Module – 4(JavaScript Basic & DOM)

1. What is JavaScript?

JavaScript is a dynamic programming language that's used for web development, in web applications, for game development, and lots more. It allows you to implement dynamic features on web pages that cannot be done with only HTML and CSS.

Many browsers use JavaScript as a scripting language for doing dynamic things on the web. Any time you see a click-to-show dropdown menu, extra content added to a page, and dynamically changing element colors on a page, to name a few features, you're seeing the effects of JavaScript.

2. What is the use of isNaN function?

NaN values are generated when arithmetic operations result is undefined or unrepresented values. Such value does not necessarily represent overflow conditions.

For example, dividing zero by zero results in a NaN — but dividing other numbers by zero does not.

Examples

Using NaN()

```
Number.isNaN(NaN); // true
Number.isNaN(Number.NaN); // true
Number.isNaN(0 / 0); // true
Number.isNaN(37); // false
```

Difference between Number.isNaN () and global is NaN ()

Number.isNaN() doesn't attempt to convert the parameter to a number, so non-numbers always return false. The following are all false:

```
Number.isNaN ("NaN");
Number.isNaN (undefined);
Number.isNaN({});
Number.isNaN ("blabla");
Number.isNaN (true);
Number.isNaN (null);
Number.isNaN ("37");
Number.isNaN ("37.37");
Number.isNaN ("");
Number.isNaN (" ");
```

The global `isNaN ()` coerces its parameter to a number:

```
isNaN ("NaN"); // true
isNaN (undefined); // true
isNaN({}); // true
isNaN("blabla"); // true
isNaN(true); // false, this is coerced to 1
isNaN(null); // false, this is coerced to 0
isNaN("37"); // false, this is coerced to 37
isNaN("37.37"); // false, this is coerced to 37.37
isNaN(""); // false, this is coerced to 0
isNaN(" "); // false, this is coerced to 0
```

3. What is negative Infinity?

No other number is lesser than this value. It can be generated using a self-made function or by an arithmetic operation.

Note: JavaScript shows the NEGATIVE_INFINITY value as -Infinity.

Negative infinity is different from mathematical infinity in the following ways:

1. Negative infinity results in 0 when divided by any other number.
2. When divided by itself or positive infinity, negative infinity returns NaN
3. Negative infinity, when divided by any positive number (apart from positive infinity) is negative infinity.
4. Negative infinity, divided by any negative number (apart from negative infinity) is positive infinity.
5. If we multiply negative infinity with NaN, we will get NaN as a result.
6. The product of NaN and negative infinity is 0.
7. The product of two negative infinities is always a positive infinity.
8. The product of both positive and negative infinity is always negative infinity.

4. Which company developed JavaScript?

JavaScript was invented by Brendan Eich in 1995. It was developed for Netscape 2, and became the ECMA-262 standard in 1997. After Netscape handed JavaScript over to ECMA, the Mozilla foundation continued to develop JavaScript for the Firefox browser.

5. What are undeclared and undefined variables?

Undefined variable means a variable has been declared but it does not have a value.

Example

```
var car;
```

```
 console.log(car);
```

Undeclared variable means that the variable does not exist in the program at all.
Example

```
console.log(bike);
```

6. Write the code for adding new elements dynamically?

HTML defines the structure of your web document and the content therein. CSS declares various styles for the contents provided on the web document.

HTML and CSS are often called markup languages rather than programming languages, because they, at their core, provide markups for documents with very little dynamism.

JavaScript, on the other hand, is a dynamic programming language that supports Math calculations, allows you to dynamically add HTML contents to the DOM, creates dynamic style declarations, fetches contents from another website, and lots more.

Example:

https://github.com/Koshtisneha/WD/blob/main/JavaScript/13_nested%20loop.html

7. What is the difference between ViewState and SessionState?

The ViewState is to manage state at the client's end, making state management easy for end-user while SessionState manages state at the server's end, making it easy to manage content from this end too.

Usage

SessionState: It can be used to store information that you wish to access on different web pages.

ViewState It can be used to store information that you wish to access from same web page.

8. What is === operator?

When we compare two variables of different type e.g. a Boolean with a string or a number with string using == operator, it automatically converts one type into another and return value based upon content equality,

While === operator is strict equality and only return true if both variable of same type and also contains same value.

Example:

0==**false** // true, because false is equivalent of 0
0===**false** // false, because both operands are of different type
2=="2" // true, auto type coercion, string converted into number
2==="2" // false, since both operands are not of same type

9. How can the style/class of an element be changed?

10. How to read and write a file using JavaScript?

fs.readFile () and **rs.writeFile ()** methods are used to read and write of a file using JavaScript. The file is read using the fs.readFile() function, which is an inbuilt method.

Syntax :

```
fs.readFile( file_name, encoding, callback_function )
```

Parameters:

filename: It contains the filename to be read, or the whole path if the file is saved elsewhere.
encoding: It stores the file’s encoding. ‘utf8’ is the default setting.
callback function: This is a function that is invoked after the file has been read. It requires two inputs:
err: If there was an error.
data: The file’s content.
Return Value: It returns the contents contained in the file, as well as any errors that may have occurred.

fs.writeFile() function is used to write data to a file in an asynchronous manner. If the file already exists, it will be replaced.

Syntax:

```
fs.writeFile (file_name, data, options, callback)
```

Parameters:

file_name: It’s a string, a buffer, a URL, or a file description integer that specifies the location of the file to be written. When you use a file descriptor, it will function similarly to the fs. write() method.
data: The data that will be sent to the file is a string, Buffer, Typed Array, or DataView.
options: It’s a string or object that may be used to indicate optional output options. It includes three more parameters that may be selected.
encoding: It’s a string value that indicates the file’s encoding. ‘utf8’ is the default setting.
mode: The file mode is specified by an integer number called mode. 0o666 is the default value.
flag: This is a string that indicates the file-writing flag. ‘w’ is the default value.

callback: This function gets invoked when the method is run.
err: If the process fails, this is the error that will be thrown.

11. What are all the looping structures in JavaScript?

if you want to run the same code over and over again, each time with a different value.

Different Types of loops:

For : loops through a block of code a number of times

For/in : loops through the properties of an object

For/of : loops through the values of an iterable object

While : loops through a block of code while a specified condition is true

Do/while : also loops through a block of code while a specified condition is true

Example:

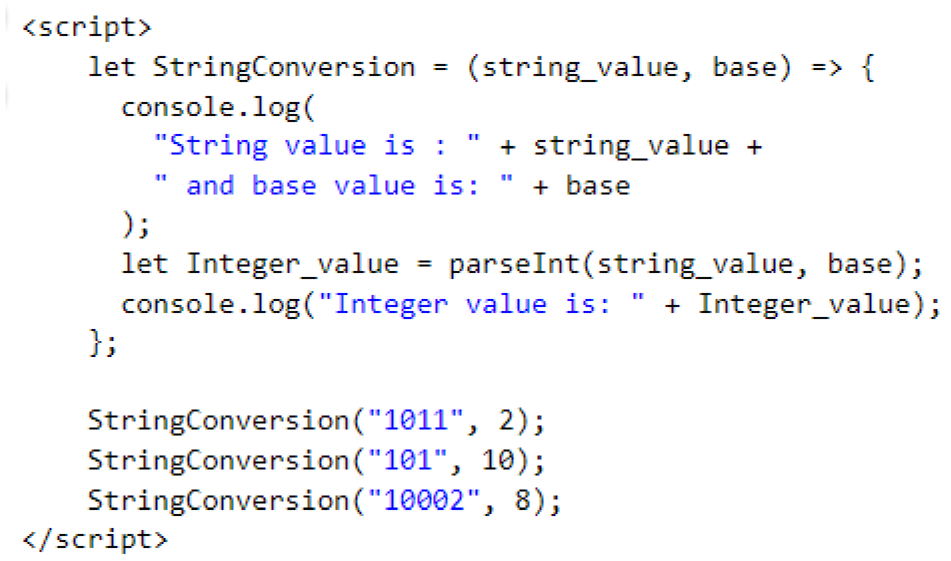
https://github.com/Koshtisneha/WD/blob/main/JavaScript/13_nested%20loop.html

12. How can you convert the string of any base to an integer in JavaScript?

parseInt () function (or a method) is used to *convert the passed in string parameter or value to an integer value itself*. This function returns an **integer** of base which is specified in second argument of **parseInt () function**.

