# 1-Introduction to JavaScript

# Introduction

- JavaScript is used to enhance the functionality and appearance of web pages.
- JavaScript is a client-side scripting language that runs entirely inside the web browser.
- All major web browsers contain JavaScript **interpreters**, which process the commands written in JavaScript.

# JavaScript Can

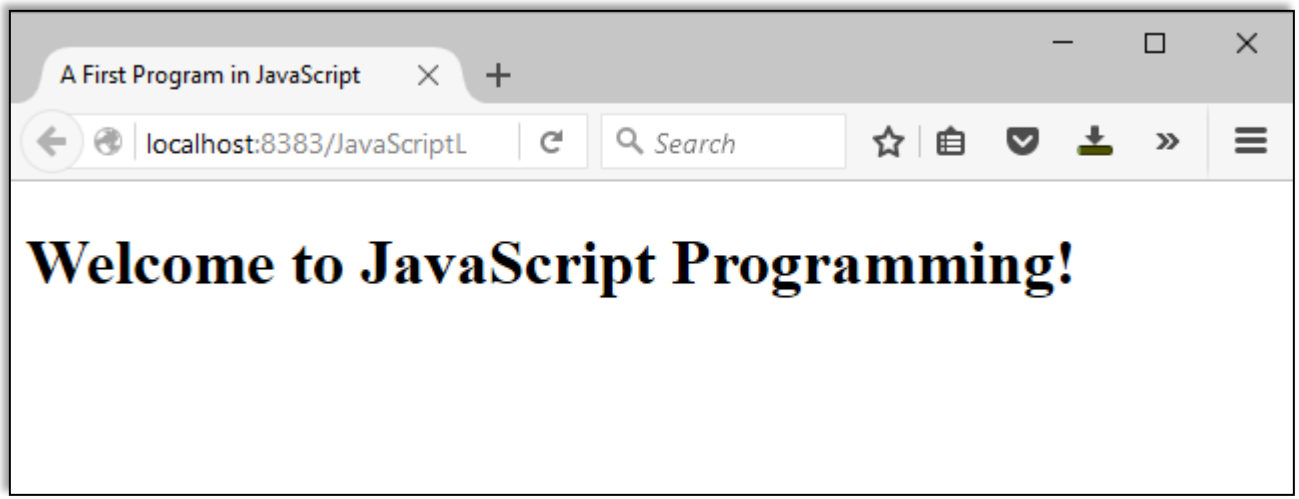
- Change HTML Content
- Change HTML Attributes
- Change HTML Styles
- Validate Data

# First JavaScript Example:

- Displaying a Line of Text with JavaScript in a Web Page
- We begin with a simple script that displays the text "Welcome to JavaScript Programming!" in the HTML5 document.

Welcome.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>A First Program in JavaScript</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <script type = "text/javascript">
        document.write( "<h1>Welcome to JavaScript Programming!</h1>");
      </script>
    </head>
    <body> </body>
</html>
```


A First Program in JavaScript
localhost:8383/JavaScriptL
Search
Welcome to JavaScript Programming!

## document.write()

- The `document` object's `write` method
  - Writes a line of HTML5 text in the HTML5 document

## statement terminator

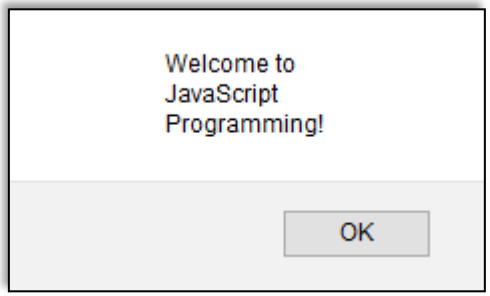
- Every statement should end with a semicolon (also known as the **statement terminator**), although none is required by JavaScript

# Alert Dialog

- Dialogs
  - Useful to display information in windows that “pop up” on the screen to grab the user’s attention.
  - Typically used to display important messages to the user browsing the web page.
  - Method `alert` requires as its argument the string to be displayed.

## Welcome3.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>Printing Multiple Lines in a Dialog Box</title>
        <script type = "text/javascript">
        alert("Welcome to\nJavaScript\nProgramming!");
      </script>
    </head>
    <body>
        <p>Click Refresh (or Reload) to run this script again.</p>
    </body>
</html>
```

