SAMUEL RODRIGUES samuelclerod@gmail.com

CONSTRUINDO UM MVP COM RAILS

e-week
22 A 24 DE NOVEMBRO/18

APRESENTAÇÃO

RUBY É UMA LINGUAGEM DINÂMICA, OPEN SOURCE COM FOCO NA SIMPLICIDADE E NA PRODUTIVIDADE.

Yukihiro “Matz" Matsumoto

RAILS IS A WEB APPLICATION DEVELOPMENT FRAMEWORK WRITTEN IN THE RUBY PROGRAMMING LANGUAGE.

rubyonrails.org

INSTALANDO O RAILS

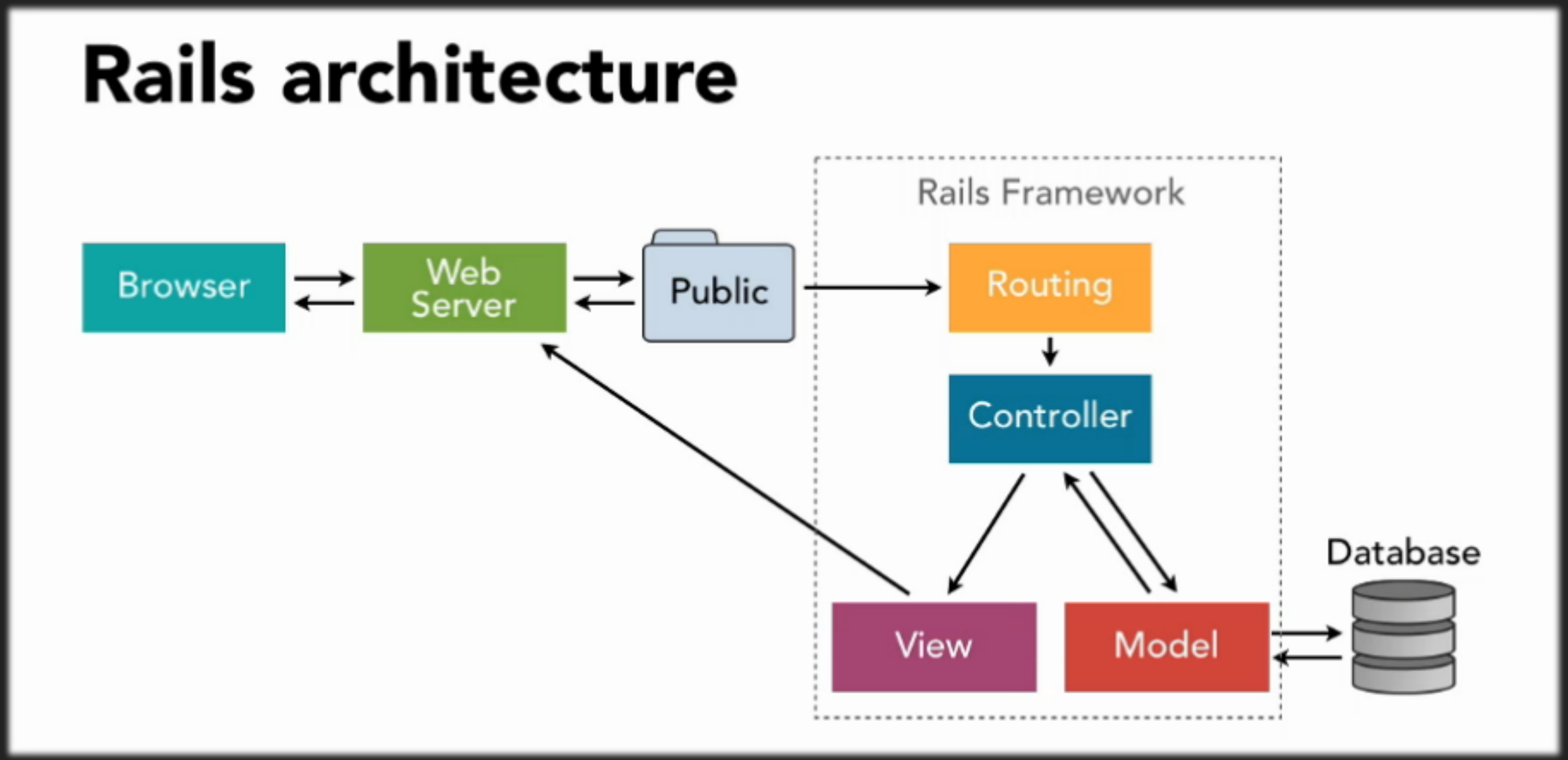
- Install Rails: http://installrails.com
- Visual Studio Code: https://code.visualstudio.com

> A MINIMUM VIABLE PRODUCT (MVP) IS THE THE MOST PARED DOWN VERSION OF A PRODUCT THAT CAN STILL BE RELEASED.
>
> Eric Ries

RUBY ON RAILS DEVELOPER-FRIENDLY FRAMEWORK BECAME VERY POPULAR WITH STARTUPS THAT WERE EAGER TO SIMPLIFY AND ACCELERATE THE PROCESSES OF THE PRODUCT DEVELOPMENT. SO THERE ARE A LOT OF MVPS WITH RAILS FRAMEWORK.

