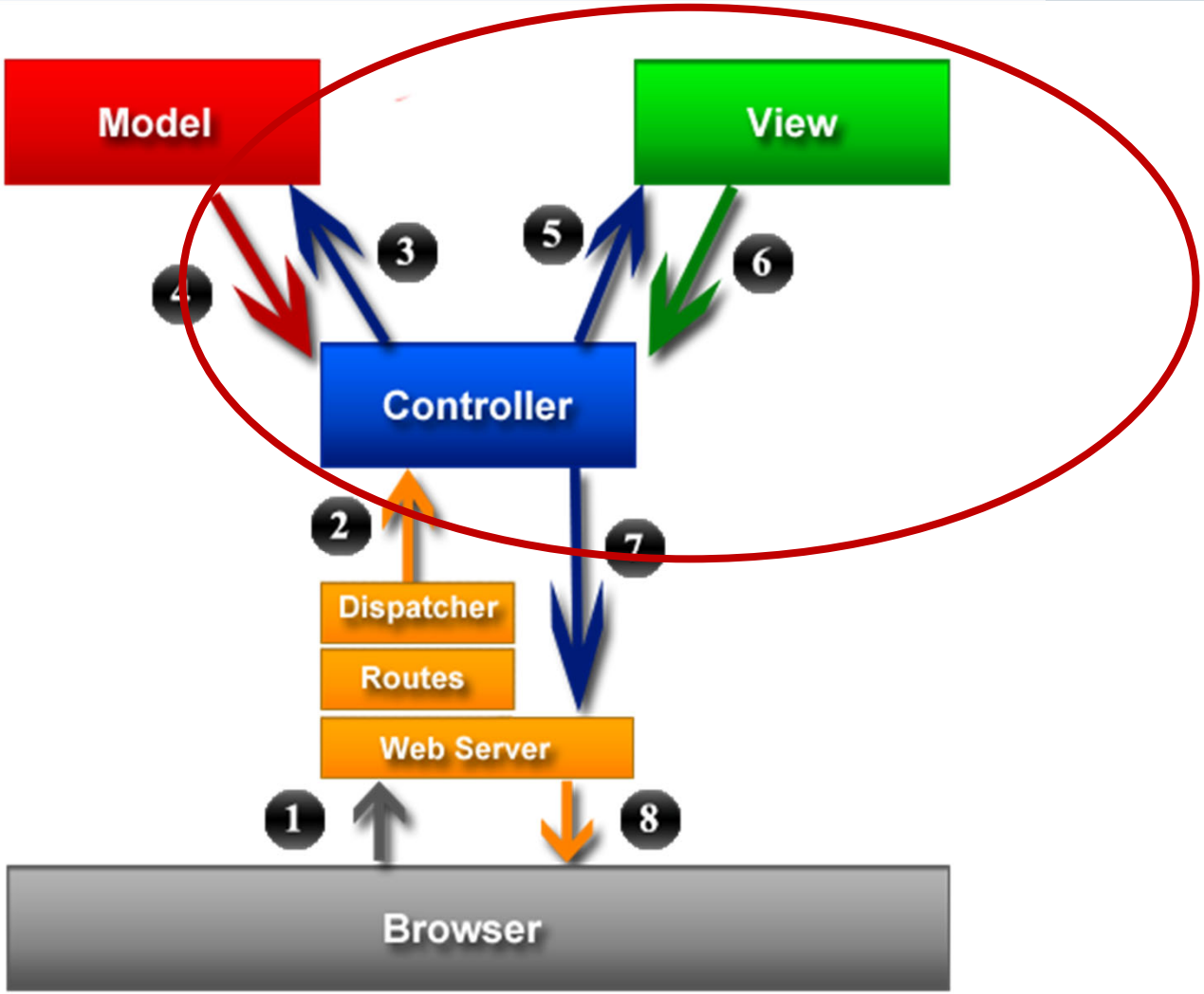
Rails: Views and Controllers

Lecture 31

Recall: Rails Architecture



Wiring Views and Controllers

- A controller is just an ordinary Ruby class
  - Extends `ApplicationController`

