

# 飛鴿傳書 Node.js

前端工程開發實務訓練 (6/8)

講師：蔣定宇 / josephj

# Agenda

- ◎ Node.js 是什麼
  - ❖ 網站: Facebook, Plurk...
  - ❖ PHP + Apache?
- ◎ 為什麼這麼流行
  - ❖ Event-driven, 模組化, JavaScript, 即時、新、快。

# 什麼是 Node.js

- ◎ 讓 JavaScript 不只運行在瀏覽器上。
  - ❖ 跟 ASP, PHP, JSP, 有相同的地位。
  - ❖ 例行性作業、工具的 Shell Script。
- ◎ 可獨自成為一台 Web Server。
- ◎ Event-driven 事件驅動。
- ◎ Non-blocking 非堵塞。

# Hello <Who>!

<http://f2eclass.com/lab/nodejs/hello.js>

寫一個基本的 Shell Script

```
var argv = process.argv,  
    who = (typeof argv[2] !== "undefined") ? argv[2] : "World";  
  
console.log("Hello " + who + " !");
```

> node hello.js “Awoo”

<http://nodejs.org/docs/latest/api/index.html>

未來工作排程也能用 JavaScript 來寫了！

# Hello <Who>! (2)

<http://f2eclass.com/lab/nodejs/http.js>

快速建立 Web 伺服器!

```
var http = require("http");

http.createServer(function (req, res) {
  res.setHeader("Content-Type", "text/html");
  res.write("Hello World!");
  res.end();
}).listen(8000);

console.log("Server running at http://localhost:8000");
```

<http://localhost:8000>

[http://nodejs.org/api/http.html#http\\_http](http://nodejs.org/api/http.html#http_http)

# Node.js 的地位

在 Yahoo! 的基礎技術架構是由 LAMP 構成的

## LAMP



### Linux

作業系統

### Apache

網頁伺服器

### MySQL

資料庫

### PHP

後端程式語言



Node.js 扮演了兩種角色、且功能更強大！

# 有什麼好處？

- ◎ 統一 Web 程式語言：前後端都寫 JS。
  - ❖ 將前端工程師從瀏覽器解放，後端一樣有能力開發！
  - ❖ 可利用 JavaScript Event-driven 的特性。
  - ❖ 瀏覽器關閉 JavaScript 仍然可以跑 JavaScript @@
- ◎ CommonJS 規範：比瀏覽器更有結構。
- ◎ NPM 套件管理：非常容易安裝管理套件。
- ◎ 活躍的社群。
- ◎ 開發更為即時的網站應用程式。

# CommonJS

<http://f2eclass.com/lab/nodejs/commonjs.js>

<http://f2eclass.com/lab/nodejs/commonjs-module.js>

## require 載入外部 JavaScript

commonjs.js

```
// 透過 require 將外部 JavaScript 載入
var mod = require("./commonjs-module.js");
mod.bar();
```

commonjs-module.js

```
// 並不會是全域變數
var foo = function () {
    console.log("foo is invoked");
};

// 所有要被外部存取的方法或屬性都需要用 exports
exports.bar = function () {
    console.log("bar is invoked");
};
```

> node commonjs.js

解決了全域變數與外部資源管理的問題



# NPM 套件管理



- Node.js Package Management
- 朴灵：「一個解決方案比詢問問題還多的社群」
- 用 Node.js 寫 Shell Script、部落格系統沒好處？
  - ❖ 可用的資源太多了，將大大減少開發時間！

