Javascript

•••

Lecture 1

Eng. Sahar Elsayed

Javascript: History

- Created by Netscape in 1995 as an extension of HTML for Netscape Navigator 2.0.
- JavaScript had as its main function the manipulation of HTML documents and form validation.
- JavaScript was called Mocha. When it first shipped in beta releases, it was officially called LiveScript and finally, when it was released by Sun Microsystems, was baptized with the name by which it is known today.
- Because of the similar names, people confuse JavaScript with Java. Although both have the lexical structure of programming, they are not the same language.
- JavaScript is an interpreted language. It means that it needs an "interpreter". In case of JavaScript, the interpreter is the browser. While Java is compiled language.

Javascript: How to start?

Required tools:

- Code editor (notepad, notepad++, sublime, VS Code)
- *.html file & *.js file (call js file inside the html file)
- Web browser (open html file on the browser & start your js journey)

Javascript: Syntax

- Javascript code can be used in HTML in 2 different ways:

1- Inline script:	**2- Include a js file:**
<script type=“text/javascript”> //javascript code goes here </script>	<script type=“text/javascript”src=“myfile.js”></script>

- The second way is preferred because:
  - It applies separation of concerns and allows code re-use.

Javascript: Syntax (continue)

- Whitespaces (e.g: spaces ,tabs or newlines are ignored).
- Statements can either end with semicolon or newline (semicolon is recommended).
- Javascript is case-sensitive language (i.e. fullName is different from FullName)
- **Javascript is executed in the same order the code appears on the page**
- Comments are made using “//” or “/* */”
    - Comments are some parts in your code you don’t want to parse it or to process and ignore them while start your webpage
- By convention, all javascript variables and function names follow camelCasing (i.e firstName, getSalary()).

Javascript: first script

- Create a project folder on your machine
  - My Project (sample folder name)
- Inside the project folder create (index.html) file
  - My Project/index.html
- Inside the project folder create (scripts.js) file
  - My Project/scripts.js
- Start your html file main tags as the following

Javascript: first script (continue)

- index.html

```
<html>
    <head>
        <script src=’script.js’></script>
    </head>
    <body>
        <!-- your page content will be here -->
    </body>
</html>
```

Javascript: first script (continue)

- Scripts.js
    - alert(‘hello i’m javascript alert message’)

Javascript: browser messages types

1- Alert():

- Alert is a function take an input as a message will be displayed to the user using a pop up box based on the browser api.
- Just an information will be displayed to the user no matter what happened but just show it.
- You will get a message box over your web page with an action “OK”.

```
Ex: var b=1, c=4, result = b+c;
        alert(result); //Will display the result to user in an alert window
```

Javascript: browser messages types

2- Prompt():

- Prompt is a message box will display a guide message to the user to add an input value inside the field input will be contained inside the displayed box.
- Once you add the value and click “OK” it return your value as a string you can save it in a variable.

  Ex: var a = prompt();   alert(a);

- If the **user cancels the prompt window**, **<u>null value</u>** will be assigned to a.

Javascript: browser messages types

3- Confirm():

- Confirm is a message box will display a guide message to the user to check for a specific value or info with two actions:
    - “OK” will return true if clicked.
    - “CANCEL” will return false in case clicked.

  Ex: var confirmMsg = Confirm(“Are you sure to delete this contact?”);

Javascript: browser messages types

4- Console.log():

- Console.log() is a function used for development and debugging purpose only.
- It prints its parameter in the development console.

  Ex: var x=8;

  ```
      console.log(x);      // will print 8 in the console
  ```

Javascript: Variables

- To define a variable:

  var count;

  var name, age;

- You can also define and initialize the variable at the same time:

  var age= 1;

  var name= "Ahmed";

- **Uninitialized variables** have a value of ***undefined***

Javascript: Variables

Variable naming:

- Do NOT start with
  - Numbers (only can be used at the end of name)
  - Any operators (+, -, *, /, ...) (can not be used at any position first, middle or end)
  - Special characters (!, @, #, ...)
    - ($, _) only the accepted special characters
- Do NOT use spaces.
- You cannot use reserved keywords as variable names

  (ex: for, if, while, boolean, ..etc)

Javascript: Variables

- Javascript is an untyped language which means that a variable can hold any datatype.

