1.ReactJS is a library

2.It is a single page application

3.nodejs installation

4.Editor vs code

5.Npx -Node package exicuter(it is alibrary (one package or one JavaScript file ).

We won't download temporary installation

1. npx create-react-app react-classes
2. Npm start
3. Don't give the app name "React" or reactDom
4. Root element
5. Import React from ‘react’;

Import ReactDOM from ‘react-dom/client’; //ReactDOM doesn't need curly braces because it is defaultly exported(Import {ReactDOM} from ‘react-dom/client’).

React.createElement(‘h1’,{}, "This is an h1 tag");

let root = React.createRoot(doocument.getElementById(‘root’));

root.render(h1);

Function test(){

<div><p>this is para</p></div>  
}

//we wont write like number line type of code we can use bable some libraryies

1. FolderStuture
2. node modules - more libraries

-React,ReactDOM, Babel, Webpack,budlers(mainly these or a few more libraries are enough, but reamining instlled dependies).

* 1. One library depends on another library, so for that remaining libraries also come in the node modules

Public - favicon(top of the page)

* 1. For the application running output comes from index.html (when the application is started using the command "npm start").
  2. manifest.json - on our website or web application for checking responsive(progressive web application).

For a responsive and complete application, the be resolution,theme color mentioned in the manifest.js

R C)robots.txt (For search results, we can access what files we can show below for users searc h (User Allow: Here, we have allowed to see user-seenpages , Disallowwe wont alow to show pages

1. src -- what ever we can written in src floder

App.css --styling

App.js- script file

App.test--

Index.js--suppose if we comment code it wont getoutput

Index.css--css

1. ReportWebVitals.js- Application performance checking
2. setupTest.js- for testing purpose
3. gitIgnore

/nodemodules- we wont push code (y nodemodules size is big)

1. package.json--libraries(npm install we use so gitignore can understand all dependencies, version everything it checks)
2. package-lock.json---18.2.0(18 is the major version, and 0--bugs).

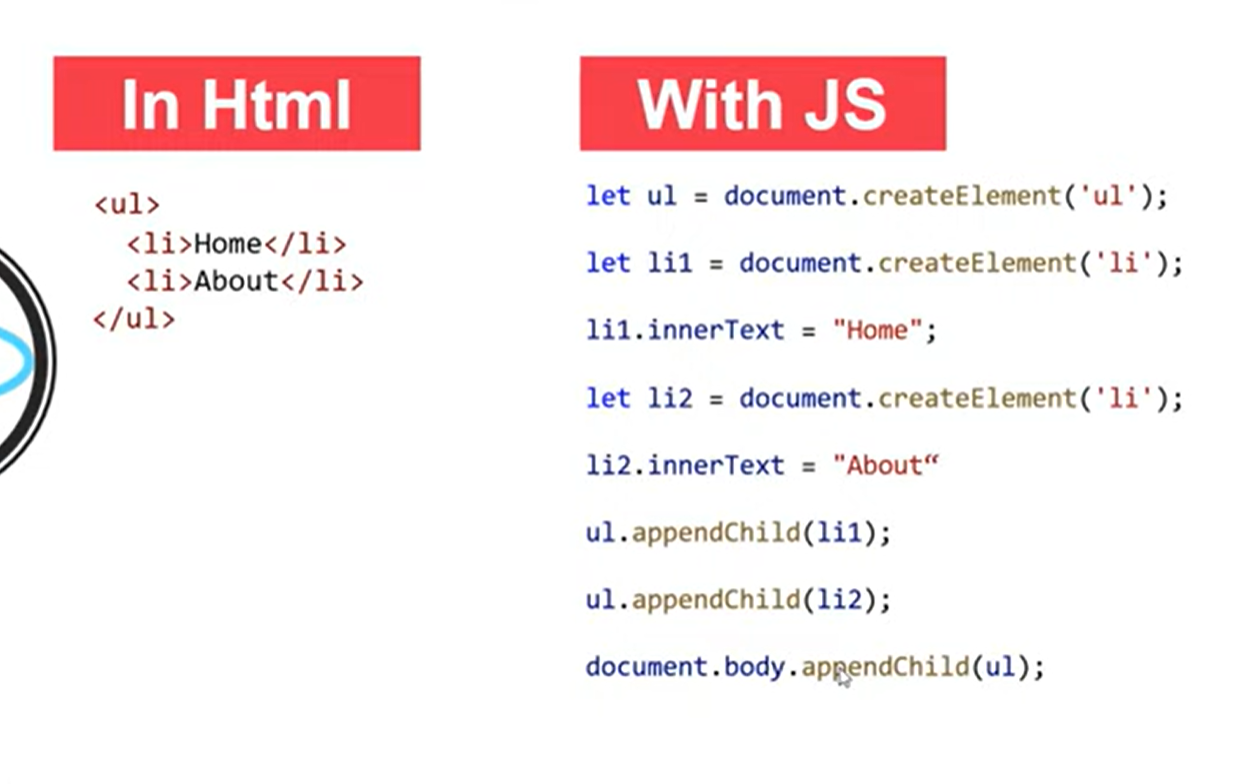
The version number **v18.7.3** follows a **semantic versioning** convention, which helps us understand the potential impact of updating to a new version. Let’s break down what each part of this version number signifies:

**1.Major Version (18)**: Major releases introduce significant new features. When updating to a new major release, developers might need to run update scripts, refactor code, perform additional tests, and learn new APIs. These changes require some effort during the update process.

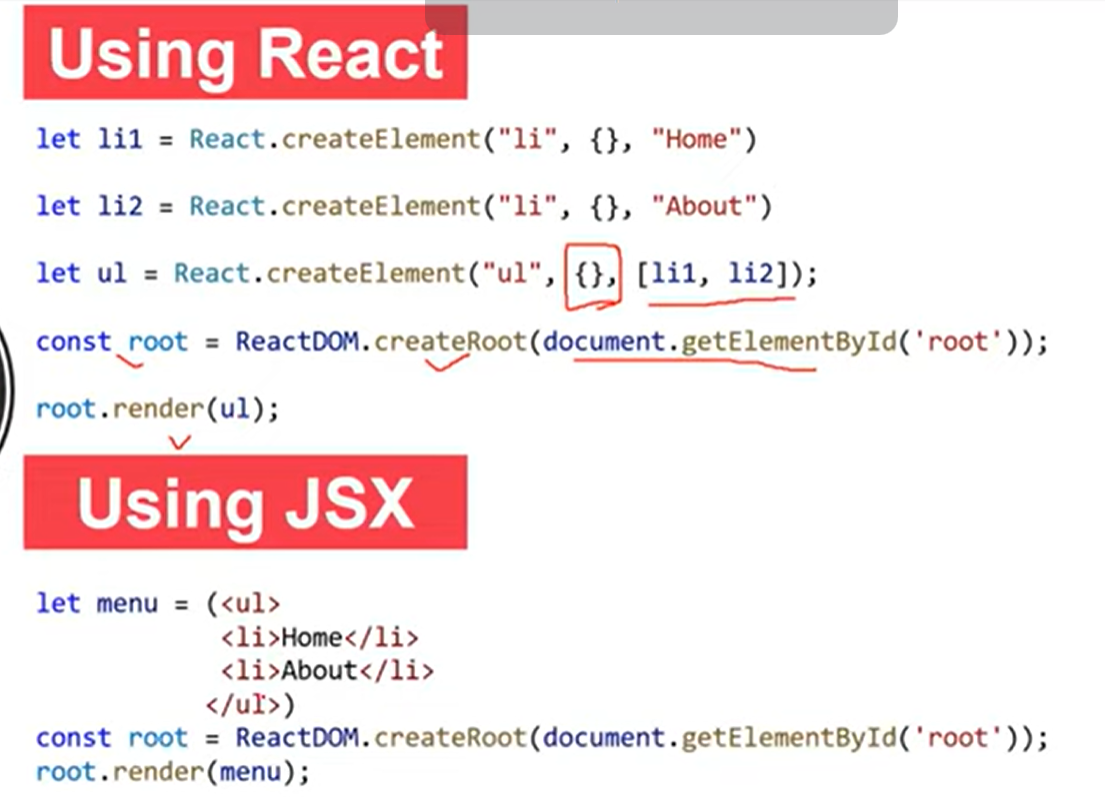
**Minor Version (7)**: Minor releases include smaller new features. They are fully backward-compatible, meaning no developer assistance is expected during the update. However, developers can optionally modify their applications and libraries to begin using new APIs, features, and capabilities added in the release.