# 使用 LiveReload

感受 npm 的強大與方便！

- ◎ 目錄檔案一但變動，網頁就會重新整理

- ❖ 按 Ctrl + F5 手動重新整理很痛苦

- ◎ 安裝 Node.js 版的 LiveReload 伺服器

- > `npm install -g simple-livereload`

- ◎ 啟動 LiveReload 伺服器

- > `cd <工作目錄>;`

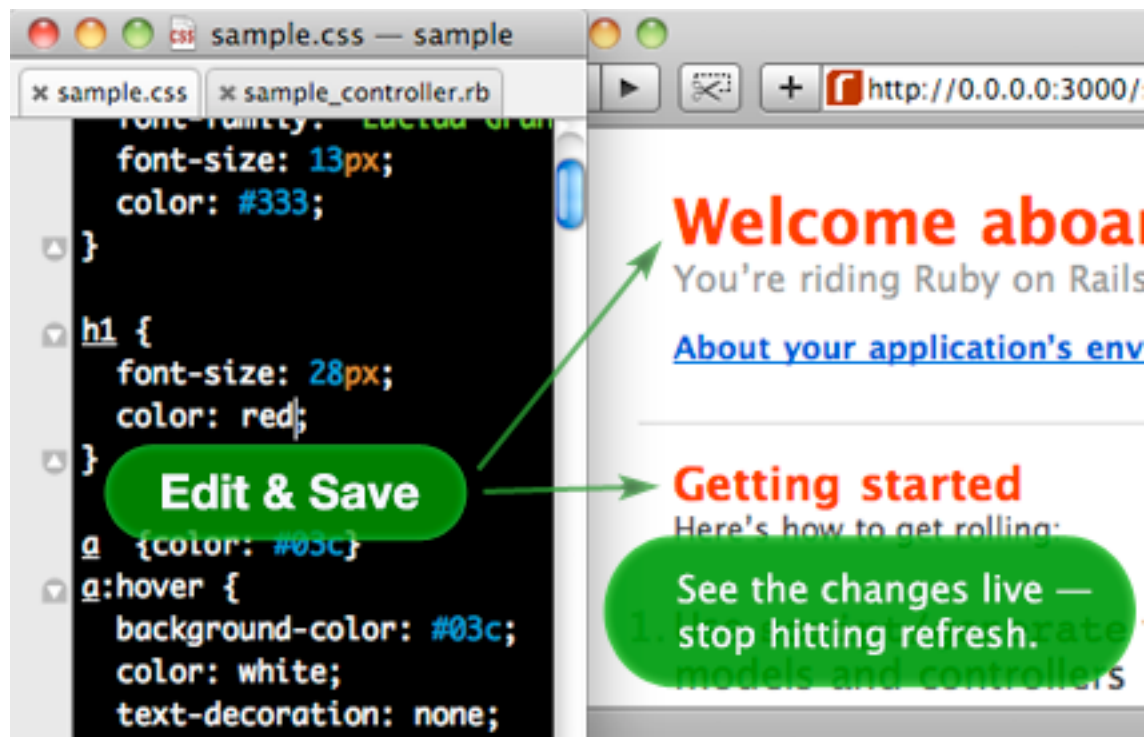
- > `simple-livereload <port>`

- ◎ 安裝 LiveReload 的客戶端

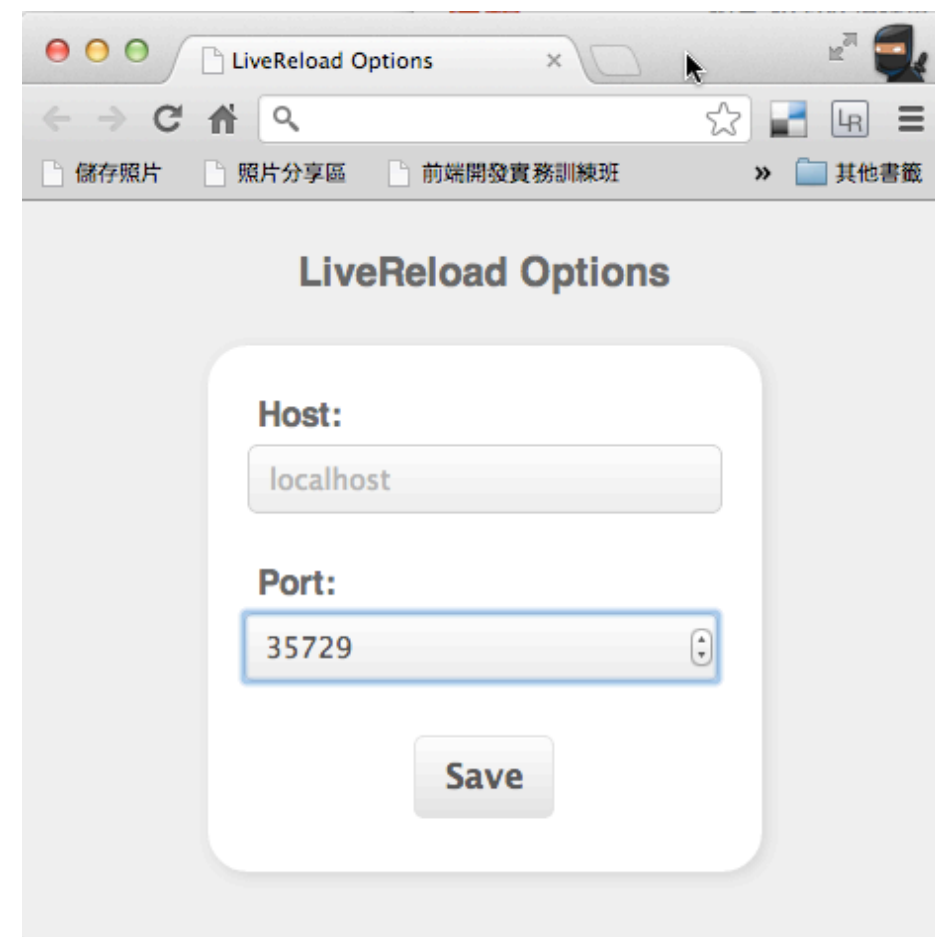
- Google 搜尋 “**Chrome LiveReload**” 就會找到

