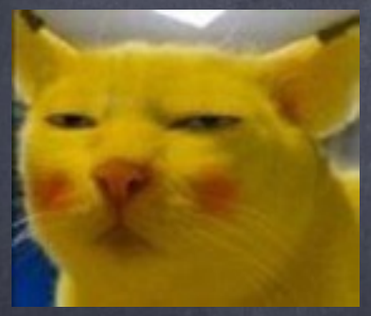
AngularJS 實作與應用

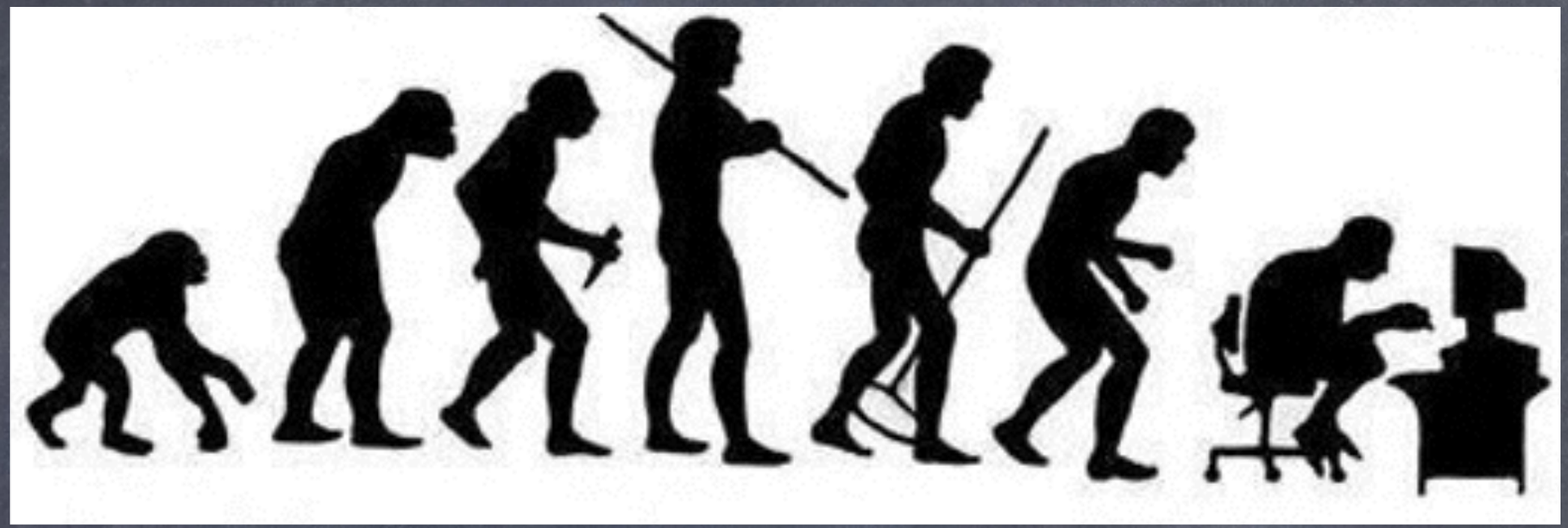


資訊服務處劉智漢

2015.04.01

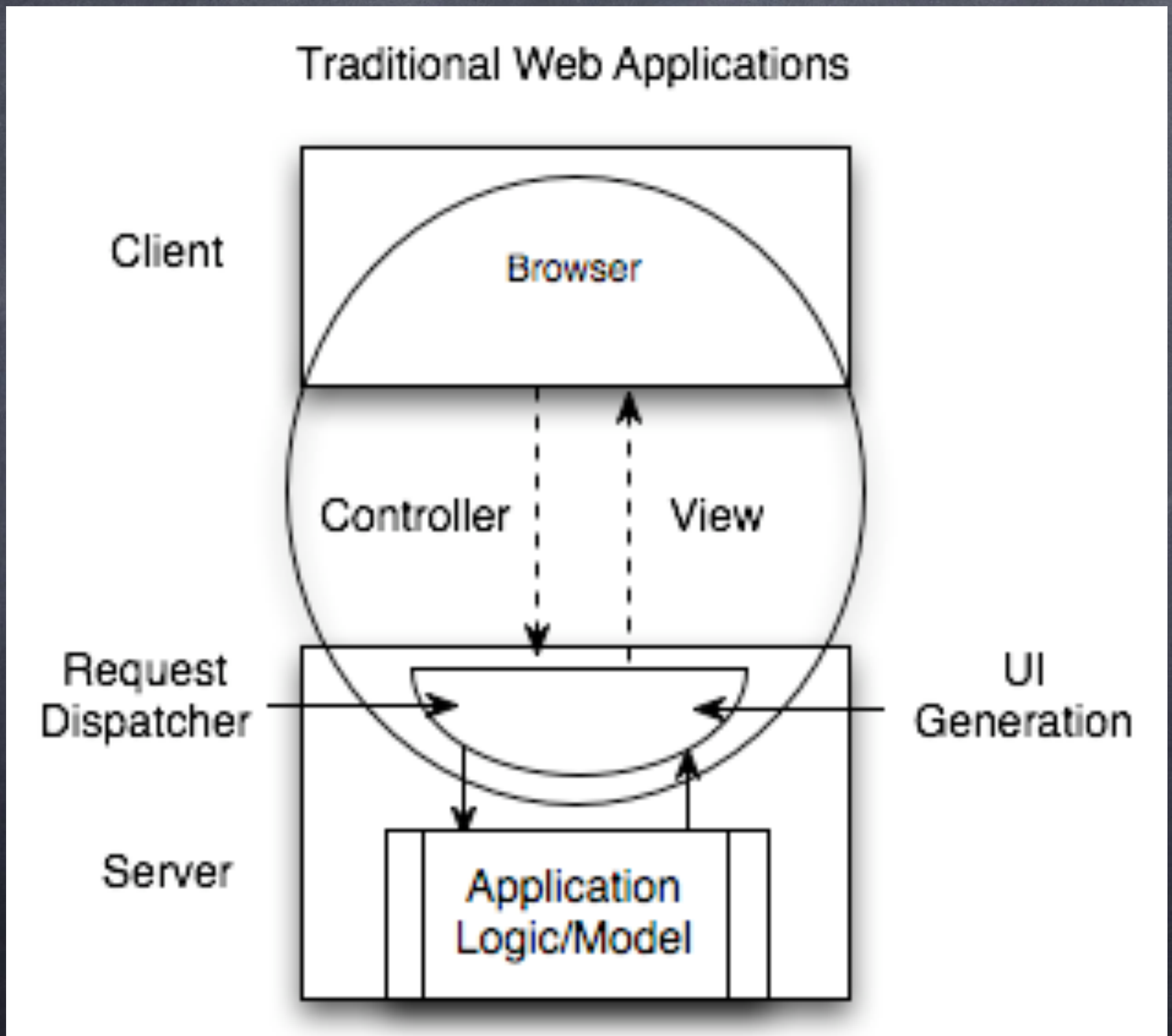
@中央研究院 人文館

Web 的演進



- 靜態HTML 檔案
- CGI 出現：由CGI產生HTML
- Server side 語言出現：php、jsp...
- Flash/silverlight RIA出現
- Jquery、AJIX
- HTML5
- MVC、SPA

