parseFloat ("10 20 30");
parseFloat ("10 years");
parseFloat ("years 10");

→ If the number cannot be converted, Nan is returned.

Number object Methods:-

These object methods belong the Number object:-

Number.isInteger()

Returns true if the argument is an integer.

Number.isSafeInteger()

Returns true if the argument is safe integer.

Number.parseFloat()

Converts a string to a number.

Number.parseInt()

Converts a string to a whole number.

→ the number methods above belong to the JavaScript Number object.

→ these methods can only be accessed like Number.isInteger().

→ using x.isInteger() where x is a variable, will result in an error.

TYPEError x.isInteger is not a function.

The Number.isInteger Method:-

returns true if the argument is an Integer.

Number.isInteger(10);

Number.isInteger(10.5);

The Number.isSafeInteger() Method:-

→ A safe integer is an integer that can be exactly

Number.isSafeInteger (123

The Number.parseFloat() method:-

Parses a string and returns a number.
Spaces are allowed. only the first number is returned.

```
Number.parseFloat ("10");
Number.parseFloat ("10.33");
Number.parseFloat ("10 20 30");
Number.parseFloat ("10 years");
Number.parseFloat ("years 10");
```

→ If the number can't be converted, NaN is returned

→ The Number methods Number.parseInt() and Number.parseFloat() are the same as the global methods parseInt() and parseFloat().

→ The Number.parseInt() method:

Parses a string and returns a whole number.
Spaces are allowed. only the first number is returned.

```
Number.parseInt ("-10");
Number.parseInt ("-10.33");
Number.parseInt ("10");
Number.parseInt ("10.33");
Number.parseInt ("10 20 30");
Number.parseInt ("10 years");
Number.parseInt ("years 10");
```

1. JavaScript EPSILON

Number.EPSILon is the difference between the smallest floating point number greater than 1 and 1.

```
let x= Number.EPSILON;
```

2. JavaScript MAX_VALUE:-

It is a constant representing the largest possible number in JavaScipL.

```
let x=Number.MAX_VALUE;
```

→ Number properties belong to the JavaScipt Number object.

→ These properties can only be accessed as Number.MAX_VALUE.

→ using x.MAX_VALUE, where x is a variable or a value, will return undefined.

```
let x=6;
x.MAXVALUE
```

JavaScipt MIN_VALUE:-

Number.MIN_VALUE is a constant representing the lowest possible number in JavaScipt.

```
let x= Number.MIN_VALUE;
```

JavaScipt MAX_SAFE_INTEGER:-

Number.MAX_SAFE_INTEGER represents the maximum safe integer in JavaScipL.

Number.MAX_SAFE_INTEGER is $(2^{53}-1)$.

```
let x=Number.MAX_SAFE_INTEGER;
```

JavaScipt MAX_SAFE_INTEGER:-

→ represents maximum safe integer in JavaScipt

MAX_SAFE_INTEGER is $(2^{53}-1)$.

let x = Number.MAX_SAFE_INTEGER;

Javascript MIN_SAFE_INTEGER():-

→ Number.MIN_SAFE_INTEGER represents the minimum safe integer in Javascript.

→ Number.MIN_SAFE_ENTER is $-(2^{53}-1)$

let x = Number.MIN_SAFE_INTEGER;

Javascript positive_INFINITY:-

let x = Number.POSITIVE_INFINITY;

POSITIVE_INFINITY is returned on overflow:-

let x = 1/0;

Javascript Negative Infinity:-

let x = Number.negative_INFINITY;

NEGATIVE_INFINITY → can, caps

is returned on overflow.

let x = -1/0;

Javascript NaN- Not a Number:-

NaN is a javascript reserved word for a number that is not a legal number.

let x = Number.NaN;

Trying to do arithmatic with a non-numeric String will result in NaN

let x = 100/"Apple";

```
<html>
<body>
<script type="text/javascript">
  alert("Hello Javatpoint");
</script>
</body>
</html>
```
→ In body tag.

