

1 JavaScript Loops (Overview)

Loops are used to **run the same code again and again** until a condition is met.

2 JavaScript for Loop

What it does

Runs code a **fixed number of times**.

Syntax

```
for (initialization; condition; increment/decrement) {  
    // code to repeat  
}
```

Example: Print numbers 1 to 5

```
for (let i = 1; i <= 5; i++) {  
    console.log(i);  
}
```

Example: Print even numbers

```
for (let i = 2; i <= 10; i += 2) {  
    console.log(i);  
}
```

Use

✓ When you **know how many times** the loop should run.

3 JavaScript while Loop

What it does

Runs code **while the condition is true**.

Syntax

```
while (condition) {  
    // code  
}
```

Example: Print numbers 1 to 5

```
let i = 1;
```

```
while (i <= 5) {  
    console.log(i);  
    i++;  
}
```

Example: Countdown

```
let i = 5;
```

```
while (i > 0) {  
    console.log(i);  
    i--;  
}
```

Use

✓ When you **don't know how many times** the loop will run.

4 JavaScript break Statement

What it does

Stops the loop immediately, even if the condition is still true.

Example

```
for (let i = 1; i <= 10; i++) {  
  if (i === 5) {  
    break;  
  }  
  console.log(i);  
}
```

Output

1
2
3
4

Use

✓ To **exit loop early** (searching, errors, stop condition).

5 JavaScript continue Statement

What it does

Skips the current iteration and continues with the next one.

Example

```
for (let i = 1; i <= 5; i++) {  
  if (i === 3) {  
    continue;  
  }  
  console.log(i);  
}
```

Output

1

2

4

5

Use

✓ When you want to **skip a value**, not stop the loop.

break vs continue

Statement What it does

break Stops loop completely

continue Skips current iteration

Real-Life Example

Find first number divisible by 7

```
for (let i = 1; i <= 50; i++) {  
  if (i % 7 === 0) {  
    console.log("First divisible by 7:", i);  
    break;  
  }  
}
```

Summary

Loop Best Use

for Fixed repetitions

while Unknown repetitions

break Stop loop

continue Skip iteration
