Node JS Basic
Developer workshop

- i4.0 IT Team
- 2020/9/14

Agenda

- 前言
- Node Introduction
- Lab 1 – Hello World
- Coding Guidelines
- Asynchronous、Synchronous
- Call back
- Promise
- Lab 2 - Promise
- Lab 3 - Sync / Await

前言

為什麼採用node？

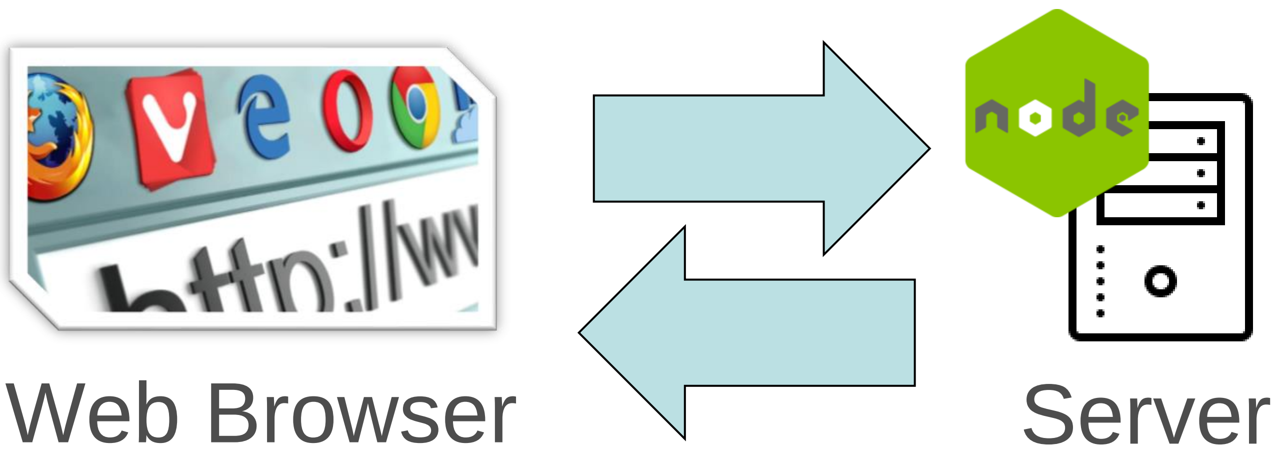
- 門檻低、效能高、擴充多
- 以前端技術，征服各種領域：
- 包括：
  - Server Side
  - Window Form
  - Mobile
  - 3D VR \ AR