為何要使用MVC?傳統網頁設計架構



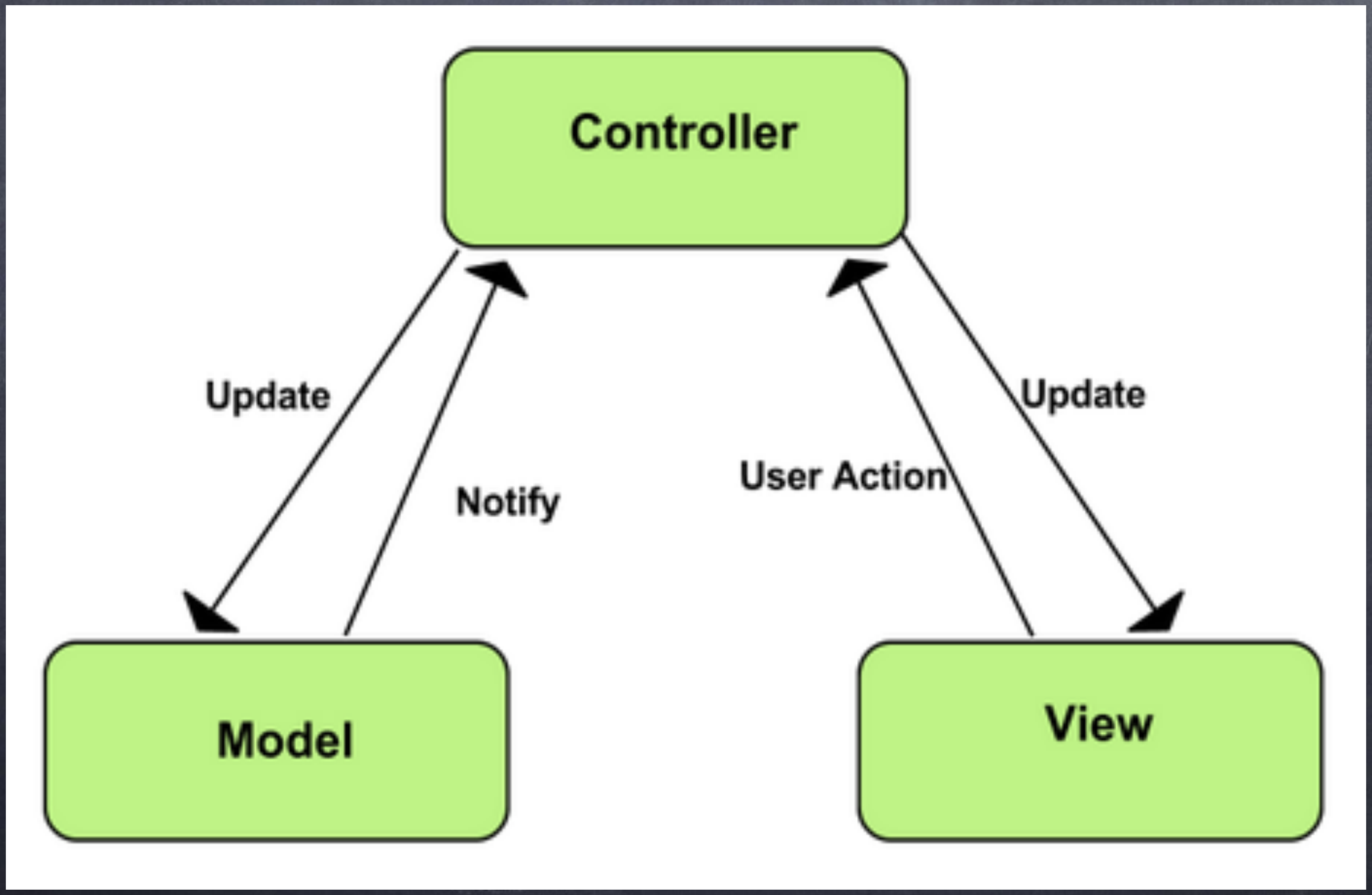
傳統網頁設計的缺點

- Server 負擔太重
- 網頁反應過慢
- 被駭客攻擊風險提高
- 無法離線存取
- 程式設計不易、UI、資料與邏輯皆混在一起

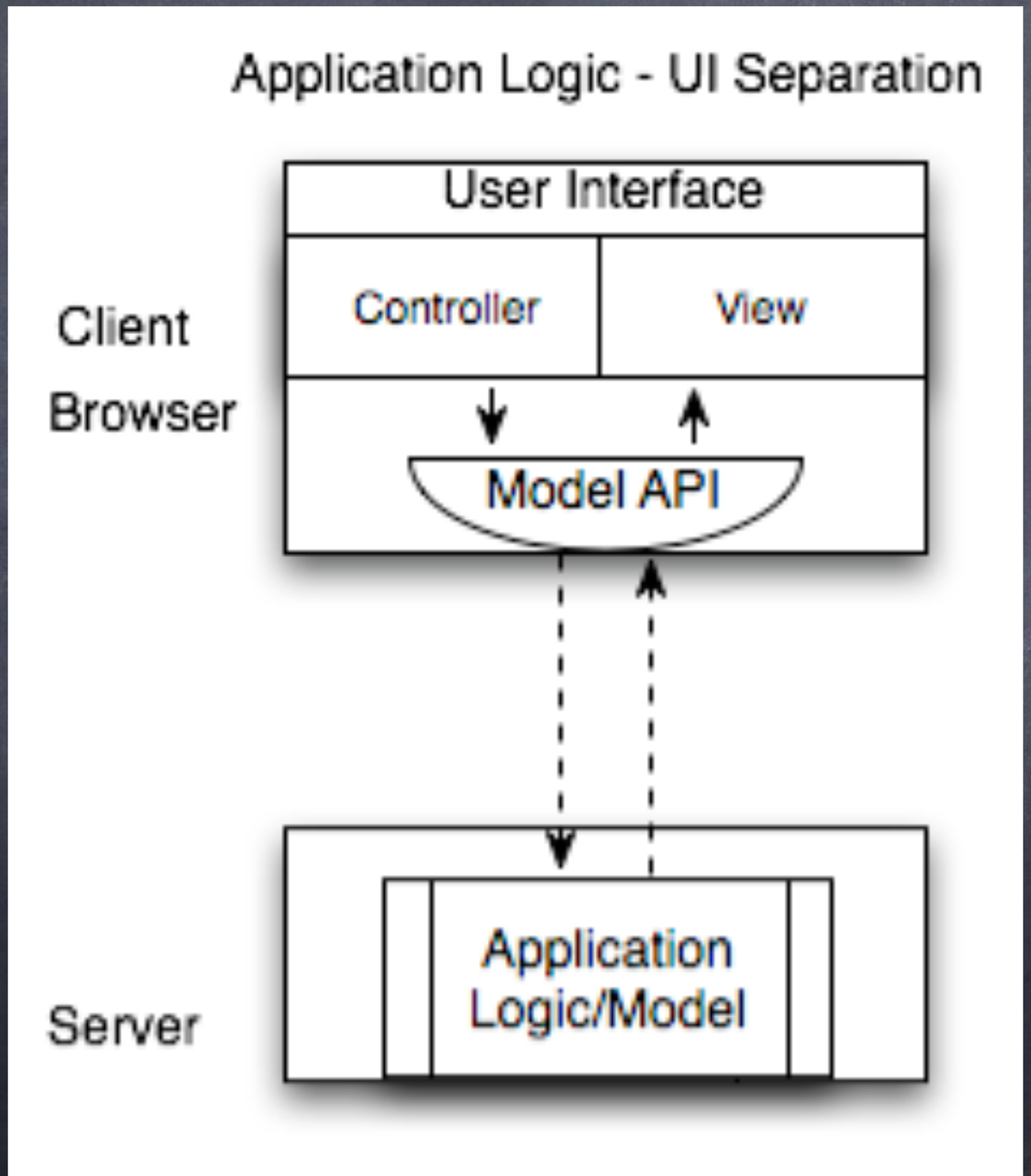
MVC 介紹

- model-view-controller 縮寫
- 20世紀70年代 Smalltalk 開始有這個概念
- 核心概念為：將資料(模型), 應用程式邏輯(控制器)及給用戶呈現資料(UI)作一分離
- Model
  - 用來存放資料，連結 view 與 Controller
  - $scope / $rootScope / Custom Model
- View
  - 以DOM 為範本(不使用string 為範本），透過 ng-model 來啟動雙向資料鏈結
  - 僅作UI呈現，不要把程式、商業邏輯放在 view 中
