Strings

Strings are collection of characters stored inside quotes (double or single) or back tick (in ES6). **Strings** represents text and can have lower case, upper case, special characters or numbers within quotes.

Strings can store data, like name, email id, age etc. Default datatype for input, select, textarea value is always a **string**.

In JavaScript, **Strings** are declared inside double quotes "" or single quotes ''. Strings started with double quotes should be closed by double quote and same strings started with single quote should be closed by single quote. Both `var x="hello"` and `var x='hello'` are same in javascript.

Backslash, i.e. `\` is used inside strings to ignore character. For exp `var x="abc\"pqr";`

We can also declare strings using let in ES6.

Declare Strings in JavaScript

```
var x="Tech Altum";    // string;
var y='Tech Altum';    // string;
var z=`Tech Altum`;    // template literal in ES6

var str=String("abc"); // String class or function
```

JavaScript String can store at max 2^53 - 1 characters as per ECMAScript 2016 (ed. 7).

Template Literals

ES6 introduced **Template Literals** or **Template Strings** in javascript by using back-tick or grave characters.

```
var str=`js string`;
```

String interpolation via template literal

To string interpolation or to insert a placeholder variable in **template literal**, use ${expression} inside.

```
var x=3;
var y=5;
console.log(`sum of ${x} and ${y} is ${x+y}`);                                    //ES6

console.log("sum of " + x + " and " + y + " is " + (x+y));                        //ES5

// returns "sum of 3 and 5 is 8"
```

Multi line string

Template literals can also add multi-line strings. Till ES5, we use + operator to do the same. See example of **Multi line string** .

```
var template=`<!doctype html>
    <html>
    <head>
    <meta charset="UTF-8">
    <title></title>
    </head>
    <body>
    </body>
    </html>`;
```

Properties of String

JavaScript Strings can have properties, like length. **Properties** are information about that string. Properties are checked using Dot notation (.) or brackets [ ] with property name in quotes. See example

String.length

String.length property shows the number of characters in string.

```
var x="hello js";
x.length;           // 8
```

String length is immutable or **Read Only** property. Means if the length of a string is 8 and we assign string.length to 10, it will still return 8. See example

```
var x="hello js";
x.length;         // 8

x.length=10;
x.length;        // 8

x.length=2;
x.length;        // 8
```

Empty String

An **empty string** is the string with length equals to zero.

```
var x="";
x.length;    // 0
```

String Methods

JavaScript Methods are build in functions used to perform an action to strings. All methods are called by method name and then parenthesis. Parameters are passed inside parenthesis if required.

For exp, `x.age` is property, but **x.getAge()** is a method.

indexOf

indexOf methods returns index of first matched substring. The return value of **indexOf** methods is always a number.

If substring is not found, it will return -1.

```
var x="tech altum";

x.indexOf("t")         // return 0;
x.indexOf("e")         // return 1;
x.indexOf("b")         // return -1;
```

lastIndexOf

lastIndexOf methods returns index of last matched substring. The return value of **lastIndexOf** method will also be number.

If substring is not found, it will return -1.

```
var x="tech altum";

x.indexOf("t")             // return 0;
x.lastIndexOf("t")         // return 7;
x.lastIndexOf("e")         // return 1;
x.lastIndexOf("z")         // return -1;
```

concat

concat method is used to concatenate or merge two strings into one. The second string should be passed as argument.

```
var x="tech";
var y="altum";

x.concat(y);    // return techaltum
```

+ operator can also concat two strings in javascript.

```
var x="tech";
var y="altum";
```

```
x+y;    // return techaltum
```

charAt

charAt method return character at given index in argument.

```
var x="techaltum";

x.charAt(0);    // return t
x.charAt(1);    // return e
x.charAt(x.length-1);    // return m
x.charAt(x.length);      // return ""
```

charCodeAt

charCodeAt method return ASCII code of character at given index in argument.

```
var x="techaltum";

x.charCodeAt(0);    // return 116
x.charCodeAt(1);    // return 101
```

toUpperCase

toUpperCase method convert all lowercase characters to uppercase.

```
var x="techaltum";

x.toUpperCase();    // return "TECHALTUM"
```

toUpperCase method only return string in uppercase, but value of x will remain same, .ie. lowercase

toLowerCase

toLowerCase method convert all uppercase characters to lowercase.

```
var x="Tech Altum";

x.toLowerCase();    // return "techaltum"
```

substr

substr method return substrings from index (*first argument*) to given no of characters (*second argument*). First argument is always index and second is no of characters. If second argument is missing, it will return substrings after first index. See examples

```
var x="techaltum";

x.substr(2);        // return "chaltum"
x.substr(4);        // return "altum"

x.substr(0,4);        // return "tech"
```

```
x.substr(2,2);        // return "ch"
x.substr(4,5);        // return "altum"
```

For IE8 and below, use substring

search

search method search for a matched pattern between string and regular expression and return index of matched pattern.

```
var x="tech altum";
x.search("t");        // return 0
x.search("T");        // return -1

x.search(/\s/);
// return 4, i.e, white space found at 4th index

x.search(/\d/);
// return -1, i.e, no digit found
```

To know more about regular expressions, click here. JavaScript Regular Expressions

trim

trim method trim extra whitespace from beginning and ending.

```
var x=" tech altum ";
x.trim();    // return "tech altum"
```

replace

replace method is used to replace a single character or word with a new characters. The first argument is replaced character, and second is replacing one.

```
var x="tech altum";
x.replace("t","T")            // return "Tech altum"
x.replace("tech","TECH")      // return "TECH altum"
x.replace(" ","-")            // return "tech-altum"
```

Replace all characters from string

```
var x="tech altum";

x.replace(/t/,"T");        // return Tech  altum
x.replace(/t/g,"T");       // return Tech  alTum
```

split

split method splits a string to array

```
var x="tech altum";
x.split(" ");        // return ["tech","altum"]
```

startsWith

string.startWith method is used to check if the given **strings starts with specific characters** or not. This will return true or false.

```
var x="tech altum";

console.log(x.startWith("t"));                                                    // true
console.log(x.startWith("T"));                                                   // false
console.log(x.startWith("a"));                                                   // false
```

endWith

string.endWith method is used to check if the given **strings ends with specific characters** or not. This will also return true or false.

```
var x="tech altum";

console.log(x.endWith("m"));                                                      // true
console.log(x.endWith("t"));                                                     // false
```

includes

string.includes method is used to check if the given **strings contain specific characters** or not. This will also return true or false.

```
var x="tech altum";

console.log(x.includes("ch"));                                                    // true
console.log(x.includes("cm"));                                                   // false
```

Get String from ASCII Code

To get string value from ASCII code, use **String.fromCharCode** method and pass ASCII code as argument. See example

```
String.fromCharCode(102);    // return f
String.fromCharCode(65);    // return A
```

"+" operator and "concat()" method, both concat strings in javascript.