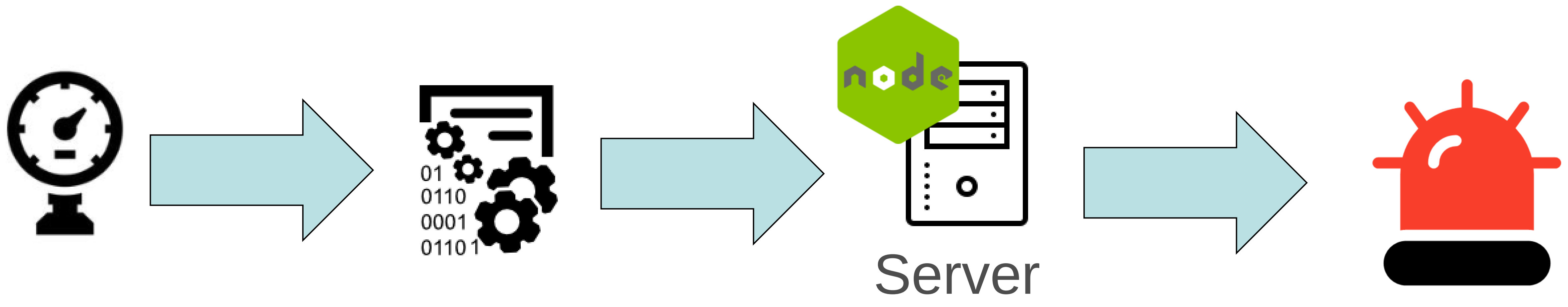
3d web powered by NodeJS & AngularJS

為什麼不採用node？

- 過於靈活，不好維護，容易產出糟糕的代碼
- 異步操作困擾又麻煩
- 不適合CPU繁重的工作
- 容錯率低

nodeJS在Wistron最常扮演什麼角色？





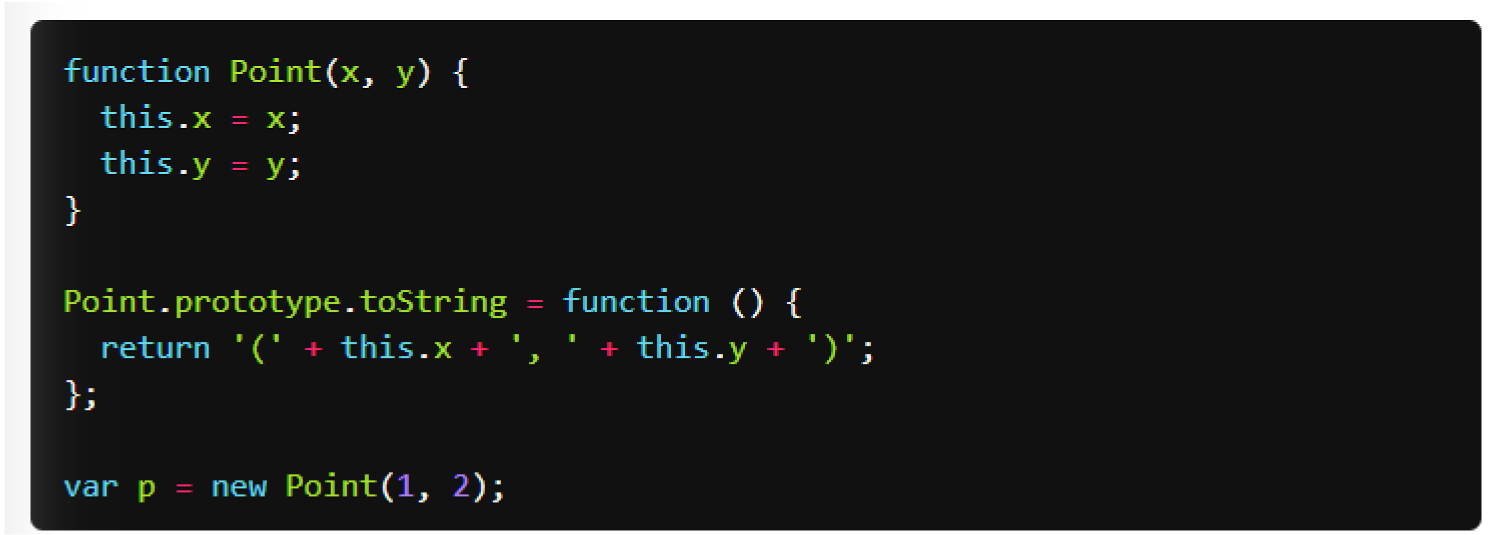
Node 介紹

了解nodeJS之前，什麼是JavaScript？

- 最早是瀏覽器腳本
- 為prototype-base-oriented的語言

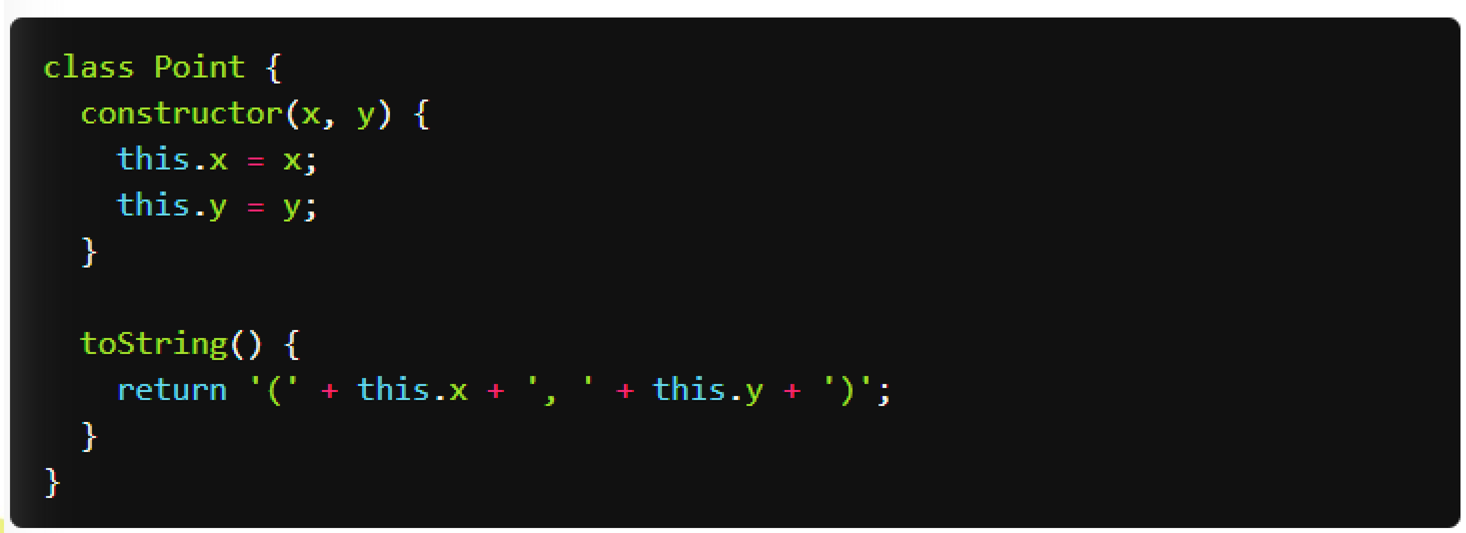
  ➔沒有Class (!?)，只有Object、Instance
- ECMAScript5、6、7、8、9、10

ES5之前



```
function Point(x, y) {
  this.x = x;
  this.y = y;
}

Point.prototype.toString = function () {
  return '(' + this.x + ', ' + this.y + ')';
};

var p = new Point(1, 2);
```

ES6之後



```
class Point {
  constructor(x, y) {
    this.x = x;
    this.y = y;
  }

  toString() {
    return '(' + this.x + ', ' + this.y + ')';
  }
}
```

