NAMATE REACT EPISODE 1:

INCEPTION: PART 01

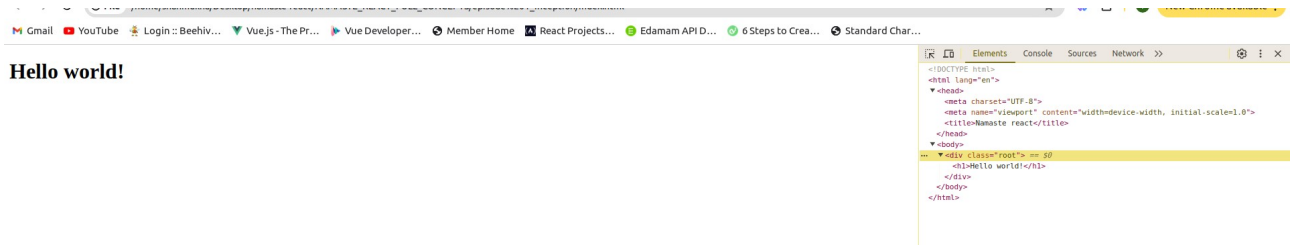
1.create namaste-react folder open in vs code and write basic html code of hello world.
2.html:5 -> command to produce html snippet.
Writing hello world program in html

index.html U

NAMASTE_REACT_FULL_CONCEPTS > episode 1_inception > index.html > html

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Namaste react</title>
7  </head>
8  <body>
9      <div class="root">
10         <h1>Hello world!</h1>
11     </div>
12 </body>
13 </html>
```

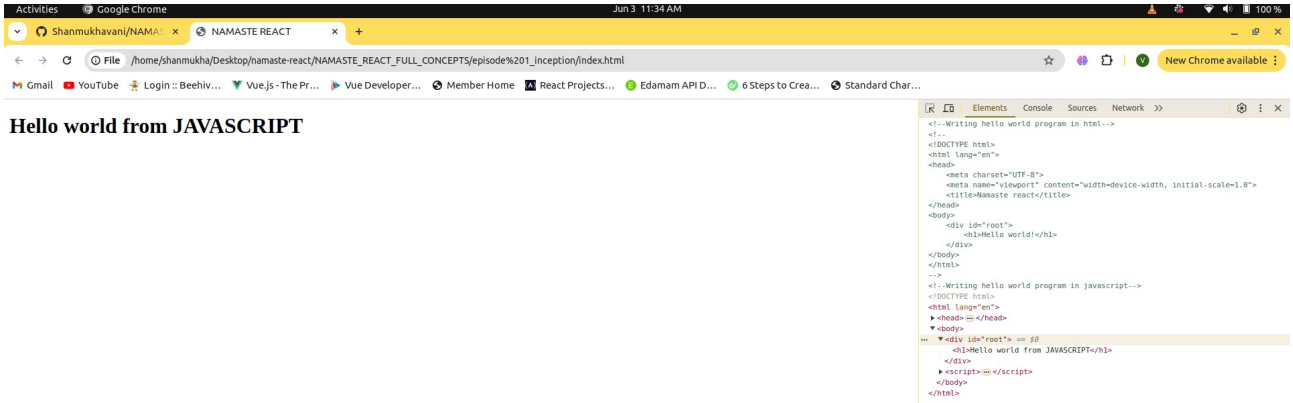
if you see in the console you will see the same code .



Writing hello world program in javaScript.

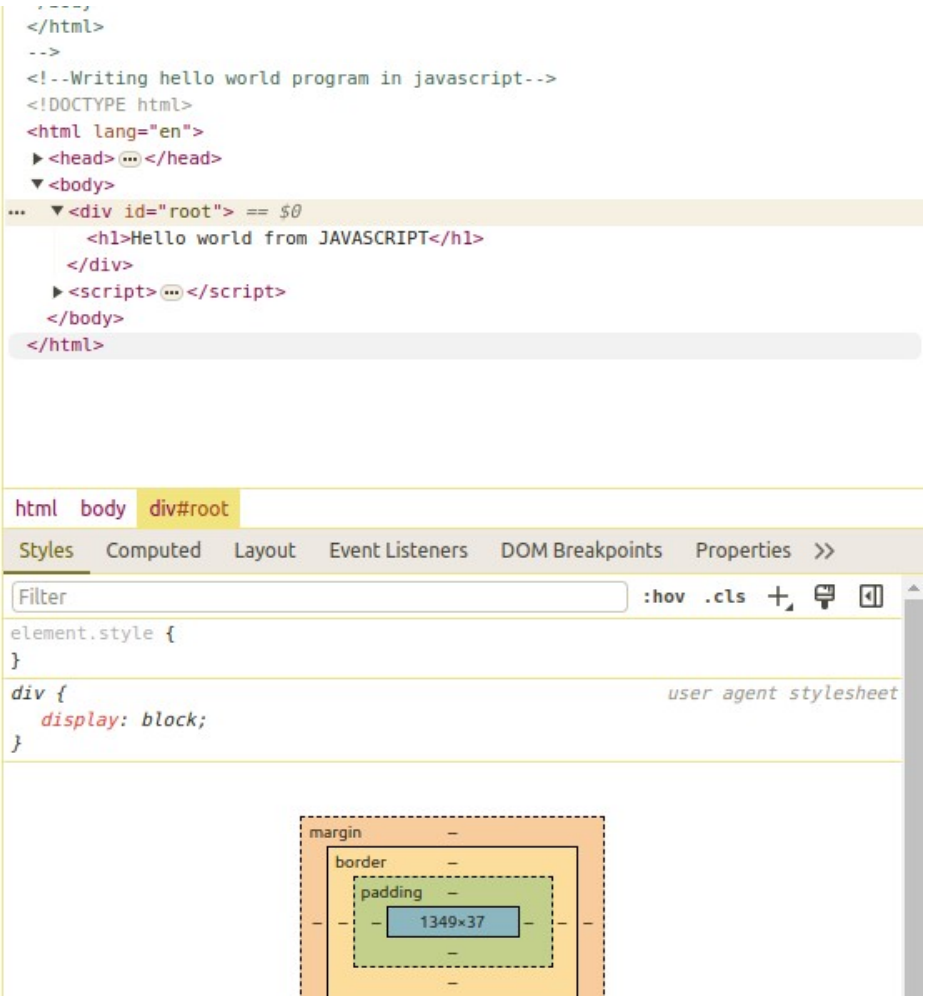
```
<!--Writing hello world program in javascript-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>NAMASTE REACT</title>
</head>
<body>
    <div id="root"></div>
    <script>
        const  heading = document.createElement("h1");
        heading.innerHTML = "Hello world from JAVASCRIPT";
        const root = document.getElementById("root");
        root.appendChild(heading);
    </script>
</body>
</html>
```

Output and console :



If you observe in the console you will notice the hello world from javascript is placed inside the root id.

INCEPTION: PART 02



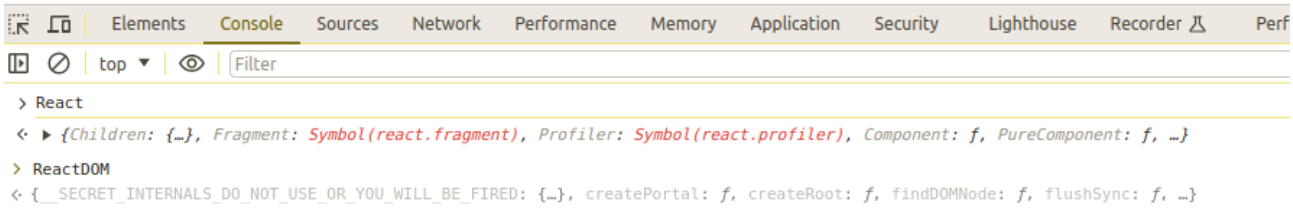
Now lets try to write hello world using react:
our browsers don’t know react they just know javascript.
So we need to bring react into our code by using the cdn links.
These cdn links hosts the react library.

Cdn links:

