What is ReactJS? What can ReactJS be used for?

If you want to create fast, dynamic user interfaces for your web applications, then you need to learn how to use ReactJS. React is a client-side JavaScript library, which means it runs on the user's machine in the browser, not on the server.

ReactJS was created in 2011 by the tech giant, Facebook.

The React library is used to build dynamic user interfaces and works by separating aspects of the user interface into what are called "components".

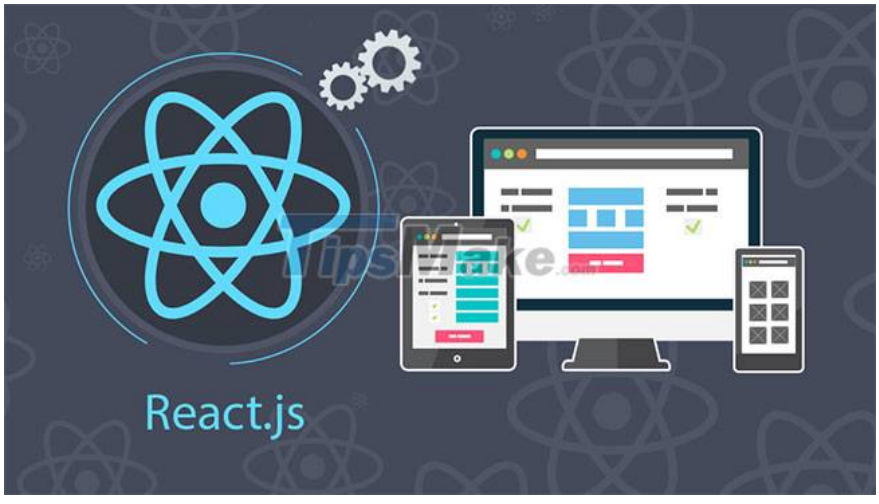
In this article, Tipsmake will walk you through everything there is to know about React and its components.

What is ReactJS?

React (also known as ReactJS) is an open source JavaScript library, often mistakenly referred to as a framework. This is because React is a direct competitor of top JavaScript frameworks like AngularJS and VueJS.

React is a library because it doesn't have a routing mechanism among other framework-specific features. However, there are tools like react-router that can be installed and used with the library to achieve framework functionality.

React is more closely related to frameworks like Angular/Vue than to other libraries in languages ??like jQuery.



What are the benefits of using ReactJS?

Developers use React for a variety of reasons. Some people use it for speed and performance. Others use React simply because of its popularity. However, there are three main benefits of using a framework that all developers can appreciate.

1. React allows you to build your interfaces using what are called "reusable components" that have state and data.
2. React uses JavaScript Syntax Extension (JSX) that allows users to write dynamic HTML.
3. It uses the Virtual Document Object Model (VDOM), which allows developers to update specific parts of a web page without having to reload the page.

What are ReactJS components?

React treats each part of the user interface as a component. Components have their own state, methods, and functions.

They allow developers to separate the user interface into specific parts that are easily combined to create complex user interfaces. Therefore, if you want to create a customer management tool, one component of the user interface can be dedicated to adding new customers, while another component of the same user interface can be reserved. to display a list of customers.

In its simplest form, each component is a JavaScript class or function. They take input values ??called 'props' and return specific aspects of the user interface as React elements. For some developers, defining a component as a function is simpler than defining it as a class. However, using either method achieves the same output in React.

Create component with a function instance

```
function Customer() { return (
```

Paul Wilson

1. Phone: 222-222-2222
2. Email:
3. Balance: $0.00

```
); } export default Customer;
```

Create component with a class instance

```
import React from 'react'; class Customer extends React.Component { render() { re
```

Paul Wilson

1. Phone: 222-222-2222
2. Email:
3. Balance: $0.00

```
); } } export default Customer;
```

As you can see from the examples above, a lot happens when you create a component using a class. The first important thing to note is that you must use the render() function when creating a class component.

As you know, you can't return directly from a class, so render() helps with this.

The main reason why a developer might choose to use a class instead of a function is because classes have state, but thanks to the advent of hooks, React functions can now also have state.