

In this lecture, we will discuss...

- ✧ Brief Rails history
- ✧ Benefits of using Rails
- ✧ Model View Controller



History - Ruby on Rails (RoR)

- ✧ Framework for making **dynamic web applications**
- ✧ Created in 2004 - 2005 by David Heinemeier Hansson



Who is Using Rails?



Why Use Rails?

✧ Convention Over Configuration (COC)

- Less code to write
 - Some code Rails automatically generates for you
 - Oftentimes, there is no need to write code at all
- Learn it once - know what to expect the next time



Why Use Rails?

- ✧ Database Abstraction Layer
 - No need to deal with low-level DB details
 - No more SQL (*Almost*)
 - Important to understand the SQL generated!



Why Use Rails?

- ✧ Agile-friendly
- ✧ Don't Repeat Yourself (DRY) principle
- ✧ Cross-platform
- ✧ Open Source
- ✧ Modular



SQLite

- ❖ Rails uses SQLite for database by default
 - Self-contained, serverless, zero-configuration, transactional, relational SQL database engine.



CLAIM: Most widely deployed SQL database engine in the world



MVC: Model View Controller



- ✧ Invented in 1979 by Trygve Reenskaug
- ✧ Well-established software pattern used by many web and desktop frameworks
- ✧ **Model** - represents the data the application is working with (and possibly business logic)
- ✧ **View** – (visual) representation of that data
- ✧ **Controller** - orchestrates interaction between the model and the view

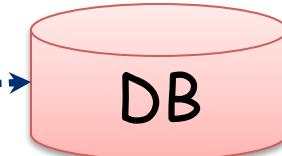


MVC Cycle

1. Request sent
2. Controller \leftrightarrow Model
3. Controller invokes View
4. View renders data



Controller



Summary

- ✧ Rails is very good for **Rapid Prototyping**
- ✧ MVC and Convention over Configuration enable you to
“think less and do more”

What's Next

- ✧ Creating your first Rails app



In this lecture, we will discuss...

- ❖ How to **create and run** your app
- ❖ Directory structure (CoC)
- ❖ Adding **static pages** to your application



Creating First App

```
~$ rails new my_first_app
  create
  create README.rdoc
  create Rakefile
  create config.ru
  create .gitignore
  create Gemfile
  create app
  create app/assets/javascripts/application.js
  create app/assets/stylesheets/application.css
  create app/controllers/application_controller.rb
  create app/helpers/application_helper.rb
  create app/views/layouts/application.html.erb
  create app/assets/images/.keep
  create app/mailers/.keep
  create app/models/.keep
  create app/controllers/concerns/.keep
  create app/models/concerns/.keep
  create bin
  create bin/bundle
  create bin/rails
  create bin/rake
  create bin/setup
  create config
  create config/routes.rb
```

`rails new appname`

(`rails new -h` for more options)



Bundler (gems manager)

```
create test/integration/.keep
create test/test_helper.rb
create tmp/cache
create tmp/cache/assets
create vendor/assets/javascripts
create vendor/assets/javascripts/.keep
create vendor/assets/stylesheets
create vendor/assets/stylesheets/.keep
run bundle install
Fetching gem metadata from https://rubygems.org/.....
Fetching version metadata from https://rubygems.org/...
Fetching dependency metadata from https://rubygems.org/..
Resolving dependencies...
Using rake 10.4.2
Using i18n 0.7.0
Using json 1.8.3
Using minitest 5.8.0
Using thread_safe 0.3.5
Using tzinfo 1.2.2
Using activesupport 4.2.3
```



Version Control Your Rails App!

- ✧ Rails automatically generates `.gitignore`
- ✧ `cd my_first_app`
- ✧ `git init`
- ✧ `git add .`
- ✧ `git commit -m "Initial commit"`

Git repo should be INSIDE

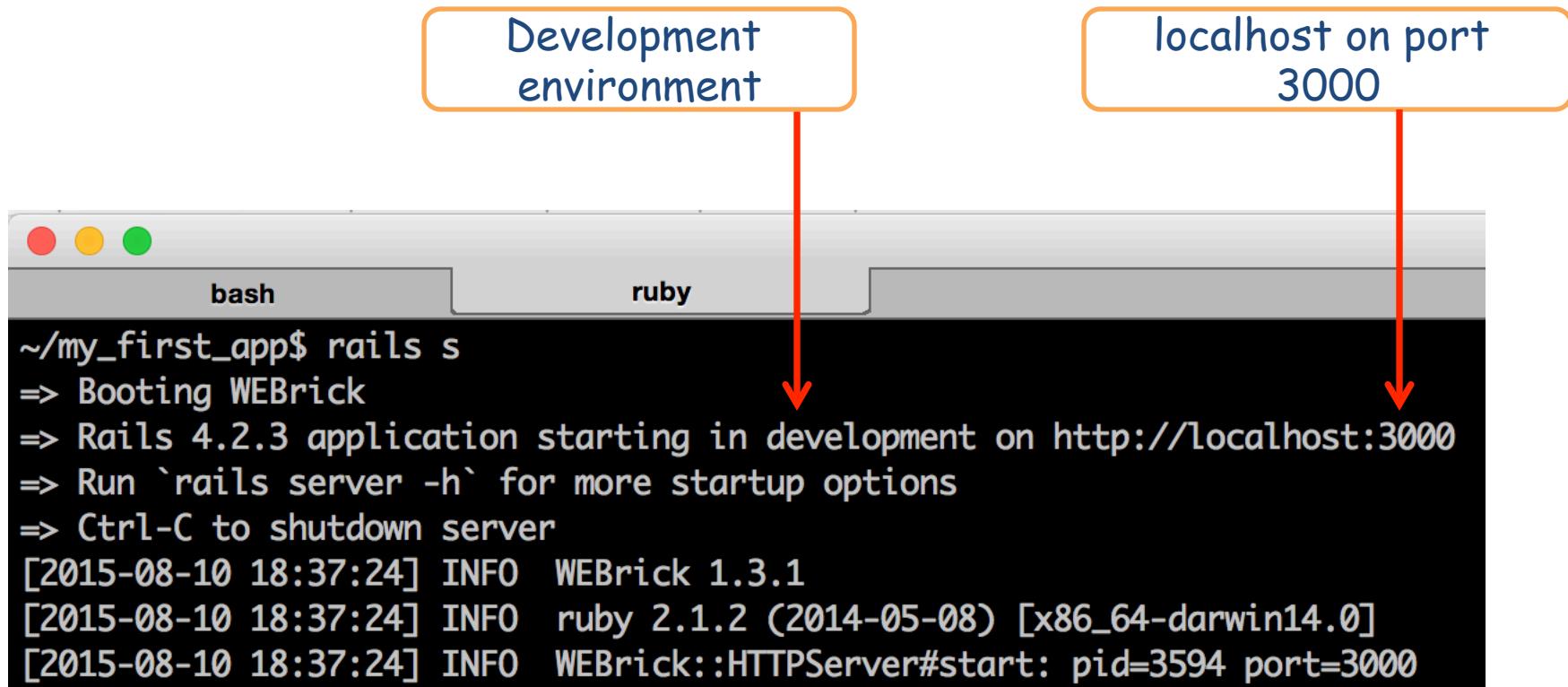


Running the App

- ✧ Now you have `my_first_app` directory with auto-generated structure / (some) code
- ✧ Rails also provides a **built-in web server**
- ✧ Time to run our app!
 - (*Optional*) Open up another terminal window / tab
 - Go into `my_first_app` directory)
 - Run `rails server` (or `rails s`)



Running the App (Continued)



Running the App (Continued)

The screenshot shows a web browser window with the URL `localhost:3000` in the address bar. The page content is as follows:

Welcome aboard
You're riding Ruby on Rails!

[About your application's environment](#)

Getting started
Here's how to get rolling:

1. Use `bin/rails generate` to create your models and controllers
To see all available options, run it without parameters.
2. Set up a root route to replace this page
You're seeing this page because you're running in development mode and you haven't set a root route yet.
Routes are set up in `config/routes.rb`.
3. Configure your database
If you're not using SQLite (the default), edit `config/database.yml` with your username and password.

