**React JS** is today's most popular **JavaScript Library** for building User Interfaces, which has created by Facebook. We can build modern, fast Single Page Applications or websites with React.

Since React is so popular in the market and beneficial to know for a Web/Frontend Developer, I decided to cover some features of React JS in my following articles, as 4 parts:

**Part 1:**React JS for Beginners — The Basics

**Part 2:**[React JS — Understanding Functional & Class Components](https://codeburst.io/react-js-understanding-functional-class-components-e65d723e909)

* **Part 3:**[React Props](https://medium.com/@cem.eygi/what-is-props-and-how-to-use-it-in-react-da307f500da0)
* **Part 4:** [React JS — Understanding State](https://codeburst.io/react-js-understanding-state-e875911e921c)

***Important:****Before learning React, you need to know****JavaScript (and ES6 Features)***

## What is ECMAScript 6?

## ECMAScript 6 is also known as ES6 and ECMAScript 2015.

## Some people call it JavaScript 6.

## some of the new features in ES6.

* JavaScript let
* JavaScript const
* JavaScript Arrow Functions
* JavaScript Classes
* Default parameter values
* Array.find()
* Array.findIndex()
* Exponentiation (\*\*) (EcmaScript 2016)

**Is React JS a Library or a Framework?**

This is one of the most unclear subjects of React. Let’s make this clear from the beginning. **React is a Library, not a Framework.**

## What is a Library?

A library in programming can be explained as a collection of codes. We use a library to write code in a much simpler way or to import a feature from it into our project. **JQuery** **is a library for example.**

We can write JavaScript much simpler by using JQuery, or we can import written JQuery features to our project. **The project itself is not dependent on a library.**

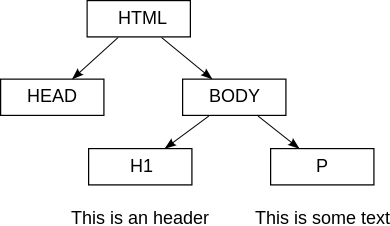
## What is a Framework?

A Framework, on the other hand, is a complete package of code with its own functionalities & libraries. A Framework has its own rules, you don’t have much flexibility and the project is dependent on the Framework you use. **Angular is an example of a framework.**

So React is for building User Interfaces, and how you program the rest of the project is up to you. Like JQuery, you can include React in your project partially, or completely. **So React JS a library.**

**React Virtual DOM**

To understand the importance of **React Virtual DOM**, first, you need to know what **DOM (Document Object Model)** is. DOM is basically the representation of the HTML code in a webpage. **The document** is the webpage itself, the **objects** are the HTML tags. And finally, the **model** of DOM is a tree structure:



**Tree Structure of the Document Object Model**

## What is the benefit of Virtual DOM?

Each time you make a change in the code, DOM will be completely updated and rewritten. This is an expensive operation and consumes lots of time. In this point, **React**provides a solution: **The Virtual DOM.**

**So when something changes:**

* React first creates an exact copy of the DOM
* Then React figures out which part is new and only updates that specific part in the **Virtual DOM**
* Finally, React copies only the new parts of the **Virtual DOM** to the **actual DOM**, rather than completely rewriting it.

This approach makes a webpage much faster than a standard webpage. **That’s also one of the reasons why React is so popular.**

In classic Frontend programming, we have separated HTML, CSS and JS file structures. React is a bit different. We don’t have separated HTML files in React.

In JSX syntax, we write HTML tags inside JavaScript.

In React, for example, a simple JavaScript variable can be like this:

**const** element **=** **<h1 >**Hello!**</h1>;**

Normally, we can’t assign an HTML tag to a JavaScript variable. But with JSX, we can. The code above you see is neither HTML nor JavaScript. It’s an example of JSX.

