German Jordanian University الجامعة الألمانية الأردنية

CS355 Web Technologies

Dr. Ismail Hababeh

German-Jordanian University

Lecture 7

Writing JavaScript Function to Validate Data

Write a JavaScript function to validate user's data entry.

```
<script language="JavaScript">
function disp( )
{
// accessing data
 var name=window.document.student.sname.value
// or var
name=window.document.getElementById("stid").val
ue
//checking name
 if(name=""||!isNaN(sname))
       window.alert("sname you entered is invalid")
else
document.write("The name you just entered is : "+ name);
}
</script>
```

//The isNaN() Not-a-Number method returns true if a value is NaN. The isNaN() method converts the value to a number before testing it

JavaScript Function to Validate Data

```
<BODY>

<form name="student">

Enter Student name:

<input type="text" name="sname“ id="stid"

        value="enter" onclick="disp( )">

</form>

</BODY>
```

JavaScript Popup Windows

- Popup Window is a small window that always shown before opening the page.
- Known as Dialog Boxes.
- The purpose of popup box is to write message or accept a response from a user.
- JavaScript provides 3 types of Dialog Boxes:
  - Alert
  - Confirm
  - Prompt

JavaScript Alert Dialog Box

- Alert Dialog Box is used to send or write warning messages to end user.
- Alert Dialog Box is created by alert method of window object.
- Alert method syntax:

```
window.alert ("message");
```

JavaScript Alert Dialog Box

- In the previous example, when a user enter invalid name, an Alert Dialog Box is used to write a warning message as follows.

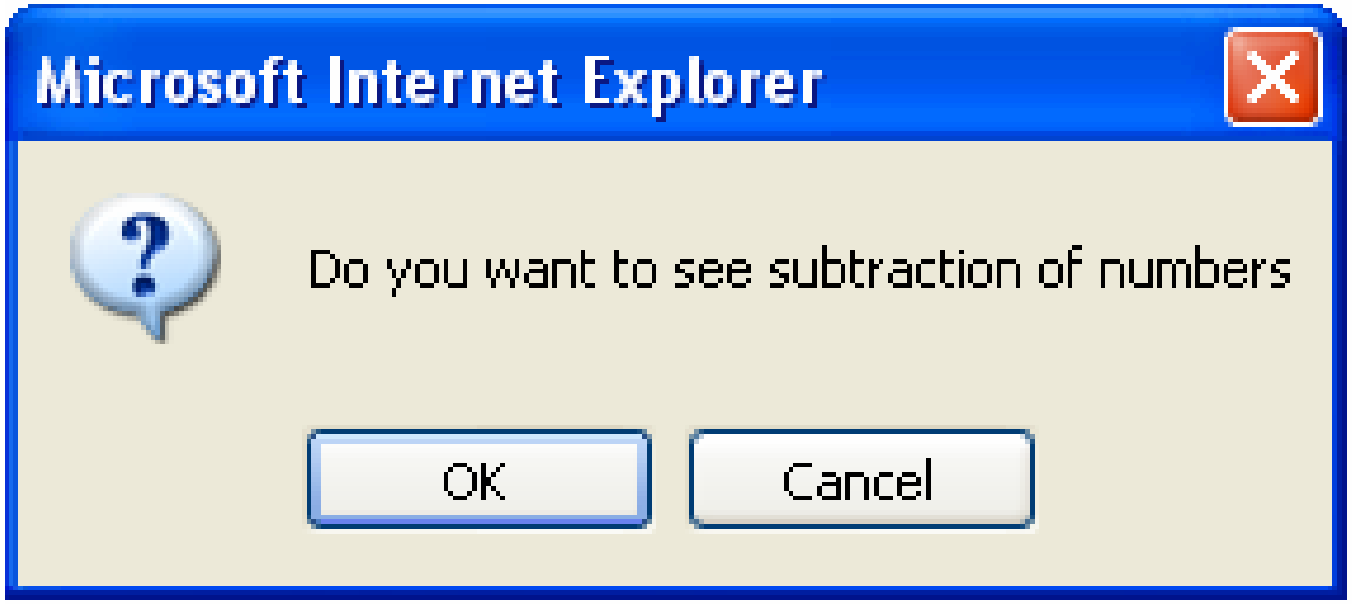
- When alert dialog box popped up, the user should click ok before continue browsing.

