Switching from Rails to Sinatra

Rushi Shah

26 October 2015

So recently I created this little Ruby on Rails app that lets you embed your Github Contribution calendar into any HTML (or Markdown) as an image. I would show you a great example, but one of the downsides to PDFs generated with LaTeX is that you have to save the image locally (and in doing so, the magic of the app would be lost). However, if you want to see an awesome example, totally check out the github repo or just get your own on the site.

It is simple, straightforward, but still pretty neat, if I do say so myself. But soon I decided to rewrite the application in Sinatra, and here's why.

1 Starting with Rails

When I started this project, I dove in with rails because I had used it before on apps like a CNN article generator and a bookshelf. After I had my basic functionality down, the file structure looked like a typical rails app. For reference, here is the file structure at that point (created using tree).

```
.
 Gemfile
 Gemfile.lock
 README.rdoc
 Rakefile
 app
  assets
   images
   javascripts
    application.js
    chart.coffee
   stylesheets
       application.css
       chart.scss
       scaffolds.scss
  controllers
   application_controller.rb
   chart_controller.rb
```

```
  concerns
 helpers
  application_helper.rb
  chart_helper.rb
 mailers
 models
  concerns
 views
     layouts
         application.html.erb
bin
 bundle
 rails
 rake
 setup
 spring
config
 application.rb
 boot.rb
 database.yml
 environment.rb
 environments
  development.rb
  production.rb
  test.rb
 initializers
  assets.rb
  backtrace_silencers.rb
  cookies_serializer.rb
  filter_parameter_logging.rb
  inflections.rb
  mime_types.rb
  session_store.rb
  wrap_parameters.rb
 locales
  en.yml
 routes.rb
 secrets.yml
config.ru
db
 schema.rb
 seeds.rb
lib
 assets
 tasks
```

```
 log
 public
  404.html
  422.html
  500.html
  favicon.ico
  robots.txt
 test
  controllers
   chart_controller_test.rb
  fixtures
  helpers
  integration
  mailers
  models
  test_helper.rb
 vendor
     assets
         javascripts
         stylesheets

35 directories, 47 files
```

What a pain, right? Especially for the fact that the entirety of my code could be boiled down to a few lines:

```ruby
#config/routes.rb
get "/:username" => "chart#generate", format: :svg


#config/initializers/mime_types.rb
Mime::Type.register "image/svg+xml", :svg


#app/controllers/chart_controller.rb
class ChartController < ApplicationController
    def generate
        svg = GithubChart.new(user: params["username"]).svg
        respond_to do |format|
            format.svg { render inline: svg}
        end
  end
end
```

2 Enter Sinatra

I wasn't happy with how heavy everything felt, so I asked my friends over at HH Ruby for a lightweight alternative to Rails. Basically, they showed me a whole new world (namely, Sinatra).

After spending about an hour and a half learning* Sinatra and refactoring, I emerged victorious!

```
.
 Gemfile
 Gemfile.lock
 README.md
 app.rb
 config.ru
 public
     index.css
     index.html

1 directory, 7 files
```

Now, the entirety of my `app.rb` can be contained more or less to 20 lines with whitespace:

```ruby
#app.rb
require 'sinatra'
require 'githubchart'

get '/' do
    send_file File.join(settings.public_folder, 'index.html')
end


get '/:username' do
    headers 'Content-Type' => "image/svg+xml"

    username = params[:username].chomp('.svg') #Chomp off the .svg extension to be back

    svg = GithubChart.new(user: username).svg

    stream do |out|
      out << svg
    end
end
```

Fancy, huh?

3 Conclusion

Look, I'm not saying you should use Sinatra for every enterprise application you and your team are going to be scaling up for the next five years. I'm just saying that if you have a small hobby application and you just want some routes and their corresponding Ruby code, look into Sinatra. If you want to skip the whole MVC shizam and don't need a database, Sinatra might be for you.

Also, if you're interested in the application or want your chart, you can find more information on the github page or on the site.

Footnotes *Not really learning, because it was essentially the same as Express+Node.js and everything basically just worked.