JavaScript

NOR ANITA FAIROS BINTI ISMAIL
noranita@utm.my

Lesson 1: Introduction to JavaScript

- Lesson Outline
  - Introduction to JavaScript
    - What is Javacript?
    - What Javascript can do?

Introduction to JavaScript

- was designed to add interactivity to HTML pages.
- Is a scripting language of the Web.
- Case-sensitive
- Interpreted from within a browser, such as Internet Explorer, Firefox, Chrome, Opera, and Safari.
- usually embedded directly into HTML pages.
- It is an object-based language that comes with many built-in objects.
- Is platform-independent as long as the browser is JavaScript-enabled.
- an interpreted language (means that scripts execute without preliminary compilation).
- Everyone can use JavaScript without purchasing a license.

Before you continue you should have a basic understanding of HTML and CSS

What Can JavaScript Do?

- **JavaScript gives HTML designers a programming tool.**
  - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax. Almost anyone can put small “snippets” of code into their HTML pages.

- **JavaScript can put dynamic text into an HTML page.**
  - A JavaScript statement like document.write(“<h1>” + name + “</h1>”) can write a variable text into an HTML page.

- **JavaScript can react to events.**
  - A JavaScript script can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element.

What Can JavaScript Do?

- **JavaScript can read and write the content of an HTML element.**
  - A JavaScript script can read and change the content of an HTML element.
- **JavaScript can be used to validate data.**
  - A JavaScript script can be used to validate form data before it is submitted to a server. This saves the server from extra processing.
- **JavaScript can be used to create cookies.**
  - A JavaScript script can be used to store and retrieve information on the visitor's computer.

Are Java and JavaScript the Same?

- Java and JavaScript is not same.
- They are two different languages in both concept and design.
- Java (developed by Sun Microsystems) .
- Java is a powerful and much more complex programming language in the same category as C and C++.
- But The fundamentals of JavaScript are similar to Java and/or C++

JavaScript's Official Name: **ECMAScript**

- ECMAScript is developed and maintained by the ECMA International organization.
- The language was invented by Brendan Eich at Netscape (with Navigator 2.0) and
- Has appeared in all Netscape and Microsoft browsers since 1996.
- ECMA-262 is the official JavaScript standard.
- The development of ECMA-262 started in 1996, and the first edition of was adopted by the ECMA General Assembly in June 1997.
- The standard was approved as an international ISO (ISO/IEC 16262) standard in 1998.
- The development of the standard is still in progress.

Lesson 2: Basic JavaScript

- Lesson Outline
  - JavaScript Basic
    - How To Put a JavaScript into an HTML Page
    - Where To Put the JavaScript
    - Using an External JavaScript
    - Sample Codes

Lesson 2a.html

Basic HTML Page: Hello World Program

```
<html>
<head>
<title>The HTML  with Hello World </title>
</head>
<body>
<h1> Hello World </h1>
<p> My Name is:</p>
</body>
</html>
```

The JavaScript script is inserted either in the HTML page itself or in a separate file.

How To Put JavaScript into an HTML Page : Hello World Program

Lesson 2b.html

```
<html><head>

 <title>JavaScript  with Hello World </title>

<script type="text/javascript">
// To insert a JavaScript script in an HTML page, use the <script> ... </script> tag.

 document.write('<h1> Hello World </h1>');
//The document.write is a JavaScript command for writing output to a page.
//text between the brackets is surrounded by quotes (' ')
//put semicolon ( ; ) at the end of the statement

</script></head>

<body> </body>

</html>
```

JavaScript Comments

- **Javascript single-line**

  Single line comments start with two forward slashes // before the text //your comments **"JavaScript code "**.

- **Javascript Multiline Comments**

  Multiline comments start with

  /* Comment goes here */

- **Comments at the End of a Line**

  the comment is placed at the end of a code line

  "JavaScript code "//your comments

- **Comments to Prevent Execution**

  a single code line (//) or multiline code (/*….*/)

- JavaScript comments can be added to explain the JavaScript script or to make the code more readable.

- Use to remind your future self of what your code is designed to do.

Where To Put a JavaScript: Scripts in <head>

Lesson 2c.html

Insert JS Script in <head>

```
<html>
<head>
<title>My first JavaScript</title>

<script type="text/javascript">

function message()
{
alert("This alert box was called with the onload event");
}
</script>
</head>

<body onload="message()">
</body>
</html>
```

Where To Put a JavaScript: Scripts in <body>

Lesson 2d.html

Insert JS in <body>

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

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

document.write("This message is written by JavaScript");

</script>
</body>

</html>
```

Where To Put a JavaScript: Scripts in <head> & <body>

Lesson 2e.html

Insert JS in <head> & <body>

```
<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>
```

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

Where To Put a JavaScript: External JS

Lesson 2f.html

```
<html><head>

<script type="text/javascript" src="myjs.js">
/* This example show how to Insert External JS Script */
</script>

</head>

<body>
<input type="button" onClick="message()" value="CLICK ME!">

</body>
</html>
```

myjs.js

```
function message()
{
alert("Hello World");
}
```

Where To Put a JavaScript: External JS

- JavaScript can also be place in external file
- External JavaScript file contain code to be used on several different web pages.
- The file extension is **.js**
- External script cannot contain the

  <script>…..</script> tag
- To use external file, point to the .js file in the “scr” attribute of the <script> tag:

```
<script type="text/javascript“   src=“filename.js">
</script>
```

Simple JavaScript Code: Example 01

Lesson 2g-1.html

```
<html>
<body>
<script type="text/javascript">
document.write("The first statement in written in JavaScript");
document.write("<BR>");
document.write("The second statement in written in JavaScript");
document.write;
</script>
</body>
</html>
```

Simple JavaScript Code: Example 02

Lesson 2g-2.html

```
<html <head>
<title> window prompt</title>
<script type="text/javascript">
var name= "Anita"; // assign value into variable
document.write("Welcome\n"+ name);
</script>
</head>

<body>
</body>
</html>
```

Simple JavaScript Code: Example 03

Lesson 2g-3.html

```
<html <head>
<title> window prompt</title>
<script type="text/javascript">
var name= window.prompt("Enter your name"); // user input

document.write("Welcome\n"+ name +
 "\n to Javascript");
// to display the value entered by the user
</script>
</head>
<body></body>
</html>
```

Window Prompt: to get user input

Simple JavaScript Code: Example 04

Lesson 2g-4.html

```
<html <head>
<title> window prompt</title>
<script type="text/javascript">

var age=window.prompt("What is your age");

age=parseInt(age)+1; //parseInt is a function that converts
  string value to integer value

document.write("Next year, you are" +age+ "years
  old");</script>
</head>
<body></body>
</html>
```

Simple JavaScript Code: Example 05

Lesson 2g-5.html

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

function prompter() {
var reply = prompt("Hey there, good looking stranger! What's your
name?", "")
alert ( "Nice to see you around these parts " + reply + "!")
</script>
</head>

<body>
<input type="button" onclick="prompter()" value="Say my name!">
</body>
</html>
```