JavaScript Confirm Dialog Box

- Confirm Dialog Box is used to accept a response from a user.
- Confirm Dialog Box is created by confirm method of window object.
- Confirm method syntax:

```
window.confirm ("message?");
```

Confirm Dialog Box - Example



- When confirm dialog box popped up, the user should click either ok or cancel button to proceed. In either cases, a Boolean value is returned.

Confirm Dialog Box Code - Example

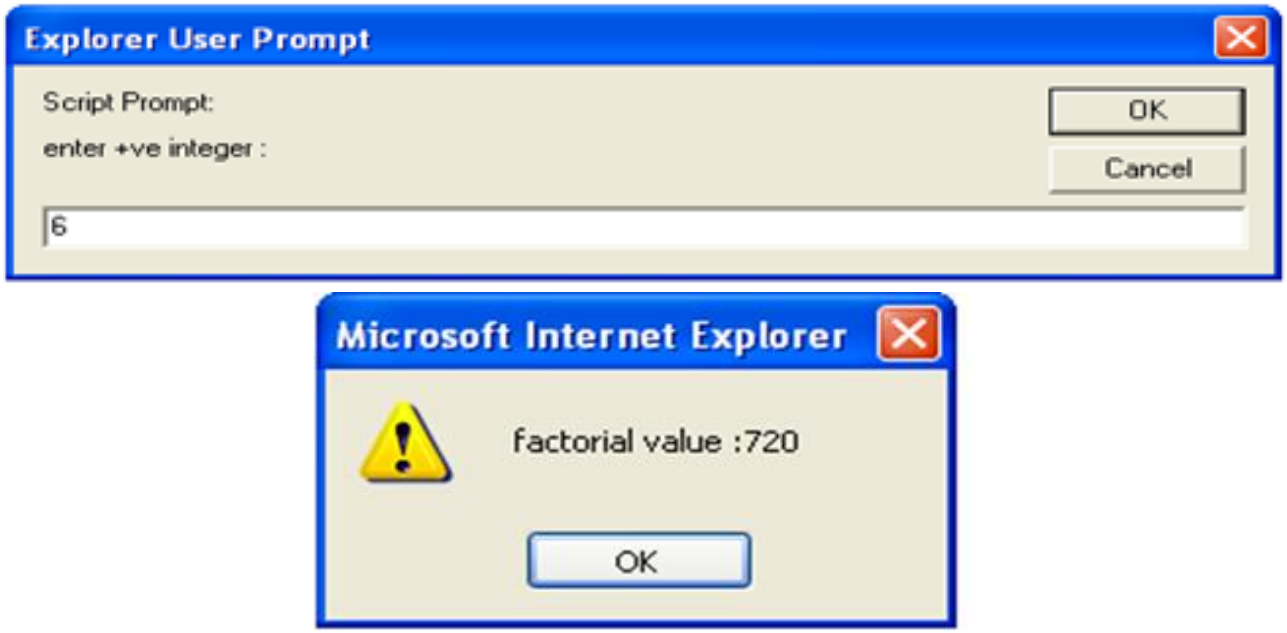
```
<HTML>
<HEAD>
<TITLE> Confirm </TITLE>
<script>
function sub( )
{
x= 10
y=3
z=x-y
Var response =
window.confirm("Do you want to
see subtraction of numbers")
if(response == true)
{
document.write("result is :" + z)
}
else
{
document.write("you clicked cancel
button")
}
}
</script>
</HEAD>
<BODY onload="sub( )">
</BODY>
</HTML>
```

JavaScript Prompt Dialog Box

- Prompt Dialog Box is used to accept data from the user at runtime.
- Prompt Dialog Box is created by prompt method of window object.
- Prompt method syntax:

```
window.prompt ("message", "default text");
```

Prompt Dialog Box - Example



- When prompt dialog box popped up, the user should enter input data to proceed then click either ok or cancel button.
- If user click ok button, the prompt box will return value, otherwise, the prompt box will return null.

