中華電信學院
Ruby on Rails

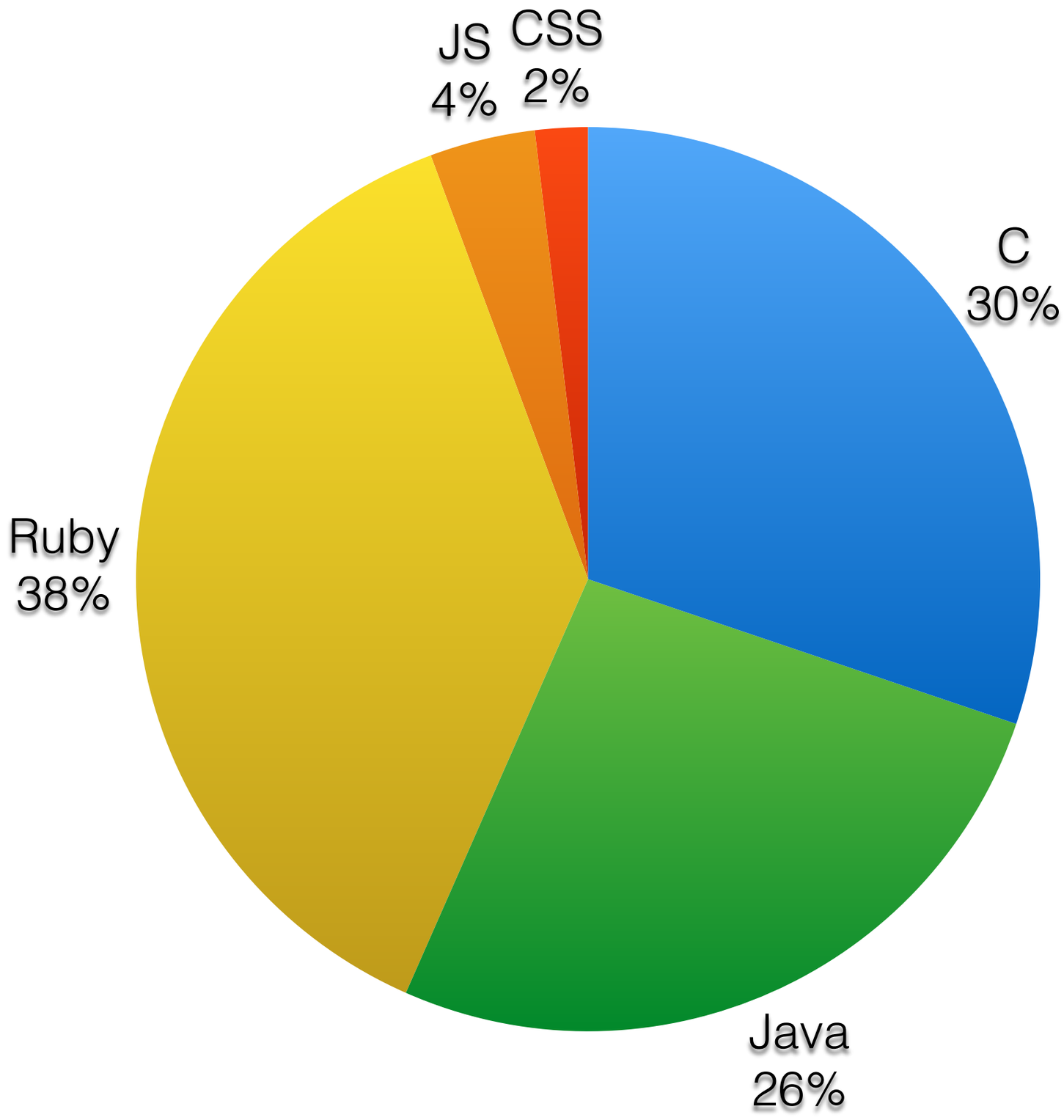
Ruby on Rails 概論

簡煒航（大兜）@tonytonyjan

簡煒航（大兜）
@tonytonyjan

- 腦袋有動工作室創辦人
- 五倍紅寶石創辦人

- Rails Girls 教練
- Yahoo Hack Day 2013 冠軍
- Ruby Kaigi 2014 講者
- ConFoo 2015 講者
- 雙鍵盤手


JS
4%
CSS
2%
C
30%
Ruby
38%
Java
26%

背景知識

前端知識

- HTML
- CSS
- JS

後端知識

- CLI
- Ruby
- RMDBS

網路知識

- HTTP
- Server
- Process
- Port

Q. 這網址代表什麼？

http://localhost:3000

Q. 用 SQL 找出 Tony 所有的文章

posts

id	title	user_id
1	Lorem 1	2
2	Lorem 2	3
3	Lorem 3	3
4	Lorem 4	1

users

id	name
1	John
2	Mary
3	Tony
4	Jason

Q. 瀏覽器載入以下 HTML，一共會送出幾次請求？

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Title</title>
  <link rel="stylesheet" href="/css/main.css">
  <script src="/js/main.js"></script>
</head>
<body>
  <img src="/img/logo.png" alt="">
</body>
</html>
```

Q. 以下三者的差異？

```ruby
{"name" => "Weihang", "age" => 24 }
{:name  => "Weihang", :age  => 24 }
{name:     "Weihang", age:     24 }
```

Q. 下列三個方法，
分別有幾個參數傳入？