# 使用 LiveReload

檔案有修改  
瀏覽器也會更新



Chrome 的 LiveReload 設定



監聽檔案變動並回報

收到變動就重新載入

伺服器

瀏覽器

持續連線  
<port>

# 有什麼壞處？

- 太新了。還不夠穩定、API 也常在更新。
- 網頁有任何修改都得重新執行 node.exe。
- 會因為一個 JavaScript Error 就讓 Server 掛掉。
- 靜態檔案處理不易，得存取 FileSystem API。
- 有很多非同步的處理（Callback Pattern）。

# 一個錯誤就會掛掉

commonjs.js

```
// 透過 require 將外部 JavaScript 載入
var mod = require("./commonjs-module.js");
// foo(); // 未定義的函式！
foo();
```

commonjs-module.js

```
// 並不會是全域變數
var foo = function () {
    console.log("foo is invoked");
};

// 所有要被外部存取的方法或屬性都需要用 exports
exports.bar = function () {
    console.log("bar is invoked");
};
```

> node commonjs.js

碰到未定義的 foo 函式就會立刻掛掉  
其他的程式語言頂多出錯、不會全部死掉

# 不適合靜態檔案的存取

<http://f2eclass.com/lab/nodejs/html.js>

<http://f2eclass.com/lab/nodejs/html.html>

## html.js

```
var http = require("http"),
    fs = require("fs");

// 必須採用 fs.readFile 將檔案讀入
fs.readFile("./html.html", function (err, html) {

    if (err) {
        throw err;
    }

    // 建立 Web 伺服器
    http.createServer(function (request, response) {
        response.writeHead(200, {
            "Content-Type": "text/html"
        });
        response.write(html);
        response.end();
    }).listen(8000);
});
```

