NexTGen Web

Session: 15

Functions and Objects

Objectives

- Explain functions
- Explain parameterized functions
- Explain return statement
- Describe objects
- Explain different browser objects
- Describe Document Object Model (DOM)

Introduction

To make the code more task-oriented and manageable, JavaScript allows to group statements before they are actually invoked.

This can be achieved by using functions.

A function is a reusable block of code that is executed on the occurrence of an event.

Event can be a user action on the page or a call within the script.

© Aptech Ltd.

Functions

Is an independent reusable block of code that performs certain operations on variables and expressions to fulfill a task.

Might accept parameters, which are variables or values on which it performs operations.

Might return the resultant value to display it in the browser after the operations have been performed.

JavaScript function is always created under the script element.

JavaScript supports both user-defined and built-in functions.

Declaring and Defining Functions 1-3

JavaScript allows declaring a function using the function keyword.

Keyword is followed by the name of the function and the parameters enclosed within the parenthesis.

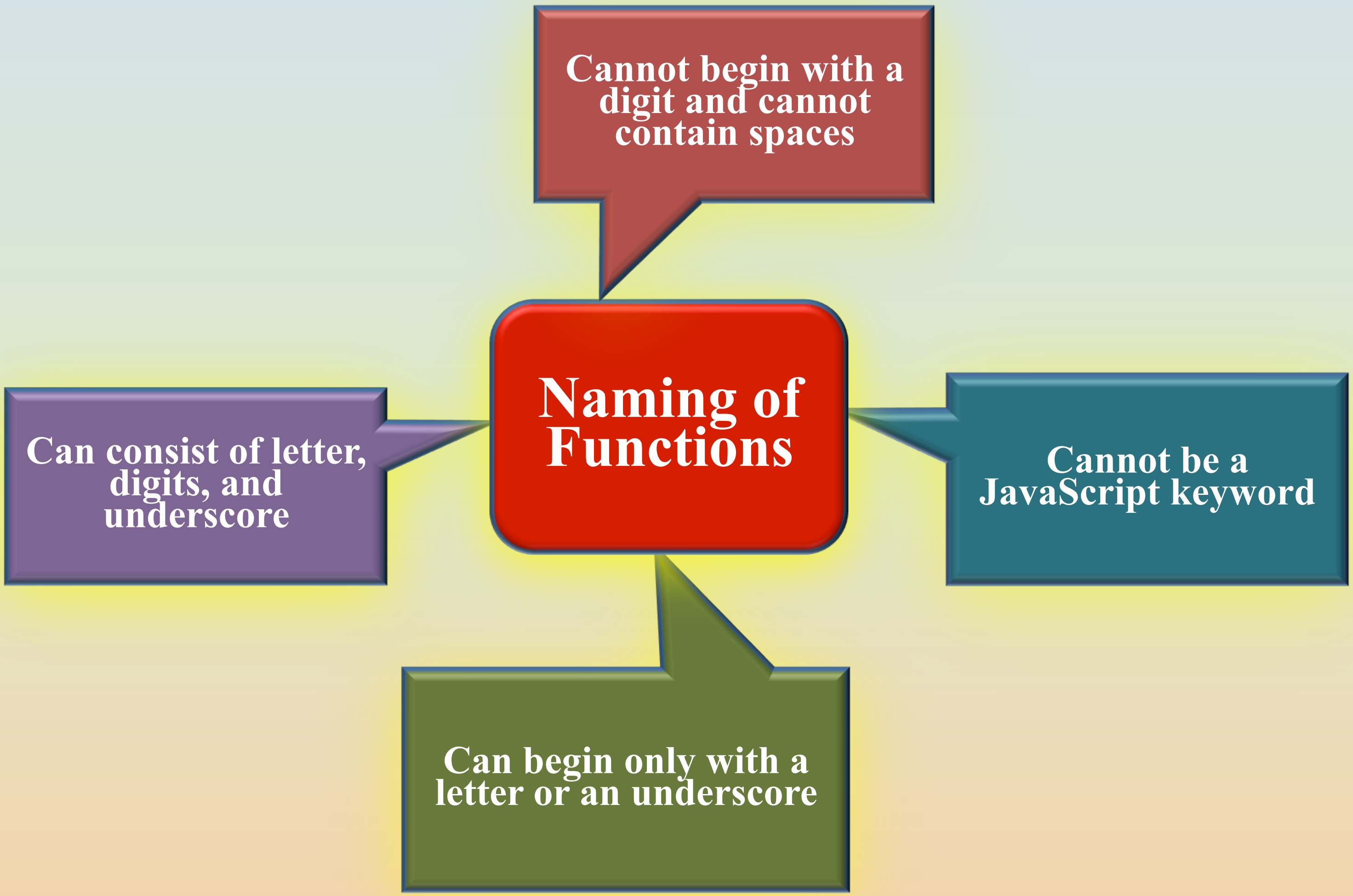
If the function does not accept any parameters, then it must be specified with an empty parenthesis.

Once the function is declared, you need to define the function by specifying the operations or instructions within the curly braces "{" and "}".

Curly braces indicate the start and end of the function block, which is collectively referred to as the body of the function.

A function must be defined before it can be invoked in the script and multiple functions can be defined within the script element.

