upGrad ENTERPRISE

React Forms & Validations

By
Narasimha Rao T
Microsoft.Net FSD Trainer
Professional Development Trainer
tnrao.trainer@gmail.com

1. Introduction to React Forms

- In React, forms are used to collect user input.
- Unlike plain HTML, React controls form elements via **state**.
- The goal: keep form data in sync with the component's state.

2. Controlled Components

- A **controlled component** is an input element whose value is controlled by React state.
- Example:

```
function ControlledForm() {
  const [name, setName] = React.useState("");

  return (
    <form>
      <input  type="text"
        value={name}
        onChange={(e) => setName(e.target.value)} />
      <p>Hello, {name}</p>
    </form>
  );
}
```

3. Form Handling

- Typical steps in React form handling:
    i. Initialize state for form fields.
    ii. Bind state to input's `value` attribute.
    iii. Update state via `onChange`.
    iv. Handle submit with `onSubmit`.

- Example:

```
function LoginForm() {
  const [email, setEmail] = React.useState("");
  const [password, setPassword] = React.useState("");

  const handleSubmit = (e) => {
    e.preventDefault();
    console.log("Submitted:", { email, password });
  };

  return (
    <form onSubmit={handleSubmit}>
      <input type="email" value={email}
        onChange={(e) => setEmail(e.target.value)}
      />
      <input  type="password"  value={password}
        onChange={(e) => setPassword(e.target.value)}
      />
      <button type="submit">Login</button>
    </form>
  );
}
```

4. Validation Basics

- Validation ensures user input meets requirements (e.g., required fields, format checks).
- Common strategies:
  - Inline validation (real-time as user types).
  - On-submit validation (when form is submitted).
- Example rules:
  - Required field: `input.trim() !== ""`
  - Email format: Use regex or string methods.

5. Displaying Error Messages

- Store errors in state and display them conditionally.

```
function SignupForm() {
  const [email, setEmail] = React.useState("");
  const [error, setError] = React.useState("");

  const handleSubmit = (e) => {
    e.preventDefault();
    if (!email.includes("@")) {
      setError("Invalid email address");
    } else {
      setError("");
      console.log("Submitted:", email);
    }
  };
```

```
  return (
    <form onSubmit={handleSubmit}>
      <input type="email" value={email}
        onChange={(e) => setEmail(e.target.value)}
      />
      {error && <p style={{color: "red"}}>{error}</p>}
      <button type="submit">Submit</button>
    </form>
  );
}
```

upGrad ENTERPRISE

How to perform validations for multiple fields?

upGrad ENTERPRISE

Perform Validations using 3rd party libraries

6. Formik for Validations

- **Formik** is a popular library for handling forms in React.
- Provides hooks and components to manage:
  - State
  - Validation
  - Submission

Example: Basic Formik Form

```
import React from "react";
import { useFormik } from "formik";

function FormikForm() {
  const formik = useFormik({
    initialValues: {
      email: "",
      password: "",
    },
    validate: (values) => {
      const errors = {};
      if (!values.email) {
        errors.email = "Email is required";
      } else if (!values.email.includes("@")) {
        errors.email = "Invalid email address";
      }

      if (!values.password) {
        errors.password = "Password is required";
      } else if (values.password.length < 6) {
        errors.password = "Password must be at least 6 characters";
      }

      return errors;
    },
    onSubmit: (values) => {
      console.log("Form data:", values);
    },
  });
```

```
  return (
    <form onSubmit={formik.handleSubmit}>
      <input  type="email"  name="email"
        value={formik.values.email}
        onChange={formik.handleChange}
      />
      {formik.errors.email && <p style={{color: "red"}}>{formik.errors.email}</p>}

      <input  type="password"  name="password"
        value={formik.values.password}
        onChange={formik.handleChange}
      />
      {formik.errors.password && <p style={{color: "red"}}>{formik.errors.password}</p>}

      <button type="submit">Submit</button>
    </form>
  );
}

export default FormikForm;
```


upGrad
ENTERPRISE
Self-Check Questions

1. **What is a controlled component in React?**
    - An input element whose value is controlled by React state.
2. **Difference between controlled and uncontrolled components?**
    - Controlled: Value managed via React state.
    - Uncontrolled: Value managed by DOM (using `ref` to access).
3. **How do you handle form submissions in React?**
    - Use `onSubmit` with `preventDefault()` to avoid page reload.

4. **What are the advantages of using Formik?**
    - Simplifies state handling, validation, and submission logic.
5. **How do you perform validation without libraries like Yup?**
    - Write custom validation functions inside Formik's `validate`.
6. **What are common challenges in form handling in React?**
    - Managing complex state, handling nested forms, validation logic.

upGrad ENTERPRISE

Q & A

Narasimha Rao T
tnrao.trainer@gmail.com