State: Concepts

In This Lesson, We Will Cover Two Hooks

01 `useState`: Allows you to use state in a functional component.

02 `useEffect`: Replaces lifecycle methods like `componentDidMount` and `componentDidUpdate`.

03 **Custom Hooks:** Create your own reusable Hooks!

Effect is a term used to describe the result of affecting the "outside world." This includes data fetching, subscribing to events, and making changes to the DOM.

Actions, Reducers, and Store

Today, we will cover three core Redux topics

01 **Action**
A declarative object that defines what the store should do to the state.

02 **Reducer**
Handles all actions by taking in the previous state and returning a new state.

03 **Store**
Grants the entire application access to the reducer function and the global state.

The Two Rules of Hooks

01 **Do not** call Hooks from within loops, conditionals, or nested functions.

- Hooks must always be called in the same order, like component lifecycle methods.
- This makes it possible for React to store the state of Hooks when using `useState` or `useEffect`.

02 **Do not** call Hooks from within regular JavaScript functions.

- This makes it so that all stateful logic is easy to find for the developer (you).

Comparing Ways to Manage State

01 Class Components with `setState()`

Advantages
- Component and children will re-render with up-to-date data.

Disadvantages
- Updating state from nested components can be difficult.
- Since state only flows one way, all components that need access to the state must be children of the same stateful component.

02 Functional Components with `useState()`

Advantages
- Easier to read and debug, and no need to use `this`
- Access to Hooks

Disadvantages
- Needs to use other Hooks to manage complex levels of state.
- Not supported by older codebases, which will still need to use class components for state.

Container vs. Presentational Components

Separate the logic from the looks

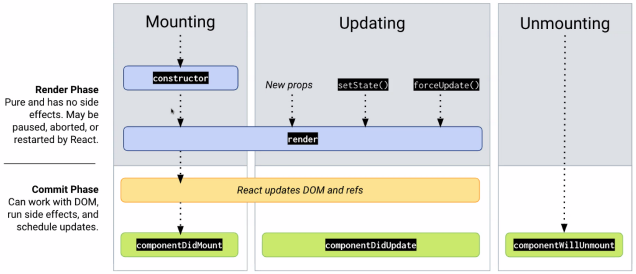
Container components include the logic of the application, which is often stored in the component's state, requiring us to use class components instead of functional components.

The Gallery Page component from the previous activity is a container component.

Presentational components are primarily used for UI elements like layout, and often use stateless functional components (often referred to as "dumb components") to render their content. All of the components in the `components` folder from the previous activity are presentational components.

Not **all** presentational components have to be stateless. Some may contain UI elements in their state.

Component Lifecycle



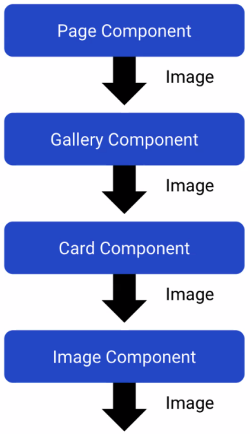
Prop Drilling

Prop drilling is the process you go through to get data to parts of the React component tree.

Although it seems tedious, prop drilling is often necessary to avoid complicating the global state of your application

Remember, it's often best to keep our state as close as possible to where it's relevant.

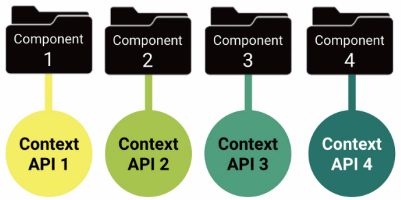
Why can't we just add state to the lowest level component?



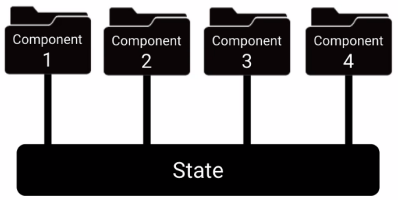
Tightly Coupled vs. Loosely Coupled

Using the Context API can be very helpful, but it comes at a cost. Adding stateful logic to a component creates a dependent relationship to another component. This is known as coupling.

When writing applications, it is best to keep components loosely coupled so that you can change one without affecting the other. This also helps ensure reusability.



Increasing complexity over time reduces scalability, reliability, portability and security.



Platform approach provides standardization, consolidation, normalization and governance.

```jsx
import React, { useState } from "react";
import Container from "../../components/Container";
import Col from "../../components/Col";
import Row from "../../components/Row";

function Signup() {
  const [username, setUsername] = useState();
  const [password, setPassword] = useState();
  const handleSubmit = e => {
    e.preventDefault();
    console.log("username is " + username);
    console.log("password is " + password);
  };
  return (
    <div>
      <div className="mt-4">
        <h2>Sign Up</h2>
      </div>
      <form onSubmit={handleSubmit}>
        <Container className="mt-3 px-5">
          <Row className="form-group">
            <Col size="12">
              <input
                className="form-control"
                type="text"
                placeholder="Username"
                name="username"
                onChange={e => setUsername(e.target.value)}
              />
            </Col>
          </Row>
          <Row className="form-group">
            <Col size="12">
              <input
                className="form-control"
                type="password"
                placeholder="Password"
                name="password"
                onChange={e => setPassword(e.target.value)}
              />
            </Col>
          </Row>
          <button className="btn btn-success" type="submit">
            Submit
          </button>
        </Container>
        <Container className="mt-4">
          <h3>Hello {username}!</h3>
          <p>I probably shouldn't tell you this, but your password is {password}!</p>
        </Container>
      </form>
    </div>
  );
}
export default Signup;
```