Declaring and Defining Functions 2-3

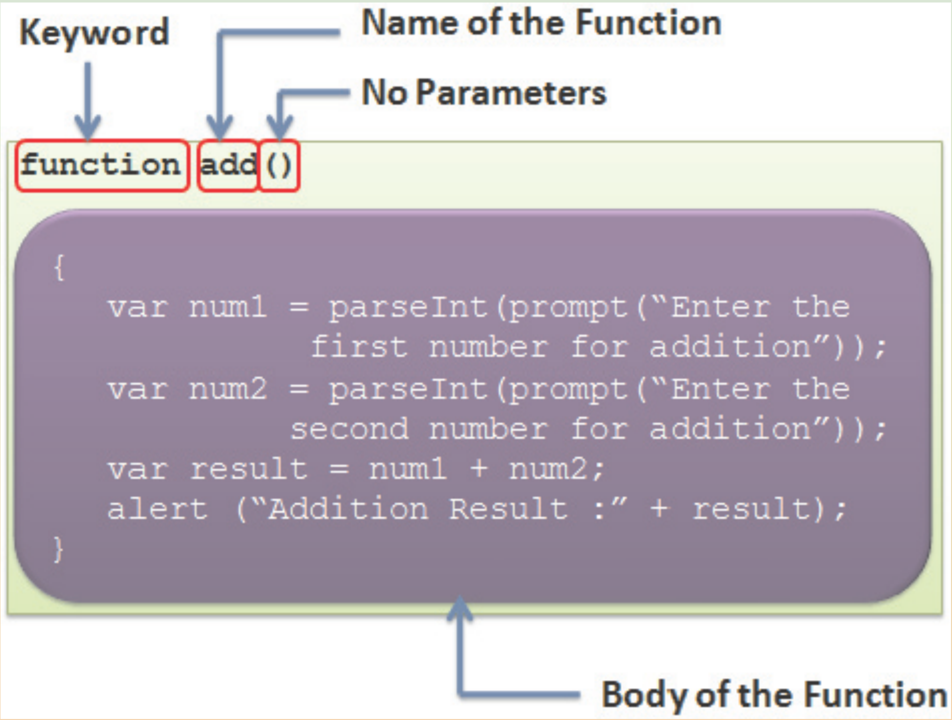


© Aptech Ltd.

Declaring and Defining Functions 3-3

- Syntax to create a function in JavaScript is as follows:

```
function function_name(list of parameters)
{
// Body of the function
}
```



Invoking Functions 1-2

A function need to be invoked or called to execute it in the browser.

To invoke a function, specify the function name followed by parenthesis outside the function block.

A function can be defined and invoked even in an external JavaScript file.

A function can be called from another function in JavaScript.

A function that invokes another function is called the calling function; whereas the function that is called is referred to as the called function.

Functions provide the benefit of code reusability by allowing the user to call a function multiple times.

© *Aptech Ltd.*

Invoking Functions 2-2

```
<script>
function add()
{
   var num1 = parseInt(prompt("Enter the
              first number for addition"));
   var num2 = parseInt(prompt("Enter the
             second number for addition"));
   var result = num1 + num2;
   alert ("Addition Result :" + result);
}

function calling_add()
{
 add();
}

calling_add();

</script>
```

Invoking the calling_add() Function

Called Function

Calling Function

Parameterized Functions

Parameterized Functions

Refer to the JavaScript functions that accept parameters

Can be created when there is a need to accept values from the user

Parameters hold values on which the function needs to perform operations

Takes two numbers and stores it in val1 and val2

```
<script>
   var val1 = parseInt(prompt("Enter the first
              number for addition"));
   var val2 = parseInt(prompt("Enter the second
              number for addition"));
add(val1, val2);

function add(num1, num2)
{
   var result = num1 * num2;
  alert("Addition Result: " + result);
}
</script>
```

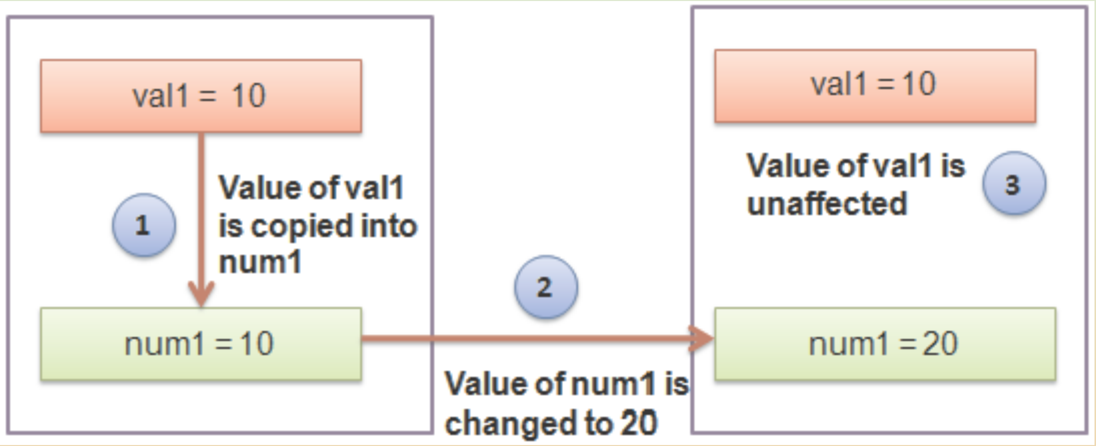
Invokes the add() function by passing val1 and val2

Holds the value of val1 and val2

Ways of Passing Arguments 1-3

There are two ways of passing arguments to a function namely, pass by value and pass by reference.

Passing by value - Refers to passing variables as arguments to a function. Called function do not change the values of the parameters passed to it from the calling function.