Prompt Dialog Box Code - Example

```
<HTML>
<HEAD>
<TITLE> Prompt Dialog
</TITLE>
<script>
function factorial( )
{
var value =
window.prompt("enter +ve
integer:","enter here")
var fact = parseInt(value)
x=1
for(i = fact ; i>=1 ; i--)
{
x = x * i
}
window.alert("factorial value :"+x)
}
</script>
</HEAD>
<BODY onload="factorial( )">
</BODY>
</HTML>
```

JavaScript Loops

```
<script type="text/javascript">
var i = 0;
while (i < 10)
{
    document.write("Step:  " + i);
    document.write("<br />");
    // The previous two lines can be written as
    // document.writeln( “Step: " + i + "<br />" );
    i++;
}
</script>
```

JavaScript Loops Break

```
<script type = "text/javascript">
for ( var i = 1; i <= 10; i++ )
{
    if ( i == 3 )
    break;
    document.writeln("Step: " + i + "<br />" );
}
</script>
```

JavaScript Loops Continue

```
<script type = "text/javascript">
for ( var i = 1; i <= 10; i++ )
{
    if ( i == 3 )
    continue;
    document.writeln("Step: " + i + "<br />" );
}
</script>
```

JavaScript Exceptions

```
<script type="text/javascript">
var value=window.prompt("Enter a
value between 0 and 100","");
try
{
    if(value<0)
    {
    throw "Error1";
    }
    else if(value>100)
    {
    throw "Error2";
    }
}
catch(er)
{
if(er=="Error1")
{
window.alert("Error! The value
can’t be negative");
}
if(er=="Error2")
{
window.alert("Error! value more
than 100 is not allowed");
}
}
</script>
```

Examples

Java Script – Example #1 – 1/2
Window.alert

```
1  <!DOCTYPE html>
2
3  <!-- Fig. 6.3: welcome3.html -->
4  <!-- Alert dialog displaying multiple lines. -->
5  <html>
6     <head>
7        <meta charset = "utf-8">
8        <title>Printing Multiple Lines in a Dialog Box</title>
9        <script type = "text/javascript">
10          <!--
11          window.alert( "Welcome to\nJavaScript\nProgramming!" );
12          // -->
13       </script>
14    </head>
15    <body>
16       <p>Click Refresh (or Reload) to run this script again.</p>
17    </body>
18 </html>
```

Fig. 6.3 | Alert dialog displaying multiple lines. (Part 1 of 2.)

Java Script – Example #1 – 2/2
Window.alert – Output

Fig. 6.3 | Alert dialog displaying multiple lines. (Part 2 of 2.)

Java Script – Example #2 – 1/2
Window.prompt

```
1  <!DOCTYPE html>
2
3  <!-- Fig. 6.5: welcome4.html -->
4  <!-- Prompt box used on a welcome screen -->
5  <html>
6     <head>
7        <meta charset = "utf-8">
8        <title>Using Prompt and Alert Boxes</title>
9        <script type = "text/javascript">
10          <!--
11          var name; // string entered by the user
12
13          // read the name from the prompt box as a string
14          name = window.prompt( "Please enter your name" );
15
16          document.writeln( "<h1>Hello " + name +
17             ", welcome to JavaScript programming!</h1>" );
18          // -->
19       </script>
20    </head><body></body>
21 </html>
```

Fig. 6.5 | Prompt box used on a welcome screen. (Part 1 of 2.)

