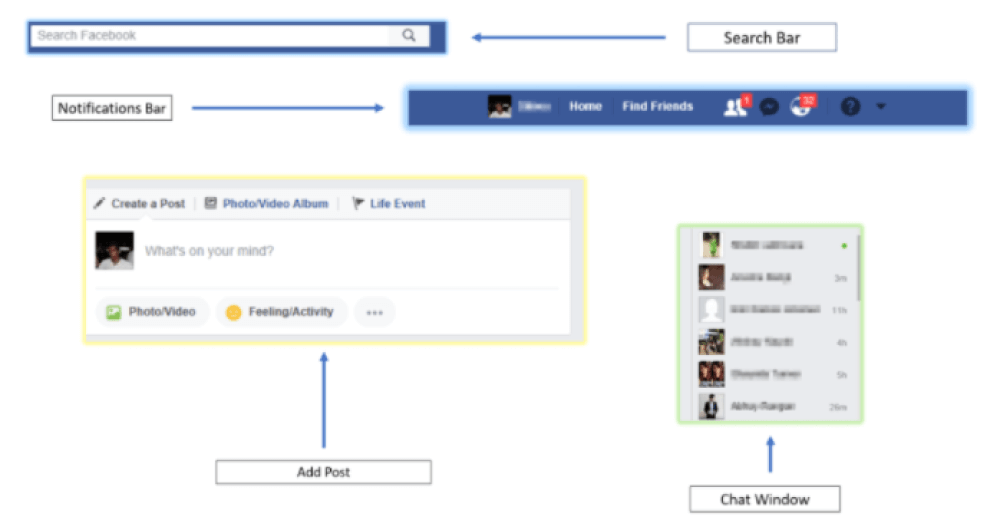
**What is ReactJS?**

React.js is a front-end JavaScript framework developed by Facebook. To build composable user interfaces predictably and efficiently using declarative code, we use React. It’s an open-source and component-based framework responsible for creating the application’s view layer.

* ReactJs follows the Model View Controller (MVC) architecture, and the view layer is accountable for handling mobile and web apps.
* React is famous for building single-page applications and mobile apps.

**Let’s take an example**: Look at the Facebook page, which is entirely built on React, to understand how react does works.



As the figure shows, ReactJS divides the UI into multiple components, making the code easier to debug. This way, each function is assigned to a specific component, and it produces some HTML which is rendered as output by the DOM.

**ReactJS History:**

Jordan Walke created React, who worked as a software engineer in Facebook has first released an early React prototype called "FaxJS.”

In 2011, React was first deployed on Facebook's News Feed, and later in 2012 on Instagram.

[](https://bit.ly/3if9dmk)

The **current version of React.JS is V17.0.1**.

**Why do people choose to program with React?**

There are various reasons why you should choose ReactJS as a primary tool for website UI development. Here, we highlight the most notable ones and explain why these specifics are so important:

* **Fast** - Feel quick and responsive through the Apps made in React can handle complex updates.
* **Modular** - Allow you to write many smaller, reusable files instead of writing large, dense files of code. The modularity of React is an attractive solution for JavaScript's visibility issues.
* **Scalable** - React performs best in the case of large programs that display a lot of data changes.
* **Flexible** - React approaches differently by breaking them into components while building user interfaces. This is incredibly important in large applications.
* **Popular** - ReactJS gives better performance than other JavaScript languages due to t’s implementation of a virtual DOM.
* **Easy to learn** - Since it requires minimal understanding of HTML and JavaScript, the learning curve is low.
* **Server-side rendering and SEO friendly** - ReactJS websites are famous for their server-side rendering feature. It makes apps faster and much better for search engine ranking in comparison to products with client-side rendering. React even produces more opportunities for website SEO and can occupy higher positions on the search result’s page.
* **Reusable UI components** - React improves development and debugging processes.
* **Community** - The number of tools and extensions available for ReactJS developers is tremendous. Along with impressive out-of-box functionalities, more opportunities emerge once you discover how giant the React galaxy is. React has a vibrant community and is supported by Facebook. Hence, it’s a reliable tool for website development.

The above reasons justify why ReactJS is more popular than other frameworks. Now, let’s familiarize ourselves with ReactJS features.

