
Desenvolvimento de Aplicações Web

Roteamento com Vue.JS

Prof. Márcio Barros

Frameworks de lado cliente





Criando uma nova
aplicação

Criação de uma aplicação Vue.JS

- ❑ Até agora, utilizamos o Vue.JS para criar pequenos exemplos e componentes que realizam operações simples, como a chamada de uma API
 - No entanto, queremos construir aplicações mais complexas, com diversos componentes que contêm partes da lógica da aplicação e interagem entre si
 - Para isto, precisamos utilizar um componente chamado Vue Router, que permite definir como será a navegação entre os componentes
 - O Vue Router funciona em uma aplicação completa do Vue.JS, que é construída usando Node.JS e uma aplicação própria chamada Vue CLI

Criação de uma aplicação Vue.JS

- ❑ Ferramentas que precisamos instalar no sistema onde a aplicação será construída
 - **NodeJS**: é uma plataforma de desenvolvimento e execução de programas JavaScript que usa o motor V8, desenvolvido pelo Google para o Chrome
 - **Node Package Manager** (NPM): gerenciador de dependências para NodeJS, similar ao Maven para programas Java
 - **VS Code**: ambiente simplificado de desenvolvimento para diversas linguagens, que possui plug-ins para VueJS e JavaScript
 - **Vue CLI**: interface de linha de comando para VueJS



Node JS: <https://nodejs.org/en/>

VS Code: <https://code.visualstudio.com/>

Criação de uma aplicação Vue.JS

- ❑ Instalação do VUE-CLI

```
> npm install -g @vue/cli
```

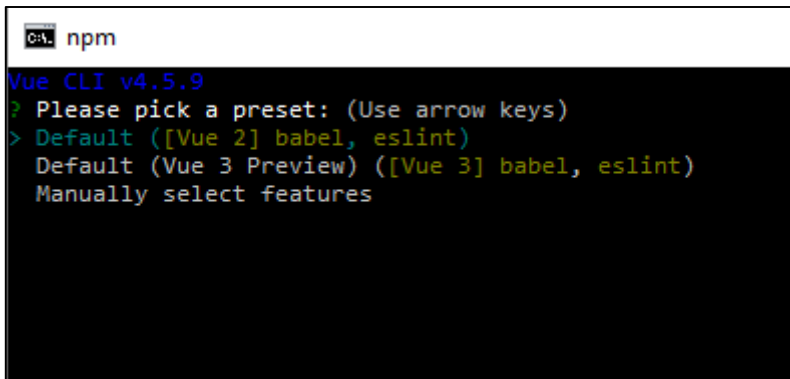
- ❑ Teste de instalação do VUE-CLI

```
> vue --version
```

- ❑ Criação de aplicação

```
> vue create my-app
```

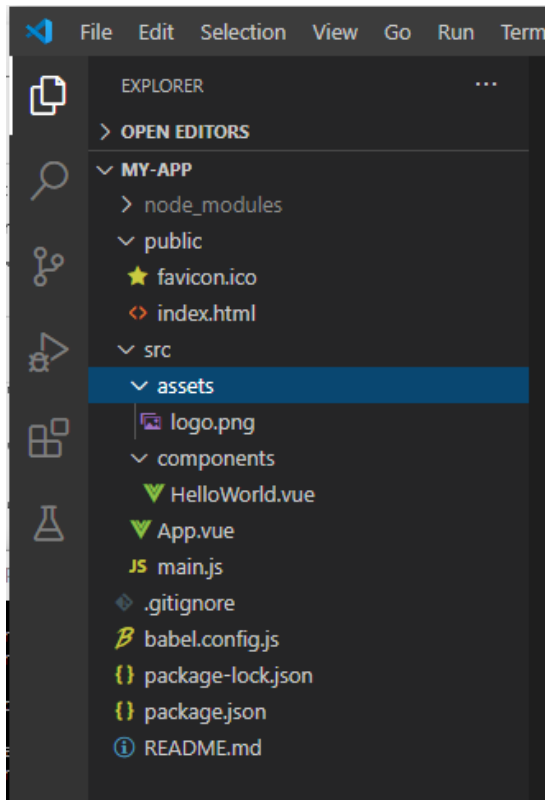
Criação de uma aplicação Vue.JS



```
npm
Vue CLI v4.5.9
> Please pick a preset: (Use arrow keys)
> Default ([Vue 2] babel, eslint)
  Default (Vue 3 Preview) ([Vue 3] babel, eslint)
  Manually select features
```

