The Medium App
An app designed for readers.
OPEN IN APP

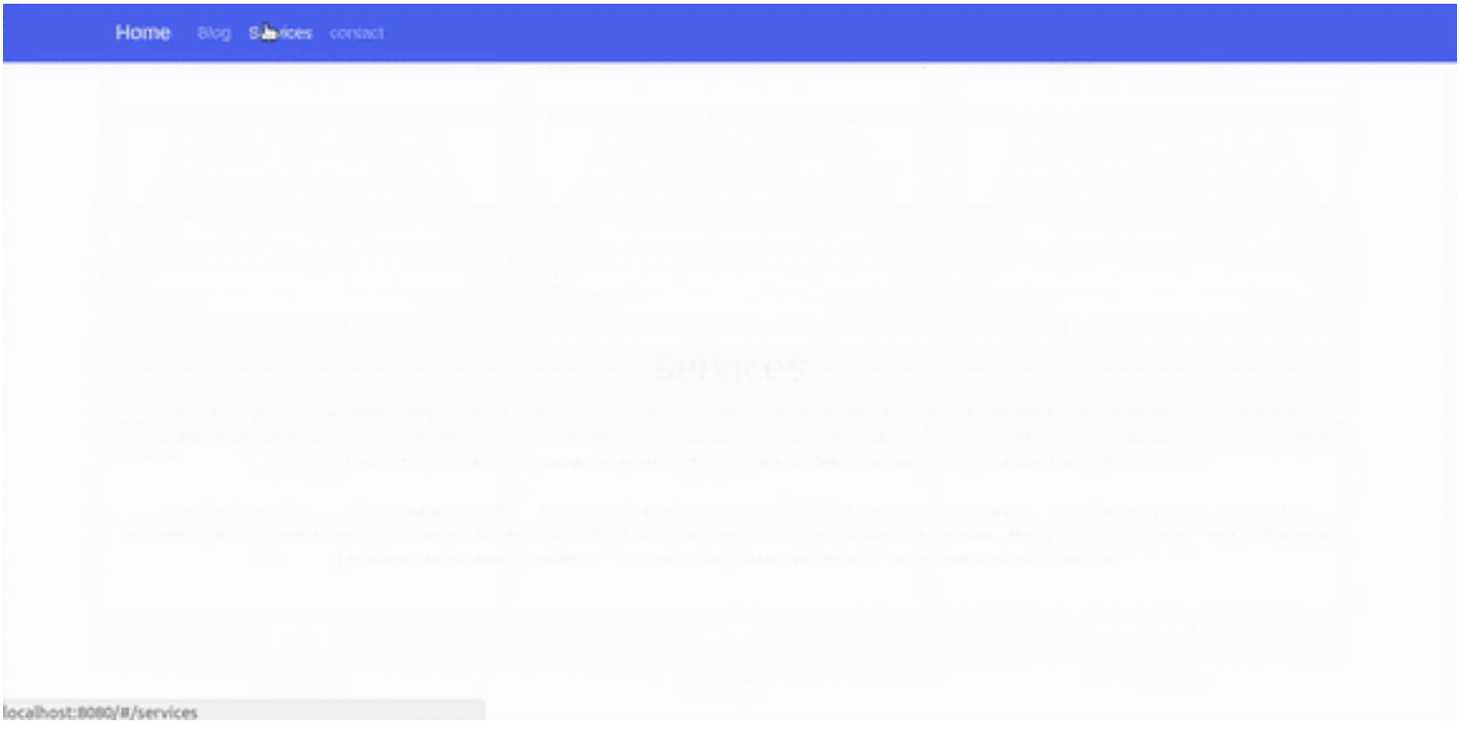
Get started

Understand Routing in Vue.js With Examples

SaidHayani@

Jun 7, 2018 · 9 min read

Zeolearn copyright



Vue.js is a great JavaScript Framework created by Evan You ,it's used to build Single web page and flexible components ,and it's the most skill required in Front End Web development ,you can learn more about Vue.js here .

. . .