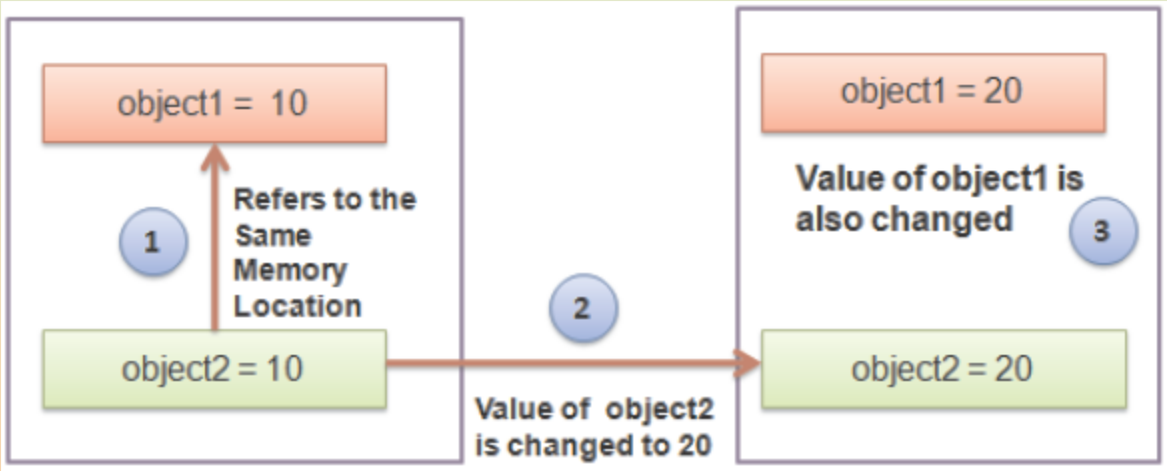
© Aptech Ltd.

Ways of Passing Arguments 2-3

Passing by reference - Refers to passing objects as arguments to a function.

The called function modifies the value of the parameters passed to it from the calling function.

This change is reflected when the control passes back to the calling function.



© Aptech Ltd.

Ways of Passing Arguments 3-3

- The Code Snippet demonstrates passing of Array object as a parameter to a function.

```
<script>
  var names =new Array('James', 'Kevin', 'Brad');
  function change_names(names){
    names[0]= 'Stuart';
  }
  function display_names(){
    document.writeln('<H3> List of Student Names:</H3>');
    document.write('<UL>');
    for(vari=0; i<names.length; i++) {
      document.write('<LI>' + names[i]+ '</LI>');
    }
    document.write('</UL>');
    change_names(names);
    document.write('<H3> List of Changed Students Names:</H3>');
    document.write('<UL>');
    for(vari=0; i<names.length; i++) {
      document.write('<LI>' + names[i]+ '</LI>');
    }
    document.write('</UL>');
  }
  display_names(names);
</script>
```

return Statement 1-2

return Statement

Returns the control to the calling function because of unexpected results

Allows sending the result back to the calling function

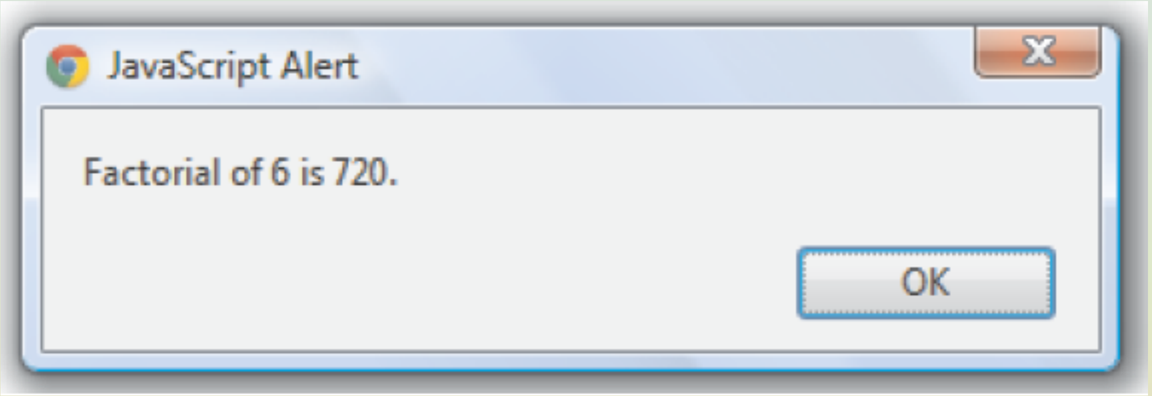
Can also be used to halt the execution of the function

Begins with return keyword followed by the variable or value

return Statement 2-2

- The Code Snippet demonstrates the use of `return` statement.

```
<script>
  function factorial(num) {
    if(num==0)
      return0;
    elseif(num==1)
      return1;
    else
    {
      var result =num;
      while(num>1)
      {
        result = result *(num-1);
        num--;
      }
      return result;
    }
  }
  varnum=prompt('Enter number:','');
  var result = factorial(num);
  alert('Factorial of ' +num+ ' is ' + result + '.');
</script>
```



© Aptech Ltd.

Objects 1-2

Are entities with properties and methods and resemble to real life objects.

Properties specify the characteristics or attributes of an object.

Methods identify the behavior of an object.

Object: Car

Properties	Make - ford Color - green Wheels – four
Methods	run() stop()

Object: Bird

Properties	Type - pigeon Color - gray Wings - two
Methods	eat() fly()

Objects 2-2

- JavaScript provides built-in objects and allows creating user-defined objects.

Built-in Objects - Are pre-defined objects which are already defined. Their properties and methods need to be called to fulfill a task.