Java Script – Example #2 – 2/2
Window.prompt

Fig. 6.5 | Prompt box used on a welcome screen. (Part 2 of 2.)

Java Script – Example #3 – 1/3
Variables input and arithmetic

```
1  <!DOCTYPE html>
2
3  <!-- Fig. 6.7: addition.html -->
4  <!-- Addition script. -->
5  <html>
6     <head>
7        <meta charset = "utf-8">
8        <title>An Addition Program</title>
9        <script type = "text/javascript">
10          <!--
11          var firstNumber; // first string entered by user
12          var secondNumber; // second string entered by user
13          var number1; // first number to add
14          var number2; // second number to add
15          var sum; // sum of number1 and number2
16
17          // read in first number from user as a string
18          firstNumber = window.prompt( "Enter first integer" );
19
20          // read in second number from user as a string
21          secondNumber = window.prompt( "Enter second integer" );
22
```

Fig. 6.7 | Addition script. (Part 1 of 3.)

Java Script – Example #3 – 2/3
Variables input and arithmetic

```
23          // convert numbers from strings to integers
24          number1 = parseInt( firstNumber );
25          number2 = parseInt( secondNumber );
26
27          sum = number1 + number2; // add the numbers
28
29          // display the results
30          document.writeln( "<h1>The sum is " + sum + "</h1>" );
31          // -->
32       </script>
33    </head><body></body>
34 </html>
```

Fig. 6.7 | Addition script. (Part 2 of 3.)

