

www.jump2learn.com



CONCEPTS OF WEB TECHNOLOGY

Jump2Learn - The Online Learning Place

Dr. Jagin M. Patel | Ms. Meghna N. Vithlani

Unit-5 TM JavaScript Functions



- 5.1 JavaScript User-Defined Function
- 5.2 Dialog Box in JavaScript
- 5.3 Form Validation with JavaScript

Jump2Learn

5.1 JavaScript User-defined Function

- Functions are one of the fundamental building blocks in JavaScript.
- Function is a block of statements that perform a specific task.
- Function is executed when it is called or when event occurs.
- It is a reusable code-block.
- Function may or may not have parameters.
- Parameters of a function are local to the function.
- Scope of variable declared within function is limited to that function only.
- Scope of a variable declared outside the function is to all the statements within the JavaScript.

5.1.1 Defining a function

- JavaScript function is defined by using the *function* keyword followed by function name and the parentheses.
- The JavaScript statements that define the function, written in curly brackets { }.
- The function name must be unique.
- The rules of naming a function are similar to naming a variable.

The Syntax: functions without parameter:

```
function Function_Name( )  
{  
    // some lines of codes  
}
```

- If nothing is mentioned in between the parentheses, then it is called an empty parentheses and function is called parameter less.

Example:

Following code declare simple function named *display()*. It display message in alert box.

```
function display( )  
{  
    alert("this is parameter less function");  
}
```

The Syntax: functions with parameter:

- Function may accept arguments or parameters.
- List of parameters are written in between the parentheses and separated by commas.

```
function Function_Name( argument1, argument2,...)  
{
```

```
    // some lines of codes  
}
```

Example:

Following code declare function name sum, which take two parameters and return sum of them.

```
function sum(val1, val2)  
{  
    return (val1 + val2);  
}
```

5.1.2 Placing Functions:

- Functions can be declared anywhere within HTML file. However, it is prefer to declare within <HEAD>...</HEAD> tag.
- The above function can be placed as shown below in example.

Example:

```
<head>  
  <script language="JavaScript">  
    function sum(val1, val2)  
    {  
      document.write (val1 + val2);  
    }  
  </script>  
</head>
```

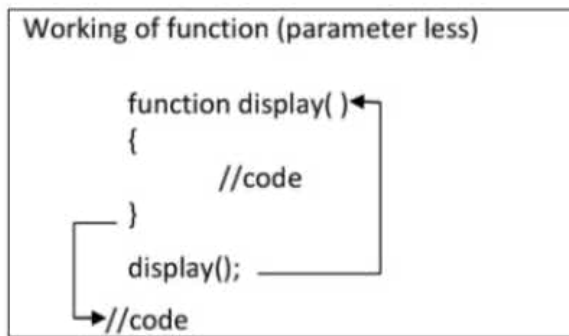
5.1.3 Calling Functions

- Functions do not run automatically. When the page loads, each function waits to run.
- To run a function you must call it.
- To call function, just write function name. If function requires parameters then mention parameter list in parentheses.
- Parameters passed may be a literal or variable.
- When the parameterized function is called, an argument is passed into the function.

Example: calling parameter less function

Following line call function *display*:

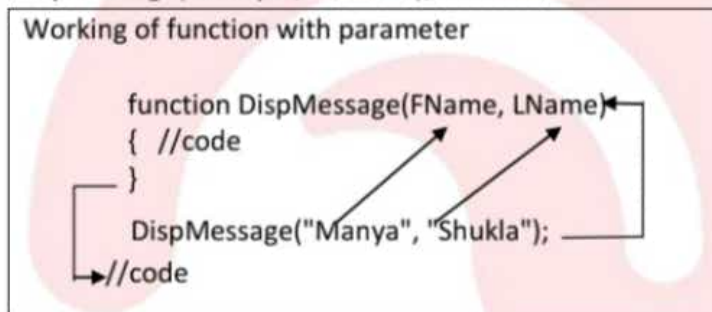
```
display();
```



Example: calling parameterized function

Following line call function *DispMessage* with two parameters "Manya" and "Shukla". Value "Manya" is assigned to first argument FName and value "Shukla" is assigned to second argument LName in function.

```
DispMessage("Manya", "Shukla");
```



Calling function from Event-handlers

- An event handler is a function, which is called when an event occurs, such as the user clicking a button.
- We have already listed out the events in chapter 3.
- The command is written in the format `on<<Event_name>>`, where Event_name is the name for a specific event.

