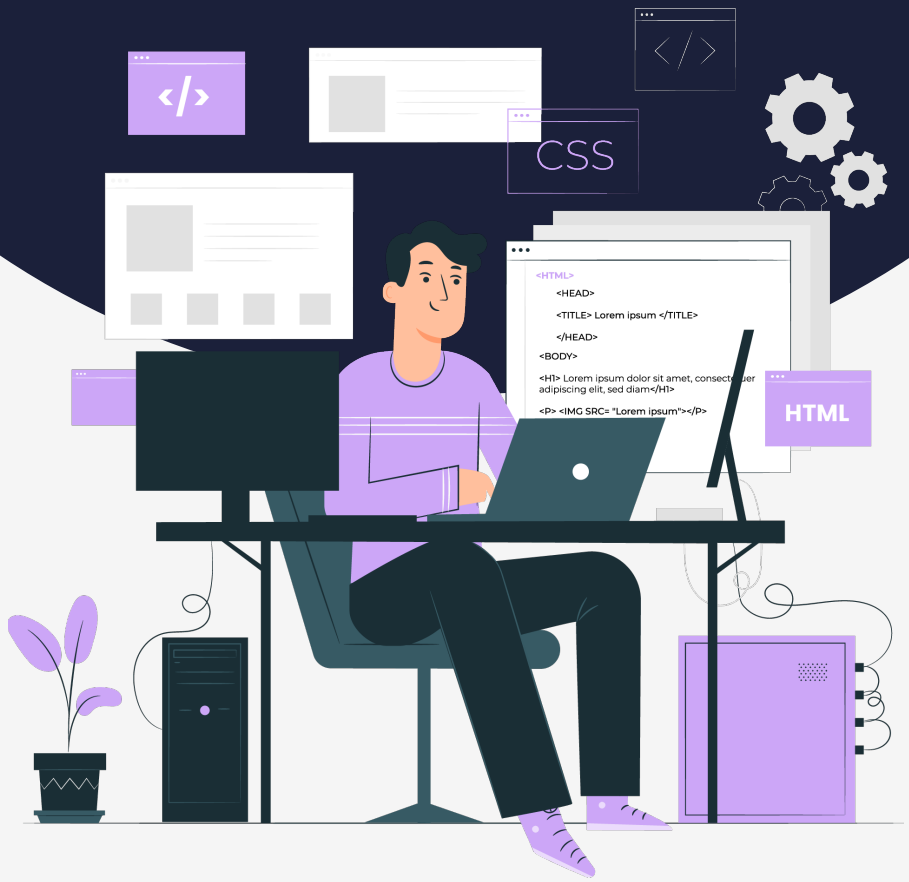
Lesson:

Math, String and Date in JavaScript



PW SKILLS

Topics

- Math with example
- String with property, method example
- Date in JavaScript

Math with Example -

In JavaScript, Math is an object with data properties and methods for processing numbers. It provides a range of mathematical functionalities.

Math Properties - Some of the commonly use math properties includes - **Math.PI, Math.E, Math.SQRT2, Math.LN2, Math.LN10**
Math.PI - It represents the ratio of the circumference of a circle to its diameter, approximately 3.14159
Math.SQRT2 - It represents the square root of 2.
Math.LN2 - It represents the natural logarithm of 2, which is approximately equal to 0.69315.
Math.LN10 - It represents the natural logarithm of 10 which is approximately 2.30259.
Example -

```javascript
console.log(Math.PI); // output -  3.141592653589793
console.log(Math.E); // output - 2.718281828459045
console.log(Math.SQRT2); //output -  1.4142135623730951
console.log(Math.LN2); // output -  0.6931471805599453
console.log(Math.LN10); // output - 2.302585092994046
```

Math methods - Some of the commonly use math properties includes - **Math.abs(), Math.ceil(), Math.exp(), Math. trunc(), Math.sqrt(), Math.round(), Math.random(), Math.pow(), Math.min(), Math.max(), Math.floor()**
Math.abs() - it returns the absolute value of a number.
Math.ceil() - it rounds up and returns the smaller integer greater than or equal to a given number.