Browse the documentation

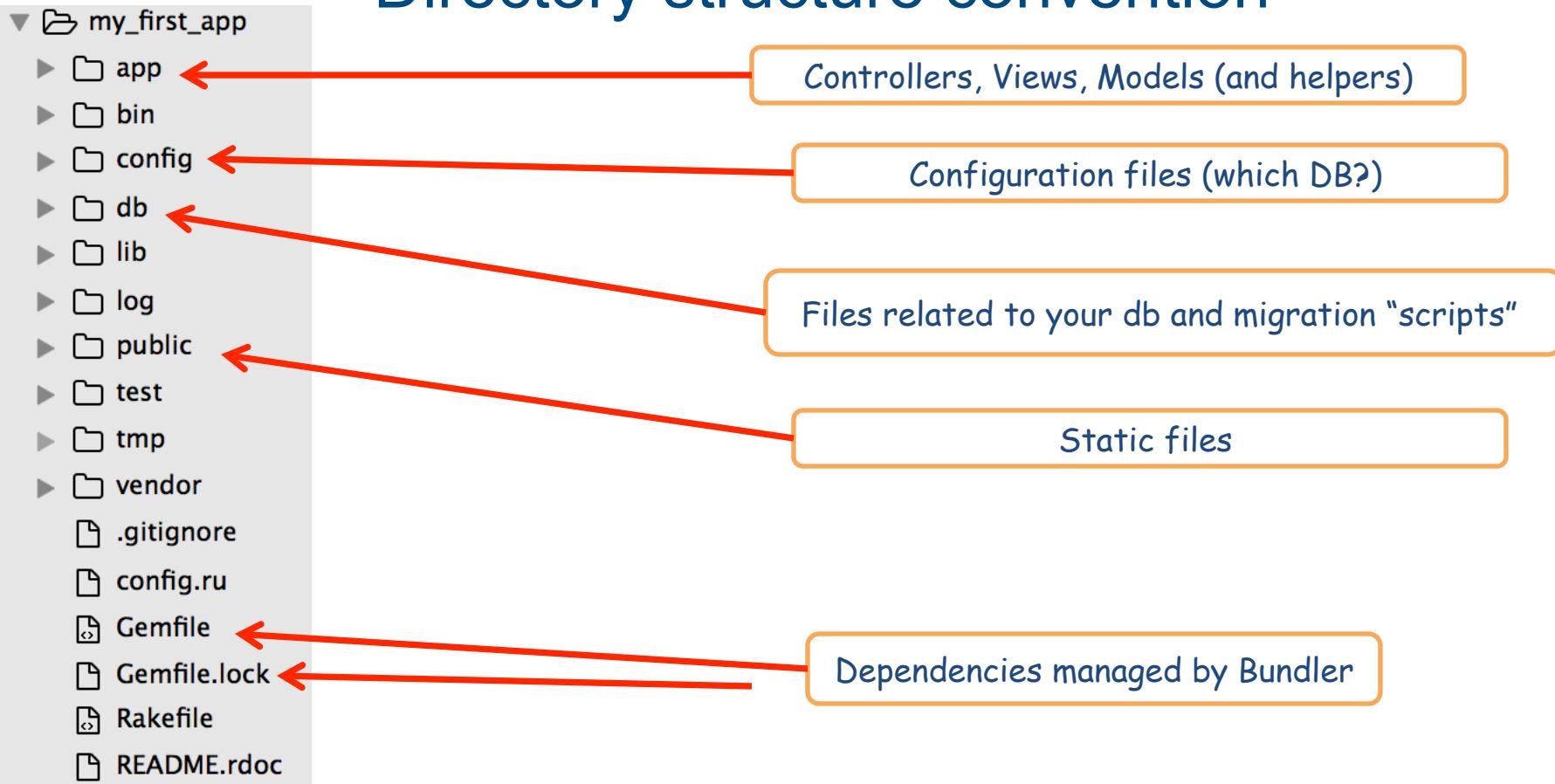
[Rails Guides](#)
[Rails API](#)
[Ruby core](#)
[Ruby standard library](#)

A red arrow points from the text "Useful Resources!" to the "Rails Guides" link in the sidebar.

Useful Resources!



Directory structure convention



public / hello_static.html

- ✧ Server looks into `public` directory **before** looking anywhere else
- ✧ So... if we want to add a **completely static web page** to our application - we can add it under `public` directory



Adding hello_static.html to public dir

FOLDERS

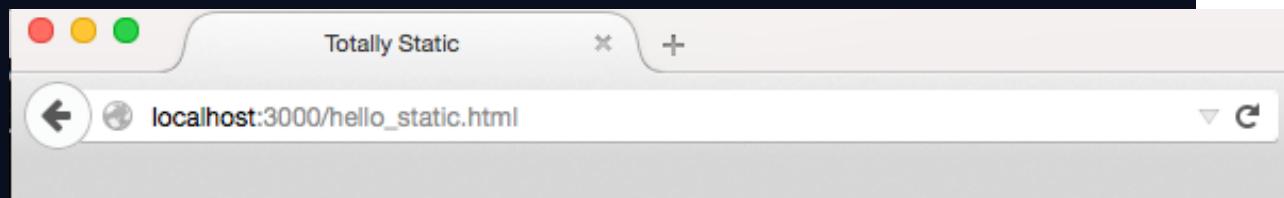
- ▼ my_first_app
 - ▶ app
 - ▶ bin
 - ▶ config
 - ▶ db
 - ▶ lib
 - ▶ log
- ▶ public

- 404.html
- 422.html
- 500.html
- favicon.ico
- hello_static.html

```
◀ ▶    hello_static.html *
```

```
1 <!DOCTYPE html>
2 <html>
3   <head><title>Totally Static</title></head>
4   <body><h2>Will never generate dynamic content :(</h2></body>
5 </html>
```

No need to restart the server - just refresh your browser



Will never generate dynamic content :(



Summary

- ✧ `rails new` creates a new app
- ✧ `rails s` (or `rails server`) runs the app (need to be inside the generated directory)
- ✧ Static pages live inside `public` directory

What's next?

- ✧ Generating Controller and View



In this lecture, we will discuss...

- ❖ How to generate a controller
- ❖ Actions
- ❖ Embedded Ruby (ERB)



Generating a Controller

- ✧ Controllers contain **actions** (Ruby methods) and **orchestrate web requests**
- ✧ Rails can **quickly** generate a controller and **0 or more actions** with their **associated views**
- ✧ `rails generate controller controller_name [action1 action2]`
 - May substitute `g` for `generate`

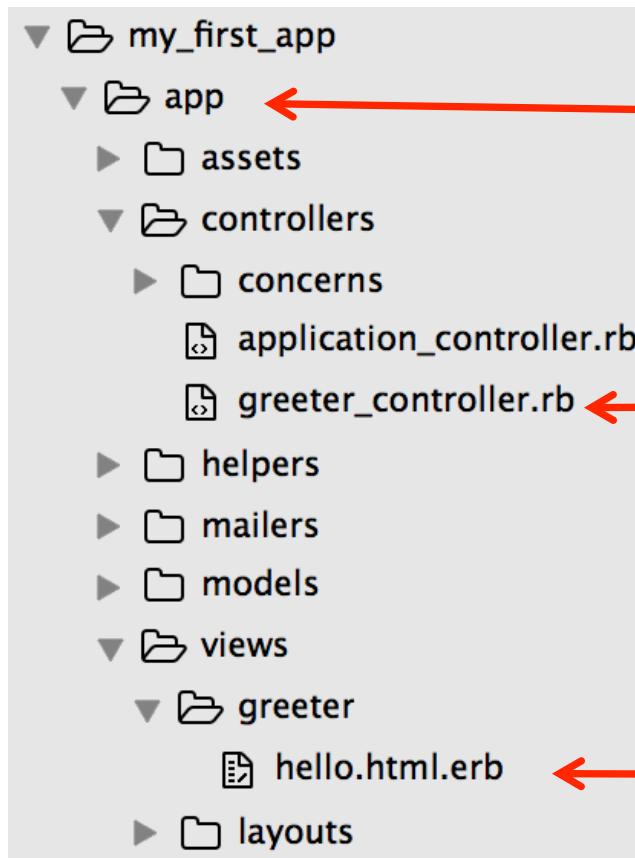


Generating a Controller

```
~/my_first_app$ rails g controller greeter hello
  create  app/controllers/greeter_controller.rb ← Controller
  route   get 'greeter/hello'
  invoke   erb
  create    app/views/greeter
  create    app/views/greeter/hello.html.erb ← View
  invoke   test_unit
  create    test/controllers/greeter_controller_test.rb
  invoke   helper
  create    app/helpers/greeter_helper.rb
  invoke   test_unit
  invoke   assets
  invoke   coffee
  create    app/assets/javascripts/greeter.coffee
  invoke   scss
  create    app/assets/stylesheets/greeter.scss
```



Generating a Controller



... app directory is
where you will be
spending most of
your time ...

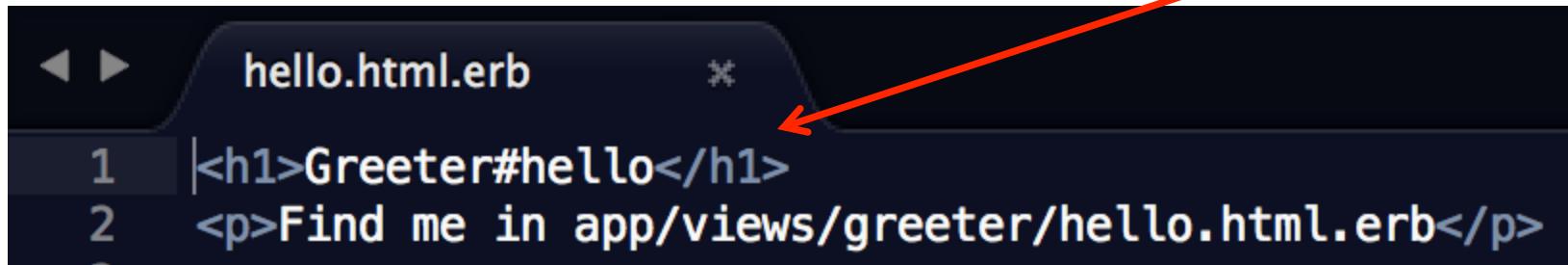
Controller

View



What Does It Look Like? (View)

No DOCTYPE, head or body elements...



```
hello.html.erb
1 <h1>Greeter#hello</h1>
2 <p>Find me in app/views/greeter/hello.html.erb</p>
```



Greeter#hello

Find me in app/views/greeter/hello.html.erb



ERB (Embedded Ruby)

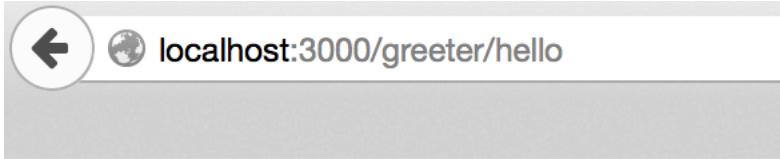
- ❖ The view file was generated and **it looks like it's an HTML file**, but it has an **.erb extension**
- ❖ ERB is a **templating library** (similar to JSP, for example) that lets you **embed Ruby into your HTML**
- ❖ Two **tag patterns** to learn:
 - `<% ...ruby code... %>` - evaluate Ruby code
 - `<%= ...ruby code... %>` - output evaluated Ruby code



New hello.html.erb

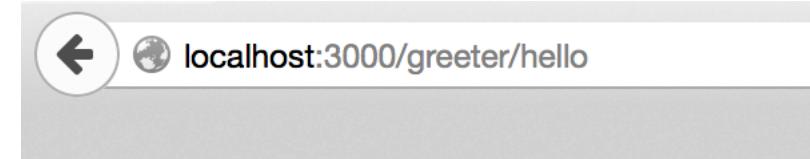
hello.html.erb

```
<% random_names = ["Alex", "Joe", "Michael"] %>
<h1>Greetings, <%= random_names.sample %></h1>
<p>The time now is <%= Time.now %> </p>
```



Greetings, Joe

The time now is 2015-08-12 16:48:57 -0400



Greetings, Alex

The time now is 2015-08-12 16:49:34 -0400



What Does It Look Like? (Controller)

```
greeter_controller.rb *
```

```
class GreeterController < ApplicationController
  def hello
  end
end
```

- ✧ `hello` action is just a **regular** (empty in this case) Ruby method
- ✧ What if we want to add a `goodbye` action to the `greeter` controller and also add a `goodbye.html.erb` to `app/views/greeter` directory?