```
before_action :set_post
get :about, :contacts, :faq, :sitemap, controller: :pages
resources :posts, only: [:index, :create, :update]
```

Ruby on Rails

Ruby 不是 Rails

Rails 不是 Ruby

Ruby Rails 比一比

	Ruby	Rails
編寫語言	C 語言	Ruby
作者	松本行弘	David Heinemeier Hansson
生日	1995	2004
角色	程式語言	網站框架

松本行弘

Ruby 之父

David Heinemeier Hansson

Rails 之父

錯誤示範

- Rails 開發者：「什麼是 Ruby？」
- Rails 裡面有沒有類似 PHP 的函式？
- Ruby 是參考 Rails 設計的嗎？
- Rails 跟 Python 比起來哪個好？
- Rails 的 @posts 前面的 @ 是什麼？

請不要寫出這種標題

「 Ruby on Rails - 深度剖析 Blocks、Procs」

Rails 設計哲學

Convention over Configuration
慣例優於設定

Don’t Repeat Yourself

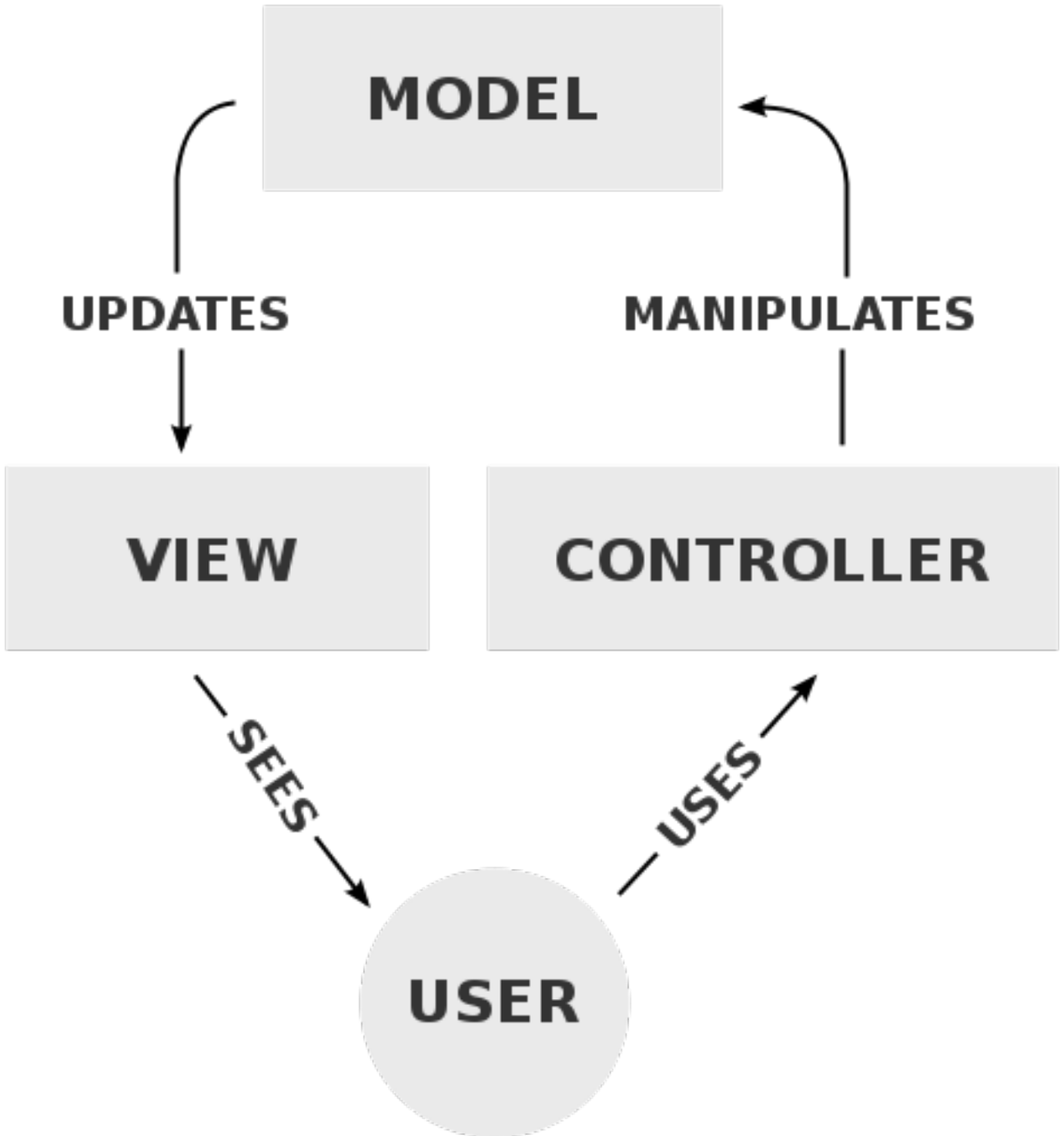
不要重複你自己

Rails 設計模式

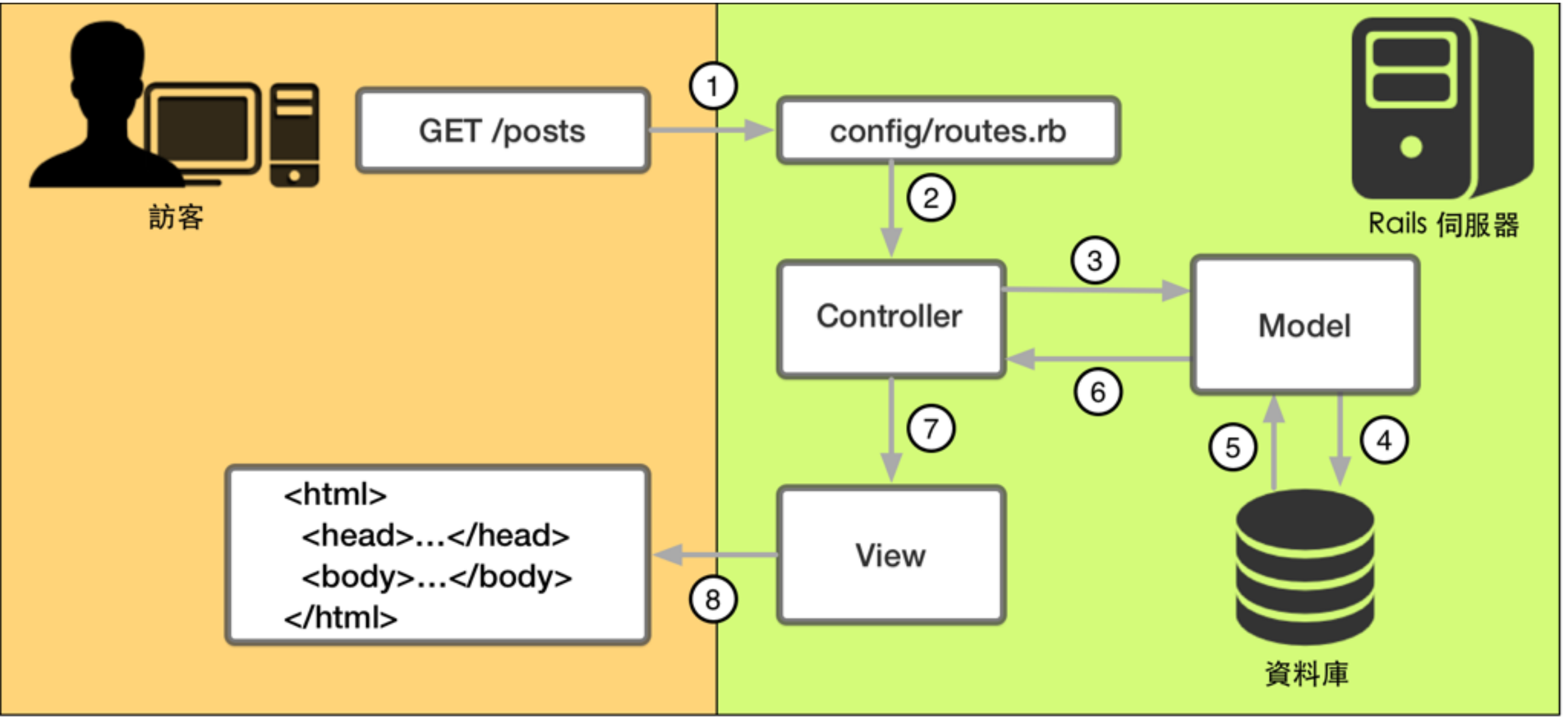
MVC

Model-View-Controller

模型、外觀、控制器


MODEL
UPDATES
MANIPULATES
VIEW
CONTROLLER
SEES
USES
USER

網站 + MVC ?


訪客
GET /posts
1
config/routes.rb
2
Rails 伺服器
Controller
3
Model
6
7
5
4
<html>
<head>...</head>
<body>...</body>
</html>
8
View
資料庫

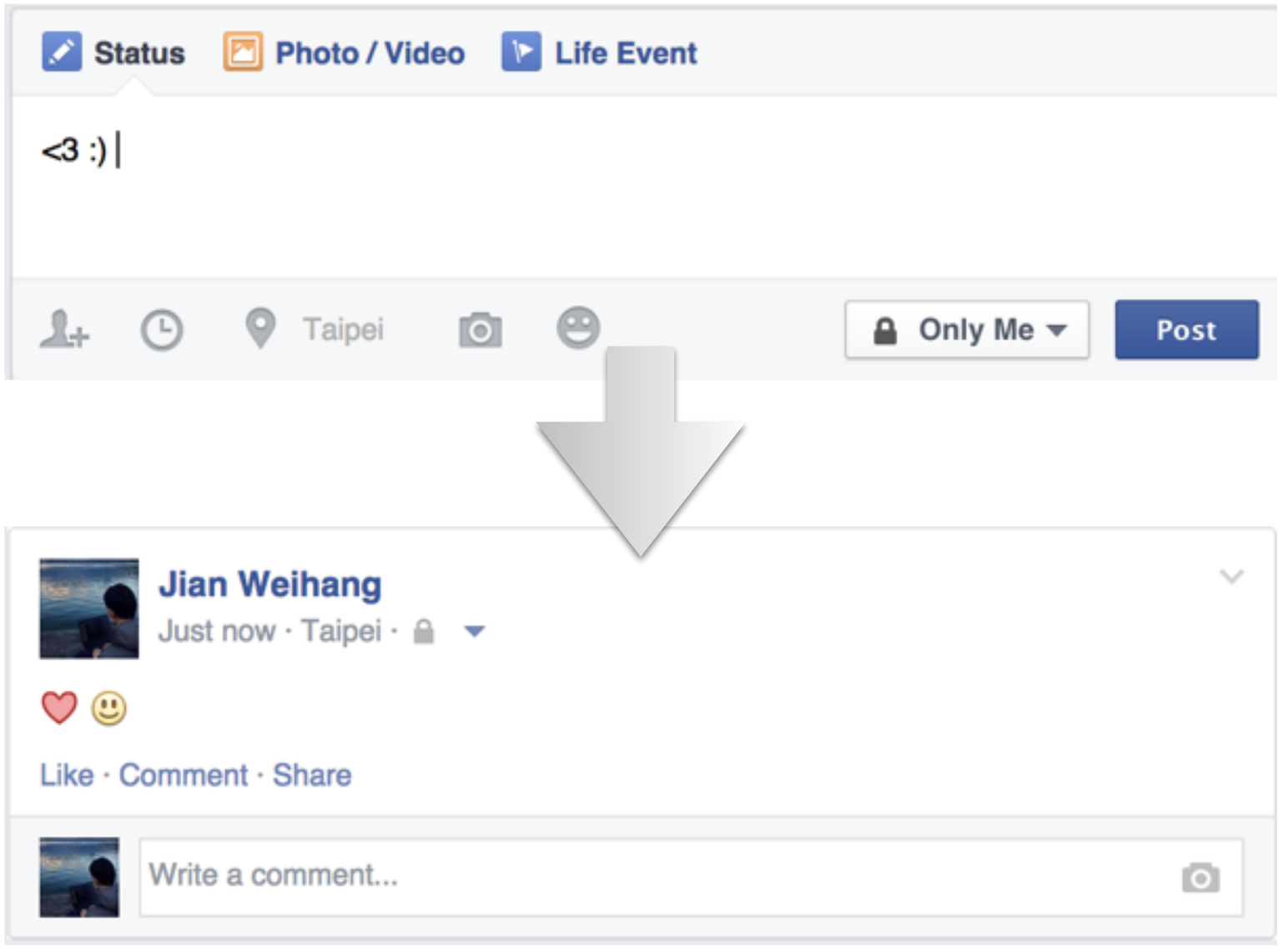
HTML 與 HTTP

上網大學問

HTML

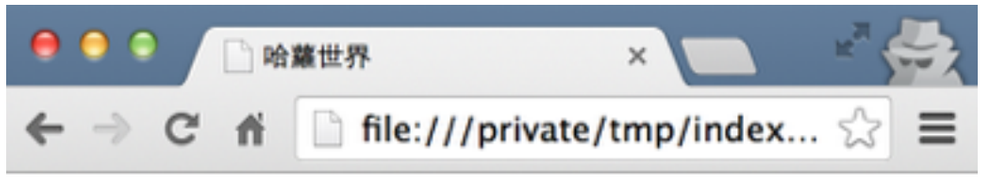
HyperText Markup Language
超文本標記語言

日常中的標記語言


Status
Photo / Video
Life Event
<3 :)
Taipei
Only Me
Post
Jian Weihang
Just now · Taipei ·
Like · Comment · Share
Write a comment...

超文本標記語言 (1/2)

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="UTF-8">
5    <title>哈蘿世界</title>
6  </head>
7  <body>
8    <h1>哈蘿世界</h1>
9    <p>哈蘿世界</p>
10   <img src="loli.jpg" width="300">
11 </body>
12 </html>
```



