React Tutorial

React is a JavaScript library for building user interfaces.

React is used to build single page applications.

React allows us to create reusable UI components.

Index.html

<body>

    <div *id*="root"></div>

    <script *src*="https://unpkg.com/react@17/umd/react.development.js" *crossorigin*></script>

    <script *src*="https://unpkg.com/react-dom@17/umd/react-dom.development.js" *crossorigin*></script>

    <script *src*="app.js"></script>

</body>

App.js

const e = React.createElement;

class Test *extends* React.Component {

    render() {

        return e(

            'h1',

            '',

            'Hello World!'

        );

    }

}

ReactDOM.render(e(Test), document.querySelector('#root'));

## What is React?

React is a JavaScript library created by Facebook.

React is a tool for building UI components.

## How does React Work?

React creates a VIRTUAL DOM in memory.

Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM.

React only changes what needs to be changed!

React finds out what changes have been made, and changes **only** what needs to be changed.

## React.JS History

Initial Release to the Public (V0.3.0) was in July 2013.

React.JS was first used in 2011 for Facebook's Newsfeed feature.

Facebook Software Engineer, Jordan Walke, created it.

The create-react-app version 2.0 package was released in October 2018.

Create-react-app version 2.0 supports Babel 7, webpack 4, and Jest23.

# React Getting Started

To use React in production, you need NPM and Node.js

To get an overview of what React is, you can write React code directly in HTML.

## React Directly in HTML

Start by including three scripts, the first two let us write React code in our JavaScripts, and the third, Babel, allows us to write JSX syntax and ES6 in older browsers.

<html>

<script *src*="https://unpkg.com/react@16/umd/react.production.min.js"></script>

<script *src*="https://unpkg.com/react-dom@16/umd/react-dom.production.min.js"></script>

<script *src*="https://unpkg.com/babel-standalone@6.15.0/babel.min.js"></script>

<body>

    <div *id*="mydiv"></div>

    <script *type*="text/babel">

        class Hello *extends* React.Component {

            render() {

                return <h1>Hello World!</h1>

            }

        }

        ReactDOM.render(<Hello />, document.getElementById('mydiv'))

    </script>

</body>

</html>

This way of using React can be OK for testing purposes, but for production you will need to set up a **React environment**.

Index.html

<!DOCTYPE *html*>

<html *lang*="en">

<head>

  <meta *charset*="utf-8" />

  <meta *name*="viewport" *content*="width=device-width, initial-scale=1" />

  <title>React App</title>

</head>

<body>

  <div *id*="root"></div>

</body>

</html>

Index.js

import React from 'react';

import ReactDOM from 'react-dom';

class Test *extends* React.Component {

  render() {

    return <h1>Mojahid Islam</h1>;

  }

}

ReactDOM.render(<Test />, document.getElementById('root'));

React ES6

React uses ES6, and you should be familiar with some of the new features like:

* Classes
* Arrow Functions
* Variables (let, const, var)

## Classes

ES6 introduced classes.

A class is a type of function, but instead of using the keyword function to initiate it, we use the keyword class, and the properties are assigned inside a constructor() method.

## Arrow Functions

### About this

The handling of this is also different in arrow functions compared to regular functions.

In short, with arrow functions there are no binding of this.

In regular functions the this keyword represented the object that called the function, which could be the window, the document, a button or whatever.

With arrow functions, the this keyword always represents the object that defined the arrow function.

# React Render HTML

React's goal is in many ways to render HTML in a web page.

React renders HTML to the web page by using a function called ReactDOM.render().

## The Render Function

The ReactDOM.render() function takes two arguments, HTML code and an HTML element.

The purpose of the function is to display the specified HTML code inside the specified HTML element.

ReactDOM.render(<p>Hello</p>, document.getElementById('root'));

## The HTML Code

The HTML code in this tutorial uses JSX which allows you to write HTML tags inside the JavaScript code: