✨ LECTURE 10 — OBJECTS IN JAVASCRIPT (PART-1)

📘 **JavaScript Core — Objects (Foundation of JS)**
🧠 *First Principles • Real-World Mapping • Clean Mental Model*

⬛ FIRST PRINCIPLE — "OBJECT KYA HOTA HAI?"

🔷 Simple Soch (Real-Life)

Socho ek **aadmi / user / product**:

- Naam
- Age
- Gender
- Balance

👉 In sab ko **alag-alag variables** me rakhna messy ho jaata hai
👉 **Object** in sab ko **ek jagah pack** kar deta hai

🟨 Conclusion

Object = related data ka bundle

⬛ WHAT IS AN OBJECT?

🔷 Definition

Object = **key-value pairs** ka collection

- **Key** → property ka naam
- **Value** → data (string, number, boolean, object, function, etc.)

```
const obj = {
  name: "Bhupendar Jogi",
  "account-balance": 150,
  gender: "male",
  age: 20
```

```
};

console.log(obj);
```

🧠 **Real-World Mapping**

`name`, `age`, `gender` → properties
`"Bhupendar Jogi"`, `20` → values

⬛ IMPORTANT RULE ABOUT KEYS (VERY IMPORTANT)

🔷 Internally

👉 **JavaScript me saari keys string hoti hain**

```
const obj = { age: 20 };
```

Internally JS ise aise treat karta hai:

```
{ "age": 20 }
```

⚠️ Special Characters / Space in Keys

❌ Invalid

```
account number: 10500
```

✅ Valid

```
"account number": 10500
```

🟨 **Rule**

Space / special char ho → **quotes mandatory**

⬛ ACCESSING OBJECT PROPERTIES

---

① Dot Notation (Fast & Clean)

```
console.log(obj.name); // "Bhupendar Jogi"

console.log(obj.age);  // 20
```

---

② Bracket Notation (Flexible & Powerful)

```
console.log(obj["gender"]);          // "male"

console.log(obj["account-balance"]); // 150
```

⚡ When to use bracket notation?

- Key me **space** ho
- Key me **special character** ho
- Key dynamic ho (variable ke through)

⬛ NUMBER KEYS IN OBJECT

```
const obj = {
  0: "zero",
  1: "one"
};


console.log(obj[0]); // "zero"
```

🧠 Behind the scenes

`0` → `"0"` (string ban jaata hai)

⬛ SPECIAL KEYS (null & undefined)

```
const obj = {
  undefined: 50,
  null: "Harshal"
};

console.log(obj.undefined); // 50
console.log(obj["null"]);   // "Harshal"
```

🟨 **Reason**

JS keys ko string bana deta hai
`"undefined"`, `"null"`

⬛ WAYS TO CREATE OBJECTS ⬛

---

① OBJECT LITERAL { } (MOST COMMON)

```
const person = {
  name: "Harshal",
  age: 30
};
```

✅ **Pros**

- Simple
- Readable
- Small data ke liye perfect

❌ Cons

- Same structure baar-baar → repetitive code

② OBJECT CONSTRUCTOR — `new Object()`

```
const person = new Object();

person.name = "Alice";

person.age = 30;
```

✅ Pros

- Dynamic creation
- Step-by-step properties add

❌ Cons

- Verbose
- Rarely used in real projects

---

③ CONSTRUCTOR FUNCTION (TEMPLATE STYLE)

```
function Person(name, age) {
  this.name = name;
  this.age = age;
}


let p1 = new Person("Alice", 20);

let p2 = new Person("Bob", 30);
```

✅ Pros

- Reusable
- Template jaisa kaam karta hai

❌ Cons

- `this` + prototype beginners ke liye tough

---

④ CLASS (ES6 — MODERN OOPS STYLE)

```
class People {
  constructor(name, age, gender) {
    this.name = name;
    this.age = age;
    this.gender = gender;
  }
}

let p1 = new People("Alice", 20, "male");
```

✅ Pros

- Clean & modern
- Industry standard
- Inheritance support

❌ Cons

- Beginners ke liye thoda advanced

⬛ MODIFYING OBJECTS

```
let person = { name: "Saurav", age: 30 };

// Add

person.gender = "male";
```

```
// Update
person.age = 31;

// Delete
delete person.name;

console.log(person);
// { age: 31, gender: "male" }
```

⬛COMMON OBJECT METHODS ⬛

---

① Object.keys()

```
Object.keys(person); // ["age", "gender"]
```

---

② Object.values()

```
Object.values(person); // [31, "male"]
```

---

③ Object.entries()

```
Object.entries(person);
// [["age", 31], ["gender", "male"]]
```

---

④ Object.assign()

```
const obj1 = { a: 1 };

const obj2 = { b: 2 };


const obj3 = Object.assign({}, obj1, obj2);
```

⚠ **Rule**

Target me direct object mat do
Hamesha {} use karo

⬛ SPREAD OPERATOR . . . (MODERN WAY)

```
let obj3 = { ...obj1, ...obj2 };
```

Conflict Case

```
let objA = { name: "Harshal", age: 21 };

let objB = { age: 25, city: "Mumbai" };


let merged = { ...objA, ...objB };
```

🟨 **Rule**

Jo baad me spread hota hai → wahi overwrite karta hai

⬛ OBJECT FREEZE vs SEAL

Feature	freeze ❄	seal 🔒
Add	❌	❌

Delete	❌	❌
Modify	❌	✅

Example: `Object.freeze()`

```
Object.freeze(user);
```

👉 **Koi change allowed nahi**

Example: `Object.seal()`

```
Object.seal(person);
```

👉 **Sirf update allowed**

⬛ SHALLOW COPY vs DEEP COPY

Shallow Copy (Dangerous)

```
let obj2 = obj1;
```

👉 Same reference → risky for nested objects

Deep Copy (Safe)

```
let obj2 = structuredClone(obj1);
```

👉 Alag memory → independent

⚠️ **Nested objects me shallow copy avoid karo**

⬛ QUICK REVISION — EK NAZAR ME ⬛

✅ Object = key-value pairs
✅ Keys internally string hoti hain
✅ Access → dot (fast), bracket (flexible)
✅ Creation → `{}`, `new Object()`, constructor, class
✅ Modify → add / update / delete
✅ Methods → keys, values, entries
✅ Merge → assign / spread
✅ Freeze ❄️ = no change
✅ Seal 🔒 = update only
✅ Shallow vs Deep copy samajhna zaroori

⬛ FINAL THOUGHT ⬛

🧠 **JavaScript = Objects ka game**
Arrays, functions, JSON, APIs —
👉 **sab objects ke around hi ghoomta hai**