# React

## Introduction:

- React is an open-source JavaScript library for building user interfaces.
- React is used to build **single page applications**.
- **Single page application** is such application which is loaded once and rest of the work is done by JavaScript without reloading the page.
- Instead of manipulating the browser's DOM directly, react creates a virtual DOM in memory, where it does all the necessary changes, before making the changes in the browser DOM.
- React is based on components based where we build a website by breaking into chunks and we can use these components again.
- React is very popular because it follows **write once use everywhere** principle.

## History:

- React is created by **Facebook**.
- React.JS was first used in 2011 for Facebook's Newsfeed feature.
- Facebook Software Engineer, **Jordan Walke** created it.
- The first version of react (V0.3.0) was released in **2013**.
- The latest version of react is (V18.2.0).

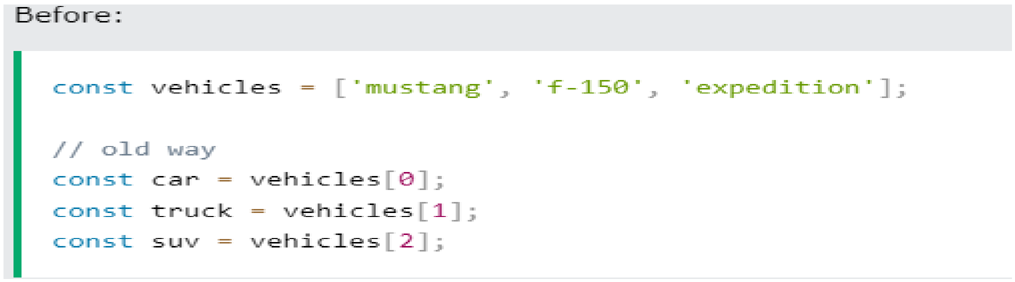
## React ES6:

- ECMAScript6 is the 6th version of JavaScript created to standardize JavaScript in 2015 and is also known as ECMAScript 2015.
- React uses ES6 as it introduced the following features:
    1. ***classes** concept.*
    2. ***arrow functions**.*
    3. ***let** and **const** variable declaration.*
    4. *array methods like .**map**().*

5. ***destructuring.***
6. ***spread operator***.
7. ***modules***.
8. *ternary operator. {if else = ? :}*

**Destructuring:**

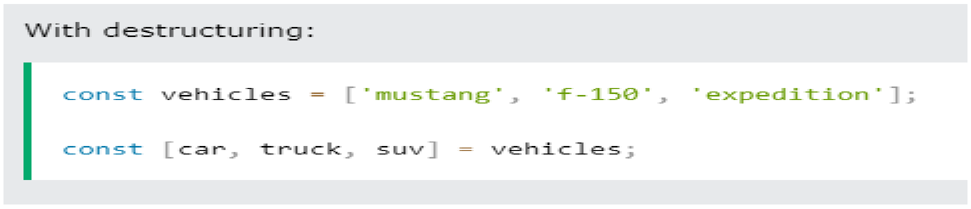
Destructuring makes it easy to extract exactly what we need from an array or an object.



Before:

```
const vehicles = ['mustang', 'f-150', 'expedition'];

// old way
const car = vehicles[0];
const truck = vehicles[1];
const suv = vehicles[2];
```



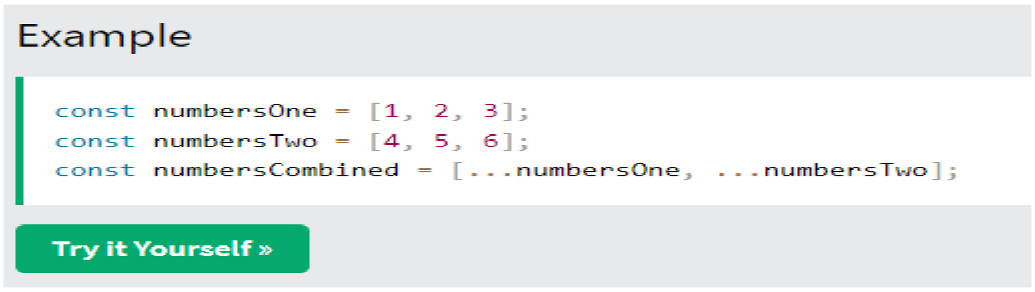
With destructuring:

```
const vehicles = ['mustang', 'f-150', 'expedition'];

const [car, truck, suv] = vehicles;
```