Example:

Following code call function `display()` when page is loaded and *onLoad* event occur.

```

<script language="JavaScript">
    function display()
    {
        document.getElementById("div1").innerHTML="Concept Of Web
        Technology";
    }
</script>
<body onLoad="display();">

```



```
<div id="div1">
</body>
```

Calling Functions from another Function

- It is possible to call other function from another function. To do so, write name of the function to be called, with its parentheses.

Example:

```
function display()
{
    display1(); // calls the another function
}

function display1()
{
    // function codes
}
```

5.1.4 Returning a Value

- A function may return value.
- To return value from function, use keyword *return* in the function.
- The *return* statement is the last statement to be executed in function. Any code after *return* is not executed.

Example:

In following code function sum take two values as parameters and return summation of two values.

```
<script language="JavaScript">
function sum(val1, val2)
{
    return val1 + val2;
};
var result = sum(10,20);
    document.write(result);
</script>
```

Output:

30

5.1.5 Page Redirection

- Page Redirection is a mechanism of transferring or redirecting the users to a different URL or web page from the current URL or web page.

- The redirected web page can be on the same web site or on another websites. It may be on same server or on another server.
- In JavaScript *location* property of the *window* object is used for page redirection.

Syntax:

```
Window.location = "URL";
```

Example:

Following code redirect user to another webpage when user click a button.

```
<html>

<head>
  <script language = "javascript">
    function Redirect()
    {
      window.location = "https://www.jump2learn.com";
    }
  </script>
</head>

<body>
  <p>Click the button to goto jump2learn home page.</p>

  <form>
    <input type = "button" value = " jump2learn" onclick = "Redirect();" />
  </form>
</body>
</html>
```

5.2 Dialog Box in JavaScript

- JavaScript has various dialog boxes that provide facility to take user input or display small text to the user.
- These dialog boxes appear as pop up windows.
- The content of dialog box are depends on the information provided by the user.
- The dialog box causes program execution to halt until user action takes place.
- JavaScript has three kinds of dialog boxes:

- Alert box,
- Confirm box, and
- Prompt box.

5.2.1 Alert Dialog Box

- JavaScript Alert box is the simplest way to display small amounts of textual output to a browser's window.
- The alert dialog box can be used to display a warning message or display some information to user.
- To display alert dialog box use *alert()* method.
- *alert()* method takes a string as an argument and displays in an alert dialog box.
- Alert dialog box has one OK button. Once alert box is open, code not continue processing until the OK button is clicked

Syntax

```
alert("message");
```

Example:

Following code generate alert box and display message "Testing of alert box" .

```
<html>
<head>
  <script language="javascript">
    function show_alert()
    {
      alert("Testing of alert box");
    }
  </script>
</head>
<body>
  <input type="button" onclick="show_alert()" value="Show alert box" />
</body>
</html>
```

Output:



5.2.2 Confirm Dialog Box

- A confirm box serve as a technique for conforming user action.
- The confirm dialog box displays the following information:
 - (i) Message
 - (ii) OK and Cancel Button
- The confirm dialog box, causes program execution to halt until user click OK or Cancel button.
- If the user clicks "OK" button, the box returns true.
- If the user clicks "Cancel" button, the box returns false.

Syntax:

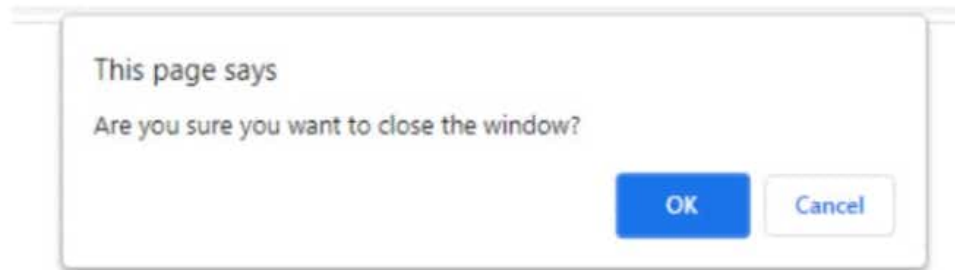
```
confirm(" sometext");
```

Example:

In the following example if user clicks ok then the current window will be closed otherwise he will be redirected to another webpage

```
<html>
<head>
  <script language="javascript">
    function check()
    {
      var ans=confirm("Are you sure you want to close the window?");
      if(ans)
      {
        window.close();
      }
      else{
        window.location("anotherpage.html");
      }
    }
  </script>
</head>
<body>
  <button type="button" onclick="check();">Close Window</button>
</body>
</html>
```

Output:



5.2.3 Prompt Dialog Box

- A prompt box is used to take input from user.
- The *prompt()* method is used to displays the prompt dialog box.
- The *prompt()* dialog box has:
 - (i) message
 - (ii) single data entry field, which accepts user input.
 - (iii) OK and Cancel buttons
- When a prompt box pops up, the user will have to click either "OK" button or "Cancel" button to proceed.
- If the user clicks "OK" button the box returns the input value. If the user clicks "Cancel" button the box returns null.

Syntax :

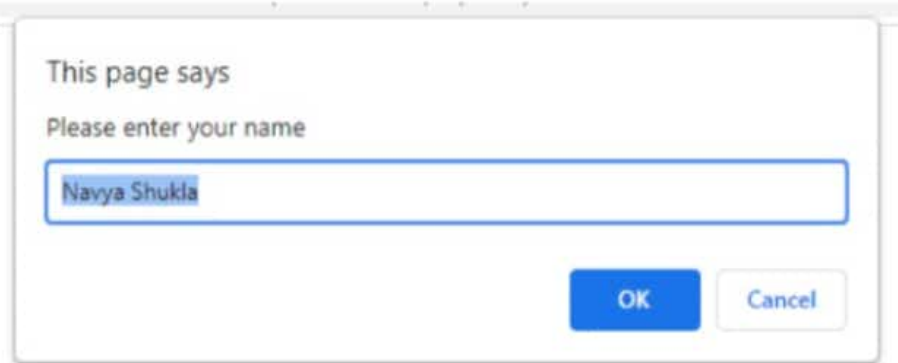
```
prompt(" sometext", " defaultvalue");
```

Example:

Following code demonstrate use of prompt message box.

```
<html>
<head>
<script language="javascript">
    function show_prompt()
    {
        var name=prompt("Please enter your name","Navya Shukla");
        if (name!=null && name!="")
        {
            document.write("Hello " + name + "! How are you today?");
        }
    }
</script>
</head>
```

```
<body>  
    <input type="button" onclick="show_prompt()" value="Show prompt  
    box" />  
</body>  
</html>
```

Output:

When user clicks ok following is displayed on browser

Hello Navya Shukla! How are you today?

5.3 Form Validation with JavaScript

- Validation is a significant part of any web application. While filling a form, it is possible that the user may enter wrong or inappropriate values. Data entered in the form are sent to the server and validated by the server. If data are incorrect, the server sends it back to the client. This is a lengthy process and puts a burden on the server. So, it is important to validate form data before sending the server. If validation occurs on the client-side, it saves time and increases performance. It ensures that user input is clean and correct.
- Using JavaScript we can validate the form on the client side. Most web developers prefer JavaScript form validation.
- Few examples where validation is required:
 - Mobile no. must have 10 digit
 - Age must not negative
 - During registration password and confirm password should match
 - Login credential must enter before submitting
 - enter a valid date
 - marks between 0 and 100
 - valid email format

5.3.1 Basic Validation:

- Sometimes users may not enter data in the field. So, it is necessary to check that the user must enter data in all the mandatory fields. This type of basic validation is done by checking each field in the form and check for data.

