Session: 12

JavaScript - II

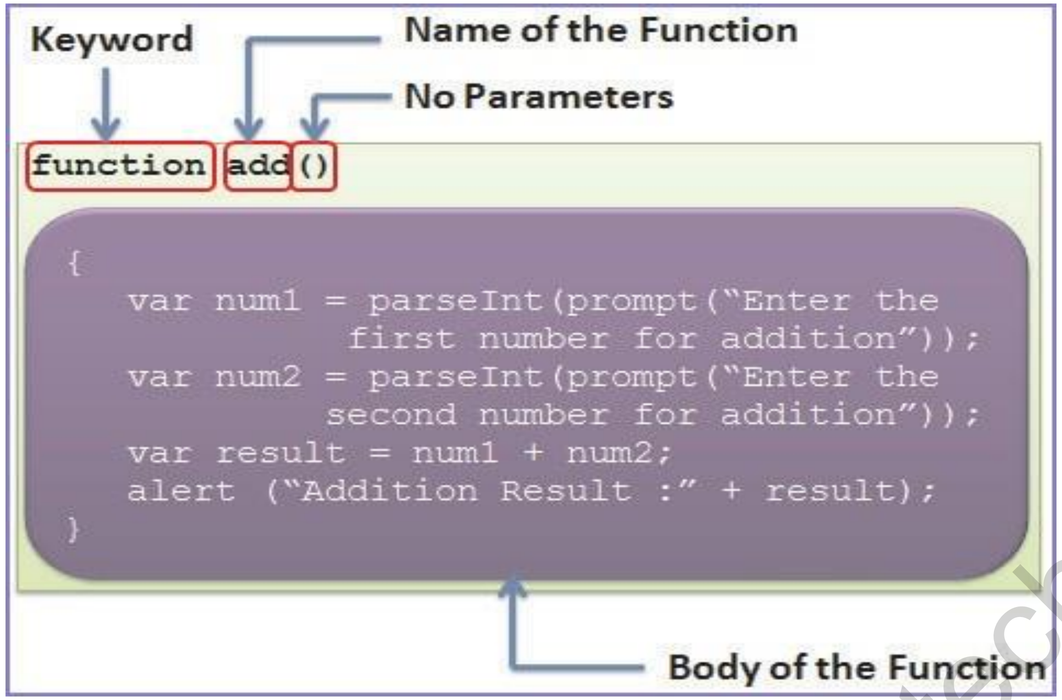
Objectives

- Explain functions
- Explain parameterized functions
- Explain return statement
- Describe objects
- Explain different browser objects
- Describe DOM and its objects
- Identify the use of Promise.any
- Explain Private class methods
- Explain JSON

Functions 1-3

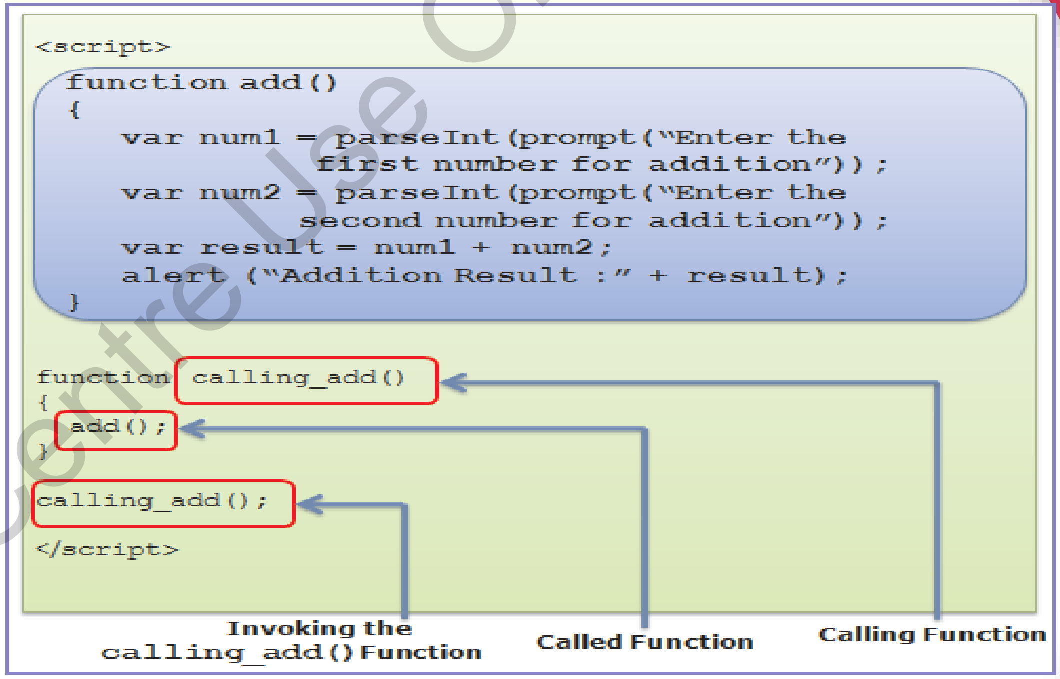
- A function is an independent reusable block of code that performs certain operations.
- It is always created under `script` element.
- A function is declared using `function` keyword.
- The keyword is followed by the name of the function and parameters enclosed within the parenthesis.
- A function needs to be invoked.
  - To invoke a function, specify the function name followed by parenthesis outside the function block.