```
<script crossorigin
src="https://unpkg.com/react@18/umd/react.development.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@18/umd/react-
dom.development.js"></script>
```

this cdn links contain a plain javascript code to create react.
React is nothing but a javascript library.
After placing the links just go to console and type React you will see something like below.

Means the react is now available for us to use.
The first link consist of code for react code and second one is for dom.

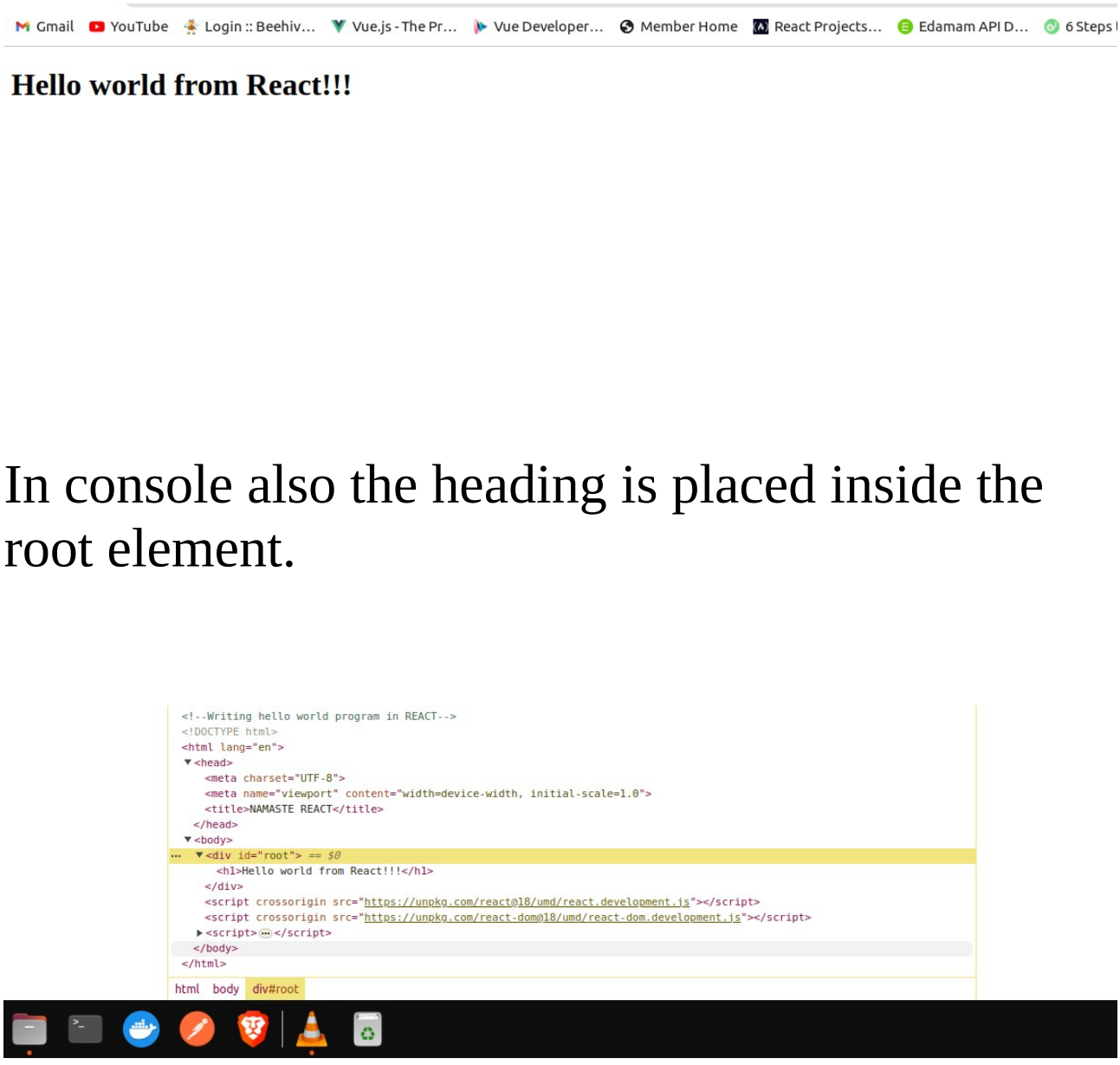


INCEPTION: PART 03:

syntax for `React.createElement` {element,{object},"text"}

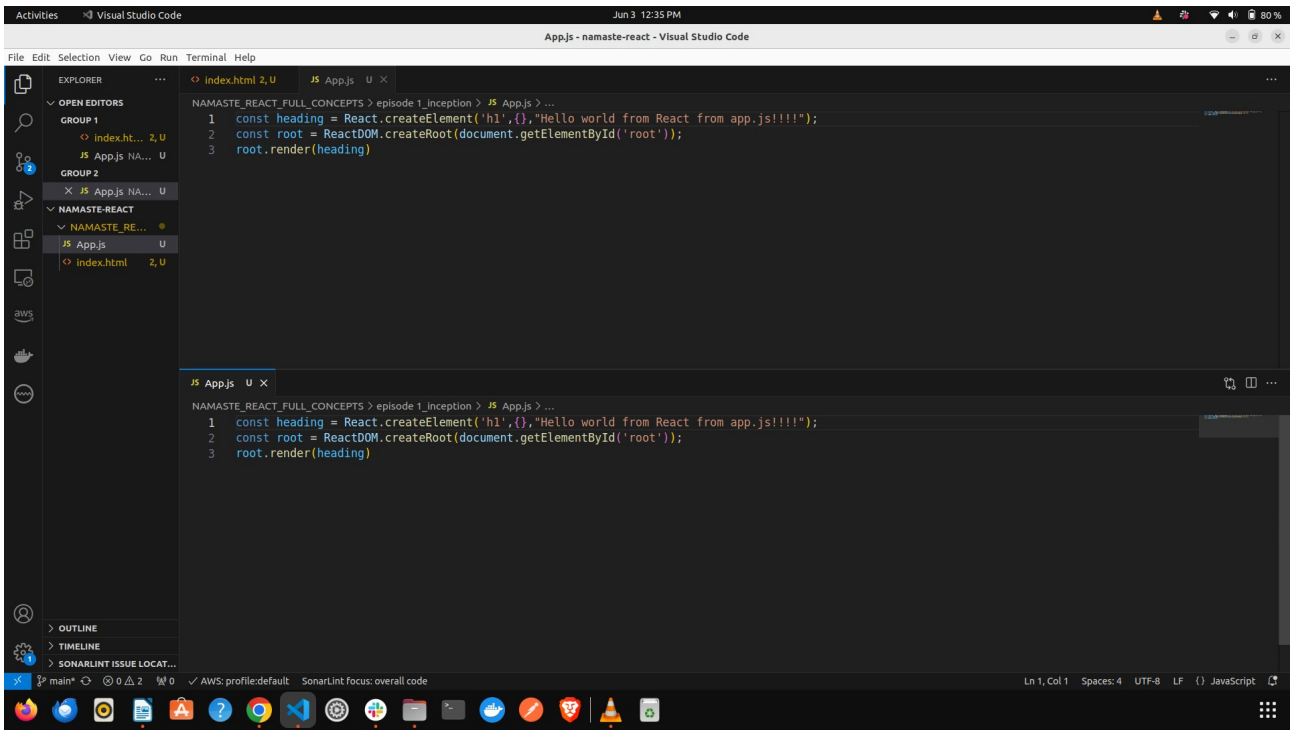
```