- Furthermore, the datatype held by the variable can change during the execution of the code.

```
var x = "abc";
var y =44;
y="Hello";
```

Javascript: Data types

- **Number** (includes integers and floats)
  - var x = 5;   //type of x ; it returns Number
- **String**
  - var x = ‘abc’;   type of x;   // it returns String
- **Boolean** (true or false)
  - var x = true;   type of x;   // it returns Boolean
- **Null**
  - var x = null;   type of x;   // it returns Object (special case in js)
- **Undefined**
  - var x;   type of x;   // it returns undefined as it’s uninitialized variable
- **Object** (arrays, objects and null)

Javascript: Arithmetic operators

Operator	**Functionality**
+	Addition (ex: var a=b+1;) Concatenation (ex: var fullName = firstName + lastName;)
-	Subtraction (ex: netSalary = grossSalary-taxes;
*	Multiplication
/	Division
%	Modulus (returns the remainder of division, ex: (var rem = 3%2; //rem will equal 1)
++	Increment by 1 (same as x=x+1) ex: a++;
--	Decrement by 1 (same as a=a-1) ex : a--;

Javascript: Assignment operators

Operator	Functionality
=	Assigns a value to a variable
+=	a+=b is the same as a=a+b
-=	a-=b is the same as a=a-b
/=	a /=b is the same as a=a/b
*=	a *=b is the same as a=a*b
%=	a%=b is the same as a=a%b;

Javascript: Comparison operators

- Comparison operators always **return boolean** values (true or false)

Operator	Functionality
==	Checks for equality ex: a=1 , b =1 , c=2 then a==b is true and b==c is false
<	Less than
<=	Less than or equal
>	Greater than
>=	Greater than or equal
===	Checks for equality and type ex: a=1 , b ="1" a==b is true a===b is false

Javascript: Logical operators

- Logical operators are used to join conditions together

Operator	**Functionality**
&&	Logical AND It's used to check if both operands are true ex: the result of (a && b) will be true only if a =true and b=true, the result would be false otherwise
||	Logical OR It's used to check if any of the operands are true ex: (a || b) will be true if a=true or b=true
!	Logical NOT Reverses the logical state of the operands , i.e. if the condition is true the result will be false Ex: ! (a&&b) will be true if a=false or b= false

Javascript: Logical operators (continue)

&&	True	False
True	True	False
False	False	False

| |	True	False
True	True	False
False	False	False

&&

- Will be fail if at least one of the conditions false
  Event the rest are true

||

- Will be success if at least one of the conditions
  Is true even the rest conditions false

Javascript: If statement

- *If statement* is used to check for a condition and executes code if condition is satisfied.

```
if(5==5) {
        // if condition is true this block of code will be executed
}
```

- Handle false case:

```
if(5==5) {
        // if condition is true, this block of code will be executed
} else {
        // if condition fails, this block of code will be executed
}
```

Javascript: If statement (continue)

- Handle multiple cases:

```
if(5==6) {
        // if condition is true, this block of code will be executed
} else if(5 > 6) {
        // If previous condition true, this block of code will be executed
} else if(5 < 6) {
        // If previous condition true, this block of code will be executed
} else {
        // if all conditions above failed, this block of code will be executed
}
```

Javascript: Switch statement

- It’s a conditional statement used to replace multiple "if ..else if" statements where the condition depends on a single value.

```
Ex: switch (x){
        case  1:
                //when the value is 1 this code will be executed
                break;       // use break to avoid “fall-through” if the condition is true
        case 2:
                //when the value is 2 this code will be executed
                break;
        Default:
                //to be executed if neither of the previous conditions were met
        }
```

Javascript: Ternary operator

- Ternary Operator is a shortcut for if else condition.

```
(condition)?(value_when_true):(value_when_false);
```

- Ex:

```
var speed = (car=="mercedes") ? 120 : 100;
```