Sai gowtham Follow
Sep 17, 2018 · 9 min read

In this tutorial series, you will learn about react router v4 which is a stable version currently used in most react projects.

If you don't know about react first go through the react tutorials

Table of contents

- Programmatically navigate

Introduction

What is react router?

React router is a routing library built on top of the react which is used to create the routing in react apps.

Is react router is static or dynamic?

- Before the react router v4 it is static after v4 it is Dynamic.

In single page apps, there is only single html page.we are reusing the same html page to render the different components based on the navigation.

But in multipage apps, you will get an entirely new page from the server when you navigate.

"If you stuck anywhere in the middle please refer to code repository"

Getting started

How to install the react router?

we are using the create-react-app to create the app.

```
npx create-react-app routing
cd routing
```

To install the react router you need to download the react-router-dom package by running the following commands.

```
npm install react-router-dom

npm start  //to run  dev server
```

Now open the routing folder in your favorite code editor.

If you navigate to the public/index.html you will see a single html file which looks like this.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
    <meta name="theme-color" content="#000000">
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json">
    <link rel="shortcut icon"
href="%PUBLIC_URL%/favicon.ico">
    <title>React App</title>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

Currently, in our app, there is only a single App component.

Let’s create two more components.

users.js

```
import React from 'react'

class Users extends React.Component {
  render() {
    return <h1>Users</h1>
  }
}

export default Users
```

contact.js

```
import React from 'react'

class Contact extends React.Component {
  render() {
    return <h1>Contact</h1>
```

```
    }
}

export default Contact
```

app.js

```
import React from 'react'
class App extends React.Component {
  render() {
    return (
      <div>
        <h1>Home</h1>
      </div>
    )
  }
}

export default App
```

Now our app has three components one is App and the other two are Users and Contact.

Routing

open the index.js file and import the three components (App,Users,Contact)

index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import App from './App'
import Users from './users'
import Contact from './contact'

ReactDOM.render(<App />, document.getElementById('root'))
```

React router gives us three components [Route,Link,BrowserRouter] which help us to implement the routing.

index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import { Route, Link, BrowserRouter as Router } from 'react-
router-dom'
import './index.css'
import App from './App'
import Users from './users'
import Contact from './contact'

ReactDOM.render(<App />, document.getElementById('root'))
```

Let's implement the routing.

In the Route component, we need to pass the two props

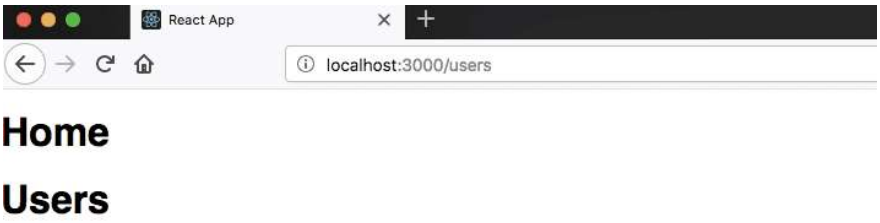
- path: it means we need to specify the path.
- component: which component user needs to see when they will navigate to that path.

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import { Route, Link, BrowserRouter as Router } from 'react-
router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'

const routing = (
  <Router>
    <div>
      <Route path="/" component={App} />
      <Route path="/users" component={Users} />
      <Route path="/contact" component={Contact} />
    </div>
  </Router>
)

ReactDOM.render(routing, document.getElementById('root'))
```

Now if you enter manually localhost:3000/users you will see Users component is rendered.



But still, Home component is also rendered in the screen this happens because of our home path is '/' and users path is '/users' **slash** is same in both paths so that it renders both components to stop this behavior we need to use the exact prop.

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import { Route, Link, BrowserRouter as Router } from 'react-
router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'

const routing = (
  <Router>
    <div>
      <Route exact path="/" component={App} />
      <Route path="/users" component={Users} />
      <Route path="/contact" component={Contact} />
    </div>
  </Router>
)

ReactDOM.render(routing, document.getElementById('root'))
```

Now if you see only users component is rendered on the screen.