Simple JavaScript Code: Example 06

Lesson 2g-6.html

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

function confirmation() {
var answer = confirm("Leave this page?")
                    if (answer) {
                       alert("Bye bye!")
                       window.location = "http://www.google.com/"; }
                        else
                        {  alert("Thanks for sticking around!") }
}
</script> </head>
<body>
<form> <input type="button" onclick="confirmation()" value="Leave This
Page"> </form>
</body>
```

Tutorial 4a

- **Tutorial Instruction:**
- Go to our e-learning and Download the Tutorial 4a.
- Follow the instruction to complete the tutorial.

Tutorial Submission:

- Place all the created files in a compressed/zipped file and submit it on the e-learning site.
- A dedicated submission link will be given for the submission.

Submission Date: 27 APRIL 2014.

Lesson 3: JavaScript Variables

- Lesson Outline
  - JavaScript Variables
  - Declaring (Creating) JavaScript Variables
  - Assigning Values to Undeclared JavaScript Variables
  - Rules when naming variables

Identifiers /variables in JavaScript

- A variables are used to hold values or expressions.
- A variable can have a short name, like *x*, or a more descriptive name, like *carname*.
- case sensitive. For example, *Name* and *name* are two different variables.
- No type!
- Can change type during execution
- Use double quote for character and string variable
- Cannot use reserve word for variable name!

How to create variables

- You can declare JavaScript variables with the **var** statement:

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

- After the declaration, the variables are empty.
- You can assign values to the variables with the statement:

```
var x=5;
var carname="Volvo";  //for string
```

- After the execution of the preceding statements, the variable x will hold the value 5,and carname will hold the value Volvo.

Rules to follow when naming variables

- The name must begin with
  - a letter or
  - an underscore or
  - a dollar sign ($)
- The rest of the name is made up of letters, numbers, or underscores or dollar sign.
- Do not use the reserved words in JavaScript.
- Avoid placing any spaces or punctuation in the name.

Variables in JavaScript

VALID	INVALID
mynumber	my number
student_no	student.no
male	*male
year2001	2001

How to name variables

- The first character must be a letter, an (_) underscore, or a ($) dollar sign:

```
var total;
var _total;
var $total;
var 5total;
```

✓ ✓ ✓ ✗ cannot start with a number

How to name variables

- Each character after the first character can be a letter, an (_) underscore, a ($) dollar sign, or a number:

```
var total;
var to_tal;
var total$;
var total5;
```

How to name variables

- Spaces and special characters other than (_) and $ are not allowed anywhere:

var total; ✓

var to tal; ✗ no spaces allowed

var total#; ✗ #character not allowed

var total£; ✗ £ character not allowed

Reserved word

- The following are reserved words in JavaScript. They cannot be used as JavaScript variables, functions, methods, loop labels, or any object names.

abstract	else	instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try
const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

Lesson 4: JavaScript Operators

- Lesson Outline
  - JavaScript Arithmetic Operators
  - JavaScript Assignment Operators
  - The + Operator Used on Strings
  - Adding Strings and Numbers
  - Comparison Operators
  - Logical Operators
  - Conditional Operators

What is an operator?

- Simple answer can be given using expression

4 + 5 is equal to 9.

- Here 4 and 5 are called **operands** and + is called **operator**.
- JavaScript language supports following type of operators:
  - Arithmetic Operators
  - Assignment Operators
  - Comparison Operators
  - Logical Operators
  - Conditional Operators

JavaScript Arithmetic Operators

- Arithmetic operators are used to perform arithmetic between variables and/or values.
- Assume variable A holds 10 and variable B holds 20 then:

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiply both operands	A * B will give 200
/	Divide numerator by denumerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator, increases integer value by one	A++ will give 11
--	Decrement operator, decreases integer value by one	A-- will give 9

Lesson 4a.html

example

Given that $y = 5$, the following table 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 Assignment Operators

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

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand	C = A + B will assigne value of A + B into C
+=	Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand	C += A is equivalent to C = C + A
-=	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand	C -= A is equivalent to C = C - A
*=	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand	C *= A is equivalent to C = C * A
/=	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand	C /= A is equivalent to C = C / A
%=	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand	C %= A is equivalent to C = C % A

Lesson 4b.html

example

Given that $x = 10$ and $y = 5$, the following table explains the assignment operators:

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

The + Operator Used on Strings

Lesson 4c-1.html
Lesson 4c-2.html

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

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

Lesson 4d.html

Adding Strings and Numbers

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http:/
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5  <title>Lesson 4d: Adding Strings and Numbers</title>
6  <script type="text/javascript">
7
8  x=5+5;
9  document.write(x);//print number
10 document.write("<br />");
11
12 x="5"+"5";
13 document.write(x); //print string
14 document.write("<br />");
15
16 x=5+"5";
17 document.write(x);
18 document.write("<br />");
19
20 x=x+2;
21 document.write(x);
22 document.write("<br />");
23 </script>
24
25 <p>The rule is: If you add a number and a string, the result
26 will be a string.</p>
27
28 </head>
```

The Comparison Operators

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

Assume variable A holds 10 and variable B holds 20 then:

Operator	Description	Example
==	Checks if the value of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
!=	Checks if the value of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

Lesson 4e.html

example

Given that $x = 5$, the following table explains the comparison operators:

Operator	Description	Example
==	is equal to value...is equal to value	x == 8 is false
===	is exactly equal to value and type	x === 5 is true x ===“5” is false
!=	is not equal	x! = 8 is true
>	is greater than	x > 8 is false
<	is less than	x < 8 is true
>=	is greater than or equal to	x >= 8 is false
<=	is less than or equal to	x <= 8 is true

Lesson 4f.html

The Logical Operators

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

Assume variable A holds 10 and variable B holds 20 then:

Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non zero then then condition becomes true.	(A && B) is true.
||	Called Logical OR Operator. If any of the two operands are non zero then then condition becomes true.	(A || B) is true.
!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false.	!(A && B) is false.

Example :

Given that $x = 6$ and $y = 3$, the following table explains the logical operators:

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

The Conditional Operator (? :)

- JavaScript also contains a conditional operator that assigns a value to a variable based on some condition.
- This first evaluates an expression for a true or false value and then execute one of the two given statements depending upon the result of the evaluation.
- The syntax is:

Operator	Description	Example
? :	Conditional Expression	If Condition is true ? Then value X : Otherwise value Y

variablename=(condition)?value1:value2

Lesson 4g.html

example

greeting=(visitor=="PRES")?"Dear President ":"Dear ";

- If the variable *visitor* has the value of "PRES", then the variable *greeting* will be assigned the value "Dear President " else it will be assigned "Dear".

Lesson 5: JavaScript Conditional Statements

- Lesson Outline
  - Conditional Statements
  - if Statement
  - if...else Statement
  - if...else if...else Statement

Conditional Statements

- Conditional statements are used to perform different actions based on different conditions.
- JavaScript has the following conditional statements:
  - **if statement.** Use this statement to execute some code only if a specified condition is true.
  - **if...else statement.** Use this statement to execute some code if the condition is true and another code if the condition is false
  - .
  - **if...else if....else statement.** Use this statement to select one of many blocks of code to be executed.
  - **switch statement.** Use this statement to select one of many blocks of code tobe executed.

