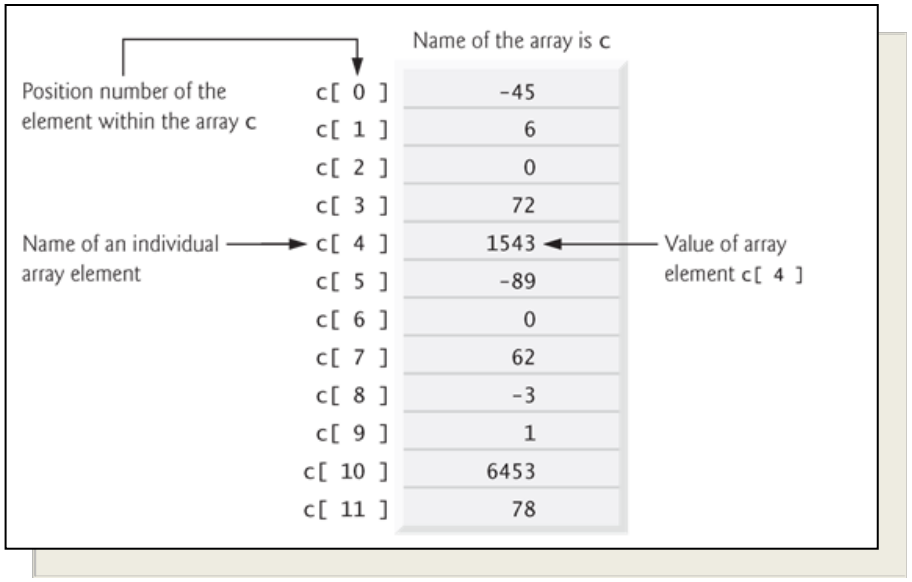
JavaScript
(Array, Function, and Objects)

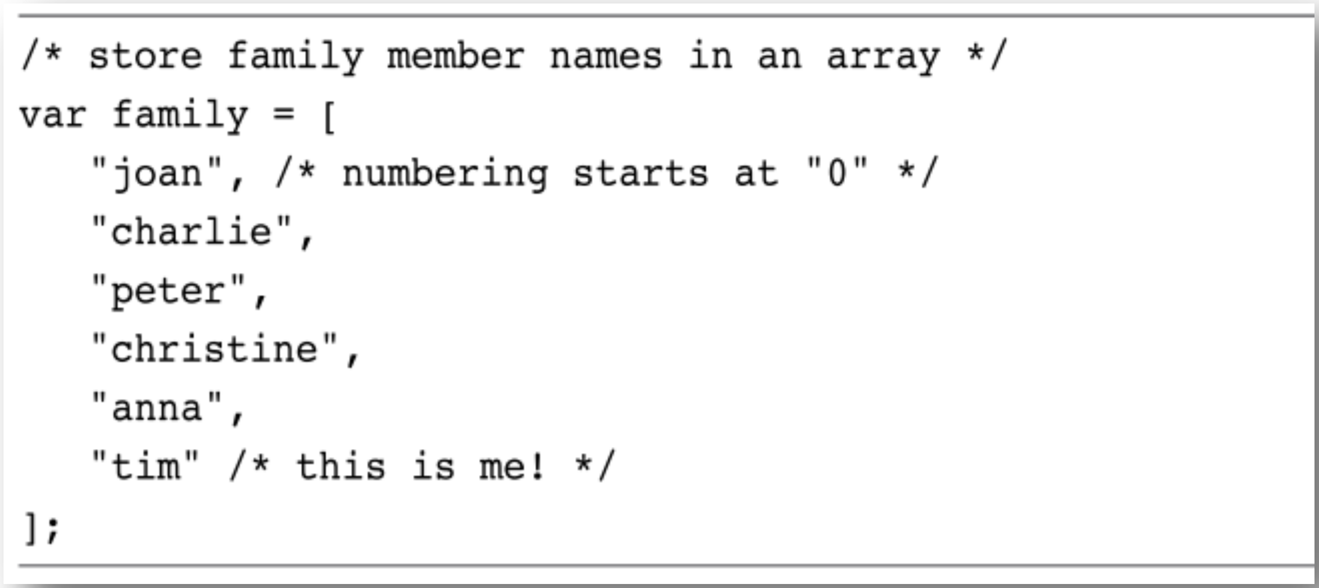
Arrays

- An array is an ordered collection of values. Each value is called an element, and each element has a numeric position in the array, known as its index..
- JavaScript arrays are dynamic entities that can change size after they are created.



