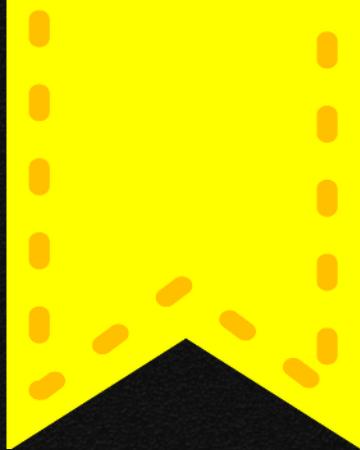


# Java Script

## Different Types of Loops

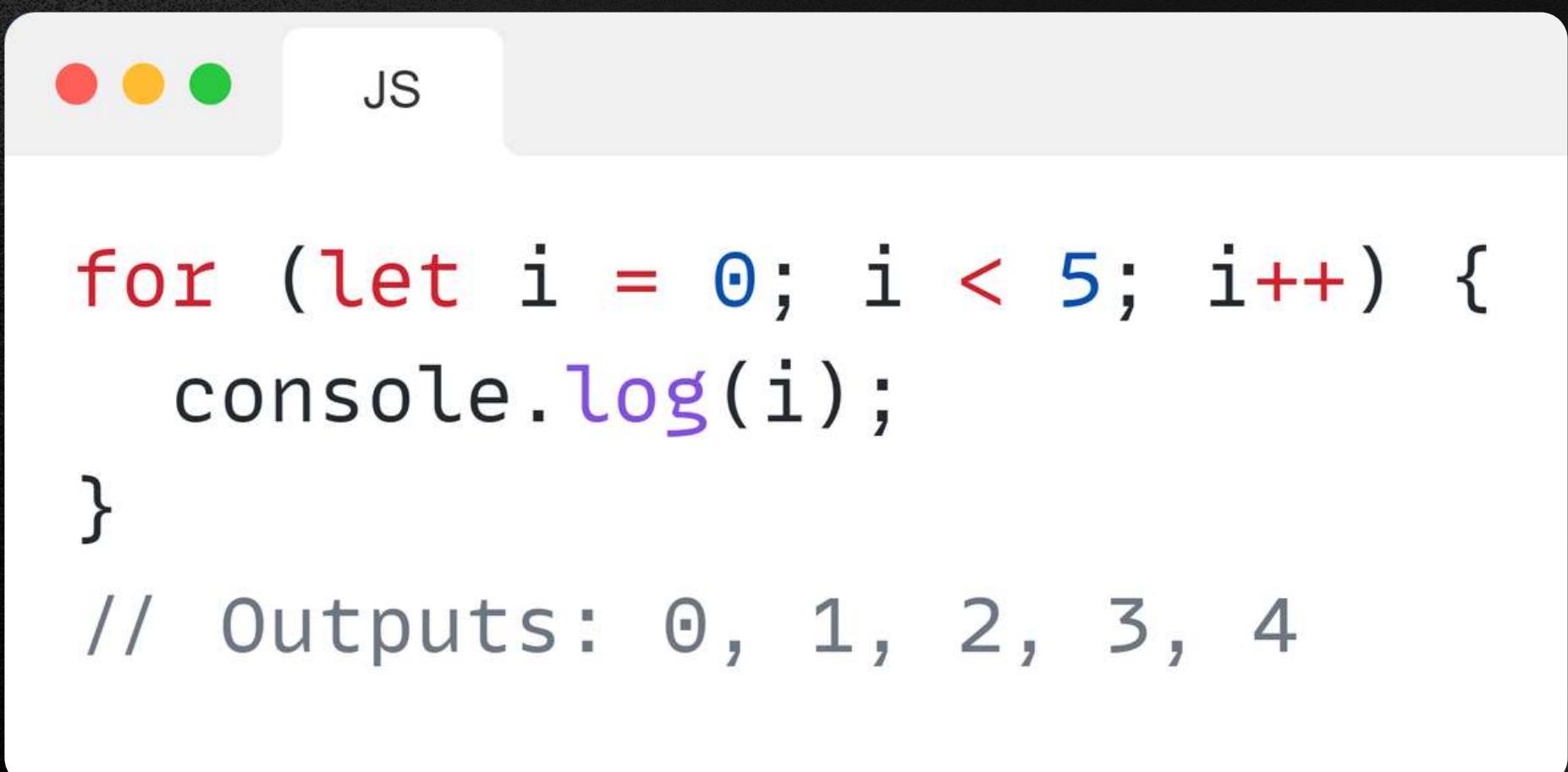
JS

[ Don't Miss It ]