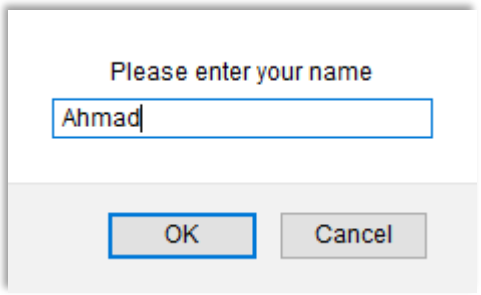


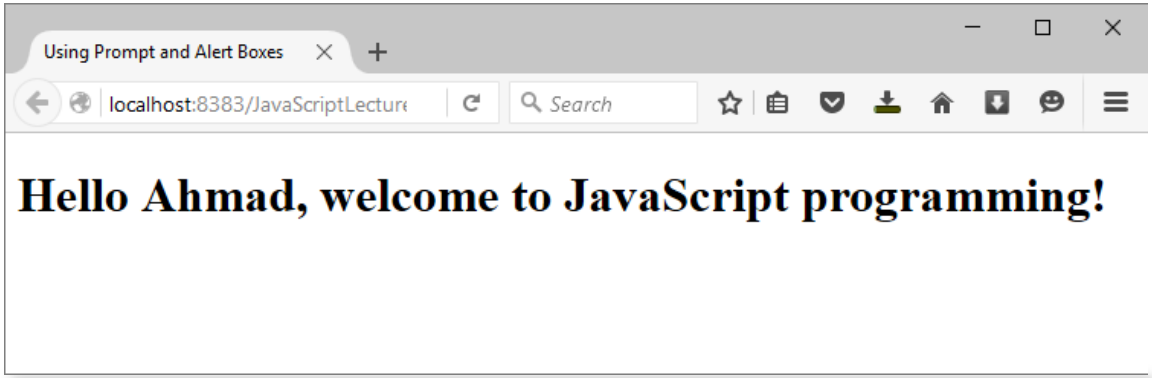
# Prompt Dialog

- The **prompt** dialog allows the user to enter a value that the script can use.
- The next script creates a dynamic welcome page that obtains the user's name, then displays it on the page.

