UNIVERSITY OF DIYALA

# *Web Design and Programming*

## *Lecture 8:*

## *JavaScript Technology*

*Assoc. Prof.*
*Ali A. Al-Ani*

5/21/2022

Department of Computer Science
College of Science

UNIVERSITY OF DIYALA

## *JavaScript*

- ***JavaScript is a lightweight programming language, often referred to as a scripting language, whose implementations allow client-side script to interact with the user and make dynamic pages****. It is an interpreted programming language with* ***object-oriented capabilities.***
- *JavaScript was first introduced into the browser in* ***Netscape 2.0****, although it was known as* ***Live Script*** *at the time. The idea behind it was to add interactive features to documents on the Web, which up to that point had been static.*
- *We should note that JavaScript is not the same as* ***Java****, which is a* ***bigger programming*** *language (although there are some similarities).*

5/21/2022

Department of Computer Science
College of Science

## *Advantages of JavaScript*

- *The merits of using JavaScript are :*
  1. ***Less server interaction:*** *We can validate user input before sending the page off to the server. This saves server traffic, which means less load on our server.*
  2. ***Immediate feedback to the visitors:*** *They don't have to wait for a page reload to see if they have forgotten to enter something.*
  3. ***Increased interactivity:*** *We can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.*
  4. ***Richer interfaces:*** *We can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to our site visitors.*

## *JavaScript Development Tools*

- *One of major strengths of JavaScript is that it does not require **expensive development tools**. We can start with a simple text editor such as **Notepad**. Since it is an interpreted language inside the context of a web browser, we don't even need to buy a **compiler**. To make our life simpler, various vendors have come up with very nice JavaScript editing tools. Some of them are :*
  1. ***Microsoft FrontPage***
  2. ***Macromedia Dreamweaver MX .***
  3. ***Macromedia Home Site.***

## *Enabling JavaScript in Browsers*

- *All the modern browsers come with **built-in support for JavaScript**. Frequently, we may need to enable or disable this support manually. Here are the steps to turn on or turn off JavaScript in Chrome :*
  1. *Click the Chrome menu at the top right hand corner of our browser. **Select Settings**.*
  2. *Under the **Privacy and Security section**, click the **Site settings button**.*
  3. *In the "**Javascript**" section, select "**Sites can use Javascript (recommended)**" or "**Don't allow sites to use Javascript**".*

## *JavaScript: Placement in HTML File*

- *There are **three places** where we can put our Java Scripts and a single HTML document can use all three because there is **no limit on the number** of scripts one document can contain:*

1. *In the **<head> of a page**: These scripts will be **called when an event triggers them**.*
2. *In the **<body> section**: These scripts will **run as the page loads**.*
3. *In an **external file**: If the link is placed inside the **<head> element**, the script is treated the same as when the script lives inside the head of the document waiting for an event to trigger it, whereas if it is placed in the <body> element it will act like a script in the body section and execute as the page loads.*

## *JavaScript: Syntax*

- *JavaScript can be implemented using JavaScript statements that are placed within the* ***<script>... </script>.***
- *We can place the* ***<script> tags****, containing our JavaScript, anywhere within our web page, but it is normally recommended that we should keep it within the* ***<head> tags****.*
- *The* ***<script> tag*** *alerts the browser program to start interpreting all the text between these tags as a script. A simple syntax of our JavaScript will appear as follows.*

```
<script ...>
  JavaScript code
</script>
```

## *JavaScript: Syntax*

- *The* ***script tag takes two important attributes****:*
  1. ***Language:*** *This attribute specifies what scripting language we are using (such as VbScript, JavaScript and so on). Typically, its value will be* ***javascript****.*
  2. ***Type:*** *This attribute is what is now recommended to indicate the scripting language in use and its value should be set to* ***"text/javascript".***

```
<script language="javascript" type="text/javascript">
  JavaScript code
</script>.
```

## *JavaScript: Placement in External File*

- ***Write JavaScript in external documents that have the file extension .js is a particularly good option if our script is used by more than one page because we do not need to repeat the script in each page that uses it, and if we want to update our script we need only change it in one place.***
- *When we place our JavaScript in an **external file**, we need to use the **src attribute** on the **<script> element**; the value of the src attribute should be an **absolute or relative URL pointing to the file containing the JavaScript**. For example:*

```
<script language="javascript"  type="text/javascript"  src="validation.js" />
```