Adding Navigation using Link component

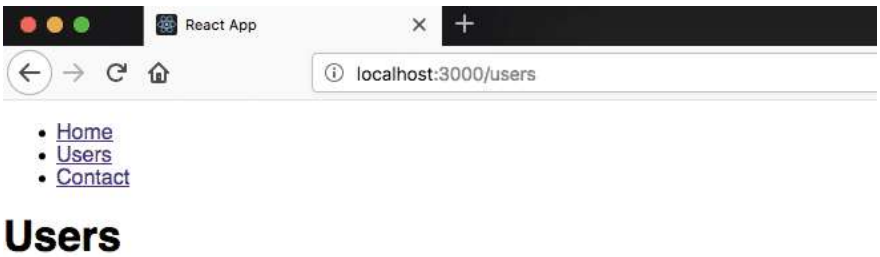
index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import { Route, Link, BrowserRouter as Router } from 'react-
router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'


const routing = (
  <Router>
    <div>
      <ul>
        <li>
          <Link to="/">Home</Link>
        </li>
        <li>
          <Link to="/users">Users</Link>
        </li>
        <li>
          <Link to="/contact">Contact</Link>
        </li>
      </ul>
      <Route exact path="/" component={App} />
      <Route path="/users" component={Users} />
      <Route path="/contact" component={Contact} />
    </div>
  </Router>
)


ReactDOM.render(routing, document.getElementById('root'))
```

After adding navigation you will see the routes are rendered on the screen. if you click on the users you will see url is changing and Users component is rendered.



users

Adding 404 Pages

What is a 404 page?

A 404 page is also called not found page it means when a user navigates to the wrong path that doesn't present in the website we need to show the not found page.

How to add a 404 page in react?

we need to import another component called Switch which is provided by the react router.

What is Switch?

Switch component helps us to render the components only when path matches otherwise it fallbacks to the not found component.

let's create a Not found component.

notfound.js

```
import React from 'react'

const Notfound = () => <h1>Not found</h1>

export default Notfound
```

index.js

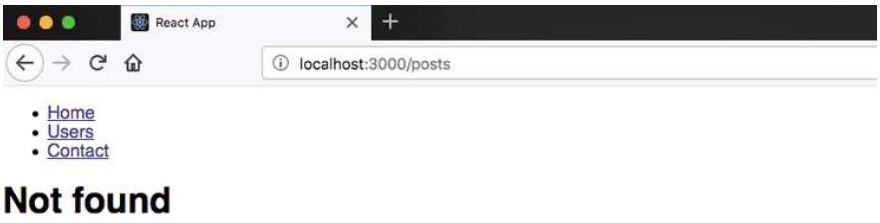
```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import { Route, Link, BrowserRouter as Router, Switch } from
'react-router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'
import Notfound from './notfound'

const routing = (
  <Router>
    <div>
      <ul>
```

```
          <li>
            <Link to="/">Home</Link>
          </li>
          <li>
            <Link to="/users">Users</Link>
          </li>
          <li>
            <Link to="/contact">Contact</Link>
          </li>
        </ul>
        <Switch>
          <Route exact path="/" component={App} />
          <Route path="/users" component={Users} />
          <Route path="/contact" component={Contact} />
          <Route component={Notfound} />
        </Switch>
      </div>
    </Router>
)

ReactDOM.render(routing, document.getElementById('root'))
```

Let's check it now by manually entering wrong path *localhost:3000/posts*.



Url Parameters

Url parameters helps us to render the same component based on its dynamic url like in our Users component assume that they are different users with id 1,2,3.