## html.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Node.js - 測試網頁</title>
<style type="text/css">
div {
    background: red;
}
</style>
</head>
<body>
    <div>
        Hello! 這是 Node.js 的測試網頁。
    </div>
</body>
</html>
```

> node html.js

一旦把 Web Server 跑起來，再怎麼改 html.html 都不會生效。一定要重啟才行。

# 解決方法

◎ 別忘了這是個很活躍的社區，一個問題至少有三個以上的解答 XD

❖ node-server

<https://github.com/cloudhead/node-static>

❖ paperboy

<https://github.com/felixge/node-paperboy>

❖ http-server

<https://github.com/nodeapps/http-server>

# node 介紹小結

這是一個很新的「程式語言 + 伺服器」，問題是很多的，但是由於活躍的社群、解答也出乎想像地多。

若你採用此技術作為開發，會看到許多最新的概念，但同時也必須不斷地去面對問題、逼迫自己不斷 Study，是很有挑戰性的。

以 Node.js 的本身語言特性與先天優秀的架構，它必定是未來幾年內最重要的程式語言之一，是很值得花時間投資的。



# Web 即時通訊

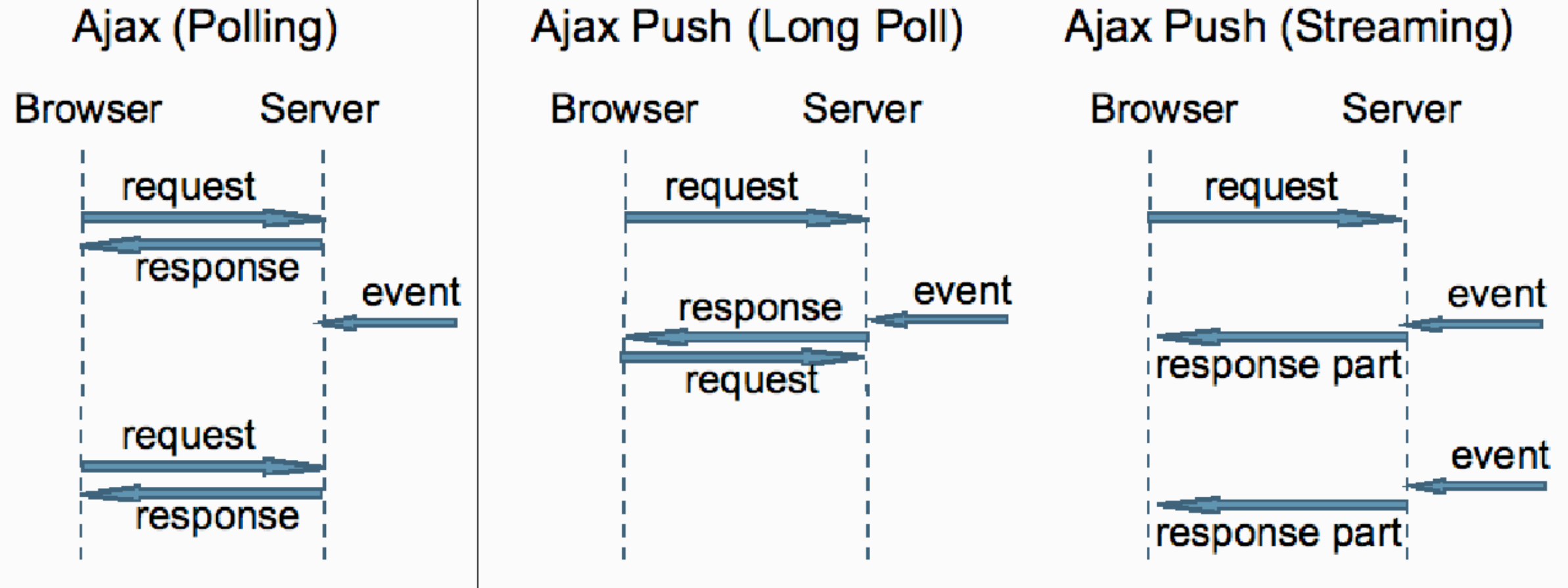
過去、現在、未來的幾種作法

最古老、不夠即時、干擾度很高的聊天室  
到最近噗浪、臉書很即時且流暢的通知  
他們各自是用什麼原理達成的呢？

這堂課讓你了解原理、引用現成工具  
未來就可以輕鬆開發 Web 的即時訊息了！

# 輪詢 Polling

long-term HTTP connections



馬上結束

把請求保留住  
有資料再回傳  
斷掉再發新的

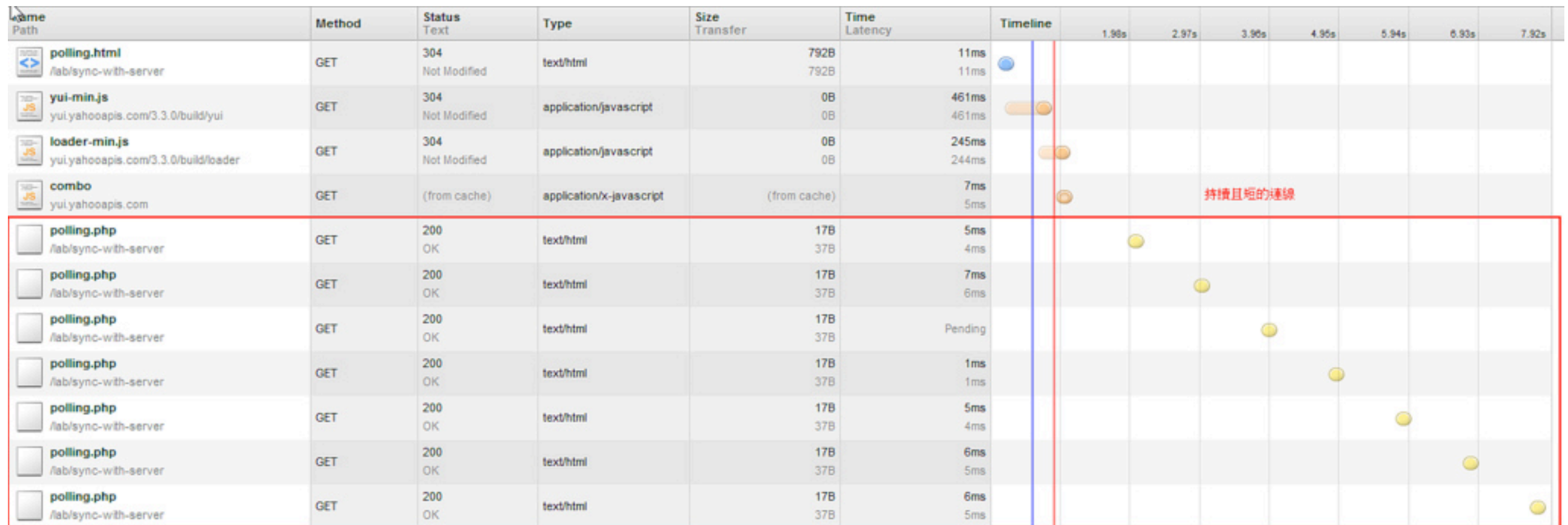
請求永遠保持

前兩種都有可能漏訊息

<http://www.slideshare.net/amix3k/comet-with-nodejs-and-v8>