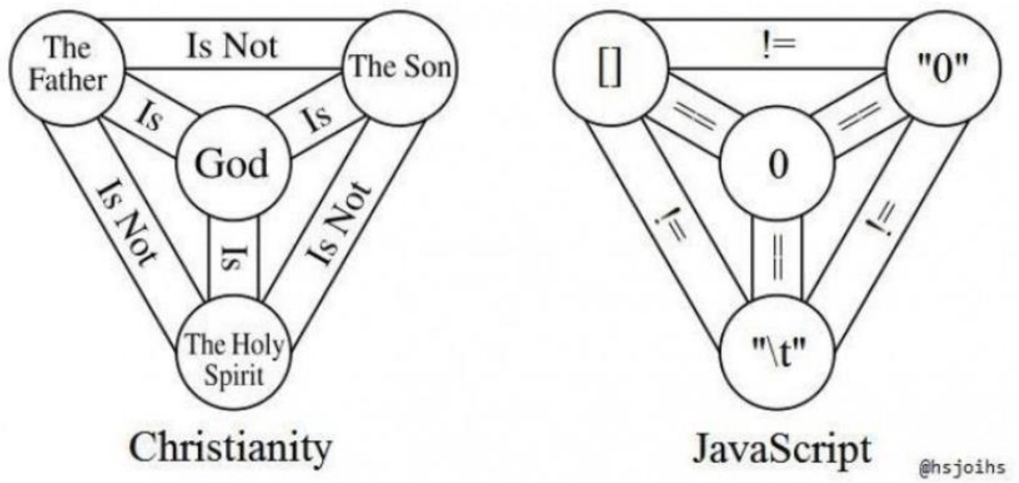
開發常常被忽略的問題

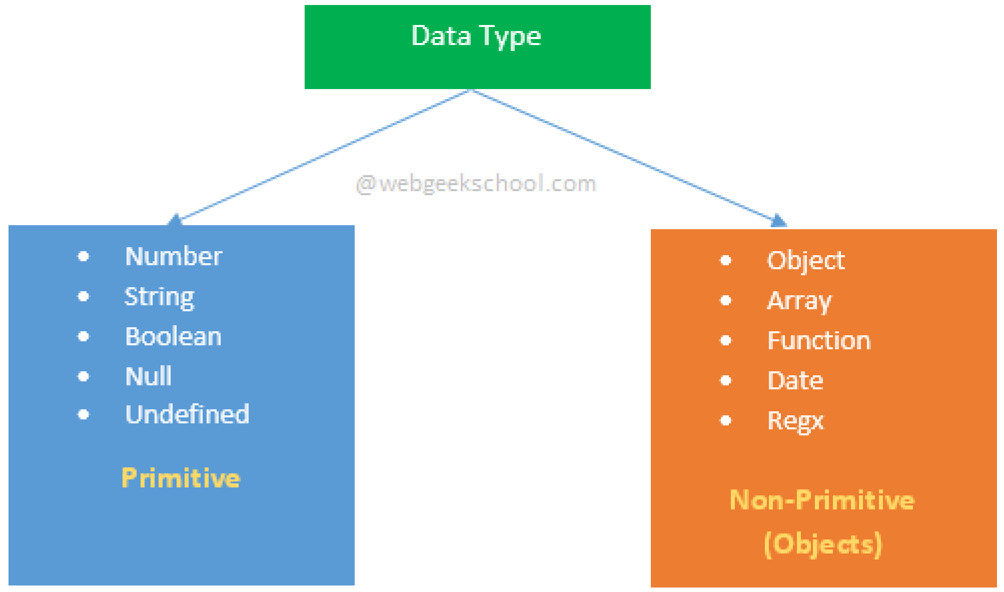
1. JS的弱型別

2. JS Object的Shallow Copy & Deep Copy

3.錯綜複雜的異步處理

4. 不好懂的this





JavaScript Data Type

What’s NPM？

Node 專用的套件管理器

Node 專案建立 & 執行

Init Project

1. npm init
2. Create application main entry : index.js

Install plugin

1. npm i(install) $required_package : ex: fs

Run your code

node . OR   node index.js

專案檔案說明

[node_modules]