Example1. Array with 12 Elements

Example:



```
/* store family member names in an array */
var family = [
   "joan", /* numbering starts at "0" */
   "charlie",
   "peter",
   "christine",
   "anna",
   "tim" /* this is me! */
];
```

Decalring an Arrays

- Using the new keyword.
  - var myList = new Array( );

- Using the shortcut [ ].
  - var myList = [ ];

Examle-1

- Var studentNames = new studentNames[ ];
  studentNames[0] = “Ahmad”;
  studentNames[1] = “Ali”;
  studentNames[2] = “Salma”;

- Var studentNames = [ ];
  studentNames[0] = “Ahmad”;
  studentNames[1] = “Ali”;
  studentNames[2] = “Salma”;

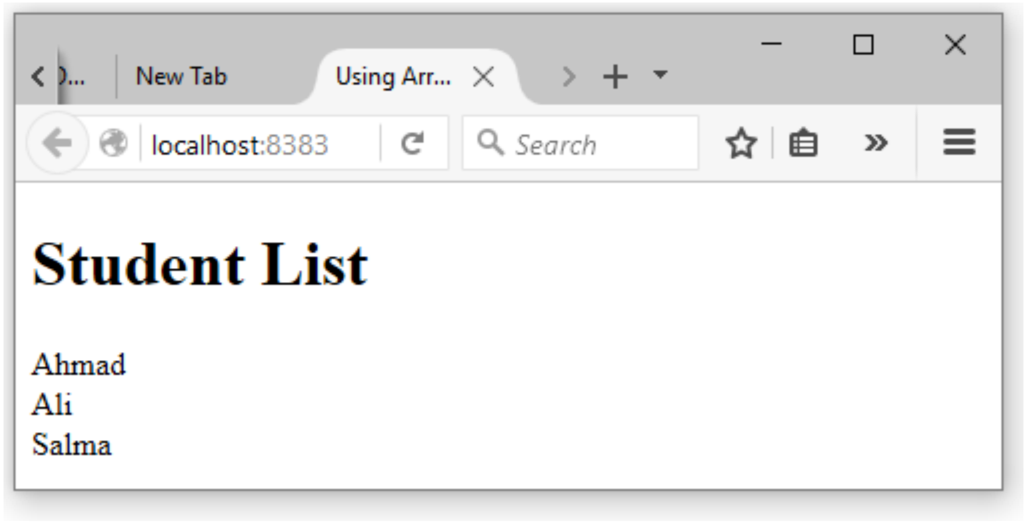
- Var studentNames = [“Ahmad”, ”Ali”, ”Salma”];

Examle-2

The following example uses an array to print student names

array.html

```
<!DOCTYPE html>
<!--. array.html-->
<html>
    <head>
        <title>Using Arrays</title>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
    <body>
        <div> <h1>Student List </h1> </div>
        <script>
       var names = ["Ahmad","Ali","Salma"];
       n = names.length;
       for ( var i = 0; i < n ; ++i ){
          document.write(names[i] +"<br>");
       }
     </script>
    </body>
</html>
```



```
Student List

Ahmad
Ali
Salma
```

