRY

因為 Same-origin Policy ( SOP ) 同源政策，Web 相對比較安全。我們探討什麼情況下符合同源政策。

但網際網路傾向於分享開放；商務方面，我們也希望 Web API 的客人多一點，方便一點。

此時，透過JSONP可以跨域。跨來源資源共用 ( CORS ) 則是更為簡潔的作法。

跨來源資源共用 ( CORS ) 的兩個重要 HTTP 標頭：

Origin

Access-Control-Allow-Origin

最後，我們學到 Web API 如何依據 Origin 配合白名單過濾訪客，以及如何限定Web API 服務只提供增查修刪的哪些功能。

【Key Points】：

Same-origin Policy ( SOP )

跨來源資源共用 ( CORS )

cors 白名單

## 程式練習題

- 題目一：如何引用 cors 模組？  
利用 cors 協助進行跨來源資源共用 ( CORS )
- 題目二：如何使用 cors 的白名單功能？  
想利用 cors 的白名單功能幫我們過濾對象，補正缺漏的程式
- 題目三：如何使用 cors 的HTTP方法過濾功能  
想利用 cors 的HTTP方法過濾功能，補正缺漏的程式

18-18



**題目一：如何引用 cors 模組？**

利用 cors 協助進行跨來源資源共用 ( CORS )

**題目二：如何使用 cors 的白名單功能？**

想利用 cors 的白名單功能幫我們過濾對象，補正缺漏的程式

**題目三：如何使用 cors 的HTTP方法過濾功能**

想利用 cors 的HTTP方法過濾功能，補正缺漏的程式

請閱讀教學系統的程式與說明，然後，到「// 作答區」填入答案。

答案有區分大寫小寫。

【Key Points】：

利用 cors 協助進行跨來源資源共用 ( CORS )

利用 cors 的白名單功能幫我們過濾對象

想利用 cors 的HTTP方法過濾功能

# 課後練習題(Lab)

- **情節描述:**  
寫作 Web API 服務並公開到網際網路後，用戶端很可能不會與我們的伺服器位於同一網域。基於安全理由，瀏覽器會限制跨域存取，以致於許多用戶端無法使用我們提供的Web API 服務。
- **預設目標:**
  - 完成Web API 服務。
  - 體驗果然無法跨域使用Web API 服務。
  - 實作 CORS 實現跨域呼叫 Web API。
- **Lab01: 體會SOP同源政策的限制**
- **Lab02: 實作 CORS**

Estimated time:

20 min.

18-19



## 【情節描述】

寫作 Web API 服務並公開到網際網路後，用戶端很可能不會與我們的伺服器位於同一網域。基於安全理由，瀏覽器會限制跨域存取，以致於許多用戶端無法使用我們提供的Web API 服務。

## 【預設目標】

- 完成Web API 服務。
- 體驗果然無法跨域使用Web API 服務。
- 實作 CORS 實現跨域呼叫 Web API。

Lab01: 體會SOP同源政策的限制

Lab02: 實作 CORS

完成後的程式與檔案，請參考 Example 資料夾的內容。

## 【Key Points】：

體會SOP

模擬DNS設定，營造兩個網域的環境

實作 CORS

# 範例程式使用說明

- 範例程式資料夾: Module\_nn\_example
- 使用步驟:
  1. 安裝 Node.js
  2. 安裝 Visual Studio Code
  3. 以 Visual Studio Code 開啟本模組的範例資料夾
  4. 編輯 C:\Windows\System32\drivers\etc\hosts，模擬兩個網域
  5. 在 Visual Studio Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗。
  6. 在終端機視窗，輸入下列指令，安裝必要的模組套件:  
npm install express cors
  7. 在終端機視窗，輸入 node index.js
  8. 啟動瀏覽器，連接 http://www.beauty.com/index.html

18-20



使用步驟:

1. 安裝 Node.js (<https://nodejs.org/en/>)
2. 安裝 Visual Studio Code (<https://code.visualstudio.com/>)
3. 以 Visual Studio Code 開啟範例資料夾  
在檔案總管，滑鼠右鍵點按「本範例資料夾」，從快捷功能表點選「以Code開啟」  
或者，啟動 Visual Studio Code 之後，功能表 File | Open Folder，選擇「本範例資料夾」
4. 以系統管理員身份啟動「記事本」，「記事本」功能表檔案 | 開啟舊檔 | 檔案名稱輸入:  
C:\Windows\System32\drivers\etc\hosts
5. 加入下列兩行並且存檔:  
127.0.0.1 www.beauty.com  
127.0.0.1 www.monster.com
6. 在 Visual Studio Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗。
7. 在終端機視窗，輸入下列指令，安裝必要的模組套件:  
npm install express cors
8. 在終端機視窗，輸入 node index.js
9. 啟動瀏覽器，連接 http://www.beauty.com/index.html

【Key Points】：

程式執行環境 Node.js

程式開發編輯環境: Visual Studio Code

在 Visual Studio Code 按下「Ctrl + 滑鼠」開啟 terminal 終端機視窗，輸入：「node 主程式.js」