

UNIVERSIDAD AUTÓNOMA DE MADRID

DEPARTAMENTO DE INFORMÁTICA

Computer Systems Project

Assignment 2.3

Roberto MARABINI
Alejandro BELLOGÍN

Changelog

Version ¹	Date	Author	Description
1.0	10.10.2022	RM	First version.
1.1	17.11.2022	RM	Code formatting.
1.2	5.12.2022	RM	Renumbering assignment 3 -> 2
1.3	12.12.2022	AB	Translation to English
1.4	28.1.2023	RM	Review before upload to <i>Moodle</i>

¹Version control is made using 2 numbers $X.Y$. Changes in Y denote clarifications, more detailed descriptions of some aspect, or translations. Changes in X denote deeper modifications that either change the provided material or the content of the assignment.

Contents

1	Goal	3
2	Creating the project	3
3	Analyzing the created files	3
4	Creating the components	5

1 Goal

When we create *Vue.js* applications, we often have several pages we want our user to visit. To manage such browsing without connecting to the backend (server), we use a “router”. In *Vue.js*, vue-router is the “routing” library. We describe its use in a simple case next.

2 Creating the project

Create a new *Vue.js* project called `twopages`, this time selecting the router option. That is:

```
npm init vue@3.2 twopages
# answer as follows the questions

Add TypeScript? No** / Yes
Add JSX Support? No** / Yes
Add \vuejs Router for Single Page Application development? No /
  ➞ Yes** <<<<<<
Add Pinia for state management? No** / Yes
Add Vitest for Unit Testing? No** / Yes
Add Cypress for End-to-End testing? No** / Yes
Add ESLint for code quality? No** / Yes
Add Prettier for code formatting? No** / Yes
```

Do not forget to initialize the project as we mentioned (`cd twopages && npm install`)

3 Analyzing the created files

Let us start discussing the main differences between the files created by default when the router is added and when is not.

main.js

If you remember, *Vue.js* applications are initialized in the `main.js` file. Now you may see that, after initializing the application, the router module is added (`app.use(router)`).

Listing 1: Content of file `main.js`, the lines related to the router are marked with a string of `#`.

```
//src/main.js
import { createApp } from 'vue'
import App from './App.vue'
import router from './router' #####
import './assets/main.css'
const app = createApp(App)
app.use(router) #####
app.mount('#app')
```

index.js

This file is similar to `urls.py` from *Django* and includes a “mapping” between URLs and `vuejs` files. The one that comes by default is quite complex, remove it and copy the following code:

```
//src/router/index.js
import { createRouter, createWebHistory } from 'vue-router'
import HomeView from '../views/HomeView.vue'

const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: [
    {
      path: '/',
      name: 'home',
      component: HomeView
    }
  ]
})
```

```
    },
    {
      path: '/page-one',
      name: 'pageone',
      component: () => import('../components/PageOne.vue')
    },
    {
      path: '/page-two',
      name: 'pagetwo',
      component: () => import('../components/PageTwo.vue')
    },
  ],
})

export default router
```

where the three mapping for URLs `/`, `/page-one`, and `/page-two` are created.

4 Creating the components

We still have not created the components `page-one` and `page-two`. Create two files called `PageOne.vue` and `PageTwo.vue` in the folder `components` with the content:

Listing 2: Template to create files `PageOne.vue` and `PageTwo.vue`. Change `XXX` to `ONE` or `TWO` to identify both pages.

```
<template>
  <!-- cambia XXX por ONE or TWO -->
  <h1>Hi! I am page XXX! </h1>
</template>

<script>
  export default {}
</script>
```

similarly, we shall simplify the file that is loaded by default (`App.vue`) and the application that is loaded by default `HomeView.vue` so that the effect of our changes are more evident in the configuration file of the router.

Listing 3: Content of `App.vue` file. The associated components to each URL will be introduced between tags `<router-view></router-view>`.

```
<script setup>
import { RouterLink, RouterView } from 'vue-router'
</script>

<template>
  <div>
    <h1>Example of Router usage</h1>
    <router-view></router-view>
  </div>
</template>

<script>
  export default {}
</script>
```

and, finally, let us simplify the view that is loaded by default `HomeView.vue`.

Listing 4: Content of `Homeview.vue` file.

```
<template>
  <main>
    <h1>Home Page</h1>
  </main>
</template>
```

now you can start the server and connect to URLs: `http://localhost:3000/`, `http://localhost:3000/one-page`, and `http://localhost:3000/two-page` and you will obtain the following result:



Figure 2: Pages corresponding to the URL `http://localhost:3000/page-two` showing the links to the URLs `http://localhost:3000/page-one` and `http://localhost:3000/page-two`.