✨ LECURE 21 — CREATING & MODIFYING DOM ELEMENTS

🌱 FIRST-THOUGHT PRINCIPLE (DESI LOGIC)

👉 **HTML likhna** = ghar ka **naksha**
👉 **DOM** = ghar ke **asli blocks (objects)**
👉 **JavaScript** = wo banda jo ghar me
🧱 **naya kamra banaye,** 🛠️ **badle,** ❌ **tod de**

Is lecture me hum seekhenge:

- DOM me **naya element banana**
- **Multiple elements add karna**
- **Text / Attribute control**
- **Exact position par insert**
- **Replace & Remove**

🏗️ INITIAL DOM STATE (SAMJHO STARTING POINT)

```
<ul id="root">
  <li>CN</li>
  <li>HTML</li>
  <li>CSS</li>
  <li>JS</li>
</ul>
```

```
const parent = document.getElementById("root");
```

👉 Ab isi DOM ko **step by step modify** karenge.

📦 1. createElement() + appendChild()

📌 Kya hota hai?

- `createElement()` → element **memory me banta**
- `appendChild()` → parent ke **end me add**

🧪 Code

```
const li1 = document.createElement("li");

li1.innerHTML = "TS";

parent.appendChild(li1);
```

📤 DOM OUTPUT

```
CN

HTML

CSS

JS

TS
```

🧠 Yaad Rakho

> Jab tak append/prepend nahi karoge, element **DOM me dikhega hi nahi**

👨‍👩‍👧 2. append() — Multiple Elements Add Karna

📌 Difference

- `appendChild()` → sirf **1 node**
- `append()` → **multiple nodes + text**

🧪 Reusable Function

```
function attach(content){

    const element = document.createElement("li");
```

```
    element.innerHTML = content;

    const element2 = document.createElement("li");
    element2.innerHTML = content + " V2.0";

    parent.append(element, element2);
}

attach("React");
attach("Node");
```

📤 DOM OUTPUT

```
React
React V2.0
Node
Node V2.0
```

🧠 **Interview Tip**

append() modern & flexible hai

📝 3. createTextNode() — Sirf Text

📌 Kab use kare?

👉 Jab HTML tag nahi, **pure text** chahiye

🧪 Code

```
const textNode = document.createTextNode("Hello Coder Army");
parent.append(textNode);
```

📤 DOM OUTPUT

```
Node V2.0

Hello Coder Army
```

🏷️ 4. Attribute Nodes (Advanced but Important)

🧪 Old-school way

```
const attr = document.createAttribute("id");

attr.value = "first";


const firstLi = parent.children[0];

firstLi.setAttributeNode(attr);
```

📤 Output:

```
<li id="first">CN</li>
```

✅ Recommended Way

```
parent.setAttribute("data-custom", "20");

parent.removeAttribute("data-custom");
```

🧠 Rule

`setAttribute()` simple & clean hai

➕ 5. prepend() vs append()

🧪 Code

```
const angular = document.createElement("li");

angular.innerHTML = "Angular";
```

```
parent.prepend(angular);
```

📥 DOM OUTPUT

```
Angular

CN

HTML

CSS

JS

...
```

👉 `prepend()` = **start**
👉 `append()` = **end**

🔄 6. insertBefore() & replaceChild()

🔷 replaceChild()

```
const vue = document.createElement("li");

vue.innerHTML = "Vue";


const secondChild = parent.children[1];

parent.replaceChild(vue, secondChild);
```

📥 DOM OUTPUT

```
Angular

Vue

HTML

CSS

JS
```

🧠 **Use Case**

Jab existing element ko **replace** karna ho

⚡ 7. insertAdjacentElement / insertAdjacentHTML

📌 Super-Power Method

Exact jagah pe insert karta hai 🔥

```
const box = document.createElement("div");

box.innerHTML = "Hello Coder Army";
```

🧪 Positions

```
parent.insertAdjacentElement("beforebegin", box);

parent.insertAdjacentHTML("afterbegin", "<li>Vue-Start</li>");

parent.insertAdjacentHTML("beforeend", "<li>Svelte-End</li>");

parent.insertAdjacentElement("afterend", box.cloneNode(true));
```

📤 Positions Meaning

- **beforebegin** → ul ke bahar upar
- **afterbegin** → ul ke andar start
- **beforeend** → ul ke andar end
- **afterend** → ul ke bahar neeche

🗑️ 8. remove() & removeChild()

🔷 remove()

```
const elementToRemove = parent.querySelector("li");

elementToRemove.remove();
```

🔷 removeChild()

```
const child = parent.children[0];

parent.removeChild(child);
```

📌 Difference:

- `remove()` → khud delete
- `removeChild()` → parent delete karta

🧠 FINAL DOM CHEAT-SHEET

Method	Kaam
createElement	Naya element
createTextNode	Text only
append / prepend	Add
insertBefore	Specific jagah
replaceChild	Replace
insertAdjacentHTML	Fast insert
remove / removeChild	Delete
set/removeAttribute	Attributes

🔁 SHORT SUMMARY — LECTURE 21 (DOM)

- **DOM dynamic hota hai** → JS se elements **create, add, replace, delete** kar sakte ho
- **createElement()** → naya element memory me
- **createTextNode()** → sirf text (no HTML)
- **append / appendChild()** → end me add
- **prepend()** → start me add
- **insertBefore / insertAdjacent*** → exact position control
- **replaceChild()** → purana element replace
- **remove() / removeChild()** → element delete
- **get / set / removeAttribute()** → attributes control
- **innerHTML** → fast but large DOM me avoid

🧠 **Golden Rule:**
👉 *Small changes* → `createElement + append`
👉 *Performance matters* → `innerHTML` kam use karo

🎯 INTERVIEW RULE OF THUMB

✅ Small DOM → `createElement + append`
❌ Large DOM → `innerHTML` avoid
🔥 Dynamic UI → `insertAdjacentHTML`
💯 Clean code → `setAttribute`