Functions 2-3



```
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);
}
```

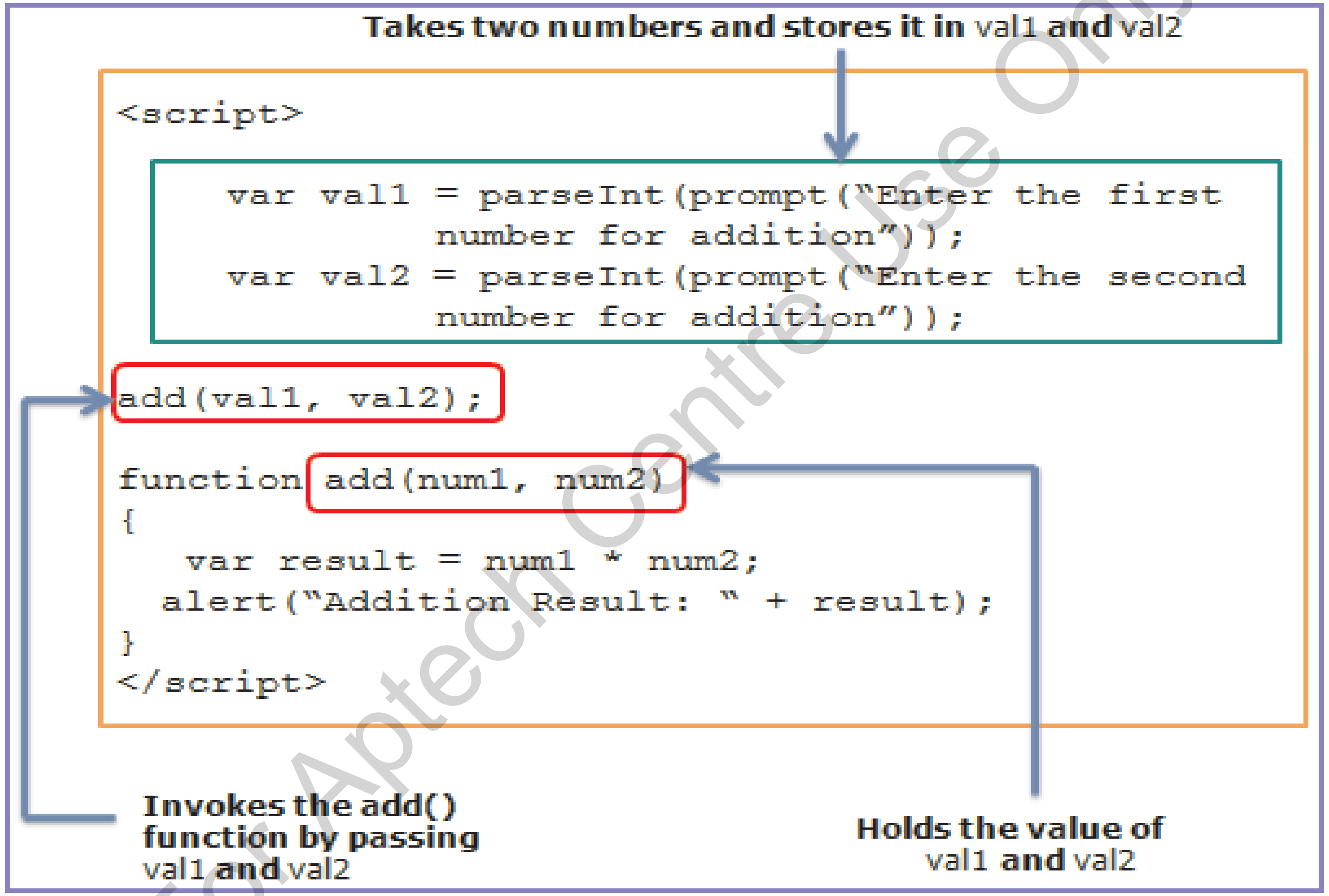
Declaration and Definition of a Function



```
<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 of Function