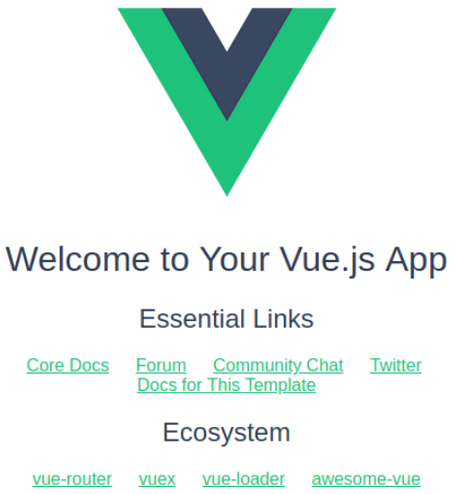
Vue.js provide bunch of features to build a reusable web components,Routing is one of those methods,it allow user switch between pages without page refreshing the thing that make the navigation easy and really nice in your web application ,so in this Article we are going to explain how Vue.js Routers work by building a Vue-template as example .

So, let's **get started with our Vue.js Router** project by installing and creating a new Vue.js project. We need to have node.js installed. We shall be using vue-cli to generate a new Vue.js project. Follow the steps given below-

Type the following code in your terminal run:

```
vue init webpack vue-router

//
cd vue-router
//
npm run dev
```

Browse to http://localhost:8080



Open the app in your text editor, inside components folder open `HellowWorld.vue` file and follow these steps:

-rename `HellowWorld.vue` with `home.vue` ,remove all the code and replace it with this one .

```
<template>
  <div class="home">
    <h1>Home</h1>
  </div>
</template>

<script>
export default {
  name: 'home',
  data () {
    return {
      msg: 'Welcome to Your
Vue.js App'
    }
  }
}
</script>

<!-- Add "scoped" attribute to
limit CSS to this component
only -->
<style scoped>

</style>
```

-And go to `index.js` inside **router** folder and replace `HelloWorld` with `home`

```
import Vue from 'vue'
import Router from 'vue-
router'
import home from
'@/components/home'

Vue.use(Router)

export default new Router({
  routes: [
    {
      path: '/',
      name: 'home',
      component: home
    }
  ]
})
```

`App.vue` file should look like this !

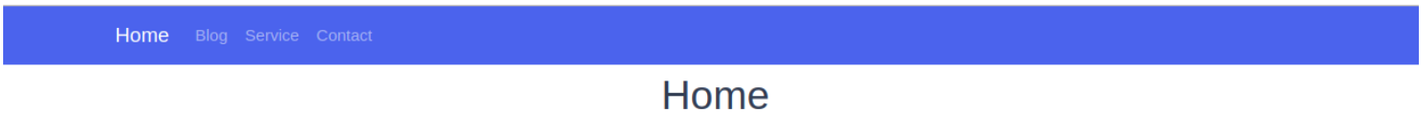
```
<template>
  <div id="app">

    <router-view/>
  </div>
</template>

<script>
export default {
  name: 'App'
}
</script>

<style>
#app {

}
</style>
```

And Now let’s write our code !!

We are now going to add a Bootswatch template. You can choose any template you like. I shall choose a Cosmo, click Ctrl

+ U to view code source and just copy the `Navbar`, we just need navbar, and paste this code into App.vue component.

Here we are 😃



The next is we gonna create 3 other components `Blog`, `Services` and `Contact`

Inside the components folder create new file and name it with `blog.vue` and push this code into it

```
<template>
 <div class="blog">
  <h1>{{blog}}</h1>
 </div>
</template>
<script>
 export default{
  name:'blog',
  data (){
   return{
    title:'Blog'
   }
  }
 }
</script>

<style scoped>

</style>
```

If you want to do the same thing for the service and contact component, you must have these files in your component folder:

- home.vue
- blog.vue
- services.vue
- contact.vue

Routers config

Now after having four components, we need to configure the routers so that we can navigate between these components.

So How we can navigate to each components using the routers?

