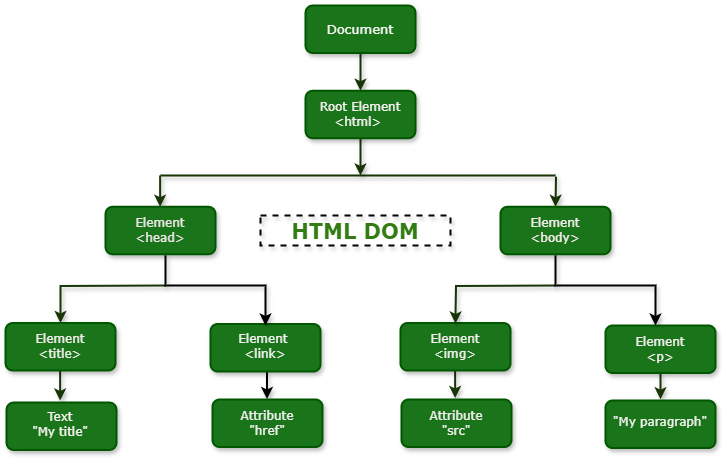
**Day 3 – React.js Learning**

**📌 1. DOM (Document Object Model)**

**English:**

* The DOM is a tree-like structure created by the browser to represent the entire HTML document.
* Each HTML element (like <div>, <h1>, <p>) is represented as a **node** in this tree.
* When we update the DOM directly, the browser has to re-render the entire UI or big sections, which is slow.

**தமிழ்:**

* DOM என்பது browser உருவாக்கும் ஒரு tree-போன்ற அமைப்பு. இது முழு HTML document-ஐ குறிக்கிறது.
* ஒவ்வொரு HTML element (<div>, <h1>, <p>) க்கும் ஒரு **node** ஆக tree-யில் இடம் உண்டு.
* DOM-ஐ நேரடியாக update பண்ணும்போது browser-க்கு பெரிய பகுதியை மறுபடியும் re-render செய்ய வேண்டியிருக்கும். அதனால் அது slow. 

**📌 2. Virtual DOM**

**English:**

* Virtual DOM is a **lightweight JavaScript representation** of the real DOM.
* Instead of updating the real DOM directly, React updates the Virtual DOM first.
* After comparing with the old Virtual DOM, React decides what minimal changes are needed in the real DOM.

**தமிழ்:**

* Virtual DOM என்பது **lightweight JavaScript பிரதிநிதி** (copy) of real DOM.
* React முதலில் Virtual DOM-ஐ update செய்யும், நேரடியாக real DOM-ஐ அல்ல.
* பழைய Virtual DOM-ஐ புதிய Virtual DOM-உடன் ஒப்பிட்டு, real DOM-ல் தேவையான changes மட்டும் React update செய்யும்.

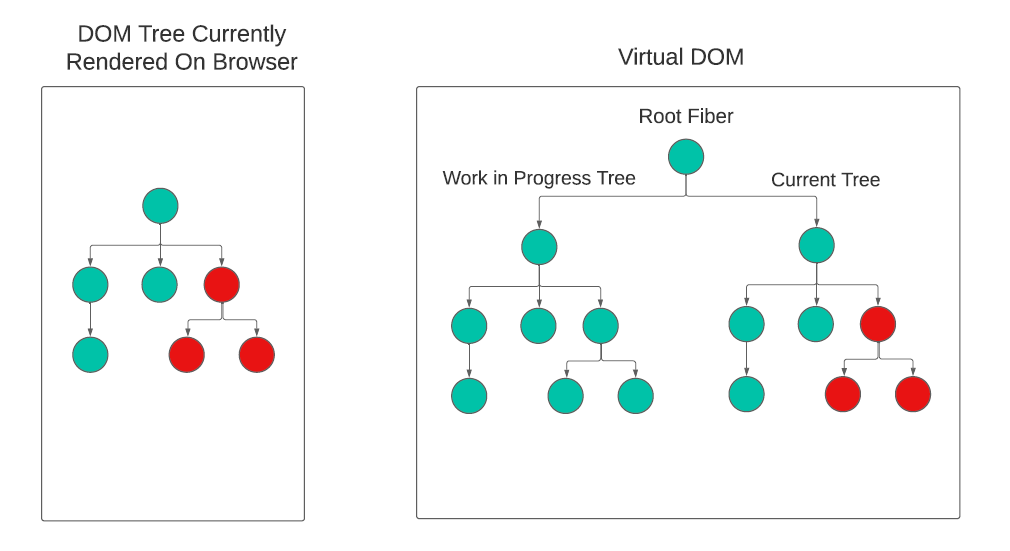
**📌 3. How Virtual DOM merges with Previous DOM**

**English:**

* Whenever state/props change, React creates a **new Virtual DOM**.
* React compares the new Virtual DOM with the previous one.
* Only the changed parts are updated in the actual DOM → this makes React very fast.

**தமிழ்:**

* State/props change ஆகும்போது React ஒரு **புதிய Virtual DOM** உருவாக்கும்.
* React புதிய Virtual DOM-ஐ பழைய Virtual DOM உடன் ஒப்பிடும்.
* வேறுபாடு உள்ள பகுதிகளை மட்டும் real DOM-ல் update செய்யும். இதனால் React வேகமாக செயல்படுகிறது.



**📌 4. Reconciliation**

**English:**

* Reconciliation is the **process** React uses to update the DOM efficiently.
* Steps:
  1. A new Virtual DOM is created after any change in state/props.
  2. React compares it with the old Virtual DOM.
  3. Finds the differences (using Diffing Algorithm).
  4. Updates only the changed parts in the real DOM.

**தமிழ்:**

* Reconciliation என்பது React real DOM-ஐ **efficient-ஆ update செய்யும் செயல்முறை**.
* படிகள்:
  1. State/props change ஆனவுடன் React புதிய Virtual DOM-ஐ உருவாக்கும்.
  2. அதை பழைய Virtual DOM-உடன் ஒப்பிடும்.
  3. Diffing Algorithm மூலம் வேறுபாடுகளை கண்டுபிடிக்கும்.
  4. அந்த மாற்றப்பட்ட பகுதிகளை மட்டும் real DOM-ல் update செய்யும்.

**📌 5. Diffing Algorithm**

**English:**

* The **Diffing Algorithm** is React’s method to find differences between old and new Virtual DOM.
* Main rules:
  1. If element types are different → Replace the entire node.
  2. If element types are the same → Update only changed attributes.
  3. For lists → React uses **keys** to track items and avoid re-rendering everything.

**தமிழ்:**

* **Diffing Algorithm** என்பது React-க்கு பழைய மற்றும் புதிய Virtual DOM-இடையிலான வேறுபாடுகளை கண்டுபிடிக்கும் வழிமுறை.
* முக்கிய விதிகள்:
  1. Element type வேறுபட்டால் → முழு node-ஐ மாற்றும்.
  2. Element type ஒரே மாதிரியாக இருந்தால் → attribute-ல் மாற்றம் உள்ள பகுதிகளை மட்டும் update செய்யும்.
  3. List-களில் → React **keys** பயன்படுத்தி items-ஐ track பண்ணும், முழு list-ஐ மீண்டும் re-render பண்ணாது.