# The Classical For Loop

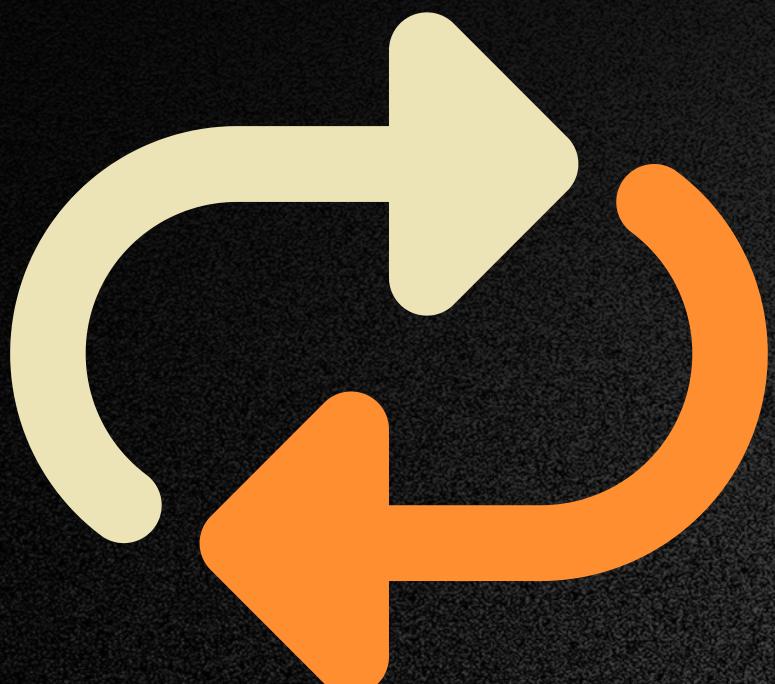
- The workhorse of loops, perfect for precise control over iteration.



A screenshot of a browser's developer tools console window. The title bar shows three colored dots (red, yellow, green) and the text "JS". The console itself contains the following code:

```
for (let i = 0; i < 5; i++) {  
    console.log(i);  
}  
  
// Outputs: 0, 1, 2, 3, 4
```

The code uses a for loop to iterate from 0 to 4, printing each value to the console. The output is shown as a comment at the bottom: "// Outputs: 0, 1, 2, 3, 4".



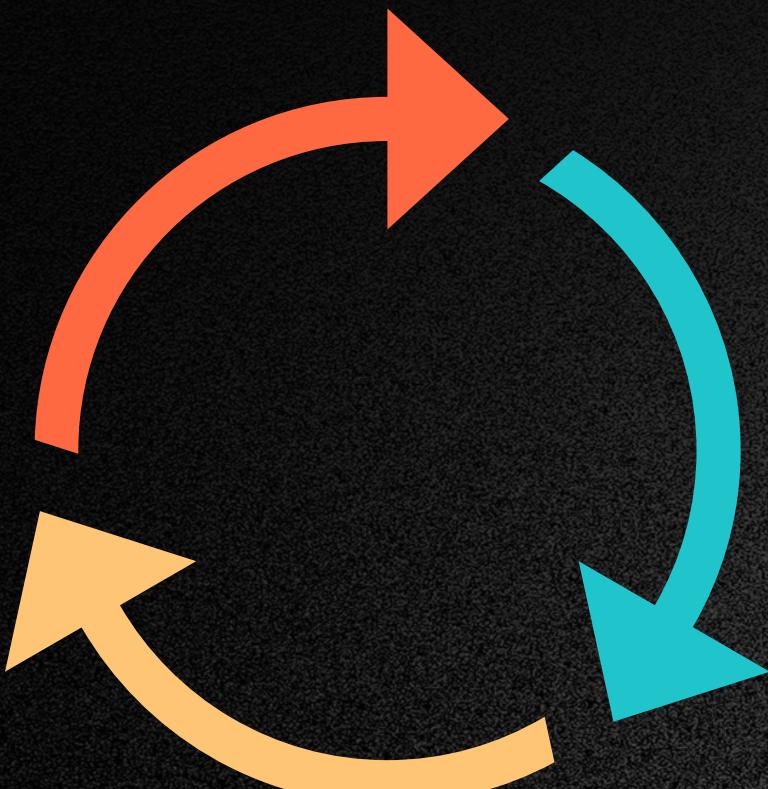
# for...of Loop

- Ideal for iterating over iterable objects like arrays or strings.

```
const techStack = ["JavaScript", "React", "Node.js"];

for (const tech of techStack) {
  console.log(tech);
}

// Outputs: JavaScript, React, Node.js
```



