

# 心理與神經資訊學

## (Psychoinformatics & Neuroinformatics)

課號: Psy5261

教室: 彷彿在雲端

識別碼: 227U9340

時間: 二789





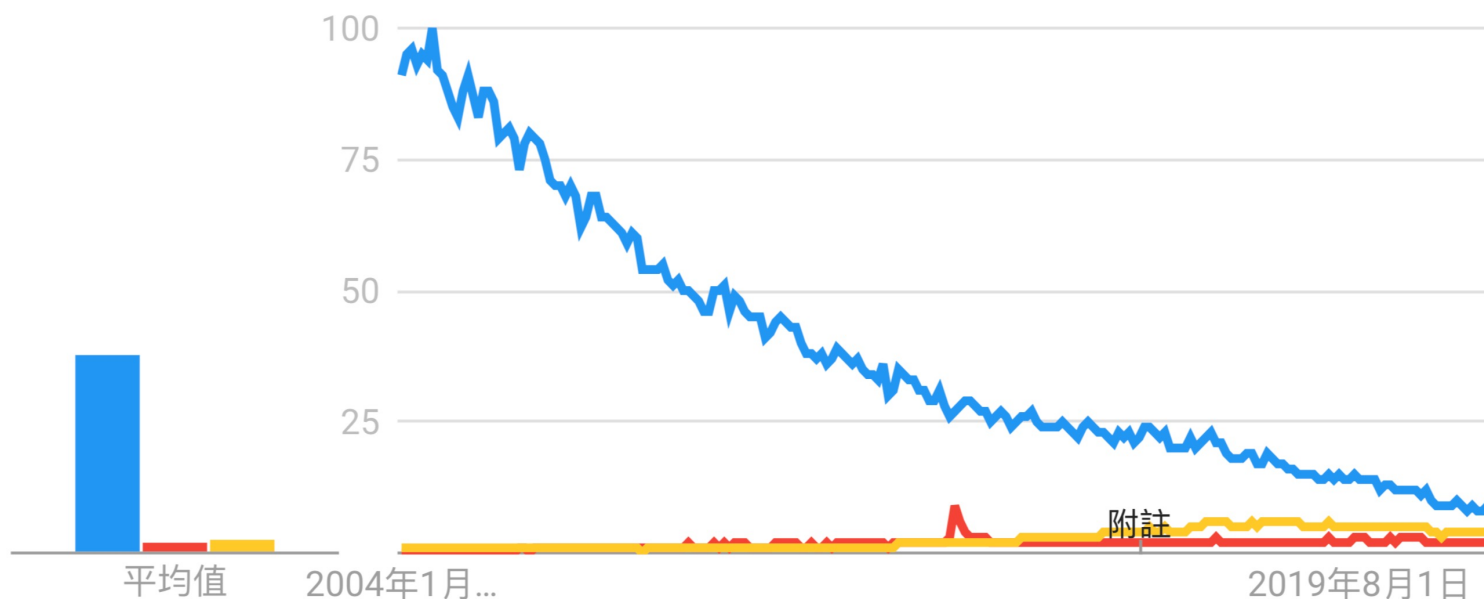
# 後端技術

## (PHP & Node.js)

# PHP vs. Django vs. Node.js

這三種後端的流程度為何？

● PHP ● Django ● Node.js



在美國PHP和Node.js快黃金交叉了

# PHP vs. Django

這兩種後端的方法要學/用哪一種？

## PHP

- 語法類似C/Java/Javascript/Perl
- 適合靜態為主動態為輔的網頁
- 易學且使用人口和參考文件都豐富

## Django

- Python語法(但有很多自己的規定)
- 適合動態為主靜態為輔的網頁
- 難學且使用人口和參考文件都較少

# PHP vs. Node.js

這兩種後端的方法要學/用哪一種？

## PHP

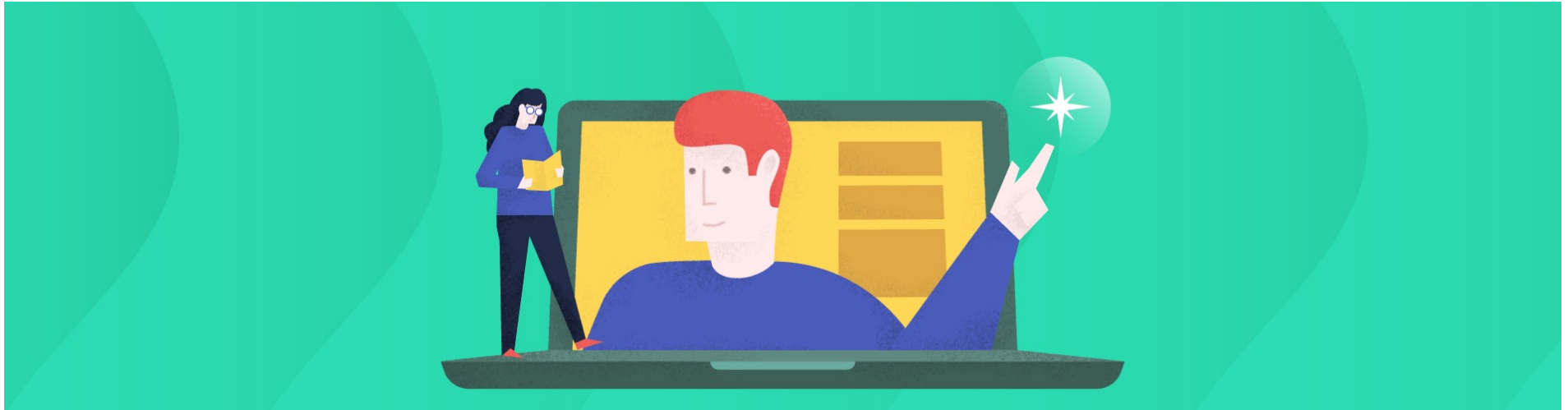
- 語法類似C/Java/Javascript/Perl
- 效能較差
- 能和HTML混搭
- 易學且使用人口和參考文件都豐富

## Node.js

- Javascript語法(但要處理很多server events)
- 效能較好(因non-blocking I/O)
- 不能和HTML混搭
- 使用人口較少且學習門檻較高

# 同步化的PHP vs. 非同步化的JS

JS中與I/O無關的指令不等待I/O的指令(即非同步化)




當秘書去查取其它資料時(即slow I/O),  
老闆可以不等待她，先做其它與資料無關的事。

非同步化/指令間不互相等待的JS工作效率比PHP高

# 非同步化/不等待的缺點

若很多blocking/sequential codes會變成callback hell

```
1  function hell(win) {  
2    // for listener purpose  
3    return function() {  
4      loadLink(win, REMOTE_SRC+'/assets/css/style.css', function() {  
5        loadLink(win, REMOTE_SRC+'/lib/async.js', function() {  
6          loadLink(win, REMOTE_SRC+'/lib/easyXDM.js', function() {  
7            loadLink(win, REMOTE_SRC+'/lib/json2.js', function() {  
8              loadLink(win, REMOTE_SRC+'/lib/underscore.min.js', function() {  
9                loadLink(win, REMOTE_SRC+'/lib/backbone.min.js', function() {  
10               loadLink(win, REMOTE_SRC+'/dev/base_dev.js', function() {  
11                loadLink(win, REMOTE_SRC+'/assets/js/deps.js', function() {  
12                 loadLink(win, REMOTE_SRC+'/src/' + win.loader_path + '/loader.js', function() {  
13                  async.eachSeries(SCRIPTS, function(src, callback) {  
14                   loadScript(win, BASE_URL+src, callback);  
15                  });  
16                 });  
17                });  
18               });  
19              });  
20             });  
21            });  
22           });  
23          });  
24         });  
25        });  
26       }  
    }  
  }
```