- Controller
  - 用來定義應用程式的商業邏輯
  - 屬性(property原始型別、物件型別）、事件(event)、方法(method)

MVC 介紹（續）— 關係圖



MVC 設計架構

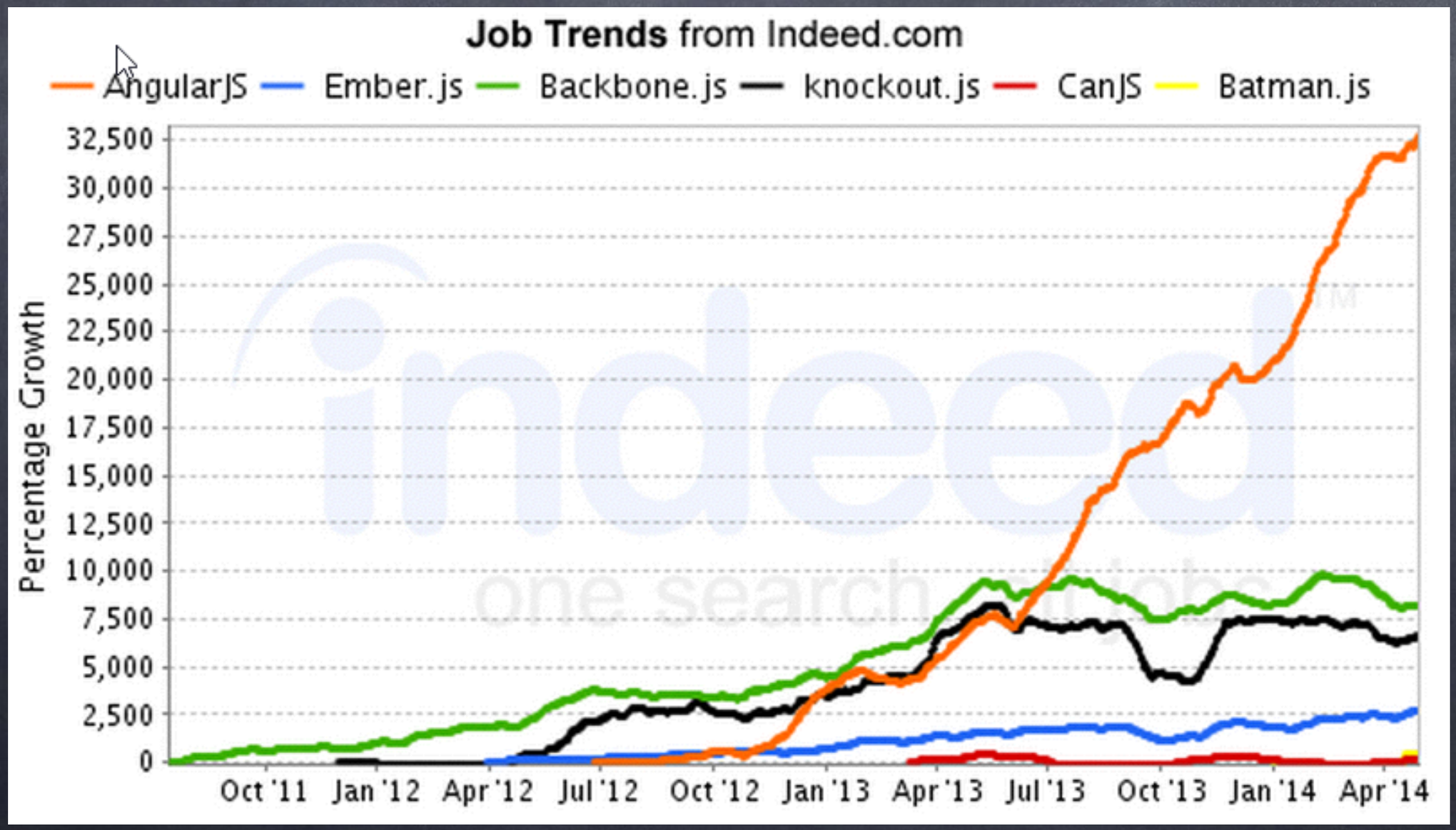

Application Logic - UI Separation
User Interface
Controller
View
Client
Browser
Model API
Application
Logic/Model
Server

MVC 設計優點

- 網頁反應迅速
- 程式前後端切分清楚、權責劃分容易
- client 端的狀態管理更加容易
- 可以搭配其他工具製作離線應用程式
- 後端可以有更多選擇

程式設計趨勢

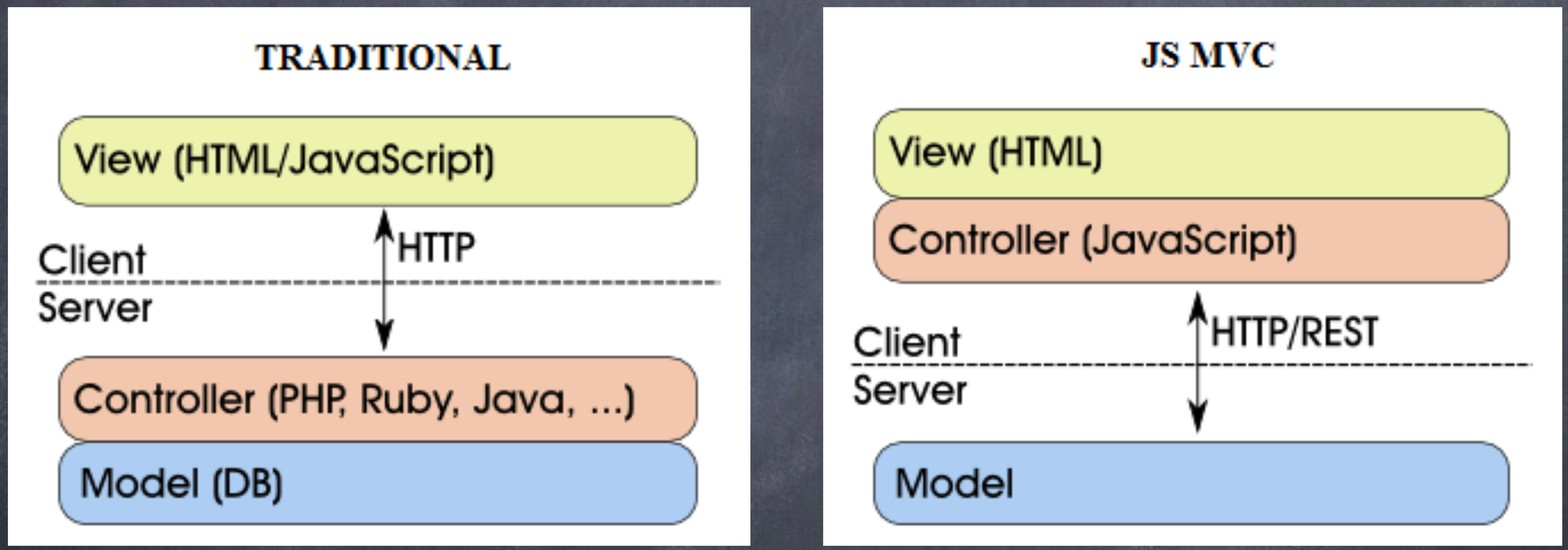
職場對JS工程師需求趨勢



(Image from indeed.com 2014)

