Vue.js + Ruby On Rails: Seu primeiro App em 10 minutos

O **Vue.js** é um framework que vem ganhando cada vez mais espaço no mundo JS devido a sua simplicidade e performance, somando essa ferramenta fantástica com o **Ruby on Rails** podemos desenvolver projetos extremamente profissionais.

Nesse tutorial rápido você vai aprender a **criar o seu primeiro APP** (Vue.js + Rails) e vai poder testar a **integração entre essas duas tecnologias incríveis**, bora?

O que vamos criar?

Criaremos uma **página de cadastro** de dados pessoais e profissionais de uma pessoa (um **mini-currículo**), onde teremos uma pré-visualização em tempo real do resultado final do currículo de acordo com a inserção de dados do usuário.

Quais ferramentas vamos usar?

- **Ruby On Rails 5.2**
- **Vue.js**
- **Bulma**
- **Yarn**

Breve Introdução ao Vue

O Vue, assim como o Angular, é um framework voltado para manipulação do front-end com um código bem enxuto. Ele utiliza o conceito de **two-way data binding**, onde ocorre o seguinte: existe uma View e um model JS e tudo o que acontece no model, é refletido a View e tudo que ocorre na View é refletido no model.

Este mapeamento Model-View é feito por meio de atributos que adicionamos nas tags html mapeado um objeto que criamos no arquivo JS.

Vamos usar como exemplo o seguinte:

```
var app = new Vue({
  el: document.getElementById("my_element"),
  data: function() {
    return {
      obj: {
        name: "OnebitCode",
        age: 25
      }
    }
  },
  methods: {
    sayHI: function(){
      alert("Hi!")
    }
  }
});
```

> Neste trecho de código nós inicializamos o Vue e passamos os parâmetros **el** para deixar o **Vue** disponível naquele elemento JS, um parâmetro **data** que deve retornar os objetos que deixaremos disponíveis para o DOM, permitindo o **two-way data binding** e um parâmetro **methods** que são os métodos que deixamos disponíveis para chamarmos através dos eventos do DOM.

Com este código do Vue, podemos ter um codigo HTML assim:

```
<div id="my_element">
  <input v-model="user.name" />
  <div v-if="user.age >= 30">Mature person</div>
  <button v-on:click="sayHi()">Say Hi</button>
</div>
```

Temos alguns itens importantes neste código:

> • **v-model** – fazemos uma ligação entre o valor da variável **obj.name** ao que está sendo digitado no input
> • **v-if** – só mostra o conteúdo dentro da tag se a condição for atendida
> • **v-on:** – dispara um método quando o evento é disparado. Com o **v-on:click**, ele detecta o evento **click** no botão e chama o método **sayHi** que disponibilizamos no atributo **methods**

> do **Vue**.

Antes de começarmos eu tenho um convite para você

No **dia 21/05** vai rolar um **evento (online e gratuito)** muito importante do OneBitCode onde criaremos um site baseado no **Netflix usando Ruby On Rails + Vue.js do zero ao deploy**, então aproveita para se cadastrar nele clicando na imagem a baixo:

Desenvolvendo nosso App

Por se tratar do primeiro post da série sobre **Vue.js + Rails** faremos uma introdução básica aos principais conceitos do **Vue.js** e evitaremos utilizar muitos recursos.

Neste post, vamos colocar o Vue para funcionar dentro do Rails.

Criando nosso projeto e preparando nosso scaffold

1. Vamos criar nosso projeto

```
rails _5.2.0_ new cv_example
```

2. Criando nosso scaffold de User

```
rails g scaffold User name email gender phone exp1 exp2 exp3 education extras description
```

3. Agora iremos criar o database e rodar as migrations:

```
rails db:create db:migrate
```

Instalando as dependências

1. No seu Gemfile, adicione a gem webpacker:

```
gem 'webpacker'
```

2. Rode o bundle:

```
bundle install
```

3. Agora podemos instalar nossas bibliotecas, rode os comandos:

```
bin/rails webpacker:install
```

```
bin/rails webpacker:install:vue
```

```
bin/yarn install
```

