JS Control Structures

Control Structures are statements used to control the flow of a program. They are used to skip over particular blocks of code, repeat blocks of code, or change the result of something based on certain conditions.

There are two types of control structures:

1. **Conditional Structures:** Conditional structures are used to tell the program to execute a block of code based on whether some conditions are met or not.
2. **Iterative Structures:** Iterative structures are used to repeat a block of code until a condition is met.

Conditional Structures

The If-statement

The if statement is used to check whether a condition evaluates to true or false and then run the code within it if it evaluates to true.

Syntax

```
If (condition to evaluate)
{
        code to run if the condition is true;
}
```

They are used to change a program's behavior based on a condition like so:

```
const type = "Text"

If (type == "Text")
{
        Console.log("Seven")
}

If (type == "Num")
{
        Console.log(7)
}
```

They are needed to have basic functionality in most programs as being able to change behavior based on user input is required in most programs.

```
Let num = prompt("What's your favourite number")

If (num > 0)
{
```

```
    Console.log("Your favourite number is positive.")
}
```

The If…else Statement

The if…else statement is used to pick between one of two blocks of code depending on whether a condition is true or false.

Syntax

```
If (condition)
{
    Code to execute if true.
}
Else
{
    Code to execute if false
}
```

The Switch Statement

The Switch statement is used to check a value and whether it matches a set of cases. It uses strict comparison (===) to compare values so their type must be the same too. Unlike in other programming languages, the operands of a switch statement can be of any data type.

Syntax

```
switch(operand whose value you want to check)
{
    Case value1:
        Code to execute
        Break;

    Case value2:
        Code to execute
        Break;

    Case value3:
        Code to execute
        Break;

    Default:
        Code to execute
        Break;

}
```

Iterative Structures

The For Loop

The for loop is used to repeat a block of code continuously until a condition is met.

Syntax

```
For (let i = 0; i < 6; i++)
{
        Code to execute
}
```

The for loop contains three expressions at the beginning. The first expression initializes a value to be used in the loop. It is optional and you can initialize the variable outside the for loop. The second expression is a condition to check for the loop to continue or end. It is optional but a break statement is needed if it is not used or else the loop will continue forever. The third statement is an increment or decrement that happens at the end of each loop. It is optional as you can do the same within the loop.

The For…In Loop

The for…in loop iterates over the properties of an object. It starts at the first property, moves to the next, and ends at the last property meaning it has no need for the expressions in the for loop.

Syntax

```
Const object = {property1: “This”, property2: “That”, property3: “There”}

For (let property in object)
{
        Code to be executed.
        The variable property contains the value of each property of object.
}
```

The For…Of Loop

The for…of loop iterates over the values of an array.

Syntax

```
Const numbers = [3, 10, 6, 5, 22, 31]

For (let x of numbers)
{
        Code to be executed
}
```

The While Loop

The while loop is used to repeat a block of code until a condition is met. The condition is evaluated first. If it’s true the block within the while loop is executed then the condition is checked again and so on.

Syntax

```
Let x = 5

While (x > 3)
{
        Do something.
        x--;
}
```

The Do…While Loop

The do…while loop is similar to the while loop except that the code inside is executed once before the condition is checked to execute the code again.

Syntax

```
Do {
        Code to be executed
}
While (conditizon)
```