# REACT

## Example: Props With Class Based Component

```
import React,{Component} from 'react';
import Child from './components/child';
import ReactDOM from 'react-dom';

class Parent extends Component {
render() {

return (
 <div>
 <Child dataFromParent = "Passing the data using props"/>
 </div>

);} }
ReactDOM.render(<Parent/>, document.getElementById('root'));
```

Class based ***Parent*** Component

Props

***Output***


```
import React,{Component} from 'react';

class Child extends Component {
render() {

return (
     <div>
     <h1> We are learning :{this.props.dataFromParent}</h1
     </div>
);}}
export default Child
```

Class based ***Child*** Component

Accessing Props in Child Component

# REACT

## Example: Props With Function Based Component

```
import React,{Component} from 'react';
import Child from './components/child';
import ReactDOM from 'react-dom';

const Parent = () => {
return (
<div>
<Child dataFromParent = "Props with function based component"/>;
</div>
);
}
ReactDOM.render(<Parent/>, document.getElementById('root'));
```

Function based ***Parent*** Component

Props

***Output***


```
import React,{Component} from 'react';

const Child = () => {

return (
<div>
<h1> We are learning :{props.dataFromParent}</h1>
</div>
);
}
export default Child
```

Function based ***Child*** Component

Accessing Props in Child Component

# REACT STATES

## States

React uses an observable object called ***state,*** to observe the changes made to the component and guide the component to behave accordingly.

01 States are ***variables*** declared within the class component which holds some information that may change over the lifetime of the component

02 They are ***mutable***, as they hold the data that change over time and controls the behaviour of the component after each change

03 They are generally updated by ***event handlers*** and are ***modified*** using ***setState()*** method

We can define state in any class as below:

```
Class Sample extends React.Component
{
   constructor()
     {
       super();
       this.state = { attribute : "value" };
     }
}
```

# REACT STATES

## Demo: Working Of States

***Demo Steps***

- In this demo, you will learn how to change the displayed text using state method
- Create a component called ***Text*** and add its path to the main component
- Later add the below snippet and execute the code

src > components > JS text.js > ...

```
1   import React,{Component} from 'react';
2   import ReactDOM from 'react-dom';
3
4   class Text extends Component{
5       constructor(){
6           super()
7           this.state = {
8               text: 'Welcome students'
9           }}
10
11          changeText() {
12          this.setState({
13              text:'This is Class 2 of React'
14          })
15      }
16      render(){
17          return(
18
19                  <div>
20                  <h1>{this.state.text}</h1>
21                  <button onClick={() => this.changeText()}>Next</button>
22                  </div>
23              );
24              }
25  }
26  export default Text
```

- Class Component
- An object holding the data
- Props
- Method called to update current state
- New text to be printed on click of button
- Props
- Accessing the state
- Handler
- Event

# REACT FORM

## Handling Forms

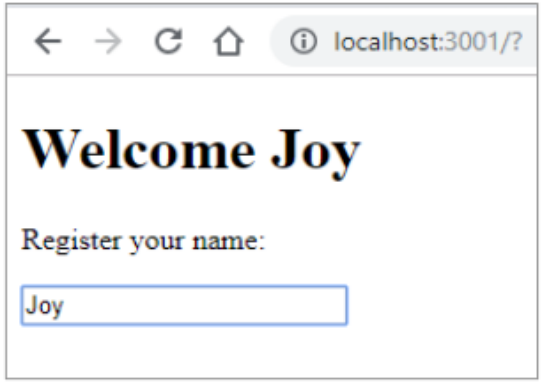
***Handling forms*** refers to managing the data on submission or when the values are changed.

```
class Form extends Component {
 constructor() {
  super();
  this.state = { participate: '' };
 }
 changeHandler = (event) => {
  this.setState({participate: event.target.value});
 }
 render() {
  return (
   <form>
   <h1>Welcome {this.state.participate}</h1>
   <p>Register your name:</p>
   <input
    type='text'
    onChange={this.changeHandler}
   />
   </form>
  );}}
ReactDOM.render(<Form />, document.getElementById('root'));
```

- State storing the data
- Records the changes
- Updated or submitted value

- React form is maintained by the React components and stored in component state
- These changes can be controlled by the addition of ***onChange*** attribute

