



Dynamic HTML

JAVASCRIPT – PART I

Haider M. Habeeb

JAVASCRIPT - JS

✖ JavaScript

- + Is the most popular scripting language on the internet, and works in all major browsers, such as Internet Explorer, Firefox, Chrome, Opera, and Safari.

✖ What You Should Already Know before learning JavaScript?

- + HTML / XHTML

JAVASCRIPT - JS

✖ What is JavaScript?

- + JavaScript was designed to add interactivity to HTML pages
- + JavaScript is a scripting language
- + A scripting language is a lightweight programming language
- + JavaScript is usually embedded directly into HTML pages
- + JavaScript is an interpreted language (means that scripts execute without preliminary compilation)
- + Everyone can use JavaScript without purchasing a license

JAVASCRIPT - JS

✖ Are Java and JavaScript the same?

NO!

✖ Java and JavaScript are two completely different languages in both concept and design!

JAVASCRIPT - JS

✖ What can a JavaScript do?

- + JavaScript gives HTML designers a programming tool
- + JavaScript can put dynamic text into an HTML page
- + JavaScript can react to events
- + JavaScript can read and write HTML elements
- + JavaScript can be used to validate data
- + JavaScript can be used to detect the visitor's browser
- + JavaScript can be used to create cookies

JAVASCRIPT - JS

- ✗ To insert a JavaScript into an HTML page, we use the `<script>` tag.
- ✗ Inside the `<script>` tag we use the `type` attribute to define the scripting language.

```
<html>
<body>
<script type="text/javascript">
...
</script>
</body>
</html>
```

- ✗ So, the `<script type="text/javascript">` and `</script>` tells where the JavaScript starts and ends

JAVASCRIPT - JS

- ✖ This example shows how to use JavaScript to write text on a web page:

```
<html>
<body>
<script type="text/javascript">
document.write("Hello World!");
</script>
</body>
</html>
```

- ✖ The document.write command is a standard JavaScript command for writing output to a page.

JAVASCRIPT - JS

- ✖ The example below shows how to add HTML tags to the JavaScript:

```
<html>  
<body>  
  <script type="text/javascript">  
    document.write("<h1>Hello World!</h1>");  
  </script>  
</body>  
</html>
```


JAVASCRIPT - JS

✖ How to Handle Simple Browsers

- + Browsers that do not support JavaScript, will display JavaScript as page content.
- + To prevent them from doing this, and as a part of the JavaScript standard, the HTML comment tag should be used to "hide" the JavaScript.
- + Just add an HTML comment tag `<!--` before the first JavaScript statement, and a `-->` (end of comment) after the last JavaScript statement.
- + Add two forward slashes at the end of comment line `//` , This prevents JavaScript from executing the `-->` tag.
- + `(//)` is the JavaScript comment symbol.

JAVASCRIPT - JS

✖ Example of how to Handle Simple Browsers

```
<html>  
<body>  
<script type="text/javascript">  
<!--  
document.write("Hello World!");  
//-->  
</script>  
</body>  
</html>
```

JAVASCRIPT - JS

✖ Where to Put the JavaScript?

- + Head section.
- + Inside the body.
- + In both the body and the head section.
- + External JavaScript.

JAVASCRIPT - JS

✖ Scripts in <head>

- + Scripts are placed in functions. When scripts are called, or when an event is triggered,
- + Functions in the head section, this way they are all in one place, and they do not interfere with page content.

```
<html>
<head>
<script type="text/javascript">
function message()
{
alert("This alert box was called
with the onload event");
}
</script>
</head>
<body onload="message()">
</body>
</html>
```

JAVASCRIPT - JS

✖ Scripts in <body>

- + If you don't want your script to be placed inside a function, or if your script should write page content, it should be placed in the body section.

```
<html>
<head>
</head>

<body>
<script type="text/javascript">
document.write("This message is
written by JavaScript");
</script>
</body>

</html>
```