程式設計趨勢

商業邏輯由Server轉移到Client



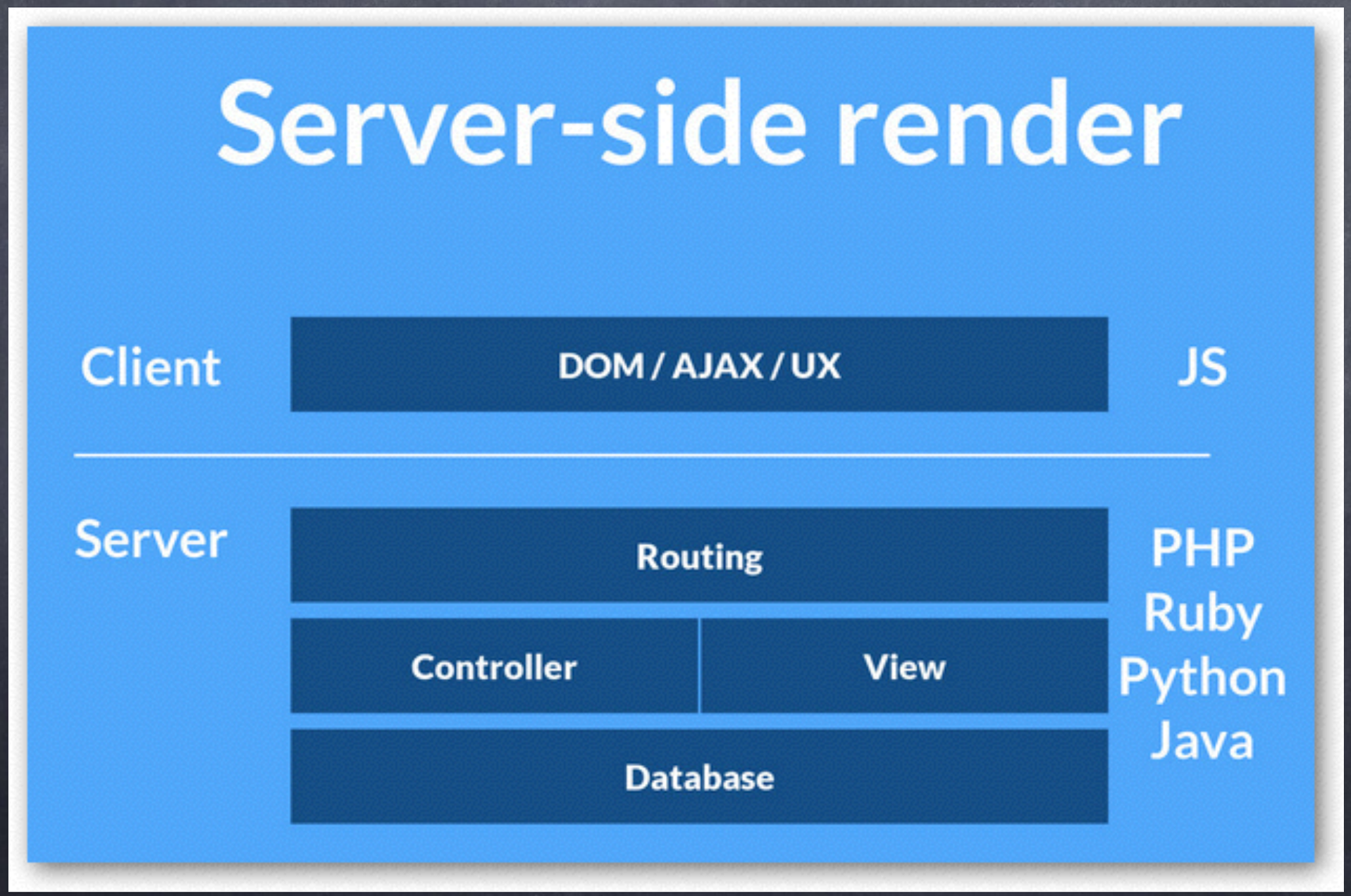
(Image from stuartsierra.com)

程式設計趨勢

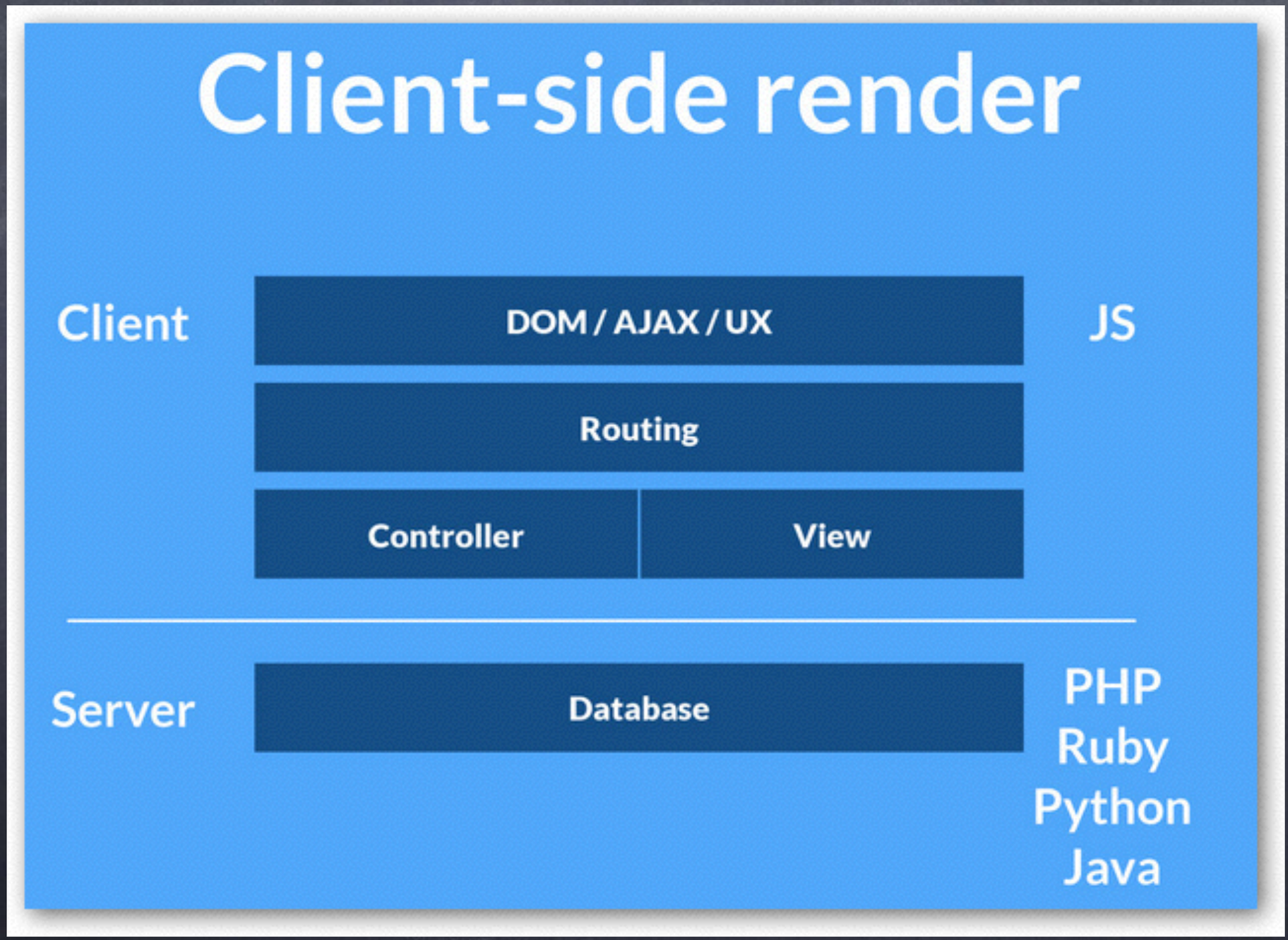
- 前後設計端逐漸分離，已非以往的全端工程師
- 新名詞出現
  - 前端工程師
  - 後端工程師
- 前後端語言web部分有逐漸統一的趨向

