Meaning

**JavaScript** is a lightweight, interpreted **programming** language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. **JavaScript** is very easy to implement because it is integrated with HTML. It is open and cross-platform.

JavaScript is one of the **3 languages** all web developers **must** learn:

   1. [**HTML**](https://www.w3schools.com/html/default.asp) to define the content of web pages

   2. [**CSS**](https://www.w3schools.com/css/default.asp) to specify the layout of web pages

   3. **JavaScript** to program the behaviour of web pages

Syntax

JavaScript syntax is the set of rules, how JavaScript programs are constructed:

var x, y, z;       // Declare Variables  
x = 5; y = 6;      // Assign Values  
z = x + y;         // Compute Values

JavaScript Values

The JavaScript syntax defines two types of values:

* Fixed values
* Variable values

Fixed values are called **Literals**.

Variable values are called **Variables**.

## JavaScript Literals

The two most important syntax rules for fixed values are:

1. **Numbers** are written with or without decimals:

10.50  
  
1001

2. **Strings** are text, written within double or single quotes:

"John Doe"  
  
'John Doe'

## JavaScript Variables

In a programming language, **variables** are used to **store** data values.

JavaScript uses the var keyword to **declare** variables.

An **equal sign** is used to **assign values** to variables.

In this example, x is defined as a variable. Then, x is assigned (given) the value 6:

var x;  
  
x = 6;

## JavaScript Operators

JavaScript uses **arithmetic operators** ( + - \* / ) to **compute** values:

Conditional Statement

## The if statement

The **if** statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.

### Syntax

if (expression) {

Statement(s) to be executed if expression is true

}

## The if...else statement

The **'if...else'** statement is the next form of control statement that allows JavaScript to execute statements in a more controlled way.

### Syntax

if (expression) {

Statement(s) to be executed if expression is true

} else {

Statement(s) to be executed if expression is false

}

## The if...else if... statement

The **if...else if...** statement is an advanced form of **if…else** that allows JavaScript to make a correct decision out of several conditions.

### Syntax

if (expression 1) {

Statement(s) to be executed if expression 1 is true

} else if (expression 2) {

Statement(s) to be executed if expression 2 is true

} else if (expression 3) {

Statement(s) to be executed if expression 3 is true

} else {

Statement(s) to be executed if no expression is true

}

The switch statement

The objective of a **switch** statement is to give an expression to evaluate and several different statements to execute based on the value of the expression. The interpreter checks each **case** against the value of the expression until a match is found. If nothing matches, a **default** condition will be used.

switch (expression) {

case condition 1: statement(s)

break;

case condition 2: statement(s)

break;

...

case condition n: statement(s)

break;

default: statement(s)

}

## The while Loop

The most basic loop in JavaScript is the **while** loop which would be discussed in this chapter. The purpose of a **while** loop is to execute a statement or code block repeatedly as long as an **expression** is true. Once the expression becomes **false,** the loop terminates.

### Syntax

while (expression) {

Statement(s) to be executed if expression is true

}

## The do...while Loop

The **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;

} while (expression);

The loop statement

The '**for**' loop is the most compact form of looping. It includes the following three important parts −

* The **loop initialization** where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.
* The **test statement** which will test if a given condition is true or not. If the condition is true, then the code given inside the loop will be executed, otherwise the control will come out of the loop.
* The **iteration statement** where you can increase or decrease your counter.

You can put all the three parts in a single line separated by semicolons.

### Syntax

for (initialization; test condition; iteration statement) {

Statement(s) to be executed if test condition is true

}