# forEach() Array Method

- A concise and expressive way to iterate through arrays.



JS

```
const numbers = [1, 2, 3, 4, 5];
```

```
numbers.forEach(number => {  
    console.log(number);  
});  
// Outputs: 1, 2, 3, 4, 5
```



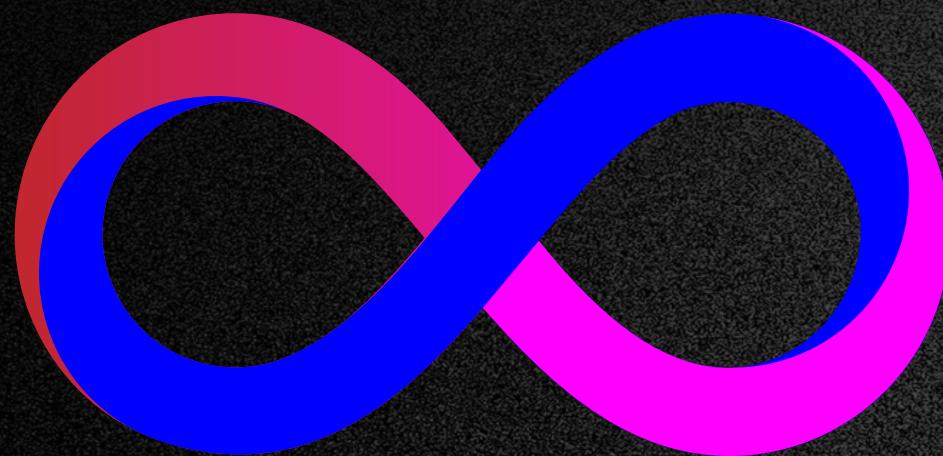
# for...in Loop

- Perfect for iterating over the properties of an object.

```
const user = { name: "TechExplorer",
               role: "Developer",
               language: "JavaScript" };

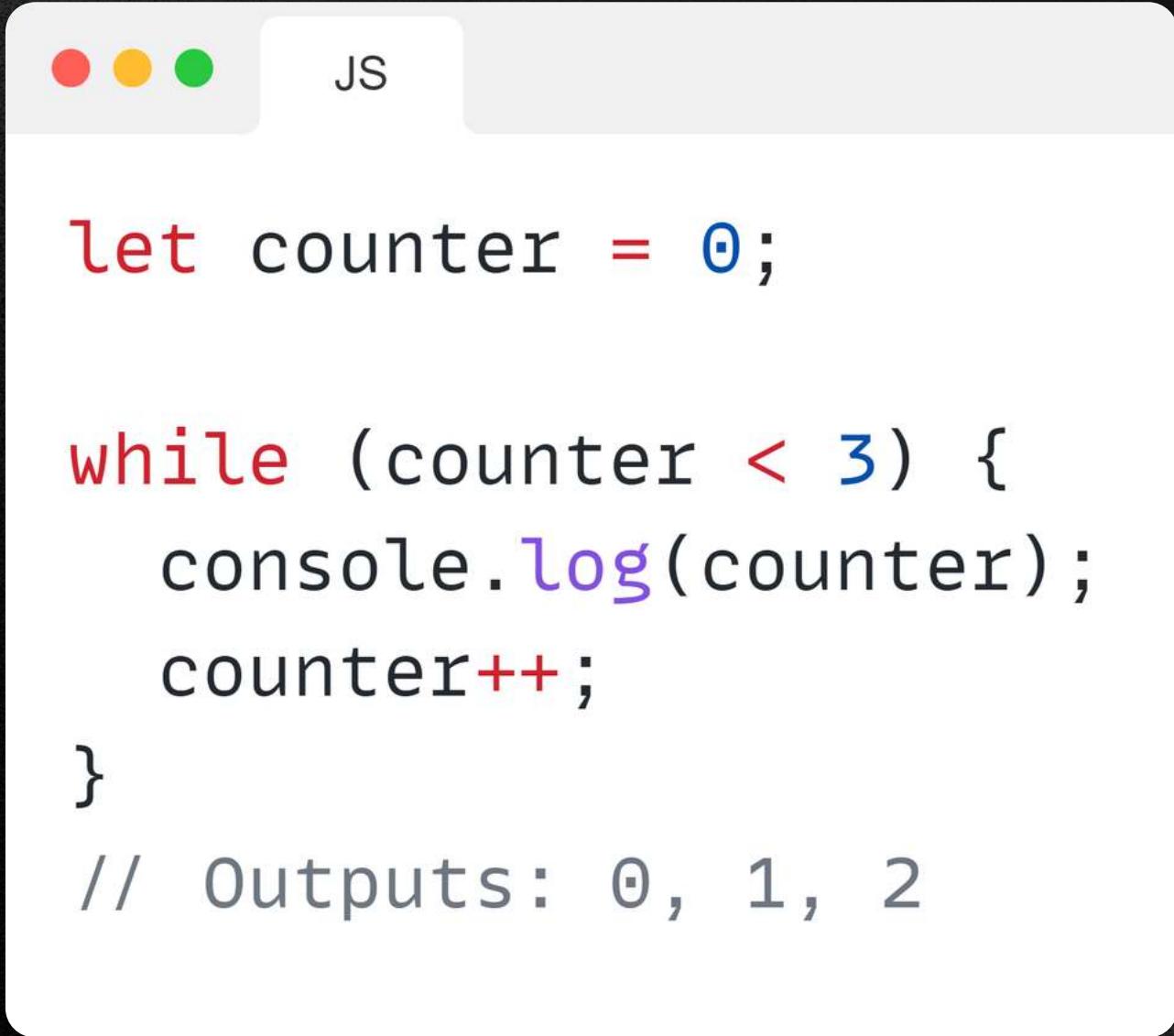
for (const key in user) {
  console.log(` ${key}: ${user[key]}`);
}

// Outputs: name: TechExplorer,
//           role: Developer,
//           language: JavaScript
