

Rails Views and Layouts

MIS 21

A. Navarro, R. Alampay



INFORMATION SYSTEMS &
COMPUTER SCIENCE

Learning Objectives

- Define view and layout
- Learn how to use Rails built-in helpers
- Include twitter bootstrap in an Application



Controllers

- Controllers are responsible for handling a web requests
- The last responsibility of a controller is to create (or *render*) a response back to the browser
- 3 types of Responses
 - render – a full response to the browser
 - redirect_to – tell the browser to redirect to another URL
 - head – returns just headers instead of a full response
- In most cases, an automatic render is used – the app finds a view that corresponds to the action called (an example of convention over configuration)



Rendering

```
1 ▼ class BooksController < ApplicationController
2   def index
3     #automatically renders app/views/index.html.erb
4   end
5
6 ▼   def show
7     render 'template'
8     #renders a view template.html.erb
9     #found under app/views/books/
10  end
11
12 ▼   def new
13     render inline: '<h1>inline HTML also allowed</h1>'
14     #BAD PRACTICE!
15  end
16 end
```



Layouts

- Layouts allow common parts of a template to be reused
- Under the app/views/layouts folder
 - If a file with the same name as the controller is found, then that is used.
 - Otherwise, a file called application.html.erb is used
 - Alternatively, you can force a layout by adding code to a controller:

```
1 ▼ class BooksController < ApplicationController
2   layout "default"
3
```

- Will look for a file called default.html.erb under app layouts



Layouts

- When Rails renders a response it combines the layout with the associated view for a controller
- `yield` identifies the section where view content will be inserted
- Multiple sections can be inserted using `yield :[section]` and `content_for :[section]`



Helpers

- Helpers – ruby methods that generate HTML code
 - javascript_include_tag
 - stylesheet_link_tag
 - image_tag
 - form_tag and form_for
 - label, text_field, submit, check_box, radio_button, text_area, password_field, hidden_field, search_field, telephone_field, date_field, email_field
- http://guides.rubyonrails.org/form_helpers.html



Including Twitter Bootstrap

- In order to quickly create User interfaces, frameworks such as Twitter Bootstrap may be used
- Steps
 1. Include the gem 'twitter-bootstrap-rails' into your Gemfile then run `bundle install`
 2. Install bootstrap assets by using the generator: `rails generate bootstrap:install static`
 3. Create a layout that uses twitter bootstrap by using the generator: `rails generate bootstrap:layout application`
- <http://github.com/seyhunak/twitter-bootstrap-rails>