**Pode demorar um pouco para baixar os pacotes*

4. Vamos adicionar duas dependências importantes para o bom funcionamento do Vue em conjunto com o Rails:

```
yarn add vue-turbolinks vue-resource
```

5. Por último, vamos instalar o Bulma que irá facilitar a construção do nosso front-end:

```
yarn add bulma
```

6. Para que o Rails tenha acesso as classes do bulma, adicione ao seu application.css

```
*= require bulma/bulma.css
```

Integrando com o Vue

1. Antes de tudo, atualize o application.html.erb colocando:

```
<!DOCTYPE html>
<html>
  <head>
    <title>CVexample</title>
    <%= csrf_meta_tags %>
    <%= csp_meta_tag %>

    <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track': 'reload' %
    <%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
    <%= javascript_pack_tag 'hello_vue' %>
  </head>

  <body>
    <%= yield %>
  </body>
</html>
```

> Na tag head adicionamos o javascript_pack_tag que possibilita a leitura do nosso código dentro do arquivo hello_vue

2. Substitua o código do seu arquivo app/javascripts/packs/hello_vue.js por:

```
import Vue from 'vue/dist/vue.esm'
import TurbolinksAdapter from 'vue-turbolinks'
import VueResource from 'vue-resource'

Vue.use(VueResource)
Vue.use(TurbolinksAdapter)

```

```
document.addEventListener('turbolinks:load', () => {
  Vue.http.headers.common['X-CSRF-Token'] = document.querySelector('meta[name="csrf-token"]

  var element = document.getElementById("user-form")

  if (element != null) {
    var user = JSON.parse(element.dataset.user)

    var app = new Vue({
      el: element,
      data: function() {
        return { user: user }
      }
    });
  }
});
```

Neste código, nós iniciamos o nosso projeto:

Fizemos os imports das dependências incluindo o TurbolinksAdapter e o VueResource que instalamos anteriormente e executamos o **Vue.use** para garantir o funcionamento das bibliotecas dentro do Vue.

Iniciamos nosso método principal e garantimos o funcionamento com Turbolinks, através do evento **turbolinks:load** criamos a variável element com um elemento do DOM para iniciarmos o **Vue** e deste elemento nós também pegamos o valor de um **data attribute** com o objeto **user** e estamos retornando na função **data** do **Vue** para que fique disponível no DOM.

3. Agora vamos adicionar um método ao nosso objeto Vue. Substitua o código do hello_vue.js por:

```
import Vue from 'vue/dist/vue.esm'
import TurbolinksAdapter from 'vue-turbolinks'
import VueResource from 'vue-resource'

Vue.use(VueResource)
Vue.use(TurbolinksAdapter)

document.addEventListener('turbolinks:load', () => {
  Vue.http.headers.common['X-CSRF-Token'] = document.querySelector('meta[name="csrf-token"]

  var element = document.getElementById("user-form")

  if (element != null) {
    var user = JSON.parse(element.dataset.user)

    var app = new Vue({
      el: element,
      data: function() {
        return { user: user }
      },
      methods: {
        saveUser() {
          this.$http.post('/users', { user: this.user }).then(response => {
            Turbolinks.visit(`/users/${response.body.id}`)
          }, response => {
            console.log(response)
          })
        }
      }
    });
```

```
31   }
32 });
```

O método saveUser é chamado no click do botão submit do nosso form que veremos mais pra frente e dentro dessa função fazemos uma requisição do tipo post em /users que é a url do users#create gerada pelo nosso scaffold (passamos o parâmetro user com as informações do formulário.)

Temos ainda duas funções de parâmetro, uma de sucesso que redireciona para a página show do record criado, usando o Turbolinks.visit e uma de erro, que exibe o erro no console.