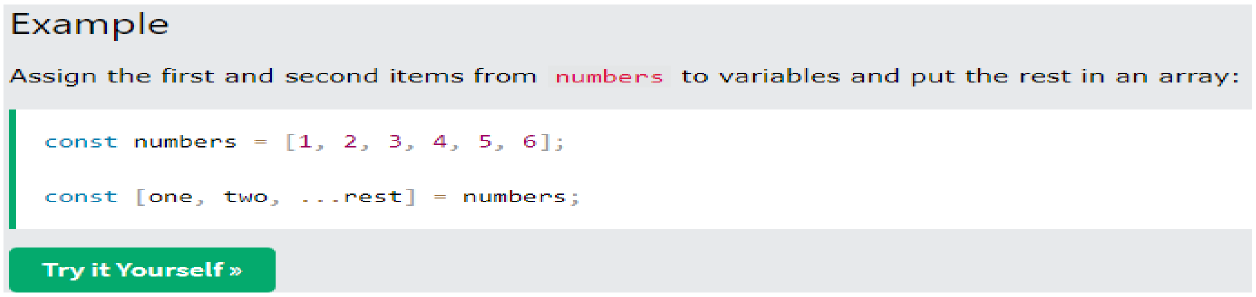
**Spread Operator:**

The JS operator (. . .) allows us to quickly copy all or part of an existing array or object into another array or object.



Example

```
const numbersOne = [1, 2, 3];
const numbersTwo = [4, 5, 6];
const numbersCombined = [...numbersOne, ...numbersTwo];
```


Example

Assign the first and second items from `numbers` to variables and put the rest in an array:

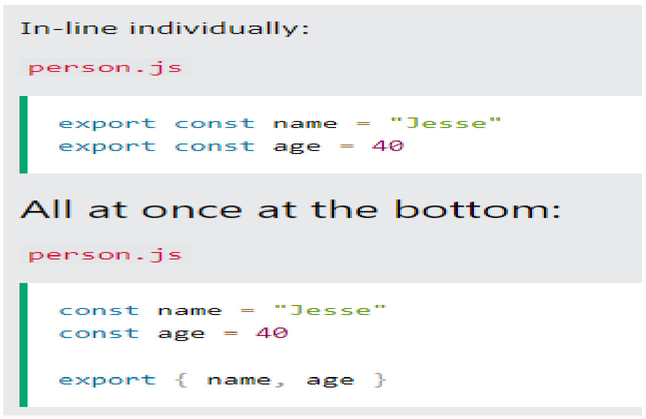
```
const numbers = [1, 2, 3, 4, 5, 6];

const [one, two, ...rest] = numbers;
```

## Modules:

- ES6 modules allow us to break up our code into separate files.
- ES6 modules rely on **import** and **export** statements.
- We can export a function or a variable from any file.
- There are two types of exports **Named** and **Default**.

  **Named exports:**


In-line individually:

`person.js`

```
export const name = "Jesse"
export const age = 40
```

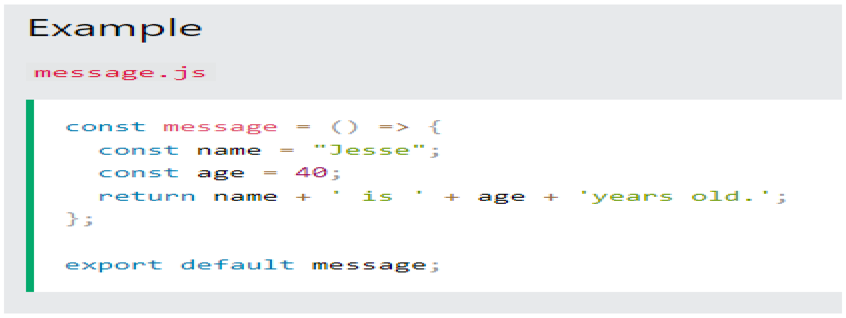
All at once at the bottom:

`person.js`

```
const name = "Jesse"
const age = 40

export { name, age }
```

**Default exports:**

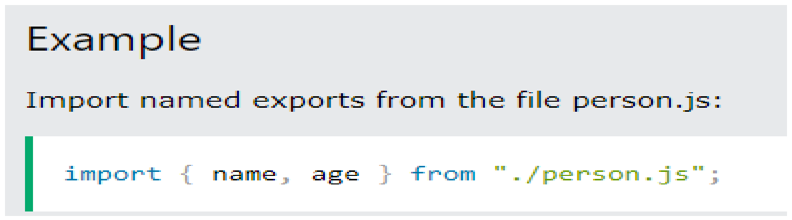

Example

`message.js`

```
const message = () => {
  const name = "Jesse";
  const age = 40;
  return name + ' is ' + age + 'years old.';
};

export default message;
```

**Imports**


Example

Import named exports from the file person.js:

```
import { name, age } from "./person.js";
```

## JSX:

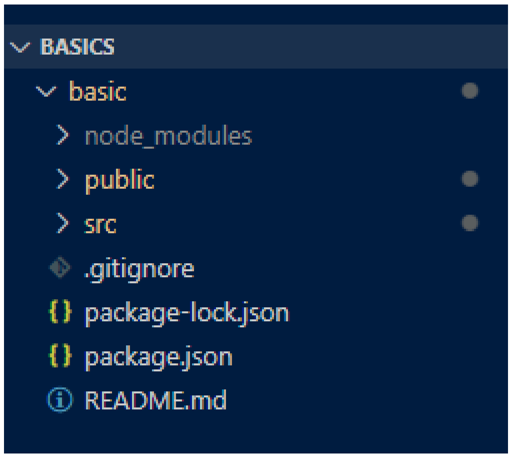
- **JSX** stands for **JavaScript XML**, **JSX** is a syntax extension for JS that lets us write HTML-Like markup inside a JS file.
- **JSX** is stricter and has a few more rules than HTML like ***Rendering a single root element, close all the tags, camelCase naming conventions.***
- Using **JSX** we can combine HTML and JS code.
- **Babel** compiles JSX to **React.createElement() calls**.
- When we want to render multiple elements in JSX, we should use **JSX fragment tag <> </>.**
- ***Example***: *Return* ***(<>****<h1>Hello</h1><div><h2>Hii</h2></div>* ***</>)***

## React Environment Setup:

- Before we start with **React.js** we need to have **Node.js** installed in our system as **React.js** is a JavaScript library and **Node.js** is the **runtime environment** that allows us to run JavaScript on the server side.
- Download latest version of Node.js here: Link
- After successfully installing verify the installation by entering the following **command** in command prompt: **node -v**
- If Node.js is installed correctly then you will be able to see a version of Node.js or else, you will get an error.
- We also require a **code editor** to work with our code and one of the most used **code editors** is **VS Code.**
- Download latest version of VS Code here: Link

## Getting Started with React App:

- Create a react app package by using the following command inside your directory: **npx create-react-app app_name**
- **npx(node package execute)** is a package executer, and it is used to execute JavaScript packages directly, without installing them.
- **npm(node package manager)** is a package manager used to install, delete, and update JavaScript packages on your machine.
- This will create a react app and we can start working on **React.js.**
- A basic react app structure looks like this:



```
BASICS
  basic
    node_modules
    public
    src
    .gitignore
    package-lock.json
    package.json
    README.md
```

**node_modules (Folder):** *Contains all the dependencies that are needed for an initial working react app.*

**.gitignore (file):** *This file specifies intentionally untracked files that Git should ignore.*

**package-lock.json (file):** It ensures that your package is consistent across various machines by storing the versions of which dependencies are installed with your package

**package.json (file):** *It specifies the dependencies being used in the project which helps npm setup same environment on different machine for our project.*

**README.md (file):** *This file can be used to define usage, build instructions, summary of project, etc. It uses markdown markup language to create content.*

***public (Folder):***

- *The "public" folder in a React project contains static assets that are directly served to the client without processing by Webpack or any other build tool.*
- *It usually includes the **HTML file(s), images, fonts, and other assets** that don't need to be processed by the JavaScript bundler.*
- *The main **HTML file (index.html)** typically resides here. This file is where your React application is injected and rendered into the DOM.*
- *Assets placed in the public folder are often referenced directly from the HTML or JSX files without the need for imports.*