## *JavaScript: Variables*

- *One of the most fundamental properties of a programming language is the set of **data types** it supports. JavaScript allows us to work with **three primitive data types**:*
  1. ***Numbers: 123, 120.50 etc.***
  2. ***Strings of text: "This text string" etc.***
  3. ***Boolean: true or false.***
- *<u>**Note:**</u> JavaScript does not make a distinction between **integer values** and **floating-point values**. All numbers in JavaScript are represented as **floating-point values**. Variables are **declared** with the **var** keyword as follows.*

```
var money;      var name =“Ali”;      var A=2, B=5.009;
```

## *JavaScript: Operators*

- *Let us take a simple expression **4 + 5 is equal to 9**. Here **4 and 5** are called **operands** and '+' is called the **operator**. JavaScript supports the following types of operators.*

  1. ***Arithmetic Operators:** (+, -, /, *, ++, --, %)*
  2. ***Comparison Operators:** (==, !=, >, <, <=,>=)*
  3. ***Logical (or Relational) Operators:** (&&, ||, !)*
  4. ***Assignment Operators:** (=, +=, -=, /=, *=, %=)*

## *JavaScript: Operators*

- *The following code shows how to use arithmetic operators in JavaScript.*

```
<script type="text/javascript">
      <!--
        var a = 33, b = 10;
        var  linebreak = "<br />";
        document.write("a + b = ");
         result = a + b;
         document.write(result);
         document.write(linebreak);
         document.write("a - b = ");
         result = a - b;
         document.write(result);
      //-->
 </script>.
```

## *JavaScript: if...else Statement*

- *JS supports conditional statements which are used to perform different actions based on different conditions.. JS supports the following forms of if..else statement:*
  1. ***if statement,***
  2. ***if...else statement,***
  3. ***if...else if... statement.***

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

## *JavaScript: Loops*

- *While writing a program, we may encounter a situation where we need to perform an action over and over again. JS supports all the necessary loops Such as:*

1. ***The while Loop:*** *The syntax of while loop in JS is as follows:*

```
while (expression){
   Statement(s) to be executed if expression is true  }
```

2. ***The do...while Loop:*** *The flow chart of a do-while loop would be as follows:*

```
do{
   Statement(s) to be executed;
} while (expression);
```

## *JavaScript: Loops*

3. ***The For Loop:*** *The syntax of for loop is JavaScript is as follows:*

***for (initialization; test condition; iteration statement)***

***{ Statement(s) }***

- *The following example to learn how a for loop works in JS.*

```
<script type="text/javascript">
      <!--
        var count;
        document.write("Starting Loop" + "<br />");
        for(count = 0; count < 10; count++){
           document.write("Current Count : " + count ); }
       document.write("Loop stopped!");
      //-->     </script>.
```

## *JavaScript: Functions*

- *A function is a group of* ***reusable code*** *which can be called anywhere in our program. This eliminates the need of writing the same code again and again. Like any other advanced programming language, JS allows us to write our own functions as well.*
- *The most common way to define a function in JS is by using the* ***function*** *keyword, followed by a* ***unique function name****, a* ***list of parameters*** *(that might be empty), and a statement block surrounded by curly braces.*

```
<script type="text/javascript">
  <!--
    function functionname(parameter-list)   {     statements    }
  //-->
</script>.
```

## *JavaScript: Functions*

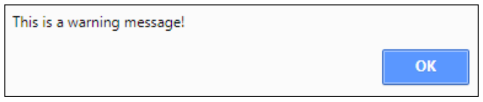
- *Try the following example. It defines a function that takes two parameters and adding them before returning the resultant in the calling program.*

```
<html>
 <head>     <script type="text/javascript">
      function add(a, b)       {  var z;    z= a+ b;    return z;    }
 function addresult()    {   var result;   result = add(10, 5);    document.write (result );     }
   </script>
 </head>
 <body>
    <p>Click the following button to call the function</p>
     <form>
       <input type="button" onclick="addresult()" value="Call Function">
    </form>      </body></html>
```

## *JavaScript: Dialog Boxes*

- *JS supports **three important types of dialog boxes**. These dialog boxes can be used to **raise an alert**, or to **get confirmation on any input** or to have a **kind of input from the users**.*

1. ***Alert Dialog Box:** An alert dialog box is mostly used to give a **warning message** to the users. For example, if one input field requires to enter some text but the user does not provide any input, then as a part of validation, **you can use an alert box to give a warning message**. **Nonetheless, an alert box can still be used for friendlier messages**. Alert box gives only one button **"OK" to select and proceed**.*