ParseInt () function returns Nan (not a number) when the string doesn't contain number.

Example:



```
<script>
    let StringConversion = (string_value, base) => {
      console.log(
        "String value is : " + string_value +
        " and base value is: " + base
      );
      let Integer_value = parseInt(string_value, base);
      console.log("Integer value is: " + Integer_value);
    };

    StringConversion("1011", 2);
    StringConversion("101", 10);
    StringConversion("10002", 8);
</script>
```

Output:

```
String value is : 1011 and base value is: 2
Integer value is: 11
String value is : 101 and base value is: 10
Integer value is: 101
String value is : 10002 and base value is: 8
Integer value is: 4098
```

13. What are all the types of Pop up boxes available in JavaScript

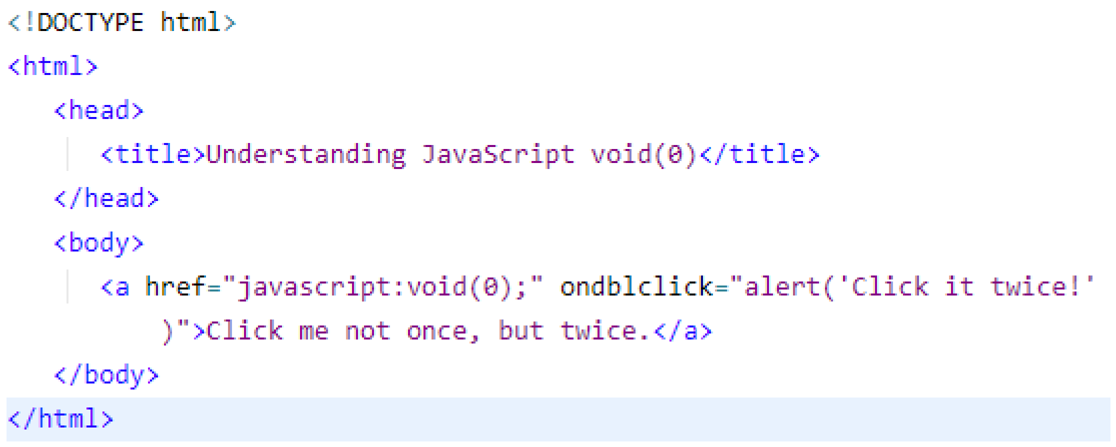
https://github.com/Koshtisneha/WD/blob/main/JavaScript/06_alertbox.html

https://github.com/Koshtisneha/WD/blob/main/JavaScript/07_confimbox.html

https://github.com/Koshtisneha/WD/blob/main/JavaScript/08_propmt.html

14. What is the use of Void (0)?

If inserting an expression into a web page results in an unwanted effect, then use JavaScript void to remove it. Adding “JavaScript: void (0)”, returns the undefined primitive value.