useState

```jsx
import React from "react";
import { Link } from "react-router-dom";
import "./style.css";

// Depending on the current path, this component sets the "active" class
//      on the appropriate navigation link item
function Navbar() {
  return (
    <nav className="navbar navbar-expand-lg navbar-light bg-light">
      <Link className="navbar-brand" to="/">
        GitHub JobWatcher
      </Link>
      <div>
        <ul className="navbar-nav">
          <li className="nav-item">
            <Link
              to="/"
              className={
                window.location.pathname === "/" || window.location.pathname === "/home"
                  ? "nav-link active"
                  : "nav-link"
              }
            >
              Home
            </Link>
          </li>
        </ul>
      </div>
    </nav>
  );
}
export default Navbar;
```

```jsx
import React from "react";
import CardBtn from "../CardBtn";
import CardContext from "../../utils/CardContext";
import CardTitle from "../CardTitle";
import "./style.css";

function Card() {
  return (
    // The most straightforward solution would be to add the Consumer to the Card component.
    // This way, all Card components can have the Card context passed directly as props
    <CardContext.Consumer>
      {({ image, handleBtnClick }) => (
        <div
          className="card"
          style={{
            backgroundImage: image ? `url(${image})` : "none"
          }}
        >
          {/* Here, we do not pass the title to demonstrate that it can also be consumed from the CardTitleText component */}
          <CardTitle />
          {!image && <i className="fa fa-spinner fa-spin" aria-hidden="true" />}
          <CardBtn style={{ opacity: image ? 1 : 0 }} onClick={handleBtnClick} data-value="back" />
          <CardBtn style={{ opacity: image ? 1 : 0 }} onClick={handleBtnClick} data-value="next" />
        </div>
      )}
    </CardContext.Consumer>
  );
}
export default Card;
```

```
import React, { useState, useEffect } from "react";
import API from "../../utils/API";
import Container from "../../components/Container";
import SearchForm from "../../components/SearchForm";
import SearchResults from "../../components/SearchResults";
import Alert from "../../components/Alert";

function Search() {
  const [search, setSearch] = useState("Wikipedia");
  const [title, setTitle] = useState("");
  const [url, setUrl] = useState("");
  const [error, setError] = useState("");

  useEffect(() => {
    if (!search) {
      return;
    }
    API.searchTerms(search)
      .then(res => {
        if (res.data.length === 0) {
          throw new Error("No results found.");
        }
        if (res.data.status === "error") {
          throw new Error(res.data.message);
        }
        setTitle(res.data[1][0]);
        setUrl(res.data[3][0]);
      })
      .catch(err => setError(err));
  }, [search]);

  const handleInputChange = event => {
    setSearch(event.target.value);
  };

  return (
    <div>
      <Container style={{ minHeight: "100vh" }}>
        <h1 className="text-center">Search For Anything on Wikipedia</h1>
        <Alert type="danger" style={{ opacity: error ? 1 : 0, marginBottom: 10 }}>
          {error}
        </Alert>
        <SearchForm
          handleInputChange={handleInputChange}
          results={search}
        />
        <SearchResults title={title} url={url} />
      </Container>
    </div>
  );
}
export default Search;
```

useEffect

```
import React from "react";
import "./style.css";

// Using the datalist element we can create autofill suggestions
          based on the props.breeds array
function SearchForm(props) {
  return (
    <form className="search">
      <div className="form-group">
        <label htmlFor="language">Search Term:</label>
        <input
          value={props.search}
          onChange={props.handleInputChange}
          name="term"
          list="term"
          type="text"
          className="form-control"
          placeholder="Type in a search term to begin"
          id="term"
        />
      </div>
    </form>
  );
}
export default SearchForm;
```

```
import axios from "axios";

// Export an object containing methods we'll use for accessing the GitHub Jobs API
export default {
  searchTerms: function(query) {
    return axios.get(
      "https://en.wikipedia.org/w/api.php?action=opensearch&search=" +
        query +
        "&limit=1&format=json&origin=*"
    );
  }
}
```

```
import React from "react";
import "./style.css";

function SearchResults(props) {
  return (
    <ul className="list-group search-results">
      <li className="list-group-item">
        <h2>{props.title}</h2>
        <a href={props.url}>{props.url}</a>
      </li>
    </ul>
  );
}
export default SearchResults;
```

Custom Hooks

```
import React, { useState, useEffect } from "react";
import API from "../../utils/API";
import Container from "../../components/Container";
import SearchForm from "../../components/SearchForm";
import SearchResults from "../../components/SearchResults";
import Alert from "../../components/Alert";
import useDebounce from "../../utils/debounceHook";

function Search() {
  const [search, setSearch] = useState("Wikipedia");
  const [title, setTitle] = useState("");
  const [url, setUrl] = useState("");
  const [error, setError] = useState("");
  const debouncedSearchTerm = useDebounce(search, 500);

  useEffect(() => {
    if (!search) {
      return;
    }
    if (debouncedSearchTerm) {
      API.searchTerms(search)
        .then(res => {
          if (res.data.length === 0) {
            throw new Error("No results found.");
          }
          if (res.data.status === "error") {
            throw new Error(res.data.message);
          }
          setTitle(res.data[1][0]);
          setUrl(res.data[3][0]);
        })
        .catch(err => setError(err));
    }
  }, [debouncedSearchTerm]);

  const handleInputChange = event => {
    setSearch(event.target.value);
  };

  return (
    <div>
      <Container style={{ minHeight: "100vh" }}>
        <h1 className="text-center">Search For Anything on Wikipedia</h1>
        <Alert type="danger" style={{ opacity: error ? 1 : 0, marginBottom: 10 }}>
          {error}
        </Alert>
        <SearchForm
          handleInputChange={handleInputChange}
          results={search}
        />
        <SearchResults title={title} url={url} />
      </Container>
    </div>
  );
}
export default Search;
```

```
import React from "react";
import { BrowserRouter as Router, Route } from "react-router-dom";
import Signup from "./pages/Signup";
import Search from "./pages/Search";
import Navbar from "./components/Navbar";
import Footer from "./components/Footer";
import Wrapper from "./components/Wrapper";
import "./App.css";

function App() {
  return (
    <Router>
      <div>
        <Navbar />
        <Wrapper>
          <Route exact path="/" component={Search} />
          <Route exact path="/signup" component={Signup} />
          <Route exact path="/search" component={Search} />
        </Wrapper>
        <Footer />
      </div>
    </Router>
  );
}
export default App;
```

```
import { useEffect, useState } from "react";

const useDebounce = (value, delay) => {
  const [debouncedValue, setDebouncedValue] = useState(value);

  useEffect(
    () => {
      const handler = setTimeout(() => {
        setDebouncedValue(value);
      }, delay);
      // Cancel the timeout if value or delay changes
      return () => {
        clearTimeout(handler);
      };
    },
    // Only call the effect if value or delay changes.
    [value, delay]
  );
  return debouncedValue;
};
export default useDebounce;
```