JAVASCRIPT - JS

✖ Scripts in <head> and <body>

- + You can place an unlimited number of scripts in your document, so you can have scripts in both the body and the head section.

```
<html>
<head>
<script type="text/javascript">
function message()
{
alert("This alert box was called with the
onload event");
}
</script>
</head>

<body onload="message()">
<script type="text/javascript">
document.write("This message is written by
JavaScript");
</script>
</body>

</html>
```


JAVASCRIPT - JS

✖ Using an External JavaScript

- + If you want to run the same JavaScript on several pages, without having to write the same script on every page, you can write a JavaScript in an external file.
- + Save the external JavaScript file with a .js file extension.
- + **Note:** The external script cannot contain the `<script></script>` tags!
- + To use the external script, point to the .js file in the "src" attribute of the `<script>` tag:

```
<html>
<head>
<script type="text/javascript" src="myscript.js"></script>
</head>
<body>
</body>
</html>
```

JAVASCRIPT - JS

- ✗ JavaScript is Case Sensitive

- + Watch your capitalization.
- + When you write JavaScript statements, create or call variables, objects and functions.

`var y`

`var Y`

- ✗ The first `y` is different of the second `Y` .

JAVASCRIPT - JS

✖ JavaScript Statements

- + Command to a browser.
- + The purpose of the command is to tell the browser what to do.

```
document.write("University of Babylon");
```

- ✖ Semicolon is optional (according to the JavaScript standard).
- ✖ End of the line is the end of the statement.
- ✖ Using semicolons makes it possible to write multiple statements on one line.

JAVASCRIPT - JS

✖ JavaScript Code

- + JavaScript code (or just JavaScript) is a sequence of JavaScript statements.
- + Each statement is executed by the browser in the sequence they are written.

```
<script type="text/javascript">  
document.write("<h1>This is a heading</h1>");  
document.write("<p>This is a paragraph.</p>");  
document.write("<p>This is another  
paragraph.</p>");  
</script>
```

JAVASCRIPT - JS

✖ JavaScript Blocks

- + JavaScript statements can be grouped together in blocks.
- + Blocks start with a left curly bracket {, and ends with a right curly bracket }.
- + The purpose of a block is to make the sequence of statements execute together.

```
<script type="text/javascript">
{
document.write("<h1>This is a heading</h1>");
document.write("<p>This is a paragraph.</p>");
document.write("<p>This is another paragraph.</p>");
}
</script>
```

- + Blocks is used to group statements together in a function or in a condition.

JAVASCRIPT - JS

✖ JavaScript Comments

- + Comments can be added to explain the JavaScript, or to make the code more readable.

✖ Single line comments

- + start with `//`.

```
<script type="text/javascript">  
// Write a heading  
document.write("<h1>This is a heading</h1>");  
// Write two paragraphs:  
document.write("<p>This is a paragraph.</p>");  
document.write("<p>This is another paragraph.</p>");  
</script>
```


JAVASCRIPT - JS

✖ Multi-Line Comments

+ Multi line comments start with `/*` and end with `*/`.

```
<script type="text/javascript">
```

```
/*
```

The code below will write
one heading and two paragraphs

```
*/
```

```
document.write("<h1>This is a heading</h1>");
```

```
document.write("<p>This is a paragraph.</p>");
```

```
document.write("<p>This is another paragraph.</p>");
```

```
</script>
```

JAVASCRIPT - JS

✖ Using Comments to Prevent Execution

```
<script type="text/javascript">  
//document.write("<h1>This is a heading</h1>");  
document.write("<p>This is a paragraph.</p>");  
document.write("<p>This is another paragraph.</p>");  
</script>
```

```
<script type="text/javascript">  
/*  
document.write("<h1>This is a heading</h1>");  
document.write("<p>This is a paragraph.</p>");  
document.write("<p>This is another paragraph.</p>");  
*/  
</script>
```

✖ Using Comments at the End of a Line

```
<script type="text/javascript">  
document.write("Babylon University"); // Write the name of the university  
document.write(" College of Information Technology"); // Write the name of the college  
</script>
```

JAVASCRIPT - JS

✖ JavaScript Variables

- + As with algebra, JavaScript variables are used to hold values or expressions.