```
<Route path="users/:id" component={Users} />
```

index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import { Route, Link, BrowserRouter as Router, Switch } from
'react-router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'
import Notfound from './notfound'

const routing = (
  <Router>
    <div>
      <ul>
        <li>
          <Link to="/">Home</Link>
        </li>
        <li>
          <Link to="/users">Users</Link>
        </li>
        <li>
          <Link to="/contact">Contact</Link>
        </li>
      </ul>
      <Switch>
        <Route exact path="/" component={App} />
        <Route path="/users/:id" component={Users} />
        <Route path="/contact" component={Contact} />
        <Route component={Notfound} />
      </Switch>
    </div>
  </Router>
)

ReactDOM.render(routing, document.getElementById('root'))
```

Open users.js file and add console.log(this.props).

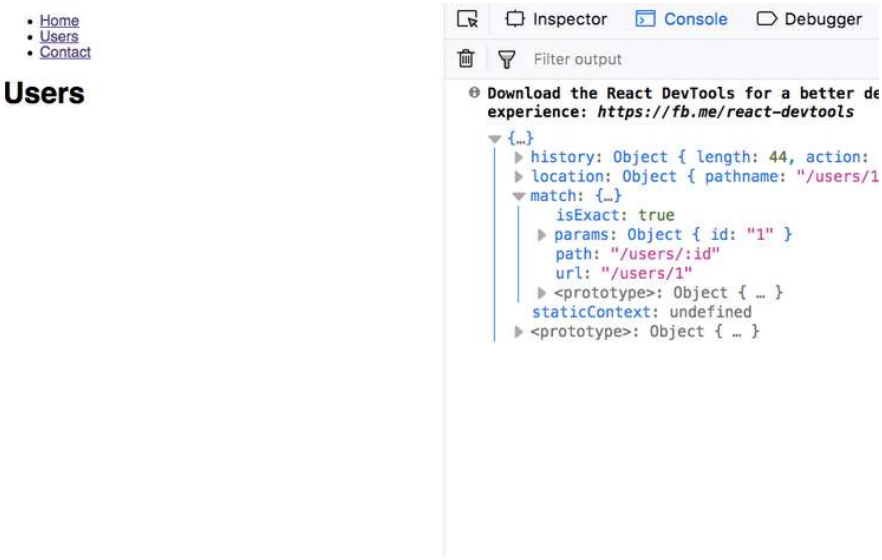
users.js

```
import React from 'react'

class Users extends React.Component {
  render() {
    console.log(this.props)
    return <h1>Users</h1>
  }
}

export default Users
```

Now open your browser and manually type this url localhost:3000/users/1 you will see an object in your console which is passed by the react router.



Have you seen the params object with id 1 the same thing we passed in the url localhost:3000/users/1.

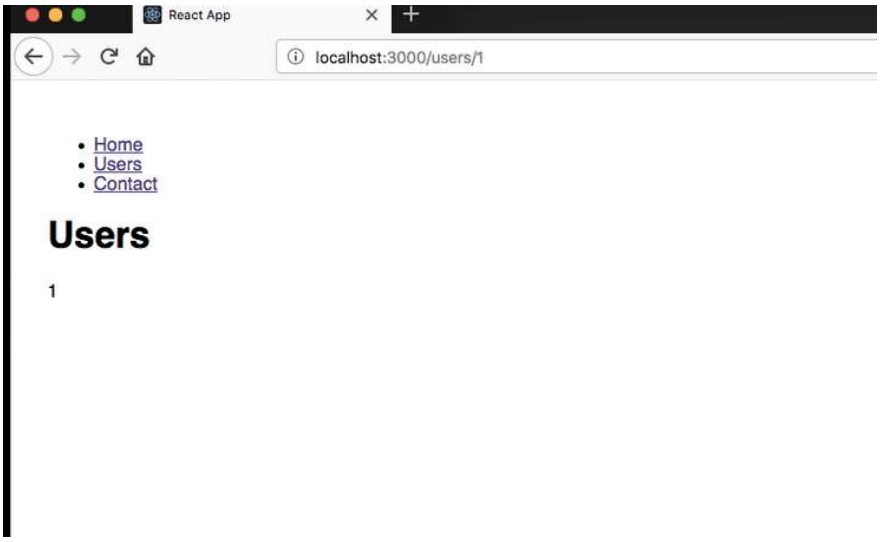
Let's render that id 1 in the screen.

users.js

```
import React from 'react'

class Users extends React.Component {
  render() {
    const { params } = this.props.match
    return (
      <div>
        <h1>Users</h1>
        <p>{params.id}</p>
      </div>
    )
  }
}

export default Users
```



Have you seen now 1 is rendered on the screen?