傳統的程式設計師

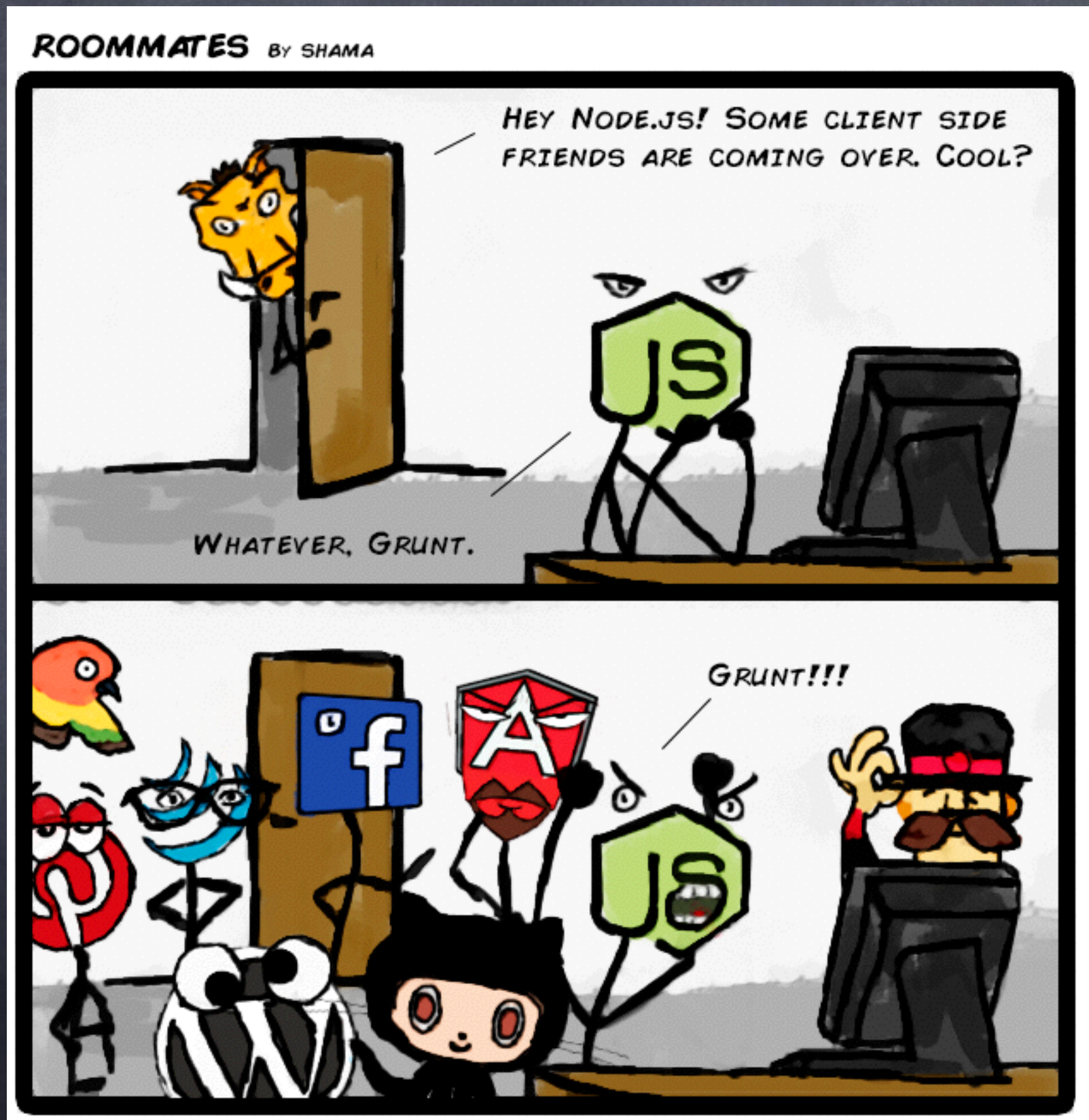
全端工程師



前/後端工程師


Client-side render
Client
DOM / AJAX / UX
JS
Routing
Controller
View
Server
Database
PHP
Ruby
Python
Java

前後端語言統一趨勢


ROOMMATES BY SHAMA
HEY NODE.JS! SOME CLIENT SIDE FRIENDS ARE COMING OVER. COOL?
JS
WHATEVER. GRUNT.
GRUNT!!!
JS
CC BY-SA
DONTKRY.COM