***src (Folder):***

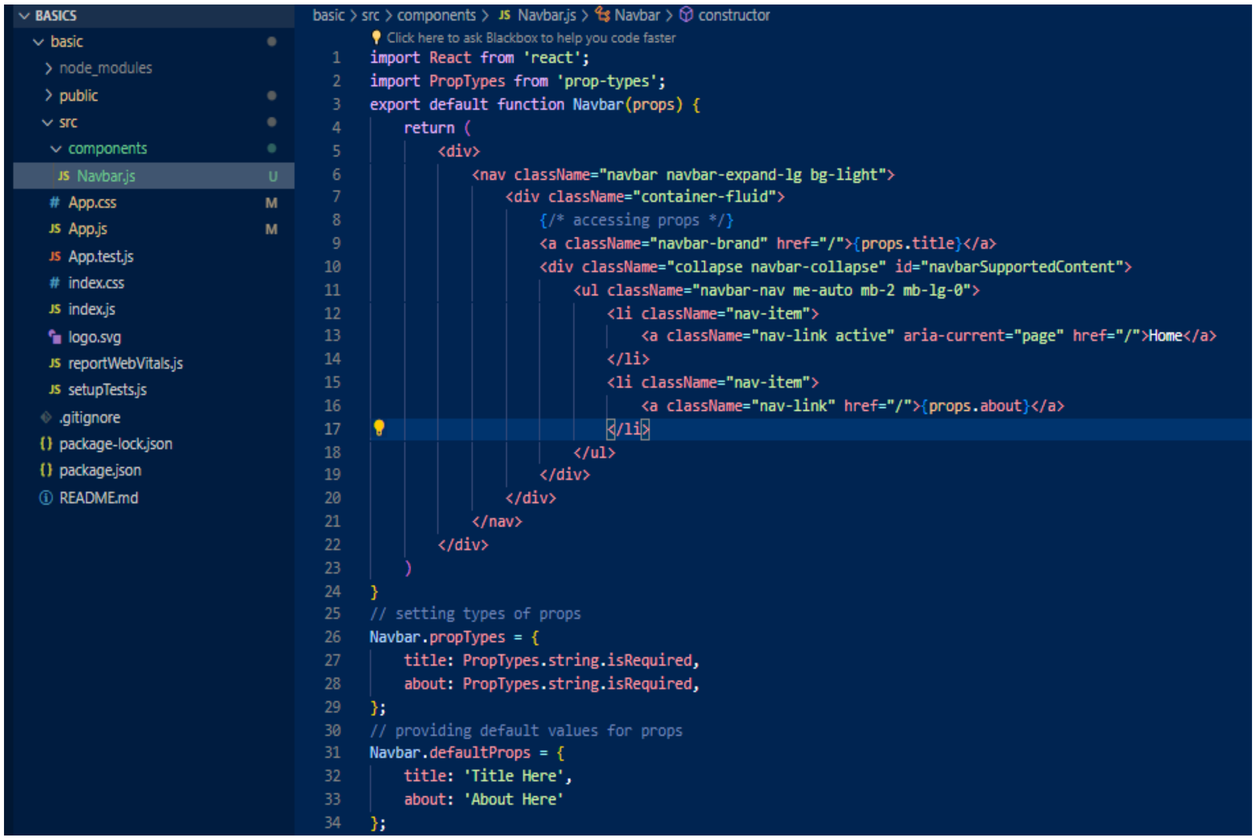
- *The "src" folder contains the source code of your React application.*
- *This is where you write your **React components, JavaScript files, CSS or Sass files, and any other code** related to your application's logic and presentation.*
- *Typically, the main **JavaScript file (index.js or App.js)** that initializes your React application is located here.*
- *React components and other modules are organized within the "src" folder, often in subdirectories based on their functionality or feature.*
- *Code in the "src" folder is processed by build tools like Webpack and Babel to transform it into a format that can be understood by the browser.*