✖ Rules for JavaScript variable names:

- + Variable names are case sensitive (y and Y are two different variables).
- + Variable names must begin with a letter or the underscore character.

```
var x;  
var carname;
```

```
var x=5;  
var carname="Volvo";
```

```
x=5;  
carname="Volvo";
```


JAVASCRIPT - JS

✖ Redeclaring JavaScript Variables

- + If you redeclare a JavaScript variable, it will not lose its original value.

```
var x=5;  
var x;
```

✖ JavaScript Arithmetic

- + As with algebra, you can do arithmetic operations with JavaScript variables:

```
y=x-5;  
z=y+5;
```

JAVASCRIPT - JS

✖ JavaScript Operators

✖ JavaScript Arithmetic Operators

+ Arithmetic operators are used to perform arithmetic between variables and/or values.

Given that $y=5$, the table below explains the arithmetic operators:

Operator	Description	Example	Result
+	Addition	$x=y+2$	$x=7$
-	Subtraction	$x=y-2$	$x=3$
*	Multiplication	$x=y*2$	$x=10$
/	Division	$x=y/2$	$x=2.5$
%	Modulus (division remainder)	$x=y\%2$	$x=1$
++	Increment	$x=++y$	$x=6$
--	Decrement	$x=--y$	$x=4$

JAVASCRIPT - JS

✖ JavaScript Assignment Operators

- + Assignment operators are used to assign values to JavaScript variables.

Given that $x=10$ and $y=5$

Operator	Example	Same As	Result
=	$x=y$		$x=5$
+=	$x+=y$	$x=x+y$	$x=15$
-=	$x-=y$	$x=x-y$	$x=5$
=	$x=y$	$x=x*y$	$x=50$
/=	$x/=y$	$x=x/y$	$x=2$
%=	$x\%=y$	$x=x\%y$	$x=0$

JAVASCRIPT - JS

✖ The + Operator Used on Strings

- + The + operator can also be used to add string variables or text values together.
- + To add two or more string variables together, use the + operator.

```
txt1="What a very";  
txt2="nice day";  
txt3=txt1+txt2;
```

```
txt1="What a very "; //add space at the end of text  
txt2="nice day";  
txt3=txt1+txt2;
```

```
txt1="What a very";  
txt2="nice day";  
txt3=txt1+" "+txt2; //add space as a string
```

JAVASCRIPT - JS

✖ Adding Strings and Numbers

- + The rule is: If you add a number and a string, the result will be a string!

```
x=5+5;  
document.write(x); // 10
```

```
x="5"+"5";  
document.write(x); // "55"
```

```
x=5+"5";  
document.write(x); // "55"
```

```
x="5"+5;  
document.write(x); // "55"
```

JAVASCRIPT - JS

✖ Comparison Operators

- + Comparison operators are used in logical statements to determine equality or difference between variables or values.

Given that `x=5`,

Operator	Description	Example
<code>==</code>	is equal to	<code>x==8</code> is false
<code>===</code>	is exactly equal to (value and type)	<code>x===5</code> is true <code>x==="5"</code> is false
<code>!=</code>	is not equal	<code>x!=8</code> is true
<code>></code>	is greater than	<code>x>8</code> is false

JAVASCRIPT - JS

✖ Logical Operators

- + Logical operators are used to determine the logic between variables or values.

Given that x=6 and y=3

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
	or	(x==5 y==5) is false
!	not	!(x==y) is true

JAVASCRIPT - JS

✖ Conditional Operator

- + JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.

✖ Syntax

- + `variablename=(condition)?value1:value2`

✖ Example

- + `grade=(mark>=50)?"Pass ":"Fail ";`

JAVASCRIPT - JS

✖ Conditional Operator

- + JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.

✖ Syntax

- + `variablename=(condition)?value1:value2`

✖ Example

- + `grade=(mark>=50)?"Pass ":"Fail ";`

THANK

YOU