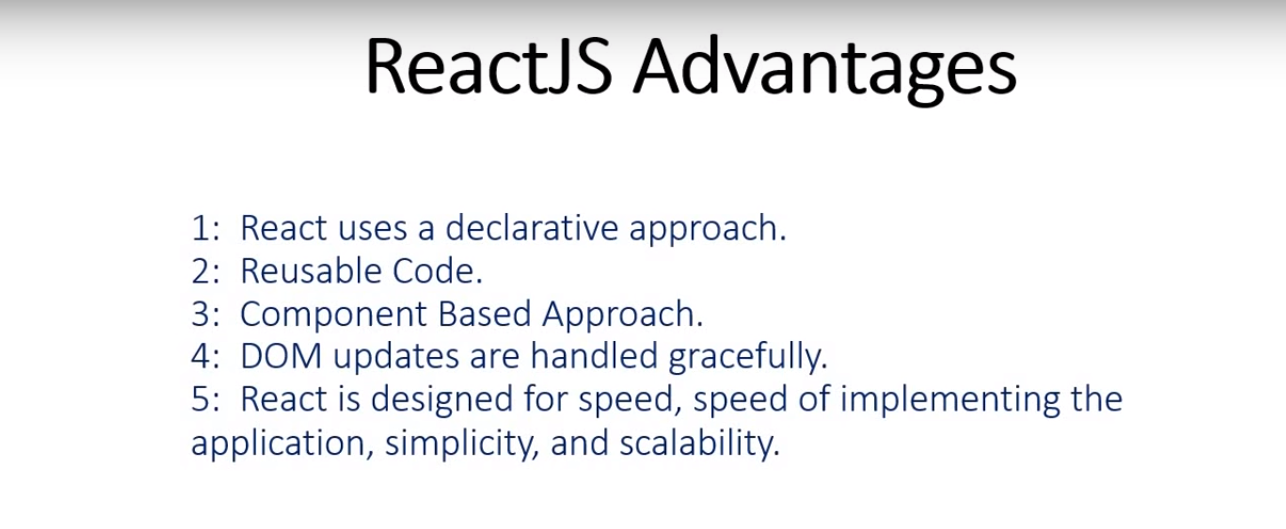
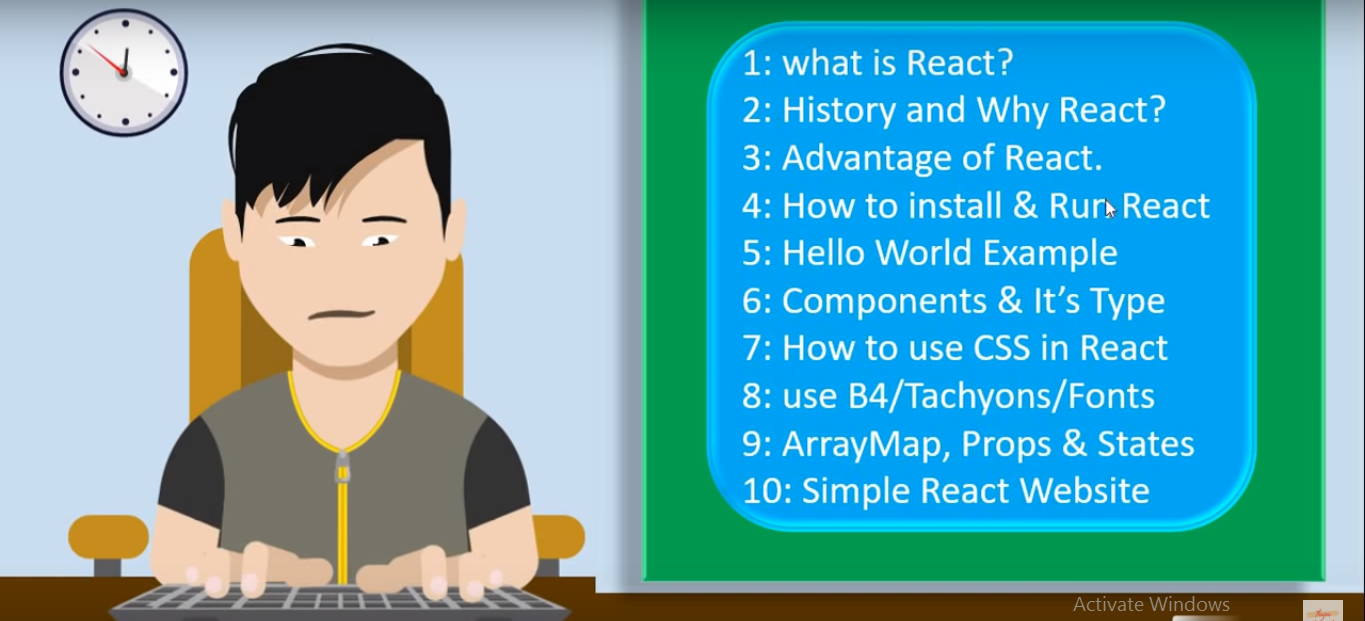
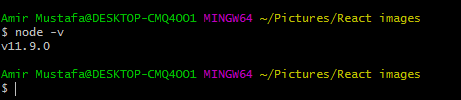
React JS



