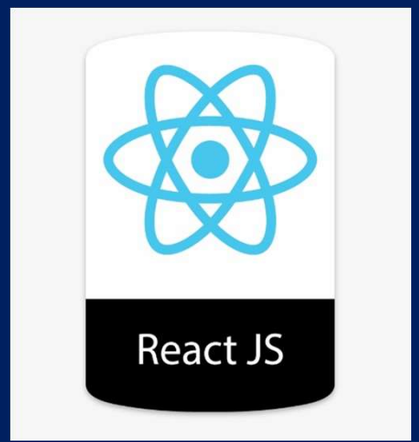
10

React Routing



Routing

Routing

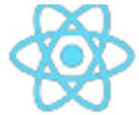
- Navigating from one page view to another is critical in an Single Page Application.
- Single Page Applications dynamically rewrite the current page instead of requesting entirely new pages from the server.
- SPA helps make web applications behave like desktop or mobile applications.
- Router provides a mechanism to distinguish between each page of the application.
- React Router is used to implement routing.

React Router

- The React Router is installed by using npm

```
npm install --save react-router-dom
```

- In the context of a browser environment React Router provides two kinds of routes
  - **BrowserRouter**
    - Helps build classic URLs and is recommended by default
    - May require additional server configuration to handle dynamic requests
  - **HashRouter**
    - Builds URLs with the hash
    - Can be a good solution for static websites



React Router

- The three important components when working with the React Router are:
  - BrowserRouter
    - Wraps all Route components
  - Link
    - Used to generate links to the routes
  - Route
    - Responsible for showing or hiding the component they contain

BrowserRouter

- A `BrowserRouter` component can have only one child element.
- The routing feature generally has to work for the entire application and therefore the `App` component is placed inside the `BrowserRouter` in `index.js`

```
import {BrowserRouter as Router} from 'react-router-dom';

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

Structuring the application

- The `App` component can be used to structure the application for modularity and comprehension.

```
<div className='app'>
    <h1>React Router Demo</h1>
    <NavBar />
    <Outlet />
</div>
```

- The `NavBar` component can contain the navigation menu and `Outlet` component can be used to depict the content of each view.

Link

- The `Link` component is used to trigger new routes.
- The `Link` component can be added to point at different routes with the `to` attribute.
- `NavLink` is a special version of the `Link` component that helps add styling attributes to the rendered element if the URL is matched.

```
import { NavLink } from 'react-router-dom';

<ul>
    <li><NavLink to='/'>Home</NavLink></li>
</ul>
```

Route

- The `Route` component is used to create a route.
- It renders a component when the URL matches the Route's path.
- Multiple Routes can be grouped inside a `Switch` component to render only the first child `Route` that matches the current URL.

```
<Route path='/' component={Home}></Route>
```

- The `Route` component expects a path `prop` that describes the `path` that the route matches.
- The `component` prop is used to specify that component that should be rendered when a route matches.
- In order to have the render only when the path is an exact match use the `exact` prop on each of the routes

The views

- All the components referred to in the `Route` return JSX
- This JSX will be rendered in the DOM when the routes for each component match the current URL.

```
const Home = () => {
      return(
            <div className='home'>
                  <h2>Welcome to our home on the web</h2>
                  <p> Feel free to browse around and learn more
                        about us.</p>
            </div>
            )
      }
```

Dynamic routes using Parameters

- URL parameters help render the same component based on its dynamic URL.

```
<Route exact path='/team/:id' component={Team}></Route>

<li><NavLink to='/team/2'>Team</NavLink></li>

const Team = ({match}) => {
const memberID = match.params.id
```