Should work... ?



Adding Goodbye Action

FOLDERS

- ▼ my_first_app
 - ▼ app
 - assets
 - ▼ controllers
 - concerns
 - application_controller.rb
 - greeter_controller.rb
 - helpers
 - mailers
 - models
 - ▼ views
 - ▼ greeter
 - goodbye.html.erb
 - hello.html.erb

The code editor displays two files side-by-side. On the left, the file `greeter_controller.rb` contains the following Ruby code:

```
1 class GreeterController < ApplicationController
2   def hello
3   end
4   def goodbye
5   end
6 end
```

On the right, the file `goodbye.html.erb` contains the following ERB template:

```
1 
```



Adding Goodbye Action

The screenshot shows a browser window with the URL `localhost:3000/greeter/goodbye`. The page has a red header bar with the text "Routing Error". Below it, the main content area displays the error message "No route matches [GET] "/greeter/goodbye"" in red. It also shows the Rails root directory: `Rails.root: /Users/kalmanhazins/my_first_app`. There are links for "Application Trace", "Framework Trace", and "Full Trace".

Routes

Routes match in priority from top to bottom

Helper	HTTP Verb	Path	Controller#Action
<u>Path / Url</u>		Path Match	
greeter_hello_path	GET	/greeter/hello(.:format)	greeter#hello



Summary

- ✧ Controllers **contain actions** (methods)
- ✧ ERB allows you to either **evaluate** expression with
`<% expression %>` or **output** an expression
`<%= expression %>`



What's Next?

- ✧ Routes



In this lecture, we will discuss...

- ✧ How routing works
- ✧ Rake
- ✧ How to analyze your current routes

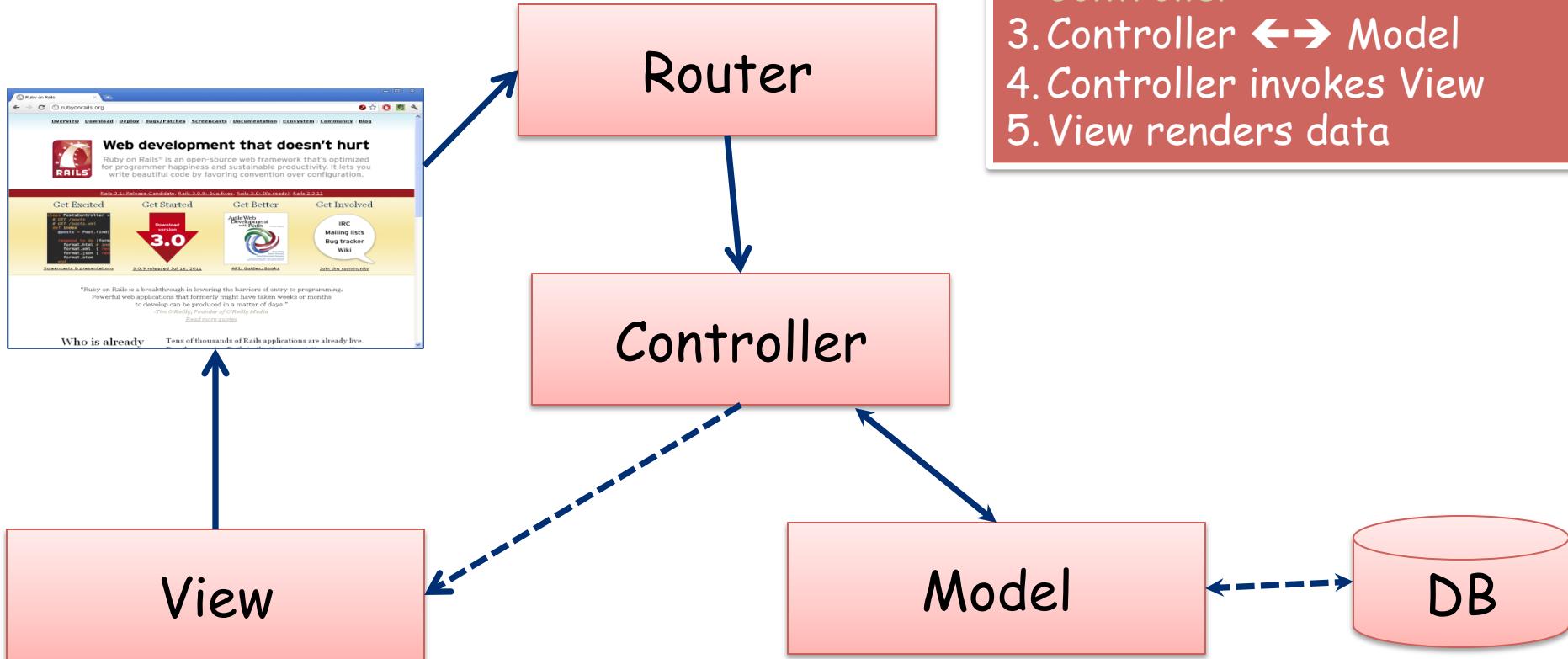


Routes

- ✧ It turns out - **we are missing an important piece**
- Routing**
- ✧ Before the Controller can **orchestrate where the web request goes** - the web request **needs to get routed** to the Controller
- ✧ So, how did the `hello` action work?
- ✧ The route for `hello` action was **automatically generated** (during `rails g controller`)



MVC(R) Cycle



1. Request sent
2. Router routes request to Controller
3. Controller \leftrightarrow Model
4. Controller invokes View
5. View renders data



routes.rb

- ❖ All the routes **need to be specified** (either generated by rails generators or manually) in the `config/routes.rb` file

So, what does `config/routes.rb` look like?



config/routes.rb

FOLDERS

- my_first_app
 - app
 - bin
 - config
 - environments
 - initializers
 - locales
 - application.rb
 - boot.rb
 - database.yml
 - environment.rb
 - routes.rb
 - secrets.yml

routes.rb

```
1 Rails.application.routes.draw do
2   get 'greeter/hello'
3
4   # The priority is based upon order of creation: first created -> highest priority.
5   # See how all your routes lay out with "rake routes".
6
7   # You can have the root of your site routed with "root"
8   # root 'welcome#index'
9
10  # Example of regular route:
11  #   get 'products/:id' => 'catalog#view'
12
13  # Example of named route that can be invoked with purchase_url(id: product.id)
14  #   get 'products/:id/purchase' => 'catalog#purchase', as: :purchase
15
16 end
```

Let's add the route for the goodbye action

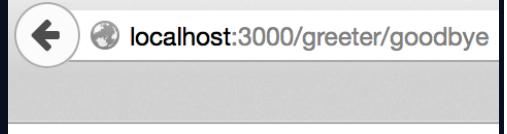


New config/routes.rb

```
routes.rb *  
Rails.application.routes.draw do  
  # get 'greeter/hello'  
  # SAME AS ABOVE  
  get 'greeter/hello' => "greeter#hello"  
  get 'greeter/goodbye'  
  
  # The priority is based upon order of creation: first created -> highest priority.  
  # See how all your routes lay out with "rake routes".  
  
  # You can have the root of your site routed with "root"  
  # root 'welcome#index'  
end
```

Controller

Action



localhost:3000/greeter/goodbye

Bye, Joe



Rake

- ✧ Ruby's build language
 - Ruby's make
 - No XML - written entirely in Ruby
- ✧ Rails uses rake to automate app-related tasks
 - Database, running tests, etc.
- ✧ To see a list of rake tasks for your app:
 - `rake --tasks`



Rake Tasks

```
~/my_first_app$ rake --tasks
rake about                                     # List versions of all Rails framew...
rake assets:clean[keep]                         # Remove old compiled assets
rake assets:clobber                            # Remove compiled assets
rake assets:environment                        # Load asset compile environment
rake assets:precompile                          # Compile all the assets named in c...
rake cache_digests:dependencies                 # Lookup first-level dependencies f...
rake cache_digests:nested_dependencies         # Lookup nested dependencies for TE...
rake db:create                                  # Creates the database from DATABASE...
rake db:drop                                    # Drops the database from DATABASE_...
rake db:fixtures:load                           # Load fixtures into the current en...
rake db:migrate                                 # Migrate the database (options: VE...
rake db:migrate:status                          # Display status of migrations
rake db:rollback                               # Rolls the schema back to the prev...
rake db:schema:cache:clear                     # Clear a db/schema_cache.dump file
rake db:schema:cache:dump                      # Create a db/schema_cache.dump file
rake db:schema:dump                           # Create a db/schema.rb file that i...
```



Individual Rake Task

- ❖ You can **zero-in** on an individual rake task and **what it does** with a **--describe** flag
- ❖ **rake --describe task_name**

```
~/my_first_app$ rake --describe routes
rake routes
    Print out all defined routes in match order, with names. Target specific controller with CONTROLLER=x.
```



Rake Routes

- ✧ `rake routes` explains your **currently defined routes**

```
~/my_first_app$ rake routes
      Prefix Verb URI Pattern          Controller#Action
greeter_hello GET  /greeter/hello(.:format)  greeter#hello
greeter_goodbye GET  /greeter/goodbye(.:format) greeter#goodbye
```



Summary

- ✧ Router **directs the request** to the **right controller**
- ✧ **rake routes** lets you see **which routes are currently defined** for your application

What's Next?

