Coding Activity 3.3

Managed And Unmanaged Forms With React (90:00)

#### **Learning Outcome Addressed**

 8.Build a form with React

**React Forms - Managed**

### React Forms

When we talk about managed vs unmanaged forms, we are talking about managed vs unmanaged state.

Note: You may hear the terms controlled and uncontrolled be used instead of managed and unmanged.

In a managed form, the state is controlled through Reacts state.

In contrast, unmanaged forms do not take advantage of Reacts state. This means data is managed through DOM as you would without React.

Note: When we talk about forms, this concept applies to React components as whole. Traditionally when you interact with data it is via forms. So that is why we use the word "form" to discuss this topic.

With managed forms it is important to remember state represents the rendered values, or in other words, what is currently on the screen. It is also important to remember that setting state is asynchronous, meaning if you console.log your state directly after calling setState you'll notice that the console.log will show the state prior to you calling setState. That's asynchronous code at work. Let's see this in action.

// name is our state, setName is our setState function  
const [name, setName] = React.useState('');  
function updateName(event) {  
  // our function that gets called when we want to update name  
  console.log(name); // output: Sa  
  console.log(event.target.value); // output: Sam  
  setName(event.target.value); // set name to "Sam"  
  console.log(name); // output: Sa  
}

#### Examine the following managed form

When examining be aware of the following:

* As someone types in the input element, the "I live in \_\_\_" is updated with what they typed.
* We create city and setCity using React.useState and initialize city as an empty string.
* setCity is the React function to set the value of city
* The input's value is using the reactive city variable
* The input's onChange uses an anonymous function to pass the event produced by onChange and calls the setCity function with the target.value of the event which is equal to what is currently typed into the input element.

function MyManagedForm() {  
  const [city, setCity] = React.useState('');  
  return (  
    <>  
      <form>  
        <label>City: </label>  
        <input type="text" value={city} onChange={(event) => setCity(event.target.value)} />  
        <input type="submit" value="Submit" />  
      </form>  
      <p>I live in {city}</p>  
    </>  
  );  
}

#### Now let's look at an unmanaged form

When examining be aware of the differences between the managed and unmanaged forms. Note the following:

* React.useState is not used so we do not have any variables or function to manage the value of the city input element, therefore we have to click the Submit button to update the screen.
* The input element uses an id to identify itself as the element for the city value
* The handleSubmit function talks directly to DOM to with document.getElementById to get the value of the input element and sets the innerHTML "I live in \_\_\_" using the same technique.

function MyUnmanagedForm() {  
  function handleSubmit(event) {  
    let city = document.getElementById('city').value;  
    document.getElementById('city-span').innerHTML = city;  
    event.preventDefault();  
  }  
  return (  
    <>  
      <form onSubmit={handleSubmit}>  
        <label>City: </label>  
        <input type="text" id="city" />  
        <input type="submit" value="Submit" />  
      </form>  
      <p>  
        I live in <span id="city-span"></span>  
      </p>  
    </>  
  );  
}

#### Activity Part 1 - Create a managed form

You will be working in the ManagedForm.jsx file for this part of the activity.

Complete the following tasks:

1. Create name and setName using React.useState and initialize with an empty string
2. On the form's input element, update the onChange to call the updateName function with the event as the argument
3. Add code to the updateName function to capture the user input and set the state of name

Task

    <script src="https://unpkg.com/react-dom@17/umd/react-dom.development.js" crossorigin></script>

    <script src="https://unpkg.com/@babel/standalone/babel.min.js" crossorigin></script>

    <!-- Load our React component. -->

    <script defer type="text/babel" src="./ManagedForm.jsx"></script>

    <script defer type="text/babel" src="./UnmanagedForm.jsx"></script>

  </body>

</html>

### React Forms - Unmanaged

#### Activity Part 2 - Create an unmanaged form

You will be working in the UnmanagedForm.jsx file for this part of the activity.

Complete the following tasks:

1. Add an id to the input element
2. Have the form call the submitForm function on submit
3. Set the innerHTML of the element with the id name-span to the value of the input element