Custom Objects - Are user-defined objects, which the developer explicitly creates in the script and defines their properties and methods.

© Aptech Ltd.

Creating Custom Objects 1-5

Object object is the parent object from which all the JavaScript objects are derived.

Custom objects can be derived from this object by using the **new** keyword.

Two main methods to create a custom object namely, direct method and by defining a template and initializing it with the **new** keyword.

Creating Custom Objects 2-5

- Syntax to create object using these methods is as follows:
- **Direct Method**

```
var object_name = new Object();
```

where,

`object_name`: Is the name of the object.

`new`: Is the keyword that allocates memory to the custom object. This is known as instantiation of an object.

`object`: Is the built-in JavaScript object that allows creating custom objects.

© *Aptech Ltd.*

Creating Custom Objects 3-5

- Template Method

An object's template refers to a structure that specifies the custom properties and methods of an object.

First, the object type is declared using a constructor function.

Second, you specify the object of the declared object type by using the **new** keyword.

JavaScript allows creating a reusable template without having to redefine properties and methods repeatedly to fulfill a particular object's requirements.

This template is known as the constructor function.

Creating Custom Objects 4-5

- Template Method

A constructor function is a reusable block that specifies the type of object, its properties, and its methods.

It might or might not take any parameters.

After creating the constructor function, you specify an object of the declared object type using the **new** keyword.

new keyword allocates memory to the object and invokes the constructor function.

- Syntax to create a constructor function is as follows:

```
function object_type(list of parameters)
{
// Body specifying properties and methods
}
```

© Aptech Ltd.

Creating Custom Objects 5-5

- Syntax to create a object using the `new` keyword is as follows:

```
object_name = new object_type(optional list of arguments);
```

- The Code Snippet shows the creation of objects using direct and template method is as follows:

```
<script>
  //create an object using direct method
  var doctor_details=new Object();
  //create an object using new keyword
  studOne = new student_info ('James', '23', 'New Jersey');
</script>
```

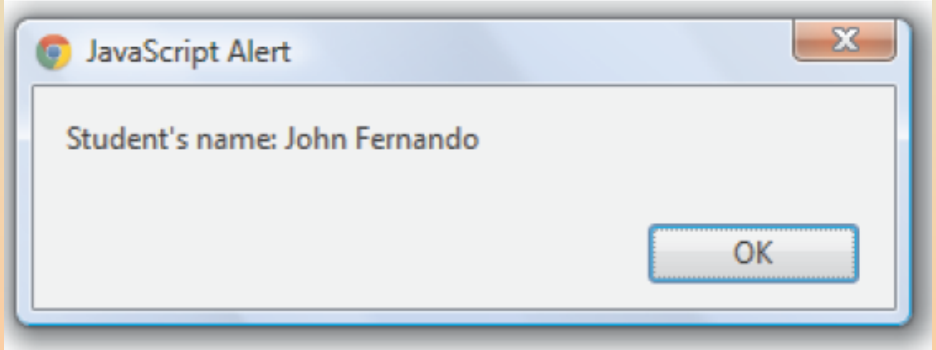
Creating Properties for Custom Objects 1-2

Properties specify the characteristics of an object created through Object or template method.

To access a property of an object created using Object object, specify the object name followed by a period and the property name.

- The Code Snippet demonstrates the script that creates the student_details object.

```
<script>
  var student_details=new Object();
  student_details.first_name= 'John';
  student_details.last_name= 'Fernando';
  student_details.age= '15';
    alert('Student\'s name:
+student_details.last_name);
</script>
```

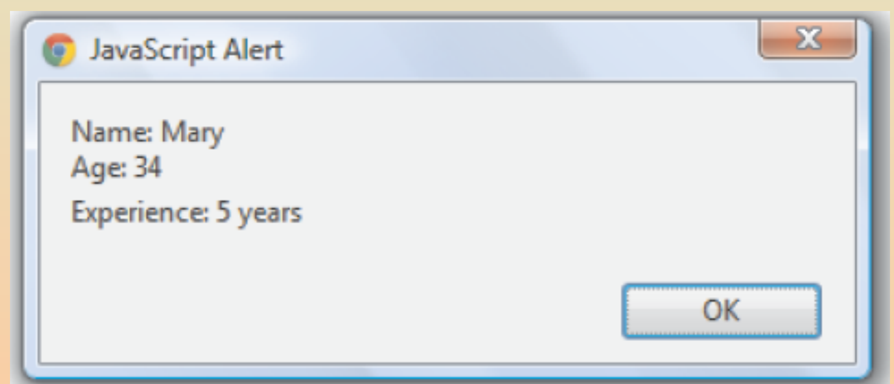


© Aptech Ltd.

Creating Properties for Custom Objects 2-2

- The Code Snippet creates the employee_info object and adds properties in the constructor function.

```
<script>
  // To define the object type
  function employee_info(name, age, experience)
  {
    this.name = name;
    this.age= age;
    this.experience= experience;
  }
  // Creates an object using new keyword
  empMary=newemployee_info(‘Mary’, ‘34’, ‘5 years’);
  alert(“Name: “+ empMary.name + ‘\n’ +”Age: “+empMary.age+ ‘\n’
+”Experience: “+empMary.experience);
</script>
```



© Aptech Ltd.

Creating Methods for Custom Objects

Methods are similar to JavaScript functions.

A method is associated with an object and is executed by referring to that object but a function is not associated with any object and is executed independently.

One or more methods can be specified after creating an object using the Object object or while creating the template.

