



執行第一個JavaScript檔



designed by freepik

Estimated time:

50 min.

資訊工業策進會 Institute for Information Industry

【Key Points】：

學習目標

- 3-1: 撰寫 JavaScript 檔
- 3-2: 執行 JavaScript 檔
- 3-3: 查看儲存訊息



3-1

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

我們即將一起學習：

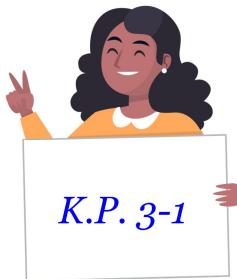
1. 在3-1會撰寫一份簡易的JavaScript檔
2. 在3-2會配合node.js執行JavaScript檔
3. 在3-3會教執行後查看儲存訊息
4. 後面會有Lab步驟，讓各位依照步驟練習
5. 最後將有個精華總複習

【Key Points】：

- 3-1: 撰寫 JavaScript 檔
- 3-2: 執行 JavaScript 檔
- 3-3: 查看儲存訊息

3-1: 撰寫 JavaScript 檔

- 安裝 VS Code
- 動手寫看看 — 開新專案
- 動手寫看看 — 專案初始化
- 動手寫看看 — app.js



designed by freepik



designed by freepik

3-2

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

接下來，我們要一起學習：

- 安裝 VS Code
- 新增檔案
- 動手寫看看 — 開新專案
- 動手寫看看 — 專案初始化
- 動手寫看看 — app.js

【Key Points】：

安裝 VS Code

新增檔案

專案初始化

安裝 VS Code 與新增檔案

- 本課程建議使用 VS Code，安裝程式下載網址：
<https://code.visualstudio.com/>
 - 安裝 VS Code 時，建議勾選下列選項：
將「以Code開啟」動作加入 Windows 檔案總管目錄的操作功能表中
 - 1. 在檔案總管，以滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟
 - 2. 在 VS Code 左側的 Explorer 窗格，
滑鼠右鍵 | New File | 輸入檔名，例如: hello.js
 - 3. 編寫 JavaScript 程式（如下），並且存檔（組合鍵 Ctrl + S）
`console.log("Hello!");`
- (或者，也可參考下一張投影片的作法...)

3-3



你可以使用任何文字編輯器編寫 JavaScript 程式。

本課程建議使用 VS Code，安裝程式下載網址：

<https://code.visualstudio.com/>

安裝 VS Code 時，建議勾選下列選項：

- 將「以Code開啟」動作加入 Windows 檔案總管檔案的操作功能表中
- 將「以Code開啟」動作加入 Windows 檔案總管目錄的操作功能表中
- 加入 PATH 中

撰寫 JavaScript 檔：

1. 在檔案總管，滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟
2. 在 VS Code 左側的 Explorer 窗格，滑鼠右鍵 | New File，檔名: Hello.js
3. 編寫 JavaScript 程式（如下），並且存檔（組合鍵 Ctrl + S）
`console.log("Hello!");`

<Note> 以本例來說，JavaScript 程式存檔於 c:\lab\hello.js

【Key Points】：

本課程建議使用 VS Code

安裝 VS Code 時，建議勾選「以Code開啟」選項

請示範一下建立檔案的過程

動手寫看看 — 開新專案

- 啟動「命名提示字元」
- 輸入：**mkdir myapi** 建立名為 myapi 的資料夾作為專案名稱。
- 輸入：**cd myapi** 切換到該目錄

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.17134.1304]
(c) 2018 Microsoft Corporation. 著作權所有，並保留一切權利。
Active code page: 65001