```



# while Loop

- Use when the number of iterations is unknown or based on a condition.



The image shows a screenshot of a code editor window. The title bar has three colored dots (red, yellow, green) on the left and the text "JS" on the right. The main area contains the following JavaScript code:

```
let counter = 0;

while (counter < 3) {
    console.log(counter);
    counter++;
}

// Outputs: 0, 1, 2
```



# do...while Loop

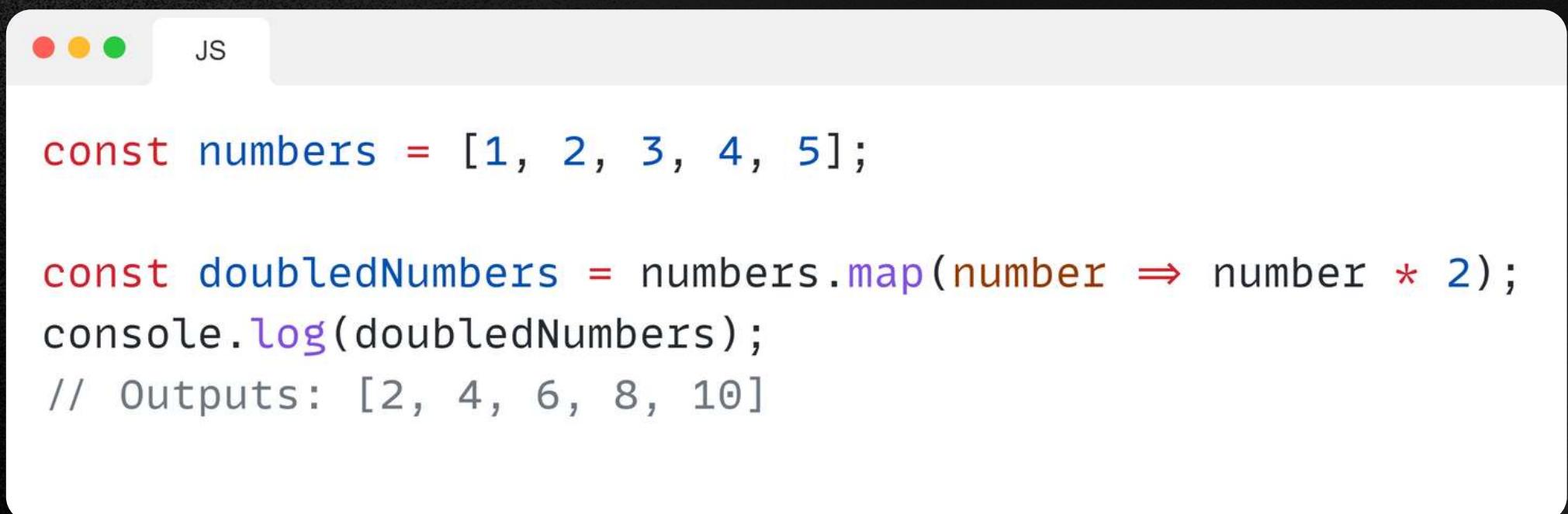
- Ensures at least one execution before checking the condition.

```
JS  
let count = 0;  
  
do {  
    console.log(count);  
    count++;  
} while (count < 3);  
// Outputs: 0, 1, 2
```