To invoke a method, they must be specified with the object name followed by a period, method name, and parenthesis with parameters, if any.

- The Code Snippet demonstrates the script that defines a custom method.

```
<script>
  var square =new Object();
  square.length=parseInt("5");
  square.cal_area=function() {
    var area =(parseInt(square.length)*parseInt("4"));
    return area;
  }
  alert("Area: "+square.cal_area());
</script>
```

© Aptech Ltd.

Built-in Objects

Object model of JavaScript language forms the foundation of the language.

These objects help to provide custom functionalities in the script.

JavaScript treats the primitive data types as objects and provide equivalent object for each of them.

JavaScript objects are categorized as built-in objects, browser objects, and HTML objects.

Built-in objects are static objects which can be used to extend the functionality in the script.

Browser objects, such as window, history, and navigator are used to work with the browser window.

HTML objects, such as form, anchor, and so on are used to access elements on the Web page.

© Aptech Ltd.

String Object 1-3

Strings in JavaScript are a set of characters that are surrounded by single or double quotes.

Built-in String object allows you to perform different text operations on them.

String object is instantiated with the **new** keyword, which invokes the predefined constructor function of the String object.

- Syntax to initialize the `String` object is as follows:

```
var object_name = new String(“Set of characters”) ;
```

- Following table lists properties of the `String` object.

Property	Description
length	Retrieves the number of characters in a string.
prototype	Adds user-defined properties and methods to the String instance.

String Object 2-3

- Following table lists the methods of the `String` object.

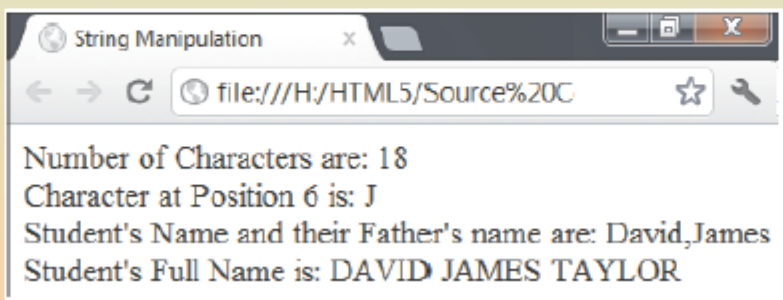
Method	Description
charAt()	Retrieves a character from a particular position within a string.
concat()	Merges characters from one string with the characters from another string and retrieves a single new string.
indexOf()	Retrieves the position at which the specified string value first occurred in the string.
lastIndexOf()	Retrieves the position at which the specified string value last occurred in the string.
replace()	Matches a regular expression with the string and replaces it with a new string.
search()	Searches for a match where the string is in the same format as specified by a regular expression.
split()	Divides the string into substrings and defines an array of these substrings.
substring()	Retrieves a part of a string between the specified positions of a string.
toLowerCase()	Specifies the lowercase display of the string.

© Aptech Ltd.

String Object 3-3

- The Code Snippet demonstrates the script that creates a `String` object and test various methods.

```
<script>
  var full_name=new String('David James Taylor');
  document.write('Number of Characters are: ' +full_name.length+
'<BR/>');
    document.write('Character at Position 6 is: '
+full_name.charAt(6)+ '<BR/>');
   document.write('Student\'s Name and their Father\'s name are:
' +full_name.split(' ',2)+ '<BR/>');
    document.write('Student\'s Full Name is: '
+full_name.toUpperCase());
</script>
```



© Aptech Ltd.

Math Object 1-2

Math object allows the user to perform mathematical operations on numeric values.

Math object is a pre-defined object that provides static properties and methods to perform mathematical operations.

Properties and methods are declared as static, thus they can be invoked directly with the object name.

- Syntax to access the properties of the `Math` object is as follows:

```
var variable_name = Math.PropertyName;
```

- Syntax to invoke the methods of the `Math` object is as follows:

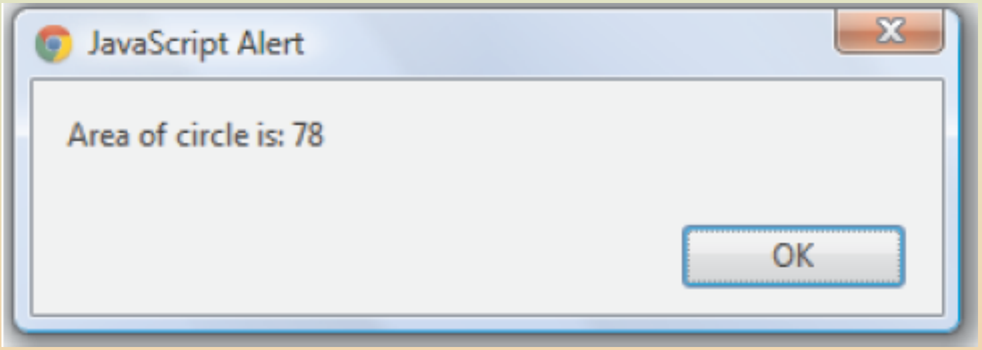
```
var variable_name = Math.MethodName(optional list of parameters);
```

© Aptech Ltd.

Math Object 2-2

- The Code Snippet demonstrates the script that creates a `Math` object.

```
<script>
  var full_name=new String('David James Taylor');
  document.write('Number of Characters are: ' +full_name.length+
'<BR/>');
  var area =Math.floor(tempArea);
  return area;
  }
  alert('Area of circle is: ' +area_circle(5));
</script>
```



Date Object 1-3

Date object allows you to define and manipulate the date and time values programmatically.