C:\Work>mkdir myapi
C:\Work>cd myapi
C:\Work\myapi>
```

3-4



讓我們從頭開始創建一個 Node 專案。

在 c:\Work (例如) 建立一個叫做 myapi 的資料夾作為專案名稱 (也可改為你想要的名字) 。

1. 按下組合鍵「Windows + R」
2. 輸入「cmd」後，點按「確定」按鈕。 (啟動「命令提示字元」)
3. 上述步驟1+2, 也可以: 按下組合鍵「Windows + S」，然後鍵盤輸入「cmd」找到「命令提示字元」。
4. 輸入：**mkdir myapi** 建立名為 myapi 的資料夾作為專案名稱。
5. 輸入：**cd myapi** 切換到該目錄

常用cmd指令整理：

查詢目錄 (dir)、建立目錄 (md, mkdir)、變更目錄 (cd, chdir)、刪除目錄 (rd, rmdir)、檔案重新命名 (ren, rename)

【Key Points】：

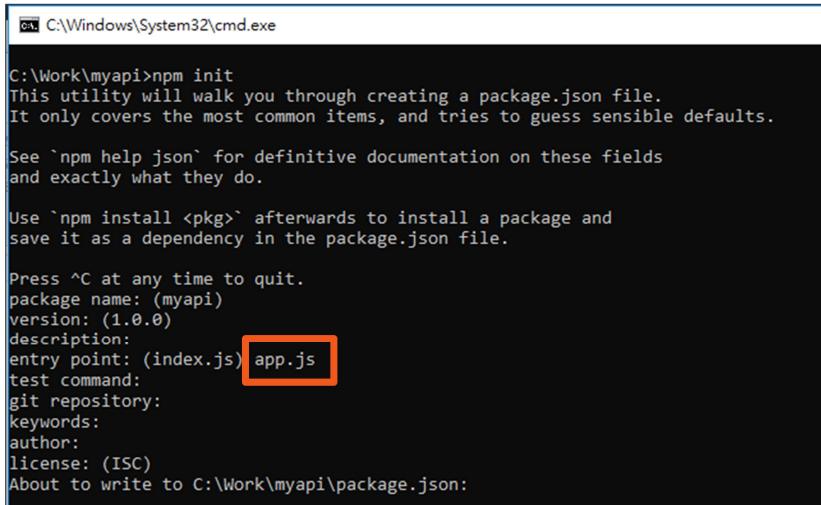
查詢目錄 (dir)

建立目錄 (md, mkdir)

變更目錄 (cd, chdir)

動手寫看看 — 專案初始化

- 輸入：**npm init** 初始化一個專案，這裡我們在 **entry point** 選項時把入口文件改成 **app.js**，其他就維持預設，最後記得輸入 **yes**。