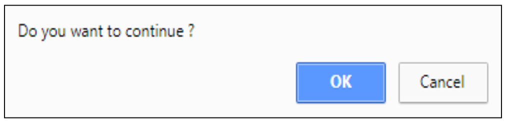


```
This is a warning message!
```

```
<script type="text/javascript“>
<!--
alert ("This is a warning message!");    //-->     </script>
```

## *JavaScript: Dialog Boxes*

2. ***Confirmation Dialog Box:*** *A confirmation dialog box is mostly used to take user's* ***consent on any option****. It displays a dialog box with two buttons:* ***Ok and Cancel****. If the user clicks on the* ***OK button****, the window method confirm() will return* ***true****. If the user clicks on the* ***Cancel button****, then confirm() returns* ***false****.*



```
<script type="text/javascript">
      <!--
        function getConfirmation(){
          var retVal = confirm("Do you want to continue ?");
          if( retVal == true ){  document.write ("User wants to continue!");   return true;   }
          else {
             document.write ("User does not want to continue!");   return false;       }
        }
      //--> <script/>
```

## *JavaScript: Dialog Boxes*

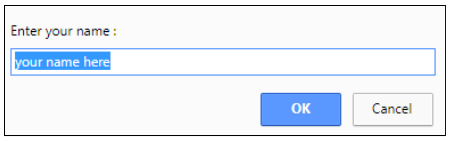
3. ***Prompt Dialog Box:*** *The prompt dialog box is very useful when we want to pop-up a text box to* ***get user input****. Thus, it enables us to interact with the user.* ***The user needs to fill in the field and then click OK.***

- *This dialog box is displayed using a method called* ***prompt()*** *which takes two parameters:* ***(i) a label which you want to display in the text box*** *and* ***(ii) a default string to display in the text box.***
- *This dialog box has two buttons:* ***OK*** *and* ***Cancel****. If the user clicks the* ***OK button****, the window method prompt() will return the* ***entered value from the text box****. If the user clicks the* ***Cancel button****, the window method prompt() returns* ***null****.*

## JavaScript: Dialog Boxes

- *The following example shows how to use a prompt dialog box:*

```
<html>   <head>
            <script type="text/javascript">
         <!--
           function getValue(){
              var retVal = prompt("Enter your name : ", "your name here");
              document.write("You have entered : " + retVal);
           }
         //-->      </script>   </head>
     <body>
       <p>Click the following button to see the result: </p>
       <form>
         <input type="button" value="Click Me" onclick="getValue();" />
       </form>   </body>         </html>.
```


Enter your name :

your name here

## JavaScript: Events

- *JS interaction with HTML is handled through **events** that occur when the user or the browser manipulates a page. **When the page loads, it is called an event**. **When the user clicks a button, that click too is an event**. Other examples include events like **pressing any key**, **closing a window**, **resizing a window**, etc.*
- *Developers can use these events to execute **JS coded responses**, which cause **buttons to close windows**, **messages to be displayed to users**, **data to be validated**, and **virtually any other type of response imaginable**. In the following, we will see a few examples to understand a relation between Event and JavaScript*

## JavaScript: Events

1. **onclick Event Type:** *This is the most frequently used event type which occurs when a user* ***clicks the left button of his mouse****. We can put our validation, warning etc., against this event type.*

```
<html>
 <head>
   <script type="text/javascript">
     <!--     function sayHello() {       alert("Hello World“)         }     //-- >  </script>
   </head>
<body>
   <p>Click the following button and see result</p>
<form>     <input type="button" onclick="sayHello()" value="Say Hello" />     </form>
</body>   </html>
```

## JavaScript: Events

2. **onsubmit Event type:** *onsubmit is an event that occurs when we try to* ***submit a form****. We can put our form validation against this event type. The following example shows how to use onsubmit. Here we are calling a* ***validate()*** *function before submitting a form data to the webserver. If* ***validate()*** *function returns true, the form will be submitted, otherwise it will not submit the data.*

```
   <head>  <script type="text/javascript">
     <!--    function validation() {   all validation goes here return either true or false    }
     //-->      </script>        </head>
<body>
   <form method="POST" action="login.html" onsubmit=" validation()">
       <input type="submit" value="Submit" />      </form>  </body>   </html>
```