但有將此階層性架構扁平化的各種JS套件



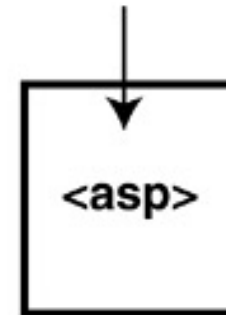
# Request傳送資料方式: Get vs. Post

## Using GET

`http://www.somedomain.com/register.asp?name=jobe&email=jobe@electrotank.com`



比較方便



## Using POST

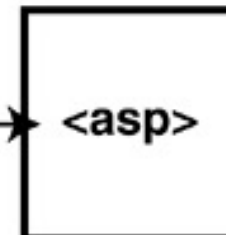
比較安全

`http://www.somedomain.com/register.asp`

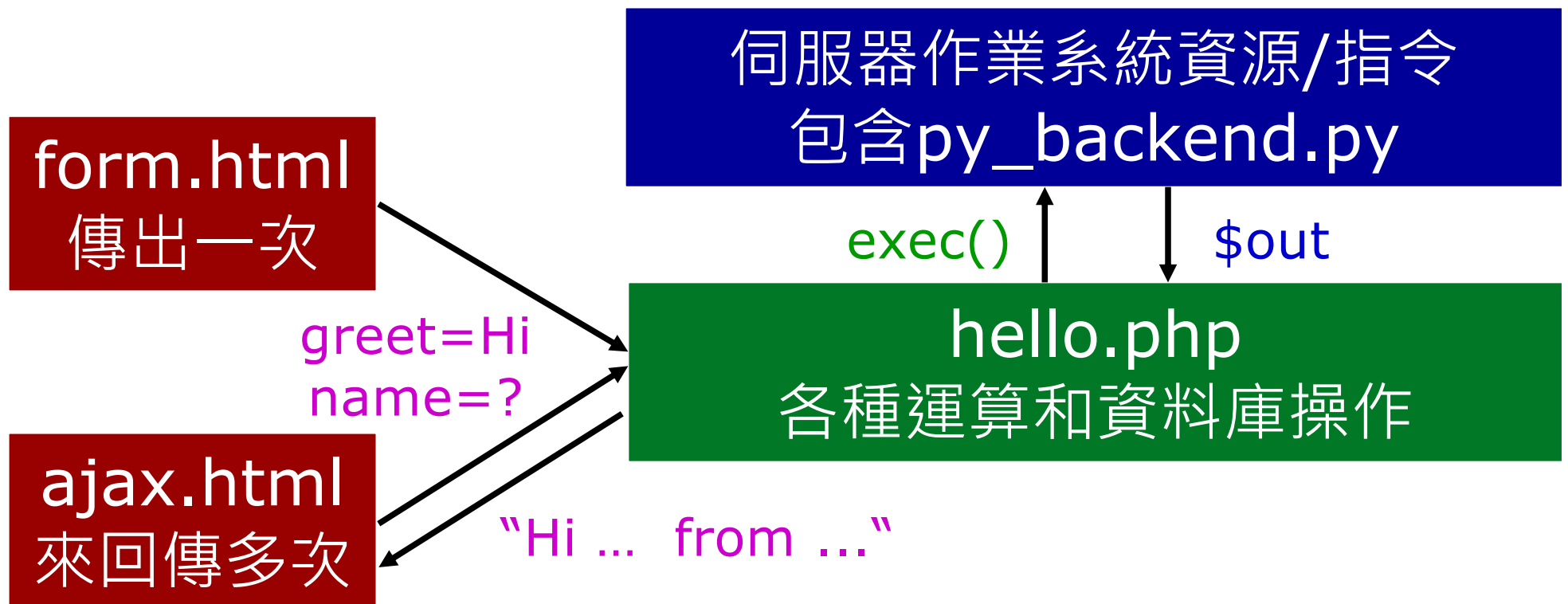


HTTP Request

`name=jobe&  
email=jobe@  
electrotank.com`



# 前端→後端 (1/2)



form.html

```
<form name="input" action="hello.php" method="get">
<input type="hidden" name="greet" value="Hi">
Username: <input type="text" name="name">
<input type="submit" value="Submit">
</form>
```

# 前端→後端 (2/2)

hello.php

```
<?php
$NAME=escapeshellarg($_GET['name']);;
$IP=$_SERVER['REMOTE_ADDR'];
echo $_GET['greet']." $NAME from $IP <br>";
exec("python py_backend.py $NAME",$out);
exec("echo $NAME >> data.txt"); // append to data.txt
for($i=0;$i<count($out);$i++){
    echo $out[$i].'<br>';
}
?>
```

py\_backend.py

```
import sys
print(str(sys.argv))
```

# Node.js(+Express)的版本

Node's req.query.name =  
PHP's \$\_GET['name']

hello3.js

```
var escapeshellarg = require('escapeshellarg')
var express = require('express');
var app = express();
const { execSync } = require('child_process');

app.get('/', function(req, res){
  req.query.name=escapeshellarg(req.query.name);
  res.send('name: ' + req.query.name);
  stdout=execSync('python py_backend.py '+req.query.name).toString();
  console.log(stdout);
});

app.listen(8080);
```

# 後端→前端

後端產生JSON格式的資料  
前端用jQuery的\$.getJSON()