It supports both the Universal Time Coordinated (UTC) and Greenwich Mean Time (GMT) conventions.

Date object calculates dates in milliseconds from 01 January, 1970.

- Syntax to instantiate the `Date` object is as follows:

```
var object_name = new Date();
var object_name = new Date(milliseconds);
var object_name = new Date(year, month, day, hour, minutes,
seconds, milliseconds);
var object_name = new Date("dateString");
```

© Aptech Ltd.

Date Object 2-3

- Following table lists the methods of the `Date` object.

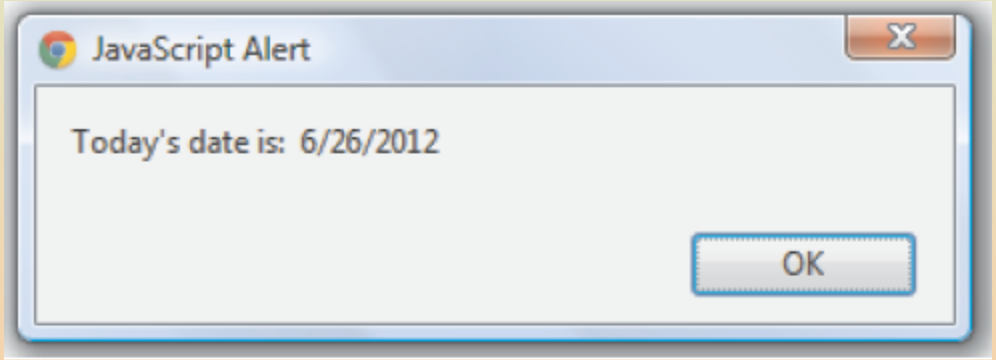
Method	Description
getDate()	Retrieves a numeric value between 1 and 31, which indicates the day of the month.
getDay()	Retrieves a numeric value between 0 and 6, which indicates the day of the week.
getTime()	Retrieves a numeric value, which indicates the time passed in milliseconds since midnight 01/01/1970.
getFullYear()	Retrieves a four digit numeric value, which indicates the year in the given date.

© Aptech Ltd.

Date Object 3-3

- The Code Snippet demonstrates the script that displays the current date in the mm/dd/yyyy format.

```
<script>
  function display_date()
  {
    var today =new Date();
    var date =today.getDate();
    var month =today.getMonth();
    month++;
    var year =today.getFullYear();
      alert(‘Today\’s date is: ‘ + month + ‘/’ + date + ‘/’ +
year);
  }
  display_date();
</script>
```



© Aptech Ltd.

with Statement

with statement allows to remove the object reference for each JavaScript statement.

with statement starts with the **with** keyword followed by the open and close brackets, which holds the statements that refer to a common object.

with statement increases the readability of the code and also reduces time required in writing each object reference in every related statement.

- Syntax to declare the `with` statement is as follows:

```
with(object_name)
{
//Statements
}
```

Refers to the car Object

Refers to the Property of the car Object

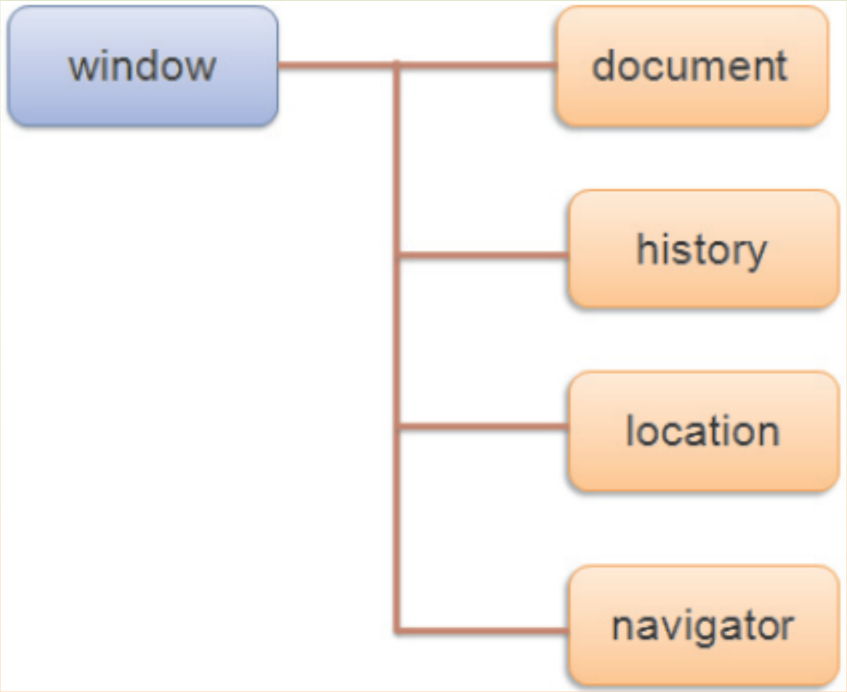
```
function displayCarInformation()
{
    var car = new object();
    car.price = '$45000';
    car.mileage = '45 miles/liters';

  with (car)

   {
    document.write ('<P> Price: ' + price + '</P>');

    document.writeln ('<P> Mileage: ' + mileage +
                                          '</P>');
   }
}
displayCarInformation();
```

Browser Objects

JavaScript also provides objects to access and manipulate various aspects of the Web browser.

These objects are called as browser objects.

They exist on all pages displayed in the browser and correspond to elements of a page.



window Object 1-4

window Object

- Provides properties that allows setting a default text for the status bar, name of the browser window, and so on
- Is the top level object in the JavaScript hierarchy
- window object represents a browser window and contains browser information
- All the objects in the hierarchy are descendants of the window object

window Object 2-4

- Following table lists the commonly used properties of the `window` object.