3rd Party Hooks

```
import React from "react";
import {
  useInput,
  useBoolean,
  useNumber,
} from "react-hanger";

function Survey() {
  const favoriteThing = useInput("");
  const showComment = useBoolean(false);
  const comment = useInput("");
  const feeling = useInput("");
  const rating = useNumber(0)

  const handleSubmit = () => {
    const form = {
      favoriteThing: favoriteThing.value,
      comment: comment.value,
      feeling: feeling.value,
      rating: rating.value
    }
    console.log(form)
  }
```

```
import React from "react";
import Survey from "./pages/Survey";
import "./App.css";

function App() {
  return (
      <div>
          <Survey />
      </div>
  );
}
export default App;
```

```
  return (
    <div className="container">
      <h1>Use this form to provide feedback for our product!</h1>

      <h4>What was your favorite thing about our product?</h4>
      <textarea {...favoriteThing.eventBind} />
      <h4>How would you rate our product?</h4>
      <div className="form-group" >
        <input type="radio" name="rating-1" onChange={() => rating.setValue(1)} />1
        <input type="radio" name="rating-1" onChange={() => rating.setValue(2)} />2
        <input type="radio" name="rating-1" onChange={() => rating.setValue(3)} />3
        <input type="radio" name="rating-1" onChange={() => rating.setValue(4)} />4
        <input type="radio" name="rating-1" onChange={() => rating.setValue(5)} />5
      </div>

      <h4>How did our product make you feel?</h4>
      <div className="form-group emoji" >
        <span role="img" aria-label="angry"
          onClick={() => {showComment.toggle(); feeling.setValue("angry")} }>□
        </span>
        <span role="img" aria-label="indifferent"
          onClick={() => {showComment.toggle(); feeling.setValue("indifferent")}}>□
        </span>
        <span role="img" aria-label="happy"
          onClick={() => {showComment.toggle(); feeling.setValue("happy")}}>□
        </span>
        <div className="response">
        {showComment.value ? (
          <textarea {...comment.eventBind} placeholder="Please add any additional comments" />
        ): null}
        </div>
        <div>
      {showComment.value ? (
          <span>You've responded that you feel {feeling.value}</span>
        ): null}
      </div>
      </div>
      <button onClick={handleSubmit}>Submit</button>
    </div>
  );
};
export default Survey;
```

Prop Drilling

```
import React from "react";
import Card from "../Card";
import "./style.css";

function CardContainer({ title, image, profileUrl, handleBtnClick }) {
  return (
    <div className="jumbotron card-container">
      <Card
        title={title}
        image={image}
        profileUrl={profileUrl}
        handleBtnClick={handleBtnClick}
      />
    </div>
  );
}
export default CardContainer;
```

```
import React from "react";
import CardBody from "../CardBody";
import CardBtn from "../CardBtn";
import CardImg from "../CardImage";
import CardHeading from "../CardHeading";
import "./style.css";

function Card({ title, image, profileUrl, handleBtnClick }) {
  return (
    <div>
      <CardHeading title={title} />
      <CardImg image={image} />
      <CardBody profileUrl={profileUrl} />
      {!image && <i className="fa fa-spinner fa-spin" aria-hidden="true" />}
      <CardBtn
        style={{ opacity: image ? 1 : 0 }}
        onClick={handleBtnClick}
        data-value="back"
      />
      <CardBtn
        style={{ opacity: image ? 1 : 0 }}
        onClick={handleBtnClick}
        data-value="next"
      />
    </div>
  );
}
export default Card;
```

```
import React, { useEffect, useState } from "react";
import API from "../utils/API";
import CardContainer from "../components/CardContainer";
import Row from "../components/Row";

function Gallery() {
  const [user, setUser] = useState({});
  const [users, setUsers] = useState([]);
  const [userIndex, setUserIndex] = useState(0);
  // When the component mounts, a call will be made to get random users.
  useEffect(() => {
    loadUsers();
  }, []);

  function nextUser(userIndex) {
    // Ensure that the user index stays within our range of users
    if (userIndex >= users.length) {
      userIndex = 0;
    }
    setUser(users[userIndex]);
    setUserIndex(userIndex);
  }

  function previousUser(userIndex) {
    // Ensure that the user index stays within our range of users
    if (userIndex < 0) {
      userIndex = users.length - 1;
    }
    setUser(users[userIndex]);
    setUserIndex(userIndex);
  }
```

```
  function handleBtnClick(event) {
    // Get the title of the clicked button
    const btnName = event.target.getAttribute("data-value");
    if (btnName === "next") {
      const newUserIndex = userIndex + 1;
      nextUser(newUserIndex);
    } else {
      const newUserIndex = userIndex - 1;
      previousUser(newUserIndex);
    }
  }

  function loadUsers() {
    API.fetchUsers()
      .then(users => {
        setUsers(users);
        setUser(users[0]);
      })
      .catch(err => console.log(err));
  }
  return (
    <div>
      <h1 className="text-center">Welcome to LinkedUp</h1>
      <p className="text-center h3">Click on the arrows to browse users</p>
      <Row>
        <CardContainer
          title={user.login}
          image={user.image}
          profileUrl={user.profileUrl}
          handleBtnClick={handleBtnClick}
        />
      </Row>
    </div>
  );
}
export default Gallery;
```

```
import axios from "axios";

// Export an object containing methods
//          we'll use for accessing the random user API
export default {
  fetchUsers: function() {
    return axios
      .get("https://api.github.com/orgs/github/public_members")
      .then(res => {
        const users = res.data;
        return users.map(user => {
          return {
            login: user.login,
            image: user.avatar_url,
            profileUrl: user.html_url
          };
        });
      });
  }
};
```

