How does React work?

Table of Contents:

- Is React a Framework or a Library?
- How does React Renders Elements
- Reconciliation

Is React a Framework or a Library?

Although React is a **library**, it often **feels** like a framework because:

- **The ecosystem is Huge**: React is not just React. It has an ecosystem with tools like React Router (for navigation), Redux (for state management), and Next.js (for server-side rendering), which together provide a framework-like experience.
- **Opinionated Conventions**: While React is flexible, most projects follow best practices like using functional components, hooks, and JSX, which gives it a structured feel like a framework.
- **Meta-frameworks Like Next.js**: Tools like Next.js and Remix build on React and provide additional features like routing, data fetching, and SSR (server-side rendering), making React feel more like a framework.

React's official documentation defines it as:

> "A JavaScript library for building user interfaces."

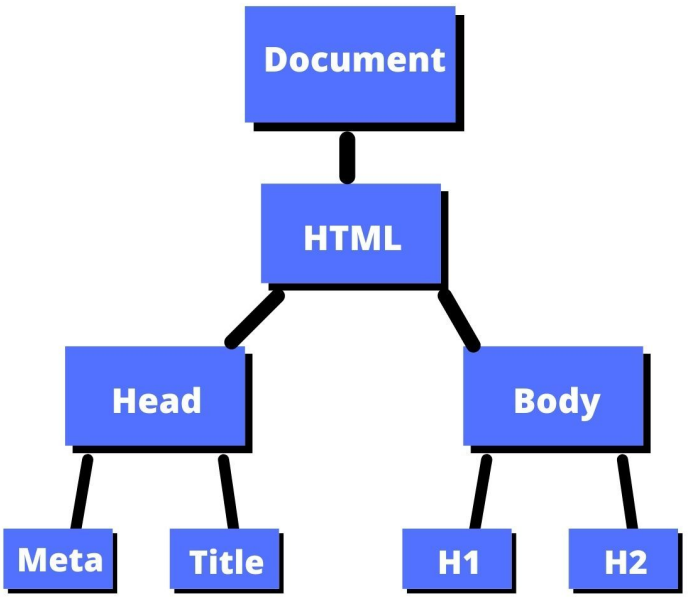
Here's why React is considered a **library** rather than a framework:

- **Focused on UI**
- **Lightweight & Flexible:** Unlike frameworks like Angular, React doesn't come with built-in solutions for everything. Developers can choose how to manage state (Redux, Context API, Recoil, etc.), how to route (React Router), and how to structure their project.
- **Integrates Easily with Other Libraries**

How does React Renders Elements

Here's how a typical Document Object Model (DOM) looks, Notice how it uses a tree of HTML tags. Manipulating DOM is an expensive operation because:

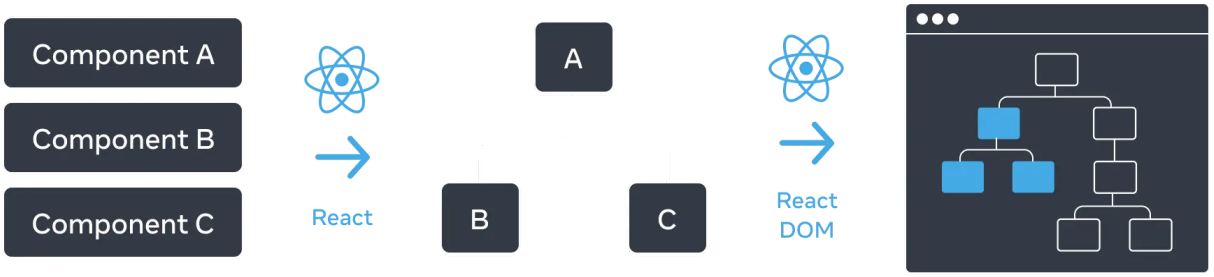
- The browser has to recalculate layouts, repaint, and re-render elements.
- Frequent updates make it **laggy and inefficient**.



In React, We call these Elements. Elements are the smallest building blocks of React apps.

```
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.



▼ **Render with root.render()**

```
const root = ReactDOM.createRoot(
  document.getElementById('root')
);
const element = <h1>Hello, world</h1>;
root.render(element);
```

Reconciliation

Reconciliation is the process React uses to efficiently update the UI when the **state** or **props** of a component change. Here's how it works:

- React creates a Virtual DOM representation of the UI - This is a lightweight copy of the actual DOM.
- When the state changes, a new Virtual DOM is created - This process is fast since Virtual DOM objects are simple JavaScript objects.
- React compares (diffs) the new Virtual DOM with the old one - This comparison uses a sophisticated **diffing algorithm.**
- Only the changed elements are updated in the real DOM - This selective updating is what makes React so efficient.