Math.exp() - it returns x (any given number) raised to the power of a number.
Math. trunc() - it returns the integer part of a number by removing any fractional digits.
Math.sqrt() - it returns the square root of a number
Math.round() - it returns the value of a number rounded to the nearest integer
Math.random() - It returns a floating-point, pseudo-random number thats greater than or equal to 0 and less than 1.
Math.pow() - it returns the value of a base raised to a power.
Syntax - Math.pow(base, exponent)
Math.min() - It returns the smallest of the numbers given as input parameters.
Math.max() - it returns the largest of the number given as input parameters.
Math.floor() - it rounds down and return the largest integer less than or equal to a given number.

Example -

```javascript
console.log(Math.abs(-19)); // output - 19
console.log(Math.ceil(19.6)); // output 20
console.log(Math.exp(2)); // 7.38905609893065
console.log(Math.sqrt(4)); // 2
console.log(Math.round(44.3)); // 44
console.log(Math.random()); // float value between 0 to 1
console.log(Math.pow(2, 3)); //  8
console.log(Math.min(1, 22, 44, 2)); // 1
console.log(Math.max(1, 22, 44, 2)); // 44
console.log(Math.floor(4.55)); // 44
```

String with property, method example -

In JavaScript, the string is an object with a sequence of characters enclosed in single or double quotes used to represent and manipulate text, and it is one of the most commonly used datatypes in Javascript.

Example

```javascript
let hello = "hello world"
```

String Immutability -

Strings are immutable, which means that once a string is created, it cannot be changed. This immutability applies to individual characters within the string as well as the string as a whole. When you perform string operations or use non-string methods on a string, a new string is created rather than modifying the original string.

Example 1 -

```javascript
let firstName = "Mang";
let fullName = firstName.concat(" Touthang"); // Returns
// a new string "Mang Touthang"
console.log(fullName); // output - Mang Touthang (new
//string )
console.log(firstName); // Output: "Hello" (original
//string is unchanged)
```

Example 2 -

```javascript
let str = "Hello";
str[0] = "b"; // try to change no change
console.log(str); // output -  Hello
```

From the above example 2, the str with a value “Hello” is an attempt to be changed by assigning the value “b” to str[0], However when consoling the str variable, the same output original string is given i.e “Hello”. Since JavaScript string is immutable.

String Properties - it includes length, which returns the number of characters present in the string.Example

```
JavaScript
console.log("hello".length); // 5
```

String method - it includes charAt(), at() concat(), endsWith(), includes(), indexOf(), padEnd(), padStart(), repeat(), replace(), replaceAll(), search(), slice(), split(), startswith(), subString, toLowerCase(), toUpperCase(), trim(), trimEnd(), trimStart()

at() - it takes an integer value and returns a new string, it allows positive and negative integers. Negative integers count back from the last string character.
charAt() - it takes an integer value and returns the string located with the corresponding integer located in it.
concat() - this method concatenates (join) string
endsWith() - This method determines whether the string ends with the characters of a specified string. Returning true or false.
includes() - this method performs a case-sensitive search and determines whether one string may be found within another string.
indexOf() - this method searches for a string and returns its index position of the forest occurrence of the specified string. It return -1 if the char is not available.
padEnd() - this method pads the current string with a given string. So that the resulting string reaches a given length. The padding starts at the end.
Syntax - String.padEnd(targetlength, padString)

padStart() - pads the current string with another string until the resulting string reaches the given length. The padding is applied from the start of the current string.
Syntax - String.padStart(targetLenght, padString)

repeat() - method constructs and returns a new string which contains the specified number of copies of the string on which it was called.
Syntax - String.repeat(numberOfTimes)

replace() - method is used to replace a specified substring with another substring.
Syntax - String.replace(pattern, replacement)

replaceAll() - method returns a new string with all matches of a pattern replaced by a replacement.
Syntax - String.replace(pattern, replacement)

slice() - method extracts a section of a string and returns it as new string and returns it as a new string, without modifying the original string.
Syntax - String.slice(separator) String.slice(separator,limit)

split() - method takes a pattern and divides a String into an ordered list of substrings by searching for the pattern, puts these substrings into an array, and returns the array.

Syntax - String.slice(separator) String.slice(separator,limit)

startswith() - method determines whether a string begins with the characters of a specified string, returning true or false as appropriate.
subString - method returns the part of the string from the start index up to and excluding the end index
Syntax - String.subString(start), String.subString(start, end)

toLowerCase() - method returns the calling string value converted to uppercase
toUpperCase() - method returns the calling string value converted to lower case
trim() - method removes whitespace from both ends of a string and returns a new string, without modifying the original string.
trimEnd() - method removes whitespace from the end of a string and returns a new string, without modifying the original string.
trimStart() - method removes whitespace from the beginning of a string and returns a new string, without modifying the original string.