4. Vamos fazer uma pequena alteração no nosso arquivo app/views/users/new.html.erb:

```
1 <%= render 'form', user: @user %>
2
3 <%= link_to 'Back', users_path %>
```

5. Agora é a vez de modificar o nosso arquivo views/users/_form.html.erb, substitua seu conteúdo por:

```
1  <%= content_tag :div, id: 'user-form', class: 'columns', data: { user: user.to_json(except:
2    <div class='column'>
3      <div id="form" class="card">
4        <div class="card-content">
5          <h2 class="title">Cadastrar Usuário</h2>
6
7          <label>Nome</label>
8          <input type='text' v-model="user.name" class="input">
9
10         <label>Email:</label>
11         <input type='text' v-model="user.email" class="input">
12
13         <label>Telefone:</label>
14         <input type='text' v-model="user.phone" class="input">
15
16         <label>Descrição</label>
17         <input type='text' v-model="user.description" class="input">
18
19         <label>Experiencia 1</label>
20         <textarea class="textarea" v-model="user.exp1"></textarea>
21
22         <label>Experiencia 2</label>
23         <textarea class="textarea" v-model="user.exp2"></textarea>
24
25         <label>Experiencia 3</label>
26         <textarea class="textarea" v-model="user.exp3"></textarea>
27
28         <label>Formação</label>
29         <textarea class="textarea" v-model="user.education"></textarea>
30
31         <label>Extras</label>
32         <textarea class="textarea" v-model="user.extras"></textarea>
33
34         <label>Sexo</label>
35         <select v-model="user.gender" class="select">
36           <option>Masculino</option>
37           <option>Feminino</option>
38         </select>
39       </div>
```

```
      <div class="card-footer">
        <button @click="saveUser" class="button is-link">Criar Currículo</button>
      </div>
    </div>
  </div>
<% end %>
```

> Iniciamos com uma (content_tag div) que possui o id user-form, que é id que o Vue busca através da opção **el** na inicialização e um atributo data onde passamos o objeto user que será enviado pro Vue.

6. Agora vamos adicionar à nossa view (views/users/_form.html.erb) a parte de pré visualização do currículo do usuário, seu arquivo deverá ficar dessa forma:

```
<%= content_tag :div, id: 'user-form', class: 'columns', data: { user: user.to_json(except:
  <div class='column'>
    <div id="form" class="card">
      <div class="card-content">
        <h2 class="title">Cadastrar Usuário</h2>

        <label>Nome</label>
        <input type='text' v-model="user.name" class="input">

        <label>Email:</label>
        <input type='text' v-model="user.email" class="input">

        <label>Telefone:</label>
        <input type='text' v-model="user.phone" class="input">

        <label>Descrição</label>
        <input type='text' v-model="user.description" class="input">

        <label>Experiencia 1</label>
        <textarea class="textarea" v-model="user.exp1"></textarea>

        <label>Experiencia 2</label>
        <textarea class="textarea" v-model="user.exp2"></textarea>

        <label>Experiencia 3</label>
        <textarea class="textarea" v-model="user.exp3"></textarea>

        <label>Formação</label>
        <textarea class="textarea" v-model="user.education"></textarea>

        <label>Extras</label>
        <textarea class="textarea" v-model="user.extras"></textarea>

        <label>Sexo</label>
        <select v-model="user.gender" class="select">
          <option>Masculino</option>
          <option>Feminino</option>
        </select>
      </div>
      <div class="card-footer">
        <button @click="saveUser" class="button is-link">Criar Currículo</button>
      </div>
    </div>
  </div>

  <div class='column is-two-thirds preview card'>
    <div id="cv-preview" class="card-content">
      <h3 class="center title name">{{ user.name }}</h3>
      <p class="center subtitle"><b v-if="user.phone != null">Telefone de contato:</b> {{ us
```

```
      <div class="description">
        <p class="subtitle" v-if="user.description != null">Descrição</p>
        <p>
          {{ user.description }}
        </p>
      </div>

      <div class="exp">
        <p class="subtitle" v-if="user.exp1 != null">Experiencia 1</p>
        <p>
          {{ user.exp1 }}
        </p>
      </div>

      <div class="exp">
        <p class="subtitle" v-if="user.exp2 != null">Experiencia 2</p>
        <p>
          {{ user.exp2 }}
        </p>
      </div>

      <div class="exp">
        <p class="subtitle" v-if="user.exp3 != null">Experiencia 3</p>
        <p>
          {{ user.exp3 }}
        </p>
      </div>

      <div class="education">
        <p class="subtitle" v-if="user.education != null">Formação</p>
        <p>
          {{ user.education }}
        </p>
      </div>

      <div class="extras">
        <p class="subtitle" v-if="user.extras != null">Extras</p>
        <p>
          {{ user.extras }}
        </p>
      </div>
    </div>
  </div>
<% end %>
```

