Config 檔和 JSON 格式

喬逸偉 (Yiwei Chiao)

1 路由表 routing table

目前 index.js 利用 routingTable 結構來依據使用者要求，傳回不同的檔案內容到瀏覽器端。路由表 (routingTable) 內容類似下面的型態：

```
const routingTable = {
  '/': {
    url: '../htdocs/index.html',
    mime: 'text/html'
  },

  '/styles.css': {
    url: '../htdocs/assets/css/styles.css',
    mime: 'text/css'
  },

  '/breakit.js': {
    url: '../htdocs/js/index.js',
    mime: 'application/javascript'
  },
};
```

而相應的 HTML 內容片段如下：

```
    <link rel="stylesheet" href="styles.css">
    <script src="breakit.js"></script>
```

可以看到在 index.html 檔案內其實無法直接看出在伺服端的檔案對應，和資料夾安排方式。這樣的好處是，如果為了系統擴充式其它原因需要更改伺服端的檔案結構或資料提供方式，系統需要的只是去修改這個 routingTable 的內容就行了，其它的部份都不需要動。

但這還有一個缺點，目前這個 routingTable 結構是 index.js 的一部份；也就是屬於程式碼的一部份。可是它其實只是**資料** (*data*)。接下來，就是準備將 routingTable 抽離 index.js，讓它回復**資料**本色，安靜的待在資料檔案裡。

2 配置檔 (configuration file)

上面的問題就是所謂**配置檔** (configuration file) 的動機。

Configuration file 的概念出現之後，這個檔案有各式各樣的檔案格式 (file format) 曾經/目前被使用過。除了少數特殊考量之外，所有的格式都是人類可閱讀的簡單文字檔 (text file)，目的是為了讓使用者能在不需要大量技術支援的情況下，快速理解並調整系統配置。

這裡 BreakIt 專案將採用 JSON 格式作為它的系統配置檔。一個原因是是 JSON 格式簡單易學；另一個原因則是 JSON 目前已經是 Web 領域，客戶端 (browser) 和伺服端 (http server) 交換資料的主要格式。好 JSON，不學嗎？

2.1 JSON 格式

JSON 其實是 JavaScript Object Notation 的字首縮寫。如它名字所表示的 JSON 檔案的內容就是一個合法的 JavaScript 物件表示式 (object literal)。或者，由這裡也可以看出為什麼 JSON 格式會在 Web 領域裡受到廣泛的歡迎。

直接看 index.js 裡的 routingTable (對了，index.js 的 routingTable 寫法就是 JavaScript 的 Object literal) 寫成 JSON 格式的模樣：

```
{
  "/": {
    "url": "../htdocs/index.html",
    "type": "text/html"
  },
  "/styles.css": {
    "url": "../htdocs/assets/css/styles.css",
    "type": "text/css"
  },
  "/breakit.js": {
    "url": "../htdocs/js/index.js",
    "type": "application/javascript"
  }
}
```

可以和原來 index.js 裡的 routingTable 對照看。原則上 JSON 和 JavaScript 的 object literals 完全相同，都採用：

```
{
  key: value
}
```

的格式。但 JSON 的定義更嚴謹，而有幾個明顯/要注意的不同點：

- key: JavaScript 的 key 是簡單字串的話，可以不用加字串引號 (single/double quote) `"'"` 或 `'"'`；而如果要加 **引號** (quotation mark)，只要前後一致，單，雙引號都可以。JSON 的 key 則**一定**要用**雙引號** (double quotes)
- 多組 `key: value` 間的分隔逗點 ",": JSON 和 JavaScript 的 `object literals` 都採用**逗點** "," 來分隔不同的 `key: value`；要注意的是**最後一組** `key: value`。JavaScript 會**容忍**最後一組的 `key: value,` 後面的那個**逗號**；但 JSON **不會**。JSON 認為那是錯誤。
- {}: {} 用來標示物件的開始和結束，在 JavaScript 裡，很自然的 { 前面可能有 =，(之類的符號，而 } 後面可能也有) 或 ; 跟著；但 JSON 裡，這些都是**不允許**的。因為沒必要。
- 註解 (comments)：JSON 格式**不允許**註解，**不允許**註解，**不允許**註解。很重要，所以說三次。這是 JSON 格式最具爭議的一個設計決定。但 JSON 設計者堅持 JSON 格式簡單到不需要註解；更不需要註解來污染這格式的簡單純粹。真需要註解，有其它格式可選，結案。

除去這些更嚴謹的設定不同，JSON 格式的文件就是 JavaScript 的 object literal；換句話說，就是個合法的 JavaScript 程式檔案，只是副檔名 (延伸檔名) 使用 '.json' 或 '.js' 的不同而已。因此，格式在 2007 (或之前) 提出後很快的就被 Web 開發社群接受。

2.2 config.json

理解 JSON 格式的意義，就可以將 routingTable 的定義移到 config.json 檔案內。利用文字編輯器 (text editor) 建立 config.json 檔案，內容如前一小節所示，就是原來 index.js 內 routingTable 的內容。

將 config.json 和 index.js 放在同一個資料夾。然後修改 index.js，將原來 routintTable 的內容移除，更改成：

```
const routingTable = require('./config.json');
```

是的，就這樣。因為如同前面對 JSON 格式的介紹，JSON 檔案本身就是個合法的 JavaScript 檔案，它的內容其實可以視作是一個 **匿名** (anonymous) 的物件宣告。所以可以直接當作 Node.js 的模組載入使用。

3 專案製作

目前為止，我們已經完成了一個簡單的 Web Server 架構，知道如何建立 Web 服務的路由表，如何寫作 JSON 檔案，利用 JSON 檔案儲存資料；在客戶端，學到了如何操作 DOM 模型，如何用 canvas 作畫，製作動畫，如何利用事件取得使用者輸入，追蹤滑鼠，等等。回首遙望，在 Web 的學習上，我們已經走了很長的一段路。

只是 Web 的世界如海廣闊，開發者們還在不斷的探索新的可能，提出新的技術，前面等待著的，是更多，更長的旅程。

「千里之行，始於足下。」無論如何，那條旅程都要自己去走，剩下的就是大家自己努力了。

- 請自己找好同伴同行：1 或 2 人一組。
- 請自行挑選一個題目：
  - 黑白棋 (Reversi)
  - 踩地雷 (MineSweeper)
  - 貪食蛇 (Snake)
  - 俄羅斯方塊 (Tetris)
  - 其它 (需事前提出討論，同意後進行)
- 學期第 17 週完成。