🌳

Assignment 03 - Laying The Foundation

Owner	Pankaj Kumar

1: What is `JSX`?

- JSX is a syntax extension to JavaScript. It is a HTML-like syntax (but not HTML) in javascript.
- Facebook developers build JSX.
- JSX sanitizes your code.
- JSX converts HTML tags into react elements, it easier to write and add HTML in React.
- JSX allows us to write HTML elements in JavaScript and place them in the DOM without any createElement() and/or appendChild() methods.
- After compilation, JSX expressions become regular JavaScript function calls and evaluate to JavaScript objects.

Example 1 - WITH JSX:

```
const element = (
  <h1 className="greeting">
    Hello, world - With JSX!
  </h1>
);
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(element);
```

Example 2 - WITHOUT JSX:

```
const element = React.createElement(
  'h1',
  {className: 'greeting'},
  'Hello, world - Without JSX!'
);
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(element);
```

2: Superpowers of `JSX`.

- JSX allows us to write HTML-Like(not HTML) syntax in JavaScript.
- JSX uses ***React.createElement*** behind the scenes.
- JSX ⇒ React.createElement ⇒ javascript-Object ⇒ HTML(DOM).
- Using JSX, you can write markup inside Javascript, providing you with a superpower to write logic and markup of a component inside a single .jsx file. JSX is easy to maintain and debug.
- JSX is very secure & makes sure that the app is safe.
- JSX Prevents Injection Attacks : This is safe. {} —> acts like a sanitiser.

Example

```
const title = response.potentiallyMaliciousInput;
// JSX Prevents Injection Attacks : This is safe
const element = <h1>{title}</h1>;
```

Note: In this example, by default, React DOM escapes any values embedded in JSX before rendering them. Thus it ensures that you can never inject anything that's not explicitly written in your application. Everything is converted to a string before being rendered. This helps prevent XSS (cross-site-scripting) attacks.

3: Role of `type` attribute in script tag? What options can I use there?

The `type` attribute specifies the type of the script. The type attribute identifies the content between the `<script>` and `</script>` tags. The type attribute gives the language of the script or format of the data. It has a Default value which is “text/javascript”.

`type` attribute can be of the following types:

- `text/javascript` : It is the basic standard of writing javascript code inside the `<script>` tag.

  #### Syntax

  ```
  <script type="text/javascript"></script>
  ```

- `text/ecmascript` : this value indicates that the script is following the `EcmaScript` standards.
- `module`: This value tells the browser that the script is a module that can import or export other files or modules inside it.
- `text/babel` : This value indicates that the script is a babel type and required bable to transpile it.
- `text/typescript`: As the name suggest the script is written in `TypeScript`.

4: `{TitleComponent}` vs `{<TitleComponent/>}` vs `{<TitleComponent></TitleComponent>}` in `JSX`.

A: The Difference is stated below:

- `{TitleComponent}`: This value describes the `TitleComponent` as a javascript expression or a variable.
  The `{}` can embed a javascript expression or a variable inside it.
- `<TitleComponent/>` : This value represents a Component that is basically returning Some JSX value. In simple terms `TitleComponent` a function that is returning a JSX value.
  A component is written inside the `{< />}` expression.
- `<TitleComponent></TitleComponent>` : `<TitleComponent />` and `<TitleComponent></TitleComponent>` are equivalent only when `< TitleComponent />` has no child components. The opening and closing tags are created to include the child components.

Example

```
<TitleComponent>
    <FirstChildComponent />
    <SecondChildComponent />
    <ThirdChildComponent />
</TitleComponent>
```

5: What is tree shaking?

- Tree shaking is a term commonly used within a JavaScript context to describe the removal of dead code or unused code.
- Tree-shaking is an important way to reduce the size of your bundle and improve performance.
- It depends on the static syntax of import and export modules in ES6 (ES2015). By taking tree-shaking concepts into consideration when writing code, we can significantly scale down the bundle size by getting rid of unused JavaScript, thereby optimizing the application and increasing its performance.

6: What is React Element?

- React Element is finally an object.
- Elements are the smallest building blocks of React apps.
- An element describes what you want to see on the screen:

```
const element = <h1>Hello, world</h1>;
```

- Unlike browser DOM elements, React elements are plain objects, and are cheap to create. React DOM takes care of updating the DOM to match the React elements.