SPA
Single Page Application

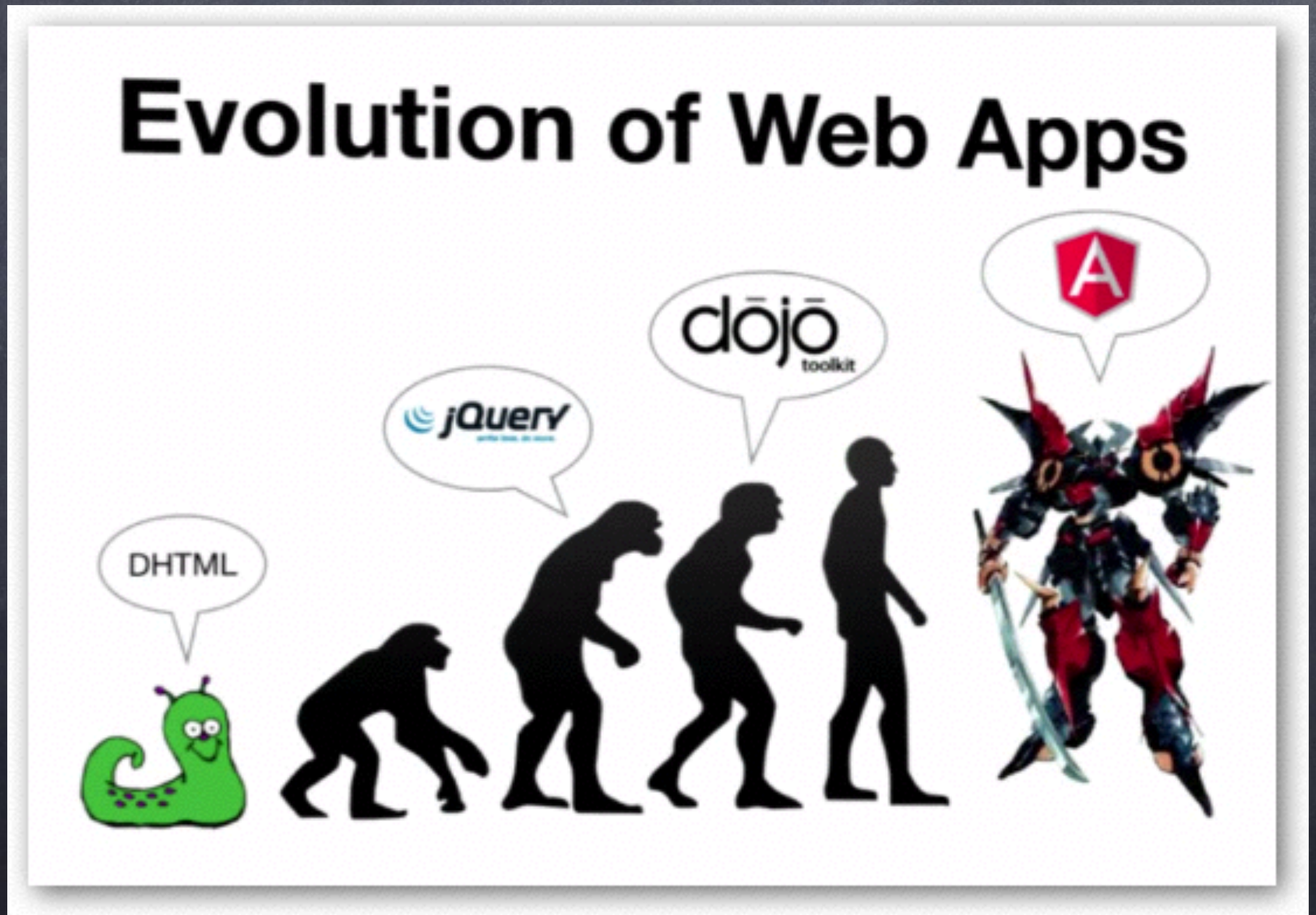
- 應用程式藉由將畫面切割成多個view來呈現網頁內容
- 這些內容更動時不需要整個頁面重整
- 只需要針對特定的view作資料重整即可
- 因此，整個應用程式就是由一個網頁組成
- 範例：科研採購網

小結

- 當前端與後端分離後，後端變成是資料提供者
- 前端：
  - 擁有完整MVC，擺脫由後端傳送畫面、邏輯
  - SPA 的出現，提升網頁使用效率

Section 2
AngujarJS 簡介

Web framework 演進過程



AngularJS 介紹

- 由Google 開發的一套MVC framework
- 目標在擴充 MVC 能力來建立 browser-based 應用程式
  - 直接拿HTML(DOM)當成 template使用（高效能）
  - 將DOM變成可重複使用的原件
  - 將程式碼(Code-behind)綁定在DOM元素上
- AngularJS 是一套完整的 SPA framework.
- 提供完整的單元測試及End-2-End測試架構
- 類似的專案：https://vaadin.com/home

ANGULARJS 適用範圍

- AngularJS 適用於 CRUD 類型的網站專案中
  - 內建提供：資料繫結、表單驗證...等
- AngularJS 並不管應用程式設計時所使用的軟體架構及設計語言（php、python、java）

MVW架構
MODEL VIEW WHATEVER

- 關注點分離原則
  - 將商業邏輯由HTML中完整抽離
  - 區分網站前、後端，簡化後端開發工作
- AngularJS 鼓勵程式開發者將程式切割成數個小區段
  - Module
  - Service、controllers
  - directives、filters、constants

AngularJS V.S JQuery

	jQuery	AngularJS
Abstracts the DOM	✔	✔
Unit Test Runner	✔	✔
Deferred Promises	✔	✔
Cross-Module Communication	✔	✔
Animation Support	✔	✔
AJAX / JSONP	✔	✔
RESTful API	✘	✔
Integration Test Runner	✘	✔
MVC Pattern Support	✘	✔
Templating	✘	✔
Two-way Data Binding	✘	✔
Dependency Management	✘	✔
Deep-Link Routing	✘	✔
Form Validation	✘	✔
Localization	✘	✔
File Size	32KB	38KB

開始撰寫AngularJS程式

- step1. 載入AngularJS framework
- step2. 建立模組module
- step3. 建立作用域及控制器

Lab.1 Hello World

建立作用域及控制器

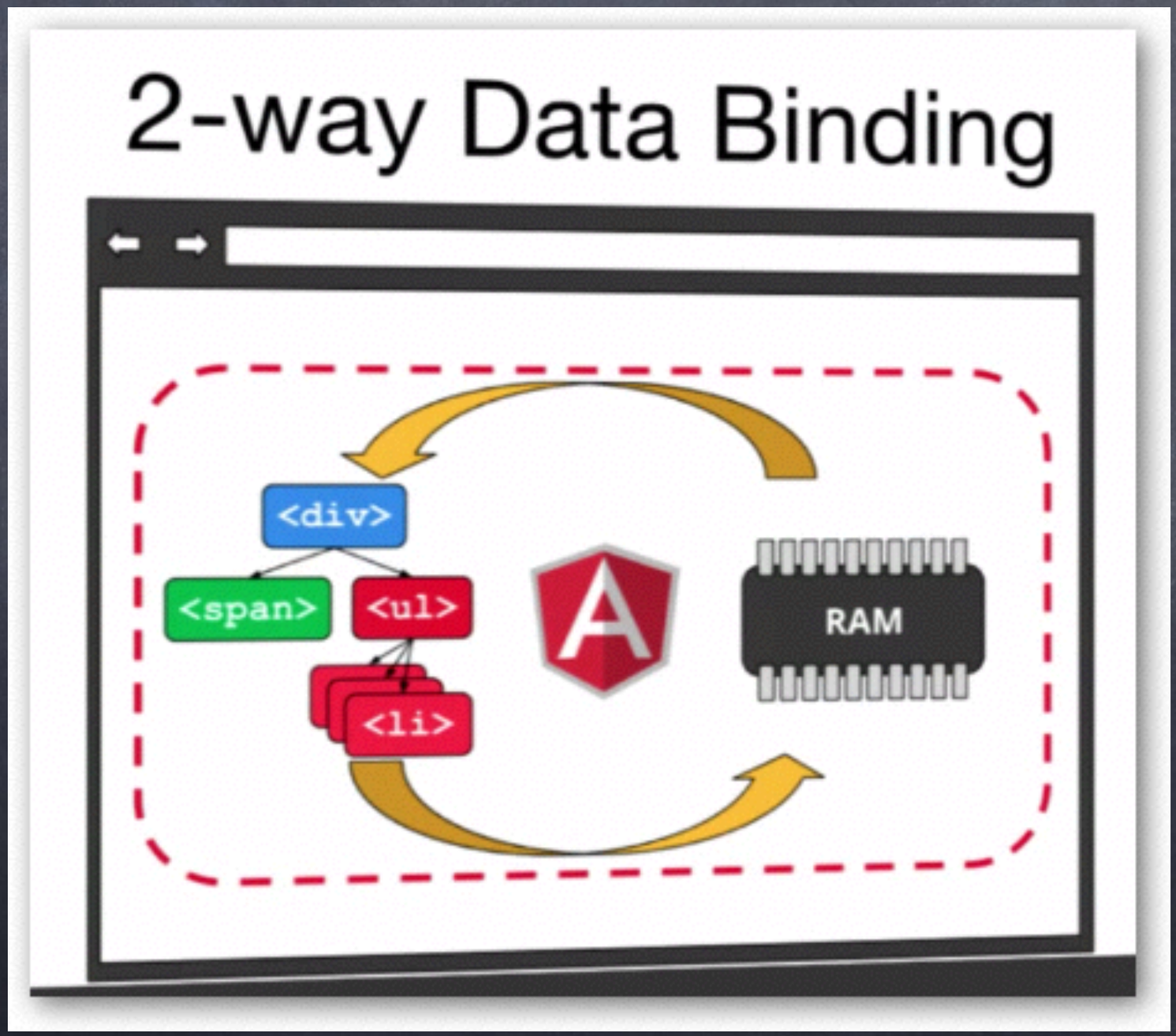
- 作用域宣告：ng-app ng-app="test"
- 一隻應用程式只能有一個作用域
- ng-app 主要是用來告知 Angular 從網頁的這個位置開始啟動（接管）
- 如果你只要在網頁的一部分執行AngularJS，只需要在特定的區域（ex:Div）鍵入 ng-app 就可以