超文本標記語言 (2/2)

```
44      <div class="row">
45        <div class="col-md-3 col-sm-3 col-xs-3">
46          <a href="/" id="logo">Learn</a>
47        </div>
48        <div class="col-md-9 col-sm-9 col-xs-9">
49        </div>
50      </div>
51    </div>
52  </header>
53      <nav>
54    <div class="container">
55      <div class="row">
56        <div class="col-md-12">
57          <div id="mobnav-btn"></div>
58          <ul class="sf-menu">
59            <li><a href="/">首頁</a></li>
60            <li class="active"><a href="/talks">講座</a></li>
61            <li><a href="/videos">影片</a></li>
62            <li><a href="/showcases">案例</a></li>
63            <li><a href="/speakers">講師</a></li>
64            <li><a href="/posts">部落格</a></li>
65            <li><a href="/faq">常見問題</a></li>
66            <li><a href="/about">關於</a></li>
```

232 行

瀏覽器是一個
超文本排版引擎

HTTP

Hypertext Transfer Protocol
超文本傳輸協定

問答式設計模式
Request-response Pattern

顧客

老闆

老闆，我要牛肚河粉

這是牛肚河粉

老闆，我要魚丸粗麵

沒有魚丸

瀏覽器

伺服器

5xruby.tw

5xruby.tw，我要 /talks/rails-junior 的內容

```
<html>
  <head>…</head>
  <body>…</body>
</html>
```

5xruby.tw，我要 /foo/bar/haha 的內容

404 Not Found

訪客

http://5xruby.tw/talks/rails-junior

伺服器

5xruby.tw

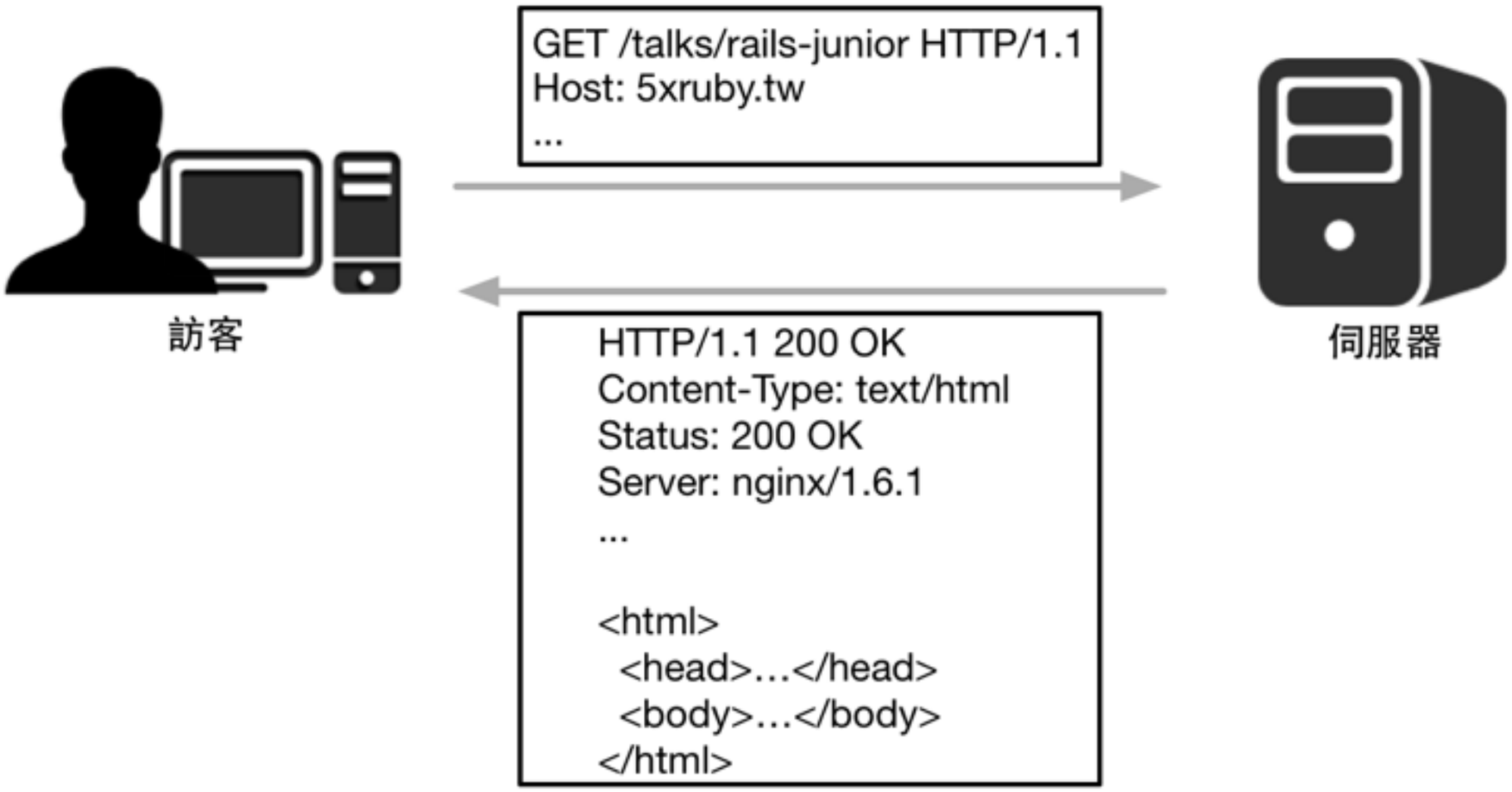
```
<html>
  <head>…</head>
  <body>…</body>
</html>
```