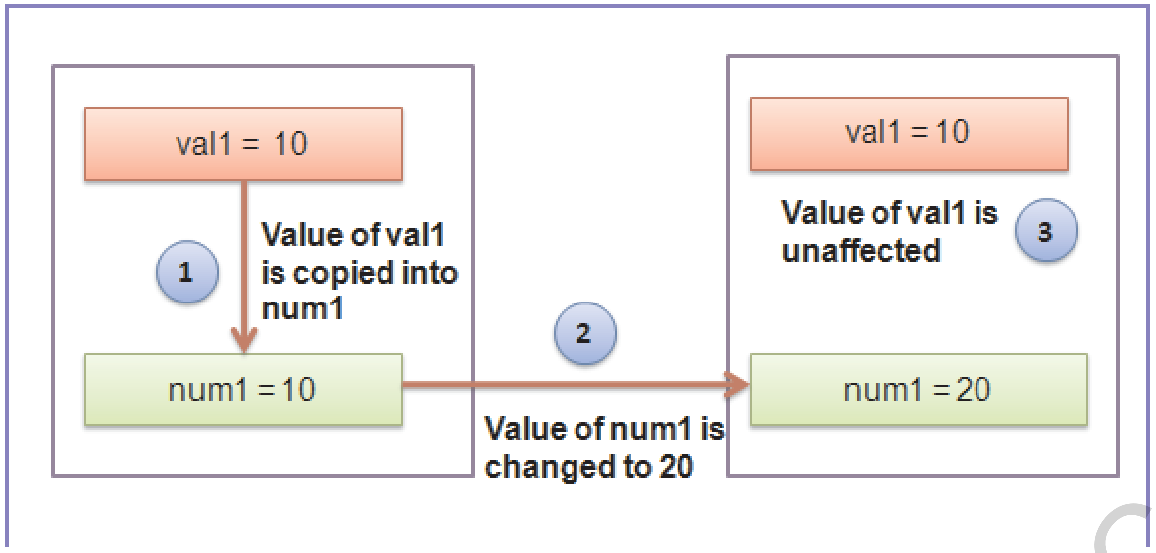
Functions 3-3


Takes two numbers and stores it in val1 and val2
Invokes the add() function by passing val1 and val2
Holds the value of val1 and val2

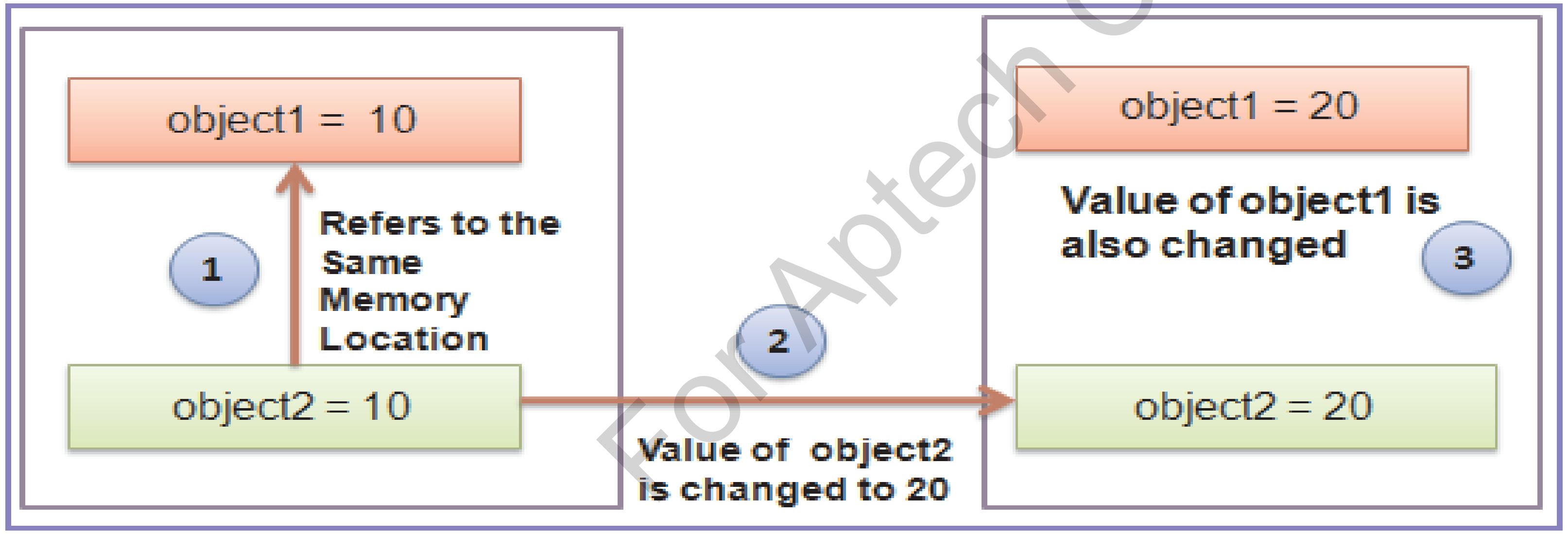
```
<script>