Welcome4.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>Using Prompt and Alert Boxes</title>
        <script type = "text/javascript">
        var name;
        name = prompt( "Please enter your name" );
        document.write("<h1>Hello " + name + ", welcome to JavaScript
programming!</h1>");
      </script>
    </head><body></body>
</html>
```


Please enter your name
Ahmad
OK
Cancel

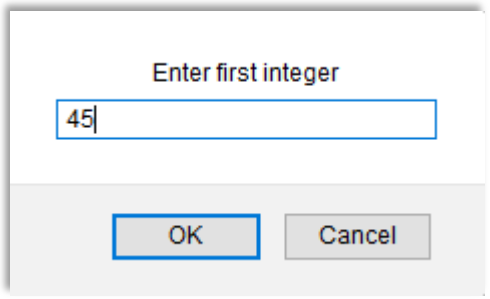

Using Prompt and Alert Boxes
localhost:8383/JavaScriptLecture
Search
Hello Ahmad, welcome to JavaScript programming!

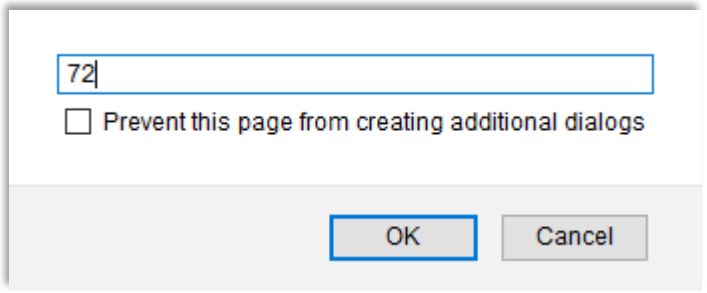
# Prompt Dialog (cont.)

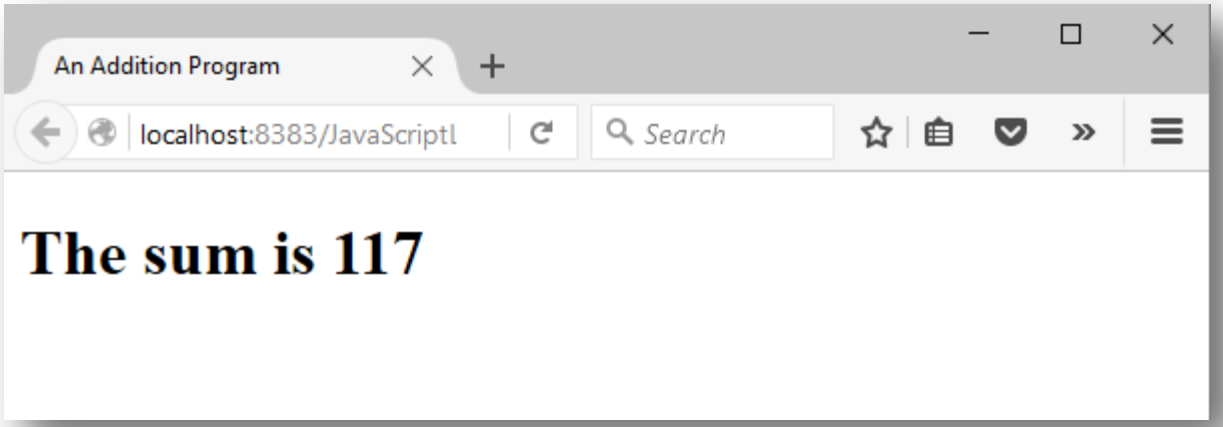
- The `window` object's `prompt` method displays a dialog into which the user can type a value.
  - The first argument is a message that directs the user to take a specific action.
  - The optional second argument is the default string to display in the text field.
- Script can then use the value that the user inputs.

## addition.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8">
    <title>An Addition Program</title>
    <script type = "text/javascript">
      var firstNumber; // first string entered by user
      var secondNumber; // second string entered by user
      var number1; // first number to add
      var number2; // second number to add
      var sum; // sum of number1 and number2
      firstNumber = prompt("Enter first integer");
      secondNumber = prompt("Enter second integer");
      number1 = parseInt(firstNumber);
      number2 = parseInt(secondNumber);
      sum = number1 + number2; // add the numbers
      document.write("<h1>The sum is " + sum + "</h1>");
    </script>
  </head><body></body>
</html>
```


Enter first integer
45
OK
Cancel


72
Prevent this page from creating additional dialogs
OK
Cancel


An Addition Program
localhost:8383/JavaScriptl
Search
The sum is 117

# *Linking JavaScript to HTML*

- In HTML, JavaScript code must be inserted between <script> and </script> tags
- JavaScript can be placed in the <body> and the <head> sections of an HTML page

# JavaScript in <head>

- Using Scripts Within a Document Head by adding the JavaScript in <head> section.

```
1   <!DOCTYPE html>
2   <html>
3     <head>
4       <title>A First Program in JavaScript</title>
5       <meta charset="UTF-8">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7       <script type = "text/javascript">
8         document.writeln( "<h1>Welcome to JavaScript Programming!</h1>");
9       </script>
10    </head>
11    <body> </body>
12  </html>
```

