Chris Parker

How to Integrate Bootstrap with ReactJS

Chris Parker

Jan 10, 2020 • 16 Min read • 32,190 Views

Jan 10, 2020 • 16 Min read • 32,190 Views

Web Development React

Introduction

👍 98

Introduction

We'll focus on the following skills in this guide:

1. Building UI in a React-based web app and using Bootstrap's look and feel.
2. Using reactstrap to develop a React app for listing contacts.

React is the most commonly used JS framework for developing interactive web apps. But since it is a view library, it does not come with techniques to build designs that are responsive and very intuitive. In order to overcome this, we can use Bootstrap, which is a front-end design framework.

Why Bootstrap Components Cannot Be Included with React

Adding a HTML tag like `<link rel=”stylesheet”/>` to an HTML file like index.html is an easy task, but it’s not the same when adding Bootstrap to React. The reason is that Bootstrap relies on jQuery to run particular user interface components. And jQuery manipulates the DOM directly, which contradicts the declarative approach of React. We can use the simple vanilla Bootstrap stylesheet for web applications if the requirement is limited to a receptive 12-column grid or components that do not involve jQuery. Otherwise, there are different libraries that can enable the use of Bootstrap along with React.

We'll examine both methods so that we can choose which one is best suited for a given scenario. We'll focus on the implementation details for integrating Bootstrap with React app.

How to Setup Bootstrap Stylesheet with React

We will be using Create React App CLI to get started with our React project. It does not require any configuration to start.

The following commands can be used install Create React App and to start the server in development mode:

```javascript
$ create-react-app my-bootstrap-react-app
$ cd my-bootstrap-react-app
$ npm start
```

The directory structure generated by Create React App is as below:

```javascript
.
├── package.json
├── public
│   ├── favicon.ico
│   ├── index.html
│   └── manifest.json
├── README.md
├── src
│   ├── App.css
│   ├── App.js
│   ├── App.test.js
│   ├── index.css
│   ├── index.js
│   ├── logo.svg
```

```
    │   └── registerServiceWorker.js
    └── yarn.lock
```

The next step is to take the latest Bootstrap library from its official website. The downloaded package consists of the compiled as well as minimized versions of JavaScript and CSS files. There is a grid-specific stylesheet as well for apps that just need to make use of grids. The next step is to make a new folder for CSS in public, copy bootstrap.min.css file there, and then add the required code in public/index.html to link it.

html

```html
<head>
<link rel="stylesheet" href="css/bootstrap.min.css">
</head>
```

Another way is to use a CDN to fetch the minimized CSS:

html

```html
<link rel="stylesheet" href= "https://maxcdn.bootstrapcdn.com/bootstrap/4.
```

How to Use Regular Bootstrap Classes With React

We can use Bootstrap classes with JSX code after the Bootstrap stylesheet has been added to a React app. In order to verify this, we'll copy some random sample code from the Bootstrap demo site:

javascript

```javascript
import React from "react";
import {render} from "react-dom";
import Forms from "./Forms";
import NavBar from "./NavBar";

const App = () => (
  <div>
    <NavBar />
    <br />
    <div className="container">
      <div className="row">
        <div className=" col-lg-offset-4 col-lg-8">
          <Forms />
        </div>
      </div>
    </div>
  </div>
);

render(<App />, document.getElementById("root"));
```

It would be really nice if the Bootstrap classes could be imported as React components to make the best use of React. For example, we could have grid, row and Ccolumn components to organize the page instead of the HTML classes as shown below:

html

```
<!--Bootstrap Using HTML styles/classes-->

<div class="container">
  <div class="row">
    <div class="col-sm">
      Col 1 of 3
    </div>
    <div class="col-sm">
      Col 2 of 3
    </div>
    <div class="col-sm">
      Col 3 of 3
    </div>
  </div>
</div>

<!--Bootstrap Using React-based Components-->

<Grid>
  <Row>
      <Col sm>
        Col 1 of 3
    </Col>
    <Col sm>
        Col 2 of 3
    </Col>
     <Col sm>
        Col 3 of 3
    </Col>
```

```
      </Row>
    </Grid>
```

Luckily, there is no need to implement our own library to make this possible, as there are libraries already available. Let's explore a few of these.

Using Third-party Libraries for React and Bootstrap

There are some libraries that attempt to develop an implementation of Bootstrap that is specific to React that will allow us to use JSX components as well as work with Bootstrap styles. Following are some of the popular Bootstrap modules that can be used with React projects.

1. React-Bootstrap is amongst the top libraries for adding Bootstrap components to React projects. But its current implementation is targeted for Bootstrap v3 and not the latest version.
2. reactstrap is another library which enables us to use Bootstrap components in a React app. In contrast to React-Bootstrap, reactstrap is developed to be used with the latest version of Bootstrap. The reactstrap implementation has components for forms, buttons, tables, layout grids, and navigation. It is currently

in development, but provides a nice alternative for developing apps with React and Bootstrap together.