- ❑ Vamos utilizar a versão estável do Vue, que é a versão Vue 2
- ❑ **Babel** é um “transpiler”, um compilador que transforma o JS ECMA 2015 usada pelo Vue em JS tradicional
- ❑ **Lint** é um analisador que procura trechos de código JS problemáticos ou que fujam dos padrões sugeridos pela linguagem
- ❑ **Webpack** é um bundler (“juntador”) que reúne diversos módulos JS em um único módulo, que pode ser minificado para reduzir o tempo de carga do programa na Web

Criação de uma aplicação Vue.JS



- A imagem ao lado mostra a estrutura de uma aplicação criada pelo Vue-CLI
- Os principais diretórios são **asset**, para arquivos estáticos usados pela aplicação, e **components**, para componentes que serão desenvolvidos
- **Main.js** é o arquivo principal da aplicação e **App.Vue** é o componente que gerencia a aplicação

Criação de uma aplicação Vue.JS

- ❑ Para executar a aplicação, usamos o comando abaixo:

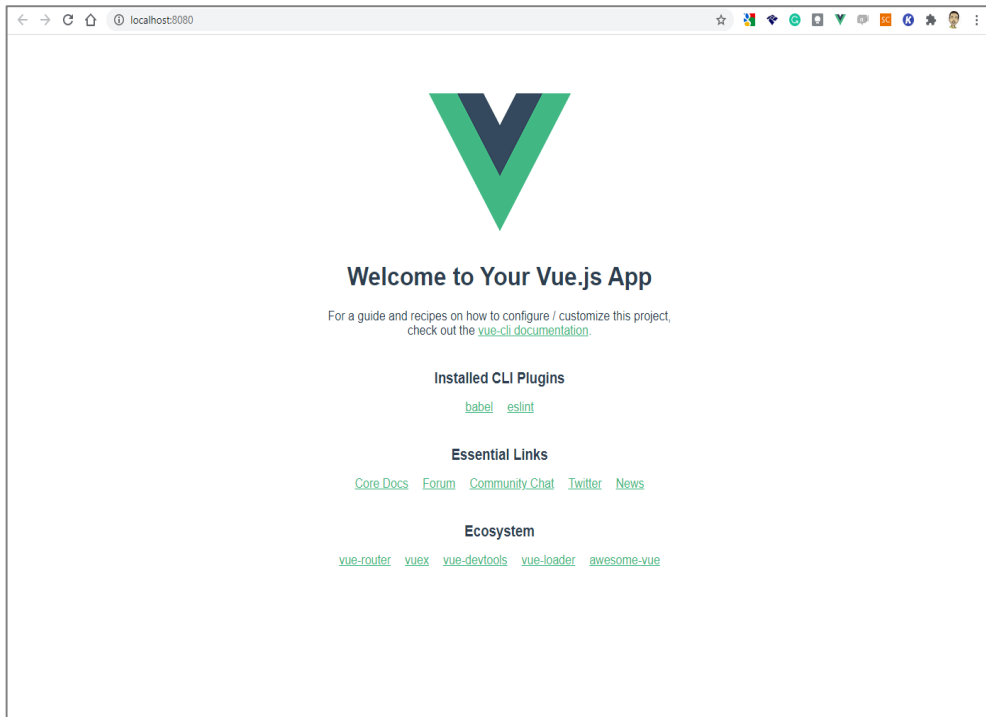
```
> npm run serve
```

```
INFO Starting development server...  
98% after emitting CopyPlugin  
DONE Compiled successfully in 2313ms
```

App running at:

```
- Local:   http://localhost:8080/  
- Network: http://192.168.0.10:8080/
```

Note that the development build is not optimized.
To create a production build, run `npm run build`.