Nested Routes

Nested routing helps us to render the sub-routes like users/1, users/2 etc.

index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import { Route, Link, BrowserRouter as Router, Switch } from
'react-router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'
import Notfound from './notfound'


const routing = (
  <Router>
    <div>
      <ul>
        <li>
          <Link to="/">Home</Link>
        </li>
        <li>
          <Link to="/users">Users</Link>
        </li>
        <li>
          <Link to="/contact">Contact</Link>
        </li>
      </ul>
      <hr />
      <Switch>
        <Route exact path="/" component={App} />
        <Route exact path="/users" component={Users} />
        <Route path="/contact" component={Contact} />
```

```
        <Route component={Notfound} />
      </Switch>
    </div>
  </Router>
)
ReactDOM.render(routing, document.getElementById('root'))
```

How to implement a nested routing?

In the users.js file, we need to import the react router components because we need to implement the subroutes inside the Users Component.

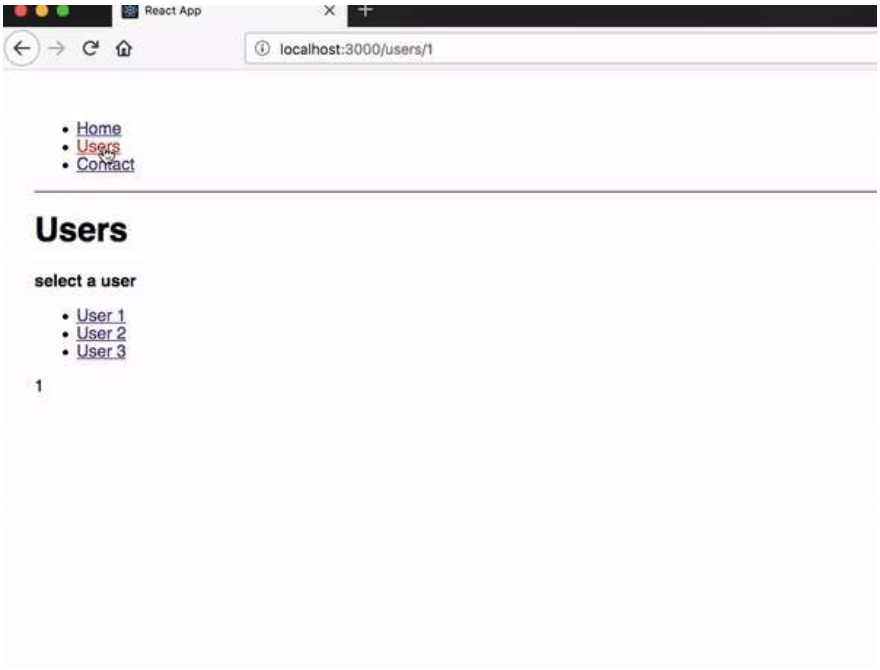
users.js