**-----       Also Read:**[**React JS Tutorial for Beginners**](https://mindmajix.com/react-js-tutorial)**-----**

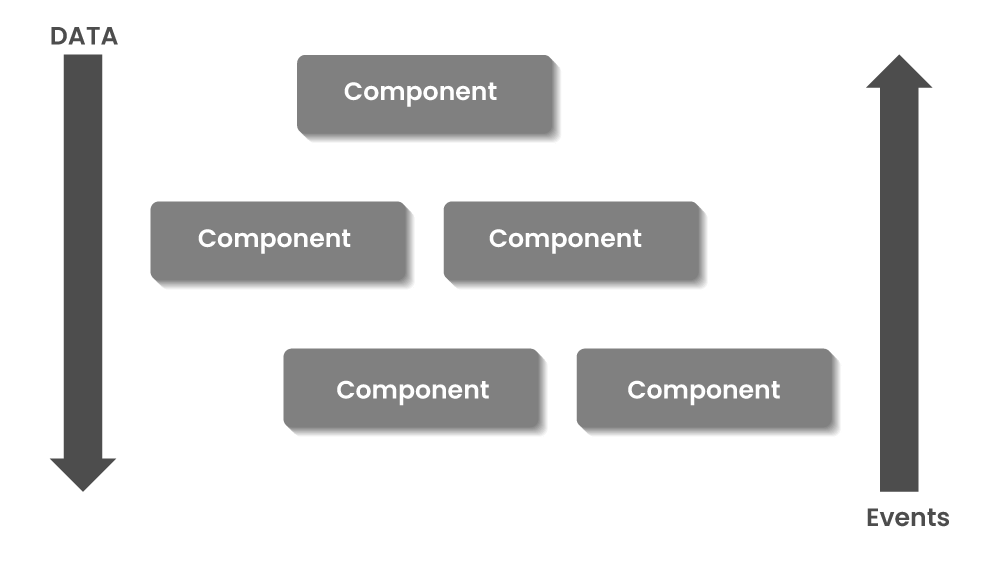
**ReactJS Features:**

**1. JSX - JavaScript Syntax Extension**

JSX is a preferable choice for many web developers. It isn't necessary to use JSX in React development, but there is a massive difference between writing react.js documents in JSX and JavaScript. JSX is a syntax extension to JavaScript. By using that, we can write HTML structures in the same file that contains JavaScript code.

**2. Unidirectional Data Flow and Flux**

React.js is designed so that it will only support data that is flowing downstream, in one direction. If the data has to flow in another direction, you will need additional features.

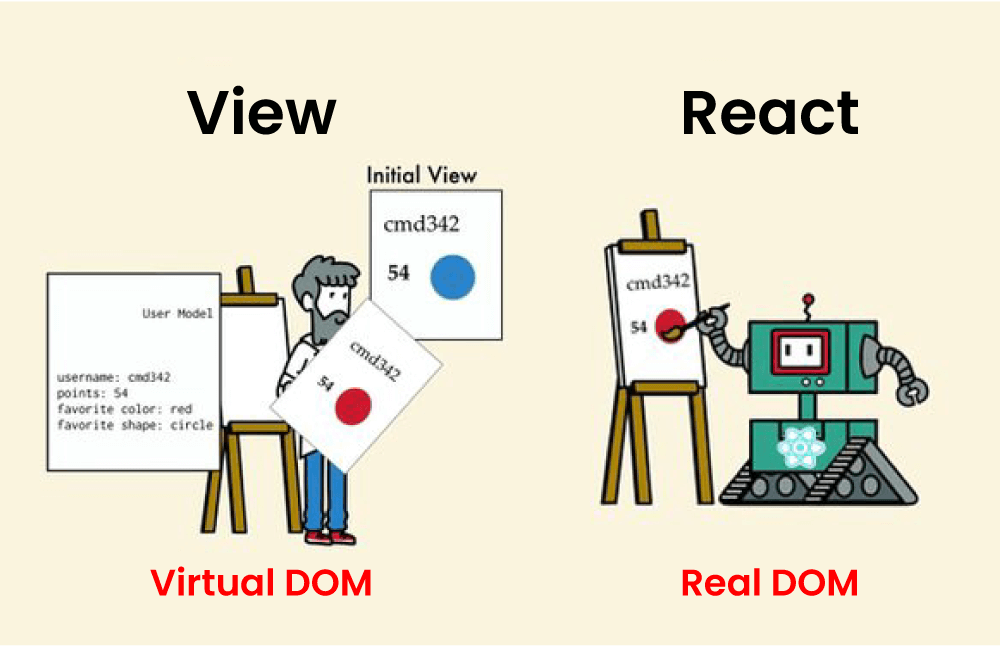


React contains a set of immutable values passed to the component renderer as properties in HTML tags. The components cannot modify any properties directly but support a call back function to do modifications.