# Array Methods (map, filter, reduce)

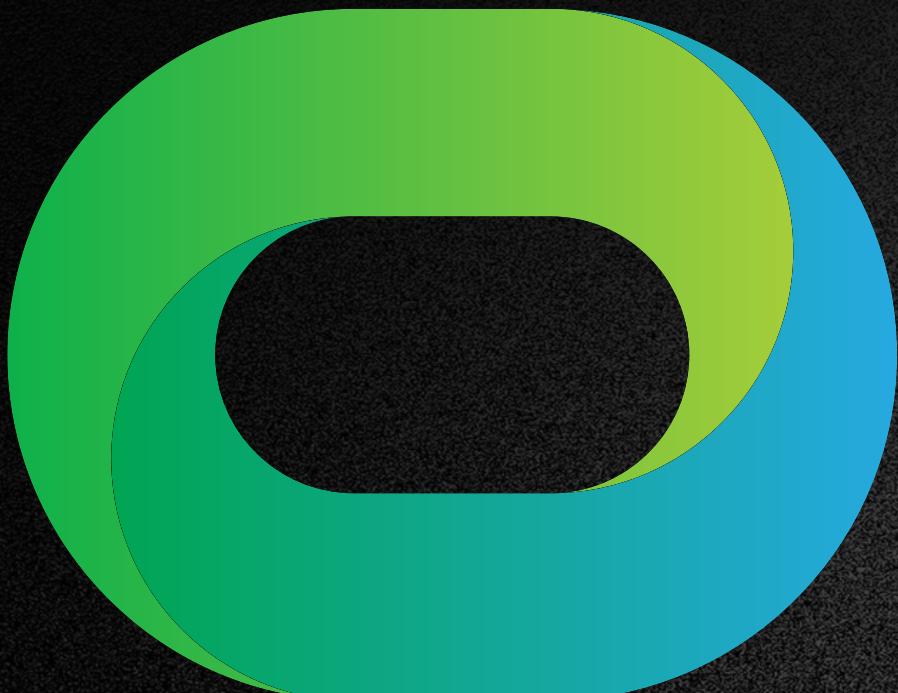
- Elevate your array operations with these functional programming gems.



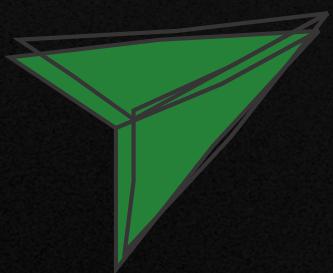
A screenshot of a browser window with a light gray header bar containing three colored dots (red, yellow, green) and the text "JS". The main content area contains the following JavaScript code:

```
const numbers = [1, 2, 3, 4, 5];

const doubledNumbers = numbers.map(number => number * 2);
console.log(doubledNumbers);
// Outputs: [2, 4, 6, 8, 10]
```



**DID  
YOU FIND  
IT  
HELPFUL ?**



Share this with a friend who needs it!



**VINCENT RAJA**  
FULL STACK WEB DEVELOPER