# JavaScript in <body>

- The <script> tag in the HTML body indicates to the browser that the text which follows is a JavaScript.

```
1   <!DOCTYPE html>
2   <html>
3     <head>
4       <title>A First Program in JavaScript</title>
5       <meta charset="UTF-8">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     </head>
8     <body>
9       <script type = "text/javascript">
10        document.writeln("<h1>Welcome to JavaScript Programming!</h1>");
11      </script>
12    </body>
13  </html>
```

# External JavaScript

- JavaScript Files can be placed in external files by using the following syntax:

**<script src="myScript.js"></script>**

```
1   <!DOCTYPE html>
2   <html>
3     <head>
4       <title>A First Program in JavaScript</title>
5       <meta charset="UTF-8">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7     </head>
8     <body>
9       <script src="myScript.js"></script>
10    </body>
11  </html>
```

# *Using Comments*

- single-line comment

```
// This is a comment
```

- multiline comments

```
/* This is a section
   of multiline comments
   that will not be
    interpreted */
```

# Variables

- JavaScript uses keyword ***var*** to declare a variable.
- Variables use the following rules:
  - A variable may include only the letters a-z , A-Z , 0-9 , the $ symbol, and the underscore ( _ ).
  - No other characters, such as spaces or punctuation, are allowed in a variable name. The first character of a variable name can be only a-z , A-Z , $ , or _ (no numbers).
  - Names are case-sensitive. Count , count , and COUNT are all different variables.

# JavaScript Data Types

- JavaScript variables can hold many **data types**: numbers, strings, arrays, objects.

```
var length = 16;                                // Number
var lastName = "Johnson";                       // String
var cars = ["Saab", "Volvo", "BMW"];            // Array
var x = {firstName:"John", lastName:"Doe"};     // Object
```

# String Variables

- JavaScript string variables should be enclosed in either single or double quotation.

**greeting = "Hello there“**

**warning = 'Be careful'**

- You may include a single quote within a double-quoted string or a double quote within a single-quoted string.

**greeting = "Hello ‘there’ “**

# Numerical Variables

- Creating a numeric variable is as simple as assigning a value

**count = 42**

**temperature = 98.4**

# Boolean Variables

- Boolean variable is used to hold on of the two values

**isCorrect = true**

**isCorrect = false**

# Operators

- Operators in JavaScript can involve mathematics, comparison and logical operations.

## 1. Arithmetic Operators

- JavaScript provides the basic arithmetic operators.

| Operator | Description | Example |
|---|---|---|
| + | Addition | `j + 12` |
| - | Subtraction | `j - 22` |
| * | Multiplication | `j * 7` |
| / | Division | `j / 3.13` |
| % | Modulus (division remainder) | `j % 6` |
| ++ | Increment | `++j` |
| -- | Decrement | `--j` |

## 1. Assignment Operators

- The assignment operators are used to assign values to variables.

| Operator | Example | Equivalent to |
|---|---|---|
| = | j = 99 | j = 99 |
| += | j += 2 | j = j + 2 |
| += | j += 'string' | j = j + 'string' |
| -= | j -= 12 | j = j - 12 |
| *= | j *= 2 | j = j * 2 |
| /= | j /= 6 | j = j / 6 |
| %= | j %= 7 | j = j % 7 |

# 1. Comparison Operators

- Comparison operators are generally used inside a control statement such as an if statement.

| Operator | Description | Example |
|---|---|---|
| == | Is *equal* to | `j == 42` |
| != | Is *not equal* to | `j != 17` |
| > | Is *greater than* | `j > 0` |
| < | Is *less than* | `j < 100` |
| >= | Is *greater than or equal* to | `j >= 23` |
| <= | Is *less than or equal* to | `j <= 13` |
| === | Is *equal* to (and of the same type) | `j === 56` |
| !== | Is *not equal* to (and of the same type) | `j !== '1'` |

# 1. Logical Operators

- JavaScript uses the following logical operators:

| Operator | Description | Example |
|---|---|---|
| && | And | `j == 1 && k == 2` |
| \|\| | Or | `j < 100 \|\| j > 0` |
| ! | Not | `! (j == k)` |

# Conditionals

- Conditional are used to alter the program flow  based on certain condition.

