The JavaScript Day 1

Yogesh Yadav

Very Powerful and popular language in this world

IDE

- **Bracket.io**
- Web strome
- Sublime
- Notepad++
- Visual Studio
- Web Matrix
- Dreamweaver
- Browser Developer tool

JavaScript

- JavaScript and Java are completely different languages, both in concept and design.
  JavaScript was invented by Brendan Eich in 1995, and became an ECMA standard in 1997.
  ECMA-262 is the official name. ECMAScript 6 (released in June 2015) is the latest official version of JavaScript.

- Javascript is one of the most simple, versatile and effective languages used to extend functionality in websites. Uses range from on screen visual effects to processing and calculating data on web pages with ease as well as extended functionality to websites using third party scripts among several other handy features.

Advantages Javascript

- JavaScript is the most popular programming language in the world.

- 1. Form validation can be done on the client side, which reduces the unnecessary round trips between the client and the server. This also means the load on the server is reduced and the application is more responsive.

  2. JavaScript uses the client machine processing power.

  3. With JavaScript partial page updates are possible i.e only portions of the page can be updated, without reloading the entire web form. This is commonly called as AJAX.

  4. JavaScript can also be used to animate elements on a page. For example, show or hide elements and sections of the page.

Disadvantages of JavaScript.

- **Security :** JavaScript runs on the client machine. So a malicious user may use Javascript to do a variety of things like tracking your browsing history, stealing passwords etc. This is one of the main reasons why people disable JavaScript.

  Browser Compatibility : Not all browsers treat the same piece of JavaScript in the same manner. This means the functionality and the user interface may vary from browser to browser. That is why cross-browser testing is very important. However, with JavaScript libraries like jQuery Browser Compatibility is no longer a major issue.

- **innerText** property is supported in IE & Chrome, but not in Firefox.

- **JavaScript can be stored either inline on the page or in an external .js file**.

- Head section include a reference to the external JavaScript file using script element as shown below
-

```
<script type="text/javascript" src="ExternalJavaScript.js"></script>
```

- **Advantages of external JavaScript over inline JavaScript :** Here are some of the general advantages of external JavaScript over inline JavaScript

  Maintainability : JavaScript in external files can be referenced on multiple pages without having to duplicate the code inline on every page. If something has to change, you only have one place to change. So external JavaScript code can be reused and maintenance will be much easier.

  Separation of Concerns : Storing JavaScript in a separate external .js file adheres to Separation of concerns design principle. In general it is a good practice to separate HTML, CSS and JavaScript as it makes it easier working with them. Also allows multiple developers to work simultaneously.

  Performance : An external JavaScript file can be cached by the browser, where as an inline JavaScript on the page is loaded every time the page loads.

•where should the script tag be placed in the html.

- Should it be placed in the body or head section of the page. In general the script tag can be placed either in the head or body section.

- JavaScript code is present before the textbox control. By the time the JavaScript code is executed, the textbox is still not loaded into browser DOM (Document Object Model). This is the reason JavaScript can't find the textbox and throws a NULL reference error.

- **what happens when a browser starts loading a web page.**
  1. The browser starts parsing the HTML

  2. When the parser encounters a <script> tag that references an external JavaScript file. The parser stops parsing the HTML and the browser makes a request to download the script file. Until the download is complete, the parser is blocked from parsing the rest of the HTML on the page.

  3. When the download is complete, that's when the parser will resume to parse the rest of the HTML.

- This means the page loading is stopped until all the scripts are loaded which affects user experience.

  In general, the suggested good practice is to place the <script> tag just before the closing <body> tag, so it doesn't block the HTML parser. However, modern browsers support **async** and **defer** attributes on scripts. These attributes tell the browser it's safe to continue parsing while the scripts are being downloaded. But even with these attributes, from a performance standpoint it is still better to place <script> tag just before closing <body> tag.

  According to HTTP/1.1 specification browsers download no more than two components in parallel. So, if the page has several images to download and if you place <script> tags at the top of the page, the script files start to download first which blocks the images download which adds to the total page load time.

```
<!doctype html>
<html>

<head>
    <meta charset="UTF-8">
    <title>Untitled Document</title>
    <script type="text/javascript">
        document.getElementById("TextBox1").style.backgroundColor = "red";
    </script>
</head>

<body>
    <input type="text" id="TextBox1">
</body>

</html>
```

JavaScript did not work in this case

JavaScript code is present before the textbox control. By the time the JavaScript code is executed, the textbox is still not loaded into browser DOM (Document Object Model). This is the reason JavaScript can't find the textbox and throws a NULL reference error.