1. In summary, **v18.7.3** represents a release with minor enhancements and bug fixes, ensuring that your applications stay up-to-date while maintaining stability. [If you’re using Angular, you might find a similar versioning scheme in their framework, where version numbers like 7.2.11 indicate major, minor, and patch levels](https://angular.io/guide/releases" \t "https://www.bing.com/_blank)

**Patch Level (3)**: Patch releases are low-risk bug fix releases. No developer assistance is expected during the update for patch versions.



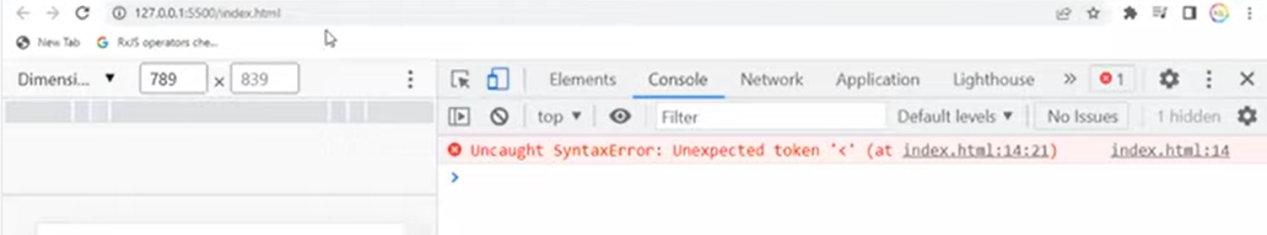
Plan reactjs



Javascript:



Output:



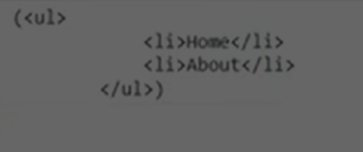
Components are 2 types

1. Functional components
2. Class components

Reactjs:



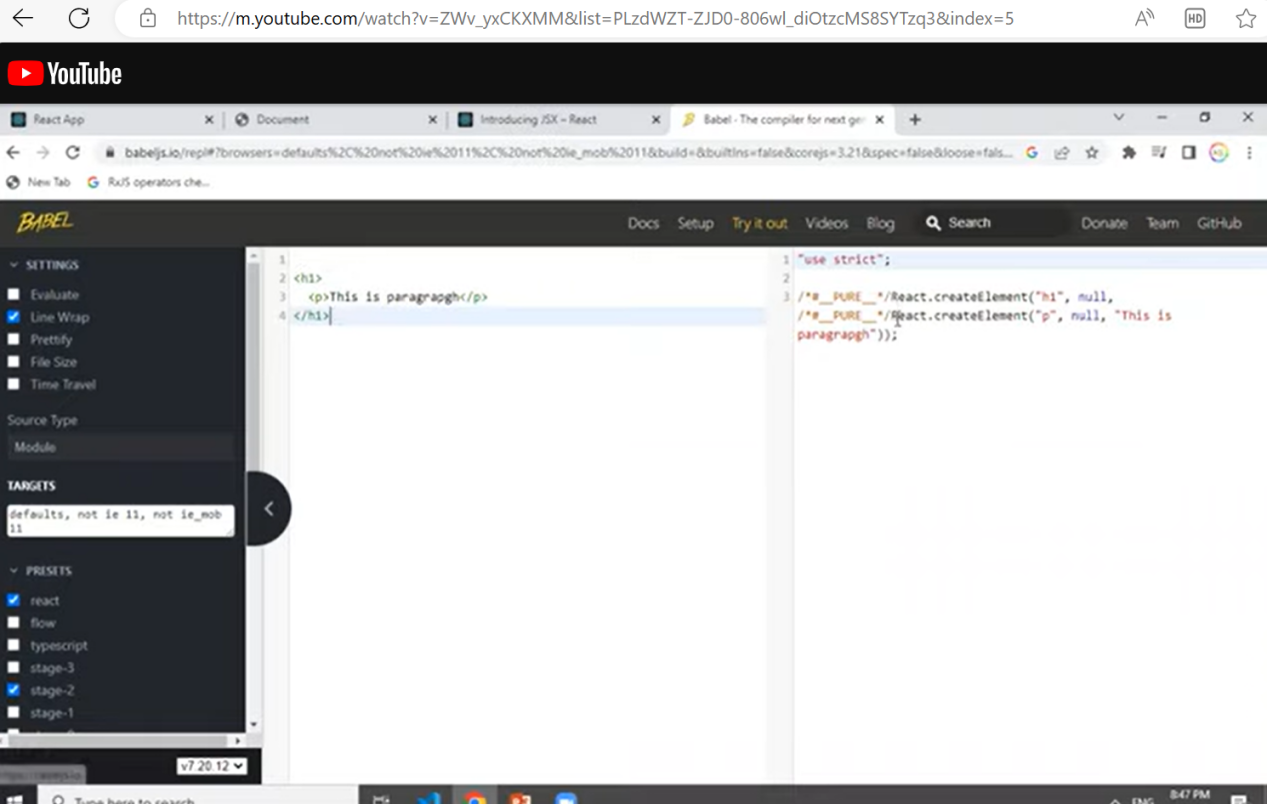
output:



[Babel · The compiler for next generation JavaScript (babeljs.io)](https://babeljs.io/repl" \l "?browsers=defaults, not ie 11, not ie_mob 11&build=&builtIns=false&corejs=3.21&spec=false&loose=false&code_lz=Q&debug=false&forceAllTransforms=false&modules=false&shippedProposals=false&circleciRepo=&evaluate=false&fileSize=false&timeTravel=false&sourceType=module&lineWrap=true&presets=env,react,stage-2&prettier=false&targets=&version=7.24.3&externalPlugins=&assumptions={})

**https://babeljs.io/repl#?browsers=defaults%2C%20not%20ie%2011%2C%20not%20ie\_mob%2011&build=&builtIns=false&corejs=3.21&spec=false&loose=false&code\_lz=Q&debug=false&forceAllTransforms=false&modules=false&shippedProposals=false&circleciRepo=&evaluate=false&fileSize=false&timeTravel=false&sourceType=module&lineWrap=true&presets=env%2Creact%2Cstage-2&prettier=false&targets=&version=7.24.3&externalPlugins=&assumptions=%7B%7D**

The above link converted normal HTML to react js(Babel is a library).



**JSX attributes |JSX Expressions**

----------------------

Export Let person = {

Name : “john”,

Age: 27,

Location:”AP”

}

Return(

<div>

<input className=”bgcolor” />

</div>

<div>

{name}

<input type=”checkbox” id=”sub”>

<label htmlFor=”sub” >Subscribe</label>

</div>

)

**Component styling in react|How to apply css in react**

Function App(){

Let s1 = {

Background:’green’,

Color:’black’,

marginTop:10px;//here normal style margin-top but in react we can use (-) removed added Captical letter

}  
return(

<>

<h1 style={s1}>This is heading</h1>

<p style={{

Background:’red’,

color:’white’,

marginTop:’10px’

}}>This is a paragraph.</p>

</>//freagments

)

**Another way to achieve global styling**

1.In the public folder, style.css

2.Path given in index.html

**Props in reactjs**

1.Props are nothing but properties

2. From the parent component to the child component calling is called props(declared in key value pairs

Functional component

-----------

App.js

----------

<User name=’John’ age=’26’ location:’hyderabad’/>

User.js

--------

Function User(props){

Return(

<>

<h1>Name:{props.name}</h1>

<h2>Name:{props.age}</h2>

<h3>Name:{props.location}</h3>

</>

)

}

ClassComponent

------------

App.js

----------

<User name=’John’ age=’26’ location:’hyderabad’/>

User.js

Class User extends React.component{

Constructor(){

Super();

}

Render(){

Return(

<>

<h1>Name:{this.props.name}</h1>

<h2>Name:{this.props.age}</h2>

<h3>Name:{this.props.location}</h3>

</>

)

}

ClassComponent

------------

Class User extends React.component{

Constructor(p){

Super();

}

Render(){

Return(

<>

<h1>Name:{this.p.name}</h1>

<h2>Name:{this.p.age}</h2>

<h3>Name:{this.p.location}</h3>

</>

)

}