We need to learn the rule of Routing. Now, we have to do some modifications inside the router folder, open `index.js`

- First, import your components into index.js

Import all the components using `import` method.

```
import home from
'@/components/home'
import blog from
'@/components/blog'
import services from
'@/components/services'
import contact from
'@/components/contact'
```

- Second import vue and Router module from vue-router module

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

// use router
Vue.use(Router)
```

if you have installed vue with vue-cli, you will have vue-router module imported by default

- Finally, inside the router folder, we have to configure the Routers to make it work. and The Router method takes an Array of objects that takes each component's properties:

```
export default new Router({
  routes: [
    {
      path: '/',
      name: 'home',
      component: home
    },
    {
      path: '/blog',
      name: 'blog',
      component: blog
    },
    {
      path: '/services',
      name: 'services',
      component: services
    },
    {
      path: '/contact',
      name: 'contact',
      component: contact
    }
  ]
})
```

- `path` : mean the path of the component
- `name` : name of the component
- `component` : the view component

To make any component as the default component set slash('/') to the path property,

```
path:'/'
```

In our example, we set the home page as default page.now when you open the project in the browser the first page will appear is the home page.

```
{
path:'/',
name:'home',
component:home
}
```

The vue-router has more advanced parameters and methods, but we are not jumping into this section in this meantime, and this is the list of properties and method that you can use with vue-router :

- Nested routers
- Named view
- Redirect and Alias
- Navigation Guard

- Router instance

Now you can browse to any components by typing the name of the component!!

router-link

Now we are going to make the navigation through the Navbar that we created using the router-link element.

To do that we should replace `</a>`

element by ‘<router-link> </router/link>’ like this:

```
<li class="nav-item">
  <router-link class="nav-
link" to="/blog">Blog</router-
link>
</li>
<li class="nav-item">
  <router-link class="nav-
link"
to="/services">Services</route
r-link>
 </li>
<li class="nav-item">
   <router-link class="nav-
link"
to="/contact">contact</router-
link>
 </li>
```

The router-link take `to='path'` attribute that take the path of the component as value.

Router-view

You will find `<router-view>` tag in the `App.vue` file, it's basically the view where the components are rendered, it's like the main div that contain all the components and it return the component that matches the current route, we will have a discussion with `route-view` in the next part when we use animation transition.

Using the parameters inside the routers

At this part we will use parameters to navigate to specific components, the parameters make The Routing dynamic.

To work with parameters we are gonna create a list of product, and an array of data, so when you click on the link of any product it will gonna take us to the page details through a parameter.

At this situation, we are not going to use a database or API to retrieve products data, so what have to do is create an Array of products that we suppose to be a database.

Inside `home.vue` component put the Array within data() method just like this:

```
export default {
  name: 'home',
  data () {
    return {
```

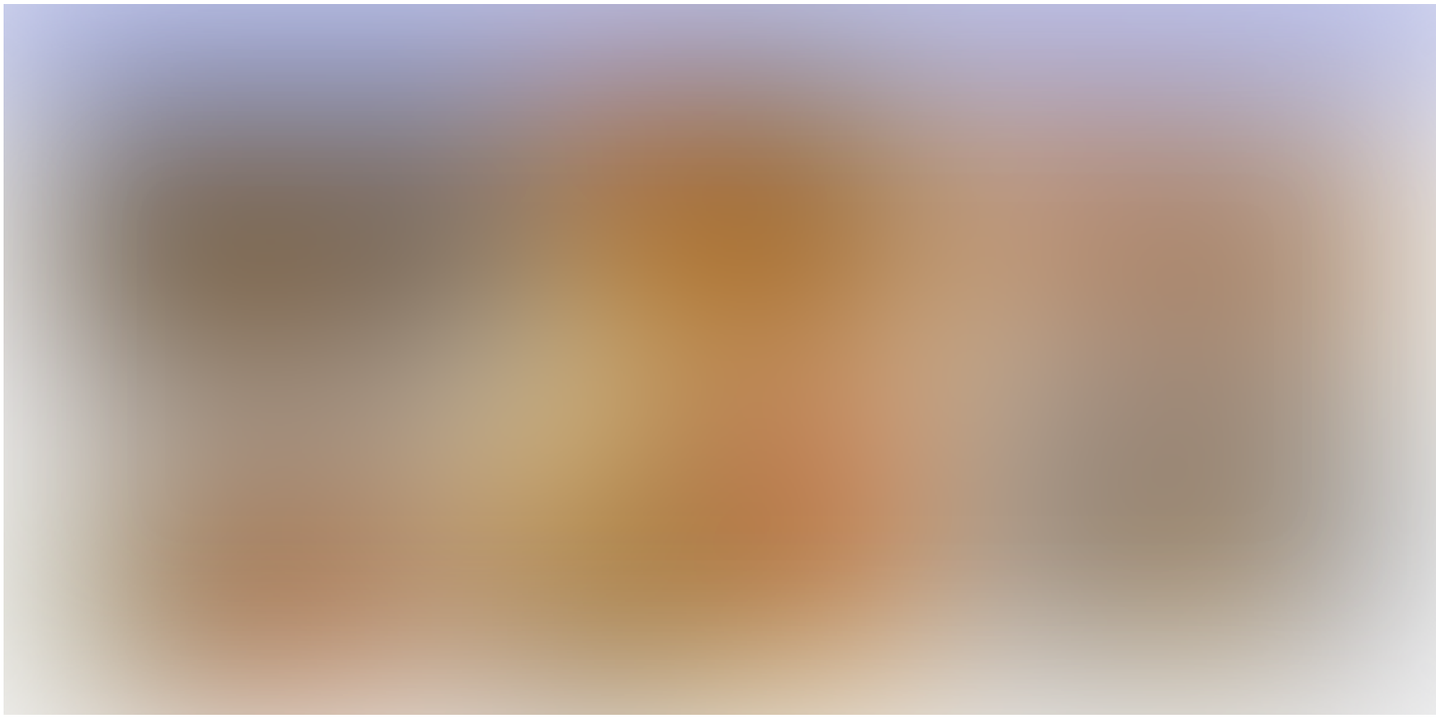
```
      title: 'Home',
      products:[
      {
        productTitle:"ABCN",
        image       :
require('../assets/images/prod
uct1.png'),
        productId:1
      },
      {
        productTitle:"KARMA",
        image       :
require('../assets/images/prod
uct2.png'),
        productId:2
      },
      {
        productTitle:"Tino",
        image       :
require('../assets/images/prod
uct3.png'),
        productId:3
      },
      {
        productTitle:"EFG",
        image       :
require('../assets/images/prod
uct4.png'),
```

