🌟 JAVASCRIPT COMPLETE NOTES (FUNCTIONS, ARRAYS, OBJECTS, DOM & VIRTUAL DOM)

1. JavaScript Functions

✓ What is a Function?

A function is a **block of code that performs a specific task**. You write it once and reuse it anywhere.

Example:

```
function greet() {
  console.log("Hello!");
}

greet(); // calling the function
```

✓ Function with Parameters

```
function add(a, b) {
  return a + b;
}

console.log(add(5, 10)); // 15
```

✓ Arrow Function (Modern JS)

Arrow functions are shorter:

```
const multiply = (a, b) => a * b;
console.log(multiply(3, 4));
```

✓ Why Functions Are Important?

- Reduce code repetition
- Organize logic
- Easy debugging
- Used everywhere in React

---

2. JavaScript Arrays

✓ What is an Array?

Array = collection of multiple values stored in one variable.

```
const fruits = ["Apple", "Banana", "Mango"];
```

Access value:

```
console.log(fruits[0]); // Apple
```

Important Array Functions (FULL EXPLANATION)

2.1 forEach()

Used to run a function for **each item** in the array.

Example:

```
fruits.forEach(item => console.log(item));
```

2.2 map()

Creates a **new array** by modifying each item.

Example:

```
const numbers = [1, 2, 3];
const doubled = numbers.map(n => n * 2);

console.log(doubled); // [2, 4, 6]
```

✓ Commonly used in React to display lists.

2.3 filter()

Used to **keep items that match a condition**.

Example:

```
const numbers = [1, 2, 3, 4];
const even = numbers.filter(n => n % 2 === 0);

console.log(even); // [2, 4]
```

✓ Used for searching or filtering data.

2.4 reduce()

Used to convert an array into **a single value**.

Example:

```
const total = [10, 20, 30].reduce((sum, n) => sum + n, 0);

console.log(total); // 60
```

✓ Used in React for totals (Expense Tracker).

2.5 find()

Returns the **first item** that matches a condition.

```
const users = [{id:1}, {id:2}];
const user = users.find(u => u.id === 2);

console.log(user); // {id:2}
```

2.6 some()

Checks if **at least one item** matches.

```
[1,2,3].some(n => n > 2); // true
```

2.7 every()

Checks if **all items** match.

```
[2,4,6].every(n => n % 2 === 0); // true
```

2.8 sort()

Sorts array (alphabet or numbers).

```
["c","a","b"].sort(); // ["a","b","c"]
```

Sorting numbers:

```
[10,2,30].sort((a,b)=>a-b); // [2,10,30]
```

---

3. JavaScript Objects

✓ What is an Object?

An object stores **data in key–value pairs**.

Example:

```
const student = {
  name: "Munees",
  age: 22,
  course: "BCA"
};
```

Access data:

```
console.log(student.name);
```

Update:

```
student.age = 23;
```

Add new value:

```
student.city = "Chennai";
```

Use in arrays:

```
const users = [
  { name: "A", age: 20 },
  { name: "B", age: 25 }
];
```

Objects + arrays = used everywhere in React.

---

4. DOM (Document Object Model)

What is the DOM?

DOM = **tree structure** created by the browser from your HTML. It allows JavaScript to **change the webpage**.

Example:

```
<p id="text">Hello</p>
```

JavaScript:

```
document.getElementById("text").innerText = "Welcome!";
```

✓ How DOM Works?

1. Browser reads HTML
2. Converts it into a **tree of nodes**

3. Each tag becomes an object
4. JavaScript can update these nodes

---

5. Virtual DOM (React)

✓ What is Virtual DOM (VDOM)?

Virtual DOM is a **lightweight copy** of the real DOM created by React.

React:

- Creates Virtual DOM
- Compares old vs new (Diffing)
- Updates **only the changed parts**
- Makes UI very fast

---

6. DOM vs Virtual DOM (Full Difference)

Feature	DOM	Virtual DOM
Type	Actual browser DOM	In-memory JS representation
Speed	Slow for many updates	Very fast
Updating	Directly updates HTML	Updates only differences
Rendering	Entire UI may re-render	Only changed components
Used in	Vanilla JS	React

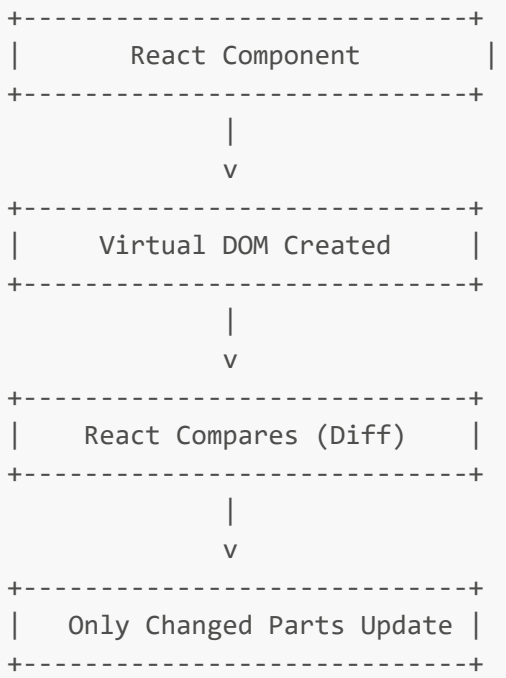
---

7. Diagram — DOM vs Virtual DOM

DOM Flow (Traditional JavaScript)

```
+-----------------------------+
              |
              v
+-----------------------------+
|   Whole Page Re-renders     |
+-----------------------------+
```

Virtual DOM Flow (React)



---

8. Why React Uses Virtual DOM

✓ Faster updates ✓ Better performance ✓ Efficient rendering ✓ Smooth user experience ✓ Handles huge apps easily

---