Lesson 5a.html

if Statement

- Use the if statement to execute some code only if a specified condition is true.
- The syntax is as follows:

```
if (condition)
{
code to be executed if
condition is true
}
```

```
<script type="text/javascript">
<!--
var age = 20;
if( age > 18 ){
   document.write("<b>Qualifies for driving</b>");
}
//-->
</script>
```

Lesson 5b.html

if...else Statement

- Use the if....else statement to execute some code if a condition is true and another code if the condition is not true.
- The syntax is as follows:

```
if (condition)
{
code to be executed if
condition is true
}
else
{
code to be executed if
condition is not true
}
```

```
<script type="text/javascript">
<!--
var age = 15;
if( age > 18 ){
   document.write("<b>Qualifies for driving</b>");
}else{
   document.write("<b>Does not qualify for driving</b>");
}
//-->
</script>
```

Lesson 5c.html

if....else if...else statement

- Use the if....else if...else statement to select one of several blocks of code to be executed. The syntax is as follows:

```
if (expression 1){
Statement(s) to be executed if expression 1 is true }
else if (expression 2){
Statement(s) to be executed if expression 2 is true }
else if (expression 3){
Statement(s) to be executed if expression 3 is true }
else {
Statement(s) to be executed if no expression is true }
```

Switch statement

Lesson 5d.html
(Lesson 4b-5:html)

- Use the switch statement to select one of many blocks of code to be executed. The syntax is as follows:

```
switch (expression)
{
case condition 1: statement(s) break;
case condition 2: statement(s) break;
.........
case condition n: statement(s) break;
default: statement(s)
}
```

- The **default** case is executed if none of the cases can be matches the value of Expression

Lesson 6: JavaScript Counting and Looping

- Lesson Outline
  - The for Loop
  - The while Loop
  - The do...while Loop
  - Loop Control

JavaScript Loops

- Loops execute a block of code a specified number of times or while a specified condition is true.
- Often when you write code, you want the same block of code to run over and over again in a row. Instead of adding several almost equal lines in a script, you can use loops to perform a task like this.

- In JavaScript, there are **two kinds** of loops:
  - **for.** Loops through a block of code a specified number of times
  - **while.** Loops through a block of code while a specified condition is true

Lesson 6a.html

The for Loop

- The for loop is used when you know in advance how many times the script should run. The syntax is as follows:

```
for (var=startvalue;  var<=endvalue;  var=var+increment)
{
code to be executed statement
}
```

or

for(*Start***;** *Condition***;** *Expression***)** *Statement*

Lesson 6b.html

The while Loop

- The while loop through a block of code a specified number of times or while a specified condition is true. The syntax is as follows:

```
while (var<=endvalue) //condition
{
code to be executed
}
```

- In the case of the while loop, the condition is checked first, so if false, the block will not be executed.

The do...while Loop

Lesson 6c.html

- The do...while loop is a variant of the while loop. This loop will execute the block of code *once*, and then it will repeat the loop as long as the specified condition is true. The syntax is as follows:

```
do
{
code to be executed
}
while (var<=endvalue); //condition
```

- In the do…while loop, the condition is checked ***after*** the block is executed; therefore the block is always executed at least once.

Additional: Loop Control Statements

Lesson 6d.html
Lesson 6e.html
Lesson 6f.html

The break Statement

- The break statement will terminate execution of the loop and continue executing.

The continue Statement

- The continue statement will terminate the current iteration and restart the loop.

JavaScript for...in Statement

- The for...in statement loops through the elements of an array or through the properties of an object.

Tutorial 4b

- Download from our e-learning and follow the instruction and try to understand.
- Submission Date: **27 APRIL 2014.**

Lesson 7: JavaScript Function

- Lesson Outline
  - How to Define a Function
  - JavaScript Function Examples
  - The return Statement

Javascript Functions

- Functions in Javascript behave similar to numerous programming languages (C, C++, PHP, etc).
- Put in head section or external
- Variables inside a function is **local**
- Use **return** to return value and exiting the function (return without value) without finishing

Javascript Functions

Involves two steps:

- Define: *to define what processes should be taken*
- Call/Invoke: *to execute the functions*
- Syntax of function definition:

```
function function_name (param1, param2, .., param_n)
                        //parameters are optional
{
  //function's code goes here

    return value_or_object; //optional
}
```

Lesson 7a.html

Function Declaration & Calling

Example: Calculate The Discount

```
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5   <title>Lesson 7a: Function</title>
6   <script type="text/javascript">
7   //<Script Language="JavaScript">
8   function calculateTheDiscount()
9   {
10    var itemPrice = 120.55;
11    var discountRate = 0.20; // = 20%
12    var discountAmount = itemPrice * discountRate;
13    var netPrice = itemPrice - discountAmount;
14    var sentence1 = "Item Price: $" + itemPrice + "<br>";
15    var sentence2 = "Discount: $" + discountAmount + "<br>";
16    var sentence3 = "Net Price: $" + netPrice + "<br>";
17
18    document.write(sentence1);
19    document.write(sentence2);
20    document.write(sentence3);
21  }
22  </Script>
23  </head>
24
25  <body>
26  <!-- ========= call function jika guna javascript ====================== -->
27  <Script Language="JavaScript">
28  calculateTheDiscount();
29  </Script>
30  </body>
31  </html>
```