- ✧ Moving business logic out of the View



In this lecture, we will discuss...

- ❖ How to move business logic out of View and into Controller in order to comply with the MVC pattern
- ❖ How long the controller's instance variables stay in place



Actions (Methods) Inside Controller

- ❖ If the action (method) is not really doing anything (i.e., it's empty) - **we can remove it**
- ❖ As long as there is a **proper route defined** and there is a **properly named view file/template**, the action method **does not have to be there** and Rails **will find the correct template** by convention



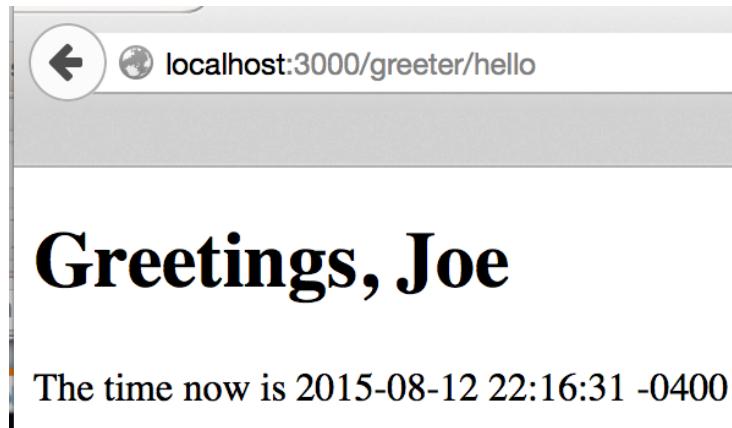
Controller: New Look

FOLDERS

- my_first_app
 - app
 - assets
 - controllers
 - greeter_controller.rb
 - concerns

```
greeter_controller.rb
```

```
1 class GreeterController < ApplicationController
2   # def hello
3   # end
4   # def goodbye
5   # end
6 end
7
8
```



Moving Business Logic Out

- ❖ Our app “works”, but **business logic does not belong** in the **View**. The **View** should have **as little Ruby code logic** as possible

Instead, let's move the logic to the controller!



Moving Business Logic Out

- Instance variables from the controller are available **inside the view**

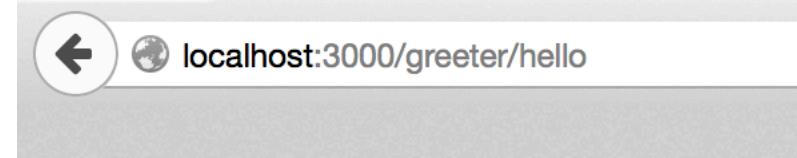
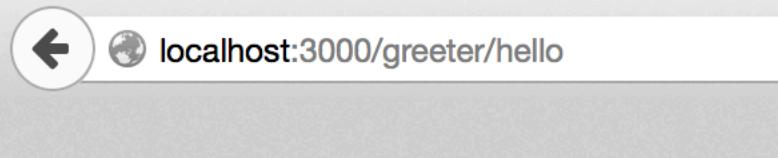
```
class GreeterController < ApplicationController
  def hello
    random_names = ["Alex", "Joe", "Michael"]
    @name = random_names.sample
    @time = Time.now
    @times_displayed ||= 0
    @times_displayed += 1
  end
end
```



New Views/Greeter/hello.html.erb

hello.html.erb *

```
<h1>Greetings, <%= @name %></h1>
<p>The time now is <%= @time %> </p>
<p>This page has been viewed <%= @times_displayed %> time(s)</p>
```



Greetings, Michael

The time now is 2015-08-12 22:40:50 -0400

This page has been viewed 1 time(s)

1 both times?

Greetings, Alex

The time now is 2015-08-12 22:42:11 -0400

This page has been viewed 1 time(s)



Instance Variables in Rails

- ❖ Unlike some other web frameworks (Servlets), you **cannot “store” values in the controller’s instance variables** in between requests
- ❖ Alternatives?
 - Session
 - Database



Summary

- ✧ Keep business logic **out of the view**
- ✧ Instance variables **in the controller** are **available to view**
- ✧ Instance variables **do not stick around** between requests

What's Next?

- ✧ Helpers



In this lecture, we will discuss...

- ❖ What are helpers and why are they useful
- ❖ Why use the `link_to` Rails helper



Helpers

- ❖ We've made the **current time available** through `@time` instance variable (controller)
- ❖ But what if we want to format **how the time looks?**
 - Should that code go in the View? **Non-reusable...**
 - Controller? **Should be “view format” agnostic**

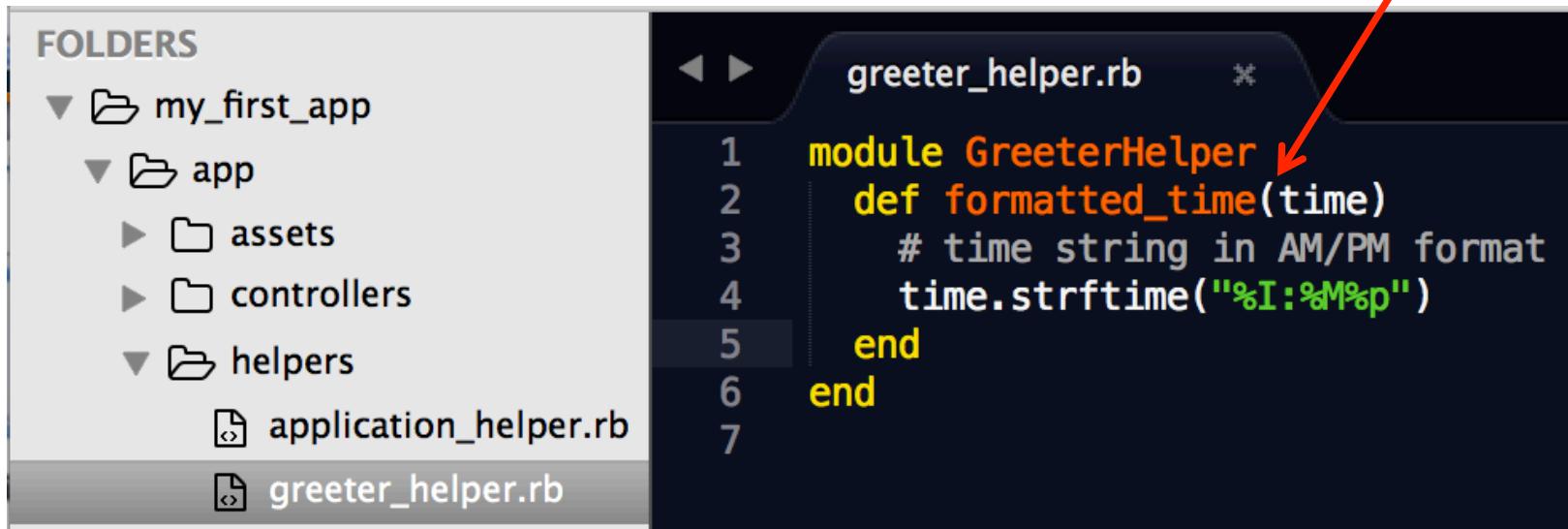
Helpers to the rescue!



Helpers

- ✧ (empty) `greeter_helper.rb` module generated
- ✧ Let's add a helper method

Available in ALL views



FOLDERS

- my_first_app
 - app
 - assets
 - controllers
 - helpers
 - application_helper.rb
 - greeter_helper.rb

```
greeter_helper.rb
1 module GreeterHelper
2   def formatted_time(time)
3     # time string in AM/PM format
4     time.strftime("%I:%M%p")
5   end
6 end
7
```



Helpers: New View

The screenshot shows a file structure on the left and a code editor/browser interface on the right.

FOLDERS:

- my_first_app
 - app
 - assets
 - controllers
 - helpers
 - mailers
 - models
 - views
 - greeter
 - goodbye.html.erb
 - hello.html.erb

Code Editor (hello.html.erb):

```
1 <h1>Greetings, <%= @name %></h1>
2 <p>The time now is <%= formatted_time(@time) %> </p>
3 <p>This page has been viewed <%= @times_displayed %> time(s)</p>
4
```

Browser Preview:

localhost:3000/greeter/hello

Output:

Greetings, Michael

The time now is 05:21PM

This page has been viewed 1 time(s)



Rails Built-In Helpers: link_to

- ❖ Rails provides **many built-in helpers**
- ❖ `link_to name, path`
 - Hyperlink generator **that displays the name** and **links** to the **path**
 - Path could either be a **regular string** or a **route defined in the routes.rb file** ending with `_url` (full path) or `_path` (relative path)
 - `_url` and `_path` used **interchangeably**, but according to the spec full path is **required in cases of redirection**



link_to in action

localhost:3000/greeter/hello

Greetings, Joe

The time now is 05:33PM

This page has been viewed 1 time(s)

[Google](#)

[Goodbye](#)

localhost:3000/greeter/goodbye

```
hello.html.erb
```

```
<h1>Greetings, <%= @name %></h1>
<p>The time now is <%= formatted_time(@time) %> </p>
<p>This page has been viewed <%= @times_displayed %> time(s)</p>

<p><%= link_to "Google", "http://www.google.com" %></p>
<p><%= link_to "Goodbye", greeter_goodbye_path %></p>
```

greeter_goodbye derived from routes.rb
(See "Prefix" column in rake routes)



Summary

- ✧ Helpers are “**macros**” / “**formatters**” for your view
- ✧ When using `link_to`, there is **no need** to change things if a path changes

What's Next?