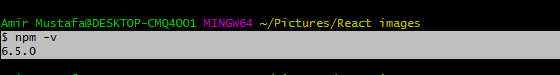


1. STEPS TO INSTALL REACT
2. Install Node

Go to node site and install node Recommended for most users



1. Check your npm version



1. Install React – Two ways:
2. Npm version > 5.2

npx create-react-app myapp

cd myapp

npm start

1. Npm < 5.2

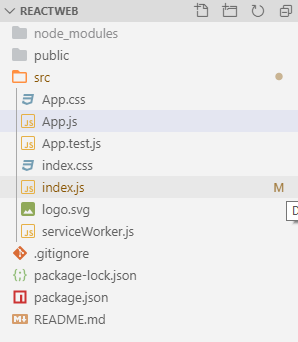
npm install –g create-react-app (create a directory and run in that)

create-react-app myapp

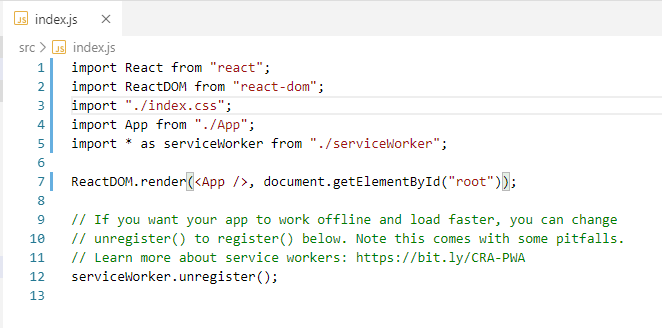
cd myapp

npm start

Directory Structure



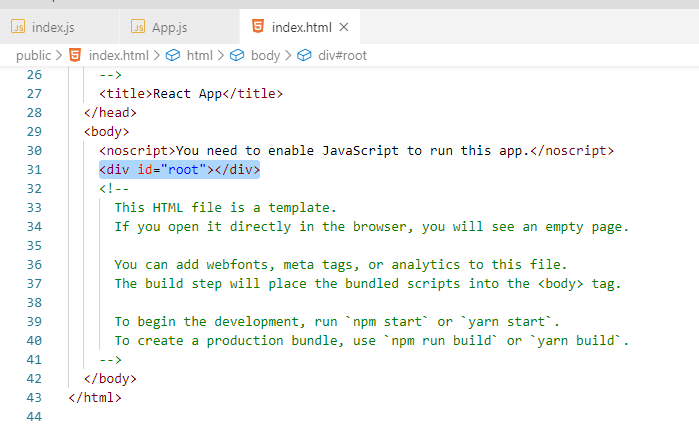
1. README.md – basic read me file
2. Package.json – basic dependencies and React version you can see here
3. Gitignore- files you need to ignore at the time of git push
4. Src – main folder (where we will code)
   1. Index.js – main parent js file