```
C:\Windows\System32\cmd.exe
C:\Work\myapi>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (myapi)
version: (1.0.0)
description:
entry point: (index.js) app.js
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to C:\Work\myapi\package.json:
```

3-5

1. 建立完後在資料夾中運行 **npm init** 將該資料夾轉成一個 node 專案。
2. 若無特殊需求，可一路按確認鍵到底。
3. 中途我們在 **entry point** 選項時需要把入口文件改成 **app.js**（或其他名稱，例如：**index.js**）
4. 在 **package.json** 這個檔案中，使用者可以定義應用名稱 (**name**)、應用描述 (**description**)、關鍵字 (**keywords**)、版本號 (**version**)、應用配置 (**config**)、主頁 (**homepage**)、作者(**author**)、版本庫 (**repository**)、bug的提交地址 (**bugs**)、授權方式(**licenses**)... 等。
5. 如果在有 **package.json** 的專案目錄下，執行 **npm install**，**npm** 便會依照 **package.json** 的內容去下載套件並且佈置好執行環境。

【Key Points】：

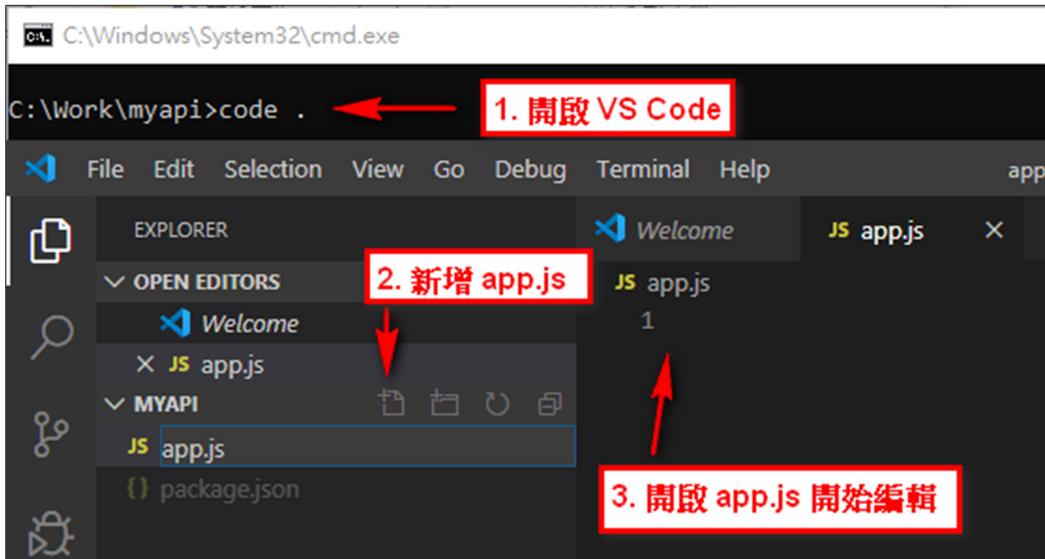
npm init

entry point 選項時需要把入口文件改成 **app.js**

老師可以帶著學生看一下 **package.json** 這個檔案，說明每個項目可能需填的內容

動手寫看看 — app.js

- 輸入：`code .` 在當下目錄開啟 VS Code，並新增 app.js 文件



3-6

- 在當前目錄下執行 `code .`，即可執行 VS Code 並且編輯當前的工作目錄
- 在 VS Code 裡面操作新增 app.js 文件
- 打開該文件編輯程式碼

<Note>也可以在檔案總管，滑鼠右鍵點按資料夾，從快捷功能表選擇「以Code開啟」

<Note>建議以Code開啟資料夾而非單一檔案，因為，專案通常由一組相關的檔案構成。

【Key Points】：

執行 `code .`，即可執行 VS Code 並且編輯當前的工作目錄
也可以在檔案總管，滑鼠右鍵點按資料夾，從快捷功能表選擇「以Code開啟」
前面章節已經有請同學們安裝過 VS Code 了，如果有還沒安裝的在請老師協助安裝

3-2: 執行 JavaScript 檔

- 撰寫 JavaScript 檔
- 執行 JavaScript 檔
- 進入Node模式
- 如何不透過 JavaScript 檔案來執行程式



designed by freepik



designed by freepik

3-7

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

Node.js 安裝好了，如何執行 JavaScript 程式呢？

我們會學到如何進入Node模式，在Node 模式採「一次一行」、「一問一答」的方式執行程式。

接下來，會學習如何在「命名提示字元」，輸入「node 檔名」指令來執行 JavaScript 程式

此外，本課程建議安裝 VS Code，以它為整合開發環境。我們會學到如何在 VS Code 啟動終端機視窗，以「node 檔名」執行 JavaScript 程式。

最後，會介紹一種直接執行 JS 程式的作法：

```
node -e "console.log('Hello World')"
```

【Key Points】：

如何進入Node模式

在「命名提示字元」，輸入「node 檔名」指令來執行 JavaScript 程式