日常上網的流程

瀏覽器發送 HTTP 請求，下載傳回的 HTML 文本，最後渲染該到螢幕上，並等待下一次發送請求。

沒這麼簡單

一份 HTTP 請求
其實包含許多訊息



瀏覽器實際傳送的訊息

一個 HTTP 請求至少包含了「請求動詞」與「請求路徑」

Rails 網址設計慣例

利用了 HTTP 動詞

REST

Representational State Transfer
含狀態傳輸

請求 = 動詞 + 路徑

GET /talks/rails-junior

Q. 以下有什麼不一樣？

```html
<form action="http://www.google.com/search" method="get">
  <input type="text" name="q">
  <input type="submit">
</form>
```

```html
<form action="http://www.google.com/search" method="post">
  <input type="text" name="q">
  <input type="submit">
</form>
```

GET /search

5xruby.tw

POST /search

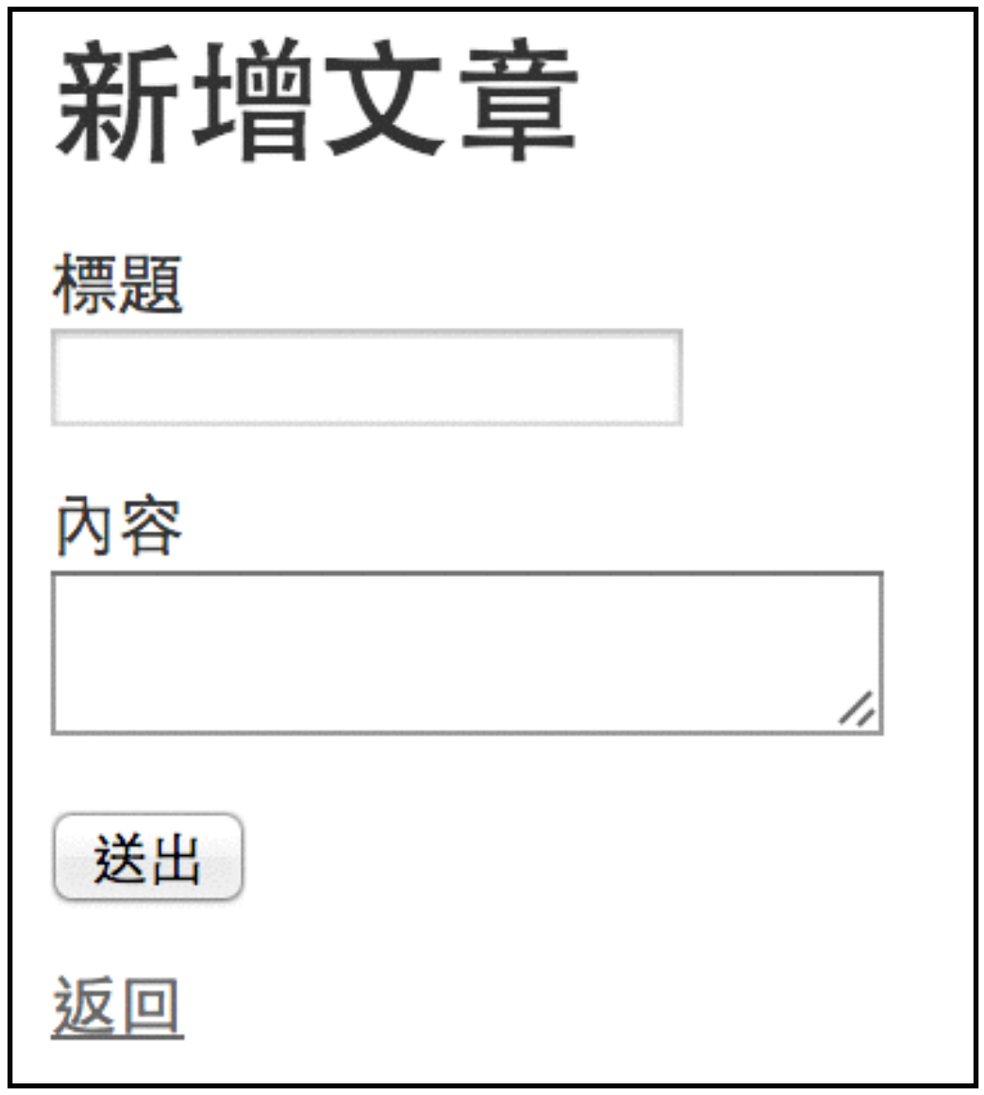
伺服器的工作

隨路徑與動詞而不同

網站架構設計

從設計網址開始

Q. 這一頁的網址是什麼？



Q. action 的網址是什麼？

```
<form action="……" method="post">
  <label for="post_title">標題</label>
  <input id="post_title" name="post[title]" type="text" />
  <label for="post_content">內容</label>
  <textarea id="post_content" name="post[content]"></textarea>
  <input type="submit" value="送出" />
</form>
```

Q. 如果沒有 HTTP 動詞 你會怎麼設計網址？

ex. 首頁	/
ex. 關於我們	/about
顯示所有文章	
新增文章表單頁	
新增文章	
刪除文章	
刪除所有文章	

HTTP 動詞

OPTIONS、 HEAD、 GET、
POST、 PUT、 DELETE、
TRACE、 CONNECT

HTTP 動詞

OPTIONS、 HEAD、 GET、
POST、 PUT、 DELETE、
TRACE、 CONNECT

HTTP 4 大動詞

動詞	用途	例子
GET	用於讀取資料	讀取某個產品介紹的頁面
POST	用於寫入資料	註冊會員，送出表單時
PUT	用於更新資料	更新個人資料，送出表單時
DELETE	用於刪除資料	刪除過去發的留言

REST 鼓勵一址多用

GET /posts	顯示所有文章
GET /posts/new	新增文章表單頁
POST /posts	新增文章
GET /posts/1	瀏覽編號 1 的文章
PUT /posts/1	更新編號 1 的文章
DELETE /posts/1	刪除編號 1 的文章

REST 風格大意是將

CRUD 對應 HTTP 動詞

CRUD
資料最基本的 4 項操作

Create、Read、Update、Delete
新增、檢視、更新、刪除

REST 風格

	CRUD	HTTP 動詞
新增	Create	POST
檢視	Read	GET
更新	Update	PUT
刪除	Delete	DELETE

REST 風格

網址風格	功能	案例
/資源	所有資源	/posts
/資源/動作	對所有資源執行動作	/products/add
/資源/編號	單一資源	/products/9527
/資源/編號/動作	對單一資源執行動作	/users/1/edit