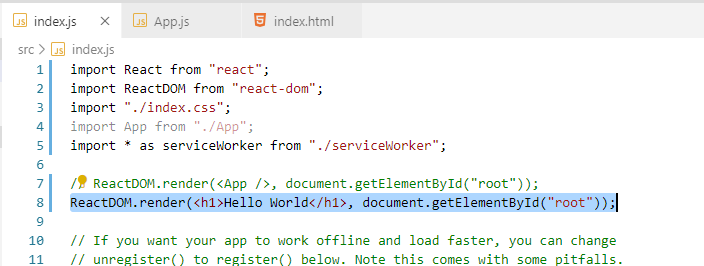
* 1. App.js – this page is where code is written react home screen logo is coming. This page is rendered from line 7 of index.js (above file)

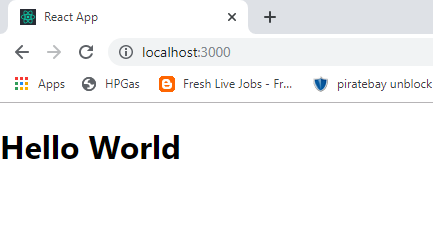


* 1. Index.html – In this page the location of div is given where page is rendered . eg. <div> with id root. Path- public/index.html



Print Hello World





Component

* This App line included is a component.
* Everything in a website is a component. Every div is a component eg. left sidebar, header, footer etc.
* Main use is reusability.

Create a New Custom Component

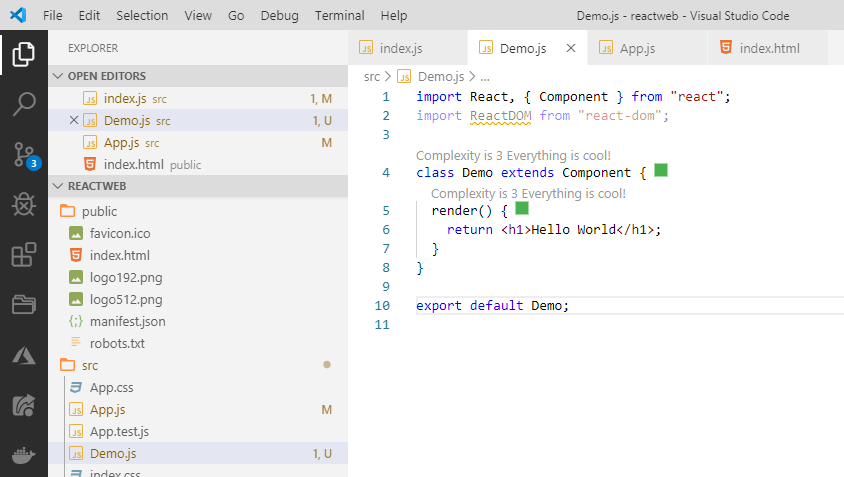
There are two types of components –

a. class (commonly used) – We will use this one

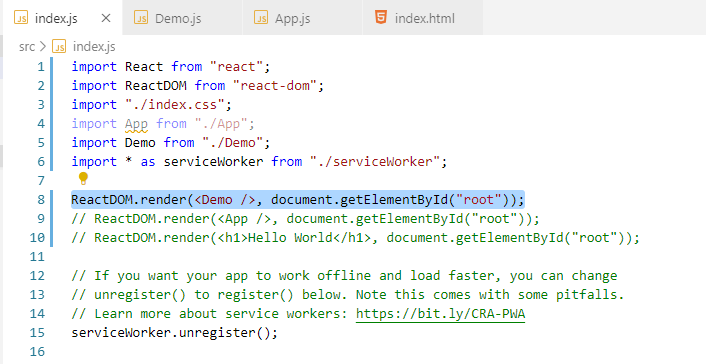
b. functional (now with hooks this is also useful) – We will see this after CSS topic below

Create a New Custom Component (Class Type)