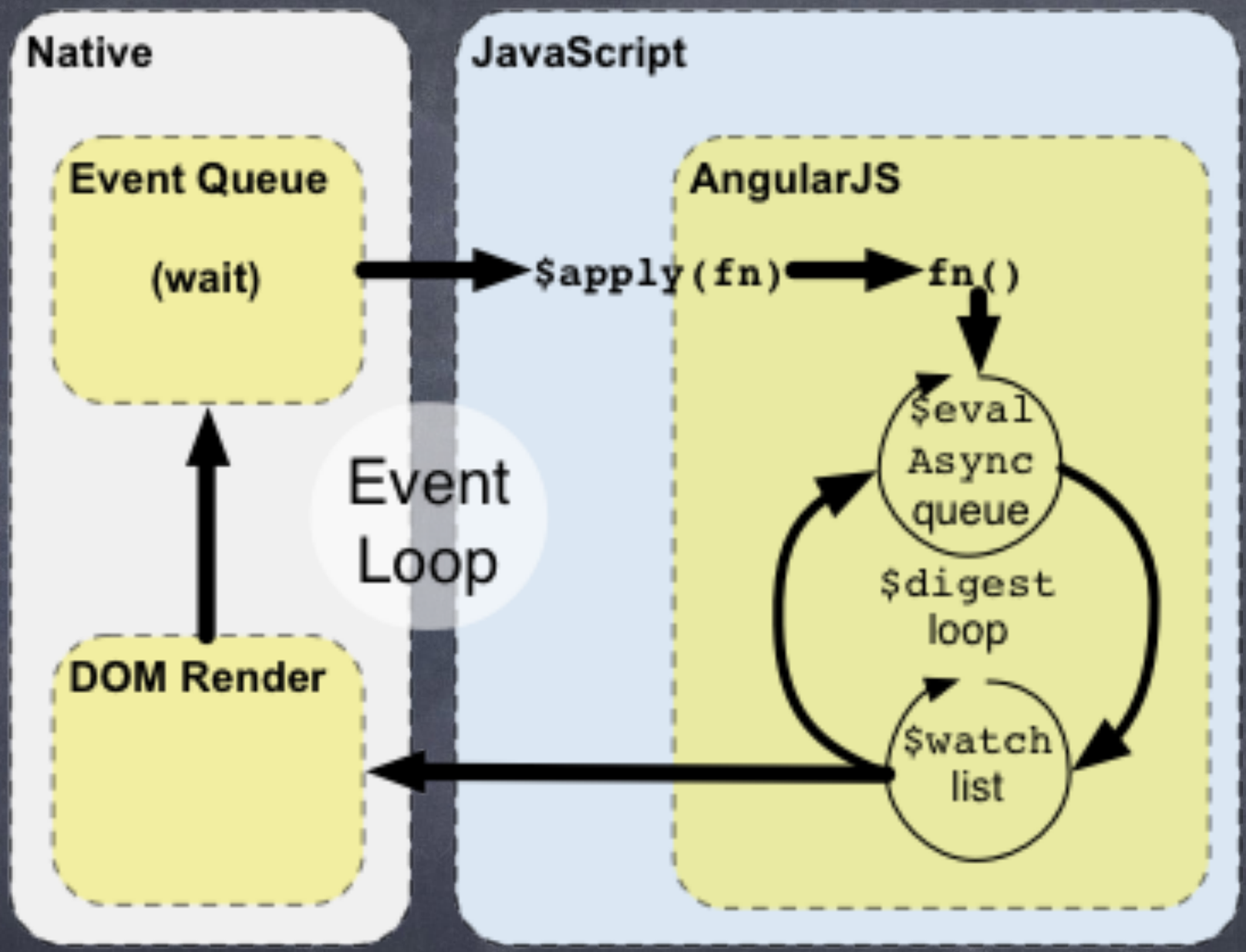
AngularJS資料顯示

- {{ 3 * 7 }} 運算元顯示
- {{ 'Hello World' }} 顯示文字
- {{ { 'key': 'value'} }}
- {{ obj = '321' }}
- {{ showObject }} 顯示物件