Java Script – Example #3 – 3/3
Variables input and arithmetic
Output

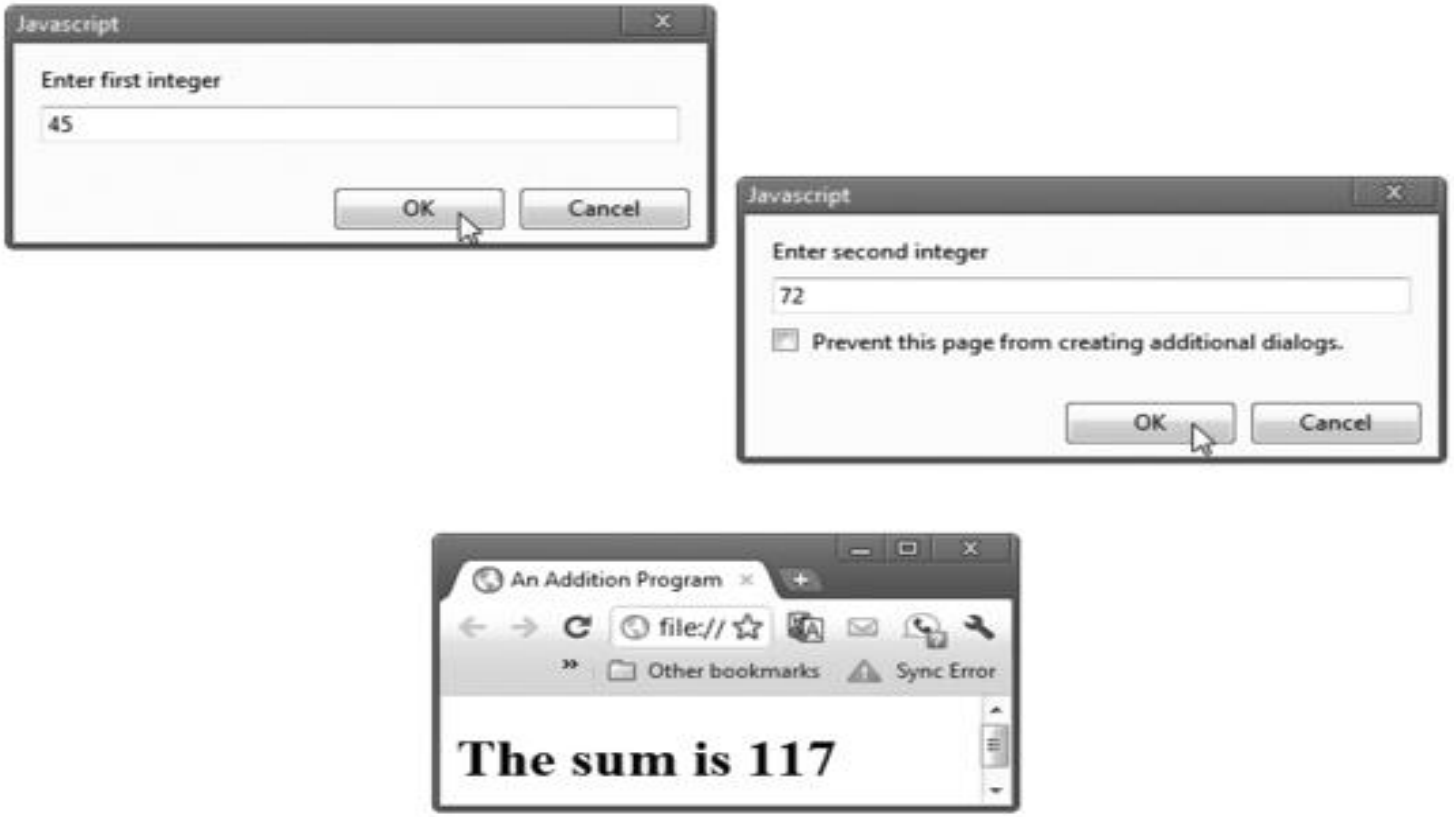


Fig. 6.7 | Addition script. (Part 3 of 3.)

Java Script – Example #4 – 1/5
Loop

```
1  <!DOCTYPE html>
2
3  <!-- Fig. 7.7: average.html -->
4  <!-- Counter-controlled repetition to calculate a class average. -->
5  <html>
6     <head>
7        <meta charset = "utf-8">
8        <title>Class Average Program</title>
9        <script>
10
11          var total; // sum of grades
12          var gradeCounter; // number of grades entered
13          var grade; // grade typed by user (as a string)
14          var gradeValue; // grade value (converted to integer)
15          var average; // average of all grades
16
17          // initialization phase
18          total = 0; // clear total
19          gradeCounter = 1; // prepare to loop
20
```

Fig. 7.7 | Counter-controlled repetition to calculate a class average. (Part 1 of 4.)

Java Script – Example #4 – 2/5
Loop

```
21          // processing phase
22          while ( gradeCounter <= 10 ) // loop 10 times
23          {
24
25             // prompt for input and read grade from user
26             grade = window.prompt( "Enter integer grade:", "0" );
27
28             // convert grade from a string to an integer
29             gradeValue = parseInt( grade );
30
31             // add gradeValue to total
32             total = total + gradeValue;
33
34             // add 1 to gradeCounter
35             gradeCounter = gradeCounter + 1;
36          } // end while
37
```

Fig. 7.7 | Counter-controlled repetition to calculate a class average. (Part 2 of 4.)

Java Script – Example #4 – 3/5
Loop

```
38            // termination phase
39            average = total / 10;   // calculate the average
40
41            // display average of exam grades
42            document.writeln(
43               "<h1>Class average is " + average + "</h1>" );
44
45         </script>
46      </head><body></body>
47   </html>
```

Fig. 7.7 | Counter-controlled repetition to calculate a class average. (Part 3 of 4.)

```
38            // termination phase
39            average = total / 10;   // calculate the average
40
41            // display average of exam grades
42            document.writeln(
43               "<h1>Class average is " + average + "</h1>" );
44
45         </script>
46      </head><body></body>
47   </html>
```

Fig. 7.7 | Counter-controlled repetition to calculate a class average. (Part 3 of 4.)