```
import React from 'react'
import { Route, Link } from 'react-router-dom'

const User = ({ match }) => <p>{match.params.id}</p>

class Users extends React.Component {
  render() {
    const { url } = this.props.match
    return (
      <div>
        <h1>Users</h1>
        <strong>select a user</strong>
        <ul>
          <li>
            <Link to="/users/1">User 1 </Link>
          </li>
          <li>
            <Link to="/users/2">User 2 </Link>
          </li>
          <li>
            <Link to="/users/3">User 3 </Link>
          </li>
        </ul>
        <Route path="/users/:id" component={User} />
      </div>
    )
  }
}

export default Users
```



active styles using NavLink

NavLink

It is used to style the active routes so that user knows on which page he or she is currently browsing on the website.

What is the difference between NavLink and Link?

The link is used to navigate the different routes on the site. But NavLink is used to add the style attributes to the active routes.

In our routing app, we have three routes which are [home,/users,/contact] Let's style them using NavLink.

We need to add a new prop called activeClassName to the NavLink component so that it applies that class whenever the route it is active.

add these styles in the index.css file

index.css

```
body {
margin: 0;
```

```
padding: 2rem;
font-family: sans-serif;
}


.active{
   color:red;
}
```

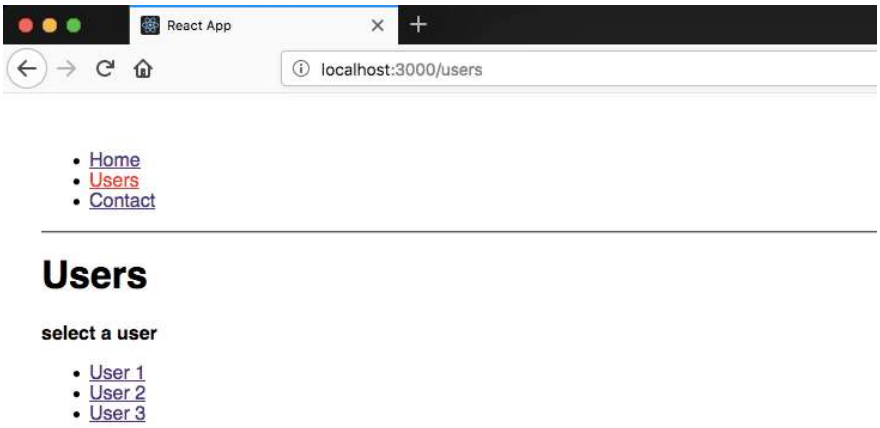
index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
import {
  Route,
  NavLink,
  BrowserRouter as Router,
  Switch,
} from 'react-router-dom'
import App from './App'
import Users from './users'
import Contact from './contact'
import Notfound from './notfound'


const routing = (
  <Router>
    <div>
      <ul>
        <li>
          <NavLink exact activeClassName="active" to="/">
            Home
          </NavLink>
        </li>
        <li>
          <NavLink activeClassName="active" to="/users">
            Users
          </NavLink>
        </li>
        <li>
          <NavLink activeClassName="active" to="/contact">
            Contact
          </NavLink>
        </li>
      </ul>
      <hr />
      <Switch>
        <Route exact path="/" component={App} />
        <Route path="/users" component={Users} />
        <Route path="/contact" component={Contact} />
        <Route component={Notfound} />
      </Switch>
    </div>
  </Router>
)


ReactDOM.render(routing, document.getElementById('root'))
```

Now you can see a red color is applied to the active routes.



Programmatically navigate

What is Programmatic navigation?

It means we need to redirect the user when an event happens on that route.

For example, when a user is successfully logged in he or she will be redirected to the home page.

How to Navigate Programmatically in react router?

To navigate programmatically we need to take the help of history object which is passed by the react router.

Let's add a contact form to our Contact component.

There is a push method available in the history object by using the push method we are redirecting the user to the Home page whenever a user submits the form.

contact.js

```
import React from 'react'

class Contact extends React.Component {
  onSubmit = () => {
    this.props.history.push('/')
  }

  render() {
    return (
      <form>
        <input placeholder="name" type="name" />
        <input placeholder="email" type="email" />
        <button onClick={this.onSubmit}>Submit</button>
      </form>
    )
  }
}

export default Contact
```

Code repository

Other interesting articles

1. React Redux tutorial with examples

2. Render props in react

3. Animations in React

originally published at reactgo.com