Q. DELETE /posts
此請求的功能應是什麼？

Q. GET /posts/1/delete 這是一個好設計嗎？

第一個 Rails 專案

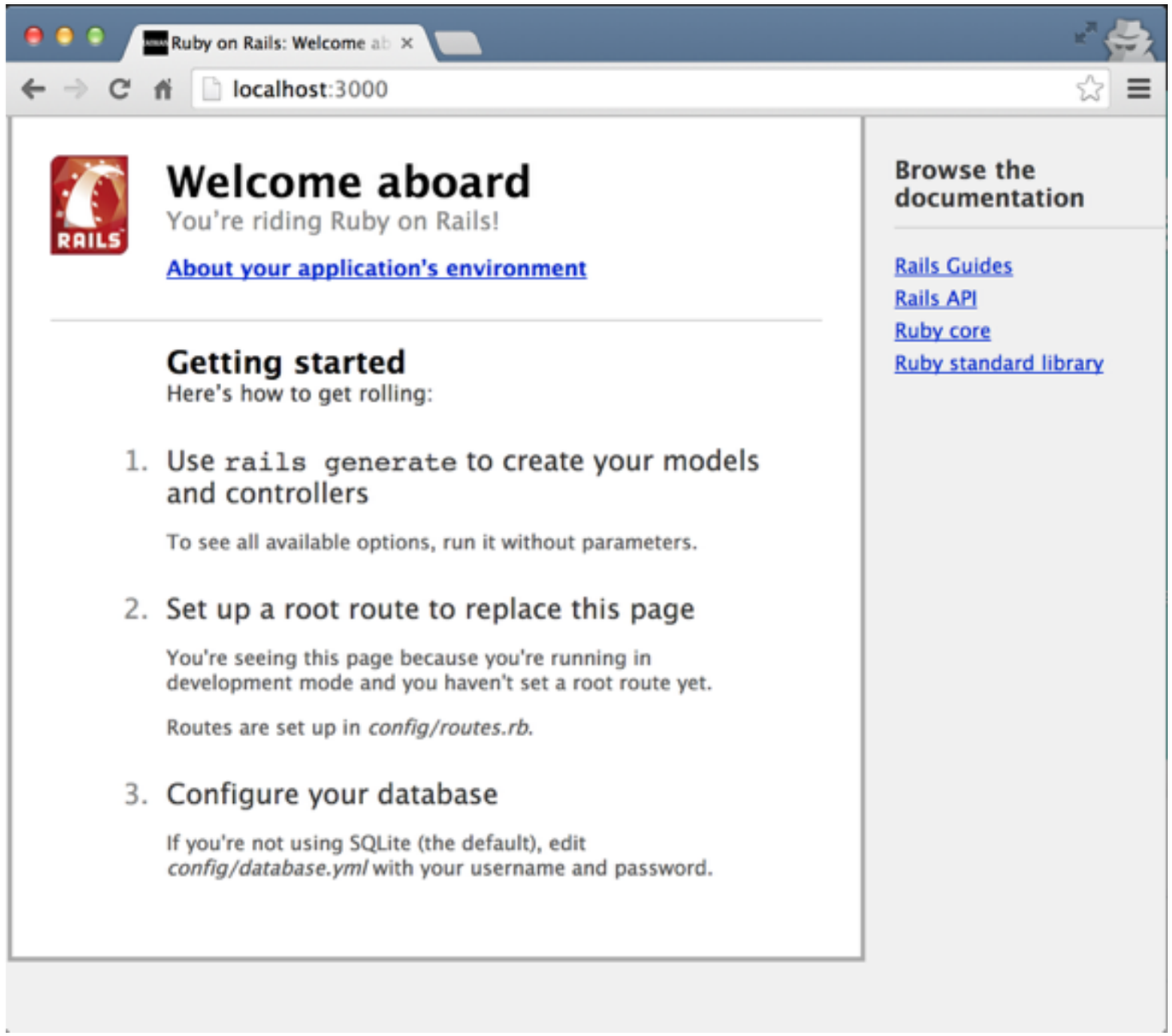
工具準備

鐵道工三器

- 編輯器（Vim、Emacs、Sublime Text、atom）
- 瀏覽器（Chrome、Firefox Developer Edition）
- 終端機（iTerm2）

```
$ rails new hello
$ cd hello
$ rails server
```

http://localhost:3000



$ rails new APP_PATH

- 在 APP_PATH 目錄下初始化 Rails 專案
- 執行 `bundle install` 安裝各種 Rails 需要的 Gem

目錄結構（1/2）

app	網站主要內容，在開發過程中最常訪問的資料夾，MVC 的設計也在這個資料夾內。
config	Rails 的設定檔，包括資料庫、網址路由等，第三方 Gem 的設定檔也會在這。
db	存放資料庫 schema 與遷移檔（migration file）
lib	存放用於擴展你的網站的程式碼，或是可獨立於網站外的模組化程式碼。
vendor	存放第三方程式碼。
public	存放靜態檔案，例如 404 頁面、favicon、robots.txt 與壓縮過的 JS 與 CSS 檔。

目錄結構（2/2）

log	網站的紀錄檔案。
config.ru	Rack 的設定檔。
test	各種網站的單元測試
tmp	暫存檔，例如 pid、session、cache 等。

$ rails server

- 可簡寫為 `rails s`
- 在本機啟動 HTTP 伺服器
- 預設閘道為 3000
- 指定閘道：`rails s -p 4000`

伺服器紀錄

```
hello $ rails s
=> Booting WEBrick
=> Rails 4.1.6 application starting in development on http://0.0.0.0:3000
=> Run `rails server -h` for more startup options
=> Notice: server is listening on all interfaces (0.0.0.0). Consider using
=> Ctrl-C to shutdown server
[2014-10-13 23:25:28] INFO  WEBrick 1.3.1
[2014-10-13 23:25:28] INFO  ruby 2.1.3 (2014-09-19) [x86_64-darwin13.0]
[2014-10-13 23:25:28] INFO  WEBrick::HTTPServer#start: pid=8141 port=3000
```

