CS142: Section 3

Riding the Rails

The Route to Nirvana

Rails is a **MVC** framework.

Model - Manages behavior and data.
View - Renders the models for the user.
Controller - Receives input, initiates a response.

Project 3 focuses on Views and Controllers.

Getting Started

First: Create a Rails application

```
$ rails new <directory>
```

This will create a **site skeleton**.

That's **a lot of files**!

True, but they are organized, I promise.

Getting Your Bearings

A few directories and what they mean:

`./app` - contains all your application logic

- `./app/controllers - contains controllers`
- `./app/models`
- `./app/views`
  - `./app/views/layouts`

`./config` - application-wide configuration

`./db` - migration and schema information

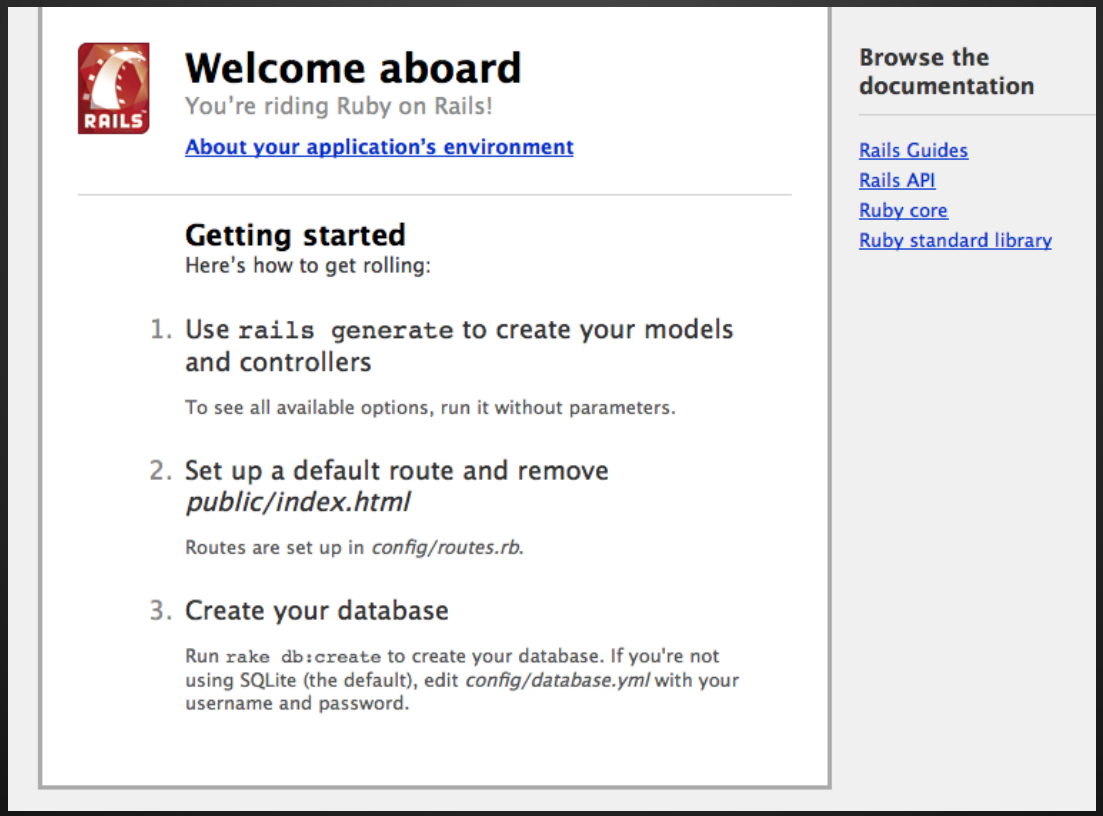
`./public` - directly accessible files

`./vendor` - third-party code libraries

Starting Your App

```
$ rails server
```

http://localhost:3000

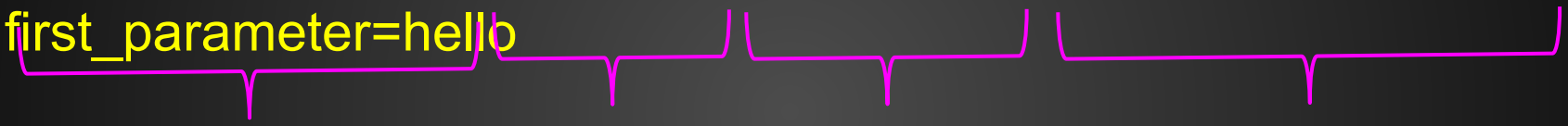


Understanding Routes

RESTful vs. non-RESTful routing

See project 3 instructions

http://localhost:3000/my_class/my_method?
first_parameter=hello



App Name Controller Action Params

my_class: look up class MyClassController defined in `/app/controllers/my_class_controller.rb`.

my_method: execute my_method in MyClassController

In MyClassController, params[:first_parameter] returns "hello"

Creating Controllers

$rails generate controller my_class

This command will generate a file for your controller and a directory for your view (more on that in later slides).

Customized Controllers

```
class MyClassController < ApplicationController

    def my_method
        ...
        ...
    end

end
```

Now what???
Find the view!!
    /apps/views/my_class/my_method.html.erb

Views

Views (also known as templates) in Rails are HTML documents that can be made dynamic through the use of embedded Ruby

```
<ul>
    <% for num in (1..100) %>
        <% if num % 2 == 0 %>
        <li><%= num %></li>
        <% end %>
    <% end %>
</ul>
```

They are located in app/views, and always have the extension .html.erb

Connecting Controllers and Views

http://localhost:3000/my_class/my_method?first_parameter=hello

The default behavior of the “my_method” action of MyClassController is to render whatever is in the file `app/views/my_class/my_method.html.erb`

We can reference MyClassController’s instance variables (e.g. @string) because they are automatically passed into this view

View Helpers

<%= stylesheet_link_tag “my_stylesheet” %>

Creates a <link> tag with a reference to the stylesheet

<%= link_to “ABC”, :action => “my_action” %>

Creates a link with text ABC that references the my_action action in the current controller

Layouts

Layouts are essentially views that wrap other views

Layouts allow you to extract common code between multiple views into a single template

Layouts generally reduce boilerplate in your views (e.g. we should use a layout instead of putting the doctype or stylesheet info in every one of our views)

Layouts are located in app/views/layouts

Layout Example

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>One Controller</title>
  </head>
  <body>
    <%= yield %>
  </body>
</html>
```

Partials

Partials (short for partial templates) provide another way to extract components from a page without code repetition

Think of partials like subroutines – they simplify views via decomposition. If you're writing a Facebook-like news feed, you might want every news item to be a partial.

Partials are like any other view, except that their filenames always begin with an underscore (e.g. _three.html.erb)

Partials are invoked from within another view using render (:partial =>)

Partial Example: passing variables

```
<%= render (:partial => "three",
            :locals => {:foo => "bar"}) %>
```

Rendering partial from view

```
... partial code ...
<div>The foo local is <%= foo %></div>
... partial code ...
```

Partial _three.html.erb