Exercises - Session 33

In case you get stuck anywhere, don't be afraid to ask the coaches! They are here to help and will gladly explain everything to you! Take notes during the exercises. Even if you never look at them again, they will help you memorise things!

Please quickly read through the documentation, as those will help you solving the following tasks:

Sequel:
https://github.com/jeremyevans/sequel
http://sequel.jeremyevans.net/rdoc/files/doc/cheat_sheet_rdoc.html
http://sequel.jeremyevans.net/documentation.html

Sinatra:
http://www.sinatrarb.com/intro.html

Introduction

In this exercise you will create a blog web application with Ruby, Sinatra and Sequel to create and edit blog posts.

Project setup

1. Create a new directory where you can save your source code files, e.g. blog
2. Create a Gemfile and add Sinatra and Sequel to it.
3. Install the dependencies

Database setup

In order to save the created blog posts we use the Sequel gem which allows us to create, retrieve, update and delete data from and to a database. First of all we need to create a new sqlite database with the following command: `sqlite3 blog.db .schema`

Now create a new file with the name `setup.rb` and add some code to create a table with the name `posts` and the following columns (see Sequel documentation above): `id` (primary key), `title`, `content` and `author`.

Run the script to create the database.

Blog Application

Note: The solution for each task can be found in the Sinatra documentation (see Sinatra documentation above).

1. Create a new file with the name `app.rb` and require Sinatra and Sequel. Furthermore, create the `get “/”` action to fetch the posts from the database and render the `posts` template.
2. Create a new directory with the name `views` and create a new file `posts.erb` in this directory and add the HTML + ruby code to render the posts.
3. Create the `post “/posts”` action in `app.rb` to retrieve a new post, save it to the database and redirect the browser to the created post.
4. Create the `get “/posts/:id”` action to show the post with the given `post id` in `app.rb`.
5. Create a new file with the name `post.erb` in the `views` directory and the HTML + ruby code to display a blog post.
6. Create the `get “/posts/:id/edit”` action to render the edit form and the `put “/posts/:id”` action to update the post with the given post id.
7. Create a new file with the name `post_edit.erb` in the `views` directory and the HTML + ruby code to edit a blog post.
    a. You need to add `<input type="hidden" name="_method" value="put">` to your form to send the form via `put` instead of `post`.