**test\_json.php:**後端產生：

`{"name": "John" , "age": 35}`

**test\_json.html:**前端取得上述資料後秀出John35

**test\_jsonp.php:**後端產生：

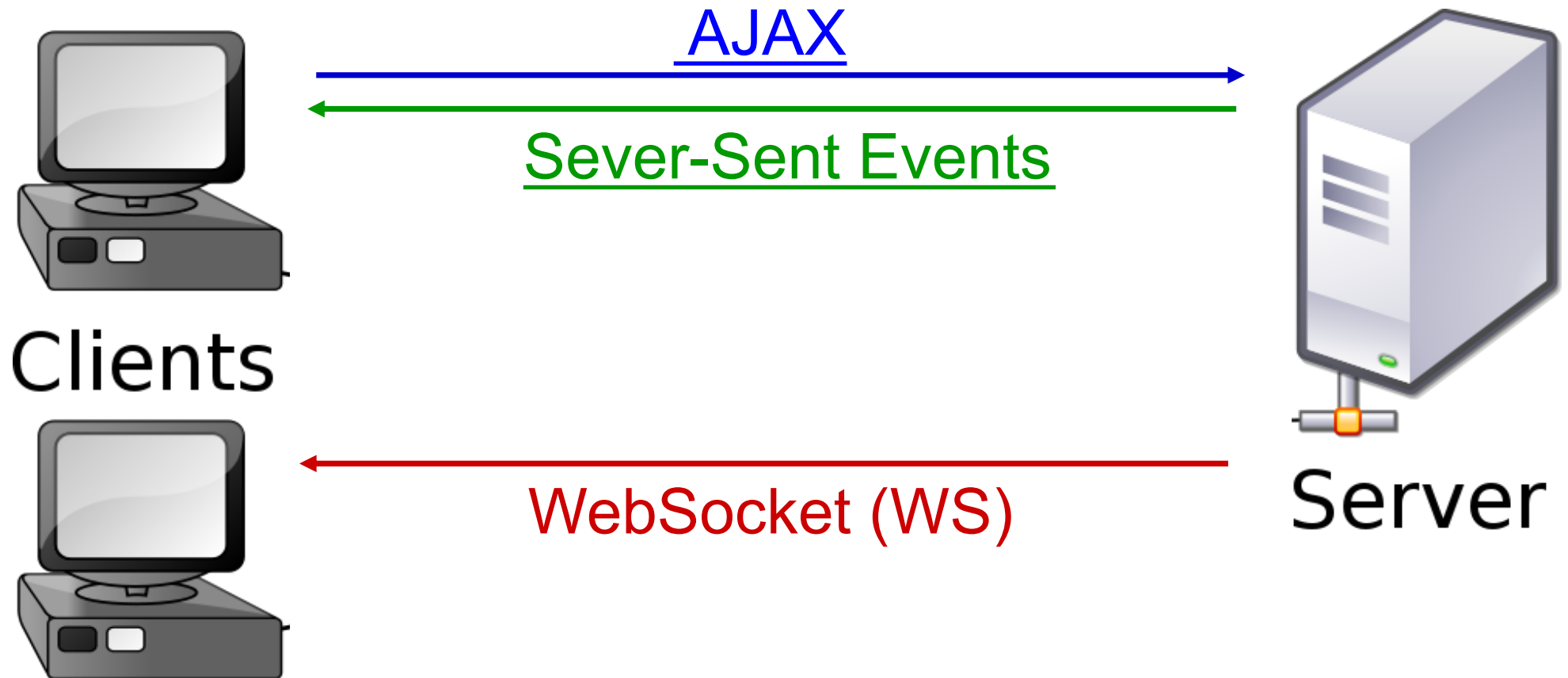
`({"name": "John" , "age": 35})`

**test\_jsonp.html:**前端取得上述資料後秀出John35

**flickr.html:** 前端取得Flickr十張圖片網址各做成<img src=...>後加入<div id="images"></div>

# Real-time後端↔前端

AJAX/SSE並無持久性的前後端連線但WebSocket有



前端HTML5與後端Node.js/PHP皆有實作WS協定

# 再比 PHP vs. Node.js

這兩種後端的方法要學/用哪一種？

## PHP

- 雖有ReactPHP: Event-driven, non-blocking I/O
- 但PHP核心和周邊各種套件本質上是序列執行
- 適合開發不需要那麼即時的大型網站

## Node.js

- JS和其套件都是event-driven, non-blocking I/O
- 要序列執行或開發大型網站反而麻煩
- 適合開發real-time bidirectional應用(如遊戲)
- 可recycle前端的JS codes (如表單驗證)