useContext

```
import React, { useState, useEffect } from "react";
import Container from "../../components/Container";
import SearchForm from "../../components/SearchForm";
import SearchResults from "../../components/SearchResults";
import Alert from "../../components/Alert";
import ArticleContext from "../../utils/ArticleContext";
import API from "../../utils/API";

function Search() {
  const [articleState, setArticleState] = useState({
    title: "",
    url: ""
  });

  const [search, setSearch] = useState("Wikipedia");
  const [error, setError] = useState("");

  // When the component mounts, update the title to be Wikipedia Searcher
  useEffect(() => {
    document.title = "Wikipedia Searcher";
    if (!search) {
      return;
    }
    API.searchTerms(search)
      .then(res => {
        if (res.data.length === 0) {
          throw new Error("No results found.");
        }
        if (res.data.status === "error") {
          throw new Error(res.data.message);
        }
        setArticleState({
          title: res.data[1][0],
          url: res.data[3][0]
        });
      })
      .catch(err => setError(err));
  }, [search]);

  const handleInputChange = event => {
    setSearch(event.target.value);
  };

  const handleFormSubmit = event => {
    event.preventDefault();
  };

  return (
    <ArticleContext.Provider value={articleState}>
      <div>
        <Container style={{ minHeight: "100vh" }}>
          <h1 className="text-center">Search For Anything on Wikipedia</h1>
          <Alert type="danger" style={{ opacity: error ? 1 : 0, marginBottom: 10 }}>
            {error}
          </Alert>
          <SearchForm
            handleFormSubmit={handleFormSubmit}
            handleInputChange={handleInputChange}
            results={search}
          />
          <SearchResults />
        </Container>
      </div>
    </ArticleContext.Provider>
  );
}
export default Search;
```

```
import React, { useContext } from "react";
import ArticleContext from "../../utils/ArticleContext";
import "./style.css";

function SearchResults() {
  const {title, url} = useContext(ArticleContext);
  return (
    <ul className="list-group search-results">
      <li className="list-group-item">
        <h2>{title}</h2>
        <a href={url}>{url}</a>
      </li>
    </ul>
  );
}
export default SearchResults;
```

```
import React from "react";

const ArticleContext = React.createContext({
  title: "",
  url: ""
});
export default ArticleContext;
```

```
import React, { useEffect, useState } from "react";
import API from "../utils/API";
import UserContext from "../utils/userContext";
import CardContainer from "../components/CardContainer";
import Row from "../components/Row";

function Gallery() {
  const [users, setUsers] = useState([]);
  const [user, setUser] = useState({});
  const [userIndex, setUserIndex] = useState(0);

  // When the component mounts, a call will be made to get random users.
  useEffect(() => {
    loadUsers();
  }, []);

  function loadUsers() {
    API.getLanguagesList()
      .then(languages => {
        API.getUsersByLanguage(languages[0]).then(users => {
          setUsers(users);
          setUser(users[0]);
        });
      })
      .catch(err => console.log(err));
  }

  function nextUser(userIndex) {
    // Ensure that the user index stays within our range of users
    if (userIndex >= users.length) {
      userIndex = 0;
    }
    setUserIndex(userIndex);
    setUser(users[userIndex]);
  }

  function previousUser(userIndex) {
    // Ensure that the user index stays within our range of users
    if (userIndex < 0) {
      userIndex = users.length - 1;
    }
    setUserIndex(userIndex);
    setUser(users[userIndex]);
  }

  function handleBtnClick(event) {
    // Get the title of the clicked button
    const btnName = event.target.getAttribute("data-value");
    if (btnName === "next") {
      const newUserIndex = userIndex + 1;
      nextUser(newUserIndex);
    } else {
      const newUserIndex = userIndex - 1;
      previousUser(newUserIndex);
    }
  }

  return (
    <UserContext.Provider value={{ user, users, handleBtnClick }}>
      <div>
        <h1 className="text-center">Welcome to LinkedUp</h1>
        <h3 className="text-center">Click on the arrows to browse users</h3>
        <Row>
          <CardContainer />
        </Row>
      </div>
    </UserContext.Provider>
  );
}
export default Gallery;
```

Dynamic Context

```
import React, { useState } from "react";
import Home from "./pages/Home";
import AlertContext from "./utils/AlertContext";

function App() {
  const [pageState, setPageState] = useState({
    display: false,
    theme: "success",
    onClick: (theme, display) => {
      // Remember, the setter method on state does not merge like this.setState does
      // We use the spread operator so that we don't lose our onClick method
            whenever the state is updated.
      setPageState({ ...pageState, theme, display });
    }
  });

  // App component that provides initial context values
  // Here we are overwritting the context object to be equal to the state of App
  return (
    <AlertContext.Provider value={pageState}>
      <Home />
    </AlertContext.Provider>
  );
}

export default App;
```

```
import React from "react";

// default context object with properties corresponding to Provider values
const AlertContext = React.createContext({
  display: false,
  msg: "default value",
  theme: "",
  onClick: () => undefined
});

export default AlertContext;
```