There are some alternatives like React-UI and some domain-specific modules like CoreUI-React, React-bootstrap-table available on GitHub that deliver extensive utilities to develop some cool UI for apps using React.

In this guide, we'll focus on reactstrap since it is the most popular and uses the latest Bootstrap version.

How to Set Up Reactstrap Library

To get started, we install the reactstrap library using npm:

javascript

```
npm install --save reactstrap@next
```

Now, the relevant components from the module can be imported as below:

javascript

```
import { Container, Row, Col} from 'reactstrap';
```

At this stage, the library will not work as expected since it does not include Bootstrap CSS. We'll need to add it manually as shown below:

javascript

```
npm install --save bootstrap
```

Next step is to import Bootstrap CSS in our `src/index.js` file:

javascript

```
import 'bootstrap/dist/css/bootstrap.css';
```

Understanding Bootstrap Grid

Bootstrap is developed with a responsive, mobile first grid system that enables the use of 12 columns per page. We'll need to import the `Container`, `Row` and `Col` components in order to use the grid. The `Container` has a fluid attribute that alters a fixed-width layout to a full-width layout. It essentially adds the `.container-fluid` bootstrap class to the grid. The Col component can be configured to work with attributes like `xs`, `md`, `sm` and `lg` that work the same way as the `col-*` classes in Bootstrap, e.g., `<Col xs=”6”> </Col>`. Another way to implement this is by passing an object to the props with optional attributes like `size`, `order`, and `offset`. The `size` attribute denotes how many columns are there in the grid, while `order` enables us to arrange the columns and works with values from 1 to 12. The columns can be moved to the right by using the `offset` property. The following code elaborates a few of the features of `Grid` in reactstrap:

javascript

```javascript
import React, {Component} from "react";
import { render } from "react-dom";
import { Row, Col, Container} from "reactstrap";
import "bootstrap/dist/css/bootstrap.css";
import "./App.css";

class App extends Component {
  render() {
    return (
      <div>
        <Container>
          <Row>
            <Col>
              <Box>.col </Box>
            </Col>
          </Row>
          <Row>
            <Col xs="4">
              <Box> .col-4 </Box>
            </Col>

            <Col xs="4">
              <Box> .col-4 </Box>
            </Col>
```

```
</Row>

<Row>
  <Col xs="6">
    <Box> .col-6 </Box>
  </Col>

  <Col xs="6">
    <Box> .col-6 </Box>
  </Col>
  <Col xs="6">
    <Box>.col-6 </Box>
  </Col>
</Row>
<Row>
  <Col>
    <Box>.col </Box>
  </Col>
  <Col>
    <Box>.col </Box>
  </Col>
  <Col>
    <Box> .col </Box>
  </Col>
  <Col>
    <Box>.col </Box>
  </Col>
</Row>
<Row>
  <Col xs="3">
    <Box> .col-3 </Box>
  </Col>
  <Col xs="auto">
    <Box>.col-auto is used for dynamic content</Box>
  </Col>
```

```
            <Col xs="3">
              <Box> .col-3 </Box>
            </Col>
          </Row>
          <Row>
            <Col sm={{ size: 4, order: 4, offset: 2 }}>
              <Box> .col-sm-4 .col-sm-order-4 .col-sm-offset-4 </Box>{" '
            </Col>
          </Row>
          <Row>
            <Col sm="12" md={{ size: 6, offset: 3 }}>
              <Box> .col-sm-10 .col-md-4 .col-md-offset-5 </Box>
            </Col>
          </Row>
        </Container>
      </div>
    );
  }
}

export default App;
const Box = props => <div className="box">{props.children} </div>;
render(<App />, document.getElementById("root"));
```

Using Bootstrap Components With React

Now that we are familiar with reactstrap and how it works, there are numerous components of Bootstrap 4 that can be used with React using reactstrap. We'll look at few of the important components in this guide.

Navigation Bar

reactstrap `Navbars` can be used for navigation bars and provide responsiveness. To organize the navigation links more efficiently, a Navbar consists of subcomponents like `Nav`, `NavItem`, `NavbarBrand`, etc . A responsive Navbar can created by adding a `<NavbarToggler>` inside our `<Navbar>` component and then wrapping `<NavItems>` into a `<Collapse>` component. We'll look at the following code to see how the `Navbar` component and React state can be used to save the toggle data;:

javascript

```javascript
export default class NavBarExample extends React.Component {
  constructor(props) {
    super(props);
    this.toggleOpenState = this.toggleOpenState.bind(this);
    this.state = {
      isOpen: false
    };
  }
  toggleOpenState() {
    this.setState({
      isOpen: !this.state.isOpen
    });
  }
  render() {
    return (
      <div>
        <Navbar color="#ffffff" light expand="md">
               <NavbarBrand href="/">
                This is a Navbar Demo
                </NavbarBrand>
               {/* Below, we'll add toggler for auto-collapse */}
                <NavbarToggler onClick={this.toggleOpenState} />
                <Collapse isOpen={this.state.isOpen} navbar>

              {/*Pull towards left */}
            <Nav className="ml-auto" navbar>
                <NavItem>
                    <NavLink href="/link/">
                        Left Navigation Link
                    </NavLink>
                </NavItem>
            </Nav>

            {/* Pull towards right */}
            <Nav className="mr-auto" navbar>
              <UncontrolledDropdown nav inNavbar>
                <DropdownToggle nav caret>
                  Chris
                </DropdownToggle>
                <DropdownMenu >
                  <DropdownItem>
                    My Account
                  </DropdownItem>
                  <DropdownItem>
                    Page Settings
                  </DropdownItem>
                  <DropdownItem divider />
                  <DropdownItem>
                    Log Out
                  </DropdownItem>
                </DropdownMenu>
              </UncontrolledDropdown>
            </Nav>
          </Collapse>
        </Navbar>
      </div>
    );
  }
}
```

Modal Window

The reactstrap `Modal` component can be used for generating a Bootstrap modal with a header, body, and footer. A modal component can be used with some props and callbacks to have the window interactive and also to have it closable. To decide whether the modal should be visible or not, the `isOpen` property is used. The toggle callback is used to switch the value of `isOpen` in the component. There are few more props that are used to animate the transitions. The available callbacks include `onEnter`, `onExit`, `onOpened`, and `onClosed`:

html

```html
{/*To open the modal window, this.state.show needs to be true, which is
{/* toggleModal updates state of "show" to false onClose*/}

<Modal isOpen={this.state.show} toggle={this.toggleModal} >

    <ModalHeader toggle={this.toggle}>
```

```
        Modal title
    </ModalHeader>

    <ModalBody>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed ulla
    </ModalBody>
    <ModalFooter>
        <Button color="pri" onClick={this.toggle}>Click on Me!</Button>{'
        <Button color="sec" onClick={this.toggle}>Cancel</Button>
     </ModalFooter>
</Modal>
```

Forms

A reactstrap form can be inline or horizontal. The input element is rendered by an `Input` component. Multiple `Input` components can be wrapped into a `FormGroup` for state validation, appropriate spacing, and to use other `FormGroup` features. To set any label, we can do that using `<Label>`. There is a lot that can be done using forms. You can check out the React documentation on forms for more details. Below is the code for our form:

html

```
<Form>
   <FormGroup row>
      <Label for="exampleEmail" sm={2}>Email</Label>
      <Col sm={10}>
          <Input type="email" name="email" id="exampleEmail" placeholder=
      </Col>
   </FormGroup>
```

```

         <FormGroup row>
            <Label for="examplePassword" sm={2}>Password</Label>
            <Col sm={10}>
               <Input type="password" name="password" id="examplePassword" plac
            </Col>
         </FormGroup>

         <FormGroup row>
            <Label for="exampleSelect" sm={2}>Select</Label>
            <Col sm={10}>
                <Input type="select" name="select" id="exampleSelect" />
            </Col>
         </FormGroup>

         <FormGroup row>
            <Label for="exampleSelectMulti" sm={2}>Select Multiple</Label>
            <Col sm={10}>
              <Input type="select" name="selectMulti" id="exampleSelectMulti"
            </Col>
          </FormGroup>

        <FormGroup row>
          <Label for="exampleText" sm={2}>Text Area</Label>
          <Col sm={10}>
            <Input type="textarea" name="text" id="exampleText" />
          </Col>
        </FormGroup>
      </Form>
```

ListGroup

The styling and control of list items can be done easily using reactstrap `ListGroup`. The `ListGroupItems` are wrapped in `ListGroup`. The `onClick` callback can be used to make it interactive. Below is the code for `ListGroup`;

html

```html
<ListGroup>
  <ListGroupItem>Item 1</ListGroupItem>
  <ListGroupItem>Item 2</ListGroupItem>
  <ListGroupItem>...</ListGroupItem>
</ListGroup>;
```

Buttons

Buttons can be the most important component for any design framework. There is a reactstrap `Button` component for buttons. Other than the general active and disabled properties, we can use `color` and `size` to set the style (`primary`, `success`, etc.) and size (`lg`, `sm`, etc.) of buttons:

html

```html
{/*ButtonToolbar helps to organize buttons */}
 <div>
    <Button color="primary">Primary btn</Button>{' '}
    <Button color="secondary">Secondary btn</Button>{' '}
    <Button color="warning">Warning btn</Button>{' '}
    <Button color="danger">Danger btn</Button>{' '}
    <Button color="success">Success btn</Button>{' '}
    <Button color="info">info btn</Button>{' '}
```

```
   <Button color="link">link btn</Button>
</div>
```

Conclusion

Everything that we need to integrate Bootstrap with React app has been covered in this guide. There are numerous libraries to integrate Bootstrap with React app and we have discussed few of the best known ones. We have also worked with one of the most commonly used libraries, reactstrap.

98