    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>
```

Parameterized Functions

Ways of Passing Arguments 1-2


val1 = 10
1
Value of val1 is copied into num1
num1 = 10
2
Value of num1 is changed to 20
val1 = 10
Value of val1 is unaffected
3
num1 = 20

Pass By Value Method

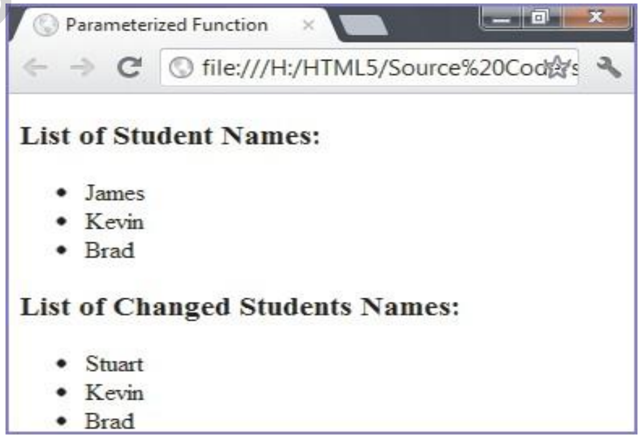

object1 = 10
1
Refers to the Same Memory Location
object2 = 10
2
Value of object2 is changed to 20
object1 = 20
Value of object1 is also changed
3
object2 = 20

Pass By Reference Method

Ways of Passing Arguments 2-2