# 輪詢 Polling

不斷地向伺服器發送請求，詢問是否有新訊息























伺服器的負擔：請求數量過多

<http://f2eclass.com/lab/node/polling.html>

# 長時間的 Polling

不斷地向伺服器發送請求，詢問是否有新訊息

Name Path	Method	Status Text	Type	Size Transfer	Time Latency	Timeline	5.12s	7.67s	10.23s	12.79s	15.35s	17.90s	20.46s
 long-polling.html /lab/sync-with-server	GET	200 OK	text/html	928B 489B	4ms 4ms								
 yui-min.js yui.yahooapis.com/3.3.0/build/yui	GET	200 OK	application/javascript	21.71KB 8.77KB	944ms 473ms								
 loader-min.js yui.yahooapis.com/3.3.0/build/loader	GET	200 OK	application/javascript	31.65KB 9.17KB	1.05s 504ms								
 combo yui.yahooapis.com	GET	(from cache)	application/x-javascript	(from cache)	6ms 4ms								
 comet.josephj.com comet.josephj.com	GET	200 OK	text/javascript	83B 82B	3.01s 3.01s								
 comet.josephj.com comet.josephj.com	GET	200 OK	text/javascript	83B 83B	3.04s 3.01s								
 comet.josephj.com comet.josephj.com	GET	200 OK	text/javascript	83B 83B	3.04s 3.00s								
 comet.josephj.com comet.josephj.com	GET	200 OK	text/javascript	83B 83B	3.04s 3.00s								