Multiple Contexts

```
import React, { useState } from "react";
import Home from "./pages/Home";
import ThemeContext from "./components/ThemeContext";
import UserContext from "./components/UserContext";
import AlertContext from "./components/AlertContext";

function App() {
  const [user, setUser] = useState({
    name: "Bob",
    getUserToken: getUserToken
  })

  const [alert, setAlert] = useState({
    display: false,
    theme: "success",
    onClick: (theme, display) => setAlert({...alert, theme, display})
  })

  function getUserToken() {
    return "SampleToken123";
  }
  // Here we are overwriting each Context Object with values
            from the state of App.js and a string literal.
    return (
      <AlertContext.Provider value={alert}>
        <UserContext.Provider value={user}>
          <ThemeContext.Provider value={"dark"}>
            <Home />
          </ThemeContext.Provider>
        </UserContext.Provider>
      </AlertContext.Provider>
    );
  }
export default App;
```

```
import React, { useContext } from "react";
import Content from "../components/Content";
import Nav from "../components/Nav";
import ThemeContext from "../components/ThemeContext";

function Home() {
  const theme = useContext(ThemeContext);

  return (
    <div>
      <Nav theme={theme} />
      <div style={{ textAlign: "center" }}>
        <h1>Multiple Context Providers!</h1>
      </div>
      <div style={{ margin: "0 auto" }}>
        <Content />
      </div>
    </div>
  );
}
export default Home;
```

```
import React from "react";
// Set up a default object with the same property types
            as the value that we pass through the Provider.

const AlertContext = React.createContext({
  display: false,
  msg: "default value",
  theme: "",
  onClick: () => undefined
});
export default AlertContext;
```

```
import React from "react";

// Set the user of the page to default to ""
// The getUserToken method defaults to an empty method
const UserContext = React.createContext({
  name: "",
  getUserToken: () => {}
});
export default UserContext;
```

```
import React from "react";

// Set the theme of the page to default to "light"
const ThemeContext = React.createContext("light");

export default ThemeContext;
```

```
import React from "react";
import MovieContext from "../utils/movieContext";
function MovieDetail() {
  return (
    // Use consumer to capture and destucture the state values
    <MovieContext.Consumer>
      {({result: {Title, Poster, Director, Genre, Released}}) => (
        <div className="text-center">
          <img alt={Title} className="img-fluid" src={Poster} style={{ margin:
"0 auto" }} />
          <h3>Director(s): {Director}</h3>
          <h3>Genre: {Genre}</h3>
          <h3>Released: {Released}</h3>
        </div>
      )}
    </MovieContext.Consumer>
  );
}
export default MovieDetail;
```

Class Context

```
import React from "react";

const MovieContext = React.createContext({
  result: {},
  search: "",
  handleInputChange: () => {},
                     handleFormSubmit: () => {}
                    });
                    export default MovieContext;
```

```
import React from "react";
import MovieContext from "../utils/movieContext";

function SearchForm() {
  return (
    // Use consumer to capture state.search and form handler functions
    <MovieContext.Consumer>
      {({search, handleInputChange, handleFormSubmit}) => (
        <form>
          <div className="form-group">
            <label htmlFor="search">Search:</label>
            <input
              onChange={handleInputChange}
              value={search}
              name="search"
              type="text"
              className="form-control"
              placeholder="Search For a Movie"
              id="search"
            />
            <br />
            <button onClick={handleFormSubmit} className="btn btn-primary">
              Search
            </button>
          </div>
        </form>
      )}
    </MovieContext.Consumer>
  );
}
export default SearchForm;
```

```
import React, { Component } from "react";
import Container from "./Container";
import Row from "./Row";
import Col from "./Col";
import Card from "./Card";
import SearchForm from "./SearchForm";
import MovieDetail from "./MovieDetail";
import API from "../utils/API";
import MovieContext from "../utils/movieContext";

class OmdbContainer extends Component {
  state = {
    result: {},
    search: ""
  };

  // When this component mounts, search for the movie "The Matrix"
  componentDidMount() {
    this.searchMovies("The Matrix");
  }

  searchMovies = query => {
    API.search(query)
      .then(res => this.setState({ result: res.data }))
      .catch(err => console.log(err));
  };

  handleInputChange = event => {
    const value = event.target.value;
    const name = event.target.name;
    this.setState({
      [name]: value
    });
  };

  // When the form is submitted, search the OMDB API for the value of `this.state.search`
  handleFormSubmit = event => {
    event.preventDefault();
    this.searchMovies(this.state.search);
  };

  render() {
    return (
      // Pass state and the two form handler functions into the provider
      <MovieContext.Provider
        value={{
          ...this.state,
          handleInputChange: this.handleInputChange,
          handleFormSubmit: this.handleFormSubmit
        }}
      >
        <Container>
          <Row>
            <Col size="md-8">
              <Card
                heading={this.state.result.Title || "Search for a Movie to Begin"}
              >
                {this.state.result.Title ? <MovieDetail /> : <h3>No Results to Display</h3>}
              </Card>
            </Col>
            <Col size="md-4">
              <Card heading="Search">
                <SearchForm />
              </Card>
            </Col>
          </Row>
        </Container>
      </MovieContext.Provider>
    );
  }
}
export default OmdbContainer;
```

```
import React, { useReducer } from "react";
import "./App.css";

function Count() {
  const dogs = [
    {
      name: "Harry",
      image: "https://images.dog.ceo/breeds/vizsla/n02100583_10960.jpg"
    },
    {
      name: "Hermione",
      image: "https://images.dog.ceo/breeds/husky/n02110185_1511.jpg"
    }
  ];

  const [state, dispatch] = useReducer(
    (state, action) => {
      if (action === "praiseHarry") {
        return { ...state, HarryPraises: state.HarryPraises + 1 };
      } else if (action === "praiseHermione") {
        return { ...state, HermionePraises: state.HermionePraises + 1 };
      } else {
        return state;
      }
    },
    { HarryPraises: 0, HermionePraises: 0 }
  );

  return (
    <div className="App">
      <h1>Praise your favorite dog!</h1>
      <div className="row mt-5">
        {dogs.map(item => (
          <div key={item.name} className="card mx-auto col-4">
            <img className="card-img-top" src={item.image} alt={item.name} />
            <div className="card-body">
              <h4 className="card-title">{item.name}</h4>
              <p className="card-text">
                {item.name} has been praised {state[item.name + "Praises"]} times!
              </p>
              <button className="btn btn-primary" onClick={()
                        => dispatch("praise" + item.name)}>
                Praise
              </button>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
}
export default Count;
```

