

# Module 3-15

## VUE Routing

# What is Routing?

— — —

- Routing allows users to be redirected to a certain component via a URL.
- Remember MVC Spring RequestMappings? Similar idea.

# The router.js file overview

---

```
import Vue from 'vue'
import Router from 'vue-router'
import Home from './views/Home.vue'
import About from './views/About.vue'
Vue.use(Router)

export default new Router({
  mode: 'history',
  base: process.env.BASE_URL,
  routes: [
    {
      path: '/',
      name: 'home',
      component: Home
    },
    {
      path: '/books',
      name: 'books',
      component: Books
    }
  ]
})
```

To define a route, a router.js file is needed. This file is a peer of App.vue.

- There is some boiler plate code beyond the scope of this class, you can carry those over for now, they are highlighted in blue.
- Our focus will be on the sections in red, which we will define.

# The router.js file: importing views

---

We need to first define the components a user can potentially be routed to, this can be achieved through imports:

```
import Home from './views/Home.vue'  
import About from './views/About.vue'  
import Books from './views/Books.vue'
```

Note that in this example the components are in a folder called views, this should make no difference, Home, About, and Books are VUE components.

# The router.js file: importing components.

---

Next, we define the routes, these physically map the view components we imported.

```
import Home from './views/Home.vue'  
import About from './views/About.vue'  
import Books from './views/Books.vue'
```

```
routes: [  
  {  
    path: '/',  
    name: 'home',  
    component: Home  
  },  
  {  
    path: '/about',  
    name: 'about',  
    component: About  
  },  
  {  
    path: '/books',  
    name: 'books',  
    component: Books  
  }  
]
```

Each route is a JSON object, comprised of three key value pairs:

- **path:** what the user types into the URL
- **name:** This is how we will refer to the route within App.vue (**this is optional**)
- **component:** The component that was imported in, that the user will be redirected to

# Adding router links to App.vue

— — —

We are almost done! The last step is want to define the routes within App.vue.

```
<template>
  <div id="app">
    <header>
      <ul class="nav">
        <router-link :to="{ name: 'home' }" tag="li" exact>Home</router-link>
        <router-link :to="{ name: 'about' }" tag="li">About the Author</router-link>
        <router-link :to="{ name: 'books' }" tag="li">Related Books</router-link>
      </ul>
    </header>
    <router-view class="content"/>
  </div>
</template>
```

Again, this is some boiler plate code highlighted in blue that we can just carry over, what matters is what's in red.

# Adding the router links to App.vue

---

This is the basic structure of a router-link

```
<router-link :to="{ name: 'home' }" tag="li" exact>Home</router-link>
```

This should match up to the name of the route specified in router.js

You can define the type of html element you want rendered as the link here.

**Let's Implement These Routes!**



# Dynamic Routing

— — —

- Sometimes data is encapsulated within a URL path:
  - Consider the following URL: (account/**135**)
  - The value 135 could very well be an ID that is associated with a “row” of data.
- The purpose of dynamic routing is to implement these URL parameters.
- Remember MVC Spring PathParams? Similar idea.

# Dynamic Routing : router.js

---

We must first implement a route , where the path parameter contains a placeholder for the path variable.

```
{  
  path: '/users/:id',  
  name: 'user',  
  component: User  
}
```

# Dynamic Routing : Defining the Router Links

— — —

```
<tr v-for="user in users" :key="user.id">
  <td><router-link :to="{name: 'user', params: {id: user.id}}">{{user.id}}</router-link></td>
  <td>{{user.name}}</td>
</tr>
```

Template section

```
users: [
  {
    "id": 1,
    "name": "Leanne Graham"
  },
  {
    "id": 2,
    "name": "Ervin Howell"
  }
]
```

State data

Here we have a v-for that will iterate through every object in the users array, each time it does so it generates a new router-link with its respective id value.

Two links are generated:

- /users/1
- /users/2

**Let's Implement Dynamic Routes!**