```
node -e "console.log('Hello World')"
```

撰寫 JavaScript 檔

1. 在檔案總管，以滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟
2. 在 VS Code 左側的 Explorer 窗格，
滑鼠右鍵 | New File | 輸入檔名，例如: hello.js
3. 編寫 JavaScript 程式（如下），並且存檔（組合鍵 Ctrl + S）
`console.log("Hello!");`

3-8



將資料夾轉成 Node.js 專案，並不是必要程序。如果覺得前兩張投影片比較難，沒關係，本課程其他模組會再說明。暫時，我們先專注於一個程式檔案就好：

撰寫 JavaScript 檔：

1. 在檔案總管，滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟
2. 在 VS Code 左側的 Explorer 窗格，滑鼠右鍵 | New File，檔名: Hello.js
3. 編寫 JavaScript 程式（如下），並且存檔（組合鍵 Ctrl + S）
`console.log("Hello!");`

<Note> 以本例來說，JavaScript 程式存檔於 c:\lab\hello.js

【Key Points】：

本課程建議使用 VS Code

請操作一下建立檔案的過程

`console.log("Hello!");` 會在端端機視窗輸出: Hello! 字樣

執行 JavaScript 檔

1. 開啟「命名提示字元」

2. 輸入: node 檔名 (含路徑) , 例如:

node c:\lab\hello.js

• 承接上一張投影片，透過 VS Code 編輯、執行 JavaScript 程式：

1. 在檔案總管，以滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟

2. 在 VS Code , 按下「 Ctrl + 撇號 」 (撇號在鍵盤的左上角)

3. 輸入 node hello.js

3-9



1. 開啟「命名提示字元」

2. 輸入: node 檔名 (含路徑) , 例如:

node c:\lab\hello.js

<Note>承接上一張投影片，如果你已將 JavaScript 程式存檔於 c:\lab\hello.js

1. 在檔案總管，以滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟

2. 在 VS Code , 按下「 Ctrl + 撇號 」 (撇號在鍵盤的左上角) 啟動終端機視窗

3. 輸入 node hello.js

<Note> 建議安裝 VS Code , 以它為整合開發環境。

【Key Points】：

在命名提示字元，輸入: node 檔名 (含路徑)

在 VS Code , 按下「 Ctrl + 撇號 」啟動終端機視窗

建議安裝 VS Code , 以它為整合開發環境

進入Node模式

- Node 模式採「一次一行」、「一問一答」的方式執行程式。
- 在「命令提示字元」輸入：
node <Enter鍵>
即可進入Node模式。
- 輸入你想執行的 JavaScript 程式，例如：
`console.log("Hello");`
- 輸入：**.exit** 或組合鍵「**Ctrl + D**」可結束 Node 模式。
- 輸入：**.help** 可查看 Node 模式相關指令。

3-10



Node 模式採「一次一行」、「一問一答」的方式執行程式。

在「命令提示字元」輸入下列指令，可進入 Node 模式

`node <Enter鍵>`

輸入你想執行的 JavaScript 程式，例如：

`console.log("Hello");`

<Note>因為 `console.log()` 方法並沒有傳回值。程式執行時，除了顯示 Hello，還會在第二列顯示 undefined 字樣。

輸入 `.exit` 或組合鍵「**Ctrl + D**」可結束 Node 模式。

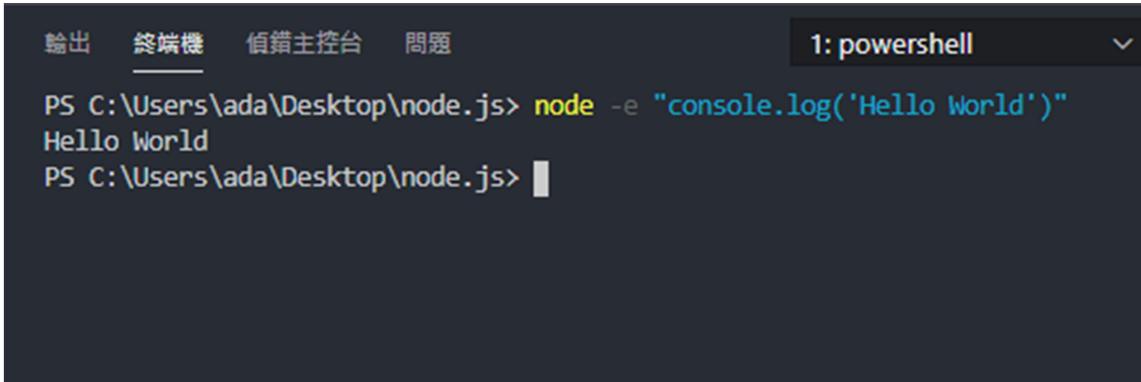
輸入 `.help` 可查看 Node 模式相關指令。

【Key Points】：

何謂 Node 模式採
如何進入 Node 模式
如何結束 Node 模式

如何不透過 JavaScript 檔案來執行程式？

- 在 VS Code 按下「 Ctrl + 撇號 」開啟終端機，輸入指令
 - node -e "console.log('Hello World')"
 - 使用上述指令，將可以達到如同使用JavaScript檔案結果



A screenshot of a terminal window in VS Code. The window has tabs at the top: '輸出' (Output), '終端機' (Terminal) which is selected, '偵錯主控台' (Debug Console), and '問題' (Problems). The status bar shows '1: powershell'. The terminal output shows the command 'node -e "console.log('Hello World')"' being run, followed by the output 'Hello World'.

