**ReactJS Components**

A Component is one of the core building blocks of React. In other words, we can say that every application you will develop in React will be made up of pieces called components. Components make the task of building UIs much easier. You can see a UI broken down into multiple individual pieces called components and work on them independently and merge them all in a parent component which will be your final UI.

You can see in the below image we have broken down the UI of GeeksforGeeks’s homepage into individual components.

Google’s custom search at the top can be seen as an individual component, the navigation bar can be seen as an individual component, the sidebar is an individual component, the list of articles or post is also an individual component and finally, we can merge all of these individual components to make a parent component which will be the final UI for the homepage.

Components in React basically return a piece of JSX code that tells what should be rendered on the screen. In React, we mainly have two types of components:

**1-Functional Components:** Functional components are simply javascript functions. We can create a functional component in React by writing a javascript function. These functions may or may not receive data as parameters, we will discuss this later in the tutorial. Below example shows a valid functional component in React:

const Democomponent=()=>

{

return <h1>Welcome Message!</h1>;

}

**2-Class Components:** The class components are a little more complex than the functional components. The functional components are not aware of the other components in your program whereas the class components can work with each other. We can pass data from one class component to other class components. We can use JavaScript ES6 classes to create class-based components in React. Below example shows a valid class-based component in React:

class Democomponent extends React.Component

{

render(){

return <h1>Welcome Message!</h1>;

}

}

**React State**

The state is an updatable structure that is used to contain data or information about the component. The state in a component can change over time. The change in state over time can happen as a response to user action or system event. A component with the state is known as stateful components. It is the heart of the react component which determines the behavior of the component and how it will render. They are also responsible for making a component dynamic and interactive.

A state must be kept as simple as possible. It can be set by using the **setState()** method and calling setState() method triggers UI updates. A state represents the component's local state or information. It can only be accessed or modified inside the component or by the component directly. To set an initial state before any interaction occurs, we need to use the **getInitialState()** method.

For example, if we have five components that need data or information from the state, then we need to create one container component that will keep the state for all of them.

Example:-

import React, { Component } from 'react';

class App extends React.Component {

constructor() {

super();

this.state = { displayBio: true };

}

render() {

const bio = this.state.displayBio ? (

<div>

<p><h3>Javatpoint is one of the best Java training institute in Noida, Delhi, Gurugram, Ghaziabad and Faridabad. We have a team of experienced Java developers and trainers from multinational companies to teach our campus students.</h3></p>

</div>

) : null;

return (

<div>

<h1> Welcome to JavaTpoint!! </h1>

{ bio }

</div>

);

}

}

export default App;

**React Props**

Props stand for "Properties." They are read-only components. It is an object which stores the value of attributes of a tag and work similar to the HTML attributes. It gives a way to pass data from one component to other components. It is similar to function arguments. Props are passed to the component in the same way as arguments passed in a function.

Props are immutable so we cannot modify the props from inside the component. Inside the components, we can add attributes called props. These attributes are available in the component as this.props and can be used to render dynamic data in our render method.

When you need immutable data in the component, you have to add props to reactDom.**render()** method in the main.js file of your ReactJS project and used it inside the component in which you need. It can be explained in the below example.

Example:-

import React, { Component } from 'react';

class App extends React.Component {

render() {

return (

<div>

<h1> Welcome to { this.props.name } </h1>

<p> <h4> Javatpoint is one of the best Java training institute in Noida, Delhi, Gurugram, Ghaziabad and Faridabad. </h4> </p>

</div>

);

}

}

export default App;

Main.js:-

port React from 'react';

import ReactDOM from 'react-dom';

import App from './App.js';

ReactDOM.render(<App name = "JavaTpoint!!" />, document.getElementById('app'));