Introduction to Sinatra

Sinatra

- A free and open source web application framework similar to Rails
- Allows you to build a web application with only one file
- Easily extensible to add database and other common web app functionality

Your first Sinatra web application

Create a new project folder with a Gemfile containing `sinatra`

- Run `bundle install`
- Create a main Ruby file ending in .rb
- In that file, put this code:

```ruby
require 'sinatra'

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

Your first Sinatra web application

- In the Terminal run

```
$ ruby main.rb
```

- Open your web browser and navigate to [localhost:4567]()
- Boom! You should see `Hello World`.

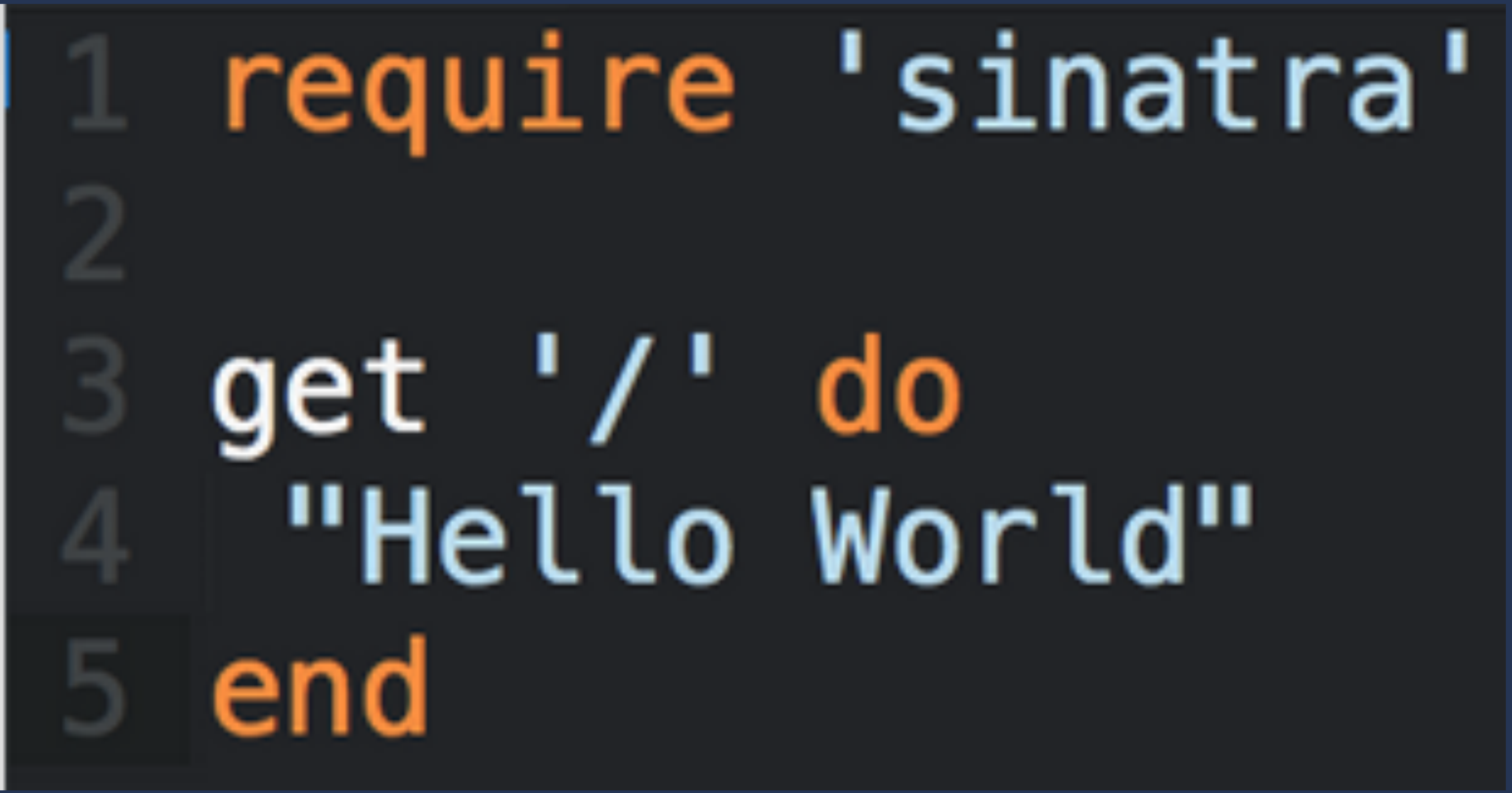
What's going on here?

- Sinatra is a **DSL** or a **D**omain **S**pecific **L**anguage
- It's a library that adds functionality to the core Ruby library so that you can easily declare web page routes

HTTP Methods

- Resource - a file, most of the time full of HTML content
- `GET` - Requests a representation of the specified resource, "load this page"
- `POST` - Requests the server to accept data input and process it as a new entity of a specified resource, "process this form data"

Sinatra Breakdown



