ANT

Abstract of New Technology

Javascript Control Flow

Prepared By : **Mr. Chen Sovanminea**

ANT
Abstract of New Technology

Control Flow

- **Conditional Statements** and **Loops** in JavaScript (Using *if…else* structures and loops (like *for* and *while*) to control program flow based on conditions and execute code repeatedly).
- There are several methods that can be used to perform Conditional Statements in JavaScript such as :
  - If statement
  - If…else statement
  - If…else…if statement
  - Switch statement
  - Ternary operator

Control Flow (cont.)

1. If statement

- Executes a block of code if a specified condition is **true**.

Syntax

```
if ( condition ) {
    // Statement(s) to be executed if expression is true
}
```

Control Flow (cont.)

ANT
Abstract of New Technology

2. If...else statement

- The **if-else** statement will perform some action for a specific condition.
- Here we are using the **else** statement in which the **else** statement is written after the **if** statement and it has no condition in their code block.

Syntax

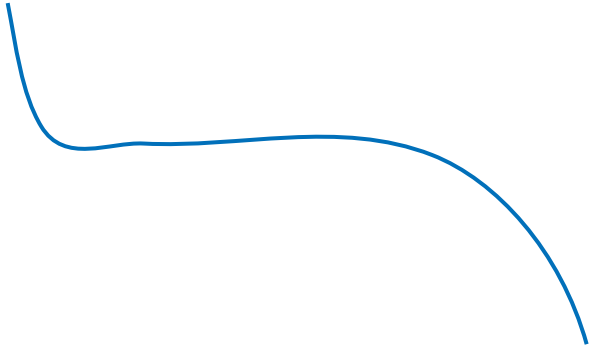
```
if ( condition ) {
    // Statement(s) to be executed if expression is true
} else {
    // Statement(s) to be executed if expression is false
}
```

Control Flow (cont.)

3. If...else if... statement

- The **else if** statement in JavaScript allows handling multiple possible conditions and outputs, evaluating **more than two options** based on whether the conditions are **true** or **false**.

Syntax



Control Flow (cont.)

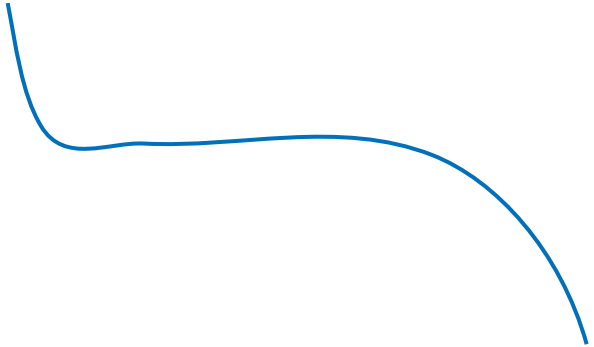
```
if (1st condition) {
    // Code for 1st condition
} else if (2nd condition) {
    // Code for 2nd condition
} else if (3rd condition) {
    // Code for 3rd condition
} else {
    // Code that will execute if all above conditions are false.
}
```

Control Flow (cont.)

4. Switch statement

- As the number of **conditions increases**, you can use multiple **else-if statements** in JavaScript. but when we dealing with many conditions, the **switch statement** may be a more preferred option.

Syntax



Control Flow (cont.)

```
switch (expression) {
    case value1:
        statement1;
        break;
    case value2:
        statement2;
        break;
    default:
        statement;
}
```

Control Flow (cont.)

5. Ternary Operator (?:)

- The conditional operator, also referred to as the ternary operator (**?:**), is a shortcut for expressing conditional statements in JavaScript.

Syntax

```
condition ? value if true : value if false
```

Example

```
let age = 21;
const result = (age >= 18) ? "You can vote." : "You cannot able to vote.";
console.log(result); // Output: You are eligible to vote.
```

Control Flow (cont.)

What is Loop?

- In JavaScript, **loop** is a control flow statement that allows code to be **executed repeatedly** based on a **condition**.
- It consists of three parts: **initialization**, **condition**, and **increment/decrement**.
  - For loop
  - While loop
  - Do...while loop
  - forEach loop
  - For...in loop
  - For...of loop

Control Flow (cont.)

1. For loop

- **For loop** is used to execute a block of code repeteatedly, until a specified condition evaluates to false.
- It can be used for iteration if the number of iteration is **fixed** and **known**.

Syntax

```
for (initialization; condition; iteration) {
    // Statement(s) to be executed if condition is true
}
```

Control Flow (cont.)

2. While loop

- **While loop** in JavaScript creates a loop that executes a block of code repeatedly, as long as the specified condition is **true**.

Syntax

```
while (expression) {
    // Statement(s) to be executed if condition is true
}
```

Control Flow (cont.)

3. Do...while loop

- **Do...while loop** is similar to the **while** loop except that the condition check happens at the end of the loop. This means that the loop will always be executed at least once, even if the condition is **false**.

Syntax

```
do{
    // Statement(s) to be executed;
} Statement(s) to be executed;
```

Control Flow (cont.)

4. forEach loop

- **forEach loop** is a built-in function that executes a provided function once **for each array element**. It does not return a new array and does not modify the original array. It's commonly used for **iteration** and **performing actions** on each array element.

Syntax

```
array.forEach(function (element, index, arr){
    console.log(index);
    console.log(element);
    console.log(arr);
});
```

Control Flow (cont.)

5. For-in loop

- **For-in loop** in JavaScript is used to loop through an object's properties.

Syntax

```
for (variableName in object) {
    // Statement(s) to be executed;
}
```

Control Flow (cont.)

6. For-of loop

- **For-of loop** in JavaScript is used to traverse elements of the iterable object.

Syntax

```
for (variableName of object) {
    // Statement(s) to be executed;
}
```

Control Flow (cont.)

Did you know?

- In Javascript, we also can terminates the loop by using **break statement**.
- When you use the break statement with the loop, the control flow jumps out of the loop and continues to execute the other code.

Syntax

```
break;
```

- We also can skip the current iteration of a loop and continue with the next iteration by using **continue statement**.

Syntax

```
continue;
```

Q & A?

ANT
Abstract of New Technology

Thanks!