# JavaScript與PHP混搭範例

rmet/index.php用JS的nextPage()累計次數  
並呼叫自己index.php?i=k來把圖片換成k.jpg

<?=\$a?>是<? echo \$a; ?>的簡寫

\$a=(empty(\$\_GET['i']) ? 1 : \$\_GET['i']) 可展開為:

```
If(empty($_GET['i'])) { //if not there
    $a=1;
}
else{
    $a=$_GET['i'];
}
```





# 驚人的網頁程式範例(1/4)

使用HTML 5的<video>當背景



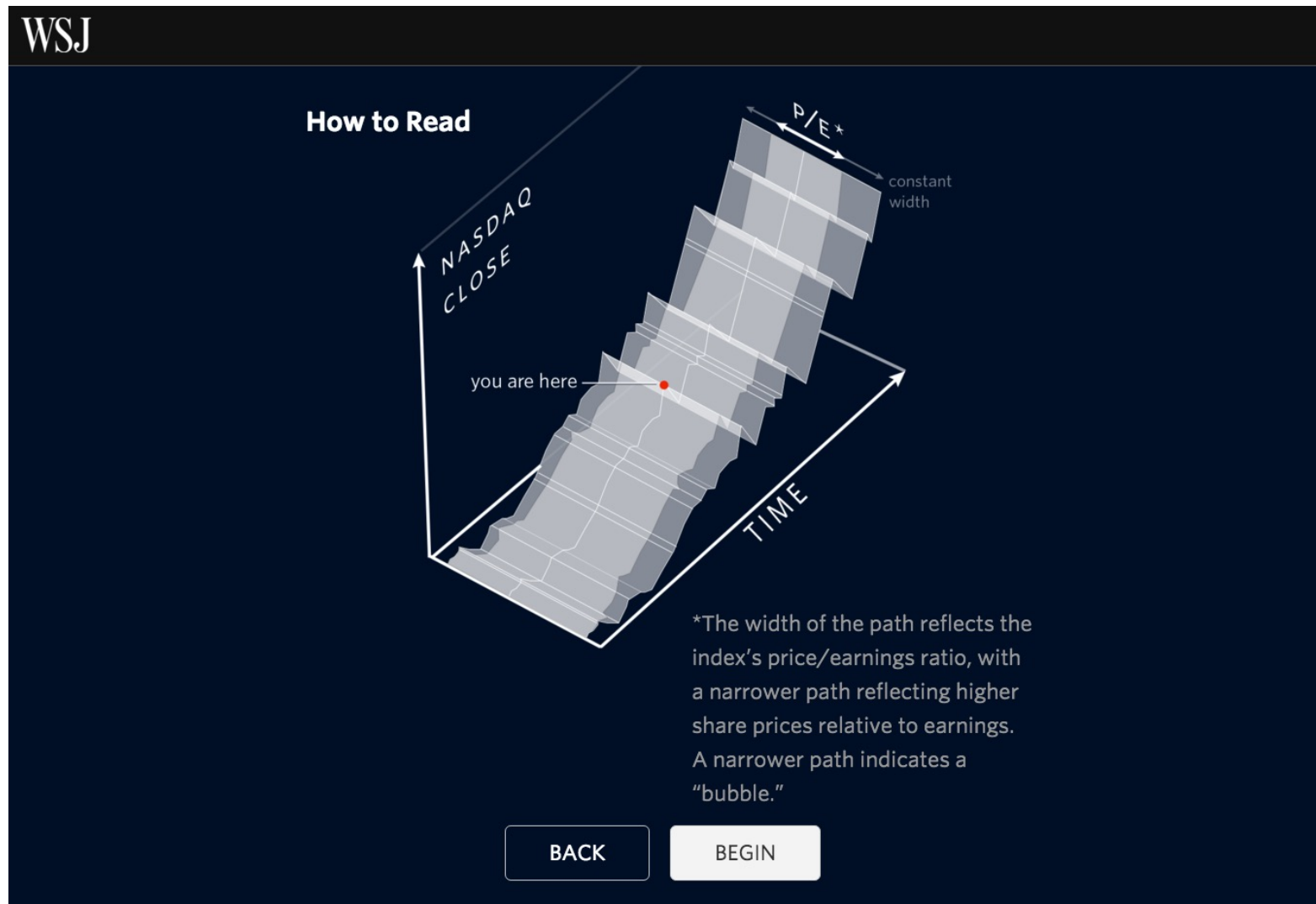
# 驚人的網頁程式範例(2/4)