Function Declaration & Calling

Lesson 7b.html

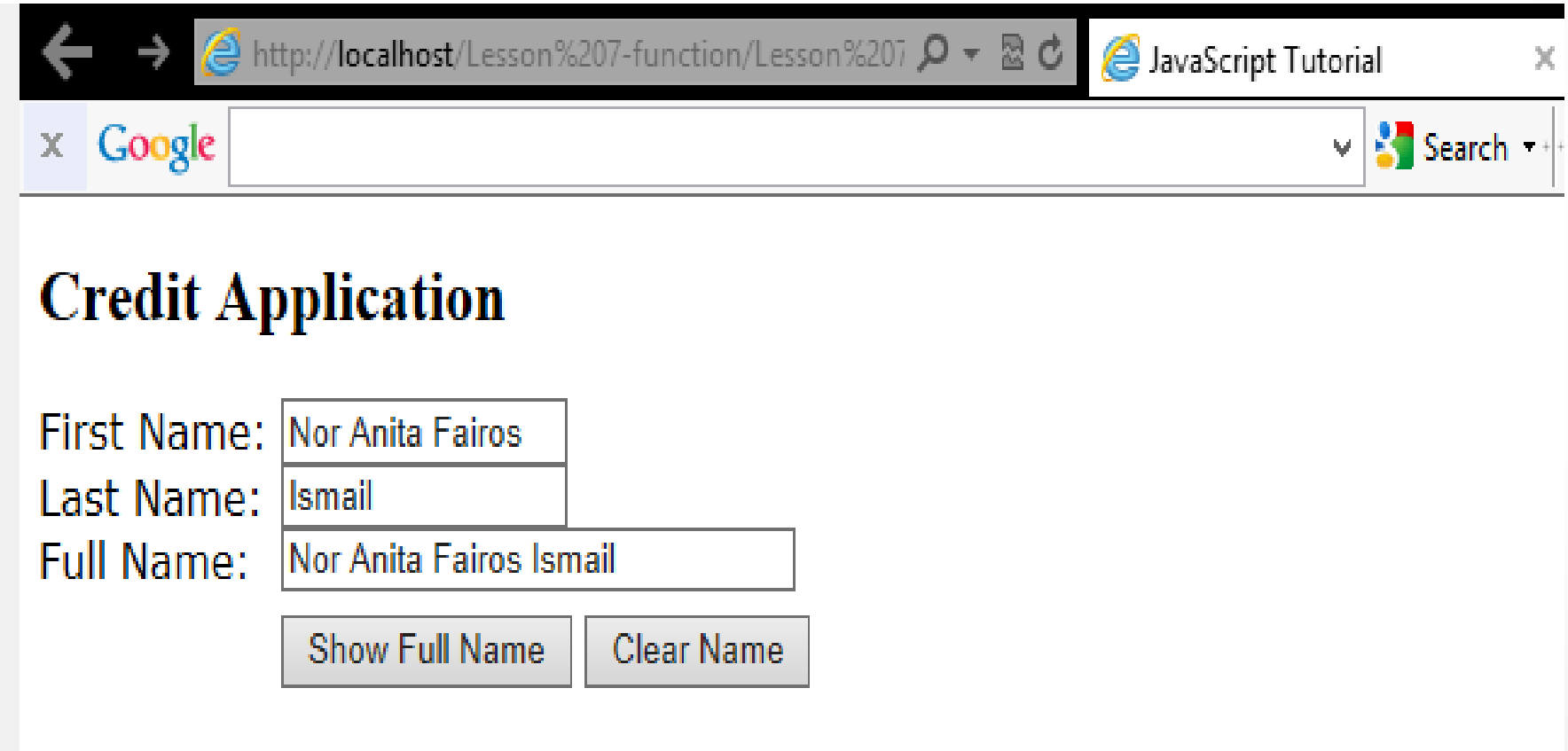
Example: Return Function

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitio
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; ch
5   <title>Retur Functions</title>
6   <Script Language="JavaScript">
7   function rectangleArea(length, height)
8   {
9     var area;
10    area = length * height * 3.14159;
11    return area;
12  }
13  function displayArea()
14  {
15    var l, h, rectArea;
16    l = 52.05;
17    h = 46.55;
18    rectArea = rectangleArea(l, h);
19    document.write("The area of the is ", rectArea);
20  }
21  </Script>
22
23  <Script Language="JavaScript">
24    displayArea();
25  </Script>
26  </body>
27  </html>
```

Lesson 7c.html

Function: More example

Example: Get your full name



Assignment 1: Individual

- In this assignment, you need to create web-based application that simulates a dry-cleaning store services.
- Write the HTML codes to display the following form for Astana Cleaning Services.
- Write your code in single page (**no need JS external file**).
- Save as your file as **Assignment1.html**
- Submit your Individual **Assignment1.html** via our e-learning and select **"Submission Individual Assignment -1"**
- Submission Date: **7 MAY 2015 (Thursday)**

- BEFORE that, You also need to create another 2 pages of html file.
  - Main page (save as index.html)
  - Profile page (save as profile.html)
  - Assignment1.html
  - Must be apply CSS at least **2 style** of template design.

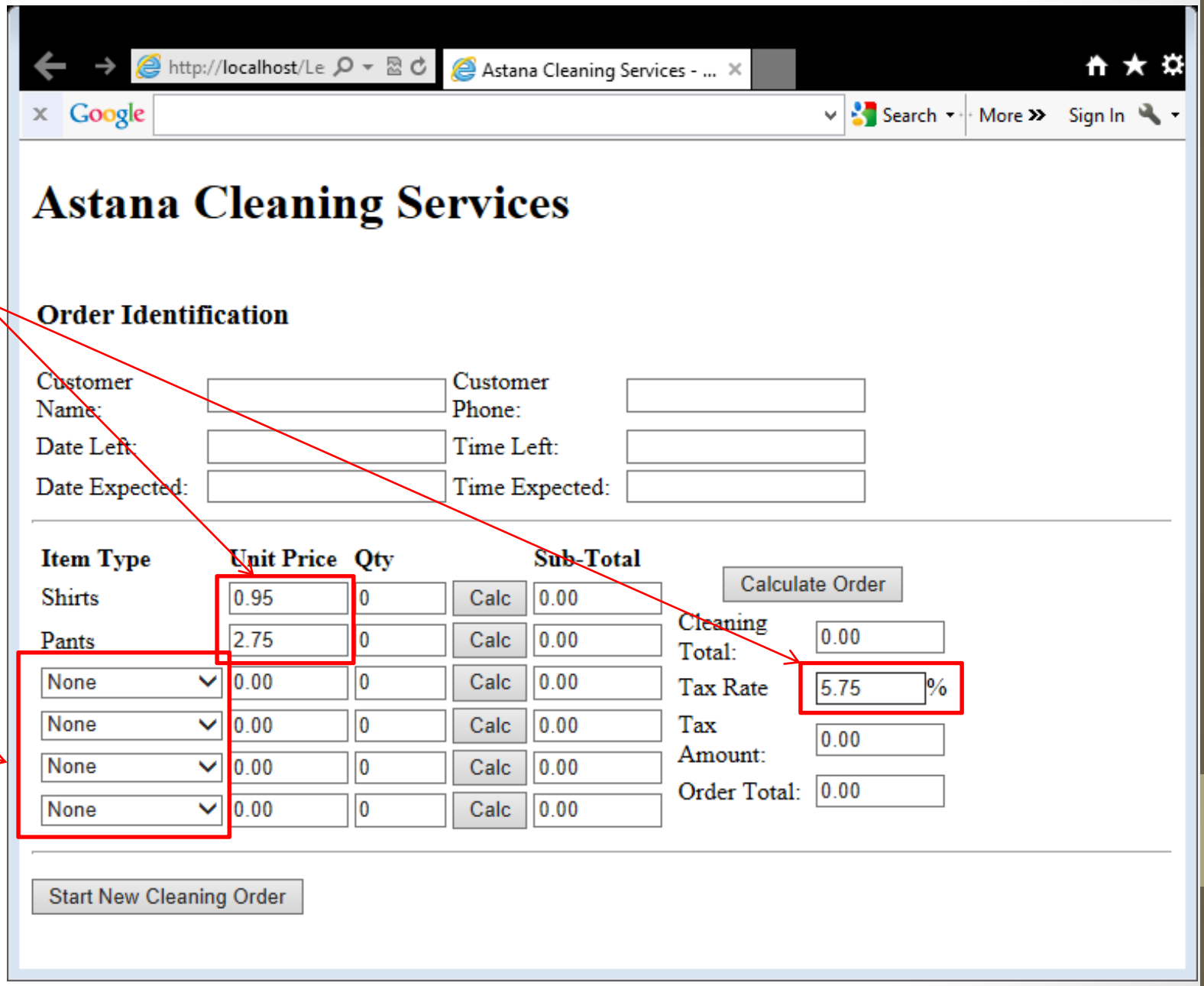
The web-based application form

Once you run the application:
By default the unit price of these items must be:

Shirt = 0.95
Pants =2.75
Tax Rate=5.75

But you can change its later with your own value.

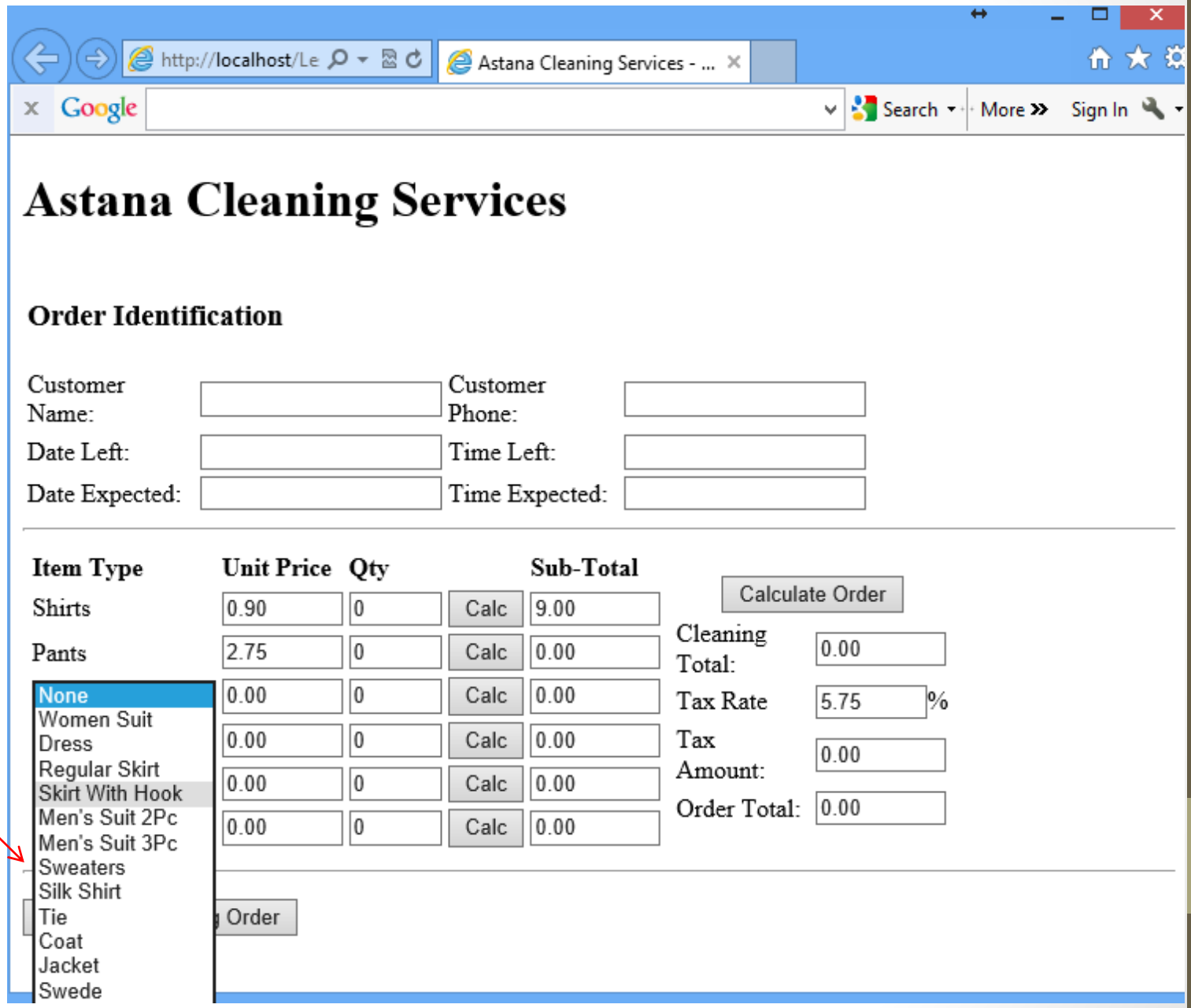
Your have 4 selection Box for 4 items type.



The web-based application form

This is a list of item display in each Selection box.

Remember, you have 4 items type here



Example data entry

Calculate sub total for each item type

Calc | 4.50

Calculate all the cleaning total price including taxAmount; OrderTotal;

Calculate Order

Clear form

Start New Cleaning Order

http://localhost/Le | Astana Cleaning Services - ...

Google | Search | More | Sign In

Astana Cleaning Services

Order Identification

Customer Name:	NOR ANITA	Customer Phone:	013-3314366
Date Left:	2/01/2014	Time Left:	8:00 AM
Date Expected:	5/01/2014	Time Expected:	10:00 AM

Item Type	Unit Price	Qty		Sub-Total
Shirts	0.90	5	Calc	4.50
Pants	2.75	5	Calc	13.75
Women Suit	3.00	3	Calc	9.00
Dress	7.00	1	Calc	7.00
Silk Shirt	5.00	1	Calc	5.00
Sweaters	10.00	1	Calc	10.00

Calculate Order

Cleaning Total:	49.25
Tax Rate	7.00 %
Tax Amount:	3.45
Order Total:	52.70

Start New Cleaning Order

Additional items

Astana Cleaning Services - Cleaning

file:///C:/Users/virtualspace/Desktop/SCSV1223%20Sesi%20201415(2)%20-%20\

Apps | Save to Mendeley | Kursus HCD UTM 20... | fisheries Logbooks | Adobe After Effects ... | Humar

Astana Cleaning Services

Customer Order Identification

Customer Name *: | Customer Phone *:

Date Left: 4/27/2015 | Time Left: 9:29 AM

Date Expected: | Time Expected:

E-mail Address *: | Current Date

Current Date and Time will be display in the text box, when user click button Current Date

Customer Order Details

Item Type	Unit Price	Qty		Sub-Total