Example:

Following example validate the name and password. The name and password are mandatory. Here, we are validating the form on form submit

```
<head>
<script language="JavaScript">
  function form_validation()
  {
    name = document.form1.uname.value;
    password= document.form1.pass_word.value;
    if (name=="")
    {
      alert("name is required");
      return false;
    }
    if (password=="")
    {
      alert("please enter password");
      return false;
    }
  }
</script>
</head>
<body>
  <form name="form1" action="" method="post" onSubmit= "return
  form_validation();">
    Student Name: <input type="text" name="uname">
    <br>
    Password : <input type="password" name="pass_word">
    <br>
    <input type="submit" value="Submit">
  </form>
</body>
```

5.3.2 Data Format Validation:

- Sometimes it is necessary to check data for correct format and value. For example, an email address must enter in the proper format, the value must be a number, etc. This type of validation needs appropriate logic to test correctness of data.

a) Number Validation

- JavaScript can be used to validate input for numeric value only.
- The isNaN() (Not-A-Number) function is used for this purpose.
- The isNaN() function determines whether a given value is an illegal number or not. It returns true if a value is a not number, otherwise returns false.

Following example check whether inputted value is number or not.

```
<script>
function num_validation()
{
    // value of the input field with id="no"
    no = document.getElementById("no").value;
    // checking value of no
    if (isNaN(no) )
    {
        alert( "Input is not number");
    }
    else
    {
        alert( "Input is number");
    }
}
</script>
<input id="no">
<button type="button" onclick="num_validation()">Submit</button>
```

b) Email validation:

- JavaScript can validate the email format.
- Email address follows various criteria such as:
 - Must contain the @ and . character
 - Minimum one character before and after the @.
 - Minimum two characters after . (dot).

Example:

Following code validate email

```
<script type = "text/javascript">
function email_validation()
```



```

    {
        email_id = document.getElementById("email").value;
        at_position = email_id.indexOf("@");
        dot_position = email_id.lastIndexOf(".");
        if (at_position < 1 || ( dot_position - at_position < 2 ))
        {
            alert("Please enter correct email ID")
            return false;
        }
        return( true );
    }
</script>


```

c) Mobile no. validation:

- JavaScript validate mobile no. Following code validate a phone number of 10 digits.

```

<script>
function mobile_number_validation()
{
    var mobile_no_format = /^d{10}$/;
    inputtxt = document.getElementById("mobile_no").value;
    if (inputtxt.match(mobile_no_format))
    {
        return true;
    }
    else
    {
        alert("please enter 10 digit only");
        return false;
    }
}
</script>
Mobile no:<input id="mobile_no" >
<button type="button" onclick="mobile_number_validation()"> Submit
</button>

```

d) Validates that a string has not all blank /whitespace characters:

```

<script>
function string_empty_validation( )
{
    x = document.getElementById("str").value;
    x=x.trim();

```

```
        if (x.length == 0)
        {
            alert("string has blank spaces only ");
            return false;
        }
        else
            return true;
    }
</script>
input string:<input id="str" >
<button type="button" onclick="string_empty_validation()">Submit</button>
```

e) Validating string for minimum length:

- Following script ensure that user must enter string of minimum 4 characters.

```
<script>
function string_length_validation()
{
    string =document.getElementById("str").value;
    if(string.length < 4 )
    {
        alert("Minimum length of string is 4");
        return false;
    }
}
</script>
input string:<input id="str" >
<button type="button" onclick="string_length_validation()">Submit</button>
```

f) Checking string for all letters

- In some situations, input value must contain alphabets (A-Z or a-z) only. Following code ensure that user must enter alphabets only. This is done using Regular Expression.

```
<script>
function all_letter_validation( )
{
    letters = /^[A-Za-z]+$;/;
    string =document.getElementById("str").value;
    if(string.match(letters))
    {
        return true;
    }
    else
    {
        alert("enter only A-Z or a-z charcters");
        return false;
    }
}
```

```
    }  
  }  
</script>  
input string:<input id="str" >  
<button type="button" onclick="all_letter_validation()">Submit</button>
```

g) Validating name:

- The following code validates that user must enter first and last name. This is done using Regular Expression.

```
<script>  
  function name_validation()  
  {  
    var regName = /^[a-zA-Z]+ [a-zA-Z]+$/;  
    string =document.getElementById("str").value;  
    if(string.match(regName))  
    {  
      alert('valid name given');  
    }  
    else {  
      alert('In Valid name given');  
    }  
  }  
</script>  
input name:<input id="str" >  
<button type="button" onclick="name_validation()"> Submit </button>
```