package.json

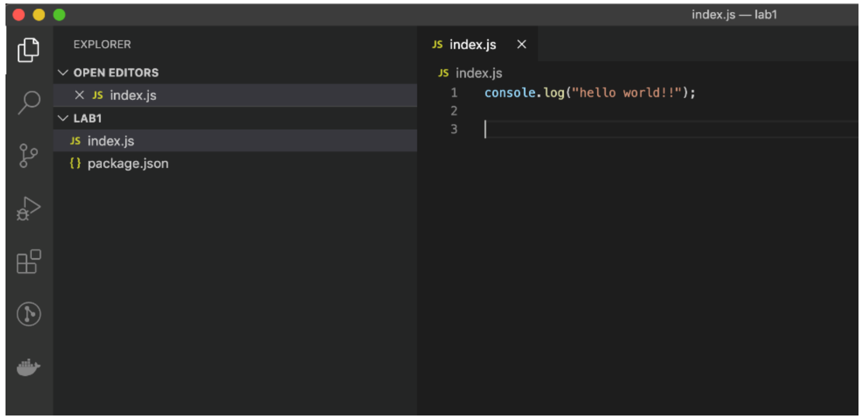
package-lock.json

index.js

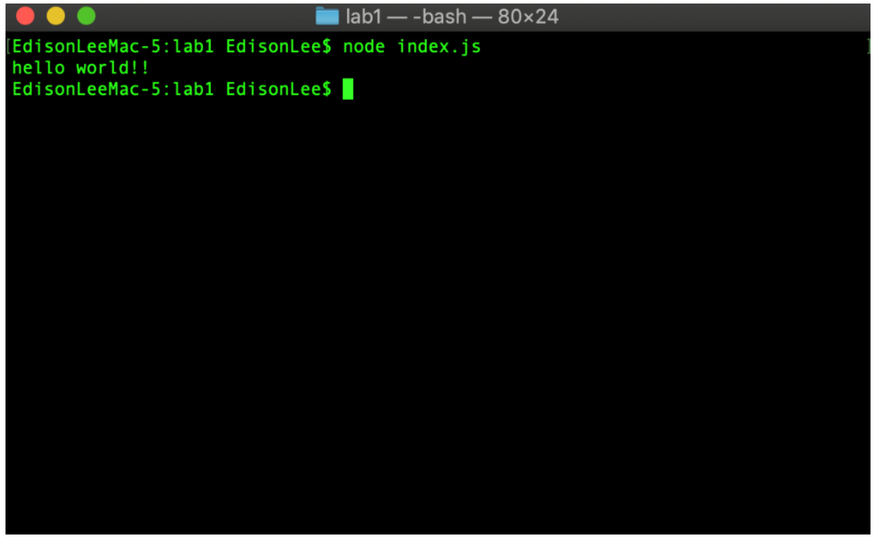
Lab 1
Hello World

在console中打印 “Hello World”

1. 使用 NPM 建立專案
2. 建立檔案 index.js
3. 在檔案中打入 console.log("hello world!!");
4. 運行程式碼 : node index.js



```
console.log("hello world!!");
```



```
EdisonLeeMac-5:lab1 EdisonLee$ node index.js
hello world!!
EdisonLeeMac-5:lab1 EdisonLee$ 
```

Coding Guildelines

Shared understanding of code quality

- **YAGNI – You Ain’t Gonna Need It!** 我們不應該為程式碼加入尚未用到的功能。
- **DRY – Don’t Repeat Yourself!** 不要使用複製貼上或者重複的邏輯、變數或功能。
- **SOLID** 物件導向設計原則 (後續說明)
- **Self-Documenting Code** Code即是文件 (後續說明)

SOLID Principles

Single Responsibility　單一職責原則

一種類別的修改應該只對應到一個理由。

Open/Closed Principle　開放封閉原則

當需求有異動時，要在不變動現在正常運行的程式碼，藉由繼承、相依性注入等方式，以實作新的需求。

Liskov Substitution Principle　里氏替換原則

物件間的可替換關係,增加系統的彈性

Interface Segregation Principle　介面分割原則

不該去依賴並不會使用到的東西

Dependency Inversion Principle　依賴反轉原則

模組與模組間，不該相互依賴，應透過注入方式來賦予關聯

SOURCE: McKinsey DnA Workshop - DevOps

Self-Documenting Code

- Commenting on why code does what is does can be incredibly helpful to developers
- Commenting on what or how the code does what it does is possibly a failure to express yourself in code

註解 why > what and how

- You don’t get extra credit for building complex solutions to simple problems

不要炫技，接手的人會抓狂

“ Code as if the next guy to maintain your code is a homicidal maniac who knows where you live. ”

- Kathy Sierra and Bert Bates

SOURCE: McKinsey DnA Workshop - DevOps

Not so Self-Documenting Code

Example 1

```
for(i=0;i<100;)console.log((++i%3?'':'Fizz')+(i%5?'':'Buzz')||i)
```

Example 2

```
console.log(
                    Array.apply(null, {length: 100}).map(function(val, index) {
                                        return (++index%3?'':'Fizz')+  (index%5?'':'Buzz')||index;}).join('\n')
);
```

Example 3

```
var i, values = [, , 'fizz', , 'buzz', 'fizz', , , 'fizz', 'buzz', ,  'fizz', , , 'fizzbuzz'];
for (i = 0; i < 100; console.log(values[i++ % 15] ||  i));
```

