**JAVASCRIPT Assignment**

**MODULE: 18 (REACT – Components, State, Props)**

* **Introduction to React.js**

1. What is React.js? How is it different from other JavaScript frameworks and libraries? Ans. React Js is an open-source Javascript library for building user interface, developed by Facebook, which focuses on building reusable components for single page application. It uses virtual DOM for efficient rendering and JSX for combining HTML with Javascript.

React.js is different from other frameworks and libraries in following ways:

1. **Library, Not a Framework**:

* React focuses only on the **view layer**.
* Frameworks like Angular provide complete solutions (routing, state management, etc.).

1. **Virtual DOM**:

* React uses Virtual DOM for faster updates.
* Angular and Vue manipulate the real DOM (less efficient).

1. **Flexibility**:

* React is unopinionated; developers choose their tools.
* Angular is opinionated with a fixed structure.

1. **JSX Syntax**:

* React uses JSX (HTML in JavaScript).
* Angular uses templates with specific syntax.

1. **Data Binding**:

* React supports one-way data binding (predictable).
* Angular uses two-way data binding (can be complex).

1. **Learning Curve**:

* React is easier to learn compared to Angular.
* Vue has the simplest learning curve.

1. **Performance**:

* React’s Virtual DOM ensures high performance.
* Angular and Vue are slightly less optimized for frequent updates.

1. **Community Support**:

* React has a large community and ecosystem.
* Angular and Vue have smaller but active communities.

1. Explain the core principles of React such as the virtual DOM and component based architecture

Ans. Below are the core principles of React:

1. **Virtual DOM:**

* React creates a virtual representation of the actual DOM in memory.

***How it works:***

* When changes occur, React updates the Virtual DOM first.
* It then compares the updated Virtual DOM with the previous version using a diffing algorithm.
* Only the changed elements are efficiently updated in the real DOM.

***Benefits:***

* Improves performance by avoiding costly direct updates to the real DOM.
* Ensures smooth rendering for dynamic and interactive UIs.

1. **Component based Architecture**:

* React divides the UI into small, reusable components.

***What are components?***

* Components are independent, self-contained units that manage their own state and logic.
* Two types of components:
* Functional Components (with React Hooks for state and lifecycle management).
* Class Components (older way with lifecycle methods).

***Benefits:***

* Reusability: Components can be reused across the app.
* Maintainability: Easy to manage and debug smaller parts of the UI.
* Scalability: Helps build complex UIs by combining simple components.

1. What are the advantages of using React.js in web development?

Ans. Following are the advantages of using React.js in web development:

1. **Component-Based Architecture**

* Breaks the UI into reusable, modular components.
* Promotes better code organization, reusability, and easier maintenance.

1. **Virtual DOM for Performance**

* React uses a Virtual DOM to minimize direct updates to the real DOM.
* Ensures faster rendering and improves app performance.

1. **Unidirectional Data Flow**

* React follows one-way data binding, making data flow predictable.
* Simplifies debugging and state management.

1. **Declarative UI**

* Developers describe "what" the UI should look like; React handles "how" to render it.
* Makes the code easier to read, write, and debug.

1. **Strong Community and Ecosystem**

* A vast community provides tools, libraries (e.g., Redux, React Router), and solutions.
* React is widely adopted and well-documented.

1. **JSX Syntax**

* Combines HTML and JavaScript in one file for better readability.
* Allows developers to write UI components in a familiar syntax.

1. **React Hooks**

* Functional components can manage state and side effects using hooks like useState, useEffect.
* Simplifies development without needing class components.

1. **SEO-Friendly**

* React applications can be rendered on the server using Server-Side Rendering (SSR).
* Improves SEO and initial page load speed.

1. **Flexibility**

* React is a library, not a framework, offering flexibility to integrate with other tools and libraries.
* Works well with modern technologies like Redux, GraphQL, and Next.js.

1. **Cross-Platform Development**

* React can be used to build web, mobile (via React Native), and desktop applications.
* **JSX (Javascript XML)**

1. What is JSX in React.js? Why is it used?

Ans. JSX (JavaScript XML) is a syntax extension for JavaScript that allows you to write HTML-like code within JavaScript. It makes React code more readable and easier to define the UI structure directly in the JavaScript file.

**Uses of JSX:**

* Allows you to write HTML and JavaScript logic in the same file.
* Example: We can use JavaScript expressions, functions, and variables directly in JSX.
* JSX is easier to read and write compared to using React.createElement() syntax
* Developers can write cleaner and more intuitive code that reflects the UI structure.
* JSX compiles to React elements that React uses to efficiently update the Virtual DOM.

1. How is JSX different from regular JavaScript? Can you write JavaScript inside JSX?

Ans. JSX allows you to write HTML-like code within JavaScript files.

Regular JavaScript requires manual creation of DOM elements using methods like React.createElement() or document.createElement().

JSX allows you to embed JavaScript expressions inside curly braces {} directly within the HTML-like code.

Regular JavaScript doesn’t have this direct feature.

In JSX, attributes like class and for are replaced with className and htmlFor to avoid conflicts with JavaScript keywords.

In regular JavaScript, such adjustments are unnecessary.

* Yes, we can write JavaScript expressions inside JSX using curly braces {}.

1. Discuss the importance of using curly braces {} in JSX expressions.

Ans. In JSX, curly braces {} are used to embed JavaScript expressions directly within the HTML-like syntax. They allow dynamic content to be displayed or computed inside React components.

* **Components (Functional and Class components)**

1. What are components in React? Explain the difference between functional components and class components.

Ans. Components are independent, self-contained units that manage their own state and logic.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Functional Component** | **Class Component** |
| Definition | Javascript function that returns JSX | ES6 classes that extends React.Component |
| Syntax | Simpler and cleaner function-based syntax. | Uses class syntax with a render() method. |
| “this” keyword | No this keyword used. | Requires this to access state, props, and methods. |
| Boilerplate code | Less boilerplate; more concise. | More boilerplate code required. |

1. How do you pass data to a component using props?

Ans. Following is the way to pass data to a component using props:

* Props are used to pass data from a parent component to a child component in React.
* In the parent component, data is passed to the child component by specifying attributes with values when rendering the child.
* The child component receives the data as an object, which is accessible through the props parameter.
* Props can contain any type of data, such as strings, numbers, objects, arrays, and even functions.
* Props are read-only, meaning that child components cannot modify the values they receive from the parent.
* Functional components receive props as a parameter, while class components access them through this.props.
* Props provide a mechanism for reusing components by allowing them to accept different values each time they are rendered.
* Default values for props can be set in case a parent does not provide a value.
* Parent components can pass functions as props to child components, enabling the child to invoke those functions, such as for event handling.
* Props enable a unidirectional data flow, ensuring that data flows from the parent to the child components only, making the application easier to understand and debug.

1. What is the role of render() in class components?

Ans. Below is the explanation of the role of render() in class components:

* The render() method is a required part of class components in React and is responsible for rendering the component's UI.
* It returns JSX (or React elements), which is the description of what should be displayed on the screen.
* The render() method is called automatically by React every time there is a change in state or props, causing the component to re-render and update the UI.
* It is pure, meaning it does not modify component state or cause side effects; it only returns JSX based on the current props and state.
* The render() method can return:
  + JSX elements (e.g., HTML-like syntax in React)
  + null or false if the component doesn't need to render anything
  + Arrays of JSX elements to render multiple components or elements