- ✧ Introduction to HTTParty



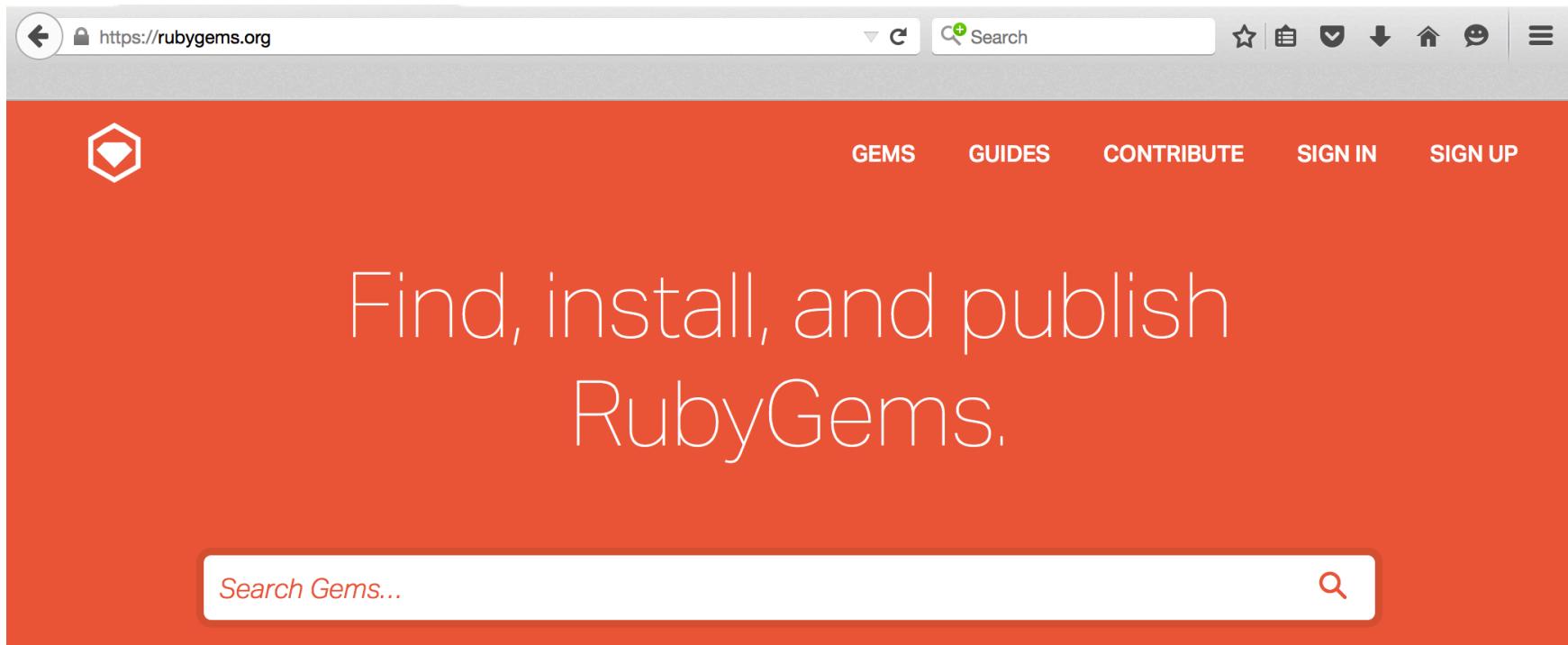
In this lecture, we will discuss...

- ✧ Ruby gems
- ✧ How to use the HTTParty Ruby gem



RubyGems

✧ Many gems (3rd party plugins / additions) for Ruby

A screenshot of the RubyGems.org homepage. The page has a light gray header bar with browser controls, a search bar containing "Search", and a navigation menu with icons for star, list, download, home, message, and more. Below the header is an orange main content area. On the left side of the orange area is a white hexagonal icon containing a blue diamond symbol. To the right of the icon are five white links: GEMS, GUIDES, CONTRIBUTE, SIGN IN, and SIGN UP. In the center of the orange area, there is large white text that reads "Find, install, and publish RubyGems.". At the bottom of the orange area is a white search bar with the placeholder text "Search Gems..." and a red magnifying glass icon.

RubyGems Package Manager

```
~$ gem
```

RubyGems is a sophisticated package manager for Ruby. This is a basic help message containing pointers to more information.

Usage:

```
gem -h/--help  
gem -v/--version  
gem command [arguments...] [options...]
```

Examples:

```
gem install rake  
gem list --local  
gem build package.gemspec  
gem help install
```



Installing HTTParty

```
~$ gem list httparty
```

```
*** LOCAL GEMS ***
```

```
~$ gem install httparty
```

```
Fetching: httparty-0.13.5.gem (100%)
```

```
When you HTTParty, you must party hard!
```

```
Successfully installed httparty-0.13.5
```

```
Parsing documentation for httparty-0.13.5
```

```
Installing ri documentation for httparty-0.13.5
```

```
Done installing documentation for httparty after 0 seconds
```

```
1 gem installed
```



More Details on Gem Command

- ❖ Use `-d` option to **get more details**

```
~$ gem list httparty -d
```

```
*** LOCAL GEMS ***
```

```
httparty (0.13.5)
```

Authors: John Nunemaker, Sandro Turriate

Homepage: <http://jnnunemaker.github.com/httparty>

License: MIT

Installed at: /Users/kalmanhazins/.rbenv/versions/2.1.2/lib/ruby/gems/2.1.0

Makes http fun! Also, makes consuming restful web services dead easy.



What Are Restful Web Services?

- ✧ **Simple** web services **implemented using HTTP** (and principles of REST) that:
 1. Have a **base URI**
 2. Support a **data exchange format** like XML or JSON (and possibly others)
 3. Support a **set of HTTP operations** (GET, POST etc.)



HTTParty Gem

- ✧ Restful web services **client**
- ✧ **Automatic parsing of JSON and XML into Ruby hashes**
- ✧ Provides **support** for:
 - Basic http authentication
 - Default request query parameters



Lots of Restful APIs Out There...

www.programmableweb.com/apis

Follow ProgrammableWeb to get API news and alerts as they break +Follow Share Sign In/Sign Up

ProgrammableWeb API News API Directory For API Providers For Developers Listings Forum

MOST POPULAR APIs

1. Facebook	Track this API	6. Pinterest	Track this API
2. Google Maps	Track this API	7. Bloomberg	Track this API
3. Fitbit	Track this API	8. Twitter	Track this API
4. LinkedIn	Track this API	9. FedEx	Track this API
5. TripAdvisor	Track this API	10. Instagram	Track this API

APIs Since 2005

Year	Number of APIs
2005	186
2006	438
2007	865
2008	1263
2009	2026
2010	3422
2011	5018

A smiling man wearing a bicycle helmet, likely the founder or a representative of ProgrammableWeb.

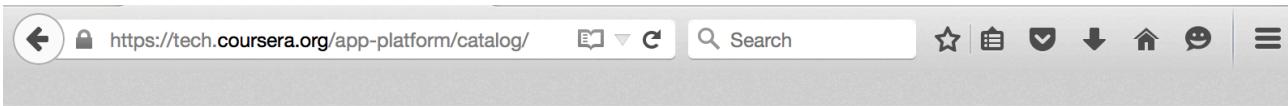


HTTParty Usage

- ✧ `include HTTPParty module`
 - And you are good to go!
- ✧ Can specify:
 - `base_uri` for your **requests**
 - `default_params` (API developer key for example)
 - `format` to tell it **which format the data is in**



Coursera Restful API



Example requests

All of the catalog APIs are consistent and behave similarly. Sending a **GET** to the root of the resource will download the entire collection. To retrieve an individual element, use either a **GET** with an **id** query parameter, or append the id as a path parameter. For example, the following are equivalent:

```
1 curl https://api.coursera.org/api/catalog.v1/courses/2
2 curl https://api.coursera.org/api/catalog.v1/courses?id=2
```

All collections support multi-get with the following syntax:

```
1 curl https://api.coursera.org/api/catalog.v1/courses?ids=2,3
```

By default, only a minimal set of fields are included in response objects. To request more fields, include their names in a **fields** query parameter. For example, the



Coursera Restful API

https://api.coursera.org/api/catalog.v1/courses?fields=smallIcon,shortDescription&q=search&query=ruby

Search

Base URI Parameters

```
{  
  elements: [  
    {  
      id: 948,  
      shortName: "webapplications",  
      name: "Web Application Architectures",  
      shortDescription: "Learn how to build and deploy modern web application architectures – applications that run over the Internet, in the \"cloud,\" using a browser as the user interface.",  
      smallIcon: https://d15cw65ipctsrr.cloudfront.net/8a/72e49851da4f79d1a3a3df7f840d5c/mooc\_logo.jpg,  
      links: {}  
    },  
    {  
      id: 117,  
      shortName: "proglang",  
      name: "Programming Languages",  
      shortDescription: "Investigate the basic concepts behind programming languages, with an emphasis on the techniques and benefits of functional programming. Use the programming languages ML, Racket, and Ruby to learn how the pieces of a language fit together to create more than the sum of the parts. Gain new software skills and the concepts needed to learn new languages on your own.",  
      smallIcon: https://d15cw65ipctsrr.cloudfront.net/e1/644d37da2af639d0c9d1f4fca323c2/course\_logo.jpg,  
      links: {}  
    },  
  ]  
}
```

JSONView Browser Plugin (Available for Chrome and Firefox)