SOURCE: McKinsey DnA Workshop - DevOps

Self-Documenting Code

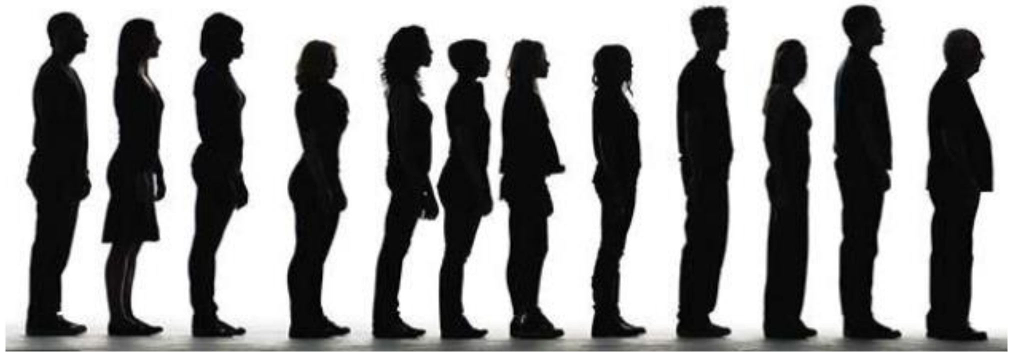
```
function fizzBuzz (num){
        val = “”;
        if (num % 3 === 0 && num % 5 === 0){  val = “Fizz Buzz”;
        }
        else if (num % 3 === 0){
                val = “Fizz”;
        }
        else if (num % 5 === 0){
                val = “Buzz”;
        }
        else {
                val = num.toString();
        }

        return val;
}

function doFizzBuzz(){
        for(var i = 1; i <= 100; i++){
                console.log(fizzBuzz(i));
        }
}
```

SOURCE: McKinsey DnA Workshop - DevOps

Synchronous and Asynchronous

wistron

Asynchronous 非同步



Synchronous同步

Asynchronous 非同步

```
console.log(1);
A();
console.log(3);

function A() {
        console.log(2);
}
```

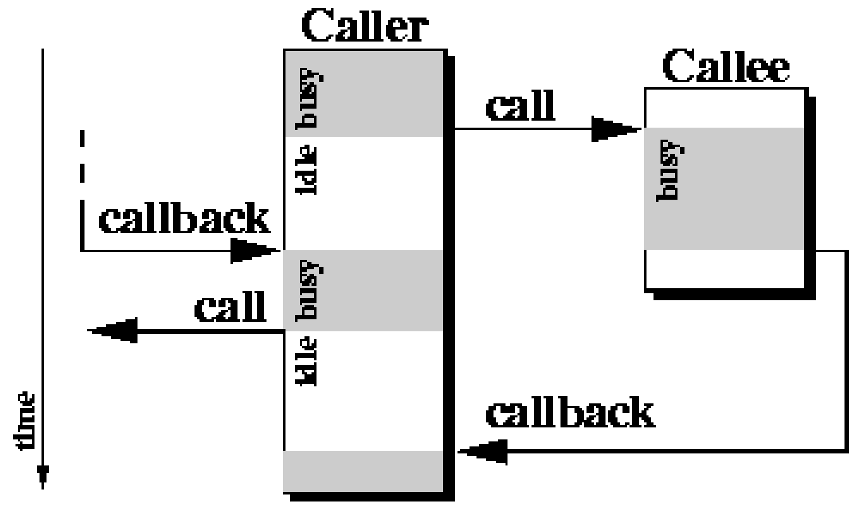
Synchronous 同步

```
console.log(1);
A();
console.log(3);

function A() {

    setTimeout(()=>
    console.log(2),0);
}
```

Call Back : 非同步回調函數

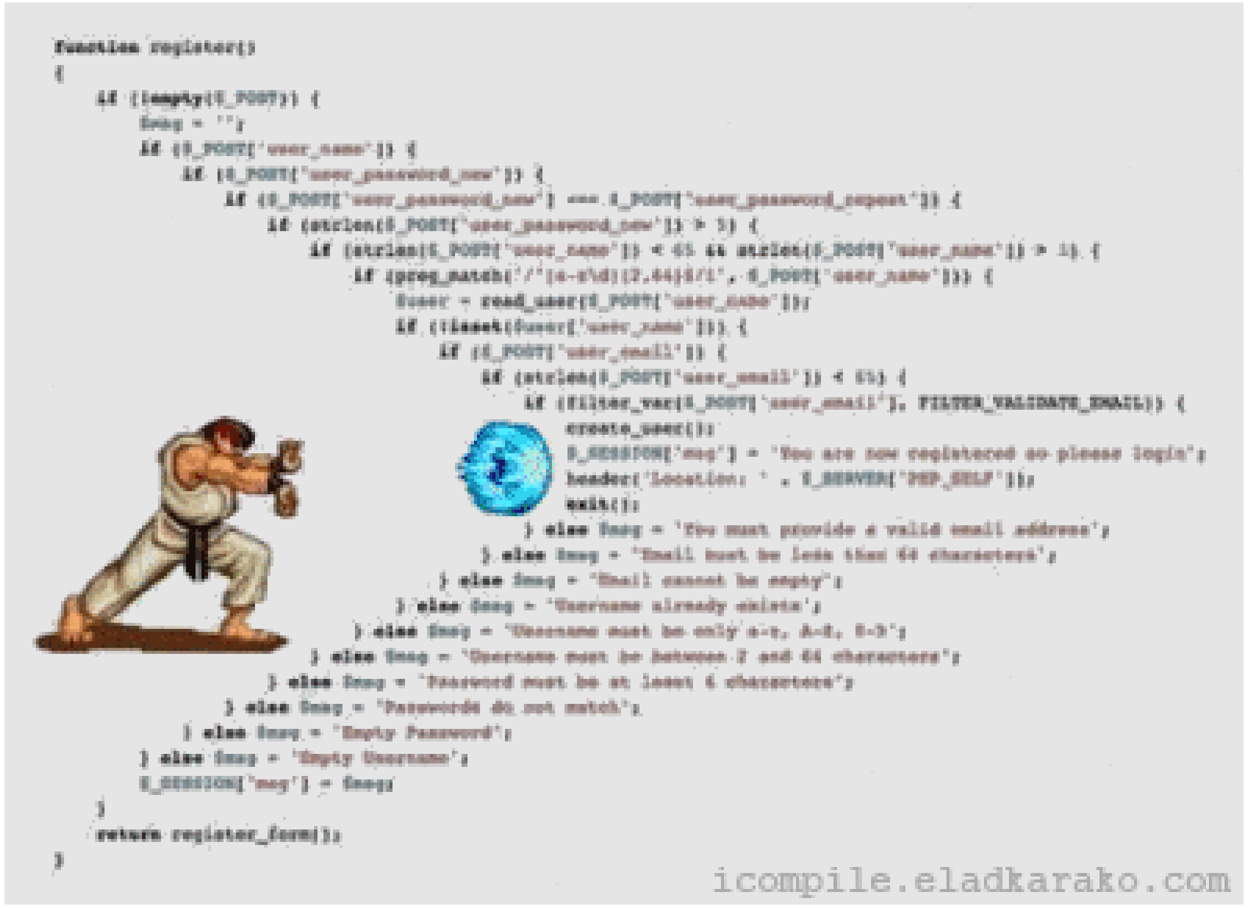
```
console.log(1);
A(A_OK);


function A_OK(){
        console.log(3);
}
```

```
function A(CallBack) {

    setTimeout(()=>{
    console.log(2);
    CallBack();
     },0);
}
```