useReducer

```
import React, { useReducer } from "react";
import "./App.css";

function App() {
  const [count, dispatch] = useReducer((state, action) => {
    if (action === "add") {
      return state + 1;
    } else if (action === "subtract") {
      return state - 1;
    } else {
      return state;
    }
  }, 0);

  return (
    <div className="App">
      <button className="btn btn-success mt-5 mb-5" onClick={() => dispatch("add")}>
        Add
      </button>
      <div>{count}</div>
      <button className="btn btn-danger mt-5" onClick={() => dispatch("subtract")}>
        Subtract
      </button>
    </div>
  );
}
export default App;
```

```
import React, { useReducer, useRef } from "react";
import "./App.css";

function Count() {
  const inputRef = useRef();

  const [count, dispatch] = useReducer((state, action) => {
    switch (action) {
    case "add":
      return state + 1;
    case "subtract":
      return state - 1;
    case "change":
      // convert the value from the input into an integer
      const newCount = parseInt(inputRef.current.value);
      // only update the count if the value is numeric
      if (isNaN(newCount)) {
        return state
      }
      return newCount;
    default:
      return state;
    }
  }, 0);

  return (
    <div className="App">
      <button className="btn btn-success mt-5 mb-5" onClick={() => dispatch("add")}>
        Add
      </button>
      <div>{count}</div>
      <button className="btn btn-danger mt-5" onClick={() => dispatch("subtract")}>
        Subtract
      </button>
      <input
        className="form-control w-25 mx-auto mt-5"
        placeholder="Type new value..."
        ref={inputRef}
      />
      <button className="btn btn-warning mt-5" onClick={() => dispatch("change")}>
        Change
      </button>
    </div>
  );
}
export default Count;
```

useRef

```
import React, { useReducer, useRef } from "react";
import "./App.css";

function TodoList() {
  const inputRef = useRef();

  const [items, dispatch] = useReducer((state, action) => {
    switch (action.type) {
    case "add":
      return [
        ...state,
        {
          id: state.length * Math.random(),
          name: action.name
        }
      ];
      // Bonus: Remove a todo from the list.
    case "remove":
      return state.filter((_, index) => {
        return index !== action.index;
      });
    default:
      return state;
    }
  }, []);

  const handleSubmit = e => {
    e.preventDefault();
    dispatch({
      type: "add",
      name: inputRef.current.value
    });
    inputRef.current.value = "";
  };

  return (
    <div className="container text-center">
      <h1>Create a Todo List!</h1>
      <form className="form-group mt-5" onSubmit={handleSubmit}>
        <input
          className="form-control"
          ref={inputRef}
          placeholder="Start typing what you need to do..."
        />
        <button className="btn btn-success mt-3 mb-5" type="submit">
          Add to List
        </button>
      </form>
      <h4>My Todo List:</h4>
      <ul className="list-group">
        {items.map((item, index) => (
          <li className="list-group-item" key={item.id}>
            {item.name}
            <button
              className="btn btn-danger ml-5"
              onClick={() => dispatch({ type: "remove", index })}
            >
              X Remove
            </button>
          </li>
        ))}
      </ul>
    </div>
  );
}
export default TodoList;
```

```
import React, { createContext, useReducer, useContext } from "react";

const TodoContext = createContext({
  id: "",
  name: "",
  priority: false
});

const { Provider } = TodoContext;

function reducer(state, action) {
  switch (action.type) {
  case "add":
    return [
      ...state,
      {
        id: state.length * Math.random(),
        name: action.name
      }
    ];
  case "remove":
    return state.filter((_, index) => {
      return index !== action.index;
    });
  case "prioritize":
    return state.map((item, index) => {
      if (index === action.index) {
        return Object.assign({}, item, {
          priority: !item.priority
        });
      }
      return item;
    });
  default:
    return state;
  }
}

function TodoProvider({ value = [], ...props }) {
  const [state, dispatch] = useReducer(reducer, []);
  return <Provider value={[state, dispatch]} {...props} />;
}

function useTodoContext() {
  return useContext(TodoContext);
}
export { TodoProvider, useTodoContext };
```