HTTParty Example

```
require 'httparty'
require 'pp'

class Coursera
  include HTTParty

  base_uri 'https://api.coursera.org/api/catalog.v1/courses'
  default_params fields: "smallIcon,shortDescription", q: "search"
  format :json

  def self.for term
    get("", query: { query: term})["elements"]
  end
end

pp Coursera.for "python"
```



HTTParty Example Output

```
[{"id"=>2760,
 "shortName"=>"algorithmicthink1",
 "name"=>"Algorithmic Thinking (Part 1)",
 "shortDescription"=>
 "Experienced Computer Scientists analyze and solve computational problems at a level of abstraction designed to train students in the mathematical concepts and process of \\"Algorithmic Thinking\\".
 "smallIcon"=>
 "https://d15cw65ipctsrr.cloudfront.net/70/6ff6409dd811e49bfb61275b434ba9/AlgTh_logo.jpg",
 "links"=>{}},
 {"id"=>3048,
 "shortName"=>"molecularevolution",
 "name"=>"Deciphering Molecular Evolution (Bioinformatics IV)",
 "shortDescription"=>
 "In this course, we will see how evolutionary trees resolve quandaries from finding the origin of life to solving puzzles from computational proteomics to test whether we can reconstruct Tyrannosaurus rex proteins from scratch.
 "smallIcon"=>
 "https://d15cw65ipctsrr.cloudfront.net/16/55b1b0f3c111e4a6a209002642cb9f/evo.jpg",
 "links"=>{}},
```



Summary

- ✧ HTTParty makes it **extremely easy** to ingest Restful services, converting them to Ruby Hashes
- ✧ **JSON** and **XML** formats **supported**

What's next?

- ✧ Bundler



In this lecture, we will discuss...

- ✧ Bundler
- ✧ Managing gems inside your Rails application



Bundler

The screenshot shows the official Bundler website at bundler.io. The page features a large, bold title "Bundler" in dark blue. To the left of the title is the subtext "The best way to manage your application's dependencies". To the right is a cartoon illustration of a white toilet paper roll with a face, wearing a blue tool belt and holding a wrench, standing next to a black pipe. A dark ribbon banner above the character says "Follow me on GitHub". The navigation bar at the top includes icons for back, forward, search, and other site functions.

What is Bundler?

Bundler provides a consistent environment for Ruby projects by tracking and installing the exact gems and versions that are needed.

Would you like to

- Get started
- Report a bug
- See what's new
- Read documentation



Bundler

- ✧ Lets you **specify gems** (and associated gem dependencies) for this Rails app inside **Gemfile** (in the root of your Rails app)
- ✧ **Preferred way** to **manage gem dependencies** in Rails
- ✧ Run **bundle install** or simply **bundle** after specifying a new gem in the Gemfile
- ✧ Run **bundle update** when **modifying** a version of a gem



Bundler

- ❖ You can **instruct Rails** (through **Gemfile**) to **only load certain gems** in specific Rails environments

```
group :development, :test do
  # Call 'byebug' anywhere in the code to stop execution and get a debugger console
  gem 'byebug'

  # Access an IRB console on exception pages or by using <%= console %> in views
  gem 'web-console', '~> 2.0'

  # Spring speeds up development by keeping your application running in the background.
  gem 'spring'
end
```



Bundler – Which Version Of Gem?

- ❖ If you **don't specify** ... You get the latest version
- ❖ Can specify an **exact version** or an **approximate version**

```
gem "nokogiri"  
gem "rails", "3.0.0.beta3"  
gem "rack", ">=1.0"  
gem "thin", ">= 1.1", "< 2.0"  
gem "thin", "~>1.1"
```

~> Pessimistic Version Constraint

Drop the final digit, then
increment to get the upper limit
version number



Bundler: require

- ✧ Occasionally, the name of the gem to be used inside `require` statement is **different** than the name of the gem

```
gem 'sqlite3-ruby', require: 'sqlite3'
```



Gemfile - Example

```
source 'http://rubygems.org'  
gem 'rails', '4.2.3'  
# Bundle edge Rails instead:  
# gem 'rails', github:'git://github.com/rails/rails.git'  
gem 'sqlite3'  
...  
...
```

- ✧ Our app can even use a **different version** of Rails if you **change the version** and run **bundle update**
- ✧ Bundler creates a **Gemfile.lock** file, which **contains the actual gem versions** your app is using with their **associated dependencies**



Bundle exec?

The screenshot shows a web browser window with the title "Bundler: The best way to m..." and the URL "bundler.io/v1.6/bundle_exec.html". The page content is as follows:

The best way to manage your application's dependencies

Bundler

bundle exec

Run the command in context of the bundle

```
$ bundle exec
```

Exec runs a command, providing it access to the gems in the bundle. While using bundle exec you can require and call the bundled gems as if they were installed into the systemwide Rubygems repository.



Summary

- ✧ Bundler **manages** gem dependencies
- ✧ **Loads** gems on application startup

What's Next

- ✧ Integrating HTTParty and Rails



In this lecture, we will discuss...

- ❖ Integrating Rails with HTTParty



HTTParty Integration - Gemfile

```
Gemfile *  
1 source 'https://rubygems.org'  
2  
3 gem 'rails', '4.2.3'  
4 gem 'sqlite3'  
5 gem 'sass-rails', '~> 5.0'  
6 gem 'uglifier', '>= 1.3.0'  
7 gem 'coffee-rails', '~> 4.1.0'  
8 gem 'jquery-rails'  
9 gem 'turbolinks'  
10 gem 'jbuilder', '~> 2.0'  
11 gem 'sdoc', '~> 0.4.0', group: :doc  
12  
13  
14 group :development, :test do  
15   gem 'byebug'  
16   gem 'web-console', '~> 2.0'  
17   gem 'spring'  
18 end  
19  
20 gem 'httparty', '0.13.5'
```

Specify version



HTTParty Integration - Gemfile

```
Installing httparty 0.13.5
Using multi_json 1.11.2
Using jbuilder 2.3.1
Using jquery-rails 4.0.4
Using bundler 1.9.5
Using sprockets 3.2.0
Using sprockets-rails 2.3.2
Using rails 4.2.3
Using rdoc 4.2.0
Using sass 3.4.16
Using tilt 1.4.1
Using sass-rails 5.0.3
Using sdoc 0.4.1
Using spring 1.3.6
Using sqlite3 1.3.10
Using turbolinks 2.5.3
Using uglifier 2.7.1
Using web-console 2.2.1
```

```
Bundle complete! 13 Gemfile dependencies, 56 gems now installed.
Use `bundle show [gemname]` to see where a bundled gem is installed.
```

Post-install message from httparty:

When you HTTParty, you must party hard!

```
~/my_first_app$
```

bundle (or bundle install)

NEED TO RESTART THE SERVER after running bundler for changes to take effect!!



Course Listing Example

- ❖ Let's show Courses based on search term

- ❖ Generate Courses Controller

Controller

Action

- ❖ `rails g controller courses index`

```
~/my_first_app$ rails g controller courses index
  create  app/controllers/courses_controller.rb
  route   get 'courses/index'
  invoke  erb
  create   app/views/courses
  create   app/views/courses/index.html.erb
  invoke  test_unit
  create   test/controllers/courses_controller_test.rb
  invoke  helper
  create   app/helpers/courses_helper.rb
  invoke  test_unit
  invoke  assets
  invoke  coffee
  create   app/assets/javascripts/courses.coffee
  invoke  scss
  create   app/assets/stylesheets/courses.scss
```



Courses Listing – config/routes.rb

```
routes.rb *  
  
Rails.application.routes.draw do  
  get 'courses/index' ←  
  
  # get 'greeter/hello'  
  
  # SAME AS ABOVE  
  get 'greeter/hello' => "greeter#hello"  
  get 'greeter/goodbye'  
  
  # You can have the root of your site routed with "root"  
  # root 'welcome#index'  
end
```



Coursera Model

The screenshot shows a file explorer on the left and a code editor on the right. The file explorer lists the following directory structure:

- my_first_app
 - app
 - assets
 - controllers
 - concerns
 - application_controller.rb
 - courses_controller.rb
 - greeter_controller.rb
 - helpers
 - mailers
 - models
 - concerns
 - coursera.rb

The code editor displays the content of the 'coursera.rb' file:

```
1 class Coursera
2   include HTTParty
3
4   base_uri 'https://api.coursera.org/api/catalog.v1/courses'
5   default_params fields: "smallIcon,shortDescription", q: "search"
6   format :json
7
8   def self.for term
9     get("", query: { query: term})["elements"]
10  end
11 end
```

By convention, controllers are named plural and model is named singular...

Also, note how HTTParty does not have to be required (thanks Bundler!)



Courses Controller

- ❖ Fill in `index` action

Assume jhu for now...

FOLDERS

```
my_first_app
  app
    assets
    controllers
    concerns
    application_controller.rb
  courses_controller.rb
```

```
courses_controller.rb
1 class CoursesController < ApplicationController
2   def index
3     @search_term = 'jhu'
4     @courses = Coursera.for(@search_term)
5   end
6 end
7
8
```

Notice, how the Controller did not have to `require Coursera...`



courses/index.html.erb

FOLDERS

```
my_first_app
  app
    assets
    controllers
    helpers
    mailers
    models
  views
    courses
      index.html.erb
    greeter
    layouts
  bin
```

```
index.html.erb
1 <h1>Searching for - <%= @search_term %></h1>
2
3 <table border="1">
4   <tr>
5     <th>Image</th>
6     <th>Name</th>
7     <th>Description</th>
8   </tr>
9   <% @courses.each do |course| %>
10    <tr>
11      <td><%= image_tag(course["smallIcon"])%></td>
12      <td><%= course["name"] %></td>
13      <td><%= course["shortDescription"] %></td>
14    </tr>
15  <% end %>
16 </table>
```



What Does It Look Like?