Call Back Hell

```
A(function(resultA){
  b(resultA , function(resultB){
                  c(resultA,function(){
                            ……
                  })
  })
})
```



```
function register()
{
    if (!empty($_POST)) {
        $msg = '';
        if ($_POST['user_name']) {
            if ($_POST['user_password_new']) {
                if ($_POST['user_password_new'] === $_POST['user_password_repeat']) {
                    if (strlen($_POST['user_password_new']) > 5) {
                        if (strlen($_POST['user_name']) < 65 && strlen($_POST['user_name']) > 1) {
                            if (preg_match('/^[a-z\d]{2,64}$/i', $_POST['user_name'])) {
                                $user = read_user($_POST['user_name']);
                                if (!isset($user['user_name'])) {
                                    if ($_POST['user_email']) {
                                        if (strlen($_POST['user_email']) < 65) {
                                            if (filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)) {
                                                create_user();
                                                $_SESSION['msg'] = 'You are now registered so please login';
                                                header('Location: ' . $_SERVER['PHP_SELF']);
                                                exit();
                                            } else $msg = 'You must provide a valid email address';
                                        } else $msg = 'Email must be less than 64 characters';
                                    } else $msg = 'Email cannot be empty';
                                } else $msg = 'Username already exists';
                            } else $msg = 'Username must be only a-z, A-Z, 0-9';
                        } else $msg = 'Username must be between 2 and 64 characters';
                    } else $msg = 'Password must be at least 6 characters';
                } else $msg = 'Passwords do not match';
            } else $msg = 'Empty Password';
        } else $msg = 'Empty Username';
        $_SESSION['msg'] = $msg;
    }
    return register_form();
}
```

Promise

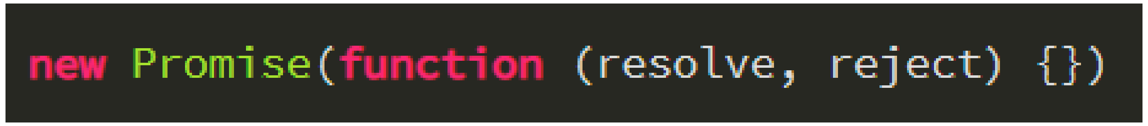
Promise，一種改善Call Back的作法

“承諾”有三種狀態：

pending：等待實現“承諾”

resolved：履行“承諾”