1. Create a new file in src – Demo.js (First letter must be capital letter) – just write import React, ReactDOM, class of the component and export



In index.js – import the Demo component, write in Render <Demo, document.getElementById(‘root’)>



JSX – This is also called Javascript extension or JSXML

Whatever we write in the render function of the component is JSX

We can even write expression here:

<https://reactjs.org/docs/introducing-jsx.html>

You must write all the html component inside div or React.Fragment

Eg.

render() {

    return (

      <div> // creates one extra div

        <h1>Hello World</h1>

        <p>Welcome to React world</p>

      </div>

    );

  }

Or

render() {

    return (

      <React.Fragment> // does not create any extra div

        <h1>Hello World</h1>

        <p>Welcome to React world</p>

      </React.Fragment>

    );

  }

Props

This is short for properties.

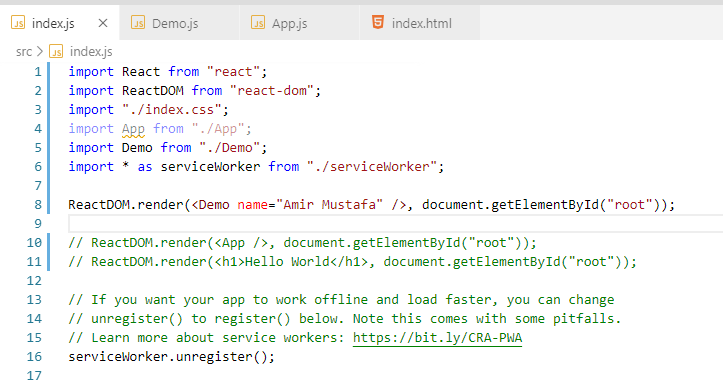
In HTML what is the property ( or attributes) of image tag - <img src=”” alt=””>

So src, alt is a property.Like wise in React we can pass property from parent to child

Parent – The page which is importing the Component (i.e. where import is written eg. src/index.js)

Child – The page which is exporting the component (where component code is written)

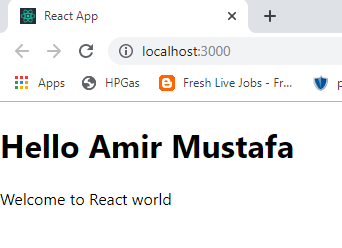
Index.js



Demo.js

We receive the props value in JSX by {this.props.<your\_propsname>}





So children (i.e. component) cannot change the value of props. Hence comes the concept of states

\_\_

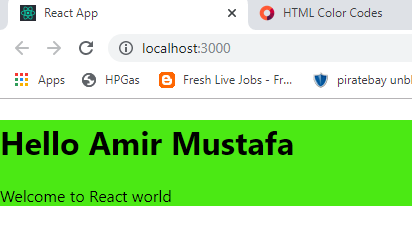
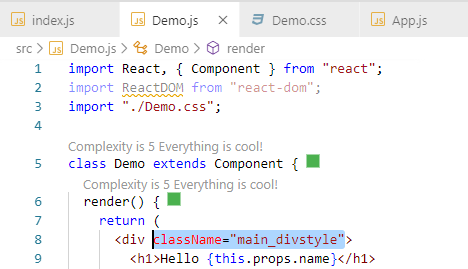
Applying CSS in JSX