A screenshot of a web browser window. The address bar shows 'localhost:3000/courses/index'. The search bar contains 'Search' with a magnifying glass icon. To the right of the search bar are various icons: a star, a clipboard, a shield, a download arrow, a house, a message bubble, and a menu icon. The main content area displays a table with three rows of course information.

Searching for - jhu

Image	Name	Description
	Introduction to Genomic Technologies	The basic biology of modern genomics and the experimental tools used for measurement. This is the first course in the Genomic Big Data Science Specialization.
	Regression Models	Learn how to use regression models, the most important statistical analysis tool in the data scientist's toolkit. This is the seventh course in the Johns Hopkins Data Science Specialization.
	Statistics for Genomic Data	An introduction to the statistics behind the most popular genomic data science projects. This is the sixth course in



Summary

- ✧ By convention, Controller names are **plural** while Model names are **singular**
- ✧ Files under **app** directory are **automatically required**

What's Next?

- ✧ Making minor, but useful, improvements to our app



In this lecture, we will discuss...

- ❖ Adding **basic styling** to our view
- ❖ Making the app **more dynamic** with a request parameter
- ❖ **Routing** with a root path



Layout

- ❖ **views/layout/application.html.erb** serves as view's container (unless overriden)

The screenshot shows a file structure on the left and a code editor window on the right. The file structure includes controllers, helpers, mailers, models, views (with courses, greeter, layouts), bin, config, and environments. The code editor window shows the contents of application.html.erb:

```
<!DOCTYPE html>
<html>
<head>
  <title>MyFirstApp</title>
  <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track' => true %>
  <%= javascript_include_tag 'application', 'data-turbolinks-track' => true %>
  <%= csrf_meta_tags %>
</head>
<body>
<%= yield %>
</body>
</html>
```

A red arrow points from a callout bubble containing the text "Display the view" to the line "<%= yield %>".



What Does It Look Like Now?

A screenshot of a web browser window. The address bar shows 'localhost:3000/courses/index'. The search bar contains 'Search' with a magnifying glass icon. To the right of the search bar are several icons: a star, a folder, a download arrow, a house, a message bubble, and a menu. Below the browser window, the slide content is displayed.

Searching for - jhu

Image	Name	Description
	Introduction to Genomic Technologies	The basic biology of modern genomics and the experimental tools used for measurement. This is the first course in the Genomic Big Data Science Specialization.
	Regression Models	Learn how to use regression models, the most important statistical analysis tool in the data scientist's toolkit. This is the seventh course in the Johns Hopkins Data Science Specialization.
	Statistics for Genomic Data	An introduction to the statistics behind the most popular genomic data science projects. This is the sixth course in



Adding Some CSS

FOLDERS

```
my_first_app
  app
    assets
      images
      javascripts
    stylesheets
      application.css
      courses.scss
      greeter.scss
  controllers
  helpers
  mailers
```

```
courses.scss
1 // Place all the styles related to the courses controller here.
2 // They will automatically be included in application.css.
3 // You can use Sass (SCSS) here: http://sass-lang.com/
4
5 table {
6   border-collapse: collapse;
7 }
8
9 td {
10   padding: 12px;
11 }
12
13 .even {
14   background-color: #D6E1C3;
15 }
```

If you are new to CSS, go to Course 4 - HTML, CSS and Javascript for Web Developers



Modify View to Include CSS Classes

FOLDERS

```
my_first_app
  app
    assets
    controllers
    helpers
    mailers
    models
    views
      courses
        index.html.erb
      greeter
      layouts
    bin
```

```
index.html.erb
```

```
1 <h1>Searching for - <%= @search_term %></h1>
2
3 <table border="1">
4   <tr>
5     <th>Image</th>
6     <th>Name</th>
7     <th>Description</th>
8   </tr>
9   <% @courses.each do |course| %>
10     <tr class=<%= cycle('even', 'odd') %>>
11       <td><%= image_tag(course["smallIcon"])%></td>
12       <td><%= course["name"] %></td>
13       <td><%= course["shortDescription"] %></td>
14     </tr>
15   <% end %>
16 </table>
```

cycle (Rails) helper



What Does It Look Like Now?

A screenshot of a web browser window. The address bar shows 'localhost:3000/courses/index'. The search bar contains 'Search' with a magnifying glass icon. To the right of the search bar are several icons: a star, a folder, a download arrow, a house, a message bubble, and a menu. Below the browser window, the slide content is displayed.

Searching for - jhu

Image	Name	Description
	Introduction to Genomic Technologies	The basic biology of modern genomics and the experimental tools used for measurement. This is the first course in the Genomic Big Data Science Specialization.
	Regression Models	Learn how to use regression models, the most important statistical analysis tool in the data scientist's toolkit. This is the seventh course in the Johns Hopkins Data Science Specialization.
	Statistics for Genomic Data	An introduction to the statistics behind the most popular genomic data science projects. This is the sixth course



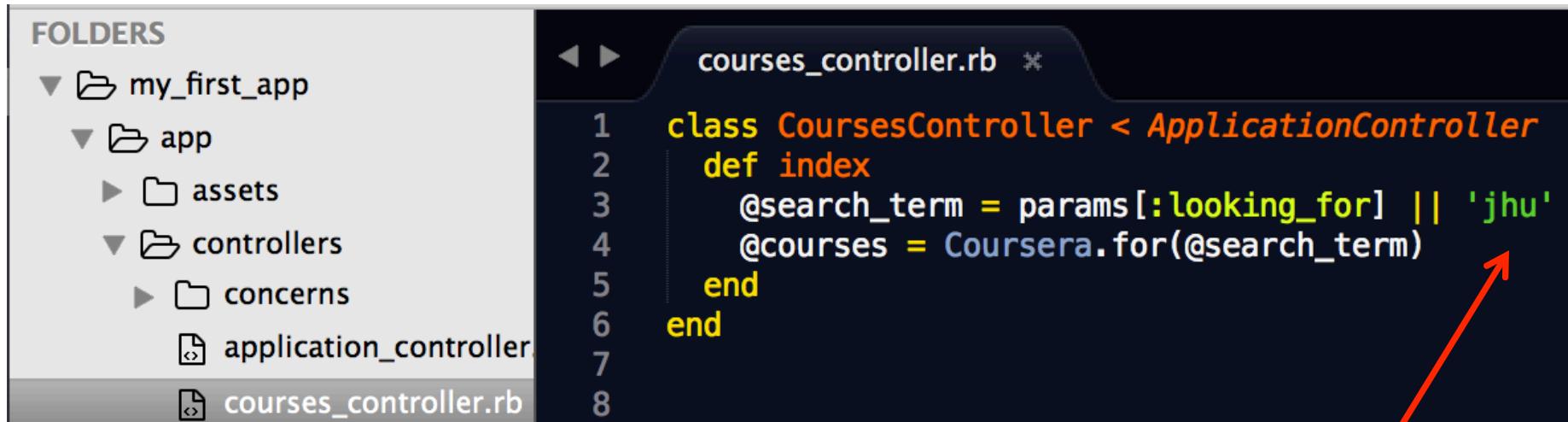
params helper

- ❖ It would be nice to **specify the search term**
- ❖ Use **params** “Hash” to **retrieve the value** (name of parameter becomes a symbol/key in the Hash)
- ❖ Returns **nil** if request parameter **not passed in** (standard Hash behavior)



params helper

- ❖ No changes to the Model or the View, **only** to the Controller



```
courses_controller.rb *
```

```
1 class CoursesController < ApplicationController
2   def index
3     @search_term = params[:looking_for] || 'jhu'
4     @courses = Coursera.for(@search_term)
5   end
6 end
```

Default to 'jhu' if request parameter not passed in



What Does It Look Like Now?

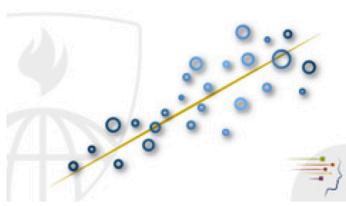


Searching for - python

Image	Name	Description
	Algorithmic Thinking (Part 1)	Experienced Computer Scientists analyze and solve computational problems at a level of abstraction that is beyond that of any particular programming language. This two-part class is designed to train students in the mathematical concepts and process of "Algorithmic Thinking", allowing them to build simpler, more efficient solutions to computational problems.
	Deciphering Molecular Evolution (Bioinformatics IV)	In this course, we will see how evolutionary trees resolve quandaries from finding the origin of a deadly virus to locating the birthplace of modern humans. We will then use methods from computational proteomics to test whether we can reconstruct Tyrannosaurus rex proteins and prove that birds evolved from dinosaurs.
	The Fundamentals of Computing Specialization Capstone	The "Fundamentals of Computing" series concludes with a two-week course that reviews the series material and whose primary assessment is a cumulative "capstone" exam.



And Without A Request Parameter?

Searching for - jhu		
Image	Name	Description
	Introduction to Genomic Technologies	The basic biology of modern genomics and the experimental tools used for measurement. This is the first course in the Genomic Big Data Science Specialization.
	Regression Models	Learn how to use regression models, the most important statistical analysis tool in the data scientist's toolkit. This is the seventh course in the Johns Hopkins Data Science Specialization.
	Statistics for Genomic Data Science	An introduction to the statistics behind the most popular genomic data science projects. This is the sixth course in the Genomic Big Data Science Specialization from Johns Hopkins University.



One Final Twist: Root Path

- ❖ What if we wanted the root path go to the `index` action? Just modify `routes.rb`

The image shows a code editor window and a file browser window side-by-side.

