React

Note: Take regular notes
Try out everything on laptop
Push progress everyday on GitHub.

Hello World

→ HTML

```
<html>
  ---
  --
<body>
  <h> Hello World </h1>
</body>
</html>
```

→ using JS

```
<script>
  const heading = document.createElement("h1");
  heading.innerHTML = "Hello World";
  const root = document.getElementById("root");
  root.appendChild(heading);
</script>
```

→ using React.

Note: document, document.createElement(), etc are browser APIs provided by browser to JS but browser doesn't know react.

So we can use CDN to include React library in our code.

<body>

<script crossorigin src="https.... react.dev..js"></script>
<script ".." ".."></script>
</body>
(CDN links)
scripts

This link has plain js which is nthg but react.

* 1st link → react → core react
2nd link → react DOM → react library useful for DOM operations
useful for modifying DOM.

* Why 2 links? Why not just 1 link?
React doesn't work only for browsers. There is React Native, React 3D etc.
So react DOM is like a bridge b/w react & browser DOM.

* To confirm the inclusion of react & react DOM in our code, try React and ReactDOM on the browser console after including the CDN in your code.

<script>
// creating element in react is bit diff from js
const heading = React.createElement("h1", {}, "Hello World");
element, properties, content

// create a root node. React needs a root to perform DOM operations
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(heading);

Note: creating a react element is part of core react.
So we used react.createElement().
But creating a root which needs to be rendered in the DOM, can be done only by using ReactDOM.
So, we used ReactDOM.createRoot().

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

↳ attributes

→ The react element which is returned to heading, is just a javascript object and not a HTML element.

basically,

```
type: "h1"
props: id: "heading"
       xyz: "abc"
       children: "Hello World"
```

this is a part of the js object which is also react element.

How to create a structure like this?

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

⇒

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

↳ array of children

Note: when rendering array of elements, react will give a warning to use a key for each element in array. We'll look into it later.

Note: You might have felt writing react code for the prev structure is quite complicated than writing plain HTML. Yes it is. complicated indeed. So we'll use JSX which will encounter this problem from later on.

Note: our react js file should always be defined after the react & react-dom cdn scripts.

Note: 'root.render()' always "replaces" content if there's already some content in the 'root' div of HTML. and "not append"

Note: HTML elements other than the root (outside root div) doesn't get affected by root.render(), they stay intact in the DOM.

=> So, some people often keep a test inside the root div.

```
<div id="root">
    <h1> Not rendered </h1>
</div>
```

So, if the h1 is displayed on the screen with "Not rendered" text, then it means the render() has some problem with it.

Because usually render() will replace the contents of the root div.

So, it is a good debugging practice.