> No Vue podemos utilizar a notação **{{ var }}**, que é a forma de mostrar na nossa view uma variável que vem do Vue.js, então **{{ user.name }}** irá mostrar o valor atribuído a essa variável.

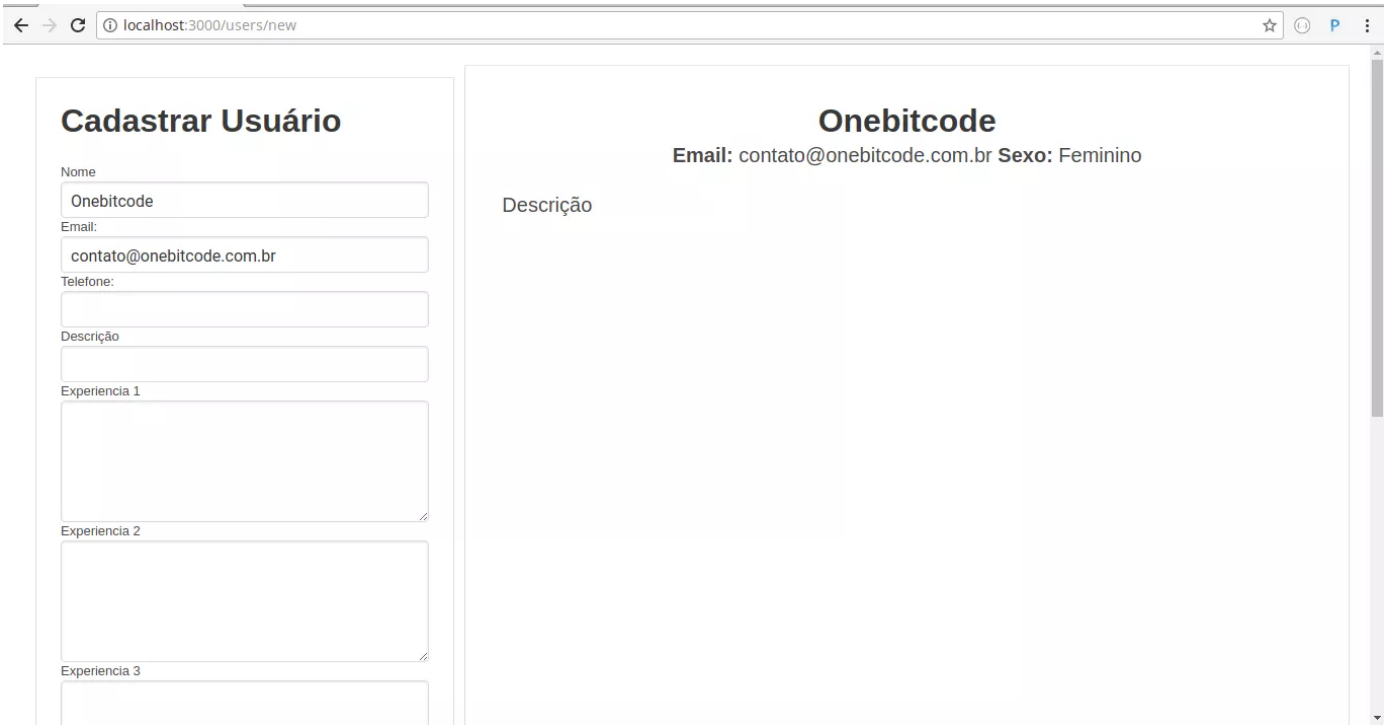
7. Por ultimo adicione ao seu users.scss

```
#cv-preview {
  margin: 20px 70px 0 70px;
}

.select {
  margin-bottom: 20px;
}

p {
  margin-bottom: 20px;
}

.center {
```

```
14   text-align: center;
15 }
```