File Browser (Left):

- my_first_app
 - app
 - bin
 - config
 - environments
 - initializers
 - locales
 - application.rb
 - boot.rb
 - database.yml
 - environment.rb
- routes.rb

Code Editor (Right):

```
routes.rb
1 Rails.application.routes.draw do
2   get 'courses/index'
3
4   # get 'greeter/hello'
5
6   # SAME AS ABOVE
7   get 'greeter/hello' => "greeter#hello"
8   get 'greeter/goodbye'
9
10  root 'courses#index' ←
11  # You can have the root of your site routed with "root"
12  # root 'welcome#index'
13 end
14
15
```

A red arrow points to the line `root 'courses#index'` in the routes.rb file.



One Final Twist – Root Path



A screenshot of a web browser window. The address bar shows 'localhost:3000/?looking_for=diet'. The search bar contains the text 'Search'. To the right of the search bar are various icons for bookmarking, sharing, and navigating. The main content area displays a table with three rows of course information.

Searching for - diet

Image	Name	Description
	The Meat We Eat	<p>The Meat We Eat is a course designed to create a more informed consumer about the quality, safety, healthfulness and sustainability of muscle foods and address current issues in animal agriculture in developed and developing countries.</p>
	Sustainability of Food Systems: A Global Life Cycle Perspective	<p>This course explores the diversity of the foods we eat, the ways in which we grow, process, distribute, and prepare them, and the impacts they have upon our environment, health, and society. We will also examine the challenges and opportunities of creating a more sustainable global food system in the future.</p>
	Nutrition, Health, and Lifestyle: Issues and Insights	<p>This seven week course will explore nutrition concepts that take center stage in mainstream media outlets and become conversation topics among consumers interested in food choice as it relates to optimal health and physical performance.</p>



Summary

- ✧ Minor CSS changes can **dramatically** enhance the app
- ✧ **params** helper **parses** request parameters
- ✧ Easy to change the root path by **tweaking routes.rb**

What's Next?

- ✧ Deploying to Heroku



In this lecture, we will discuss...

- ✧ Deploying to Heroku



Deploying to Heroku

- ✧ Sign up for new account at heroku.com

The screenshot shows the Heroku homepage with a purple header bar. The header includes the Heroku logo, a navigation menu with links to Features, Pricing, Elements, Blog, Documentation, Support, Contact, and a sign-in/sign-up button, and a search bar. Below the header, the main content area has a dark purple background with white text. A large central heading says "Focus on the app" with a "Sign up for free" button below it. To the left, a section titled "Powerful platform built by developers for developers" features a screenshot of the Heroku dashboard showing a graph and terminal output for a deployment. Below this is a row of icons for various languages and frameworks. To the right, a section titled "Start-up style app dev fitted for the enterprise" features a screenshot of the Heroku Enterprise dashboard with a similar layout. At the bottom, there's a call-to-action for new users: "New to Heroku? Explore Heroku's developer experience in a live technical session. →".



Heroku Toolbelt

https://toolbelt.heroku.com

heroku

How It Works | Pricing | Add-ons | Dev Center | Help | Login

heroku toolbelt

everything you need to get started using heroku

Mac OS X Windows Debian/Ubuntu Standalone

[Download Heroku Toolbelt for Mac OS X](#)

What is it?

- Heroku client - CLI tool for creating and managing Heroku apps



Final Steps

- ✧ Heroku uses Postgres and recommends
[`rails_12factor`](#) gem
- ✧ Put [`sqlite`](#) gem into development group and “heroku”
gems in production



The screenshot shows the GitHub repository page for `rails_12factor`. The page includes the repository name, a green "build passing" badge, and a brief description: "Makes running your Rails app easier. Based on the ideas behind 12factor.net". Below this, there's a "What" section explaining the gem's purpose and how it improves Rails applications, and an "Install" section with instructions for adding the gem to a Gemfile.

Rails 12factor build passing

Makes running your Rails app easier. Based on the ideas behind 12factor.net

What

Rails gets a lot right when it comes to twelve-factor apps, but it could still be better. The two biggest areas right now are that in production [logs should be directed to stdout](#) and [dev/prod parity](#) while delivering assets.

This gem enables serving assets in production and setting your logger to standard out, both of which are required to run a Rails 4 application on a twelve-factor provider. The gem also makes the appropriate changes for Rails 3 apps.

Install

In your `Gemfile` add:

```
gem 'rails_12factor', group: :production
```



Final Steps

```
Gemfile          ×  
  
source 'https://rubygems.org'  
  
gem 'rails', '4.2.3'  
gem 'sqlite3', group: :development ←  
gem 'sass-rails', '~> 5.0'  
gem 'uglifier', '>= 1.3.0'  
gem 'coffee-rails', '~> 4.1.0'  
gem 'jquery-rails'  
gem 'turbolinks'  
gem 'jbuilder', '~> 2.0'  
gem 'sdoc', '~> 0.4.0', group: :doc  
  
group :development, :test do ...  
end  
  
group :production do ←  
  gem 'pg'  
  gem 'rails_12factor'  
end  
  
gem 'httparty', '0.13.5'
```

After adding the gems, run bundle and commit changes to repo!



Final Steps

```
~/my_first_app$ heroku login
Enter your Heroku credentials.
Email: kalmanh@gmail.com
Password (typing will be hidden):
Authentication successful.
~/my_first_app$ heroku create search-coursera-jhu
Creating search-coursera-jhu... done, stack is cedar-14
https://search-coursera-jhu.herokuapp.com/ | https://git.heroku.com/search-coursera-jhu.git
Git remote heroku added
~/my_first_app$ git remote -v
heroku  https://git.heroku.com/search-coursera-jhu.git (fetch)
heroku  https://git.heroku.com/search-coursera-jhu.git (push)
origin  https://github.com/jhu-ep-coursera/fullstack-course1-module3.git (fetch)
origin  https://github.com/jhu-ep-coursera/fullstack-course1-module3.git (push)
```



Final Steps

```
~/my_first_app$ git push heroku master
Counting objects: 159, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (144/144), done.
Writing objects: 100% (159/159), 155.86 KiB | 0 bytes/s, done.
Total 159 (delta 29), reused 0 (delta 0)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Ruby app detected ←
remote: -----> Compiling Ruby/Rails
remote: -----> Using Ruby version: ruby-2.0.0
remote: -----> Installing dependencies using 1.9.7
remote:       Running: bundle install --without development:test --path vendor/bundle --binstubs vendor/bundle/bin -j4 --deployment
remote:       Fetching gem metadata from https://rubygems.org/.....
remote:       Fetching version metadata from https://rubygems.org/...
remote:       Fetching dependency metadata from https://rubygems.org/..
remote:       Rubygems 2.0.14 is not threadsafe, so your gems must be installed one at a time. Upgrade to Rubygems 2.1.0 or higher to enable parallel gem installation.
```

Make sure that the top level directory of your git repo contains app directory and other Rails files and directories!



Final Steps

```
remote: ##### WARNING:  
remote: You have not declared a Ruby version in your Gemfile.  
remote: To set your Ruby version add this line to your Gemfile:  
remote: ruby '2.0.0'  
remote: # See https://devcenter.heroku.com/articles/ruby-versions for more information.  
remote:  
remote: ##### WARNING:  
remote: No Procfile detected, using the default web server (webrick)  
remote: https://devcenter.heroku.com/articles/ruby-default-web-server  
remote:  
remote: -----> Discovering process types  
remote: Procfile declares types -> (none)  
remote: Default types for Ruby -> console, rake, web, worker  
remote:  
remote: -----> Compressing... done, 29.4MB  
remote: -----> Launching... done, v5  
remote: https://search-coursera-jhu.herokuapp.com/ deployed to Heroku  
remote:  
remote: Verifying deploy... done.  
To https://git.heroku.com/search-coursera-jhu.git  
 * [new branch]      master -> master  
~/my_first_app$ heroku open  
Opening search-coursera-jhu... done
```



Final Steps

A screenshot of a web browser window. The address bar shows the URL https://search-coursera-jhu.herokuapp.com. The search bar contains the query "Search". To the right of the search bar are several icons: a star, a clipboard, a Twitter bird, a download arrow, a house, a smiley face, and a menu icon. The main content area displays a search result table.

Searching for - jhu

Image	Name	Description
	Introduction to Genomic Technologies	The basic biology of modern genomics and the experimental tools used for measurement. This is the first course in the Genomic Big Data Science Specialization.
	Regression Models	Learn how to use regression models, the most important statistical analysis tool in the data scientist's toolkit. This is the seventh course in the Johns Hopkins Data Science Specialization.
	Statistics for Genomic Data Science	An introduction to the statistics behind the most popular genomic data science projects. This is the sixth course in the Genomic Big Data Science Specialization from Johns Hopkins University.



Final Steps

A screenshot of a web browser window. The address bar shows the URL: https://search-coursera-jhu.herokuapp.com/?looking_for=diet. The page title is "Search for - diet". The search bar contains the text "Search". Below the search bar are various icons for bookmarking, sharing, and navigating. The main content area displays three course cards.

Searching for - diet

Image	Name	Description
	The Meat We Eat	<p>The Meat We Eat is a course designed to create a more informed consumer about the quality, safety, healthfulness and sustainability of muscle foods and address current issues in animal agriculture in developed and developing countries.</p>
	Sustainability of Food Systems: A Global Life Cycle Perspective	<p>This course explores the diversity of the foods we eat, the ways in which we grow, process, distribute, and prepare them, and the impacts they have upon our environment, health, and society. We will also examine the challenges and opportunities of creating a more sustainable global food system in the future.</p>
	Nutrition, Health, and Lifestyle: Issues and Insights	<p>This seven week course will explore nutrition concepts that take center stage in mainstream media outlets and become conversation topics among consumers interested in food choice as it relates to optimal health and physical performance.</p>



Summary

- ✧ Sign up for a Heroku account and download the toolbelt
- ✧ Don't forget to change the **Gemfile**

What's Next?

- ✧ Blackbox testing with RSpec and Capybara