<!--Writing hello world program in REACT-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>NAMASTE REACT</title>
</head>
<body>
    <div id="root"></div>
    <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
    <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
    <script>
        const heading = React.createElement('h1',{},"Hello world from React!!!");
        const root = ReactDOM.createRoot(document.getElementById('root'));
        root.render(heading)
    </script>
</body>
</html>
```

output and console:

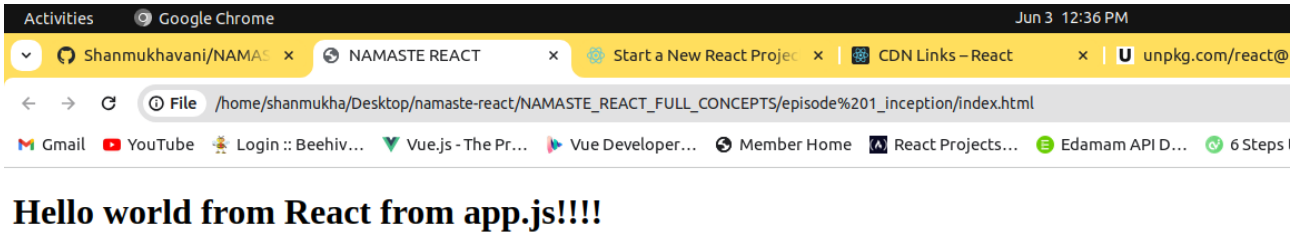


In console also the heading is placed inside the root element.

SEPARATING REACT INTO DIFF FILE(App.js)



output and console:



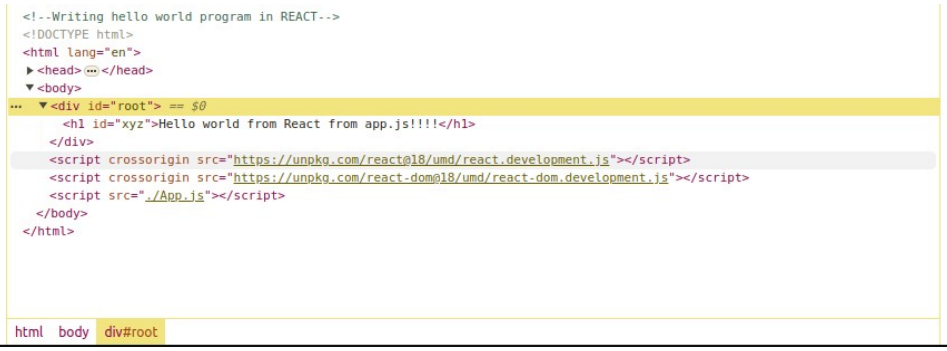
In ReactDOM create element:

syntax:

```
const heading = React.createElement('h1',{},"Hello world from React from app.js!!!!");
```

(tag,object,text)

The object contains the attributes of the tag like id etc.;



you can observe id xyz is added to h1.

Adding css

create index.css

App.js U

NAMASTE_REACT_FULL_CONCEPTS > episode 1_inception > App.js > heading

```
const heading = React.createElement(
    "h1",
    {id:"heading" , xyz: "abc"},
    "Hello world from React from app.js!!!!"
);
const root = ReactDOM.createRoot(document.getElementE
root.render(heading)
```

index.css U

NAMASTE_REACT_FULL_CONCEPTS > episode 1_inception > index.css > #heading

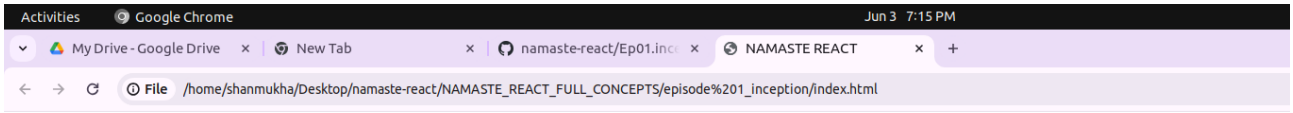
```
#heading{
    color : red;
}
```

index.html 2, U

NAMASTE_REACT_FULL_CONCEPTS > episode 1_inception > index.html > html > head > link

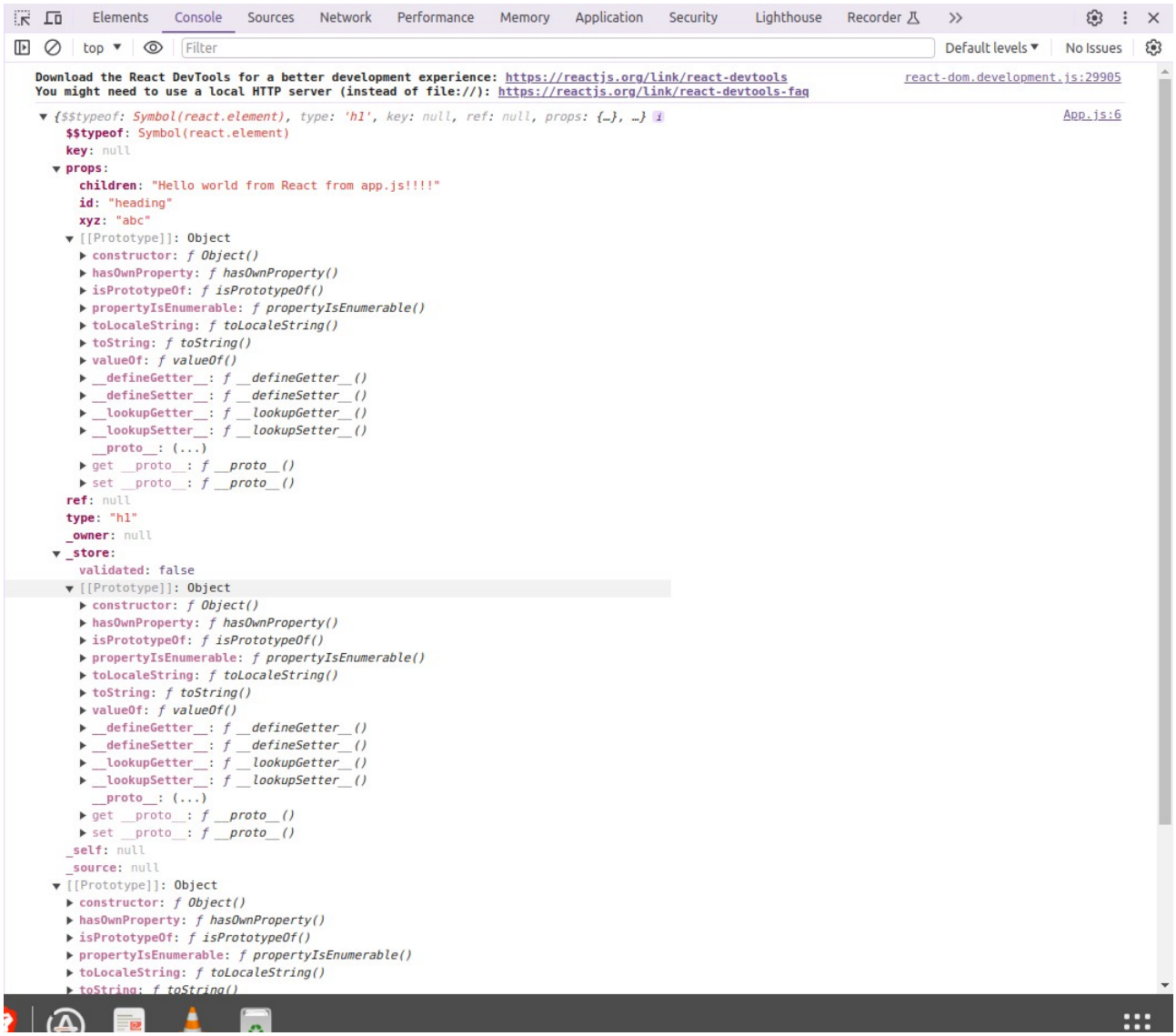
```
