05

Configurando rotas

Transcrição

Para que o VueRouter saiba qual componente carregar precisamos registrar rotas para esses componentes. Rotas nada mais são que endereços especiais que são interceptados pelo `VueRouter` e a partir do endereço ele decidirá qual componente deve ser exibido em `App` que é o primeiro componente a ser exibido em nossa aplicação.

Vou estipular que os seguintes endereços serão válidos:

```
http://localhost:8080/#/
http://localhost:8080/#/cadastro
```

O primeiro carregará o componente `Home` e o segundo o componente `Cadastro`. Você deve estar achando estranho o `#` no endereço. Ele é importante, porque eles fazem com que o browser não dispare uma requisição para o servidor, pois não é uma URL válida. No entanto, sendo algo totalmente válido para o `VueRouter`, ele extrairá a informação que vem logo após o `#` para saber qual componente carregar. Ele faz um dê para entre o pedaço da url que vem logo após o `#` com o seu respectivo componente.

Sendo assim, é uma boa prática declarar as rotas da aplicação em um arquivo em separado. Vamos criar o arquivo `alurapic/src/routes.js`. Nele exportaremos uma constante que um um array:

```
// alurapic/src/routes.js

export const routes = [
  /* rotas aqui */
];
```

Quando queremos exportar o valor de uma variável é necessário usar o prefixo const. Agora, vamos importar os componentes `Home` e `Cadastro` que equivalem a páginas:

```
// alurapic/src/routes.js

import Home from './components/home/Home.vue';
import Cadastro from './components/cadastro/Cadastro.vue';

export const routes = [
```

No array `routes`, precisamos ter um objeto Javascript com as propriedades `path` e `component`. O primeiro é a caminho que identifica o componente, o segundo o componente que será carregado para este caminho presente na url do navegador:

```
// alurapic/src/routes.js

import Home from './components/home/Home.vue';
import Cadastro from './components/cadastro/Cadastro.vue';
```

```
export const routes = [

    { path: '', component: Home },
    { path: '/cadastro', component: Cadastro }

];
```

Veja que para o componente `Home` usamos o `path` como uma string em branco. Esse é o padrão quando queremos acessar o componente como `/#/`. Já `path` do componente `Cadastro` é `/cadastro` que se traduzirá em uma URL como `http://localhost:8080/#/cadastro`.

Mas inda falta mais duas configurações. A primeira, é passar as rotas que configuramos para o `VueRouter`. Para isso, vamos importar `routes` de `routes.js`:

```
// alurapic/src/main.js

import Vue from 'vue'
import App from './App.vue'
import VueResource from 'vue-resource';
import VueRouter from 'vue-router';

// tem que vir entre chaves, porque não é default
import { routes } from './routes';

Vue.use(VueRouter);
Vue.use(VueResource);

new Vue({
  el: '#app',
  render: h => h(App)
})
```

Agora que importamos a rota, vamos criar uma instância de `VueRouter` passando como parâmetro um objeto JavaScript com a propriedade `routes` que deve receber como parâmetro as rotas que importamos. No caso, tanto a propriedade quando as rotas importadas possuem o mesmo nome:

```
// alurapic/src/main.js

import Vue from 'vue'
import App from './App.vue'
import VueResource from 'vue-resource';
import VueRouter from 'vue-router';

import { routes } from './routes';

Vue.use(VueRouter);

const router = new VueRouter({
  routes : routes
});

Vue.use(VueResource);
```

```
new Vue({
  el: '#app',
  render: h => h(App)
})
```

Em ES6, quando o valor e a propriedade possuem o mesmo nome, podemos simplesmente fazer assim:

```
// alurapic/src/main.js

import Vue from 'vue'
import App from './App.vue'
import VueResource from 'vue-resource';
import VueRouter from 'vue-router';

import { routes } from './routes';

Vue.use(VueRouter);

const router = new VueRouter({
  routes
});

Vue.use(VueResource);

new Vue({
  el: '#app',
  render: h => h(App)
})
```

Agora que temos efetivamente nossas rotas, precisamos passá-la como parâmetro para a view instance, aquela que renderiza nosso componente `App` :

```
// alurapic/src/main.js

import Vue from 'vue'
import App from './App.vue'
import VueResource from 'vue-resource';
import VueRouter from 'vue-router';

import { routes } from './routes';

Vue.use(VueRouter);

const router = new VueRouter({
  routes : routes
});

Vue.use(VueResource);

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

Como o nome da propriedade tem o mesmo nome da nossa variável, podemos fazer apenas `router` ao invés de `router: router`.

Por fim, precisamos usar uma diretiva especial do `VueRouter`, uma que indica em que lugar do template de `App` os componentes serão carregados. Essa diretiva se chama `router-view`:

```
<!-- alurapic/src/App.vue -->
<template>
  <div class="corpo">

    <router-view></router-view>

  </div>
</template>

<script>
</script>

<style>

  .corpo {
    font-family: Helvetica, sans-serif;
    margin: 0 auto;
    width: 96%;
  }
</style>
```

Quando o CLI rodando, veja que nossa aplicação será aberta automaticamente na URL `http://localhost:8080/#/`. Se quisermos acessar a página de cadastro, fazemos `http://localhost:8080/#/cadastro`.

Por fim, não há nada de errado com o `#` no endereço, é algo completamente válido e muito usado. No entanto, podemos removê-lo usando o modo `history` do `VueRouter`. No entanto, para este modo funcionar, seu backend que compartilha sua aplicação em Vue deve retornar sempre `index.html` para todos para qualquer endereço que chegar até ele, inclusive deve retornar `index.html` para páginas de erro. O Vue CLI já faz isso por padrão, mas se você for hospedar sua aplicação seja lá onde for, lembre-se desse detalhe.

Para ativarmos o modo `history` basta adicionarmos a propriedade `mode` com o valor `history` na instância de `VueRouter`.

```
import Vue from 'vue'
import App from './App.vue'
import VueResource from 'vue-resource';
import VueRouter from 'vue-router';

import { routes } from './routes';

Vue.use(VueRouter);

// adicionando a propriedade mode com o valor history.

const router = new VueRouter({
  routes,
  mode: 'history'
```

```
});

Vue.use(VueResource);

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

Veja que agora podemos acessar `Home` através de `http://localhost:8080/` e `Cadastro` através de `http://localhost:8080/cadastro` . Esses endereços não dispararam uma requisição para o servidor e serão interceptados pelo VueRouter para saber qual componente carregar.