```
<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(var i=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(var i=0; i<names.length; i++){
document.write('<li>' + names[i]+ '</li>');
}
document.write('</ul>');
}
display_names(names);
</script>
```

Passing an Array Object to Function



List of Student Names:

- James
- Kevin
- Brad

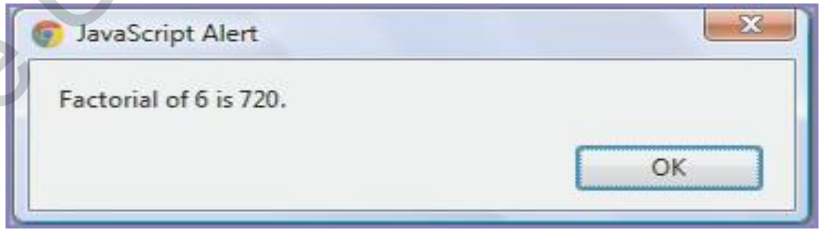
List of Changed Students Names:

- Stuart
- Kevin
- Brad

`return` Statement

JavaScript sends result to the calling function by using `return` statement.

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

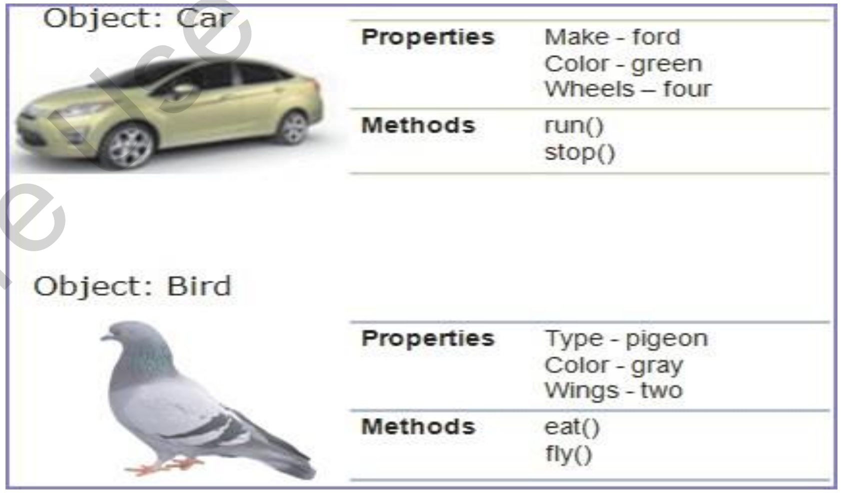


Factorial of 6 is 720.

Factorial of Number

Objects

- Objects are entities with properties and methods.
  - Properties specify the characteristics or attributes of an object.
  - Methods identify the behavior of an object.
- Objects can be built-in or custom.


Object: Car

Properties	Make - ford Color - green Wheels – four

Methods	run()
	stop()

Object: Bird

Properties	Type - pigeon Color - gray Wings - two

Methods	eat()
	fly()

Creating Custom Objects

- The `object` object is the parent object.
  - All JavaScript objects are derived from this object.
- An object can be created using the built-in `Object` object or by defining a template.

Syntax using the built-in `Object` object:

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

Syntax using the template:

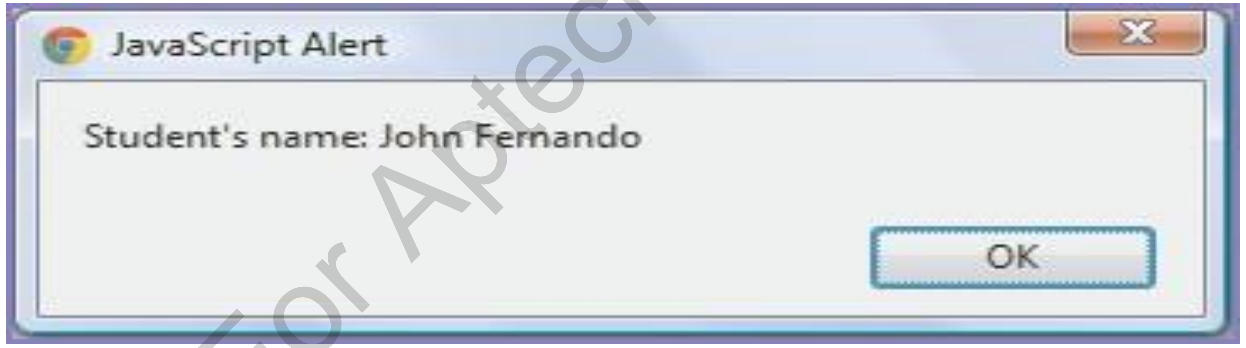
```
function object_type(list of parameters)
 {
   // Body specifying properties and methods
 }
```

Example:

```
<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

```
<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.first_name+ ' ' + student_ details.last_name);
</script>
```

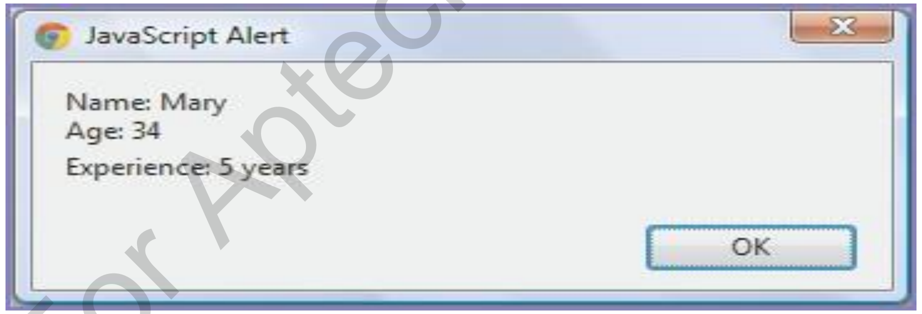


Student's name: John Fernando

student_details Object

Creating Properties for Custom Objects 2-2

```
<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 = new employee_info('Mary', '34', '5 years');
   alert ("Name: "+ empMary.name + '\n' +"Age: "+empMary.age+ '\n' +"Experience: "+empMary.experience);
</script>
```