<!--Writing hello world program in REACT-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="./index.css" />
    <title>NAMASTE REACT</title>
</head>
<body>
    <div id="root"></div>
    <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
    <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
    <script src="./App.js"></script>
</body>
</html>
```

Then you can see output as below:



Hello world from React from app.js!!!!

if you try to console.log(heading)
the output will be as follows which is an object.
*Basically React.createElement creates an object.



INCEPTION: PART 04:

creating nested elements in react.

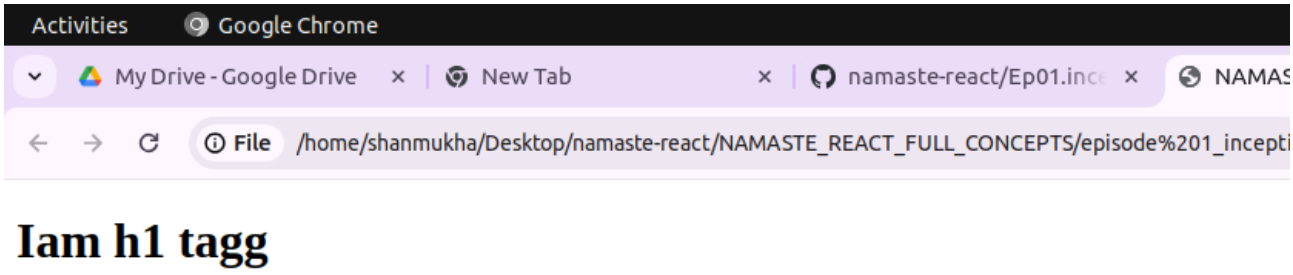
In html we write as follows:

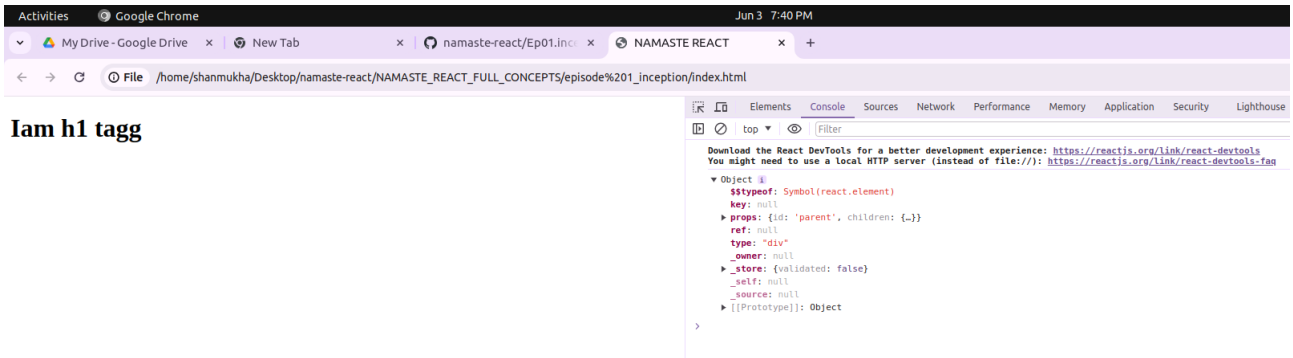
```html
<div id="root">
    <div id="parent">
        <div id="child">
            <h1>I am h1 tag</h1>
        </div>
    </div>
</div>
```

when it comes to react we will do as following