 comet.josephj.com comet.josephj.com	GET	200 OK	text/javascript	83B 83B	3.06s 3.01s								
 comet.josephj.com comet.josephj.com	GET	200 OK	text/javascript	83B 83B	3.04s 3.01s								

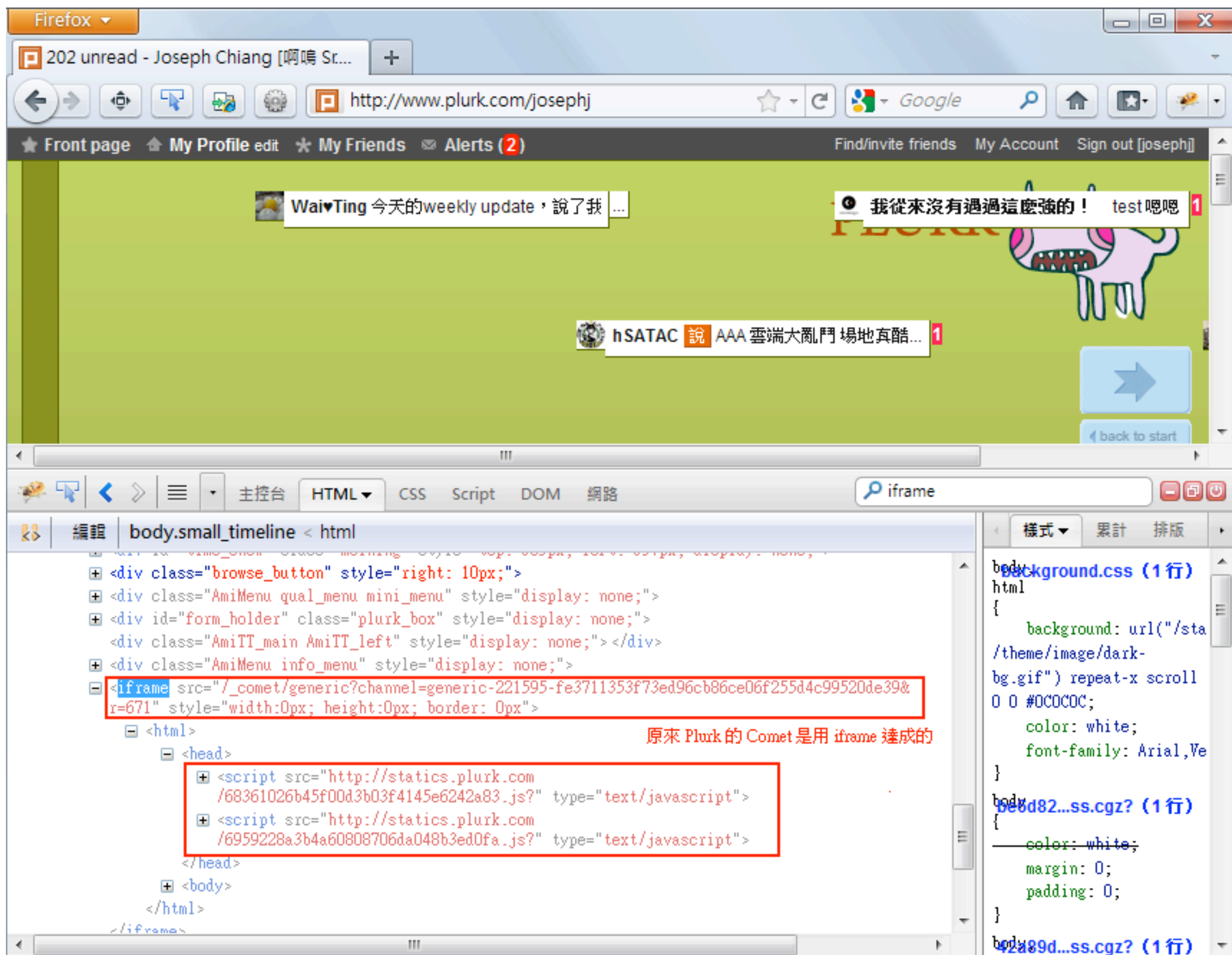
較長且不規律的連線、Server 有值才終止  
(範例設定的太規律了 :p)

伺服器的負擔：請求數量仍多、而且連線時間長

<http://f2eclass.com/lab/node/long-polling.html>







# 關於輪詢










Long-polling 在傳統像是 Apache 的網站伺服器是非常吃記憶體，最好還是採用 nginx 或 node

兩種 Polling 都是單向的：

由 Browser 向 Server 詢問是否有新訊息 (pull)  
而非有新訊息就由 Server 推向 Browser (push)

# Web Socket

## 建立雙向通道

Name Path	Method	Status Text	Type	Size Transfer	Time Latency	Timeline	44ms	66ms	87ms	109ms	131ms	153ms
 <b>web-socket.html</b> /~joseph_chiang/sync-with-server	GET	200 OK	text/html	1.21KB 585B	11ms 7ms							
 <b>yui-min.js</b> yui.yahooapis.com/3.3.0/build/yui	GET	200 OK	application/javascript	21.71KB 8.77KB	39ms 21ms							
 <b>loader-min.js</b> yui.yahooapis.com/3.3.0/build/load	GET	200 OK	application/javascript	31.65KB 9.17KB	43ms 24ms							
 <b>combo</b> yui.yahooapis.com	GET	(from cache)	application/x-javascript	(from cache)	1ms 1ms							