伺服器紀錄

```
Started GET "/" for 127.0.0.1 at 2014-10-13 23:25:30 +0800
Processing by Rails::WelcomeController#index as HTML
  Rendered /Users/tonytonyjan/.rbenv/versions/2.1.3/lib/ruby/g
Completed 200 OK in 17ms (Views: 8.7ms | ActiveRecord: 0.0ms)
```

新增頁面

app/views/pages/home.html

```
<!-- app/views/pages/home.html -->
<h1>歡迎來到首頁</h1>
```

app/views/pages/about.html

```
<!-- app/views/pages/about.html -->
<h1>關於本站</h1>
```

app/controllers/pages_controller.rb

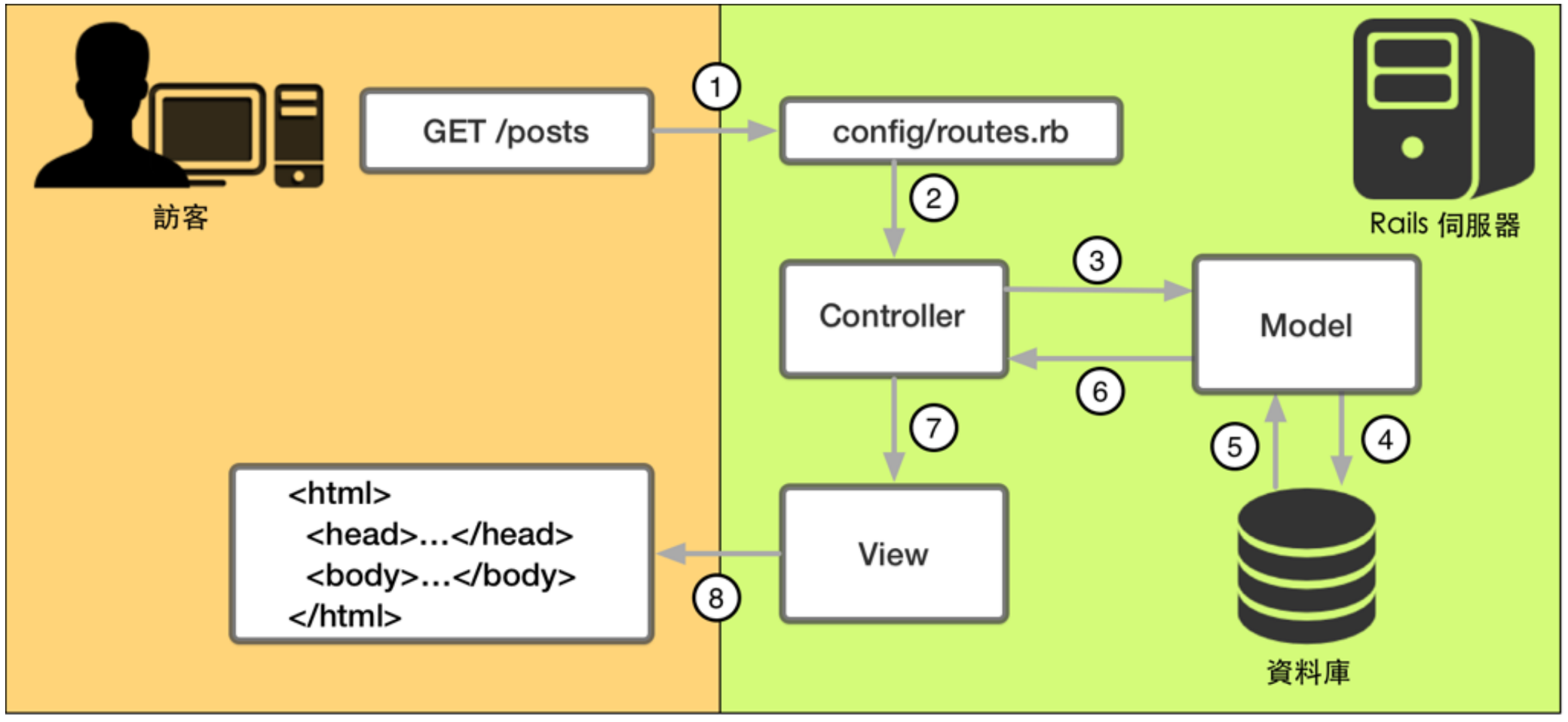
```ruby
# app/controllers/pages_controller.rb
class PagesController < ApplicationController
  def home
    render 'pages/home'
  end

  def about
    render 'pages/about'
  end
end
```

app/config/routes.rb

```ruby
# config/routes.rb
Rails.application.routes.draw do
  get 'home', to: 'pages#home'
  get 'about', to: 'pages#about'
end
```

http://localhost:3000/home

一個 HTTP 請求到回應在 Rails MVC 的過程（1/2）



一個 HTTP 請求到回應在 Rails MVC 的過程（2/2）

1. 瀏覽器以 GET 方法對 localhost:3000 送出請求，路徑是 /home。

2. 伺服器收到了新的請求，首先查看 routes.rb 是否有定義 GET /home 這個路徑，並將此請求交給相對的 controller 處理。

3. 依照 config/routes.rb 產生的路由表，得知 GET /home 請求應該交給 pages controller 中的 home 方法處理。

4. 執行 PagesController 中的 home 方法。

5. PagesController#home 裡面的 `render 'pages/home'` 表示要渲染的頁面位在 app/views/pages/home.html。

```
Started GET "/home" for 127.0.0.1 at 2014-10-15 23:48:40 +0800
Processing by PagesController#home as HTML
  Rendered pages/home.html within layouts/application (0.4ms)
Completed 200 OK in 376ms (Views: 374.6ms | ActiveRecord: 0.0ms)
```

```
Started GET "/about" for 127.0.0.1 at 2014-10-15 23:50:04 +0800
Processing by PagesController#about as HTML
  Rendered pages/about.html within layouts/application (0.2ms)
Completed 200 OK in 21ms (Views: 20.0ms | ActiveRecord: 0.0ms)
```

網址路由

URL Routing

$ rake routes

```
hello $ rake routes
Prefix Verb URI Pattern                Controller#Action
  home GET  /home(.:format)            pages#home
 about GET  /about(.:format)           pages#about
```