```
15
16  const parent = React.createElement(
17      "div",
18      {id : "parent"},
19      React.createElement(
20          "div",
21          {id : "child"},
22          React.createElement(
23              "h1",
24              {},
25              "Iam h1 tagg"
26          )
27      )
28
29  );
30  console.log(parent)
31
32  const root = ReactDOM.createRoot(document.getElementById('root'));
33  root.render (parent)
```

OUTPUT :





react(object) => HTML(browser understanding)

ReactElement is an object while rendering into the DOM it converts to HTML.

NESTED ELEMENTS WITH SIBLING ELEMENTS:

HTML:

REACT:

For that basically we do create array of children

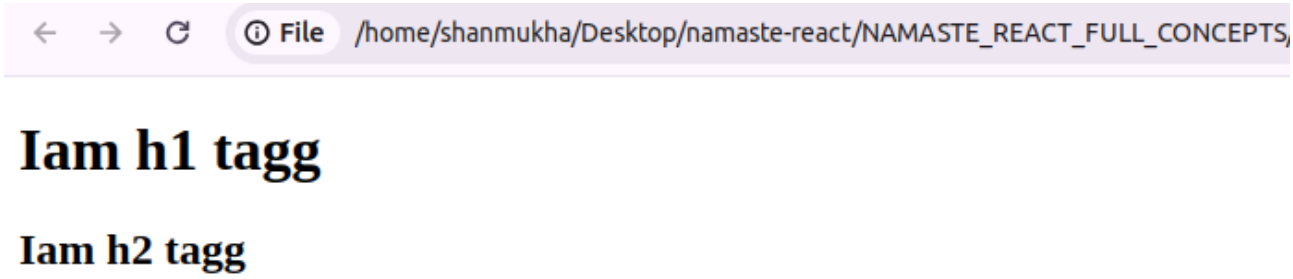
```
//CREATING NESTED ELEMENT WITH SIBLING TAGS USING REACT