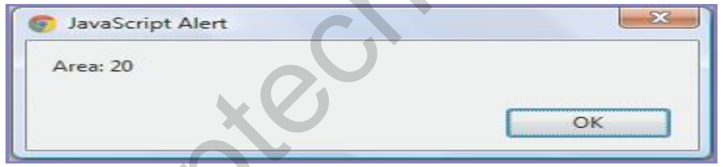

JavaScript Alert

```
Name: Mary
Age: 34
Experience: 5 years
```

employee_info Object

Creating Methods for Custom Objects

```
<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>
```



Area: 20

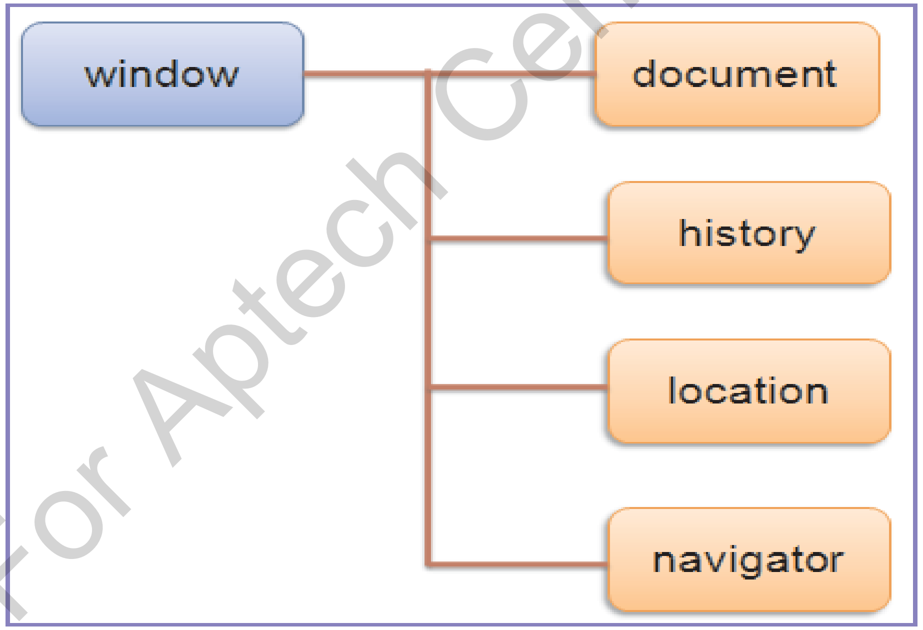
Output of the Area of Square

Built-in Objects

- The built-in objects are static objects.
- They help extend the functionality in the script.
- Some of these objects are: `String`, `Math`, and `Date`.

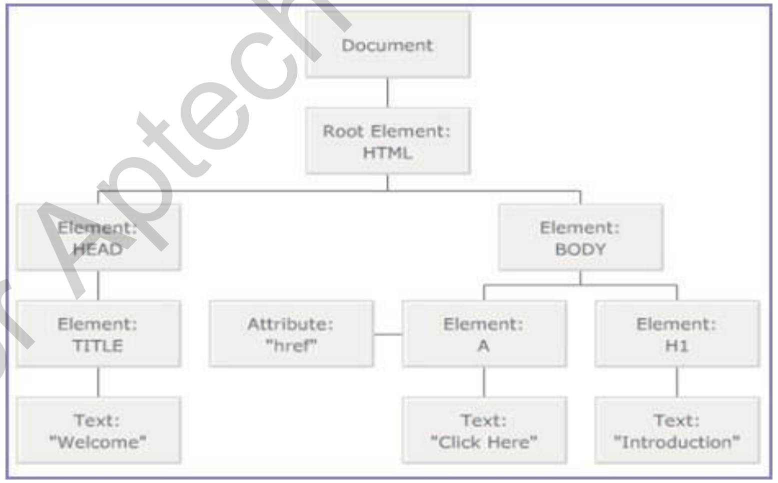
Browser Objects

- Browser objects help manipulate various aspects of a Web browser.
- They exist on all pages displayed in the browser.