Shirts	0.95	0	Calc	0.00
Pants	2.75	0	Calc	0.00
None	0.00	0	Calc	0.00
None	0.00	0	Calc	0.00
None	0.00	0	Calc	0.00
None	0.00	0	Calc	0.00

Calculate Order

Cleaning Total: 0.00

Tax Rate 5.75 %

Tax Amount: 0.00

Order Total: 0.00

Start New Cleaning Order | Check Field

** this field cannot be empty*

Please Enter a Value

OK

Not a Valid Email

OK

Validate the input
Customer Name, Customer Phone and E-mail Address is required. If empty, display the message "Please Enter a Value".
E-mail must follow the email format. If wrong, display the message "Not a Valid Email"

Hints

- You should has these function to calculate:

sub total for each item type

- function processShirts()

  subTotal = unitPrice * quantity; //calculate sub total for shirts

- function processPants() //calculate sub total for Pants
- function processItem1() //calculate sub total for item type in selection Box 1.
- function processItem2() //calculate sub total for item type in selection Box 2.
- function processItem3() //calculate sub total for item type in selection Box 3.
- function processItem4() //calculate sub total for item type in selection Box 4.

Tax amount charge four your

- function calculateTaxAmount()

  tax amount = cleaning total * tax / 100;

Calculate all the cleaning total price

- function processOrder()

  cleaningTotal = shirtsTotal + pantsTotal + item1Total +
  item2Total + item3Total + item4Total;

  OrderTotal = cleaningTotal + taxAmount;

Lesson 8: JavaScript Objects Intro

- Lesson Outline
  - Object-Oriented Programming
  - Properties
  - Methods

Object-Oriented Programming

- JavaScript as a programming language has strong object-oriented capabilities.
- An Object-Oriented (OOL) language enables you to model data using objects consisting of **properties** and **methods** that operate on those properties.
- We start by looking at the **built-in** JavaScript objects and how they are used.
- Note that :
  - An object is just a special kind of data or ‘thing’.. It is usually visible on the screen. Examples of:
    - built-in objects include document, window, navigator.
    - user-defined objects include forms.
  - An object has properties and methods.

Lesson 8a.html

Properties

- Properties are the values associated with an object.
- In the following example, we use the length property of the String object to return the number of characters in a string:

```
<script type="text/javascript">
var txt="Hello World!";
document.write(txt.length);
</script>
```

The output of the previous code will be

12

Lesson 8b.html

Methods

- Methods are the actions that can be performed on objects.
- In the following example, we use the **toUpperCase() method** of the String object to display a text in uppercase letters:
- Example String Object:

```
<script type="text/javascript">
var str="Hello world!";
document.write(str.toUpperCase());
</script>
```

The output of the previous code will be

HELLO WORLD!

User-Defined Objects

- All user-defined objects and built-in objects are descendants of an object called Object.

- **The *new* Operator:**
- The ***new*** operator is used to create an instance of an object. To create an object, the ***new*** operator is followed by the constructor method.
- The constructor methods are Object(), Array(), and Date(). These constructors are built-in JavaScript functions.

