



JavaScript

Tahaluf Training Center 2021



TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.







Day 3

- 1 JS Comments
- 2 JS Operators
- 3 JS Data Types
- 4 How to get an element in JS?
- innerText Vs innerHTML
- 6 Removing HTML Elements
- 7 Removing All or a Specific Class



JS Comments



Single line comments start with //.

Multi-line comments start with /* and end with */.



JS Arithmetic Operators



Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
<mark>%</mark>	Modulus (Division Remainder)
++	Increment
	Decrement



JS Assignment Operators



Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y



JS Comparison Operators



Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator





JS Logical Operators



Operator	Description
&&	logical and
П	logical or
!	logical not



JS Assignment



The = assignment operator assigns a value to a variable.

$$var x = 10;$$



JS Assignment







JavaScript variables can hold many **data types**: numbers, strings, objects and more.





JavaScript has dynamic types. This means that the same variable can be used to hold different data types.





A string (or a text string) is a series of characters like "Dana Kanaan".

Strings are written with quotes. You can use single or double quotes

```
var carName1 = "Volvo XC60";  // Using double quotes
var carName2 = 'Volvo XC60';  // Using single quotes
```





You can use quotes inside a string, as long as they don't match the quotes surrounding the string.





JavaScript has only one type of numbers. Numbers can be written with, or without decimals.

```
var x1 = 27.00;  // Written with decimals
var x2 = 27;  // Written without decimals
```





Booleans can only have two values: true or false.





JavaScript arrays are written with square brackets.

Array items are separated by commas.

The following code declares (creates) an array called cars, containing three items (car names):

```
var cars = ["Audi", "Volvo", "BMW"];
```





JavaScript objects are written with curly braces {}.

Object properties are written as name:value pairs, separated by commas.

```
var person = {
   firstName:"Dana",
   lastName:"Kanaan",
   age:25,
   eyeColor:"brown"
};
```



How to get an element in JS?



Often, with JavaScript, you want to manipulate HTML elements.

To do so, you have to find the elements first. There are several ways to do this:

- Finding HTML elements by id.
- Finding HTML elements by tag name.
- Finding HTML elements by class name.
- Finding HTML elements by CSS selectors.
- Finding HTML elements by HTML object collections.



How to get an element in JS?



How to get an element by name attribute in JavaScript?

```
<input name="email" placeholder="email" />
<input name="password" placeholder="password" />
```

To get the element by name attribute, we need to use the document.querySelector() method by passing [attribute = value] as an argument.



How to get an element in JS?



```
const email =
document.querySelector('[name=email]');
const password =
document.querySelector('[name=password]');
console.log(email);
console.log(password);
```





innerText

The innerText property is used to sets or gets the text from an element.





```
Welcome to happy world
```

Now, if we call a innerText property on p element we will get the text present inside a element.

```
const p = document.getElementById('short');
const text = p.innerText;

// getting text
console.log(text); // Welcome to happy world

// setting new text
p.innerText = "you can see me";
```





innerHTML

The innerHTML property is used to sets or gets the html from an element.

Example:

```
<div id="box">
     <h1>My first post</h1>
     Some text
</div>
```





Now, if we call a innerHTML property on div element we will get the html present inside a <div> element.

```
const div = document.getElementById('box');
// getting Html
const html = div.innerHTML;
console.log(html);
```





Output:



Removing HTML Elements



Removing HTML Elements

We need to use the element.remove() method to remove the html element from a DOM tree.

Example:

Consider we have the following Html.



Removing HTML Elements



Now, we are removing the h1 element from a dom tree by calling the element.remove() method.

```
const h1 =
document.getElementById('tt-1');
h1.remove();
```



Removing All Class Names



Removing all class names

We can remove all class names from an element by setting the className property to an empty string (").



Removing All Class Names



```
<div class="container center red" id="left-div">
    some text
</div>
```

```
const div = document.getElementById('left-div');
div.className = ''; // it removes all class names
from a div
```







Removing a specific class name

To remove a specific class name from an element, we need to use the classList.remove() method by passing a class name as an argument.

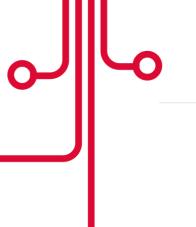


Removing a Specific Class Names



```
const p = document.getElementById('text');
p.classList.remove('box');
```





Day Three Task



On the E-Learning Portal