 <b>test</b> 172.17.66.2	GET	101 WebSocket Protocol Handshake	Pending	0B 0B	Web Socket 通道一直有在做事，但看不出來 Pending							

有資料的時候才互相傳遞，效能最好的作法！

<http://f2eclass.com/lab/node/websocket.html>



# 即時通訊小結

WebSocket 是 HTML5 所帶來的新協定，瀏覽器實作不一，所以沒辦法畢其功於一役。我們仍得針對較差的瀏覽器採用 Long-polling、甚至採用過渡時期的 Flash WebSocket。

要看瀏覽器支援度與是否安裝 Flash、實作 Long-polling, Flash WebSocket, WebSocket（前後端都要改），這也太辛苦了吧！

# 最佳解決方案！



Socket.io 也是 Node.js 的一個套件  
可以自行偵測瀏覽器環境、決定要採用何種方案  
你的程式只要寫一份就可以啦 :D

<http://socket.io/>

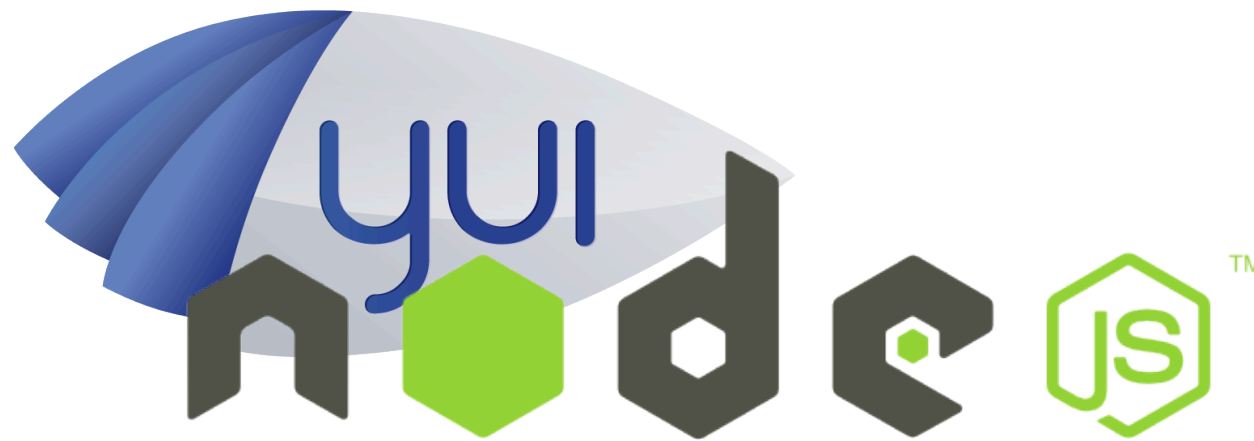
# 在不同瀏覽器執行

<http://f2eclass.com/lab/node/socket-io.js>

觀察看看瀏覽器是採用何種方法取得新訊息的。

# YUI in Node.js

YUI 與 Node.js 的完美結合



Write it once, use everywhere!  
讓你寫的 YUI 在前後端都能夠執行！

<https://github.com/davglass/nodejs-yui3>  
<http://www.youtube.com/watch?v=quMt0w7aK4s>

# 實作照片網站的通知

當有人上傳了一張新照片  
其他正在線上的人都能收到通知

# 相關資源

- ◎ 固定聚會: Node.js Taiwan Party  
<http://nodejs.tw/>
- ◎ Node.js 台灣社群協作電子書  
<http://book.nodejs.tw/>
- ◎ CNode: Node.js 专业中文社區  
<http://cnodejs.org>
- ◎ Node Packaged Modules  
<https://npmjs.org/>