```
var employee = new Object();
var books = new Array("C++", "Perl", "Java");
var day = new Date("August 15, 1947");
```

The *Object()* Constructor

- A constructor is a function that creates and initializes an object.
- JavaScript provides a special constructor function called *Object()* to build the object.
- The return value of the Object() constructor is assigned to a variable.
- The variable contains a reference to the new object. The properties assigned to the object are not variables and are not defined with the var keyword.

JavaScript Array

Create an Array

Lesson 8c-1.html

- An array can be defined in three ways.
- The following code creates an Array object called myCars:

1.

```
var myCars=new Array();
    // create a new array with no elements
    // new Array(n); will create a new array of length n
myCars[0]="Saab";
myCars[1]="Volvo";
myCars[2]="BMW";
```

2.

```
var myCars=new Array("Saab","Volvo","BMW");
// create a new array with the specified elements
```

3.

```
var myCars=["Saab","Volvo","BMW"];
//examples 2 & 3 are functionally equivalent
```

JavaScript Array

Access an Array

- You can refer to a particular element in an array by referring to the name of the array and the index number. The index number starts at 0.
- The following code line:

```
document.write(myCars[0]);
```

results in the following output:

Saab

JavaScript Array

Modify Values in an Array

- To modify a value in an existing array, just specify a new value for the element at the given index.

```
myCars[0]="Opel"; // overwrite the current value of myCars[0]
```

- Now, the following code line:

```
document.write(myCars[0]);
```

results in the following output:

Opel

JavaScript Array

Sorting in an Array

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
<script type="text/javascript">
var myCars=new Array(); //step1:create object with contructor new operator
// create a new array with no elements
// new Array(n); will create a new array of length n
myCars[0]="Saab";
myCars[1]="Volvo";
myCars[2]="BMW";
//myCars[0]="Opel"; // step 3: overwrite the current value of myCars[0]

document.write(myCars[0]); //step2: Access an Array
/*You can refer to a particular element in an array by referring
to the name of the array and the index number. The index number starts at 0.*/

document.write("<br>"); //step 4: sorting array
document.write("Before Sort:"+ myCars);
document.write("<br>");
document.write("After Sort:"+ myCars.sort());
//using sort() method to sorting an array
</script>
</head>
<body>
</body>
</html>
```

JavaScript Array

Lesson 8c-2.html

write the values to the output.

```
1  <html>
2  <body>
3  <script type="text/javascript">
4  /* Examples
5  The following example demonstrates how to create an array, assign values
   to it, and write the values to the output.
6  */
7  var mycars = new Array();
8  mycars[0] = "Saab";
9  mycars[1] = "Volvo";
10 mycars[2] = "BMW";
11
12 for (i=0;i<mycars.length;i++)
13 {
14 document.write(mycars[i] + "<br />");
15 }
16 </script>
17 </body>
18 </html>
```

Lesson 8c-3.html

Try: Others Array Method

- myCars.slice(0,1)
- myCars.join()
- myCars.pop()
- myCars.push("Myvi")
- myCars.sort()
- myCars.shift()
- myCars.unshift("Fords","Audi")

See the result and try to understand

Lesson 8c-4.html

Try: sort() example

Demonstrates how to **sort numerically** and ascending.

```
<html>
<body>
<script type="text/javascript">
function sortNumber(a, b)
{
return a - b;
}
var n = ["10", "5", "40", "25", "100", "1"];
document.write(n.sort(sortNumber));
</script>
</body>
</html>
```

Now how to sort numerically and descending?
return b - a;

Lesson 8c-5.html

Nested Array

```
<html>
<body>
<script type="text/javascript">
var myArray = new Array(3)

//create the second dimension with fix column(3)
for (i=0; i<3; i++)
        myArray[i]=new Array(3);

//fill in the array
for (i=0; i<3; i++)
        for (j=0; j<3; j++)
        myArray[i][j]=j;

//print the array
for (i=0; i<3; i++)
{       for (j=0; j<3; j++)
                document.write(myArray[i][j]+",");
                document.write("<br/>"); }
</script>
</body> </html>
```

Example how to join two arrays using concat

Lesson 8c-6.html

```
<html>
<body>
<script type="text/javascript">
var parents = ["Jani", "Tove"];
var children = ["Cecilie", "Lone"];
var family = parents.concat(children);
document.write(family);
</script>
</body>
</html>
```

example how to join three arrays using concat

Lesson 8c-7.html

```
<html>
<body>
<script type="text/javascript">
var parents = ["Jani", "Tove"];
var brothers = ["Stale", "Kai Jim", "Borge"];
var children = ["Cecilie", "Lone"];
var family = parents.concat(brothers, children);
document.write(family);
</script>
</body>
</html>
```

Lesson 8d-1.html

This example demonstrates how to create an User-Define Objects: book

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">
var book = new Object();   // Create the object
    book.subject = "Perl"; // Assign properties to the object
    book.author  = "Mohtashim";
</script>
</head>
<body>
<script type="text/javascript">
   document.write("Book name is : " + book.subject + "<br>");
   document.write("Book author is : " + book.author + "<br>");
</script>
</body>
</html>
```

Creating Your Own Objects

Create a Direct Instance of an Object

- There are two ways to create a new object:
- You can create a direct instance of an object (directly using **variable declaration)** , or ….
- you can create a template of an object (from **an existing classes**).

Lesson 8d-1.html

This example demonstrates how to create an User-Define Object: book

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">
var book = new Object();   // Create the object
    book.subject = "Perl"; // Assign properties to the object
    book.author  = "Mohtashim";
</script>
</head>
<body>
<script type="text/javascript">
   document.write("Book name is : " + book.subject + "<br>");
   document.write("Book author is : " + book.author + "<br>");
</script>
</body>
</html>
```

Create a Direct Instance of an Object

The following code creates an instance of an object and adds two properties to it.

This example demonstrates how to create an User-Define Object: book

Lesson 8d-2.html

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">
//This example demonstrates how to create an object in javascript

var book = {"subject":"Perl",
            "author":"Mohtashim"}
</script>
</head>
<body>
<script type="text/javascript">
   document.write("Book name is : " + book.subject + "<br>"); //access object properties
   document.write("Book author is : " + book.author + "<br>");
</script>
</body>
</html>
```

// You also can write like this way

Creating Your Own

Lesson 8e.html

Objects : create a Template of an Object

- The template defines the structure of an object so that you can more easily create multiple instances of that object.

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">
function book(title, author){
    this.title = title;
    this.author  = author;
}
</script>
</head>
<body>
<script type="text/javascript">
   var myBook = new book("Perl", "Mohtashim");
   document.write("Book title is : " + myBook.title + "<br>");
   document.write("Book author is : " + myBook.author + "<br>");
</script>
</body>
</html>
```

Creating Your Own

Objects : create a Template of an Object

- The example demonstrates how to create an object **with a User-Defined Function.**
- Here ***this*** keyword is used to refer to the object that has been passed to a function.