rejected：拒絕“承諾”



```
new Promise(function (resolve, reject) {})
```

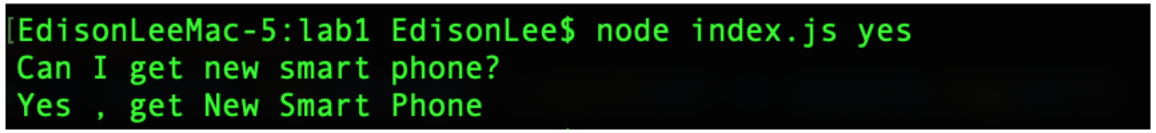
Think Great · Act Smart

wistron

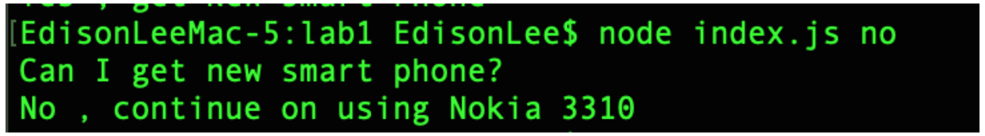
Lab2
Promise

Promise根據參數回覆不同訊息

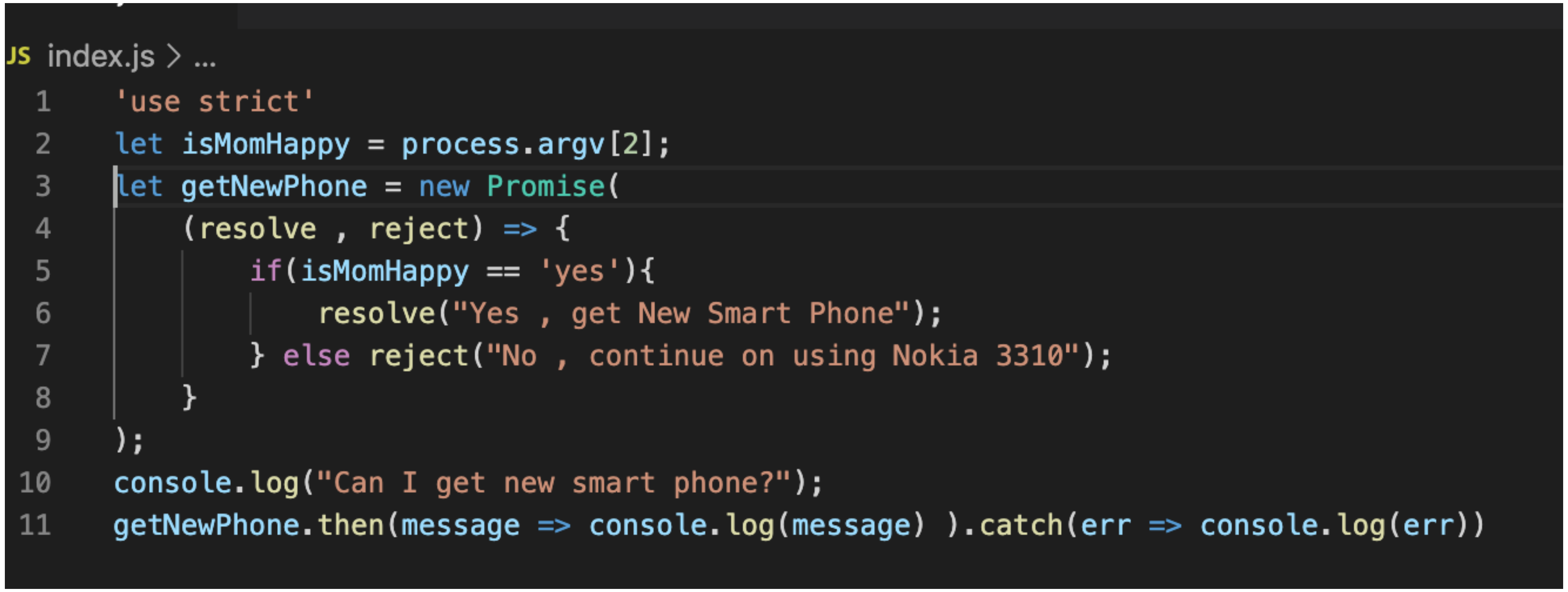
1. 使用 NPM 建立專案
2. 建立檔案 index.js
3. 在檔案中寫入代碼
4. 運行程式碼 : node index.js



```
EdisonLeeMac-5:lab1 EdisonLee$ node index.js yes
Can I get new smart phone?
Yes , get New Smart Phone
```



```
EdisonLeeMac-5:lab1 EdisonLee$ node index.js no
Can I get new smart phone?
No , continue on using Nokia 3310
```



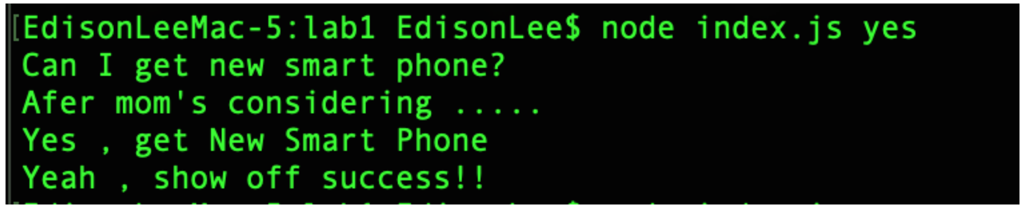
index.js > ...

```javascript
'use strict'
let isMomHappy = process.argv[2];
let getNewPhone = new Promise(
    (resolve , reject) => {
        if(isMomHappy == 'yes'){
            resolve("Yes , get New Smart Phone");
        } else reject("No , continue on using Nokia 3310");
    }
);
console.log("Can I get new smart phone?");
getNewPhone.then(message => console.log(message) ).catch(err => console.log(err))
```