```
1 require 'sinatra'
2
3 get '/' do
4  "Hello World"
5 end
```

- On line 1, we require the Sinatra library to gain access to its functionality
- On line 3, we declare a "route" to be accessed using the `HTTP GET` method whenever the user hits "/", the top level of the site
- On line 3, the `do` keyword indicates that this is a Ruby **block**
- On line 4, we specify what this block should return to the user
- On line 5, we end the Ruby block

Exercise

- Create a Sinatra “app” that serves up 3 different pieces of text depending upon which URL the user hits on your site
- Keep in mind that you need to restart your app every time you change the main Ruby file
- To restart the app, kill it with CTRL + C then type `ruby yourapp.rb`

Templating

- Wouldn’t it be nice to use HTML in your new website instead of just plain text?
- Enter ERB - "**E**mbedded **R**u**B**y"
- ERB is just like HTML except you can put Ruby in it!

Using ERB

- Create a folder inside of your project folder called `views`
- Inside of this folder, create a file called `home.erb` and put some HTML in it
- To render this view inside of a Sinatra route, use the following code:

```
erb :home
```

Using ERB

```
get "/home" do
  erb :home
end
```

Exercise

- Make one of your routes for the three-route app you created before into a route that displays an ERB view instead of returning plain text

Running Ruby code within ERB

Try adding the following code to your ERB file:

```
<% 10.times do %>
  <strong>
    THE PLANET IS 'SPLODIN'!
  </strong>
<% end %>
```

Running Ruby code within ERB

- To clarify, the syntax for running ruby code is:

```
<% #put your Ruby code here %>
```

- The syntax for running Ruby code **and** displaying its output is:

```
<%=“hi” + “there” %>
```

- Notice the = sign, this is what indicates you’d like the result of your Ruby code to be displayed

Exercises

- Add Ruby code to your ERB view that:
  - Assigns an array of names to a variable
  - Loops over that array variable and displays each name inside of a <b> tag
- **Bonus exercise**: Try creating a navigation menu inside of a <ul> that takes navigation items from an array and uses a loop (to avoid repeating <li> and

The public folder

- You already have a “views” folder for files that need to be processed by Ruby
- The public folder is at the top level of your app's directory, like the views folder
- It is used to serve assets that don’t need to be compiled, including images, CSS, and JavaScript files
- Files placed in this directory can be accessed at `localhost:4567/`, i.e. `/image.jpg` - this is the “top level” of your website
- You can also put subfolders inside of the public folder, for instance, an images folder to access images at /images/image.jpg

Using Layouts

- If you use a file called `layout.erb`, you can avoid having to put boilerplate HTML in all of your views
- Inside of this file, put the following line to demarcate where the chosen view's HTML will be inserted:
  `<%= yield %>`

Exercise

- Take a personal website project and convert it into a Ruby/Sinatra-based website.
    - Start with the basics: declare the correct routes in your main app file.
    - Then move over the views into the /views folder.