JavaScript TUTORIAL

What is JavaScript

JavaScript is a programming language initially designed to interact with elements of web pages. Within web browsers, JavaScript consists of three main parts:

- ECMAScript provides the core functionality.
- The Document Object Model (DOM) provides interfaces for interacting with elements on web pages
- The Browser Object Model (BOM) provides the browser API for interacting with the web browser.

JavaScript allows you to add interactivity to a web page. Typically, you use JavaScript with HTML and CSS to enhance a web page's functionality, such as validating forms, creating interactive maps, and displaying animated charts.

When a web page is loaded, i.e., after HTML and CSS have been downloaded, the JavaScript engine in the web browser executes the JavaScript code. The JavaScript code then modifies the HTML and CSS to update the user interface dynamically.

The JavaScript engine is a component of web browsers responsible for interpreting and executing JavaScript code. It includes a parser to analyze the code, a compiler to convert it into machine code, and an interpreter to run the compiled code.

Notable JavaScript engines include V8 in Chrome, SpiderMonkey in Firefox, and JavaScriptCore in Safari.

Initially, JavaScript engines were implemented as interpreters. However, modern JavaScript engines are commonly implemented as just-in-time compilers that compile JavaScript code to bytecode for improved performance.

Client-side vs. Server-side JavaScript

When JavaScript is used on a web page, it is executed in web browsers, serving as a client-side language.

JavaScript can run on both web browsers and servers. A popular JavaScript server-side environment is Node.js. Unlike client-side JavaScript, server-side JavaScript executes on the server and allows you to access databases, file systems, etc.

JavaScript History

In 1995, JavaScript was developed by Brendan Eich, a Netscape developer. Initially named Mocha, it was later renamed to LiveScript.

Netscape decided to rebrand LiveScript to JavaScript to capitalize on the popularity of Java. The decision was made shortly before the release of Netscape Navigator 2 web browser, leading to the introduction of JavaScript version 1.0.

Netscape launched JavaScript 1.1 in Netscape Navigator 3. In the meantime, Microsoft introduced its web browser called Internet Explorer 3 (IE 3) as a competitor to Netscape. However, IE featured its own JavaScript implementation called JScript. Microsoft used the name JScript to avoid potential licensing conflicts with Netscape.

Hence, two distinct versions of JavaScript were in the market:

- JavaScript in Netscape Navigator
- JScript in Internet Explorer.

At this time, JavaScript lacked standardized syntax and features, prompting the community to advocate for the standardization of the language.

In 1997, JavaScript 1.1 was proposed to the European Computer Manufacturers Association (ECMA) as a proposal. Technical Committee #39 (TC39) was assigned the task of standardizing the language, maiming to transform it into a general-purpose, cross-platform, and vendor-neutral scripting language.

TC39 came up with ECMA-262, establishing a standard for defining a new scripting language called ECMAScript (often pronounced Ek-ma-script).

Following that, the International Organization for Standardization and International Electrotechnical Commissions (ISO/IEC) adopted ECMAScript (ISO/IEC-16262).

JavaScript overview

> Note that the following section offers a solid introduction to JavaScript code. If you're not yet familiar with it, that's perfectly fine. You'll have the opportunity to learn in the upcoming tutorial.

To define a variable in JavaScript, you use `var` keyword. For example:

```
var x = 10;
var y = 20;
```

ES6 added a new way to declare a variable with the `let` keyword:

```
let x = 10;
let y = 20;
```

There are differences between `var` and `let`. And it's a good practice to use the `let` keyword to declare variables.

To declare a function, you use the `function` keyword. The following example defines a function that calculates the sum of two arguments:

```
function add( a, b ) {
   return a + b;
}
```

To call the `add()` function, you use the following syntax:

```
let result = add(x, y);
```

To log the result into the console window of the web browser or terminal (in the case of node.js), you use the `console.log()` function:

```
console.log(result);
```

Now, you should see `30` in the output.

JavaScript provides you with condition statements such as `if-else` and `switch` statements. For example:

```
let a = 20,
    b = 30;

function divide(a, b) {
    if(b == 0) {
       throw 'Division by zero';
    }
    return a / b;
}
```

In the `divide()` function, we check whether the de-numerator (b) is zero. If yes, we throw an exception. Otherwise, we return the result of a / b.

To declare an array, you use the following syntax:

```
let items = [];
```

To declare an array with initial elements, you specify those elements within the square brackets:

```
let items = [1, 2, 3];
```

You can access the number of elements in the `items` array through its `length` property:

```
console.log(items.length); // 3
```

To iterate over the elements of the `items` array, you use the `for` loop statement as follows:

```
for(let i = 0; i < items.length; i++) {
    console.log(items[i]);
}
```

Alternatively, you can use the `for...of` loop in ES6:

```
for(let item of items) {
    console.log(item);
}
```

JavaScript is an evolving language with rich features that you will explore in the upcoming tutorials.

In this tutorial, you learned about what JavaScript is and gained an overview of the JavaScript language.