Iren Korkishko (Syndicode)

COMPONENTES



EXEMPLOS DE MVP QUE FORAM CONSTRUÍDOS EM RAILS

- AirBNB
- GitHub
- Dribbble
- Shopify
- Idiegogo
- Groupon
- Basecamp
- Hulu
- Kickstarter
- Goodreads
- SoundCloud

POR QUE RAILS?

- Existem muitas aplicações que já foram desenvolvidas em Rails desde 2004;
- É um Framework Open Source, com contribuição de mais de 4500 no código fonte;
- Foca na otimização da felicidade do programador com o princípio da conversão acima da configuração;
- A comunidade é grande e bastante amigável.

O MVP QUE IREMOS CRIAR

- Iremos construir uma plataforma simples para brainstorm, onde pode-se cadastrar problemas e provocações para serem desenvolvidos através de idéias de forma colaborativa.
- https://github.com/samuelclerod/brainstorm_app
- Seguiremos uma abordagem de tutorial.
- **Vamos começar 👊👊👊**

CRIANDO O APP

- Criar o Projeto:

```
> rails new brainstorm_app
```

- Acessar a pasta:

```
> cd brainstorm_app
```

- Testar o Puma:

```
> rails s
```

CRIAR O CRUD TOPIC

- Para gerar o CRUD completo da entidade **Topic**, execute:

```
> rails g scaffold Topic author description:text
```

- Execute a migração para efetuar as mudanças no banco de dados:

```
> rails db:migrate
```

GERAR O MODEL E CONTROLLER IDEAS

- Para gerar o model **Idea** , execute:

```
> rails g model Idea author description:text
topic:references
```

```
> rails db:migrate
```

- Para gera o Controller, execute:

```
> rails g controller Ideas
```

RELACIONANDO MODELOS

- Incluir no modelo **app/models/topic.rb**

```
class Topic < ApplicationRecord
    has_many :ideas, dependent: :destroy
end
```

- Incluir no modelo **app/models/Idea.rb**:

```
class Idea < ApplicationRecord
  belongs_to :topic
end
```

ALTERANDO AS ROTAS

- Construir rotas aninhadas editando **config/routes.rb**:

```
Rails.application.routes.draw do
  root to: 'topics#index'
  resources :topics do
    resources :ideas
  end
end
```

- Para verificar essas rotas digite

```
> rails routes
```

ALTERANDO AS VISÕES

- Inclua em **app/views/topics/show.html.erb**

```
(...)

<h2>Ideas</h2>
<%= render  @topic.ideas %>

<h2>Add an Idea</h2>
<%= render  "ideas/form" %>

<%= link_to 'Edit', edit_topic_path(@topic) %> |
<%= link_to 'Back', topics_path %>
```

ALTERANDO AS VISÕES

- Crie o partial app/views/ideas/**_form.html.erb**

```
<%= form_with( model:[@topic, @topic.ideas.build] , local: true) do |form|
%>
<p>
   <%= form.label :author %><br>
   <%= form.text_field :author %>
</p>
<p>
   <%= form.label :description %><br>
   <%= form.text_area :description %>
</p>
<p>
   <%= form.submit %>
</p>
<% end %>
```

ALTERANDO AS VISÕES

- Crie o partial app/views/ideas/**_idea.html.erb**

```
<p>
    <strong>Author:</strong> <%= idea.author %>
</p>
<p>
    <strong>Idea:</strong> <%= idea.description %>
</p>
 <p>
  <%= link_to 'Destroy Idea', [idea.topic, idea],
                method: :delete,
                data: { confirm: 'Are you sure?' } %>
</p>
```

IMPLEMENTANDO O IDEAS CONTROLLER

- Crie o partial app/views/ideas/**_idea.html.erb**

```
class IdeasController < ApplicationController
    def create
        @topic = Topic.find(params[:topic_id])
        @idea = @topic.ideas.create(idea_params)
        redirect_to topic_path(@topic)
    end

    def destroy
        @topic = Topic.find(params[:topic_id])
        @idea = @topic.ideas.find(params[:id])
        @idea.destroy
        redirect_to topic_path(@topic)
    end

    private
     def idea_params
        params.require(:idea).permit(:author, :description)
     end
end
```

INCLUINDO O SEMANTIC UI

- Adicione a seguinte gem no **Gemfile**:

```
gem 'semantic-ui-sass'
```

- Execute o comando abaixo para incluir a gem no nosso projeto:

```
> bundle install
```