Store

```
import React from "react";
import Form from "./components/Form";
import TodoList from "./components/TodoList";
import { TodoProvider } from "./utils/GlobalState";
import "./App.css";

function App() {
  return (
    <div className="container">
      <TodoProvider>
        <Form />
        <TodoList />
      </TodoProvider>
    </div>
  );
}
export default App;
```

```
import React, { useRef } from "react";
import { useTodoContext } from "../utils/GlobalState";

function Form() {
  const inputRef = useRef();
  const [_, dispatch] = useTodoContext();

  function handleSubmit(e) {
    e.preventDefault();
    dispatch({
      type: "add",
      name: inputRef.current.value
    });
    inputRef.current.value = "";
  }
  return (
    <div>
      <h1>Create a Todo List!</h1>
      <form className="form-group mt-5" onSubmit={handleSubmit}>
        <input
          className="form-control"
          ref={inputRef}
          placeholder="Start typing what you need to do..."
        />
        <button className="btn btn-success mt-3 mb-5" type="submit">
          Add to List
        </button>
      </form>
    </div>
  );
}
export default Form;
```

```
import React from "react";
import { useTodoContext } from "../utils/GlobalState";

function TodoList() {
  const [state, dispatch] = useTodoContext();
  return (
    <div>
      <h4>My Todo List:</h4>
      <ul className="list-group">
        {state.map((item, index) => (
          <li className="list-group-item col-12" key={item.id}>
            <button
              className="btn btn-warning mr-4"
              onClick={() => dispatch({ type: "prioritize", index })}
            >
              Prioritize
            </button>
            <button
              className="btn btn-danger mr-4"
              onClick={() => dispatch({ type: "remove", index })}
            >
              X Remove
            </button>
            {index}:<span className={item.priority ?
              "font-weight-bold" : ""}> {item.name}</span>
          </li>
        ))}
      </ul>
    </div>
  );
}
export default TodoList;
```

Mini-Project (partial) pt1

```
const db = require("../models");

// Defining methods for the postsController
module.exports = {
  findAll: function(req, res) {
    db.Post.find(req.query)
      .sort({ date: -1 })
      .then(dbModel => res.json(dbModel))
      .catch(err => res.status(422).json(err));
  },
  findById: function(req, res) {
    db.Post.findById(req.params.id)
      .then(dbModel => res.json(dbModel))
      .catch(err => res.status(422).json(err));
  },
  create: function(req, res) {
    db.Post.create(req.body)
      .then(dbModel => res.json(dbModel))
      .catch(err => res.status(422).json(err));
  },
  update: function(req, res) {
    db.Post.findOneAndUpdate({ _id: req.params.id }, req.body)
      .then(dbModel => res.json(dbModel))
      .catch(err => res.status(422).json(err));
  },
  remove: function(req, res) {
    db.Post.findById({ _id: req.params.id })
      .then(dbModel => dbModel.remove())
      .then(dbModel => res.json(dbModel))
      .catch(err => res.status(422).json(err));
  }
};
```

```
const path = require("path");
const router = require("express").Router();
const apiRoutes = require("./api");

// API Routes
router.use("/api", apiRoutes);

// If no API routes are hit, send the React app
router.use(function(req, res) {
  res.sendFile(path.join(__dirname, "../client/build/index.html"));
});

module.exports = router;
```

```
const router = require("express").Router();

const postRoutes = require("./posts");

// Post routes
router.use("/posts", postRoutes);
module.exports = router;
```

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;

const postSchema = new Schema({
  title: { type: String, required: true },
  author: { type: String, required: true },
  body: String,
  date: { type: Date, default: Date.now }
});

const Post = mongoose.model("Post", postSchema);
module.exports = Post;
```

```
const router = require("express").Router();
const postsController = require("../../controllers/postsController");

// Matches with "/api/posts"
router
  .route("/")
  .get(postsController.findAll)
  .post(postsController.create);

// Matches with "/api/posts/:id"
router
  .route("/:id")
  .get(postsController.findById)
  .put(postsController.update)
  .delete(postsController.remove);

module.exports = router;
```

```
const express = require("express");
const mongoose = require("mongoose");
const routes = require("./routes");
const app = express();

const PORT = process.env.PORT || 3001;

// Define middleware here
app.use(express.urlencoded({ extended: true }));
app.use(express.json());

// Serve up static assets (usually on heroku)
if (process.env.NODE_ENV === "production") {
  app.use(express.static("client/build"));
}

// Add routes, both API and view
app.use(routes);

// Connect to the Mongo DB
mongoose.connect(process.env.MONGODB_URI ||
        "mongodb://localhost/reactcms");

// Start the API server
app.listen(PORT, function() {
  console.log(`□  ==>
        API Server now listening on PORT ${PORT}!`);
});
```

Mini-Project (partial) pt2

```
import React from "react";
import { BrowserRouter as Router, Route, Switch } from "react-router-dom";
import Home from "./pages/Home";
import Detail from "./pages/Detail";
import NoMatch from "./pages/NoMatch";
import Nav from "./components/Nav";
import { StoreProvider } from "./utils/GlobalState";
import FavoritesList from "./pages/FavoritesList";

function App() {
  return (
    <Router>
      <div>
        <StoreProvider>
          <Nav />
          <Switch>
            <Route exact path="/" component={Home} />
            <Route exact path="/home" component={Home} />
            <Route exact path="/favorites" component={FavoritesList} />
            <Route exact path="/posts/:id" component={Detail} />
            <Route component={NoMatch} />
          </Switch>
        </StoreProvider>
      </div>
    </Router>
  );
}
export default App;
```

```
export const UPDATE_POSTS = "UPDATE_POSTS";
export const REMOVE_POST = "REMOVE_POST";
export const SET_CURRENT_POST = "SET_CURRENT_POST";
export const ADD_POST = "ADD_POST";
export const LOADING = "LOADING";

export const ADD_FAVORITE = "ADD_FAVORITE";
export const REMOVE_FAVORITE = "REMOVE_FAVORITE";
export const UPDATE_FAVORITES = "UPDATE_FAVORITES";
```