Example -

JavaScript

```
/**
 ******* String method ********8
 */
console.log("hello".at(1)); // e
console.log("hello".at(-1)); // o

console.log("chartAt".charAt(3)); // r

console.log("hello" + " " + "world"); // hello world

console.log("endswith".endsWith("with")); // true
console.log("endswith".endsWith("end")); // true

console.log("includes".includes("c")); // true
console.log("includes".includes("o")); // false

console.log("indexOf".indexOf("O")); // 5
console.log("indexOf".indexOf("m")); // 5

console.log("padEnd".padEnd(10, "!")); // padEnd!!!!

console.log("4514".padStart(10, "*")); // ******4514

console.log("hello".repeat(3)); // hellohellohello

console.log("hello world world".replace("world", "earth")); //
hello earth
console.log("hello world world".replaceAll("world", "earth"));
// hello earth earth

console.log("Hello World!".slice(6)); // World!
console.log("Hello World!".slice(6, 11)); // World

console.log("hello,world,hello,world".split(","));
// output - [ 'hello', 'world', 'hello', 'world' ]

console.log("hello World".startsWith("hell")); // true

console.log("hello".substring(2)); // llo
console.log("hello".substring(1, 3)); // el

console.log("Hello Word".toLowerCase()); // hello world
console.log("hello world".toUpperCase()); // HELLO WORLD

console.log("  hello world    ".trim());

console.log("  Hello    ".trimEnd()); //  "   Hello"
console.log("  Hello    ".trimStart()); // "Hello    "
```

Date JavaScript -

In JavaScript dates are object which works with dates and time. It provides methods for creating, manipulating, and formatting dates.

It can be created using the new Date() keyword.

Example -

```javascript
let date = new Date()
console.log(date)
// output - 2023-05-30T11:35:26.252Z
```

Date method - some of the importance date method include now(), getDate(), getDay(), getFullYear(), getHours(), getMilliseconds(), getHours(), getMillisiconds(), getMinutes(), getMonth(), getTime(), setDate(), setMonth(), setFullyear(), toDateString()

now() - method returns current date and time in milliseconds
getDate() - method of Date instances returns the day of the month for this date according to local time.
getDay() - method of Date instances returns the day of the week for this date according to local time, where 0 represents Sunday.
getFullYear() - method of Date instances returns the year for this date according to local time.
getHours() - method of Date instances returns the hours for this date according to local time.
getMillisiconds() - method of Date instances returns the milliseconds for this date according to local time.
getMinutes() - method of Date instances returns the minutes for this date according to local time.
getMonth() - method of Date instances returns the month for this date according to local time, as a zero-based value
getTime() - method of Date instances returns the number of milliseconds for this date since the January 1, 1970, UTC.
setDate() - method of Date instances changes the day of the month for this date according to local time.
setMonth() - method of Date instances changes the month for this date according to local time.
setFullyear() - method of Date instances changes the year, for this date according to local time.
toDateString() - method returns the date portion of a Date object interpreted in the local timezone in English.

Example -

```javascript
/**
 * Dates in javascript -
 */
/**
 * date in javascript
 */
let date = new Date();
console.log(date); // output - 2023-05-30T11:35:26.252Z

// now()
console.log(Date.now()); // 1685447212121 in millisecond

// getDate()
console.log(date.getDate()); // 30 present date of the day

// getDay()
console.log(date.getDay()); // week day in number i.e 2 which
is tuesday

// getFullYear()
console.log(date.getFullYear()); // 2023

// getHours()
console.log(date.getHours()); // 17

// getMilliseconds
console.log(date.getMilliseconds()); // 556

// getMinutes()
console.log(date.getMinutes()); // 28 current minutes
```

```
// getMonth()
console.log(date.getMonth()); // 4 present month in number.

// getTime()
console.log("GetTime ", date.getTime()); // 1685449330596

// Time
console.log(date.getDate()); // 30

// setDate
console.log(date.setDate(15)); // 1684152190884

// setMonth
console.log(date.setMonth(4)); // 1684152543926

// setFullYear
console.log(date.setFullYear(2020)); // 1589544543926

console.log(date.toDateString()); // Fri May 15 2020
```