

---

---

# React - Form

---

---

# How to add a form

You add a form with React like any other element:

```
function MyForm() {  
  return (  
    <form>  
      <label>Enter your name:  
        <input type="text" />  
      </label>  
    </form>  
  )  
}
```

# Handling form data

Handling forms is about how you handle the data when it changes value or gets submitted.

In HTML, form data is usually handled by the DOM.

In React, form data is usually handled by the components.

When the data is handled by the components, all the data is stored in the component state.

You can control changes by adding event handlers in the `onChange` attribute.

We can use the `useState` Hook to keep track of each inputs value and provide a "single source of truth" for the entire application.

# Control components

- The form elements are controlled via React state.
- React handles the value of the input fields via the `useState` hook.

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

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

  const handleChange = (event) => {
    setName(event.target.value); // Updates state when input changes
  };

  const handleSubmit = (event) => {
    event.preventDefault();
    alert("Submitted: " + name);
  };

  return (
    <form onSubmit={handleSubmit}>
      <input type="text" value={name} onChange={handleChange} />
      <button type="submit">Submit</button>
    </form>
  );
}
```

# Handling Multiple Inputs

For forms with multiple fields, use an object in state:

```
function MultiInputForm() {  
  const [formData, setFormData] = useState({ name: "", email: "" });  
  
  const handleChange = (event) => {  
    setFormData({  
      ...formData,  
      [event.target.name]: event.target.value,  
    });  
  };  
  
  const handleSubmit = (event) => {  
    event.preventDefault();  
    console.log("Form Data:", formData);  
  };  
  
  return (  
    <form onSubmit={handleSubmit}>  
      <input name="name" value={formData.name} onChange={handleChange} placeholder="Name" />  
      <input name="email" value={formData.email} onChange={handleChange} placeholder="Email" />  
      <button type="submit">Submit</button>  
    </form>  
  );  
}
```

# Handling Checkboxes

Use *checked* attribute to set the value

Use *onChange event* when check/uncheck

```
<input type="checkbox" checked={isChecked} onChange={handleCheckboxChange} />
```

```
const [isChecked, setIsChecked] = useState(false);
```


```
const handleCheckboxChange = (event) => {  
  setIsChecked(event.target.checked);  
};
```

# Handling Radio Buttons

```
<input type="radio" name="gender" value="male" onChange={handleRadioChange} /> Male  
<input type="radio" name="gender" value="female" onChange={handleRadioChange} /> Female
```

```
const [gender, setGender] = useState("");
```

```
const handleRadioChange = (event) => {  
  setGender(event.target.value);  
};
```

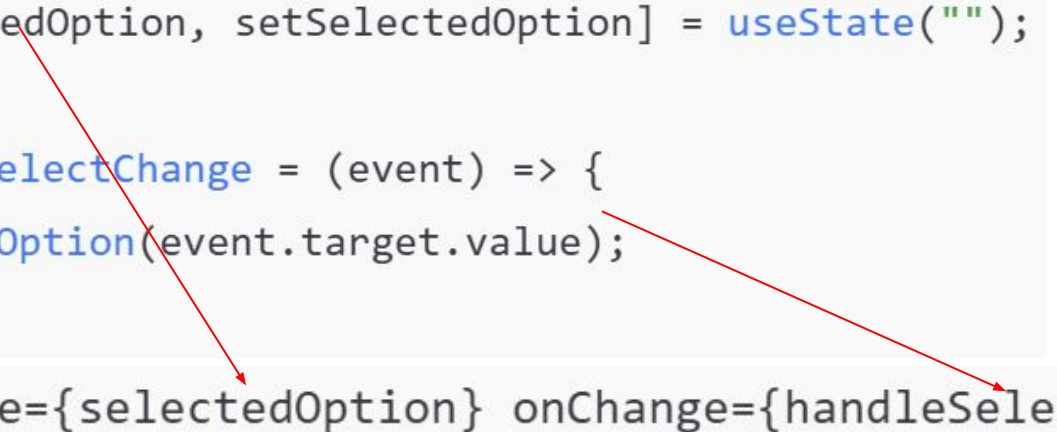


# Handling Select Dropdowns

```
const [selectedOption, setSelectedOption] = useState("");
```

```
const handleSelectChange = (event) => {  
  setSelectedOption(event.target.value);  
};
```

```
<select value={selectedOption} onChange={handleSelectChange}>  
  <option value="apple">Apple</option>  
  <option value="banana">Banana</option>  
  <option value="cherry">Cherry</option>  
</select>
```

A diagram with two red arrows. The first arrow starts at the 'selectedOption' variable in the first code block and points to the 'value={selectedOption}' prop in the first code block of the second code block. The second arrow starts at the 'handleSelectChange' function in the second code block and points to the 'onChange={handleSelectChange}' prop in the first code block of the second code block.




# Form validation

- You can use Html attributes:  
[https://www.w3schools.com/TAGs/ref\\_attributes.asp](https://www.w3schools.com/TAGs/ref_attributes.asp)
- **Formik** : <https://formik.org/>
- **React Hook Form**: <https://react-hook-form.com>

# Handling File Uploads

```
const handleFileChange = (event) => {  
  const file = event.target.files[0];  
  console.log("Selected file:", file);  
};
```

Uploading images, documents, etc.



```
<input type="file" onChange={handleFileChange} />
```

# Binding existing data to a form

Use **value** attribute for input with type :textbox, email, select

Use **checked** attribute for input with type :radio,checkbox

```
<div className="item">
  <label>Age:</label>
  <input type="number" max={150} min={1} required name="age" onChange={handleChange} value={formData.age} />
</div>
```

```
<div className="item">
  <label>Gender</label>
  <input type="radio" name="gender" value="male" onChange={handleChange} checked={formData.gender == "male"} /> Male
  <input type="radio" name="gender" value="female" onChange={handleChange} checked={formData.gender == "female"} /> Female
</div>
```

# Form Submission & API Calls

If you need to send form data to a backend:

```
const handleSubmit = async (event) => {  
  event.preventDefault();  
  const response = await fetch("https://api.example.com/submit", {  
    method: "POST",  
    headers: { "Content-Type": "application/json" },  
    body: JSON.stringify(formData),  
  });  
  const result = await response.json();  
  console.log("Server Response:", result);  
};
```