$ rake routes

- 印出全站網址路由表（類似網站地圖）
- 資訊包含 HTTP 動詞、網址路徑、控制器方法

$ rake routes

Prefix	用於提示 URL Helper 的方法名稱。
Verb	收到的請求方法。
URI Pattern	收到的請求路徑，`(.:format)` 可用於判別請求格式，例如 /home.json 或 /home.xml 等。
Controller#Action	井字號左邊是 controller 名稱，右邊是該 controller 的方法名稱。

各種設定方式

```
# config/routes.rb
Rails.application.routes.draw do
  get 'home', controller: :pages, action: :home
  get 'home', controller: :pages
  get 'home' => 'pages#home'
  get home: 'pages#home'
end
```

共用參數

```ruby
# config/routes.rb
Rails.application.routes.draw do
  get 'home', 'about', controller: :pages

  scope controller: :pages do
    get 'home', 'about'
  end
end
```

Q. 修改路由，
將首頁導到 home.html

Q. 實作以下路由

```
hello $ rake routes
  Prefix Verb URI Pattern                Controller#Action
    root GET  /                          pages#home
    home GET  /home(.:format)            pages#home
hometown GET  /hometown(.:format)        pages#home
   about GET  /about(.:format)           pages#about
about_me GET  /about/me(.:format)        pages#about
      me GET  /me(.:format)              pages#about
```

設定渲染檔案

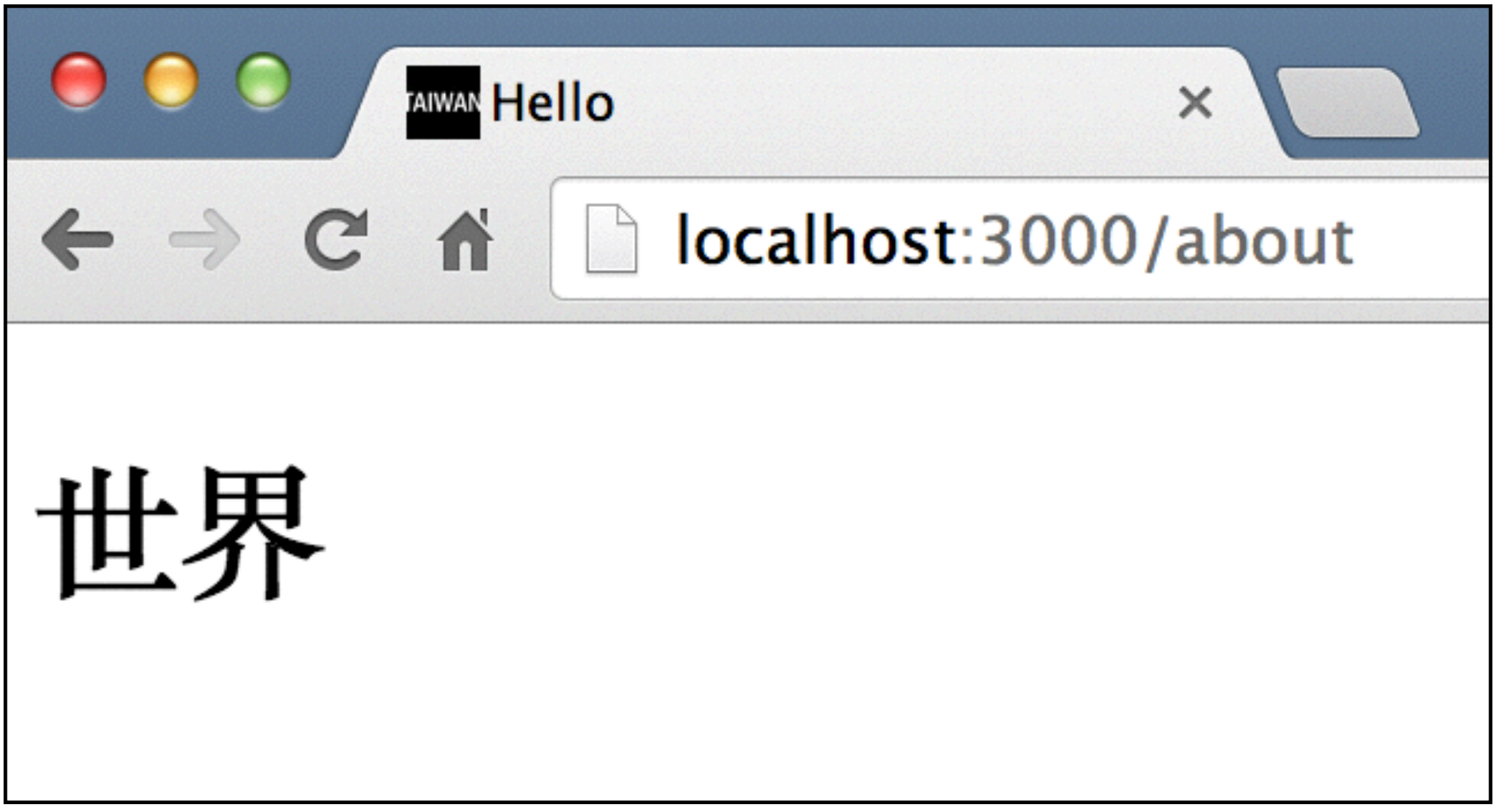
app/views/hello/world.html

```
<!-- app/views/hello/world.html -->
<h1>世界</h1>
```

app/controllers/pages_controller.rb

```
def about
    render 'hello/world'
end
```

http://localhost:3000/about



Q. 新增 app/views/profiles/me.html，使 /about/me 路徑渲染該檔案。

```
Started GET "/about/me" for 127.0.0.1 at 2014-10-16 01:29:39 +0800
Processing by PagesController#about as HTML
  Rendered profiles/me.html within layouts/application (0.3ms)
Completed 200 OK in 26ms (Views: 25.3ms | ActiveRecord: 0.0ms)
```

$ rails generate controller pages home about

$ rails generate controller NAME [action ...]

- `rails generate` 可簡寫為 `rails g`。
- `rails g -h` 可以看到所有的產生器。
- NAME 是 controller 名稱，如果是 pages，則會產生 app/controllers/pages_controller.rb。
- action 是該 controller 的方法名稱。

指令整理

- rails new NAME
- rails server -p PORT
- rails generate controller CONTROLLER ACTION