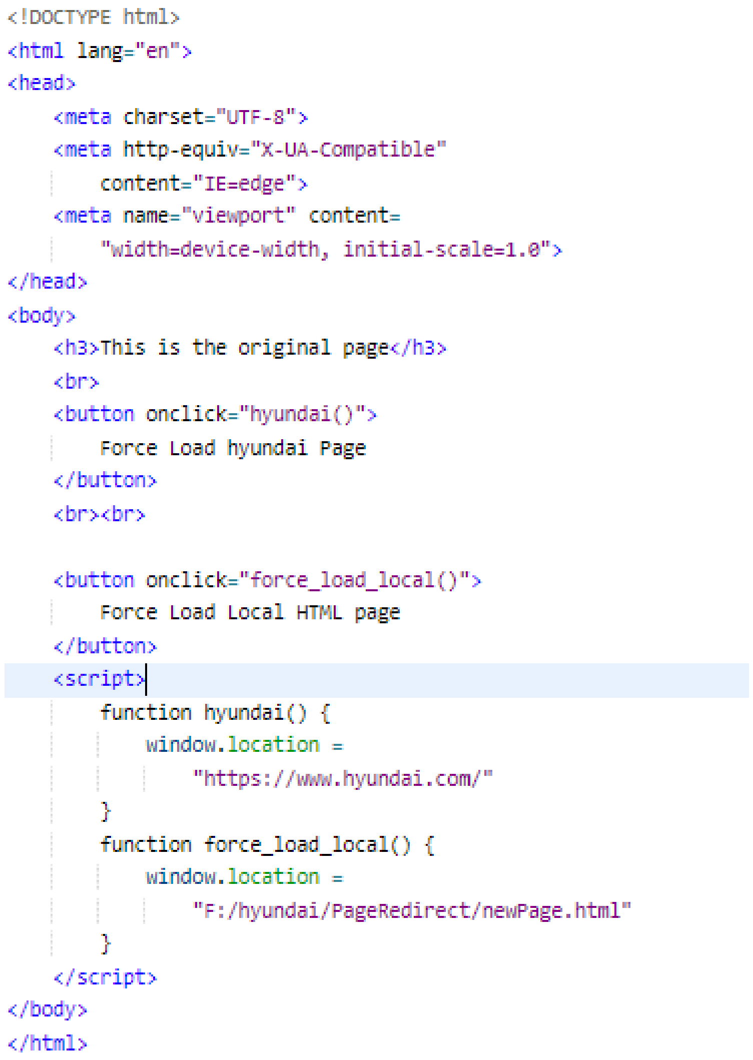
```
<!DOCTYPE html>
<html>
   <head>
      <title>Understanding JavaScript void(0)</title>
   </head>
   <body>
      <a href="javascript:void(0);" ondblclick="alert('Click it twice!'
          )">Click me not once, but twice.</a>
   </body>
</html>
```

15. How can a page be forced to load another page in JavaScript?

We can use ***window.location*** property inside the *script* tag to forcefully load another page in JavaScript. It is a reference to a Location object that is it represents the current location of the document. We can change the URL of a window by accessing it.

Syntax:

```
<script>
    window.location = <Path / URL>
</script>
```



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
        content="IE=edge">
    <meta name="viewport" content=
        "width=device-width, initial-scale=1.0">
</head>
<body>
    <h3>This is the original page</h3>
    <br>
    <button onclick="hyundai()">
        Force Load hyundai Page
    </button>
    <br><br>

    <button onclick="force_load_local()">
        Force Load Local HTML page
    </button>
    <script>
        function hyundai() {
            window.location =
                "https://www.hyundai.com/"
        }
        function force_load_local() {
            window.location =
                "F:/hyundai/PageRedirect/newPage.html"
        }
    </script>
</body>
</html>
```

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible"
        content="IE=edge">
    <meta name="viewport" content=
        "width=device-width, initial-scale=1.0">
    <title> New Page </title>
</head>

<body>
    <h3>This is the new loaded page</h3>
</body>

</html>
```

16. What are the disadvantages of using innerHTML in JavaScript?

The use of innerHTML very slow: The process of using innerHTML is much slower as its contents as slowly built, also already parsed contents and elements are also re-parsed which takes time.

Content is replaced everywhere: Either you add, append, delete or modify contents on a webpage using innerHTML, all contents is replaced, also all the DOM nodes inside that element are reparsed and recreated.

Appending to innerHTML is not supported: Usually, += is used for appending in JavaScript. But on appending to an Html tag using innerHTML, the whole tag is re-parsed.

Can break the document: There is no proper validation provided by innerHTML, so any valid HTML code can be used. This may break the document of JavaScript. Even broken HTML can be used, which may lead to unexpected problems.