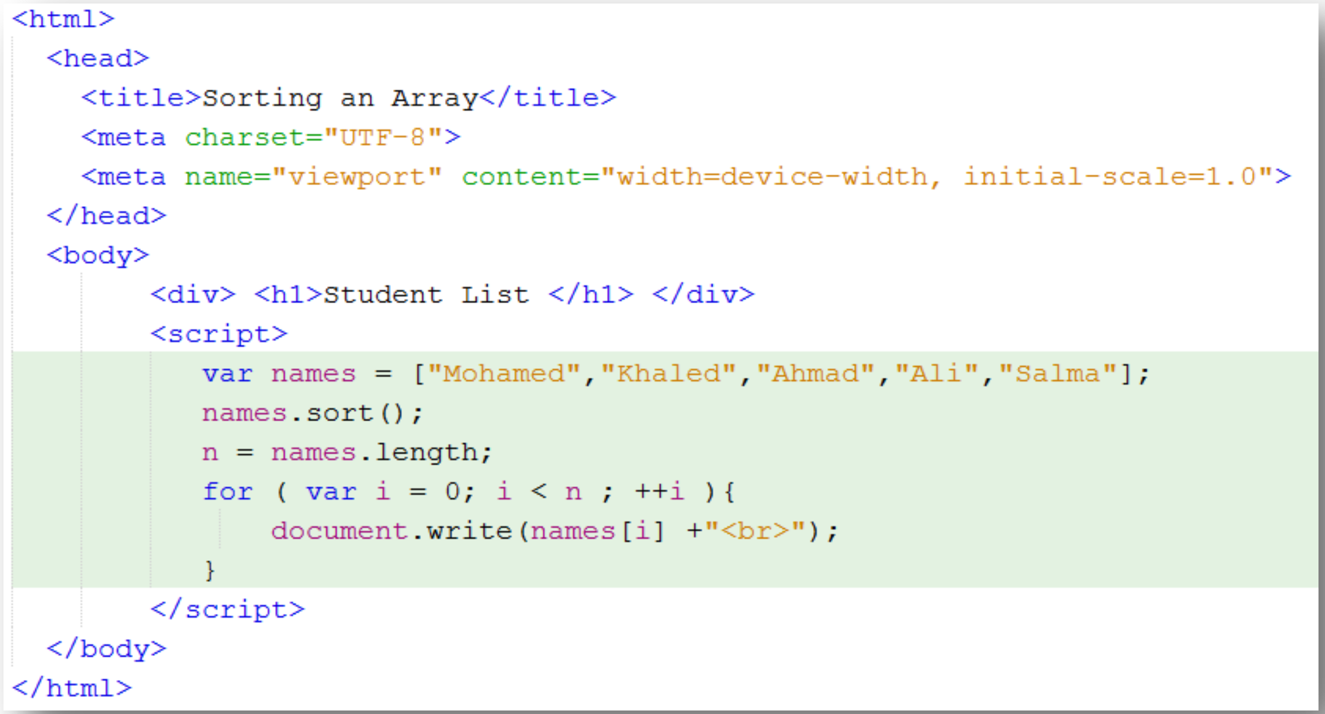
Array Methods

Sort()

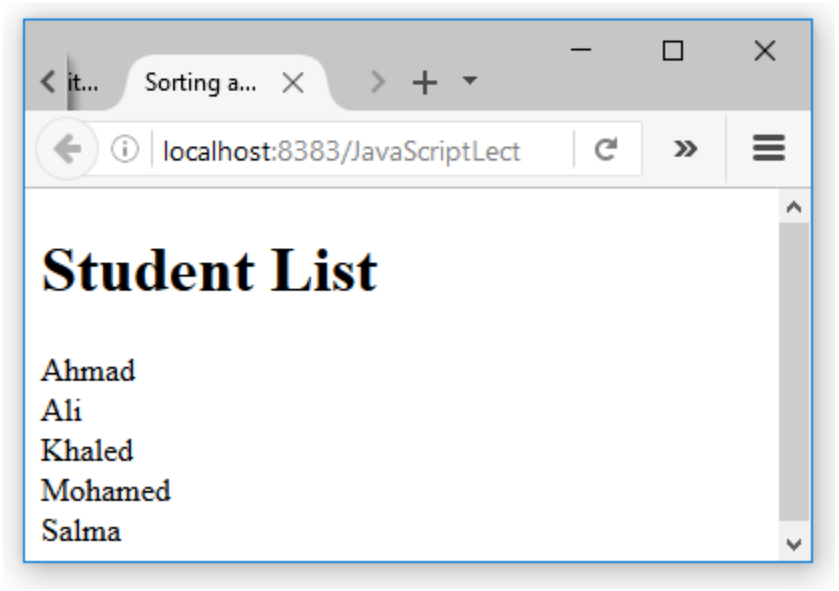
- JavaScript includes a sort method for arrays, which returns an alphabetically sorted version of the array.