const parent = React.createElement(
    "div",
    {id : "parent"},
    React.createElement(
        "div",
        {id : "child"},
       [React.createElement("h1",{},"Iam h1 tagg"),React.createElement("h2",{},"Iam h2 tagg")]
    )

);
console.log(parent)

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render (parent)
```

output:



console:

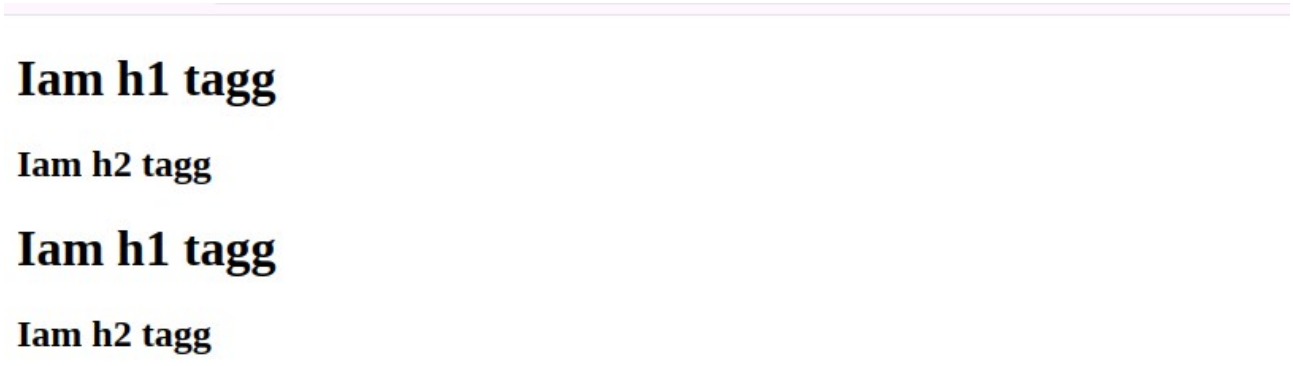
```
<!--Writing hello world program in REACT-->
<!DOCTYPE html>
<html lang="en">
▶ <head> ··· </head>
▼ <body>
  ▼ <div id="root">
    ▼ <div id="parent">
      ▼ <div id="child"> == $0
          <h1>Iam h1 tagg</h1>
          <h2>Iam h2 tagg</h2>
        </div>
      </div>
    </div>
    <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
    <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
    <script src="./App.js"></script>
  </body>
</html>
```

```
<div id="parent">
    <div id="child">
        <h1>I'm h1 tag</h1>
        <h2>I'm h2 tagg</h2>
    </div>
    <div id="child">
        <h1>I'm h1 tag</h1>
        <h2>I'm h2 tagg</h2>
    </div>
</div>
```

```
32
33  //CREATING NESTED ELEMENT WITH SIBLING TAGS USING REACT
34
35
36  const parent = React.createElement(
37      "div",
38      {id : "parent"},
39      [React.createElement(
40          "div",
41          {id : "child"},
42         [React.createElement("h1",{},"Iam h1 tagg"),React.createElement("h2",{},"Iam h2 tagg")]
43      ),
44      React.createElement(
45          "div",
46          {id : "child"},
47         [React.createElement("h1",{},"Iam h1 tagg"),React.createElement("h2",{},"Iam h2 tagg")]
48      )
49      ]
50
51  );
52  console.log(parent)
53
54  const root = ReactDOM.createRoot(document.getElementById('root'));
55  root.render (parent)
```

But this kind of code is complex so we use jsx to write react in HTML format only we will see it in next episode

OUTPUT:



CONSOLE:

INCEPTION: PART 05

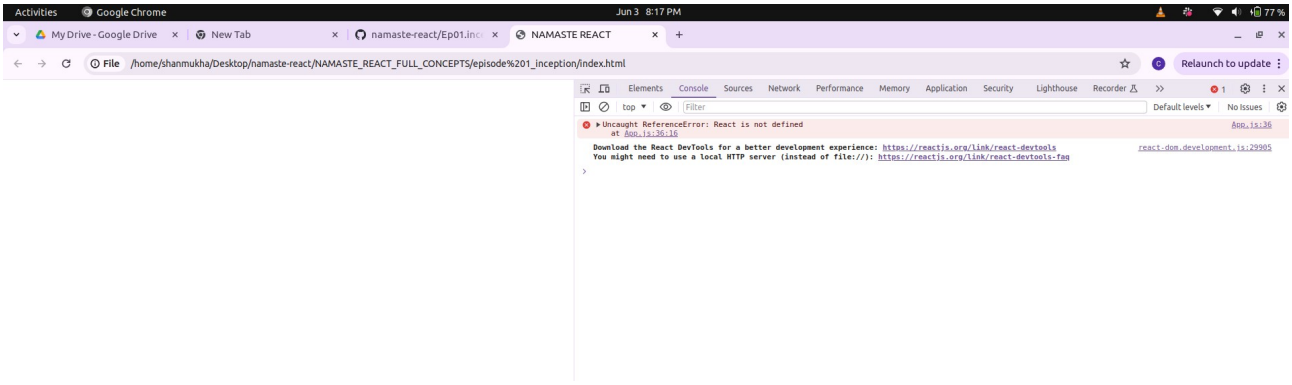
- Order in html code

- App.js should always be after react code otherwise it throws error.”REACT IS NOT DEFINED”

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="./index.css" />
    <title>NAMASTE REACT</title>
</head>
<body>
    <div id="root"></div>
    <script src="./App.js"></script>
    <script crossorigin src="https://unpkg.com/react@18/umd/react.development.js"></script>
    <script crossorigin src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></scrip

</body>
</html>
```