## 1. if Statement

- The if statement t is used to choose among alternative courses of action in a script.

## If-example.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>if statement example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script type = "text/javascript">
        a = 15;
        if (a < 100)
        {
           document.write("a is less than 100");
        }
</script>
  </body>
</html>
```

## 2. *if ... else Statement*

- When a condition has not been met, you can execute an alternative by using an else statement.

**If-else-example.html**

```
<!DOCTYPE html>
<html>
  <head>
    <title>if...else example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script type = "text/javascript">
       a = 105;
       if (a > 100)
       {
          document.write("a is greater than 100");
       }
       else
       {
          document.write("a is less than or equal to 100");
       }
    </script>
  </body>
</html>
```

## 3. *switch Statement*

- The switch statement is a multiple-selection statement because it selects among many different actions, depending on the value of an expression

# Switch-example.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>To do supply a title</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script>
      page = "About"
      switch (page)
      {
        case "Home":
          document.write("You selected Home");
          break
        case "About":
          document.write("You selected About");
          break
        case "News":
          document.write("You selected News");
          break
        case "Login":
          document.write("You selected Login");
          break
        case "Links":
          document.write("You selected Links");
          break
        default:
          document.write("Unrecognized selection");
          break
      }
    </script>
  </body>
</html>
```

## 4. The ? Operator

- The ? Operator combined with the : characters provides a quick way for doing if...else test.

q-example.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>The ? Operator Example </title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script>
      mark = 73;
      document.write( mark >= 50 ? "Pass" : "Fail" );
    </script>
  </body>
</html>
```

# Looping

- Loops is used to execute a block of code a number of times.

## 1. while Loops

- A JavaScript while loop first checks the value of an expression and starts executing the statements within the loop only if that expression is true . If it is false , execution skips over to the next JavaScript statement.

## While-loop-example.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>While loop example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  </head>
  <body>
    <script>
      counter = 0;
      while (counter < 5)
      {
        document.write("Counter: " + counter + "<br>");
        ++counter;
      }
    </script>
  </body>
</html>
```

## 2. do...while Loops

- do...while loop is similar to a while loop, except that the test expression is checked only after each iteration of the loop.

do-loop-example.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>do loop example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script>
      count = 1;
      do
      {
        document.write(count + " times 7 is " + count * 7 + "<br>");
      } while (++count <= 7)
    </script>
  </body>
</html>
```

## 3. *for Loops*

- for loop is a single looping construct that uses three parameters for each statement:
  - An initialization expression
  - A condition expression
  - A modification expression

**for-loop-example.html**

```
<!DOCTYPE html>
<html>
  <head>
    <title>For loop Example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script>
      for (count = 1; count <= 7; ++count)
      {
        document.write(count + "times 7 is " + count * 7 + "<br>");
      }
    </script>

  </body>
</html>
```

# Breaking Out of a Loop

- it is used to break out of a loop when certain condition is true.

**Break-example.html**

```
<!DOCTYPE html>
<html>
  <head>
    <title>break example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script>
      colors = new Array();
      colors[17] = "Red";
      for (j = 0; j < 20; ++j)
      {
        if (colors[j] == "Red")
        {
          document.write("<br>- Found at location " + j);
          break
        }
        else
          document.write(j + ", ");
      }
    </script>

  </body>
</html>
```

# The continue Statement

- It is used to skip the remaining statements just for current iteration of the loop.

**continue-example.html**

```
<!DOCTYPE html>
<html>
  <head>
    <title>continue example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <script>
      colors = new Array();
      colors[4] = "Red";
      colors[11] = "Red";
      colors[17] = "Red";
      for (j = 0; j < 20; ++j)
      {
        if (colors[j] == "Red")
        {
          document.write("<br>- Found at location " + j + "<br>");
          continue
        }
        document.write(j + ", ");
      }
    </script>
  </body>
</html>
```

# Thanks !