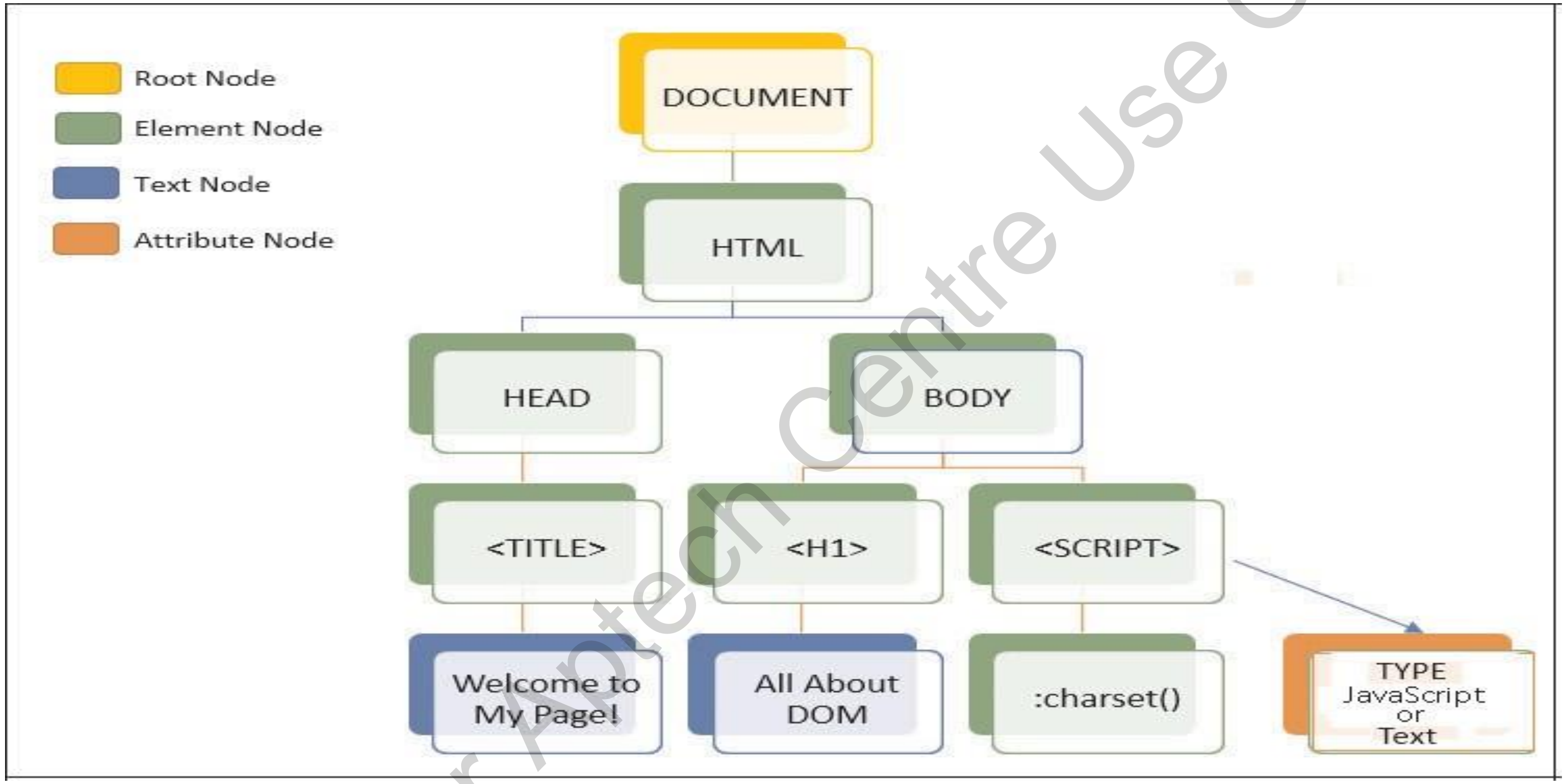


Document Object Model (DOM)

- DOM is a cross-platform and language-independent interface.
- It considers an XML or HTML document as a tree structure.
  - Each node is an object representing a part of the document.
- DOM represents a document with a logical tree.
  - Each branch of the tree ends in a node.
  - Each node contains objects.



