


LECTURE 15 — CALLBACK FUNCTIONS, `for...of`, `forEach`, `filter()` & `map()`

1 `for...of` LOOP — Iterable Values ke liye

Definition

 `for...of` loop iterable cheezon ke **VALUES** par loop lagata hai.

 Iterable = **Array**, **String**, **Set**, **Map**

 Plain objects iterable nahi hote

Syntax

```
for (let value of iterable) {  
    // code  
}
```

Example 1 — Array

```
const arr = [10, 20, 11, 18, 13];
```

```
for (let value of arr) {  
    console.log(value);  
}
```

Output

10

20

11

18

13

Example 2 — String

```
let str = "Saurav";
```

```
for (let char of str) {  
  console.log(char);  
}
```

Output

S

a

u

r

a

v

Important Rule

Objects iterable nahi hote

Kyunki ek key ke baad next memory location ka idea nahi hota.

✔ Object ko **for...of** ke saath kaise loop kare?

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

◆ Keys

```
for (let key of Object.keys(user)) {  
    console.log(key);  
}
```

Output

name

age

◆ Values

```
for (let value of Object.values(user)) {  
    console.log(value);  
}
```

Output

Harshal

21

◆ Entries (Best & Recommended)

```
for (let [key, value] of Object.entries(user)) {  
    console.log(key, value);  
}
```


Output

name Harshal

age 21

2 CALLBACK FUNCTION — “Function ke andar Function”

Definition

 **Callback function** wo function hota hai jo **dusre function ko argument ke roop me diya jata hai** aur **baad me execute hota hai**.

 Simple line:

“Main function ke kaam ke baad callback chalta hai”

Example 1 — Simple Callback

```
function greet(callback) {  
    console.log("Main greet function hu");  
    callback();  
}
```

```
function hello() {  
    console.log("Main callback function hu");  
}
```

```
greet(hello);
```

Output

```
Main greet function hu
```

```
Main callback function hu
```

Real-Life Example (Teacher–Student)

```
function teacher(checkHomework) {  
    console.log("Teacher: Padhata hu");  
    checkHomework();  
}
```

```
}
```

```
function student() {  
  console.log("Student: Homework dikhata hu");  
}
```

```
teacher(student);
```

● Output

Teacher: Padhata hu

Student: Homework dikhata hu

⚠ SUPER IMPORTANT

- ✓ `callback` → reference pass hota hai
- ✗ `callback()` → turant execute

📌 Golden Rule

Hamesha function ka reference pass karo

● ◆ `forEach()` — Action-Only Array Loop

📖 Definition

👉 `forEach()` array ke har element par function chalata hai, lekin kuch return nahi karta.

🔧 Example 1 — Print Values

```
let arr = [10, 20, 30, 40];
```

```
arr.forEach(num => {  
    console.log(num);  
});
```

● Output

```
10  
20  
30  
40
```

🖋 Example 2 — Index ke saath

```
arr.forEach((num, index) => {  
    console.log(num, index);  
});
```

● Output

```
10 0  
20 1  
30 2  
40 3
```

🖋 Example 3 — Array Modify karna

```
let nums = [10, 20, 30];  
nums.forEach((num, i, arr) => {  
    arr[i] = num + 2;  
});
```

```
console.log(nums);
```

● Output

```
[12, 22, 32]
```

⚠ Important Notes

- ❌ return value nahi hoti (undefined)
- ❌ break / continue allowed nahi
- ✅ sirf **action** ke liye best

📌 Memory Line

forEach = “Kaam kar, return mat kar”

🔴 ◆ filter() — Condition ke basis par chhantna

📖 Definition

👉 **filter()** condition ke basis par **naya array** banata hai.

- Original array ❌ change nahi hota
 - Sirf **true** wale elements select hote hain
-

🔧 Example 1 — Even Numbers

```
let arr = [10, 22, 33, 41, 50];
```

```
let even = arr.filter(num => num % 2 === 0);
```

```
console.log(even);
```

● Output

```
[10, 22, 50]
```

Example 2 — Objects Filter

```
const students = [  
  { name: "Rohan", marks: 40 },  
  { name: "Mohan", marks: 80 },  
  { name: "Saurav", marks: 90 }  
];  
  
const passed = students.filter(s => s.marks > 50);  
console.log(passed);
```

Output


```
[  
  { name: "Mohan", marks: 80 },  
  { name: "Saurav", marks: 90 }  
]
```

Memory Line

filter = “Jo chahiye wahi rakh”

map() — Transform karke naya array

Definition

 **map()** har element ko **modify/transform** karta hai aur **same length ka naya array** banata hai.

Example 1 — Square

```
let arr = [1, 2, 3, 4];
```

```
let squares = arr.map(n => n * n);  
console.log(squares);
```

● Output

```
[1, 4, 9, 16]
```

🔧 Example 2 — Objects se Names

```
const users = [  
  { name: "Amit", age: 20 },  
  { name: "Sumit", age: 25 }  
];  
  
const names = users.map(u => u.name);  
console.log(names);
```

● Output

```
["Amit", "Sumit"]
```

🔥 filter + map (Most Used Pattern)

```
let nums = [1, 2, 3, 4, 5, 6];  
  
let result = nums  
  .filter(n => n % 2 === 0)  
  .map(n => n * n);  
  
console.log(result);
```




Output

[4, 16, 36]

Memory Line

map = “Sabko badal, naya array bana”

6 FINAL QUICK COMPARISON

Method	Kya karta hai	Return
forEach	Action only	 undefined
filter	Select items	 new array
map	Transform items	 new array

SUPER-SHORT MEMORY LINES

- **for...of** → iterable values
- **callback** → baad me chalne wala function
- **forEach** → kaam kar, return nahi
- **filter** → jo chahiye wahi rakh
- **map** → sabko badal