Examle-2

The following example sorts an array and print it.



```
<html>
  <head>
    <title>Sorting an Array</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
        <div> <h1>Student List </h1> </div>
        <script>
           var names = ["Mohamed","Khaled","Ahmad","Ali","Salma"];
           names.sort();
           n = names.length;
           for ( var i = 0; i < n ; ++i ){
               document.write(names[i] +"<br>");
           }
        </script>
  </body>
</html>
```



```
Student List

Ahmad
Ali
Khaled
Mohamed
Salma
```

reverse()

- The Array.reverse() method reverses the order of the elements of an array and returns the reversed array.

```
var a = [1,2,3];
a.reverse();      // a is now [3,2,1]
```

concat()

- The Array.concat() method creates and returns a new array that contains the elements of the original array on which concat() was invoked, followed by each of the arguments to concat().

```
var a = [1,2,3];
a.concat(4, 5)       // Returns [1,2,3,4,5]
```

slice()

- The slice() method returns a slice, or subarray, of the specified array. Its two arguments specify the start and end of the slice to be returned.

```
var a = [1,2,3,4,5];
a.slice(0,3);    // Returns [1,2,3]
```

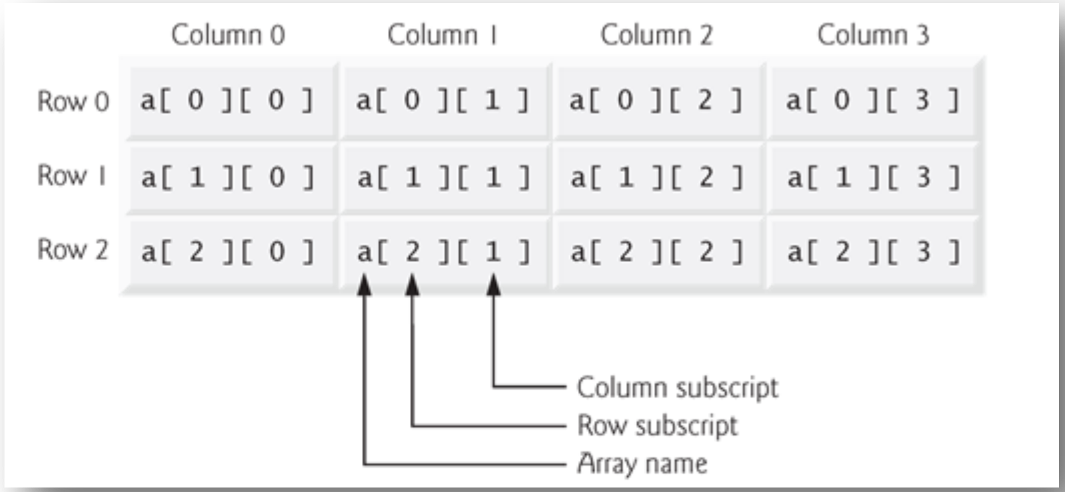
join()

- The Array.join() method converts all the elements of an array to strings and concatenates them, returning the resulting string. You can specify an optional string that separates the elements in the resulting string.

```
var a = [1, 2, 3];     // Create a new array with these three elements
a.join();              // => "1,2,3"
a.join(" ");           // => "1 2 3“
a.join(“-");           // => "1-2-3“
```

Two-dimensional arrays.

- Two dimensional arrays are often used to represent tables of values consisting of information arranged in rows and columns.