→ in head:-

```
<html>
<head>
<script type="text/javascript">
  function msg(){
    alert("Hello Javatpoint");}
</script>
</head>
<body>
<p> Welcome to Javascript </p>
<form>
<input type="button" value="click"
 onclick="msg()"/>
</form>
</body>
</html>
```

Disadvantages of external Javascript:

→ the stealer may download the code's code using the url of the js file.

→ If two js files depend on one another, then a failure in one file may affect the execution of other dependent file.

→ the web browser needs to make an additional http request to get the js code.

→

Conditional statements:

→ If
→ If-else
→ if-else-if

Switch:-

```
<script>
```

```
var result;
switch(grade){
case 'A':
  result="A Grade";
  break;
case 'B':
  result="B Grade";
  break;
default:
  result="NO Grade";
}
document.write(result);
</script>
```

16/1/16

Loops:-

for loop → `for(i=1;i<=5;i++){ code to be executed }`

while → `while(condition){ code to be executed }`

do-while

for-in → used to iterate properties of an object.

```
do{
code to be executed
}while(condition);
```

Functions arguments:-

```
<script>
function getcube(number){
  alert(number*number*number);
}
</script>
<form>
<input type="button" value="click"
</form>
```

```
<script>
function msg(){
alert("hello! this is message");}
</script>
<input type=button onclick=msg() value="Call function"/>
```

With Return value:-

```
<script>
function getInfo(){
  return "Hello javatpoint";}
</script>
<script>
document.write(getinfo());
</script>
```

code globally.

new Function ([arg1],[arg2],... argn)

Function methods:-

apply():- It is used to call a function containing this value and a single Array of functions

bind() → used to create a new function.

call() - used to call a function containing this value and an argument list.

tostring:- returns result in form of string.

Array:- that represents a collection of similar type of elements.

3 ways to construct array in javascript

1. By array literal
2. By creating instance of Array directly (using new keyword).
3. By using an array constructor (using new keyword)

1) Array literal:-

var arrayname=[value1,value2,value3,...valueN];

<script>
var emo=["sonoo","vimal","patan"]; → sonoo vimal patan
for(i=0;i<emp.length;i++){ → it returns length of array
document.write(emp[i]+"
");
}
</script>

2) var arrayname=new Array(); → used to create instance of array

<script>
var i;
var emp=new Array(); → var array=[1,2,3,4,5]
emp[0]="Arun";
emp[1]="Varun";
emp[2]="John";
for(i=0;i<emp.length;i++){
document.write(emp[i]+"
");
}

Set:- Holds collection of unique values. We don't have duplicate elements.

→ Arun

```
<script>
var emp= new Array("jai","vijay","smith");
for(i=0;i<emp.length;i++){
document.write(emp[i]+"<br>");
}
</script>
```

→ o/p:- jai
vijay
smith.

document.getElementById():-

document.getElementById → returns element of specified id
→ to get value of the input text. But we need to define id for the input fields.

document.getElementByname()

Web browser:- Is software used to access and view websites on the internet. It interprets HTML, CSS and javascript code from web-servers and displays it as a readable, interactive webpage.

Ex:- Google chrome, FireFox, Safari.

→ requests content from a web-server, retrieves data,

Web server:- web server is a computer or program that hosts websites and serves content to web browsers upon request. It stores, processes, and delivers web pages to users browsers over the internet.

→ ex:- Apache, Microsoft IIS.

→ Receives requests from browsers (clients), fetches the appropriate resources (HTML, CSS, images, etc.) and sends them back to the browser for display.

website:- collection of web-pages, multimedia content and data often connected under a single domain name, that provides information, services or entertainment to users.

→ Acts as an online resource accessible via a browser, containing organized information and interactive functionality for users.

Ex:- www.google.com,
www.wikipedia.org.

→ The biggest difference in HTML4 and HTML is more HTML 4 is very big. syntax for doc-type and also meta charset.
→ very less code in HTML-5

<meta charset="UTF-8"/>

→ <head> is logical part. → title → ~~There~~ is no
→ <body> is visible part. → link → link external
meta.
script
style

→ In title
...(If title unable to display the end of this ... will provide which is called as elipses. If mouseover on 3-dots, then after will display.

link:-

<link href="path of resource" rel="realtion"

single or double or no quotations we can use
↓
But as per industry double is preferred.

<link href="html doc" type="type of source"

→ Global path → doesn't require any address.
→ local path → For real time it is not recommended, because. local path will different from different browsers.
→ file path
→ web path → All browsers support file path
→ root realted path

rel:-

<!Doctype html>
<html lang="en-IN">
<head>
<title>
Naresh i technologies
</title>
<link href="html5.png" rel="icon" type="image/ico"/>
</head>
<body>
</html>

→ for adding favicon.
Any image extension you must treat as ico.
extens

Meta:- Metadata. realted to search engine optimization

→ This are self closed tags.

font
color : bold;

<Script> → javascript
</Script> ; → Javascript Statement

↓
used to validate forms.

<Style> → Cascading style we can implement.
↓
type="text/css">
Styles;
Styles;
</Style>

Body :- text, images, hyperlinks, special characters, lists, tables, frames, forms etc.

Attribute is nothing but property.

HTML:-
Webapps → group of organisation.
HTML5 = HTML + CSS + JS
HTML5 = WHATWG + W3C
↳ 2-organisations.

Semantics:- Mainly used for E-newspapers.
↓ offers structural elements to construct better document.
3 types:-
→ General semantics → 13
→ Arabic semantics → 2 → 18
→ Chinese semantics → 3

General:- <section> defines sections in a document. Such as headers, footers or any other sections of document. It is paired tag.
<article> It is used to represent an article, more specifically, the content within the <article> tag is independent from the other content on the site. This could be forum post, a magazine or newspaper article, paired tag.
<header> :- used to display header for subheadings, version information, navigational

<footer> --- </footer>

```
<!doctype html>
<html lang="en-IN">
<head>
<title>
HTML5 semantics
</title>
</head>
<body>
<section>
<header> Power of HTML5!!
</header>
<article> <header> Introduction to HTML </header>
</article> <footer> All rights reserved &copy;!! </footer>
<footer>All rights reserved
&copy;!! </footer>
</section>
```

<hgroup>
→ indicates all groups of headings → we should mention all these in one heading.

<h1> Javascript </h1>
<h2> Javascript </h2>

<aside> → Display the existing text left or right.
aside by default nature is left>

<aside style='color:blue; font-style:italic; float:left;
width:120px; font-family:tahoma;>
↓
if we want right mention right.

we can keep image, audio,

details: tag can used to create an interactive widget that the user can open and close. The content of a <details> element should not be visible unless the open attribute is set.
→ the details tag currently supported in chrome and in safari on Mac.

<details> --- </details>

<summary> --- </summary>
<details>
<summary> Features of HTML5 !! </summary>
 HTML semantics
 HTML5 inline Elements
 HTML5 Forms
 HTML5 multimedia
 HTML5 Forms
 HTML5 WebRTC
 HTML5 Webstorage
</details>
</body>
</html>

⇒ Features of HTML5!!
→ if we want to show all these, then click on that, then only we can see.

open="open"
If we want to open by defaut.

<figure> → if we give figure, then it will show on middle.
<img src="html5.png">
</figure> <figcaption> Here this we show on left side.
↓ i.e auto adjustment or Padding is automatic done.
not for multiple lines.
HTML5 logo from WHITA tag </figcaption>.

Inline elements:-
or
Block-level elements:-

<mark> HTML5 </mark>
↓ wherever we required we can mention.
<mark style="background-color: light green"> Pupa </mark>

this we use for indicating specific lines.
<mark> The HTML tag is used for indicating text as marked or highlighted for reference purpose. This tag is used to highlight the text in the html document. The highlighted text get the much more emphasis with the surrounding text. We can mark the text using many properties from global attributes of HTML5 e.g changing font, background color font color etc.

i, em, cite, dfn
All are inline functions
H, code → different font style.
i, em, ++ → different font
same font