```
輸出 終端機 偵錯主控台 問題 1: powershell
PS C:\Users\ada\Desktop\node.js> node -e "console.log('Hello World')"
Hello World
PS C:\Users\ada\Desktop\node.js>
```

3-11



如何不透過JavaScript檔案執行程式？

先在 VS Code，按下「 Ctrl + 撇號 」（撇號在鍵盤的左上角）啟動終端機視窗

接下來，輸入下列程式：

```
node -e "console.log('Hello World')"
```

請注意，字串若以單引號開始，就以單引號結尾；雙引號開始，雙引號結束，不可交錯使用。

因此，如果引號的內容有引號的話，內外層應該分別使用對應的引號。以下列指令來說，外層用一對雙引號，內層便改用一對單引號：

```
node -e "console.log('Hello World')"
```

【Key Points】：

在 VS Code，按下「 Ctrl + 撇號 」啟動終端機視窗

協助學員了解基本 node -e 指令使用方式

單雙引號要正確配對

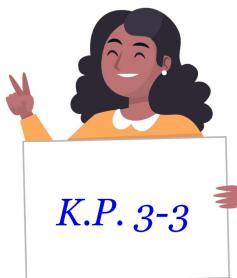
3-3: 查看儲存訊息

- 執行JavaScript檔案發生錯誤時會看到什麼？

- JavaScript 語法錯誤

- require 不到檔案時

- 未安裝express
 - 模組路徑錯誤



designed by freepik



designed by freepik

3-12

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

接下來，我們一起來研究看看為什麼程式無法執行：

- 為什麼會有錯誤？
- 是什麼樣的錯誤？
- 如何從錯誤訊息中判斷是何種錯誤呢？
- 初學者常見錯誤一：語法錯誤
- 初學者常見錯誤二：require之前沒有先將模組套件安裝妥當

【Key Points】：

為什麼會有錯誤

從錯誤訊息中判斷是何種錯誤

初學者常見錯誤

有個JavaScript檔案語法錯誤，以node執行

- 開啟終端機，進入Node模式：

- 輸入錯誤的JavaScript語法
- 執行後的結果與警告，給予開發者警告訊息
- 修正為正確寫法再次執行



A screenshot of a Windows PowerShell window. The title bar says "Windows PowerShell". The content shows the following text:
請嘗試新的跨平台 PowerShell <https://aka.ms/powershell>
PS C:\Users\ada\Desktop> node
Welcome to Node.js v12.16.0.
Type ".help" for more information.
> consle.log('123');
Uncaught ReferenceError: consle is not defined
> |

3-13



開啟終端機並輸入 node (進入Node模式)，進行一些實驗：

- 嘗試輸入錯誤程式，例如：方法名稱打錯
- 以console.log("123");為例，誤打成consle.log("123");
- 執行後的結果與警告，給予開發者警告訊息
- 修正為正確寫法，再次執行
- 會得到結果為123

【Key Points】：

嘗試輸入錯誤程式

開啟終端機並輸入 node (進入Node模式)

修正為正確寫法，再次執行

有個檔案的程式 require("") 不到模組時

1. 新增一個檔案 index.js
2. 程式: require("express")
3. 以 node index.js 執行
4. 看到以下錯誤

Cannot find module '....'

原因: 尚未安裝express

```
JS index.js > ...
1 const express = require('express');
2
3 console.log('express :',express);

輸出 終端機 伺服器主控台 問題

請嘗試新的跨平台 PowerShell https://aka.ms/pscore6

PS C:\Users\ada\Desktop\node.js> node index.js
internal/modules/cjs/loader.js:984
    throw err;
^

Error: Cannot find module 'express'
Require stack:
- C:\Users\ada\Desktop\node.js\index.js
  at Function.Module._resolveFilename (internal/modules/cjs/loader.js:981:15)
  at Function.Module._load (internal/modules/cjs/loader.js:863:27)
  at Module.require (internal/modules/cjs/loader.js:1043:19)
  at require (internal/modules/cjs/helpers.js:77:18)
  at Object.<anonymous> (C:\Users\ada\Desktop\node.js\index.js:1:17)
  at Module._compile (internal/modules/cjs/loader.js:1157:30)
  at Object.Module._extensions..js (internal/modules/cjs/loader.js:1177:10)
  at Module.load (internal/modules/cjs/loader.js:1001:32)
  at Function.Module._load (internal/modules/cjs/loader.js:900:14)
  at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:74:12) {
  code: 'MODULE_NOT_FOUND',
  requireStack: [ 'C:\Users\ada\Desktop\node.js\index.js' ]
}
PS C:\Users\ada\Desktop\node.js> |
```

3-14



1. 建立一個index.js的檔案
2. 輸入以下內容

```
const express = require('express');
console.log('express :',express);
```
3. 開啟終端機
4. 輸入以下指令