## So what is this JSX?

**JSX (JavaScript XML)** is a syntax extension to JavaScript used by React. JSX is basically used to write HTML tags inside JavaScript. Later, the JSX code will be translated into normal JavaScript, by **Babel.**

In summary, React doesn’t have HTML files, HTML tags are rendered directly inside JavaScript. **This approach makes React faster.**

## Do I have to work with JSX?

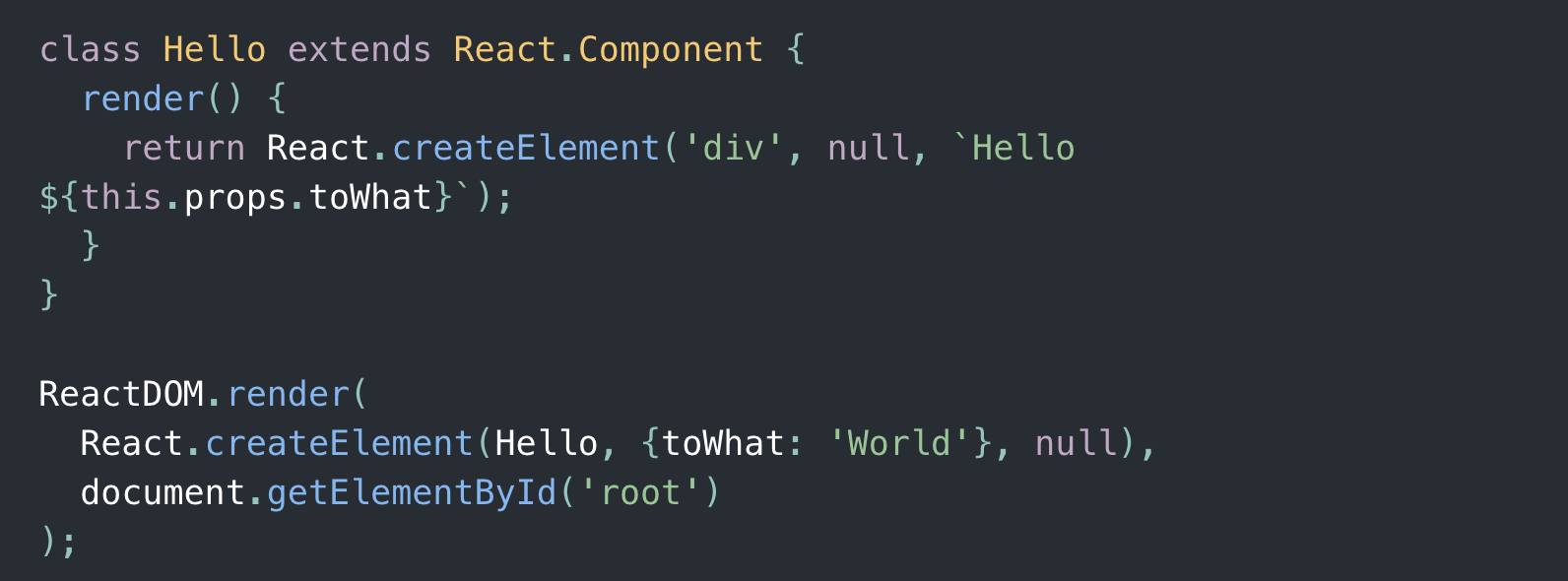
You don’t have to use JSX with React, but it is strongly recommended. JSX simplifies React and makes it easier to read. Let me give an example of React code with and without JSX.

**React with JSX:**



[**https://reactjs.org/docs/react-without-jsx.html**](https://reactjs.org/docs/react-without-jsx.html)

We have a React class named **“Hello”**. This is a **default**React class with its **render** function. The class returns an HTML **div** element so it can be rendered later as a component, anywhere in your project. Below the class, there is a special React DOM **Render**function which is calling the Hello class, as a component **(<Hello />)** and specifies where **(root)** your React code will be printed.**React without JSX:**



Same React code without JSX And here is the same React code as JavaScript but without JSX. Which one is easier for you?

**Some important rules about JSX:** We can’t return more than one HTML element at once, but we can wrap the elements inside a parent HTML tag:

**class** Test extends React.Component {  
**render()** {  
**return** (  
<div className=”Allborder” >  
<p>Hello</p>  
<p>World</p>  
</div>  
);  
}  
}

We can use JSX inside **for** loops, **if-else**cases:

**render()** {  
**if**(condition==true) {  
**return <p>**This text**</p>**;  
} **else** {  
**return <p>**Another text**</p>**;  
}  
}

HTML attribute names like **“class”** becomes **“className”**,

**“tabindex”**becomes **“tabIndex”**as **camelCase.**

**<div** **className**="myClass"**></div>**

* HTML tags must always be closed