DOM and JavaScript



New Features in JavaScript DOM

- Arrow functions help create functions in a simple manner.
- Arrow functions are useful to work with functions that require another function as an argument.

```
document.addEventListener("DOMContentLoaded" ,
   ()=>{ console.log("loaded");
})
```

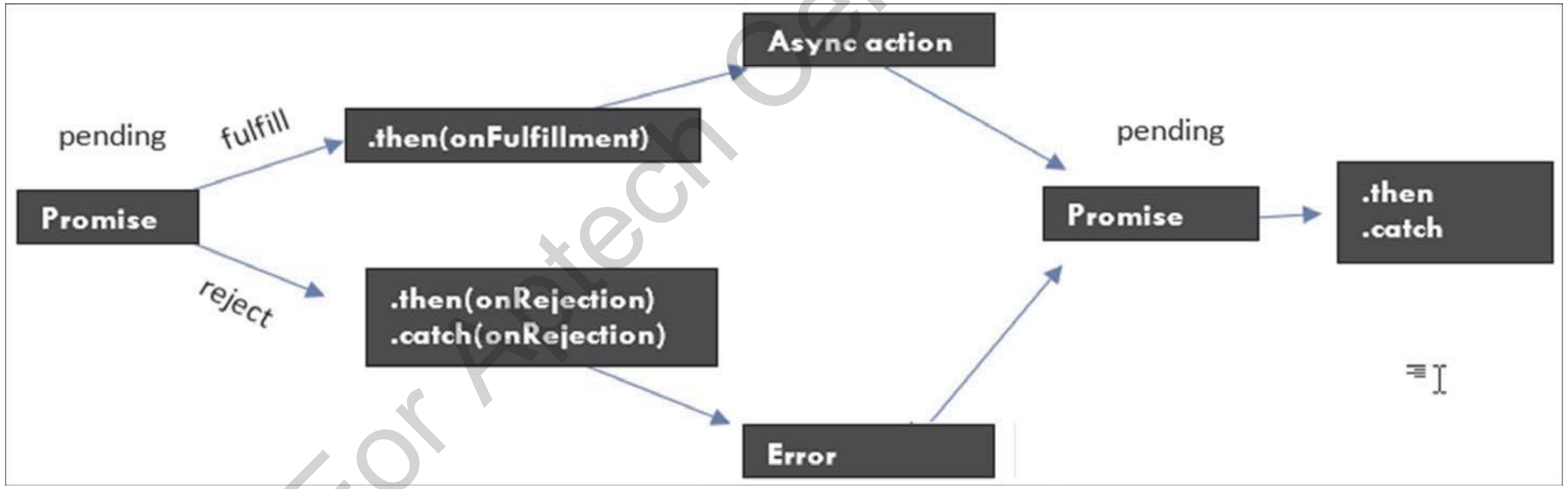
New Features in JavaScript DOM

- The `for of` loop statement creates a loop that repeats over iterable objects. such as arrays, maps, strings, and more.

```
const webFrameworks = ["React" , "Angular" , "Rails" ,
"Node.js"];
let text = "";
for (let x of webFrameworks) {
   text += x;
}
console.log(text);
```

JavaScript Promises

- Promises are a new feature in JavaScript.
- Promise represents eventual success or failure of an asynchronous operation.
- Promises can handle multiple asynchronous operations and provide better error handling.



Private Class Features

- A private method means only those objects belonging to the same class can access it.
- To declare a private class field, prefix the name of the class field with # (hash) tag.
- Private fields can be accessed on the class constructor from within the class declaration.

```
// Create new class class
MyClass {
 // Declare private class field
 #myPrivateField = 'This is a personal account.'
}
```

JavaScript Object Notation (JSON) 1-2

```
{
   "page":1,
   "results":[
      {
         "first_air_date":"2005-03-26",
         "genre_ids":[ 28,
            12,
            18,
            878
         ], "id":57243,
         "original_name":"Doctor Who",
                 "origin_country":[
            "GB"
         ],
         "name":"Doctor Who"
      },
      {
         "first_air_date":"2007-09-24",
         "genre_ids":[ 18,
            35
         ], "id":1418,
         "original_name":"The Big Bang Theory", "origin_country":[
            "US"
         ],
        "name":"The Big Bang Theory"
      },
```

JavaScript Object Notation (JSON) 2-2

```
    {
     "first_air_date":"2015-08-23",
         "genre_ids":[ 18,
            27
         ], "id":62286,
                "original_name":"Fear the Walking Dead", "origin_country":[
            "US"
         ],
         "name":"Fear the Walking Dead"
      }
       ],
   "total_pages":3116,
   "total_results":62309

}
```

JSON Serialization and Deserialization

Serialization - an object is converted into a string so that it can be recreated

Deserialization - a string is converted into an object

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 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 JavaScript to access all HTML elements presented on the page.