```
node index.js
```
5. 看到錯誤訊息
Error: Cannot find module 'express'
6. 此為找不到express模組
7. 輸入 `npm install express` 安裝 express 後，再次執行 (Note: 關於 `npm install`，本課程後續的模組會教到)

【Key Points】：

請解釋require用法

請解釋express模組

請示範安裝express後，再執行一次

有個檔案的程式 require("") 路徑錯誤時

1. 建立一個sum.js檔案

- 與 index.js 同目錄下
- 建立 sum 模組

```
JS sum.js > ...
1 module.exports = (a, b) => {
2   return a + b;
3 };
4

JS index.js > ...
1 const sum = require('sum');
2 console.log('sum : ', sum(1,2));

輸出 總編譯 值鎖主控台 問題
PS C:\Users\ada\Desktop\node.js> node index.js
internal/modules/cjs/loader.js:984
throw err;
^

Error: Cannot find module 'sum'
Require stack:
- C:\Users\ada\Desktop\node.js\index.js
  at Function.Module._resolveFilename (internal/modules/cjs/loader.js:981:15)
  at Function.Module._load (internal/modules/cjs/loader.js:863:27)
  at Module.require (internal/modules/cjs/loader.js:1042:10)
  at require (internal/modules/cjs/helpers.js:77:18)
  at Object. (C:\Users\ada\Desktop\node.js\index.js:1:13)
  at Module._compile (internal/modules/cjs/loader.js:1177:30)
  at Object.Module._extensions..js (internal/modules/cjs/loader.js:1177:10)
  at Module.load (internal/modules/cjs/loader.js:1061:32)
  at Function.Module._load (internal/modules/cjs/loader.js:996:14)
  at Function.executeUserEntryPoint [as runMain] (internal/modules/run_main.js:74:12) {
    code: 'MODULE_NOT_FOUND',
    requireStack: [ 'C:\Users\ada\Desktop\node.js\\index.js' ]
  }
PS C:\Users\ada\Desktop\node.js> 
```

3-15



1. 建立一個index.js的檔案

2. 輸入以下內容

```
const sum = require("sum");
console.log("sum : ",sum(1,2));
```

3. 開啟終端機

4. 輸入以下指令

```
node index.js
```

5. 看到錯誤訊息

Error: Cannot find module 'express'

6. 此為找不到sum模組

7. 請修正路徑後後再次執行

8. 將sum.js改成以下內容

```
const sum = require('./sum');
console.log('sum : ',sum(1,2));
```

9. 再次以終端機執行node index.js

10. 將得到結果為3

```
JS index.js > ...
JS index.js > ...
1 const sum = require('./sum');
2
3 console.log('sum : ',sum(1,2));

輸出 總編譯 值鎖主控台 問題
PS C:\Users\ada\Desktop\node.js> node index.js
sum : 3
```

【Key Points】：

1. 請解釋路徑如何撰寫
2. 向學生提問不同路徑應該如何寫
3. 示範如何更正錯誤，再執行一次

錯誤與例外處理

- 程式開發者錯誤(Programmer errors)
- 運算錯誤(Operational errors)，也稱為執行時期錯誤
- 以 `try ... catch` 語法，可有效攔截執行時期錯誤
- `Error`物件是JavaScript內建，用於記錄錯誤資訊的物件

3-16



- 程式開發者錯誤(Programmer errors):
程式設定師在寫程式時，不小心造成的錯誤，例如：嘗試讀取某個"undefined"的屬性，將文字資料指定給某個應該是陣列的變數
- 運算錯誤(Operational errors)，也稱為執行時期錯誤：
程式本身並沒有臭蟲(bugs)，在執行時與外部環境互動產生的異常情況，例如；網路連線失敗、檔案讀取失敗或是記憶體不足等等。
- 以 `try ... catch` 語法，可有效攔截執行時期錯誤
- `Error`物件是JavaScript內建，用於記錄錯誤資訊的物件
- 舉例來說：

