# Sinatra, ERB, Bundler and the Gemfile

Devpoint Labs - Jake Sorce / Dave Jungst

# Sinatra - Intro

A small, yet incredibly powerful DSL for quickly creating Ruby web applications

DSL - domain specific language. A domain-specific language is created specifically to solve problems in a particular domain and is not intended to be able to solve problems outside it.

# Sinatra - What you’ll need

- gem install sinatra

- your brain

# Sinatra - Getting started

- create a new folder and inside that folder create a new file called hello_world.rb

- At the very top:
  - require 'rubygems'
  - require 'sinatra'

# Sinatra - Getting started cont...

```
get '/' do
  "Hello, World!"
end
```

What is the http verb we are using?

=> GET

What is this going to render to the browser?

=> string “Hello World” right in the body tag

# Sinatra - URL Parameters

```
get '/hello/:name' do
  “Hello #{params[:name]}”
end
```

What is the parameter that is being passed?

=> :name

How do we call this route in the browser?

=> http://localhost:4567/hello/jake

# Sinatra - URL Parameters cont...

```
get '/hello/:name/:city' do
  "Hey there #{params[:name]} from #{params[:city]}."
end
```

What is the parameter that is being passed?

=> :name, :city

How do we call this route in the browser?

=> http://localhost:4567/hello/jake/sandy

# Sinatra - URL Parameters cont...

```
get '/query_string' do
  “I like: #{params[:sport]}”
end
```

What is the parameter that is being passed?

=> whatever you pass in the query string

How do we call this route in the browser?

=> http://localhost:4567/query_string?sport=snowboarding

# Sinatra - Error Pages

Right now if you go to a URL which there isn't a route for (eg. /bogus), you'll see Sinatra's fancy error message. On a real app we'd want to display our own error.

```
not_found do
  status 404
  'not found'
end
```

# ERB - Intro

ERB is a feature of Ruby that enables you to conveniently generate any kind of text, in any quantity, from templates. The templates themselves combine plain text with Ruby code for variable substitution and flow control, which makes them easy to write and maintain.

# ERB - Intro

Although ERB is most commonly seen generating Web pages, it is also used to produce XML documents, RSS feeds, source code, and other forms of structured text file. It can be extremely valuable when you need to create files which include many repetitions of a standard pattern.

# ERB - What does it look like?

- When writing ERB as an HTML template

- File extension is .erb
  - secret.erb

# ERB - What does it look like cont...

```
<h3>Message Encryptor!</h3>
<form action="/secret" method="post">
  <input type="text" name="message">
  <input type="submit">
</form>
```

# Sinatra and ERB

```
get '/secret' do
  erb :secret
end

post '/secret' do
  params[:message].reverse
end
```

# Sinatra and ERB cont...

```
get ‘/favorite_sport/:sport’ do
  @sport = params[:sport]
  erb :sport
end
```

What HTTP verb are we using?

=> GET

What is the param that is being passed?

=> sport

How are we accessing it?

=> params[:sport]

What are we rendering?

=> erb template called sport

Where is sinatra trying to find that file?

=> views/sport.erb

# Sinatra and ERB cont...

```
<h3>My favorite sport is:</h3>
<%= @sport %>


<h3>My favorite sport is: <%= @sport %></h3>


<h3>
  My favority sport is: <%= @sport %>
</h3>
```

# ERB - Syntax

- <%= some_variable %>
    - ERB tag to echo a variable value out
        - echo: print out

- <% %>
    - ERB tag to create a variables, do loops, conditionals, binding.pry, any ruby logic that you don’t want printed out

# ERB - Syntax examples

- <%= @sport %>
  - output the value of the sport instance variable from the route

-
  ```
  <% @sports.each do |sport| %>
          <p><%= sport %></p>
  <% end %>
  ```

# ERB - Syntax examples cont...

-
  ```
  <% if @sport == ‘snowboarding’ %>
      <h3>You are the coolest</h3>
  <% else %>
      <h3>Please try snowboarding</h3>
  <% end %>
  ```

# ERB - Syntax examples cont...

- <% binding.pry %>

-
  ```
  <% if @sport == ‘snowboarding’ %>
    <h3>You are the coolest</h3>
  <% else %>
    <h3>Please try snowboarding</h3>
  <% end %>
  ```

# ERB - Syntax examples cont...

-
  ```
  <ul>
    <% (1..10).each do |number| %>
      <li><%= number %></li>
    <% end %>
  </ul>
  ```

# Bundler and Gemfile - Intro

- Bundler is a gem that helps you manage your application's dependencies

- When you run bundler there needs to be a file named Gemfile in the directory that you are running in

# Bundler and Gemfile - Examples

- In the root of your folder create a file named Gemfile
  - there is no extension on the Gemfile
- Add all gem dependencies into this file
  - source 'https://rubygems.org'
  - ruby '2.2.2'
  - gem ‘sinatra’
  - gem ‘pry’
  - gem ‘babbler’

# Bundler and Gemfile - Examples

- After adding all of your dependencies into the gemfile
  - type bundle install into your terminal
- There are 3 types of bundles that you need to know about
  - bundle install - installs the gems to your system that your system doesn’t already have
  - bundle update - installs and updates the gems on your system
  - bundle update gem_name - updates a single gem to the latest version

# Gemfile.lock

- The last thing you need to know is when you run bundle install or bundle update it creates or modifies the Gemfile.lock file.

- This file keeps track of what Gems your application is dependent on and what versions those Gems are at. When you run a bundle update it updates your Gemfile.lock file.

- Remember you need to commit your Gemfile and Gemfile.lock files to Git, do not ignore these files.

# Sinatra Mini Project

- Create a new basic sinatra project
- Create the root route GET ‘/’ as your home page for an about me web page
- Create a GET ‘/contact’ route for a contact me page
- Create a GET ‘/resume’ route for a resume page
- Make all views with links to the other routes and style it all pretty like css.