**3. Virtual Document Object Model (VDOM)**

React contains a lightweight representation of real DOM in the memory called Virtual DOM. Manipulating real DOM is much slower compared to VDOM as nothing gets drawn on the screen. When any object’s state changes, VDOM modifies only that object in real DOM instead of updating whole objects.

That makes things move fast, particularly compared with other front-end technologies that have to update each object even if only a single object changes in the web application.



**4. Extensions**

React supports various extensions for application architecture. It supports server-side rendering, Flux, and Redux extensively in web app development. React Native is a popular framework developed from React for creating cross-compatible mobile apps.

**5. Debugging**

Testing React apps is easy due to large community support. Even Facebook provides a small browser extension that makes React debugging easier and faster.

Next, let’s understand some essential concepts of ReactJS.

**Building Components of React - Components, State, Props, and Keys.**

**1. ReactJS Components**

Components are the heart and soul of React. Components (like JavaScript functions) let you split the UI into independent, reusable pieces and think about each piece in isolation.

Components are building blocks of any React application. Every component has its structures, APIs, and methods.

In React, there are two types of components, namely stateless functional and stateful class.

* **Functional Components** - These components have no state of their own and contain only a render method. They are simply Javascript functions that may or may not receive data as parameters.

Stateless functional components may derive data from other components as properties (props).

An example of representing functional component is shown below:

function WelcomeMessage(props) {

return <h1>Welcome to the , {props.name}</h1>;

}

* **Class Components** - These components are more complex than functional components. They can manage their state and to return JSX on the screen have a separate render method. You can pass data from one class to other class components.

An example of representing class component is shown below:

class MyComponent extends React.Component {

render() {

return (

<div>This is the main component.</div>

);

}

}

**2. React State**

A state is a place from where the data comes. The state in a component can change over time, and whenever it changes, the component re-renders.

A change in a state can happen as a response to system-generated events or user action, and these changes define the component’s behavior and how it will render.

class Greetings extends React.Component {

state = {

name: "World"

};

updateName() {

this.setState({ name: "Mindmajix" });

}

render() {

return(

<div>

{this.state.name}

</div>

)

}

}

The state object is initialized in the constructor, and it can store multiple properties.

For changing the state object value, use this.setState() function.

To perform a merge between the new and the previous state, use the setState() function.

**-----      Also Read:**[**React JS vs Angular JS**](https://mindmajix.com/react-js-vs-angular-js)**-----**

**3. React Props**

Props stand for properties, and they are read-only components.

Both Props and State are plain JavaScript objects and hold data that influence the output of render. And they are different in one way: State is managed within the component (like variable declaration within a function), whereas props get passed to the component (like function parameters).

Props are immutable, and this is why the component of a container should describe the state that can be changed and updated, while the child components should only send data from the state using properties.

**4. React Keys**

In React, Keys are useful when working with dynamically created components. Setting the key value will keep your component uniquely identified after the change.

They help React in identifying the items which have changed, are removed, or added.

In summary, State, Props, keys, and components are the few fundamental React concepts that you need to be familiar with before working on it.

[Frequently Asked React JS Interview Question & Answers](https://mindmajix.com/reactjs-interview-questions)

**ReactJS Advantages:**

* React.js creates its own virtual DOM. This will improve apps performance since JavaScript virtual DOM is faster than the regular DOM.
* ReactJS helps to create awesome UI.
* ReactJS is SEO-friendly.
* Component and Data patterns improve readability which helps to maintain larger apps.
* React can be used with other frameworks.
* React simplifies the complete process of the scripting environment.
* It facilitates advanced maintenance and increases productivity.
* Ensures faster rendering
* The best thing about React is the provision of a script for mobile app development.
* A strong community backs ReactJS.
* React JS comes with a helpful developer toolset
* Both startups and fortune 500 companies use React.

**Is React Worth Learning?**

React has been the most used web framework by developers worldwide over the years. Most of the businesses are inclining towards React because of the simplicity it offers.

React developers are drawing lucrative salaries compared to other technology professionals. Due to the rising demand for React across the companies, the demand for talented professionals is high across all geographies.

Indeed shows the average salaries for [React developers in the US](https://www.payscale.com/research/US/Skill=React.js/Salary) range between 55k to 110k USD.

The popularity of React has been consistently gaining, and the Google Trends chart also shows the speed with which it is rising consistently with adding new features over time.