Resultado

Depois de seguir os passos anteriores seu APP deve ter ficado desta forma:



Conclusão

Criamos um exemplo simples, porém já é possível perceber a praticidade do Vue.js e como é fácil fazê-lo funcionar dentro do Rails (caso você queira saber mais sobre ele acesse nosso post introdutório: primeiro contato com o Vue)

Compartilhe esse post com seus amigos e comente aí embaixo, seu feedback é muito importante para nós! 🙂
Obrigado por nos acompanhar,
Estamos juntos \o/

Você é novo por aqui?

Primeira vez no OneBitCode? Curtiu esse conteúdo? O OneBitCode tem muito mais para você!

O OneBitCode traz conteúdos de qualidade e em português sobre programação com foco em Ruby on Rails e outras tecnologias como Angular, Ionic, React, desenvolvimento de Chatbots e etc.

Se você deseja aprender mais, de uma forma natural e dentro de uma comunidade ativa, visite nosso **Facebook** e nosso **Twitter**, veja os screencasts e talks no **Youtube**, alguns acontecimentos no **Instagram,** ouça os **Podcasts** e faça parte de nossa **Newsletter**.

Além disso, também estamos com alguns **e-Books** muito interessantes para quem deseja se aprimorar como programador e também como freelancer (*os e-Books são gratuitos!*):

- **WorkFlow Super Full Stack**
  Desenvolvendo seus projetos como um profissional
- **Guia do Freelancer**
  PDF com links fundamentais para quem quer ser um freelancer de sucesso
- **Guia One Bit Code de Gems**
  Baixe gratuitamente seu e-Book com 60 Gems separadas por categorias

Espero que curta nossos conteúdos e sempre que precisar de ajuda com os tutoriais, fala com a gente! Seja por **Facebook** ou **e-mail**, estamos aqui para você 🙂

Bem-vindo à família OneBitCode \o/

MAIO 16, 2018

CRISTIALLAN

WEBSITE :

CURSOS | POSTS

Deixe um comentário

16 Comentários em "Vue.js + Ruby On Rails: Seu primeiro App em 10 minutos"

CONNECT WITH:

b i link b-quote u ul ol li code spoiler

Entre na discussão

Subscribe ▾ ▲ newest ▲ oldest ▲ most voted

Visitante

Romenig Lima Damasio

muito bom, vlw pelo tuto!!
só pra constar, quando vai editar ele cria um novo post com os dados existentes.

★ Carregando...

1 Responder 22 dias atrás

Leonardo Scorza

Legal Romening,
Ficamos felizes que você curtiu \o/

★ Carregando...

0 Responder 22 dias atrás

CristiAllan

Autor

E aí Romening, obrigado pelo feedback!:D
Então cara, nesse caso nós não usamos um botão submit comum e sim um botão customizado chamando a função saveUser, então independente da página ele vai criar um novo recorde, o desafio proposto no final do post é justamente esse, criar a lógica necessária para na action edit editar o post existente. Caso queira implementar e ficar com alguma dúvida de como fazer pode me chamar no Facebook ou no LinkedIn que quebramos cabeça juntos 🙂
Abraços!

★ Carregando...

1 Responder 22 dias atrás

Visitante

Edson Lima

Pode me ajudar com este erro:

Showing
/Users/edsonlima/projetos/rails_vue/cv_example/app/views/layouts/application.html.erb
where line #6 raised:

undefined local variable or method `csp_meta_tag' for #<#:0x007ff0d49a6808>
Did you mean? csrf_meta_tag
csrf_meta_tags

Extracted source (around line #6)

CvExample

★ Carregando...