1. Demo.css - Create a file eg Demo.css in src/Demo.css



1. Demo.js - In the div of the render file add the class. Import the above css file.

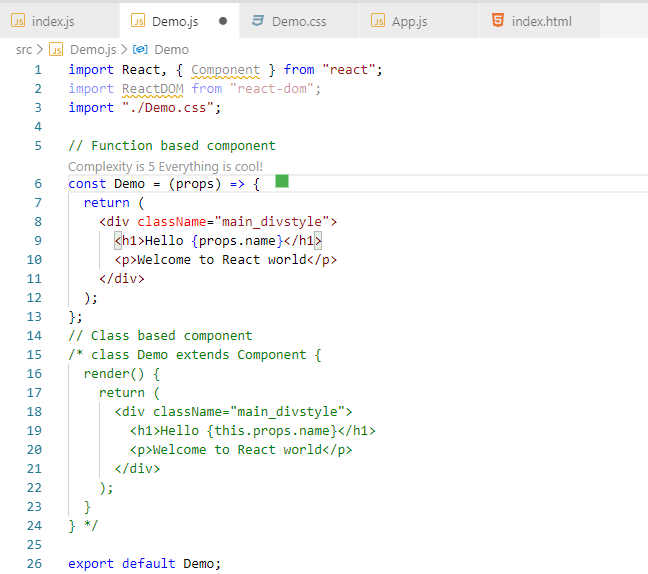
Note instead of class we use className in JSX as class is reserved keyword.



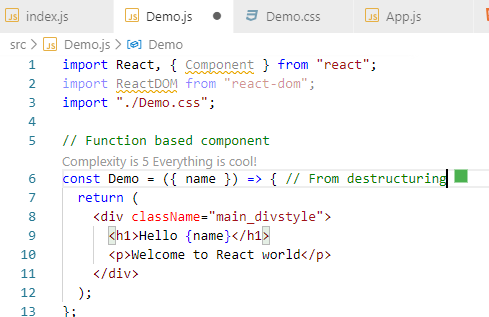
Create a New Custom Component (Functional Type)

We can create arrow type function to use create Demo component. What ever is written inside render **return must be copied .**

**Props** is passes as argument and used as **props.name** instead of **this.props.name**



TRICK 2 – With destructring we can directly write {name} in bracket as use it



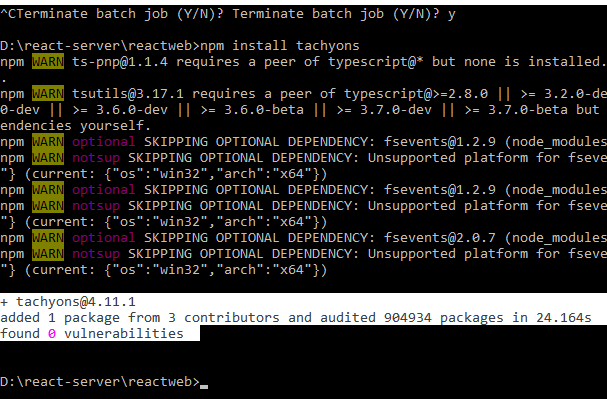
We will create a simple webite using Tachyons(like bootstrap)

TO install Tachyons

Npm install tachyons

Npm start

And in components type -> import ‘tachyons’;



Array map

To use loop in react arraymap is used

* Suppose we are printing 4 component with different attributes

<Avatarlist id="1" name="Amir Mustafa" works="Web Developer" />

      <Avatarlist id="2" name="Techsoft DK" works="Full stack" />

      <Avatarlist id="3" name="Priyanka Yadav" works="JS Developer" />

      <Avatarlist id="4" name="Atin Panday" works="Frontend Developer" />

So instead of using it we will take array of objects and loop with array map

const avatarlistarray = [

  {

    id: "1",

    name: "Amir Mustafa",

    works: "Web Developer"

  },

  {

    id: "2",

    name: "Tech Soft DK",

    works: "Python Developer"

  },

  {

    id: "3",

    name: "Priyanka Yadav",

    works: "JavaScript Developer"

  },

  {

    id: "4",

    name: "Atin Panday",

    works: "Frontend Developer"

  }

];

// Getting data in loop - with the help of array map

const avatardata = avatarlistarray.map((current, i) => {

  return (

    <Avatarlist id={current.id} name={current.name} works={current.works} /> // This we have written in loop instead of writing 4 times

  );

});

{ avatardata  } // prints the data 4 times

State

* Now data of parent (i.e. one having import) to child (i.e. having export - component) can be passed through props.
* Parent data cannot be changed from child. For this **state** concept came

TRICK:

State can be changed in:

1. Functional component – useState hook
2. Class component – this.state