Basic Syntax

- JavaScript is case sensitive programming language. Variable names, keywords, methods, object properties and event handlers all are case sensitive.

- **Example 1 :** alert() function name should be all small letters
  <script>
- alert("JavaScripts Basics Tutorial");
- </script>
- **Example 2 :** Alert() is not same as alert(). Throws Alert is not defined error. To see the error press F12 key.
  <script>
- Alert("JavaScripts Basics Tutorial");
- </script>

Comments in JavaScript

- 1) Single Line Comment

  Example :
  <script>
- // This is a sinle line comment
- </script>
-

  2) Multi Line Comment

  Example:
  <script>
- /* This is a
- multi line
- comment */
- </script>

Data types in JavaScript

- The following are the different data types in JavaScript
  Numbers - 5, 5.234
  Boolean - true / false
  String - "MyString", 'MyString'

  To create a variable in JavaScript use var keyword. Variable names are case sensitive.

- With JavaScript we always use var keyword to create any type of variable. Based on the value assigned the type of the variable is inferred.
  var a = 10;
  var b = "MyString";

- **JavaScript is a dynamically typed language**. This means JavaScript data types are converted automatically as needed during script execution. Notice that, in myVariable we are first storing a number and then a string later.
  <script>
- var myVariable = 100;
- alert(myVariable);
- myVariable = "Assigning a string value";
- alert(myVariable);
- </script>

Operators

- When a + operator is used with 2 numbers, JavaScripts adds those numbers.
  <script>
- var a = 10;
- var b = 20;
- var c = a + b;
- alert(c);
- </script>
- Output : 30
- When a + operator is used with 2 strings, JavaScript concatenates those 2 strings
  <script>
- var a = "Hello "
- var b = "JavaScript";
- var c = a + b;
- alert(c);
- </script>
- Output : Hello JavaScript

- When a + operator is used with a string and a number, JavaScript converts the numeric value to a string and performs concatenation.
  <script>
- var a = "Number is : "
- var b = 10;
- var c = a + b;
- alert(c);
- </script>
- Output : Number is 10

  <script>
- var a = "50"
- var b = 10;
- var c = a + b;
- alert(c);
- </script>
- Output : 5010

- But if you use a minus operator, numeric value is not converted to string
  <script>
- var a = "50"
- var b = 10;
- var c = a - b;
- alert(c);
- </script>
- Output : 40

getElementById

- This page contains some examples of what JavaScript can do.
- One of many HTML methods is **getElementById()**.

- This example uses the method to "find" an HTML element (with id="demo"**)**, and changes the element content (**innerHTML**) to "Hello JavaScript“

- document.getElementById("demo").innerHTML = "Hello JavaScript";

Strings in JavaScript

- A string is any text inside quotes. You can use either single or double quotes.
- var string1 = "string in double quotes"
- var string2 = 'string in single quotes'

- Concatenating strings : There are 2 options to concatenate strings in JavaScript. You could either use + operator or concat() method

Example : Concatenating strings using + operator

```
var string1 = "Hello"
var string2 = "JavaScript"
var result = string1 + " " + string2;
alert(result);
```

Output : Hello JavaScript

Example : Concatenating strings using concat() method

```
var string1 = "Hello"
var string2 = "JavaScript"
var result = string1.concat(" ", string2);
alert(result);
```

Output : Hello JavaScript

- If you want single quotes inside a string, there are 2 options
- Option 1 : Place your entire string in double quotes, and use single quotes inside the string wherever you need them.

- Example :
- var myString = "Welcome to 'JavaScript' Training";
- alert(myString);

- Output : Welcome to 'JavaScript' Training

- Option 2 : If you prefer to place your entire string in single quotes, then use escape sequence character \ with a single quote inside the string.

- Example :
- var myString = 'Welcome to \'JavaScript\' Training';
- alert(myString);

- Output : Welcome to 'JavaScript' Training