```
        productId:4
      },
      {
        productTitle:"MLI",
        image       :
  require('../assets/images/prod
  uct5.png'),
        productId:5
      },
      {
        productTitle:"Banans",
        image       :
  require('../assets/images/prod
  uct6.png'),
        productId:6
      }
      ]
    }
  }
}
```

Then fetch and loop into Products Array using `v-for` directive .

```
<div class="row">
      <div class="col-md-4
col-lg4" v-for="(data,index)
in products" :key="index">
        <img :src="data.image"
class="img-fluid">
         <h3>
{{data.productTitle}}</h3>
      </div>
    </div>
```

The Result:



To navigate to details component we have first to add a click event

```
<h3 @click="goTodetail()" >
{{data.productTitle}}</h3>
```

Then add methods

```
methods:{
  goTodetail() {

this.$router.push({name:'detai
ls'})
  }
```

if you click the title it will return undefined because we didn't create the details component yet so let's create one:

details.vue

```
<template>
 <div class="details">
  <div class="container">
   <h1 class="text-primary
text-center">{{title}}</h1>
  </div>
 </div>
</template>
<script>
 export default{
  name:'details',
  data(){
   return{
    title:"details"
   }
  }
 }
</script>
```

Now we can navigate without getting an error 😃

Now how we can browse to the details page and get the matched data while we don’t have a database ?

We gonna use the same products Array in details component, so we can match the id that comes from the URL

```
products:[
      {
        productTitle:"ABCN",
```

```
        image       :
require('../assets/images/prod
uct1.png'),
        productId:1
      },
      {
        productTitle:"KARMA",
        image       :
require('../assets/images/prod
uct2.png'),
        productId:2
      },
      {
        productTitle:"Tino",
        image       :
require('../assets/images/prod
uct3.png'),
        productId:3
      },
      {
        productTitle:"EFG",
        image       :
require('../assets/images/prod
uct4.png'),
        productId:4
      },
      {
        productTitle:"MLI",
```

```
        image       :
require('../assets/images/prod
uct5.png'),
        productId:5
      },
      {
        productTitle:"Banans",
        image       :
require('../assets/images/prod
uct6.png'),
        productId:6
      }
      ]
```

First, we have to set the id to `goTodetail()` method as parameter,

```
<h3
@click="goTodetail(data.produc
tId)" >{{data.productTitle}}
</h3>
```