COMP 9783 Front-end Development

Lab-3-3 - React Events Lab. (4% of the course mark)

Name:

Student Number:

The React Events Lab is designed to provide a comprehensive, hands-on experience with handling events in React applications. Events are fundamental to creating interactive user interfaces, and this lab will cover the basics of event handling for ensuring responsive and efficient event-driven applications. Participants will learn how to handle various types of events such as the use of synthetic events.

Lab objectives:

1. Overview of event handling in React.
2. Demonstrate how to handle click and keyboard events.
3. Using event objects to access event properties.

Create a react app:

On **VSCode**, open the **terminal** and type the following commands:

1. **create-react-app react-events-app** and **press enter**. This will create the files and folders required by React.
2. On the **src folder**, **delete** the following files:
    a. App.css
    b. App.test.js
    c. index.css
    d. logo.svg
    e. reportWebVitals.js

f. setupTests.js

3. **Modify ./src/index.js** and **remove the following code** and **save the changes**:
    a. import './index.css';
    b. import reportWebVitals from './reportWebVitals';
    c. // If you want to start measuring performance in your app, pass a function
    d. // to log results (for example: reportWebVitals(console.log))
    e. // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
    f. reportWebVitals();
4. **Modify ./public/index.html** and **update the title html code** and **save the changes**:
    a. <title>react-events-app</title>
5. This lab will use **bulma** as the **css frontend**. To install **bulma** type: **npm install bulma** and **press enter**.
6. Overwrite **App.js** with the following code and **save the changes**:

```
import "bulma/css/bulma.min.css";
import Content from "./components/Content";

function App() {
  return <Content></Content>;
}

export default App;
```

7. Inside the **src** folder create a folder named: **components**.
8. Inside the **components folder**, create a file named: **ChangeEvent.js** and add the following code below:

```
function ChangeEvent() {
  return (
    <>
      <div class="field">
        <label class="label">Name</label>
        <div class="control">
          <input
            onChange={changeHandler}
            class="input"
            type="text"
            placeholder="Text input"
            id="textInput"
          />
        </div>
      </div>
      <div class="field">
        <label class="label">GBC Campus</label>
        <div class="select">
          <select id="dropDownList" onChange={changeHandler}>
            <option>Casaloma</option>
            <option>Downtown</option>
            <option>Waterfront</option>
          </select>
        </div>
      </div>
```

```
    <div class="field">
      <label class="label">Status</label>
      <div class="control">
        <label class="radio">
          <input
            id="fullTimeStatus"
            onChange={changeHandler}
            type="radio"
            name="status"
          />
            Fulltime
        </label>
        <label class="radio">
          <input
            id="partTimeStatus"
            onChange={changeHandler}
            type="radio"
            name="status"
          />
            Parttime
        </label>
      </div>
    </div>
  </>
 );
}

function changeHandler(event) {
 alert(`${event.target.id}: ${event.target.value}`);
```

```
}

export default ChangeEvent;
```

9. Inside the **components folder**, create a file named: **ClickEvent.js** and add the following code below:

```
function ClickEvent() {
  return (
    <div class="is-flex is-justify-content-center buttons">
      <button onClick={(event) => clickHandler(event)} class="button is-info">
        Info
      </button>
      <button
        onClick={(event) => clickHandler(event)}
        class="button is-success"
      >
        Success
      </button>
      <button
        onClick={(event) => clickHandler(event)}
        class="button is-warning"
      >
        Warning
      </button>
      <button onClick={(event) => clickHandler(event)} class="button is-danger">
        Danger
      </button>
    </div>
```

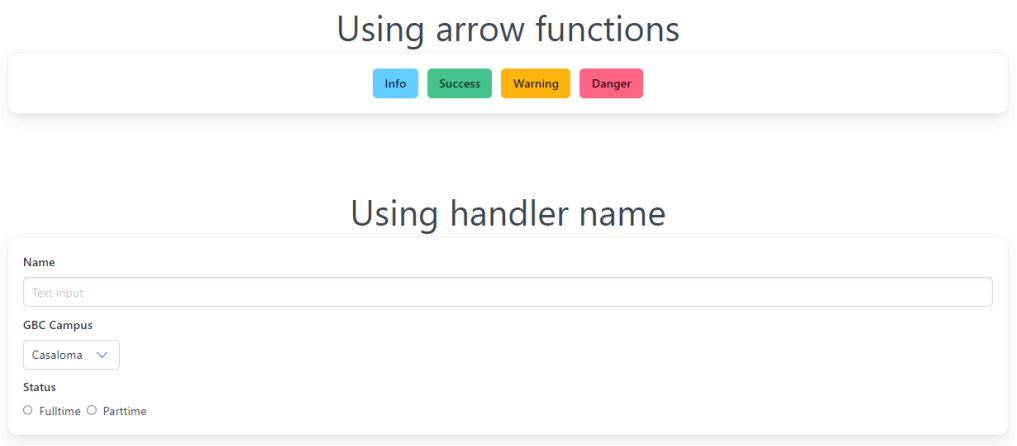
```
  );
}

function clickHandler(event) {
  alert(`Button: ${event.target.innerText} was clicked!`);
}

export default ClickEvent;
```

10. Inside the **components folder**, create a file named: **Content.js** and add the following code below:

```
import ClickEvent from "./ClickEvent";
import ChangeEvent from "./ChangeEvent";

function Content() {
  return (
    <>
      <section className="section">
        <div className="container">
          <h1 className="is-size-1 has-text-centered">Using arrow functions</h1>
          <div className="box">
            <ClickEvent></ClickEvent>
          </div>
        </div>
      </section>

      <section className="section">
        <div className="container">
```

```
        <h1 className="is-size-1 has-text-centered">Using handler name</h1>
        <div className="box">
          <ChangeEvent></ChangeEvent>
        </div>
      </div>
    </section>
   </>
  );
}

export default Content;
```

11. Open the **terminal**, ensure that you are on the **root folder of the app**, type **npm run start** and **press enter**. This should open a browser session and display the web app (**see below**). Make sure that your app is working before proceeding to the next step. Leave the app running.



12. Interact with the ui elements of the web app by clicking the buttons, radio buttons, drop down lists and entering text. The handler should display an alert box when an event is triggered.

13. **Update** the **onChange handler** of **ChangeEvent.js** and **save the changes**.

From: changeHandler

To: (event) => changeHandler(event)

14. **Update** the **onClick handler** of **ClickEvent.js** and **save the changes**.

From: (event) => clickHandler(event)

To: clickHandler

15. Navigate to http://localhost:3000 and the web app should still work since we just interchanged the 2 different ways of handling events.

Submission:

1. Create a new folder named **react** and copy **ChangeEvent.js** and **ClickEvent.js**.
2. Create a **zip file** of the **react folder**.
3. Submit the **zip file** to **GBC - D2L**.