Creating Your Own Objects

create a Template of an Object

- After you have the template, you can create new instances of the object:

```
var myBook2 = new book("PHP", "Stephen");
document.write("Book title is : " + myBook2.title + "<br>");
document.write("Book author is : " + myBook2.author + "<br>");
```

- you also can create new instances of the object, with null:

```
var myBook3 = new book(null);
    myBook3.title="OpenGL"; //now lets assign the value of the properties
    myBook3.author="A.Saleem";
document.write("Book title is : " + myBook3.title + "<br>");
document.write("Book author is : " + myBook3.author + "<br>");
```

Creating Your Own

Lesson 8e.html

Objects : Defining Methods for an Object

- The previous examples demonstrate how the constructor creates the object and assigns properties.
- But we need to complete the definition of an object by assigning methods to it.
- Here is a simple example to show how to add a function along with an object.
- Note that methods are just functions attached to objects.
- Then you will have to write the **addPrice ()** function

Creating Your Own Objects : Defining Methods for an Object

Lesson 8e.html

```
<html>
<head>
<title>User-defined objects</title>
<script type="text/javascript">

// Define a function which will work as a method
function addPrice(amount){
    this.price = amount;
}

function book(title, author){
    this.title = title;
    this.author  = author;
    this.addPrice = addPrice; // Assign that method as property.
}

</script>
</head>
<body>
<script type="text/javascript">
   var myBook = new book("Perl", "Mohtashim");
   myBook.addPrice(100);
   document.write("Book title is : " + myBook.title + "<br>");
   document.write("Book author is : " + myBook.author + "<br>");
   document.write("Book price is : " + myBook.price + "<br>");
</script>
</body>
</html>
```

Lesson 9: Event Handler

- Lesson Outline
  - Introduction to Events and Event Handlers
  - System Events
  - Mouse Events
  - Data Entry Events

JavaScript Events

- Events are actions that take place in a document, such as clicking on a button or selecting a text.
- Event handlers are JavaScript-related HTML attributes that modify the behavior of a document.
- JavaScript event handlers can be divided into the following categories:
  - **System Events**
  - **Mouse Events**
  - **Data Entry Events**

Lesson 9a.html

System Events

- Events that don't require user interaction. For example the onloading and unloading of a web page. These events are written in the BODY tag.
- **onLoad**
  - __Is activated after the HTML page is completed loaded
- **onUnload**
  - __Is activated when the user exits a HTML page.

System Events:
onLoad , onUnload

Lesson 9a.html

```
<HTML>
<HEAD>
 <TITLE>Lesson 9a: Events Handling Ex 1</TITLE>
 <SCRIPT>
   function welcome()
   { var name=prompt("Please register your name > ","");
     alert('Welcome ' +name+ ' to my world '); }
   function goodbye()
   { alert('Going off so soon? ');
     alert('Please come back again ');  }
 </SCRIPT> </HEAD>
<BODY onLoad="welcome()” onUnload="goodbye()">
   <H2>onUnload Event Handler </H2>
   <HR></BODY>  </HTML>
```

Mouse Events:

onClick

Lesson 9b.html

- Mouse Events will require user interaction in order to be triggered.
- They have to do with mouse movements and mouse clicks.

onClick

__Is activated when the user clicks an object that accepts such an event. For example, clicking on a radio button.

onMouseOver

Occurs when the mouse crosses over an object. For example over a hyperlink or an image.

onMouseOut

__Occurs when the mouse leaves an object.

Mouse Events:

onClick

Lesson 9b.html

```
<html>
<head>
<TITLE>Lesson 9b: Mouse Event onClick</TITLE>
<script type="text/javascript">
<!--
function sayHello() {
  alert("Hello World")
}
//-->
</script>
</head>
<body>
<input type="button" onClick="sayHello()" value="Say Hello" />
</body>
</html>
```

Mouse Events:
onMouseOver

Lesson 9c-1.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
<H2> onMouseOver Event Handler </H2>
<HR>

<A HREF="yourpagename.html"
   onMouseOver="window.status='Check out my cool page' ">Click
   Here to go to a cool site </A>

</BODY>
</HTML>
```

Mouse Events:
onMouseOut

Lesson 9c-2.html
Lesson 9c-3.html

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<H2> onMouseOut Event Handler </H2>
<HR>
<A HREF="yourpagename.htm" onMouseOver="window.status='You are over the
   link!';return true;" onMouseOut="window.status='You are out of the link!
   ';return true;">Bring your mouse over here and out of here!! </A>
</BODY>
</HTML>
```

Data Entry Events

- Data entry events will require user interaction to be triggered. They happen when the user attempts to change or submit the contents of a HTML form.

- **onFocus**

  Occurs when an objects is selected. Can be used only with text, textarea, password and select objects.

- **onBlur**

  Occurs when an **object is no longer in focus**. Applies to text, textarea, password and select objects.

Data Entry Events - Cont'd

- **onChange**
  - Is activated whenever an object has lost focus **and** it's value has been changed. Applies to text, textarea, password and select objects.

- **onSelect**
  - Event occurs when the user highlights text in a text, textarea or password object.

- **onSubmit**
  - Is used with form object. Occurs when a user submits a form.

- **onReset**
  - Counterpart to OnSubmit. Occurs when the user clicks the Reset button.

Data Entry Events : onFocus

Lesson 9d-1.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
  <H2> onFocus Event Handler </H2>
  <HR>

  <FORM>
    <INPUT TYPE= "text" SIZE="30" onFocus="window.status='Get It?' " >
  </FORM>
</BODY>
</HTML>
```

Data Entry Events: onBlur

Lesson 9d-2.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
  <H2> onBlur Event Handler </H2>
  <HR>
  <FORM>
     <INPUT TYPE= "text" SIZE="65" VALUE="Overwrite this piece of text and click on
       somewhere else on this page" onBlur="window.status='Get It?' " >
  </FORM>
</BODY>
</HTML>
```

Data Entry Events: onChange

Lesson 9d-3.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
  <H2> onChange Event Handler </H2>
  <HR>

  <FORM>
     <INPUT TYPE= "text" SIZE="35" VALUE="JavaScript is cool.."
       onChange="alert('You have changed the text !!') " >
  </FORM>
</BODY> </HTML>
```

Data Entry Events: onSelect

Lesson 9d-4.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
  <H2> onSelect Event Handler </H2>
  <HR>

  <FORM>
   <INPUT TYPE= "text" SIZE="35" VALUE="Highlight this text" onSelect="alert('You
       have selected this text !!') " >
  </FORM>
</BODY> </HTML>
```

Data Entry Events: onSubmit

Lesson 9d-5.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
  <H2> onSubmit Event Handler </H2>
  <HR>
  <FORM onSubmit="alert('You have decided to format your harddisk !!')">
  <INPUT TYPE= "submit" VALUE="Check it Out!"  >
  </FORM>
</BODY>
</HTML>
```

Data Entry Events: onReset

Lesson 9d-6.html