INCLUINDO O SEMANTIC UI

- Altere **app/assets/stylesheets/application.css** alterando o arquivo para **application.css.scss** e inclua a linha:

```
@import 'semantic-ui';
```

- Inclua ainda linha seguinte no arquivo a**pp/assets/javascripts/application.js** antes dos demais requires:

```
//= require semantic-ui
```

ALTERE OS TEMPLATES A SEGUIR CONFORME LINKS

- app/views/ideas/_form.html.erb
- app/views/ideas/_idea.html.erb
- app/views/layouts/application.html.erb
- app/views/topics/_form.html.erb
- app/views/topics/index.html.erb
- app/views/topics/show.html.erb

INCLUINDO O DEVISE

- Adicione a seguinte gem no **Gemfile**:

```
gem ‘devise'
```

- Execute o comando abaixo para incluir a gem no nosso projeto:

```
> bundle install
```

- E o comando a seguir para instalar o devise

```
> rails g devise:install
```

INCLUINDO O DEVISE

- Adicionar área de notificações a **app/views/layouts/application.html.erb**:

```
  <body>
    <p class="notice"><%= notice %></p>
    <p class="alert"><%= alert %></p>
(…)
```

- Remova a área de notificações do template **index** e **show**

GERANDO OS ARTEFATOS

- Gere o modelo e a base de autenticação com os seguintes comandos:

```
> rails g devise User
```

```
> rails db:migrate
```

- E o comando a seguir para gerar as visões de autenticação

```
> rails g devise:views
```

- Verifique as visões, modelos e rotas geradas.

INCLUINDO LINKS DE REGISTRO E AUTENTICAÇÃO

- Altere o men de **app/views/layouts/application.html.erb** com os seguintes links:

```
<% if current_user %>
   <%= link_to 'Sign out', destroy_user_session_path,
                           method: :delete,
                           class: 'item' %>
<% else %>
   <%= link_to 'Sign up', new_user_registration_path, class: 'item'%>
   <%= link_to 'Sign in', new_user_session_path, class: 'item'%>
<% end %>
```

ALTERANDO MODELO IDEA

- Remover campo Author

```
> rails g migration removeAuthorFromIdeas
author:string
```

```
> rails db:migrate
```

- Adicionando o campo User

```
> rails g migration addUserToIdea user:references
```

```
> Rails db:migrate
```

ALTERANDO MODELO IDEA

- Associar app/models/topics.rb a user e criando um método para exibir username.

```
class Idea < ApplicationRecord
  belongs_to :topic
  belongs_to :user
    def username
        self.user.email.split('@').first.capitalize
    end
end
```

ALTERANDO MODELO TOPIC

- Remover campo Author

```
> rails g migration removeAuthorFromTopics
author:string
```

```
> rails db:migrate
```

- Adicionando o campo User

```
> rails g migration addUserToTopic user:references
```

```
> Rails db:migrate
```

ALTERANDO MODELO TOPIC

- Associar app/models/topics.rb a user e criando um método para exibir username.

```
class Topic < ApplicationRecord
    has_many :ideas, dependent: :destroy
    belongs_to :user
    def username
        self.user.email.split('@').first.capitalize
    end
end
```

ALTERANDO CONTROLLER TOPICS

- Altera método **create** app/controllers/topics_controller.rb:

```
@topic = Topic.new(topic_params)
@topic.user = current_user
```

- Remover campo author no método topic_params

```
def topic_params
   params.require(:topic).permit(:description)
end
```

ALTERANDO CONTROLLER IDEAS

- Altera método **create** app/controllers/ideas_controller.rb:

```
@idea = @topic.ideas.create(idea_params)
@idea.user=current_user
@idea.save
```

- Remover campo author no método idea_params

```
def ieda_params
   params.require(:idea).permit(:description)
end
```

ALTERAÇÃO FINAL DAS VISÕES

- Remover campo Author dos formulários **app/views/topics/_form.html.erb** e **app/views/ideas/_form.html.erb**.
- Alterar **app/views/topics/index.html.erb** : topic.username ao invés de topic.author
- Alterar **app/views/topics/show.html.erb** : @topic.username ao invés de @topic.author
- Alterar **app/views/ideas/_idea.html.erb** : idea.username ao invés de idea.author

ALTERAÇÃO FINAL DAS VISÕES

- Em todas operações restritas ao usuário envolver com código:

```
<% if idea.user==current_user %>
  (...)
<% end %>
```

- Em todas as operações que são necessárias estar logado, envolver com o código

```
<% if current_user%>
  (...)
<% end %>
```

ONDE POSSO APRENDER MAIS ?

- Rails Guides - https://guides.rubyonrails.org/index.html
- Série Lets Build: With Ruby on Rails - https://www.youtube.com/watch?v=4ABesTeDKmQ
- User And Admin Accounts With Devise - https://www.youtube.com/watch?v=7v2EMmfBJL8

samuelclerod@gmail.com

AGRADEÇO A ATENÇÃO