

Control Structures & Functions

Web Development Essentials - Session 10

Session Overview

Learning Goals for Today:

- Understand how conditional statements work using `if-else`
- Learn how to use loops to repeat code
- Introduction to functions: declaring, invoking, and passing parameters

What Are Control Structures?

Definition: Control structures allow developers to control the flow of a program by executing code conditionally or repeatedly.

Types of Control Structures:

- **Conditional Statements:** Code that runs based on conditions (e.g., `if-else`).
- **Loops:** Repeat a block of code multiple times (e.g., `for`, `while`).

Conditional Statements (if-else)

What is an if-else Statement?

- Conditional statements allow you to execute code based on whether a condition is true or false.

```
if (condition) {  
    // Code runs if condition is true  
} else {  
    // Code runs if condition is false  
}
```

Example

```
let age = 18;
if (age >= 18) {
  console.log("You are an adult.");
} else {
  console.log("You are a minor.");
}
```

Multiple Conditions (else-if)

What if there are more than two conditions?

- You can chain multiple conditions using `else if` for more complex decision-making.

```
if (condition1) {  
    // Code for condition 1  
} else if (condition2) {  
    // Code for condition 2  
} else {  
    // Code if none of the conditions are true  
}
```

Example

```
let score = 85;
if (score >= 90) {
  console.log("Grade: A");
} else if (score >= 75) {
  console.log("Grade: B");
} else {
  console.log("Grade: C");
}
```

Logical Operators in Conditions

Logical Operators:

- **&& (AND)**: Both conditions must be true.
- **|| (OR)**: At least one condition must be true.
- **! (NOT)**: Reverses a condition (true becomes false, false becomes true).

```
let age = 25;
let hasLicense = true;

if (age >= 18 && hasLicense) {
  console.log("You can drive.");
} else {
  console.log("You cannot drive.");
}
```


Loops (for, while)

What are Loops?

- Loops allow you to run the same block of code multiple times.

Types of Loops:

1. **for** loop
2. **while** loop

The for Loop

What is a for Loop?

- A loop that repeats code a specific number of times.

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

Example:

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

The while Loop

What is a while Loop?

- A loop that continues running as long as the condition is true.

```
while (condition) {  
    // Code to repeat  
}
```

Example:

```
let i = 0;  
while (i < 5) {  
    console.log("Count: " + i);  
    i++;  
}
```

Functions in JavaScript

What are Functions?

- A function is a block of code designed to perform a specific task. It can be reused multiple times by "invoking" it.

Declaring a Function:

- Use the `function` keyword to declare a function.

```
function functionName() {  
    // Code to execute  
}
```

Example

```
function greet() {  
  console.log("Hello, World!");  
}  
  
// Calling (invoking) the function  
greet(); // Output: "Hello, World!"
```

Parameters and Arguments

What Are Parameters?

- Parameters are variables that are passed into a function when it's called, allowing the function to work with different inputs.

```
function functionName(parameter1, parameter2) {  
  // Code to execute  
}
```

Example:

```
function greet(name) {  
  console.log("Hello, " + name);  
}  
  
greet("Alice"); // Output: "Hello, Alice"  
greet("Bob");   // Output: "Hello, Bob"
```

Returning Values from Functions

What is a Return Statement?

- A return statement allows a function to return a value back to the calling code.

```
function functionName() {  
  return value;  
}
```

Example:

```
function add(a, b) {  
  return a + b;  
}  
  
let sum = add(5, 10);  
console.log(sum); // Output: 15
```

Hands-On Activity: Creating Functions

Goal: Write a function that accepts two numbers, adds them, and returns the result.

Instructions:

1. Declare a function called `addNumbers`.
2. Pass two parameters to the function.
3. Return the sum of the two parameters.
4. Call the function and display the result.

```
function addNumbers(a, b) {  
  return a + b;  
}  
  
let result = addNumbers(4, 6);  
console.log(result); // Output: 10
```


Common Mistakes to Avoid

Not Declaring Variables: Always use `let`, `const`, or `var` to declare variables.

Infinite Loops: Make sure loop conditions eventually become false, or the loop will run indefinitely.

```
let i = 0;
while (i < 5) {
  console.log(i);
  // Missing i++ will result in an infinite loop
}
```

Forget to Call Functions: Remember to "invoke" the function to execute the code inside it.

Summary

What We Learned Today:

- Control structures using conditional statements (`if-else`).
- Loops (`for`, `while`) to repeat code.
- Declaring and invoking functions, and passing parameters.

Questions?

Q&A Session

- Any questions before we wrap up?

Thank You & See You in the Next Class!