React Forms and Form Validation

Lesson 10

Capgemini

Lesson Objectives

At the end of this module on React fundamentals you will be able to:

- Explain and demonstrate

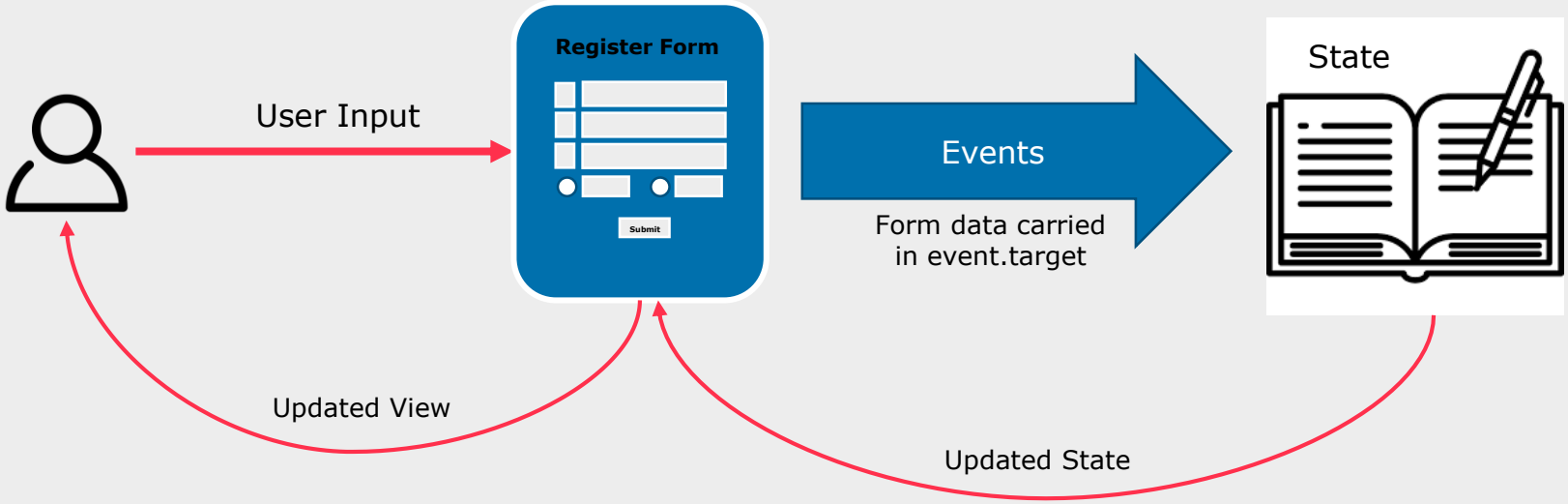
- Creating a Custom Dynamic Input Component
- Setting Up a JS Config for the Form
- Dynamically Create Inputs based on JS Config
- Adding a Dropdown Component
- Handling User Input
- Handling Form Submission
- Adding Custom Form Validation
- Fixing a Common Validation
- Adding Validation Feedback
- Showing Error Messages
- Handling Overall Form Validity

Introduction of Forms

- HTML form elements work a little bit differently from other DOM elements in React, because form elements naturally keep some internal state
- An input form element whose value is controlled by React in this way is called a "controlled component".

- **Controlled Components:**
- Each Form Elements in HTML maintains their own state and updates it based on user's input.
- React component that renders a form also controls what happens in that form on subsequent user input.
- render() { return <input type="text" name="title" value={this.state.title} /> }
  - The above code snippet is an example of Controlled components

React Forms




```
render() {
        return (
        <form onSubmit={this.handleSubmit}>
        <label> Name: <input type="text" value={this.state.value}
                                        onChange={this.handleChange} />
        </label> <input type="submit" value="Submit" />
         </form>
); }
```

Handling User Inputs

- React Forms can render different form elements to user to enter data
- React forms supports all type of HTML Elements
  - Like <input>, <select>, <textarea> etc

For Input element :

```
<input type="text" value={this.state.value} onChange={this.handleChange} />
```

For Text area :

```
<textarea value={this.state.address} onChange={this.handleChange} />
```

In the above first code snippet,

'value' attribute generally helps to get value from user ie whatever user types in text box. So whatever user types now gets stored in state.

{this.handleChange} is used to call function to handle any changes happening in text box. It uses setState() to change state of component, ie update whatever user is typing to state variable

Handling Form Submission

```
render() {
        return (
        <form onSubmit={this.handleSubmit}>
        <label> Name: <input type="text" value={this.state.value}
                                        onChange={this.handleChange} />
        </label> <input type="submit" value="Submit" />
         </form>
); }
```

Or you can give like

```
<Button
action={this.handleFormSubmit}>
```

Where inside the handleFormSubmit method we have to write logic after clicking submit as shown below:

```
handleFormSubmit(e) {
e.preventDefault();
let userData = this.state.newUser;
alert("you typed : "+JSON.stringify(userData));
console.log("Successful " + userData);
}
```

Adding Custom Form Validation

For adding a custom validation we have to make use of regular expression.

For our example we have used a CustomValidator js file which hold all our validation logic for different scenario, like validation must happen even if user is keep modifying the data in text box.

All the error messages are recorded in this files

Validation will happen in iterated manner in 3 different methods to validate data dynamically when user types.

In the demo for this we have used 3 methods

1. handleInputChange(event, inputPropName)
2. updateValidators(fieldName, value)
3. resetValidators()

Summary

- By now You should be clear with:

  - How to create React component
  - How to use props
  - How to perform prop validation
  - How to perform Static methods
  - How to use Nested-components
  - How to define Inline styles
  - How to add key for dynamic children
  - How to transfer props
  - How to use Event handlers
  - How to add State to a React component and how to use State-In-Composable-Components
  - In React, application data flows unidirectionally
  - How to execute React-component-life-cycle-methods
  - How to use Mixins

Review:

What are two ways the get values from <FORM> element?

1. Contolled components
2. Nested Components
3. input ref’s
4. routing

The _____ operator is called as spread operator.

1. “…”
2. “..”
3. []
4. !.!