Property	Description
defaultStatus	Specifies or retrieves the default text to be displayed in the status bar of the browser window.
document	Represents an HTML document that contains different elements.
history	Contains history of the visited Uniform Resource Locators (URLs).
location	Contains the content of the specified URL.

© *Aptech Ltd.*

window Object 3-4

- Following table lists the methods of the `window` object.

Method	Description
alert()	Displays an alert box that states the message and an OK button.
confirm()	Prompts a dialog box that displays a message with the OK and Cancel buttons.
createPopup()	Creates a pop-up window.
focus()	Focuses the current window.
open()	Opens the specified file in a new browser window.
prompt()	Prompts a dialog box that accepts input from the user.

© Aptech Ltd.

window Object 4-4

- The Code Snippet demonstrates the methods of the `window` object.

```
<!DOCTYPE html>
<head>
  <title> window Object </title>
  <script>
  function new_window() {
    if(confirm('Do you want to open a new page?')) {
        window.open('http://www.aptech-education.com/pages/about-
us/about-aptech.html', '_parent');
    }
    else {
      window.alert('In the Current Window');
    }
  }
  </script>
</head>
<body>
    <input type="button" value="Click to move to the next
page"onClick="new_window();"/>
</body>
</html>
```

history Object 1-2

history Object

- Allows you to determine the number of URLs in the history list by using the length property
- Is a part of the window object
- Is an array that allows referring to a particular URL by specifying its index number in the array
- Contains a set of URLs visited by a user in a browser window

history Object 2-2

- Following table lists the methods of the `history` object.

Method	Description
back()	Retrieves and displays the previous URL from the history list.
forward()	Retrieves and displays the next URL from the history list.
go()	Displays the specified URL. It accepts a parameter, which can either be a string or a number to go to specific page.

navigator Object 1-2

Contains information about the browser used by the client

navigator Object

Allows the user to retrieve information, such as name, version number, and so on

- Following table lists the properties of the `navigator` object.

Property	Description
appName	Retrieves the name of the browser.
appVersion	Retrieves the version number and platform of the browser.
browserLanguage	Retrieves the language of the browser.
cookieEnabled	Determines whether the cookies are enabled in the browser.
platform	Retrieves the machine type such as Win32, of the client browser.

© Aptech Ltd.

navigator Object 2-2

- The Code Snippet demonstrates the use of `navigator` object to retrieve information of the browser.

```
<!DOCTYPE html>
<head>
  <title> navigator Object </title>
  <script>
  function display_browser() {
    document.write('Browser name: ' +navigator.appName+ '<BR/>');
      document.write('Browser version: ' +navigator.appVersion+ '<BR/
>');
      document.write('Browser language: ' +navigator.browserLanguage+
'<BR/>');
    document.write('Platform: ' +navigator.platform+ '<BR/>');
    if(navigator.cookieEnabled) {
    document.write('Cookie is enabled in the browser.');
  }
}
</script>
</head>
<body>
  <input type="button" value="Browser Information"
onclick="display_browser()"/>
</body>
</html>
```

location Object

location Object

Allows to access complete information of the URL loaded in the browser window

Is a part of the Window object

- Following table lists the properties and methods of the `location` object.

Property/Method	Description
host	Retrieves hostname and port number of the URL.
href	Specifies or retrieves the entire URL.
pathname	Specifies or retrieves the path name of the URL.
assign()	Loads a new document with the specified URL.
reload()	Reloads the current document by again sending the request to the server.
replace()	Overwrites the URL history for the current document with the new document.

© *Aptech Ltd.*

Document Object Model 1-3

A Web page contains various elements, such as buttons, text boxes, check boxes, and so on.

These elements exist in a hierarchy and overall represent an HTML document.

JavaScript allows the user to access HTML elements and also change the existing structure of an HTML page by using Document Object Model (DOM) specification.

DOM is an Application Programming Interface (API) that defines the object structure for accessing and manipulating HTML elements.

Is used with JavaScript to add, modify, or delete elements and contents on the Web page.

DOM specifications are laid by World Wide Web Consortium (W3C) and are implemented by all the browsers to overcome incompatibility issues.

DOM reads all the elements contained in an HTML page and treats the HTML elements as nodes.