**Props:**

- Props is short form of properties and “props” are like arguments in a function.
- In React.JS, props are used to pass data from one component to another component.
- Props are **Read-Only**, we can set typed of props by using **propTypes,** we can also provide default values for **props using defaultProps, and we can also make props to be mandatory by using isRequired.**

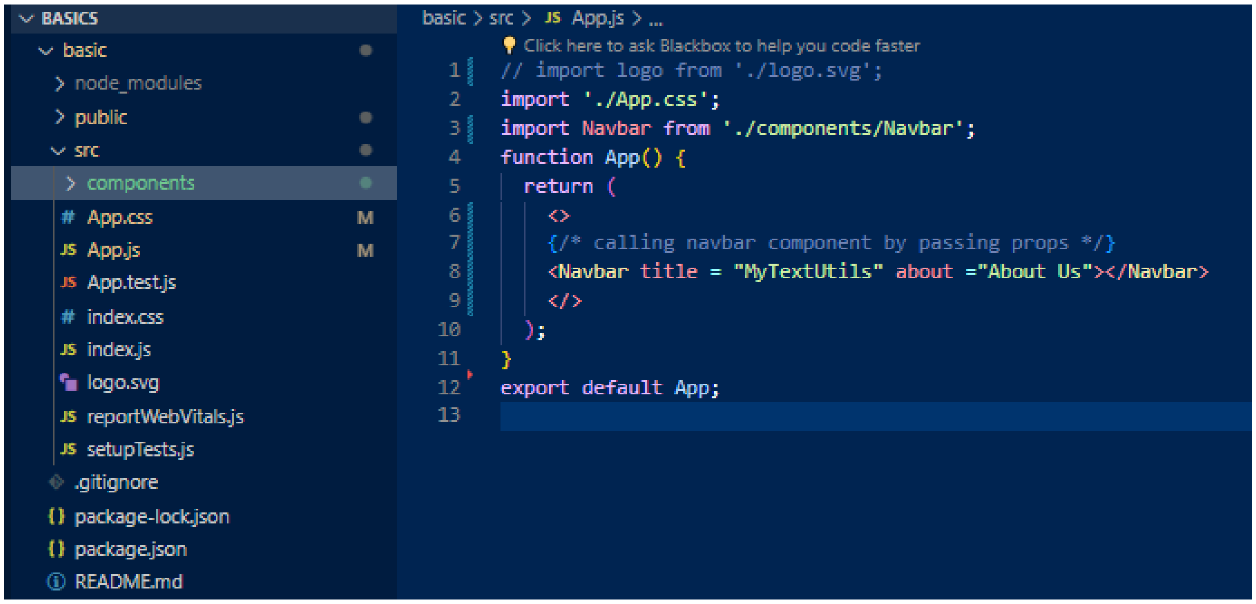
**Example:**

**components/Navbar.js**



```
import React from 'react';
import PropTypes from 'prop-types';
export default function Navbar(props) {
    return (
        <div>
            <nav className="navbar navbar-expand-lg bg-light">
                <div className="container-fluid">
                    {/* accessing props */}
                    <a className="navbar-brand" href="/">{props.title}</a>
                    <div className="collapse navbar-collapse" id="navbarSupportedContent">
                        <ul className="navbar-nav me-auto mb-2 mb-lg-0">
                            <li className="nav-item">
                                <a className="nav-link active" aria-current="page" href="/">Home</a>
                            </li>
                            <li className="nav-item">
                                <a className="nav-link" href="/">{props.about}</a>
                            </li>
                        </ul>
                    </div>
                </div>
            </nav>
        </div>
    )
}
// setting types of props
Navbar.propTypes = {
    title: PropTypes.string.isRequired,
    about: PropTypes.string.isRequired,
};
// providing default values for props
Navbar.defaultProps = {
    title: 'Title Here',
    about: 'About Here'
};
```

**App.js**



```
// import logo from './logo.svg';
import './App.css';
import Navbar from './components/Navbar';
function App() {
  return (
    <>
    {/* calling navbar component by passing props */}
    <Navbar title = "MyTextUtils" about ="About Us"></Navbar>
    </>
  );
}
export default App;
```

**Disclaimer**

- The following notes are under construction, once the notes are completed they will be updated over here: [React Notes]