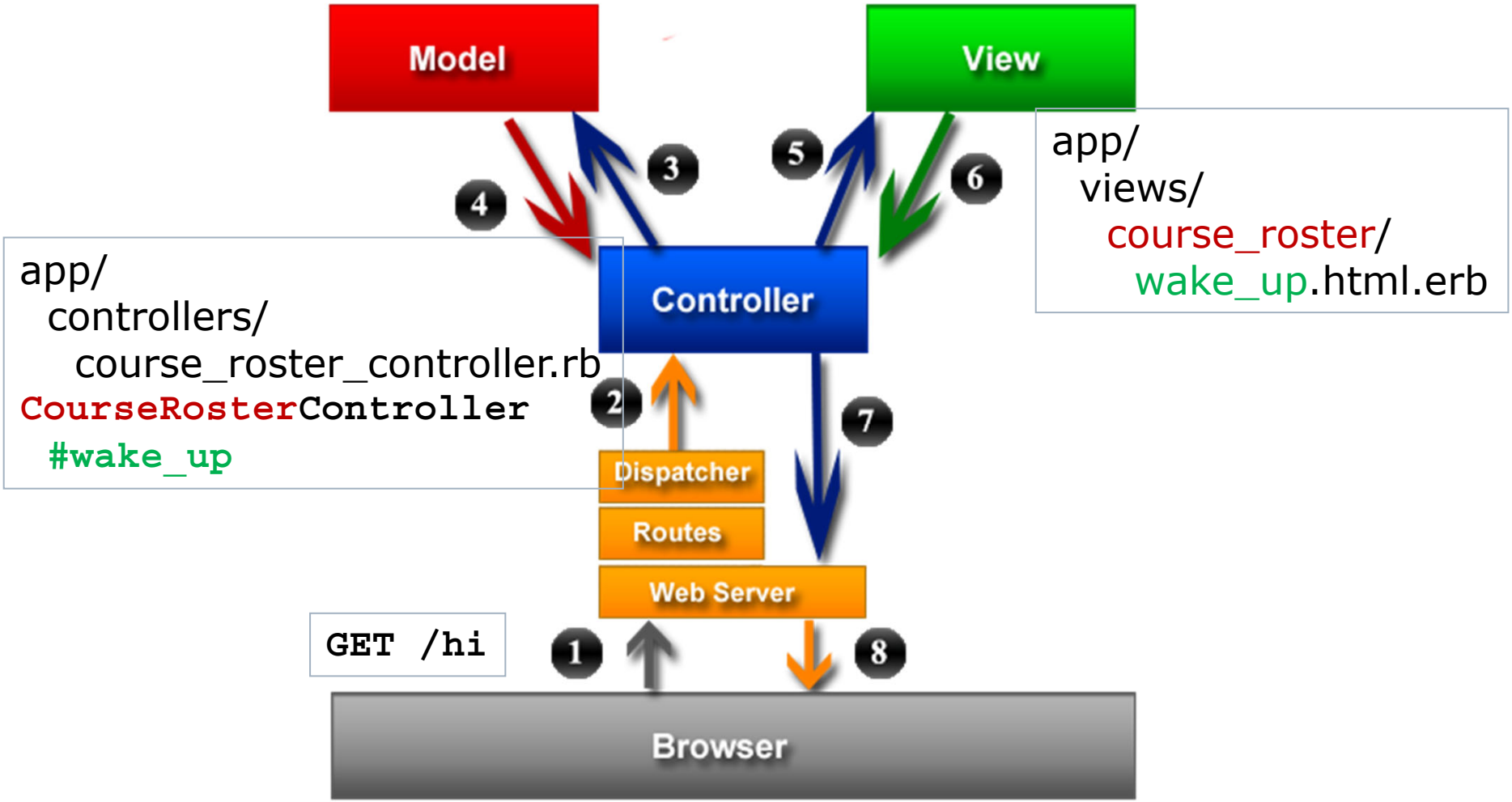
```
class CourseRosterController <
        ApplicationController
```

  - Location: `app/controllers/`
  - Filename: `course_roster_controller.rb`
- Actions are methods in that class

```
def wake_up
...
end
```

- A view is an HTML page (kind of) that corresponds to that action
  - Location: `app/views/course_roster/`
  - Filename: `wake_up.html.erb`
  - Has access to *instance* variables (*e.g.*, `@student`) of corresponding controller!

Recall: Rails Architecture



Demo: Building From Scratch


```
$ rails new demo
```

- Create CourseRosterController
  - Location: app/controllers

```
class CourseRosterController <
                    ApplicationController
```

  - Create (empty) method `wake_up`
- Add route to config/routes.rb

```
get 'hi', to: 'course_roster#wake_up'
```

- Create view (wake_up.html.erb)
  - Location: app/views/course_roster

```
$ rails server
```

Example: Controller


```
# in app/controllers/
# filename course_roster_controller.rb

class CourseRosterController <
        ApplicationController
  def wake_up
    # for this simple eg, no code needed
  end
end
```

Example: Route Definition


```
# in config/
# filename routes.rb

Rails.application.routes.draw do
  get 'hi', to: 'course_roster#wake_up'
# equivalent to (but shorter than):
#   match 'hi', to: 'course_roster#wake_up',
#                via: [:get]
end
```

Example: View


```
<!-- in app/views/course_roster/
     filename wake_up.html.erb -->

<h1>Yo!!</h1>

<p>Are you awake?</p>
```

Single Point of Control

- Notice the duplication in names
- Controller name (course_roster) used in:
  - Name of the controller class
  - Filename of controller class implementation
  - Route
  - Directory name containing views
- Action name (wake_up) used in:
  - Name of the method within controller class
  - Route
  - Filename of view source
- "Solution": generate all these parts

```
$ rails g controller course_roster
wake_up
```

Demo: Generating A Controller


```
$ rails generate controller prof
ask_question visit_office
```

- Results in:
  - Addition of new routes to config/routes.rb

    ```
    get 'prof/ask_question'
    ```
  - Creation of `ProfController` class

    ```
    app/controllers/prof_controller.rb
    ```
  - Definition of methods in `ProfController`

    ```
    def ask_question … end
    def visit_office … end
    ```
  - Creation of 2 views (*i.e.* one per action)

    ```
    app/views/prof/ask_question.html.erb
    app/views/prof/visit_office.html.erb
    ```

```
$ rails server
```

ERb: Embedded Ruby

- General templating mechanism
  - "Template" = a string (usually contents of some file)
  - Contains (escaped) bits of ruby
    - `<% code %>` execute ruby code ("scriplet")
    - `<%= expr %>` replace with result of ruby expr
    - `<%# text %>` ignore (a comment)
- Example: a text file

```
This is some text.
<% 5.times do %>
Current Time is <%= Time.now %>!
<% end %>
```

- Process using erb tool to generate result

```
$ erb example.txt.erb > example.txt
```

- Naming convention: ***filename.outputlang*.erb**
  - Example **index.html.erb**
- Many alternatives, eg HAML

Example: books/index.html.erb


```
<h1>Books</h1>
<table>
  <tr>
    <th>Title</th> <th>Summary</th> <th colspan="3"></th>
  </tr>
<% @books.each do |book| %>
  <tr>
    <td><%= book.title %></td>
    <td><%= book.content %></td>
    <td><%= link_to 'Show', book %></td>
    <td><%= link_to 'Edit', edit_book_path(book) %></td>
    <td><%= link_to 'Destroy', book, method: :delete
                               { confirm: 'Are you sure?' } %></td>
  </tr>
<% end %>
</table>
<br /> <%= link_to 'New book', new_book_path %>
```

Solution: Layouts

- HTML formed from: Layout + Template
  - Layout is the common structure of HTML pages
  - Layout uses `yield` to include (page-specific) template
- File: layout.erb

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title> … etc
  </head>
  <body>
    <%= partial "navigation" %>
    <%= yield %>
    <%= partial "footer" %>
  </body>
<html>
```

- Layout is where you put site-wide styling
  - *e.g.*, navigation bar, div's with CSS classes, footers

Defining and Choosing Layouts

- Default layout for responding to action in **ProfController**
  - app/views/layouts/prof.html.erb
  - If not found, then use app/views/layouts/application.html.erb
- Or controller can explicitly name layout

```
class ProfController < ApplicationController
  layout "people/snazzy"
  # layout "people/snazzy", except: [:show]
```

- There is an application-wide controller that can also specify a fall-back layout

```
class ApplicationController <
                        ActionController::Base
  layout "main"
```

Demo With Parameters

- Pass parameter to action ask_question
  - Add a segment to the route

```
get 'prof/aq/:msg', # or prof/aq/(:msg)
    to: 'prof#ask_question'
```

  - Change ask_question to access params

```
def ask_question
  @q = params[:msg]
end
```

  - Use instance variable in view

```
<p>You said: <%= @q %>!</p>
```

Summary

- View/Controller coupling
  - Location of view from name of controller
  - Filename of view from name of action
  - Controller instance variables available
- ERb
  - Template for generating HTML
  - Scriplets and expressions
  - Other templating approaches exist (eg HAML)
- Layouts and templates