Java Script – Example #4 – 5/5
Loop

a) This dialog is displayed 10 times. User input is 100, 88, 93, 55, 68, 77, 83, 95, 73 and 62. User enters each grade and presses **OK**.

b) The class average is displayed in a web page

Fig. 7.7 | Counter-controlled repetition to calculate a class average. (Part 4 of 4.)

FULL Example 5 1/4
HTML + CSS + JavaScript

```
1  <!DOCTYPE html>
2
3  <!-- Fig. 9.3: maximum.html -->
4  <!-- Programmer-Defined maximum function. -->
5  <html>
6     <head>
7        <meta charset = "utf-8">
8        <title>Maximum of Three Values</title>
9        <style type = "text/css">
10          p { margin: 0; }
11       </style>
12       <script>
13
14          var input1 = window.prompt( "Enter first number", "0" );
15          var input2 = window.prompt( "Enter second number", "0" );
16          var input3 = window.prompt( "Enter third number", "0" );
17
18          var value1 = parseFloat( input1 );
19          var value2 = parseFloat( input2 );
20          var value3 = parseFloat( input3 );
21
22          var maxValue = maximum( value1, value2, value3 );
23
```

Fig. 9.3 | Programmer-defined `maximum` function. (Part 1 of 4.)

FULL Example 5 2/4
HTML + CSS + JavaScript

```
24          document.writeln( "<p>First number: " + value1 + "</p>" +
25             "<p>Second number: " + value2 + "</p>" +
26             "<p>Third number: " + value3 + "</p>" +
27             "<p>Maximum is: " + maxValue + "</p>" );
28
29          // maximum function definition (called from line 22)
30          function maximum( x, y, z )
31          {
32             return Math.max( x, Math.max( y, z ) );
33          } // end function maximum
34
35       </script>
36    </head><body></body>
37 </html>
```

Fig. 9.3 | Programmer-defined `maximum` function. (Part 2 of 4.)

FULL Example#5 3/4
HTML + CSS + JavaScript

Fig. 9.3 | Programmer-defined `maximum` function. (Part 3 of 4.)

FULL Example#5 4/4
HTML + CSS + JavaScript
Output

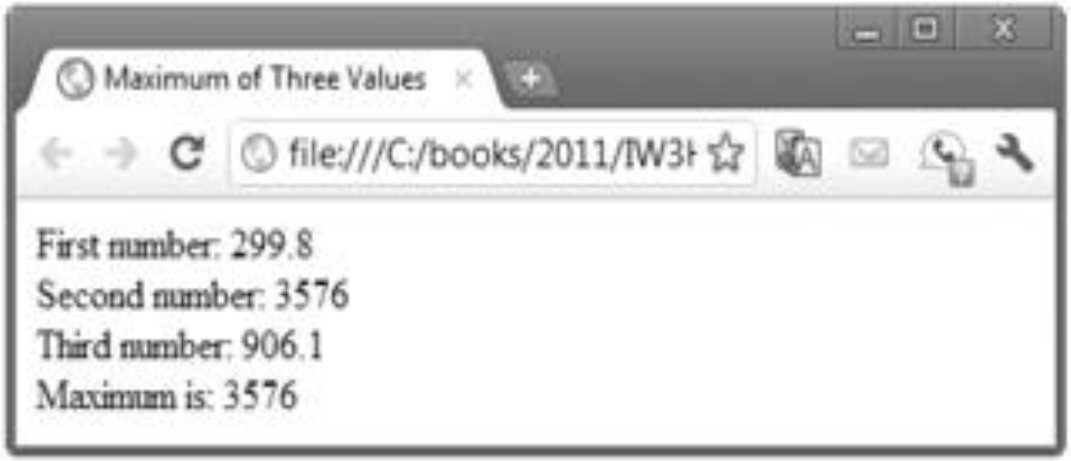


Fig. 9.3 | Programmer-defined `maximum` function. (Part 4 of 4.)