***Output***



# REACT FORM

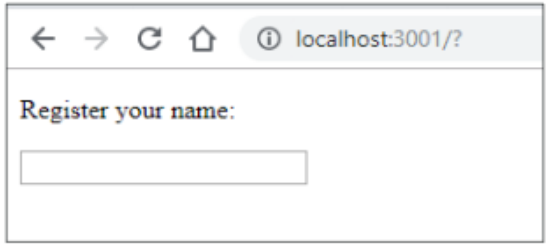
## Conditional Rendering

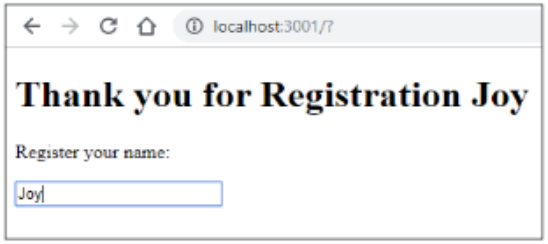
```
class Form extends Component {
  constructor(props) {
    super(props);
    this.state = { participate: '' };
  }
  changeHandler = (event) => {
    this.setState({participate: event.target.value});
  }
  render() {
    let header = '';
    if (this.state.participate) {
      header = <h1>Thank you for Registration {this.state.participate}</h1>;
    }
    return (
      <form>
      {header}
      <p>Register your name:</p>
      <input
        type='text'
        onChange={this.changeHandler}
      />
      </form>
    );}}
ReactDOM.render(<Form />, document.getElementById('root'));
```

***Conditional Rendering*** is usually preferred to display the data after user interaction (submission).

Condition to ***render Header*** after participate registration

***Output***



# REACT FORM

## Forms Submission

```
class Form extends Component {
  constructor() {
    super();
    this.state = { participate: '' };
  }
  submitHandler = (event) => {
    event.preventDefault();
    alert(this.state.participate + " Registered" );
  }
  changeHandler = (event) => {
    this.setState({participate: event.target.value});
  }
  render() {
    return (
      <form onSubmit={this.submitHandler}>
      <h1>Welcome</h1>
      <p>Register your name and click on submit:</p>
      <input
        type='text'
        onChange={this.changeHandler}
      />
      <input
        type='submit'
      />
      </form>
    );}}
ReactDOM.render(<Form />, document.getElementById('root'))
```

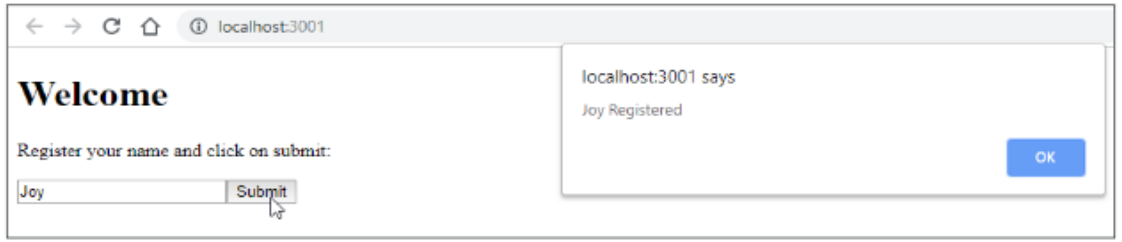
***Form Submission*** refers to submission of data with user confirmation by clicking the submit button.

Event to be submitted after clicking submit button

Submits the entered data

Defines a submit button which submits all form values to a form-handler

***Output***



# REACT FORM

## Multiple Input Fields

```
class Form extends Component {
  constructor() {
    super();
    this.state = {
      participate: '',
      roll_no: null,
    };}
  changeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
      <h1>Hello {this.state.participate} </h1>
      <p>Register your name:</p>
      <input
        type='text'
        name='participate'
        onChange={this.changeHandler}
      />
      <p>Enter your roll_no:</p>
      <input
        type='text'
        name='roll_no'
        onChange={this.changeHandler}
      />
      <h2>Your roll_no is {this.state.roll_no}</h2
>
      </form>
    );}}
ReactDOM.render(<Form/>, document.getElementById('
root'));
```

***Multiple input fields*** include different categories to be mentioned in the form.

- Manages the updated values of name and roll_no
- Collects the username
- Collects the user roll_no