0 Responder 22 dias atrás

Leonardo Scorza

E ai Edson, como vai?
Qual a sua versão do Rails?

★ Carregando...

0 Responder 22 dias atrás

Autor

CristiAllan

Parece que você escreveu errado o nome da tag csrf_meta_tag no seu arquivo application.html.erb cara. confere no arquivo e me avisa aqui depois. 🙂

★ Carregando...

1 Responder 22 dias atrás

Sergio Lima

Visitante

Parabéns pelo ótimo artigo!
Por que escolheram o Bulma? Alguma razão especial?
Obrigado!

★ Carregando...

1 Responder 20 dias atrás

Autor

CristiAllan

Olá Sérgio, obrigado pelo feedback.
Cara o blog já tem tutoriais usando bootstrap e materialize, só quis trazer uma opção aos dois, nenhuma razão especial. =)

★ Carregando...

0 Responder 17 dias atrás

Membro

Line

Olá.
Estou usando o Cloud9 para criar os projetos.
Quando executei o comando ***bin/yarn install*** me deparei com o seguinte erro:
Yarn executable was not detected in the system.
Download Yarn at https://yarnpkg.com/en/docs/install
Como resolvo já que uso uma IDE no navegador?

★ Carregando...

0 Responder 20 dias atrás

Autor

CristiAllan

Olá Line, isso está acontecendo por que o yarn não está instalado na sua máquina do cloud9. Acredito que após instalar você irá conseguir seguir o tutorial normalmente. Para instalar é só você seguir o passo a passo da documentação do yarn rodando os comandos no terminal do cloud9. Obrigado!

★ Carregando...

1 Responder 17 dias atrás

Membro

Line

Obrigada.

Utilizei o comando : ***curl -o- -L*** *https://yarnpkg.com/install.sh* ***| bash -s — – nightly***
Link: https://yarnpkg.com/en/docs/install#alternatives-nightly

★ Carregando...

1 Responder 17 dias atrás

Visitante

Luan

Parabéns pelo artigo!
Rodei todos os comandos corretamente, porém estou com um problema:
Webpacker::Manifest::MissingEntryError in Users#new
Showing /home/luan/projetos/cv_example/app/views/layouts/application.html.erb where line #10 raised:

Webpacker can't find hello_vue.js in /home/luan/projetos/cv_example/public/packs/manifest.json. Possible causes:
1. You want to set webpacker.yml value of compile to true for your environment unless you are using the `webpack –w` or the webpack-dev-server.
2. webpack has not yet re-run to reflect updates.
3. You have misconfigured Webpacker's config/webpacker.yml file.
4. Your webpack configuration is not creating a manifest.
Your manifest contains:
{
}
Extracted source (around line #10):
8
9
10
11
12

Rails.root: /home/luan/projetos/cv_example

★ Carregando...

0 Responder 15 dias atrás

Membro

André Gonçalves Rodrigues

Estou com o mesmo problema Luan, conseguiu resolver?

★ Carregando...

0 Responder 12 dias atrás

Visitante

Fabio

Estou com um problema na hora de gem 'webpacker' diz que o diretorio não existe, ja dei sudo gem install webpacker e gem install webpacker, só que ate agora nada

★ Carregando...

0 Responder 13 dias atrás

CristiAllan

Autor

Oi Fábio, beleza ? Cara pode ser que seja problema de encoding ao copiar o texto do snippet, escreve na mão -> gem 'webpacker', '~>3.5'
Vê se o bundle roda qualquer coisa me avisa por aqui =)

★ Carregando...

0 Responder

13 dias atrás

Visitante

Francisco

Fala Galera, gostei do tutorial, bem bacana. Só acho que encurtaria alguns passos se já criassem a aplicação usando os atalhos que o rails fornece, como por exemplo: rails new cv_example -d postgresql –webpack

★ Carregando...

0 Responder

4 dias atrás

E-BOOKS