Instructor: Sudad H. Abed / Email: shabed@uoanbar.edu.iq

Lecture Reference: www.w3schools.com

❖ JavaScript String Methods

String methods help you to work with strings.

Method	Description
length	The length property returns the length of a string.
indexOf()	The indexOf() method returns the index of (the position of) the first occurrence of a specified text in a string.
lastIndexOf()	The lastIndexOf() method returns the index of the last occurrence of a specified text in a string.
str.slice(x, y);	• slice() extracts a part of a string and returns the extracted part in a new string. • The method takes 2 parameters: the starting index (position), and the ending index (position). • If you omit the second parameter, the method will slice out the rest of the string.
str.substr(x, y);	substr() is similar to slice(). The difference is that the second parameter specifies the length of the extracted part.
str.replace(x , y)	The replace() method replaces a specified value with another value in a string.

Note:

✓ Both the indexOf(), and the lastIndexOf() methods return -1 if the text is not found.

❖ JavaScript Number Methods

Number methods help you work with numbers.

• Number Properties

Property	Description
MAX_VALUE	Returns the largest number possible in JavaScript
MIN_VALUE	Returns the smallest number possible in JavaScript
NEGATIVE_INFINITY	Represents negative infinity (returned on overflow)
NaN	Represents a "Not-a-Number" value
POSITIVE_INFINITY	Represents infinity (returned on overflow)

✓ Number properties belongs to the JavaScript's number object wrapper called **Number**.

✓ These properties can only be accessed as **Number**.MAX_VALUE.

❖ JavaScript Math Object

The JavaScript Math object allows you to perform mathematical tasks on numbers.

• Math Object Methods

Method	Description
abs(x)	Returns the absolute value of x
ceil(x)	Returns the value of x rounded up to its nearest integer
floor(x)	Returns the value of x rounded down to its nearest integer
pow(x, y)	Returns the value of x to the power of y
random()	Returns a random number between 0 and 1
round(x)	Returns the value of x rounded to its nearest integer

❖ JavaScript If...Else Statements

Conditional statements are used to perform different actions based on different conditions.

• Conditional Statements

In JavaScript we have the following conditional statements:

- Use **if** to specify a block of code to be executed, if a specified condition is true.
- Use **else** to specify a block of code to be executed, if the same condition is false.
- Use **else if** to specify a new condition to test, if the first condition is false.
- Use **switch** to specify many alternative blocks of code to be executed.

1- The if Statement

Use the **if** statement to specify a block of JavaScript code to be executed if a condition is true.

Ex:

```
if (hour < 18) {
    greeting = "Good day";
}
```

2- The else Statement

Use the **else** statement to specify a block of code to be executed if the condition is false.

Ex:

```
if (hour < 18) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

3- The else if Statement

Use the **else if** statement to specify a new condition if the first condition is false.

Ex:

```
if (time < 10) {
    greeting = "Good morning";
} else if (time < 20) {
    greeting = "Good day";
} else {
    greeting = "Good evening";
}
```

4- The JavaScript Switch Statement

Use the switch statement to select one of many blocks of code to be executed.

Ex:

```
switch (new Date().getDay()) {
    case 0:
        day = "Sunday";
```

```
            break;
        case 1:
            day = "Monday";
            break;
        case 2:
            day = "Tuesday";
            break;
        case 3:
            day = "Wednesday";
            break;
        case 4:
            day = "Thursday";
            break;
        case 5:
            day = "Friday";
            break;
        case 6:
            day = "Saturday";
    }
```

❖ JavaScript For Loop

Loops can execute a block of code a number of times.

JavaScript supports different kinds of loops:

- **for** - loops through a block of code a number of times
- **for/in** - loops through the properties of an 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

1- <u>The For Loop</u>

The for loop is often the tool you will use when you want to create a loop.

Ex:

```
for (i = 0; i < 5; i++) {
    text += "The number is " + i + "<br>";
}
```

2- The For/In Loop

The JavaScript for/in statement loops through the properties of an object:

Ex:

```
var person = {fname:"John", lname:"Doe", age:25};

var text = "";
var x;
for (x in person) {
    text += person[x];
}
```

3- The While Loop

The while loop loops through a block of code as long as a specified condition is true.

Ex:

```
while (i < 10) {
    text += "The number is " + i;
    i++;
}
```

4- The Do/While Loop

The do/while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

Ex:

```
do {
    text += "The number is " + i;
    i++;
}
while (i < 10);
```