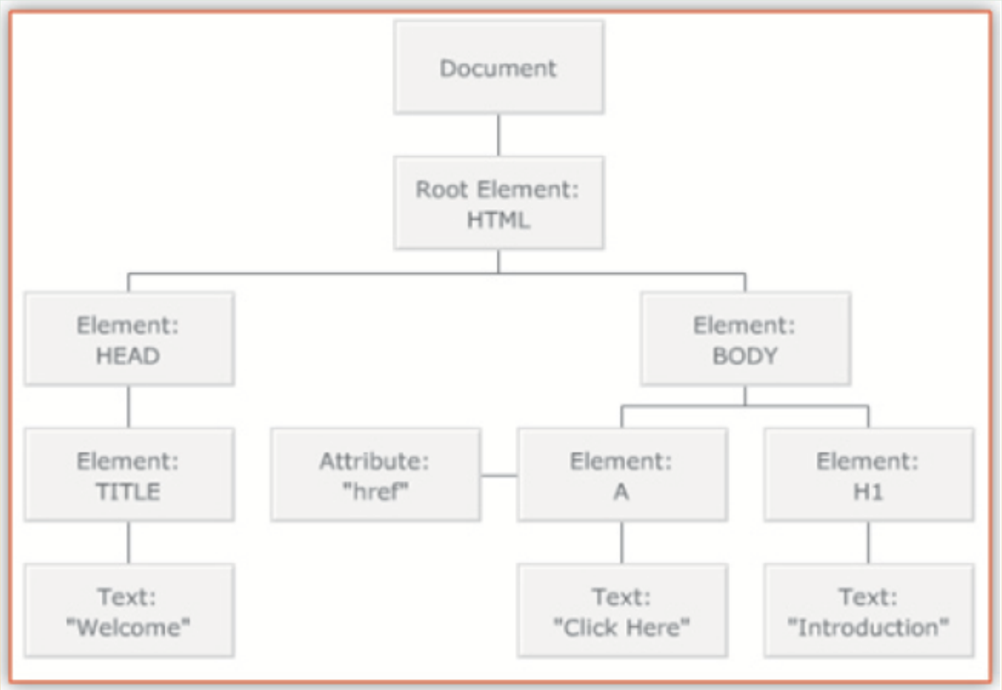
The entire HTML document represents a document node. This document node consists of element nodes, attribute nodes, and text nodes. Document node is the highest level node and text nodes are the lowest ones.

© Aptech Ltd.

Document Object Model 2-3

- The Code Snippet shows an HTML document.

```
<!DOCTYPE html>
  <html lang="en">
  <head>
    <title>Welcome</title>
  </head>
  <body>
    <h1> Introduction </h1>
    <ahref="#">Click Here</a>
  </body>
</html>
```



Document Object Model 3-3

- All the nodes present in the node hierarchy contain certain properties.
- These properties provide information about the node. The different node properties are as follows:

nodeName - Represents the name of the node. It contains the tag name of the HTML element in upper case.

nodeValue - Represents the text contained within the node. This property is only available for attribute nodes and not for document and element nodes.

nodeType - Represents the type of the node. For example, the document node, element node, and so on.

- HTML DOM provides standard objects for HTML documents and some of these objects are as follows:
    - Document object
    - Form object
    - Link object
    - Table object

© Aptech Ltd.

Document Object 1-3

Is used within the JavaScript to access all HTML elements presented on the page.

Represents the entire HTML document and provides access to other elements, such as links, anchors, and so on.

Has only one document object which is created when the BODY element is loaded on the Web page.

Is also the part of the window object and is accessed as, **window.document**.

Provides properties that allow the user to specify or retrieve the information about the elements and its content.

© *Aptech Ltd.*

Document Object 2-3

- Following table lists the commonly used properties of the `document` object.
- Following table lists the commonly used methods of the `document` object.

Method	Description
close()	Closes a data stream and displays the data collected using the open() method.
getElementById()	Retrieves a collection of HTML elements by using the specified ID.
getElementsByName()	Retrieves a collection of HTML elements by using the specified name.
getElementsByTagName()	Retrieves a collection of HTML elements with the specified tag name.
open()	Opens a stream to accept the output from write() or writeln() methods.
write()	Writes the text or HTML expression to a document.

© Aptech Ltd.

Document Object 3-3

- The Code Snippet demonstrates how to use the document object.

```
<!DOCTYPE html>
<head> <title> Document Object </title>
  <script>
  function change_image() {
    var imgText=document.getElementById('myImg').alt;
    if(imgText=="ford") {
      document.getElementById('myImg').src="ferrari.jpg";
      document.getElementById('myImg').alt ="ferrari";
      document.getElementById('mytext').value ="Ferrari Car";
    } else {
      document.getElementById('myImg').src="ford.jpg";
      document.getElementById('myImg').alt ="ford";
      document.getElementById('mytext').value ="Ford Car";
    } }
  </script> </head>
<body>
  <imgid="myImg" src="ford.jpg" width="300" height="300" alt="ford" /
  > <br/>
      Model: <input type="text" id="mytext" value="Ford Car"
  readonly="readonly"/><br/><br/>
<input type="button" value="Change Image" onclick="change_image()"/>
</body>
```

Form Object 1-2

Accepts input from the user and sends the user data for validation.

A single HTML document can contain multiple forms.

DOM specification provides a form object that represents an HTML form which is created for each <form> tag in an HTML document.

Form Object 2-2

- The Code Snippet demonstrates how to use the `form` object.

```
<!DOCTYPE html>
<head>
  <title> Form Object </title>
  <script>
    functiondisplay_length() {
      var count =document.getElementById("form1").length;
      alert('Number of controls on the form: ' + count);
    }
  </script>
</head>
<body>
  <form id="form1" action="welcome.php">
      First name: <input type="text" name="firstname" value="John" /
   ><br />
      Last name: <input type="text" name="lastname" value="Smith" /
   ><br />
    Age : <input type="text" name="age" value="40" /><br />
        <input type="button" value = "Controls"
   onClick="display_length()"/>
  </form>
</body>
</html>
```

HTML

Summary

- A function is reusable piece of code, which performs calculations on parameters and other variables.
- The return statement passes the resultant output to the calling function after the execution of the called function.
- Objects are entities with properties and methods and resemble to real life objects.
- There are two ways to create a custom object namely, by directly instantiating the Object object or by creating a constructor function.
- JavaScript provides various built-in objects, such as String, Math, and Date.
- JavaScript also provides browser objects, such as window, history, location, and navigator.
- DOM is a standard technique for dynamically accessing and manipulating HTML elements. The DOM provides a document object which is used within the JavaScript to access all HTML elements presented on the page.

© *Aptech Ltd.*