***Output***


# REACT FORM

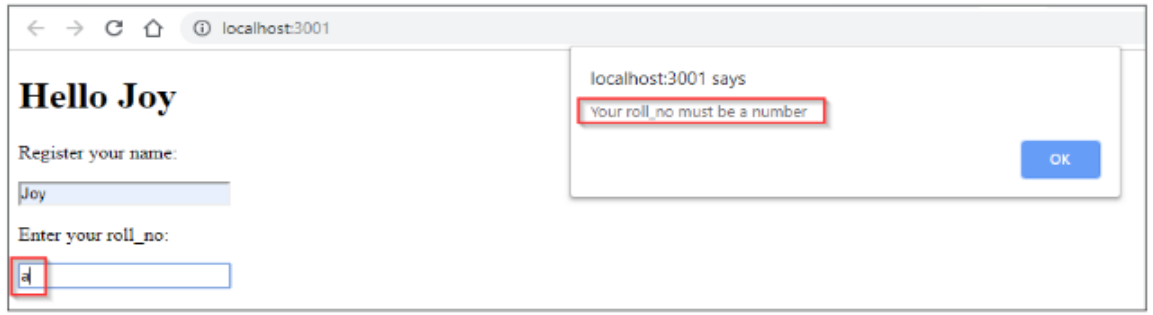
## Validating Form Input

```
class Form extends Component {
  constructor() {
    super();
    this.state = {
      participate: '',
      roll_no: null,
    };}
  changeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    if (nam === "roll_no") {
      if (!Number(val)) {
        alert("Your roll_no must be a number");
      } }
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
      <h1>Hello {this.state.participate} </h1>
      <p>Register your name:</p>
      <input
        type='text'
        name='participate'
        onChange={this.changeHandler}
      />
      <p>Enter your roll_no:</p>
      <input
        type='text'
        name='roll_no'
        onChange={this.changeHandler}
      />
      <h2>Your roll_no is {this.state.roll_no}</h2>
      </form>
    );}}
ReactDOM.render(<Form />, document.getElementById('root
'))
```

***Form validation*** refers to entering the right input, if user enters some wrong values then the input is not accepted.

***Output***: When you enter right data


***Output***: When you enter wrong data



# REACT FORM

## *Textarea*

***Textarea*** is one of the features of form, where data can be entered in textbox.

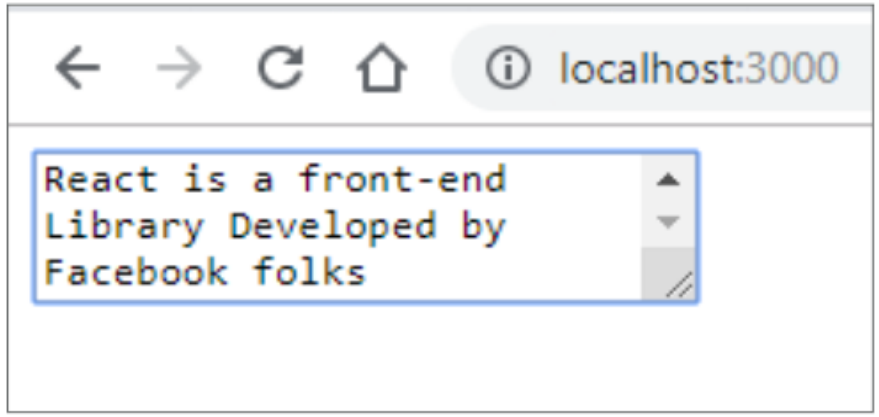
```
import React,{Component} from 'react';
import ReactDOM from 'react-dom';

class Form extends Component {
  constructor() {
    super();
    this.state = {
      description: 'React is a front-
end Library Developed by Facebook folks'
    };
  }
  render() {
    return (
      <form>
      <textarea value={this.state.description} />
      </form>
    );}}
ReactDOM.render(<Form />, document.getElementById('root'));
```

***Note***

In React the value of a textarea is placed in a ***value attribute***

***Output***



# REACT FORM

## Select

***Select*** feature offers list of options, where user is supposed to make a choice of appropriate option.

```
import React,{Component} from 'react';
import ReactDOM from 'react-dom';

class Form extends Component {
  constructor() {
    super();
    this.state = {
      myTraining: "choose"
    };
  }
  render() {
    return (
      <form>
      <select value={this.state.myTraining}>
        <option value="React">React</option>
        <option value="Angular">Angular</option>
        <option value="Node">Node.js</option>
      </select>
      </form>
    );}}
ReactDOM.render(<Form />, document.getElementById('root'));
```

***Note***

In React, the selected value is defined with a value attribute on the select tag

***Output***