使用HTML 5的<canvas> + websocket



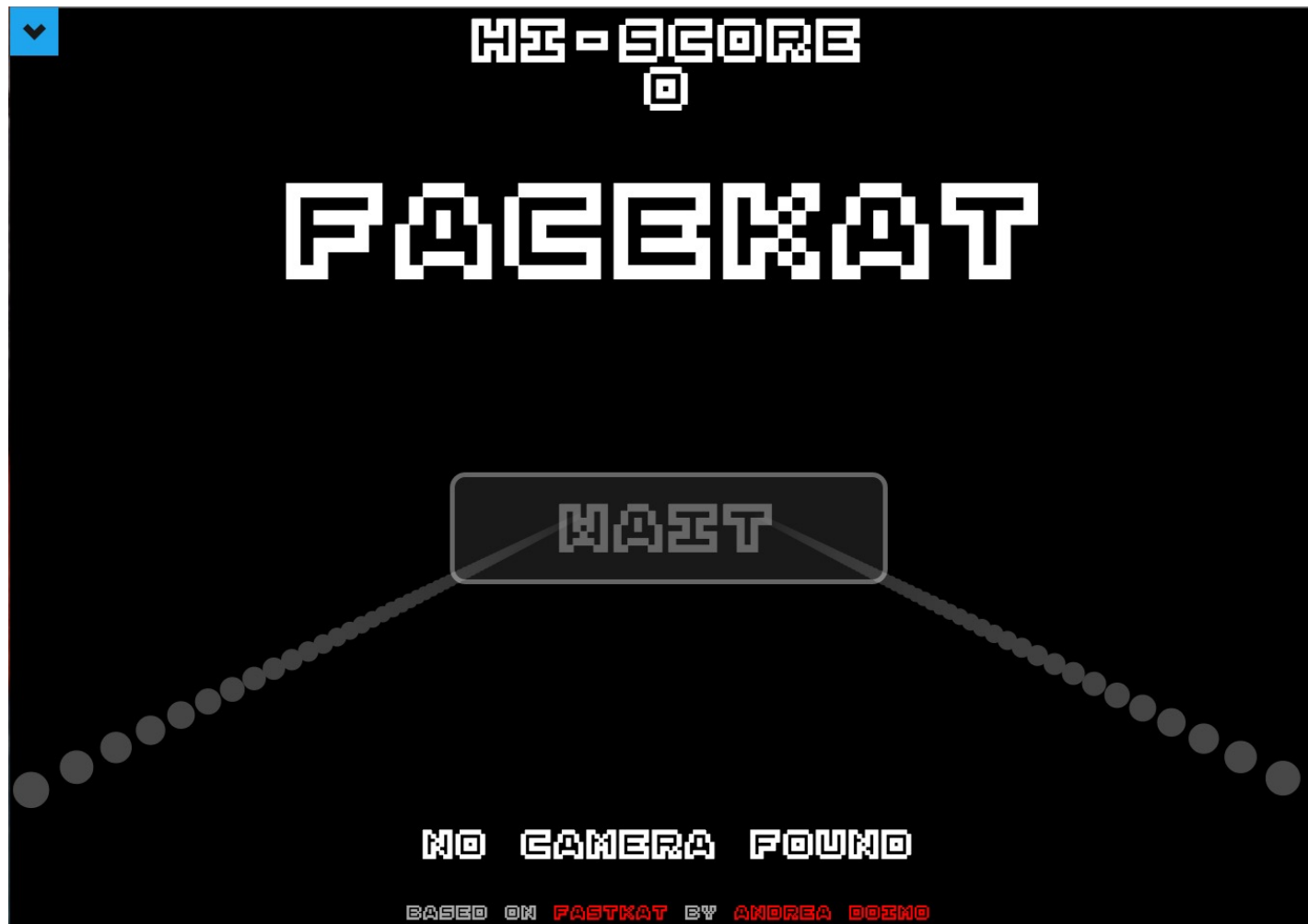
# 驚人的網頁程式範例(3/4)

<canvas> 也可以畫3D場景



# 驚人的網頁程式範例(4/4)

使用WebRTC的getUserMedia



# 開發HTML 5的遊戲

有Construct 3與GDevelop 5等遊戲引擎可以加速開發





# 本週作業

進一步研究Javascript/jQuery

1. 在jq\_sex.html用javascript的document.write()配合迴圈寫出#page1到#page3的內容來取代原始html的冗餘寫法。(4分)

2. 在測試結束後於螢幕上列印出回應正確率。(4分)



-----或-----

3. 將第三週的動物偵測圖形版改寫成網頁版本。(8分)

GAME Over