Lab.2 加入輸入欄位

雙向繫結
two-way binding





- 主要靠 dirty check(digest loop)、ng-model 建成

$scope 介紹

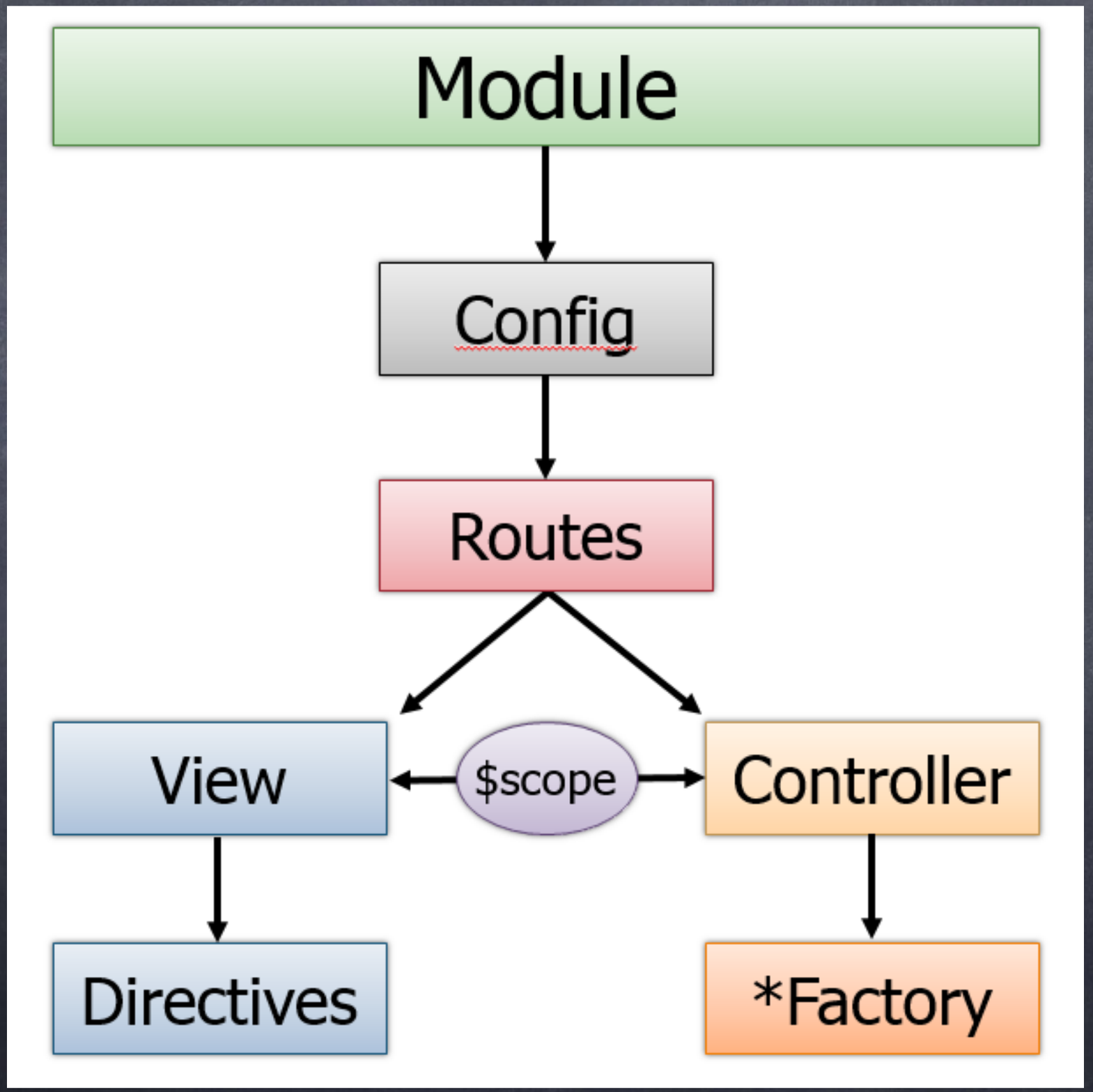
- AngularJS 核心（可以視為model）
- controller 與 view 溝通的介面
- 提供AngularJS觀察資料改變的依據

Lab.3 $scope 介紹

Lab.3-1 $scope 練習

- 設計兩個input欄位，分別可以讓使用者輸入
  first name、last name 及 分別顯示
  first name、last name 及 full name

AngularJS 程式設計架構



未學走先偷跑 $http

- $http({url:, method:, params:})
  .success(function(data) {
  }).error(function(data){ });
- $http.get("url").success(function(data
) { });

Lab.4 $http 範例

有了資料以後...
ng-repeat 介紹

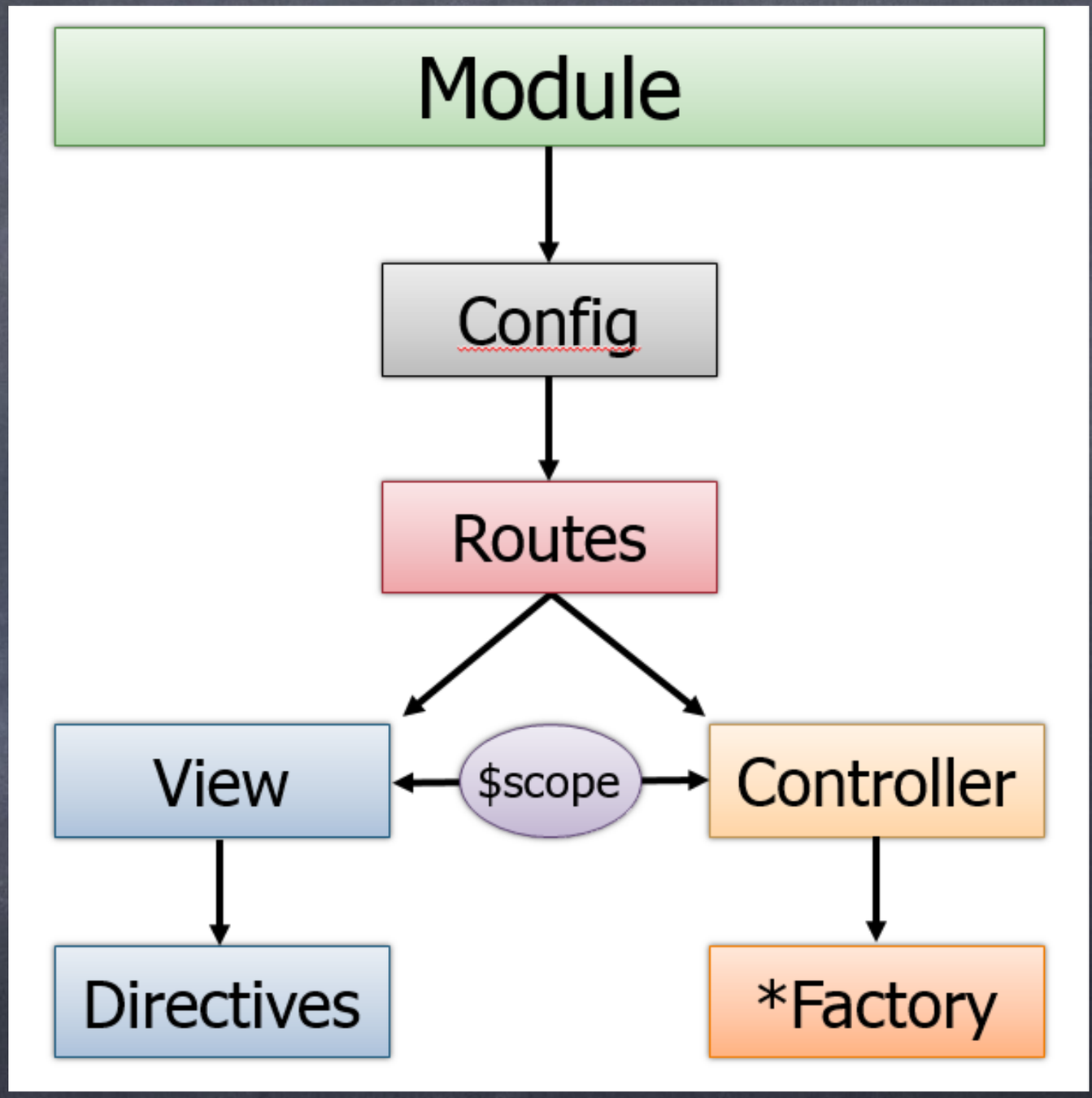
- ng-repeat 屬於View(UI)使用的顯示方式
- 可以讓你快速的輸出重複的資料
- 語法： ng-repeat="obj in obj"

Lab.5 ng-repeat 範例

Lab.5-1 ng-repeat 範例

- 將 lab4 抓回來的資料以grid or table 方式透過ng-repeat條列顯示在畫面上

UI 與 controller



controller 如何抓取 view 的事件

model 設計原則

- 以物件為單位

```
person = {};
person.name = "Chih-Han";
person.age = "20";
person.email = "andyliu@sinica"
```

Javascript 陣列操作

- 增加陣列元素：objs.push(obj)
- 陣列中找到某元素 objs.indexOf(obj)
- 在陣列中移除元素 objs.slice(idx, 1)