```
try {  
    throw new Error('Whoops!');  
} catch (error) {  
    console.log(error.name + ': ' + error.message);  
}
```

【Key Points】：

運算錯誤(Operational errors)，也稱為執行時期錯誤

`try...catch`

`Error`物件是JavaScript內建，用於記錄錯誤資訊的物件

Summary 〈精華回顧〉

- 安裝 VS Code
- 撰寫 JavaScript 檔
- 執行 JavaScript 檔
- 初學者常見錯誤
- 從錯誤訊息中判斷是何種錯誤



3-17

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

重點精華回顧:

- 安裝 VS Code
- 撰寫 JavaScript 檔:
 1. 在檔案總管，滑鼠右鍵點按例如 c:\lab 資料夾 | 以 Code 開啟
 2. 在 VS Code 左側的 Explorer 窗格，滑鼠右鍵 | New File，檔名: Hello.js
 3. 編寫 JavaScript 程式（如下），並且存檔（組合鍵 Ctrl + S）

```
console.log("Hello!");
```
- 執行 JavaScript 檔
 1. 在 VS Code，按下「Ctrl + 滑鼠」（滑鼠在鍵盤的左上角）啟動終端機視窗
 2. 輸入 node hello.js
- 初學者常見錯誤
- 從錯誤訊息中判斷是何種錯誤

【Key Points】：

撰寫 JavaScript 檔

執行 JavaScript 檔

從錯誤訊息中判斷是何種錯誤

線上程式題

- **3-1 建立 HTTP Server**

某位程式設計師想用 Node.js 內建的 HTTP 模組，建立一套簡易的 Web 伺服器。請問下列程式漏寫了什麼？

- **3-2 輸出圖片**

如果 HTTP Request 的結果是要傳一張 gif 圖檔給用戶端，請問如何指定 HTTP Header 的內容？

- **3-3 依據網址進行路由分流**

http://localhost/user1

http://localhost/user2

想分別看到不同的執行結果，請問程式該怎麼寫？

3-18



題目名稱: 3-1 建立 HTTP Server

某位程式設計師想用 Node.js 內建的 HTTP 模組，建立一套簡易的 Web 伺服器。請問下列程式漏寫了什麼？

題目名稱: 3-2 輸出圖片

如果 HTTP Request 的結果是要傳一張 gif 圖檔給用戶端，請問如何指定 HTTP Header 的內容？

題目名稱: 3-3 依據網址進行路由分流

http://localhost/user1

http://localhost/user2

想分別看到不同的執行結果，請問程式該怎麼寫？

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】：

3-1 建立 HTTP Server

3-2 輸出圖片

3-3 依據網址進行路由分流

課後練習 (Lab)

- 打開 vscode 並新增一個檔案，檔案名稱叫做 **helloWorld.js**
- 程式打開 **helloWorld.js** 之後，輸入程式碼
 - `console.log('Hello World');`
- 在終端機上輸入指令並指名要執行的 JavaScript 檔案
 - `$ node helloWorld.js`
 - 此時終端機就會顯示 Hello World
- 如何不透過新增 JavaScript 檔案來執行程式？

3-19



3. 如何不透過新增 JavaScript 檔案來執行程式？

3-1.vscode 打開終端機

3-2.輸入以下指令

```
$ node -e "console.log('Hello World')"
```

使用上述指令，將可以達到如同步驟二的結果

3-3.亦或者先執行以下指令

```
$ node
```

啟動執行腳本的環境，接著輸入想執行的程式碼

```
> console.log('Hello World')
```

將會在終端機上顯示 Hello World

3-4.並非只能單純透過 `console.log()` 輸出所輸入的值，也可透過執行 `function` 並回傳值

```
> function sum(a, b) { return a + b }
```

可以看到上述程式碼是一個相加並將結果回傳的函式，接著執行這個函式

```
> sum(1, 2)
```

將可以得到 3 這個值

【Key Points】：

新增一個檔案，檔案名稱叫做 **helloWorld.js**

在 VS Code，按下「Ctrl + 撇號」（撇號在鍵盤的左上角）啟動終端機視窗

輸入指令: `node helloWorld.js`

範例程式使用說明

- 範例程式資料夾: Module_03_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. 在 Visual Studio Code 按下「Ctrl + 滑鼠右鍵」開啟 terminal 終端機視窗。
 5. 在終端機視窗，輸入 node helloWorld.js

3-20



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. 在 Visual Studio Code 按下「Ctrl + 滑鼠右鍵」開啟 terminal 終端機視窗。
5. 在終端機視窗，輸入 node helloWorld.js

【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 + 滑鼠右鍵」開啟 terminal 終端機視窗，輸入：「node 主程式.js」