VUE ROUTER



SPA: Single page application

- ❑ Uma aplicação de página única tem apenas um arquivo HTML
 - Este arquivo carrega a aplicação Vue.JS que possui um componente central, responsável pelo gerenciamento da aplicação
 - O componente central possui um *placeholder*, um local onde são adicionados outros componentes da aplicação
 - A navegação é realizada entre componentes e não entre páginas da aplicação (embora seja possível navegar entre páginas)
 - Um roteador de lado cliente é responsável por controlar o componente que está visível e quais são os próximos componentes

Vue Router

- ❑ O Vue Router é um componente e pode ser instalado pelo NPM

```
> npm install vue-router
```

- ❑ Ele deve ser carregado e inicializado no arquivo `main.js`

```
import Vue from 'vue'  
import Router from 'vue-router'  
  
...  
  
Vue.use(Router)
```

Vue Router

- ❑ Outros componentes também devem ser carregados no arquivo **main.js**

```
import Vue from 'vue'
import Router from 'vue-router'

import App from './App.vue'
import Homepage from './components/Homepage.vue'
import Login from './components/login/login/Login.vue'

/* Criacao de conta */
import CriacaoConta from './components/login/criacao-conta/CriacaoConta.vue'
import CriacaoContaSucesso from './components/login/criacao-conta/CriacaoContaSucesso.vue'

/* Esquecimento de senha */
import EsquecimentoSenha from './components/login/esquecimento-senha/EsquecimentoSenha.vue'
import EsquecimentoSenhaSucesso from './components/login/esquecimento-senha/EsquecimentoSenhaSucesso.vue'
import RecuperacaoSenha from './components/login/esquecimento-senha/RecuperacaoSenha.vue'
import RecuperacaoSenhaSucesso from './components/login/esquecimento-senha/RecuperacaoSenhaSucesso.vue'
```

Vue Router

- ❑ Finalmente, as rotas de navegação são definidas no arquivo **main.js**
 - **path** indica o caminho associado à rota
 - **name** é o nome da rota
 - **component** é o componente apresentado quando se navega para a rota

```
const router = new Router({
  mode: 'history',
  routes: [
    {
      path: '/',
      name: 'home',
      component: Homepage
    },
    {
      path: '/login',
      name: 'login',
      component: Login,
    },
    {
      path: '/login/new',
      name: 'create-account',
      component: CriacaoConta,
    },
  ],
})
```

Vue Router

- ❑ A aplicação define o componente onde sua execução tem início (App) e indica o uso do roteador
- ❑ A aplicação também mantém dados globais que podem ser acessados pelos componentes

```
import Vue from 'vue'
import Router from 'vue-router'
import App from './App.vue'

...

new Vue({
  data: {
    credentials: null,
    config: {
      url: "http://localhost:8080"
    }
  },

  el: '#app',
  render: h => h(App),
  router
})
```

Vue Router

- ❑ O template do componente principal deve definir o local onde devem ser apresentados os componentes
- ❑ O atributo “to” indica o nome ou o caminho (path) da rota
- ❑ O template (deste e de outros componente) também contém links para outras rotas

```
<template>
  <div id="app">
    <nav class="navbar navbar-inverse navbar-fixed-top">
      ...
      <router-link :to="{ name: 'login' }">
        Login
      </router-link> |
      <router-link to="/login/new">
        Criar conta
      </router-link>
      ...
    </nav>

    <div class="container">
      <router-view></router-view>
    </div>
  </div>
</template>
```