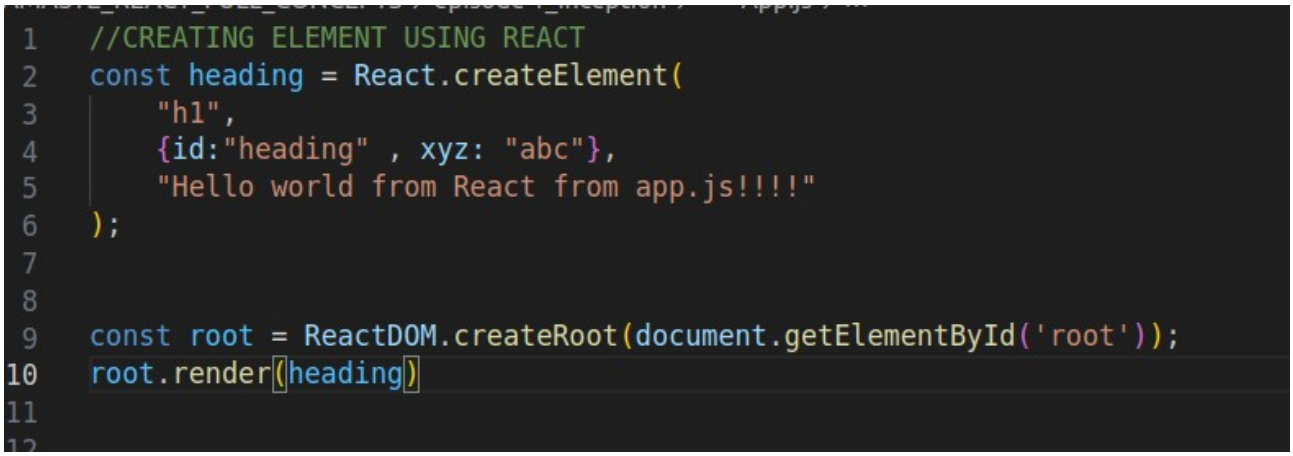
We can’t see any output.An error is thrown in console.



what is crossorigin in script tag of react?

The crossorigin attribute sets the mode of the request to an HTTP CORS Request. Web pages often make requests to load resources on other servers. Here is where CORS comes in. A cross-origin request is a request for a resource (e.g. style sheets, iframes, images, fonts, or scripts) from another domain

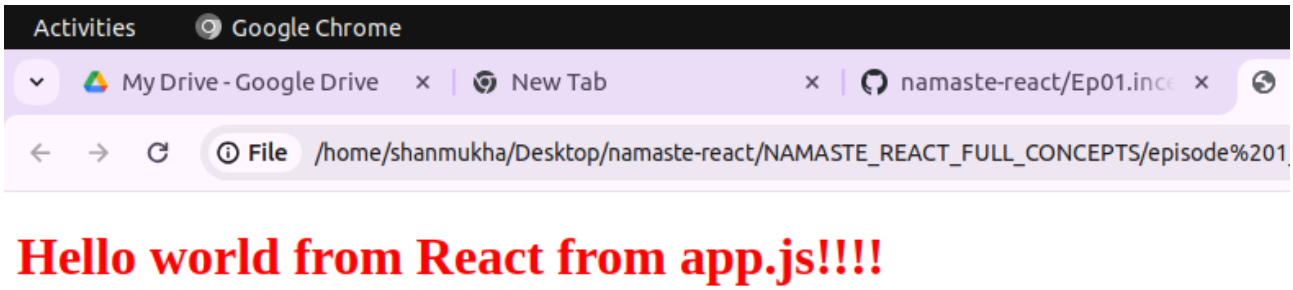
* **what happens if there is already a element in root?**



In html code HELLO WORLD is already present in root id.
From the react element we are trying to replace HELLO WORLD with Hello world from React App.js

The output is HELLO WORLD will be replaced with Hello world from React App.js
If you refresh page you can observe that.

OUTPUT:



Why this is happening?

Basically the code execution starts from html .Since the root has HELLO WORLD it will be rendered in browser.In later lines react is

added and react code it replacese evertthing from the root

HELLO WORLD to Hello world from React App.js.So it is appeared finally.

REACT is a library because it is a javascript code that can be used in any part of the code .In our case we used in root element.