```
<HTML>
<HEAD>
</HEAD>

<BODY>
  <H2> onReset Event Handler </H2>
  <HR>
  <FORM onReset="alert('You are going to clear all text you have entered !!!')">
     Enter a text here: <INPUT TYPE="text" size=15 value=" "><p>
                                <INPUT TYPE= "reset" VALUE="Reset Form!" >
  </FORM>
</BODY> </HTML>
```

Lesson 10: JavaScript Popup Boxes

Lesson Outline:

- Popup Boxes
  - Alert Box
  - Confirm Box
  - Prompt Box
- page redirection

Popup Boxes

- JavaScript has three types of popup boxes: alert box, confirm box, and prompt box.

Alert Box

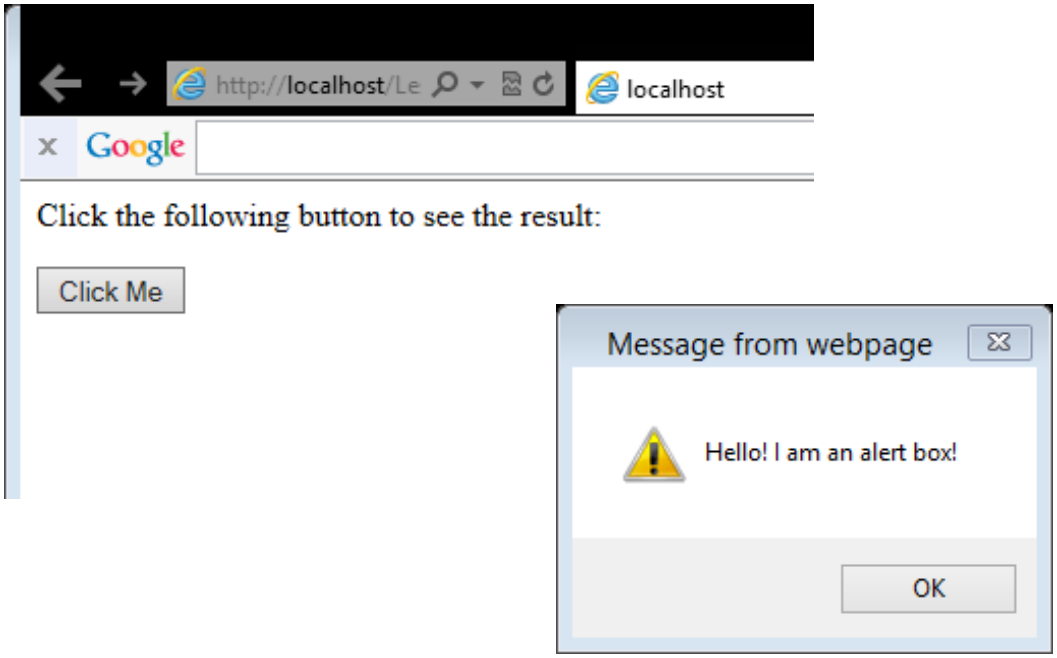
- An alert box is often used when you want to display information to the user.
- When an alert box pops up, the user will have to click OK to proceed.
- The syntax is as follows:

```
alert("sometext");
```

Lesson 10a-1.html

Example alert box

- When you click the button, the alert box is appear



Example
alert box with line breaks

Lesson 10a-2.html

The following example creates an alert box with line breaks: **\n**

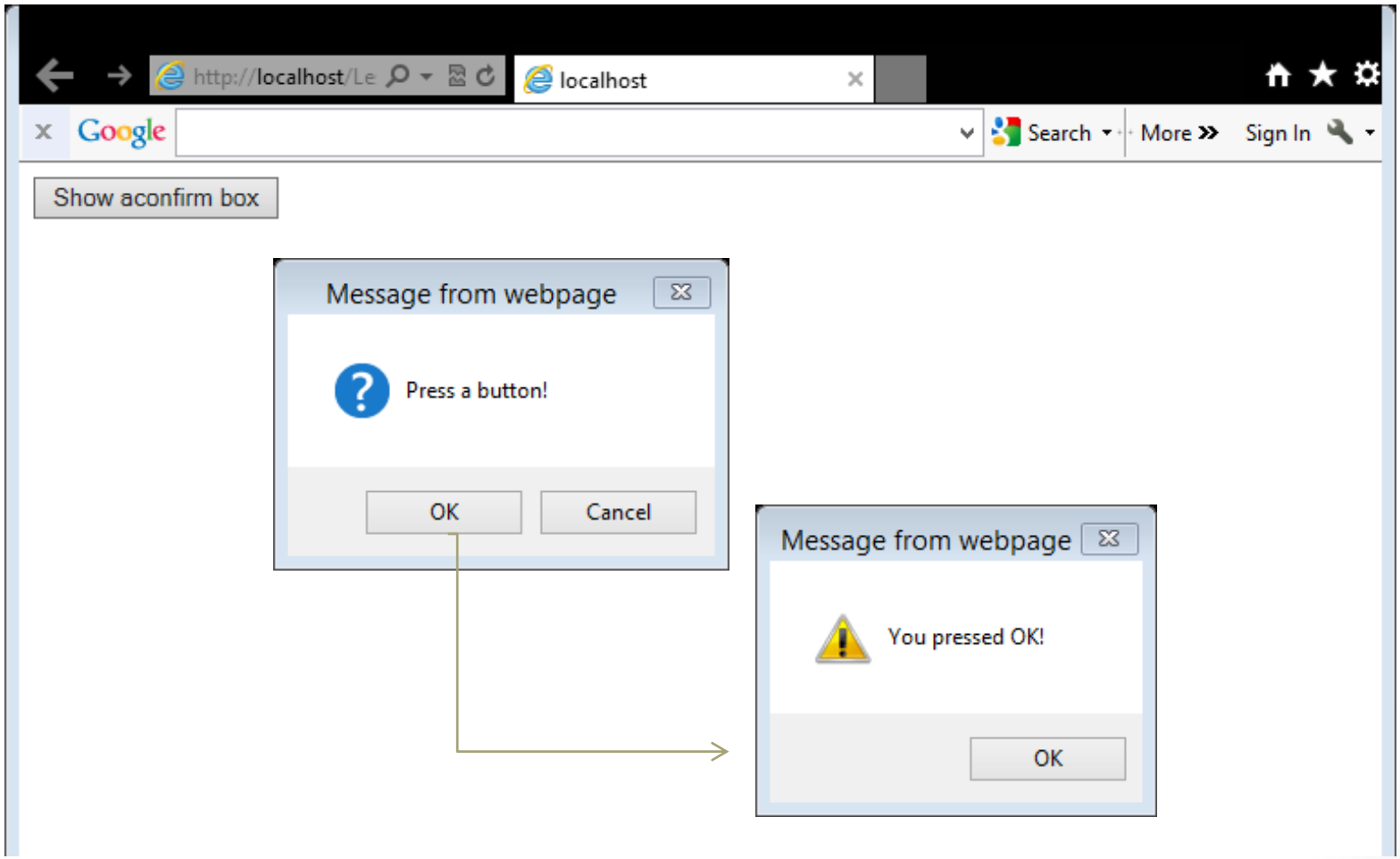
Confirm Box

- A confirm box is often used if you want the user to verify or accept something.
- When a confirm box pops up, the user will have to click either **OK** or **Cancel** to proceed.
- If the user clicks OK, the box returns ***true***. If the user clicks Cancel, the box returns ***false***.
- The syntax is as follows:

```
confirm("sometext");
```

Lesson 10b-1.html

Example Confirm Box



Lesson 10c-1.html

Prompt Box

- A prompt box is often used if you want the user to input a value while on a page or from a page.
- When a prompt box pops up, the user will have to click either **OK** or **Cancel** to proceed after entering an input value.
- If the user clicks **OK**, the box **returns the input value**. If the user clicks Cancel, the box r**eturns null**.
- The syntax is as follows:

```
prompt("sometext","defaultvalue");
```

page redirection

Lesson 10d-1.html
Lesson 10d-2.html

- When you click a URL to reach to a page X but internally you are directed to another page Y that simply happens because of page re-direction.
- To send user to your new website location:
- The syntax is as follows:

window. location="URL your new website to go ";