## *JavaScript: Form Validation*

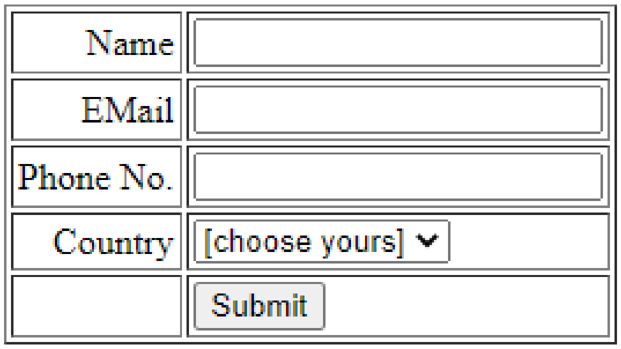
- *Form validation normally used to occur at the **server**, after the client had entered all the necessary data and then pressed the **Submit button**. If the data entered by a user was **incorrect or was simply missing**, the server would have to send all the data back to the user and request that the form be **resubmitted with correct information**. This was really a **lengthy process** which used to put a lot of **load work on the server**.*
- ***JS** provides a way to validate form's data on the client's computer **before sending it to the web server.** Form validation generally performs two functions.*
  1. ***Basic Validation.***
  2. ***Data Format Validation.***

## *JavaScript: Form Validation*

1. ***Basic Validation:** First of all, the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for data.*
2. ***Data Format Validation:** Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test correctness of data.*

- *We will take an example to understand the process of validation. In the following is a simple form in html format.*

# JavaScript: Form Validation



| | |
|---|---|
| Name | |
| EMail | |
| Phone No. | |
| Country | [choose yours] |
| | Submit |

```
<html>
    <head>  <title>Form Validation</title>
    <script type="text/javascript">
      <!--
        // Form validation code will come here.
      //-->
    </script>
</head>
  <body>
  <form action="login.html"  name="myForm"  onsubmit="return(validate());">
       <table cellspacing="2" cellpadding="2" border="1">
          <tr>
                <td align="right">Name</td>
       <td><input type="text" name="Name" /></td>
       </tr>
       <tr>
                 <td align="right">EMail</td>
                <td><input type="text" name="EMail" /></td>
       </tr>
       <tr>
                <td align="right">Phone No.</td>
                <td><input type="text" name="PhoneNo" /></td>
       </tr>
       <tr>
                <td align="right">Country</td>
                <td>
                          <select name="Country">
                           <option value="-1" selected>[choose yours]</option>
                          <option value="1">USA</option>
                          <option value="2">UK</option>
                          <option value="3">INDIA</option>
                           </select>
                 </td>
       </tr>
       <tr>
                  <td align="right"></td>
                  <td><input type="submit" value="Submit" /></td>
       </tr>
       </table>
     </form>
  </body> </html>
```

# JavaScript: Form Validation

1. ***Basic Form Validation:*** *First let us see how to do a basic form validation. In the above form, we are calling* ***validate()*** *to validate data when* ***onsubmit*** *event is occurring. The following code shows the implementation of this validate() function.*

```
<script type="text/javascript">
 <!--
     function validate()
   {
      if( document.myForm.Name.value=="" )
     {
       alert( "Please provide your name!" );
       document.myForm.Name.focus();
       return false;
     }
      if( document.myForm.EMail.value=="" )
     {
       alert( "Please provide your Email!" );
       document.myForm.EMail.focus();
       return false;
     }
      if( document.myForm.PhoneNo.value=="")
      {
       alert( "Please provide your  Phone No." );
       document.myForm.PhoneNo.focus();
       return false;
     }
      if( document.myForm.Country.value=="-1" )
     {
       alert( "Please provide your country!" );
       return false;
     }
      return( true );
   }
 //-->     </script>
```

# JavaScript: Form Validation

2. ***Data Format Validation:*** *Now we will see how we can validate our entered form data before submitting it to the web server. The following example shows how to validate an entered email address and phone number. An email address must contain at least a '@' signand and a phone numbers must contain 11 numbers in length at laest. The following Java Script try to validate the email and phone number field.*

```
<script type="text/javascript">
  <!--
   function validateEmail()
   {
     var emailID = document.myForm.EMail.value;
     atsign = emailID.indexOf("@");
     if (atsign < 1)
     {
       alert("Please enter correct email ID");
       document.myForm.EMail.focus();
       return false;
     }
    If (document.myForm.PhoneNo.value.length != 11 )
     {
       alert( "Please provide a Phone No. 11 numbers at laest." );
       document.myForm.PhoneNo.focus();
       return false;
     }
      return( true );
    }  //-->     </script>
```

# *The End*