React

Is React a framework or a library?

A framework is used for building and deploying an application quickly. When we use a framework, we can use resources to facilitate faster development and a greater user experience.

A library is a collection of reusable, compiled and tested code used to enhance the functionality of application.

React is referred to as unopinionated because there are lots of options to do something and we have to choose our tools carefully and create a proper design system for a specific project. That is why React is called a library, not a framework.

Writing a heading in Html vs JS:

Ex:

```
<html>
    <body>
        <h1> Hello Everyone </h1>
    </body>
</html>
```

Ex:

```
<body>
    <script>
const document.createElement('h1');
heading =
    heading.innerHTML = 'Hello Everyone';
const content = document.getElementById('root')
    content.appendChild(heading)
```

Html file

```
<div id = 'root'>
</div>
```

Writing a heading in React:

```
const heading = React.createElement("h1", {}, "Hello World");
```

↓

Here h1 is core HTML element, hence we are using 'React'

```
const root = ReactDOM.createRoot(document.getElementById("root"));
```

↓

Here we are manipulating the DOM by using ReactDOM and creating a root in the div by the id 'root'

```
root.render(heading);
```

Use of React:

In general, when we perform any DOM manipulation operation on the browser like clicking a button, we are indirectly making changes in the DOM tree. Usually these are costly operations.

So, frameworks like React try to optimize it.

↳

```
const heading = React.createElement("h1", {id: "heading"}, "Hello");
```

↓

This object is used for specifying attributes for tags.

↳ React.createElement creates a javascript object which is a React element (it is not an h1 tag here)

```
console.log(heading)
```

↳ This is that object above

↳ root.render()

↳ Basically takes the object and renders in the DOM.

Creating nested HTML Structure Inside React!

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

↳

```
const parent = React.createElement ("div", {id: "parent"},
                    React.createElement ("div", {id: "child"},
                    React.createElement ("h1", { }, "I am h1 tag")));
```

In order to create two elements at the same level, we need to combine the child elements in an array.

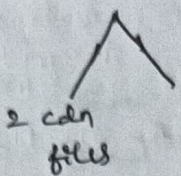
```
<h1> I am h1 tag </h1>
<h2> I am h2 tag </h2>
```

```
const parent = React.createElement ("div", {id: "parent"},
                    React.createElement ("div", {id: "child"},
                    [React.createElement ("h1", {}, "I am h1 tag"),
                    React.createElement ("h2", {}, "I am h2 tag")])
```

→ If there are more nested elements in HTML, then this React code becomes complicated to understand. Here there is a concept called JSX.

Note:

In the index.html file created before, we have placed 3 script tags.



Here the sequence of those files matters because, we cannot place App.js file before those cdn files as React is not specified in that situation.

So cdn files first, then only App.js.

↳ In the previous example, on this line

root.render(parent)

↓

This renders the parent object into the div section present in index.html

For now or initially, the div section is empty so whatever parent has, is rendered onto div.

However if there are any content already present in div before rendering. Then the initial content is completely replaced and the newly created parent object is rendered.

↳ It is very clear in these examples, that we can apply react to a small portion of code and it works fine. This makes React library instead of a full fledged framework.