Vue Router

- ❑ Os dados disponíveis na aplicação podem ser acessados em todos os componentes utilizando o `$root`

```
export default {
  props: ['item'],

  data() {
    return {
      error: {},
      success: false,
      httpOptions: {
        baseUrl: this.$root.config.url,
        headers: {
          'Accept': 'application/json',
          'Content-Type': 'application/json',
          'Authorization': 'Bearer ' + this.$root.credentials.token
        }
      }
    },
  },
  ...
}
```

Vue Router

- ❑ A navegação também pode ser realizada de forma programada
- ❑ A função *replace* substitui a URL atual; a função *push* adiciona a URL no histórico de navegação
- ❑ Por default, a navegação começa no componente associado ao diretório “/”

```
<template>
  <a class="link" @click="logout">Logout</a>
  ...
</template>

<script>
  export default {
    ...,

    methods: {
      logout: function() {
        this.$root.credentials = null;
        this.$router.replace('/');
      }
    }
  }
</script>
```

Vue Router: parâmetros

- ❑ É possível definir rotas que dependem de parâmetros
- ❑ Para passar o parâmetro, a função *push* deve receber um objeto que tem o nome da rota e os seus parâmetros

```
{  
  path: '/details/:id',  
  name: 'details',  
  component: details  
}  
  
-----  
  
methods: {  
  goDetails(id) {  
    this.$router.push({name: 'details', params: {id: id}})  
  }  
}
```

Vue Router: parâmetros

- ❑ A tag *router-link* também pode passar parâmetros pelo atributo *:to*

```
<router-link :to="{ name: 'details', params: { id: 1 } }">  
  Produto #1  
</router-link>
```

Vue Router: parâmetros

- ❑ O valor do parâmetro é geralmente transferido para uma variável na seção de dados e deste ponto pode ser utilizado no template e nos métodos do componente

```
export default {  
  name: 'details',  
  
  data() {  
    return{  
      id: this.$route.params.id,  
      title: "details"  
    }  
  }  
  
  ...  
}
```

Vue Router: parâmetros

- ❑ Outra forma de receber os parâmetros é através das propriedades
- ❑ A rota indica que recebe parâmetros usando o atributo **props**
- ❑ O componente destino indica que recebe parâmetros e os nomes destes parâmetros

```
{  
  path: '/details/:id',  
  name: 'details',  
  component: details,  
  props: true  
}
```

```
-----  
  
<template>  
  <div>ID: {{id}}</div>  
</template>
```

```
<script>  
export default {  
  props: ["id"]  
}  
</script>
```

Vue Router: guardas de navegação

- ❑ São métodos chamados antes de uma navegação, que podem ser utilizados para evitar que a navegação aconteça, ou depois de uma navegação, que podem ser utilizados para inicializar o novo componente
- ❑ Podem ser definidos de forma global para o roteador no **main.js** ...

```
router.beforeEach((to, from, next) => {  
  // Código antes do router entrar na nova rota  
  next();  
});  
  
router.afterEach((to, from) => {  
  // Código após do router entrar na nova rota  
});
```

Vue Router: guardas de navegação

- ❑ Podem ser definidos para cada rota ...

```
routes: [  
  {  
    path: "/",  
    component: Home,  
    beforeEnter: (to, from, next) => {  
      // Código antes do router entrar nesta rota  
    }  
  }  
],
```

Fonte: <https://www.origamid.com/slide/vue-js-completo/#/0703-navigation-guards>

Vue Router: guardas de navegação

- ❑ Ou podem ser definidos nos componentes ...

```
<script>
export default {
  beforeRouteEnter(to, from, next) {
    // Antes de entrar no router, não tem acesso ao this do novo componente
  },
  beforeRouteUpdate(to, from, next) {
    // Método chamado quando um router já ativo é atualizado
  },
  beforeRouteLeave(to, from, next) {
    // Antes de sair do router
    // next("/") é possível passar outros caminhos para o next()
  }
};
</script>
```

Tópicos avançados

- ❑ O Vue Router oferece outros tópicos que podem ser úteis em certos projetos:
 - Router active class
 - Rotas aninhadas
 - Transições
 - Redirecionamento
 - Lazy loading