```
import axios from "axios";
export default {
  // Gets all posts
  getPosts: function() {
    return axios.get("/api/posts");
  },
  // Gets the post with the given id
  getPost: function(id) {
    return axios.get("/api/posts/" + id);
  },
  // Deletes the post with the given id
  deletePost: function(id) {
    return axios.delete("/api/posts/" + id);
  },
  // Saves a post to the database
  savePost: function(postData) {
    return axios.post("/api/posts", postData);
  }
};
```

```
import React, { createContext, useReducer, useContext } from "react";

import {
  SET_CURRENT_POST,
  REMOVE_POST,
  UPDATE_POSTS,
  ADD_POST,
  ADD_FAVORITE,
  UPDATE_FAVORITES,
  REMOVE_FAVORITE,
  LOADING
} from "./actions";

const StoreContext = createContext();

const { Provider } = StoreContext;

const reducer = (state, action) => {
  switch (action.type) {
  case SET_CURRENT_POST:
    return {
      ...state,
      currentPost: action.post,
      loading: false
    };
  case UPDATE_POSTS:
    return {
      ...state,
      posts: [...action.posts],
      loading: false
    };
  case ADD_POST:
    return {
      ...state,
      posts: [action.post, ...state.posts],
      loading: false
    };
  case REMOVE_POST:
    return {
      ...state,
      posts: state.posts.filter((post) => {
        return post._id !== action._id;
      })
    };
  case ADD_FAVORITE:
    return {
      ...state,
      favorites: [action.post, ...state.favorites],
      loading: false
    };
  case UPDATE_FAVORITES:
    return {
      ...state,
      favorites: [...state.favorites],
      loading: false
    };
  case REMOVE_FAVORITE:
    return {
      ...state,
      favorites: state.favorites.filter((post) => {
        return post._id !== action._id;
      })
    };
  case LOADING:
    return {
      ...state,
      loading: true
    };
  default:
    return state;
  }
};

const StoreProvider = ({ value = [], ...props }) => {
  const [state, dispatch] = useReducer(reducer, {
    posts: [],
    currentPost: {
      _id: 0,
      title: "",
      body: "",
      author: ""
    },
    favorites: [],
    loading: false
  });
  return <Provider value={[state, dispatch]} {...props} />;
};

const useStoreContext = () => {
  return useContext(StoreContext);
};
export { StoreProvider, useStoreContext };
```

Mini-Project (partial) pt3

```
import React from "react";

// Exporting the Container, Row, and Col components from this file
// This Container component allows us to use a bootstrap container without worrying about class names
export function Container({ fluid, children }) {
  return <div className={`container${fluid ? "-fluid" : ""}`}>{children}</div>;
}
// This Row component lets us use a bootstrap row without having to think about class names
export function Row({ fluid, children }) {
  return <div className={`row${fluid ? "-fluid" : ""}`}>{children}</div>;
}
// This Col component lets us size bootstrap columns with less syntax
// e.g. <Col size="md-12"> instead of <div className="col-md-12">
export function Col({ size, children }) {
  return (
    <div
      className={size
        .split(" ")
        .map(size => "col-" + size)
        .join(" ")}
    >
      {children}
    </div>
  );
}
```

```
import React from "react";
import "./style.css";

// This file exports both the List and ListItem components
export function List({ children }) {
  return (
    <div className="list-overflow-container">
      <ul className="list-group">{children}</ul>
    </div>
  );
}

export function ListItem({ children }) {
  return <li className="list-group-item">{children}</li>;
}
```

```
import React from "react";
import "./style.css";

// The ...props means, spread all of the passed props onto this element
// That way we don't have to define them all individually
function DeleteBtn(props) {
  return (
    <span className="delete-btn" {...props} role="button" tabIndex="0">
      
    </span>
  );
}
export default DeleteBtn;
```

```
import React, { useRef } from 'react';
import { useStoreContext } from '../../utils/GlobalState';
import { ADD_POST, LOADING } from '../../utils/actions';
import API from '../../utils/API';

function CreatePostForm() {
  const titleRef = useRef();
  const bodyRef = useRef();
  const authorRef = useRef();
  const [state, dispatch] = useStoreContext();

  const handleSubmit = (e) => {
    e.preventDefault();
    dispatch({ type: LOADING });
    API.savePost({
      title: titleRef.current.value,
      body: bodyRef.current.value,
      author: authorRef.current.value,
    })
      .then((result) => {
        dispatch({
          type: ADD_POST,
          post: result.data,
        });
      })
      .catch((err) => console.log(err));
    titleRef.current.value = '';
    bodyRef.current.value = '';
  };
```

```
return (
    <div>
      <div className="jumbotron">
        <img
          className="img-fluid img-thumbnail"
          src="https://images.pexels.com/photos/459688/
                pexels-photo-459688.jpeg?auto=compress&
                cs=tinysrgb&dpr=2&h=650&w=940"
        />
      </div>
      <h1>Create a blog post</h1>
      <form className="form-group mt-5 mb-5"
              onSubmit={handleSubmit}>
        <label htmlFor="title">Title:</label>
        <input
          className="form-control mb-5"
          required
          ref={titleRef}
          id="title"
          placeholder="Title"
        />
```

```
        <label htmlFor="body">Body:</label>
        <textarea
          className="form-control mb-5"
          required
          ref={bodyRef}
          id="body"
          placeholder="Body"
        />
        <label htmlFor="screen name">Screen Name:</label>
        <input
          className="form-control mb-5"
          ref={authorRef}
          id="screen name"
          placeholder="Screen name"
        />
        <button
          className="btn btn-success mt-3 mb-5"
          disabled={state.loading}
          type="submit"
        >
          Save Post
        </button>
      </form>
    </div>
  );
}
export default CreatePostForm;
```