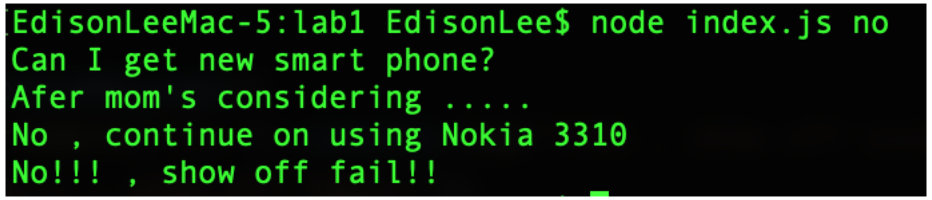
Lab3
Sync/Await

使用Sync / Await 處理Promise

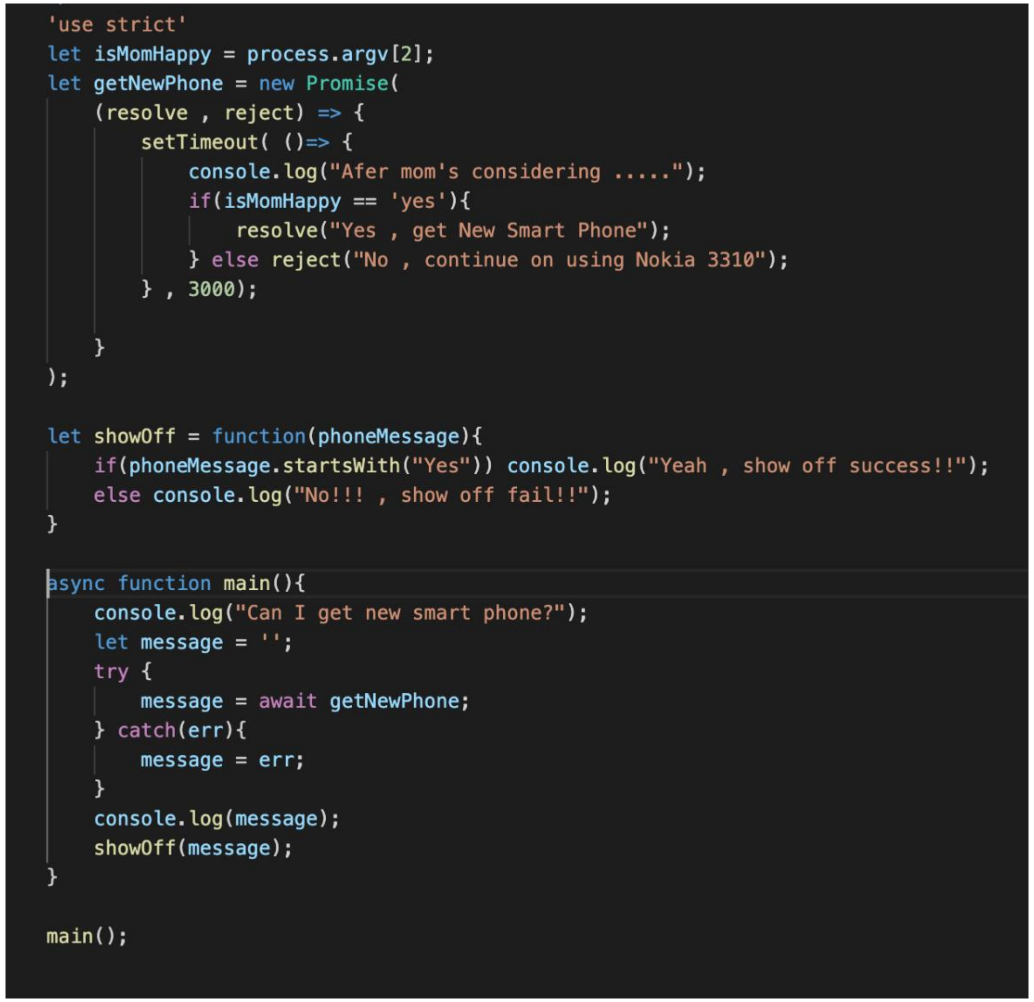
1. 使用 NPM 建立專案
2. 建立檔案 index.js
3. 在檔案中寫入代碼
4. 運行程式碼 : node index.js



```
[EdisonLeeMac-5:lab1 EdisonLee$ node index.js yes
Can I get new smart phone?
Afer mom's considering .....
Yes , get New Smart Phone
Yeah , show off success!!
```



```
EdisonLeeMac-5:lab1 EdisonLee$ node index.js no
Can I get new smart phone?
Afer mom's considering .....
No , continue on using Nokia 3310
No!!! , show off fail!!
```



```javascript
'use strict'
let isMomHappy = process.argv[2];
let getNewPhone = new Promise(
    (resolve , reject) => {
        setTimeout( ()=> {
            console.log("Afer mom's considering .....");
            if(isMomHappy == 'yes'){
                resolve("Yes , get New Smart Phone");
            } else reject("No , continue on using Nokia 3310");
        } , 3000);

    }
);

let showOff = function(phoneMessage){
    if(phoneMessage.startsWith("Yes")) console.log("Yeah , show off success!!");
    else console.log("No!!! , show off fail!!");
}

async function main(){
    console.log("Can I get new smart phone?");
    let message = '';
    try {
        message = await getNewPhone;
    } catch(err){
        message = err;
    }
    console.log(message);
    showOff(message);
}

main();
```

Thank You