	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

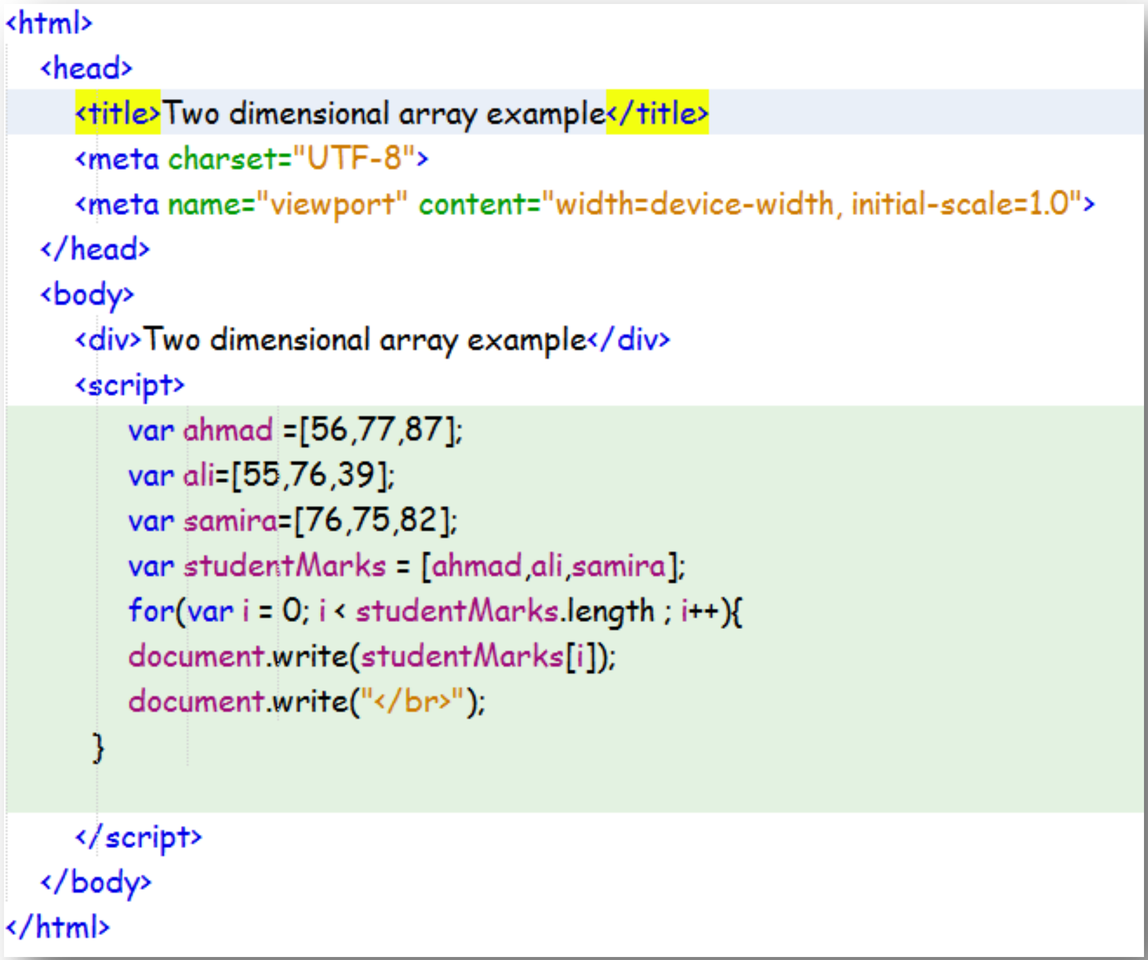
- Arrays can be initialized in declarations like a one-dimensional array.

```
var b = [ [ 1, 2 ], [ 3, 4 ] ];
```

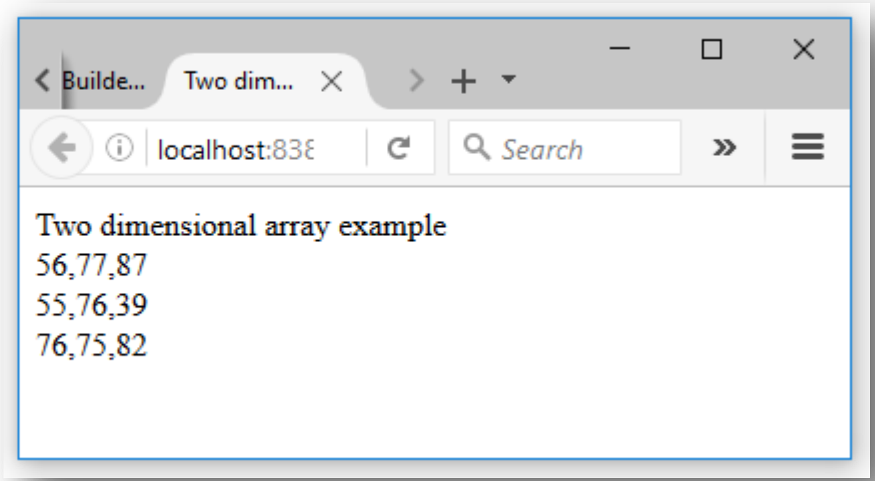
- Declaring Array of Array

```
// Create a two mensional array
var table = new Array(10);            // 10 rows of the table
for(var i = 0; i < table.length; i++)
table[i] = new Array(10);         // Each row has 10 columns
```

Examle-3



```
<html>
  <head>
    <title>Two dimensional array example</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <div>Two dimensional array example</div>
    <script>
       var ahmad =[56,77,87];
       var ali=[55,76,39];
       var samira=[76,75,82];
       var studentMarks = [ahmad,ali,samira];
       for(var i = 0; i < studentMarks.length ; i++){
       document.write(studentMarks[i]);
       document.write("</br>");
     }

    </script>
  </body>
</html>
```



```
Two dimensional array example
56,77,87
55,76,39
76,75,82
```

Functions

Functions

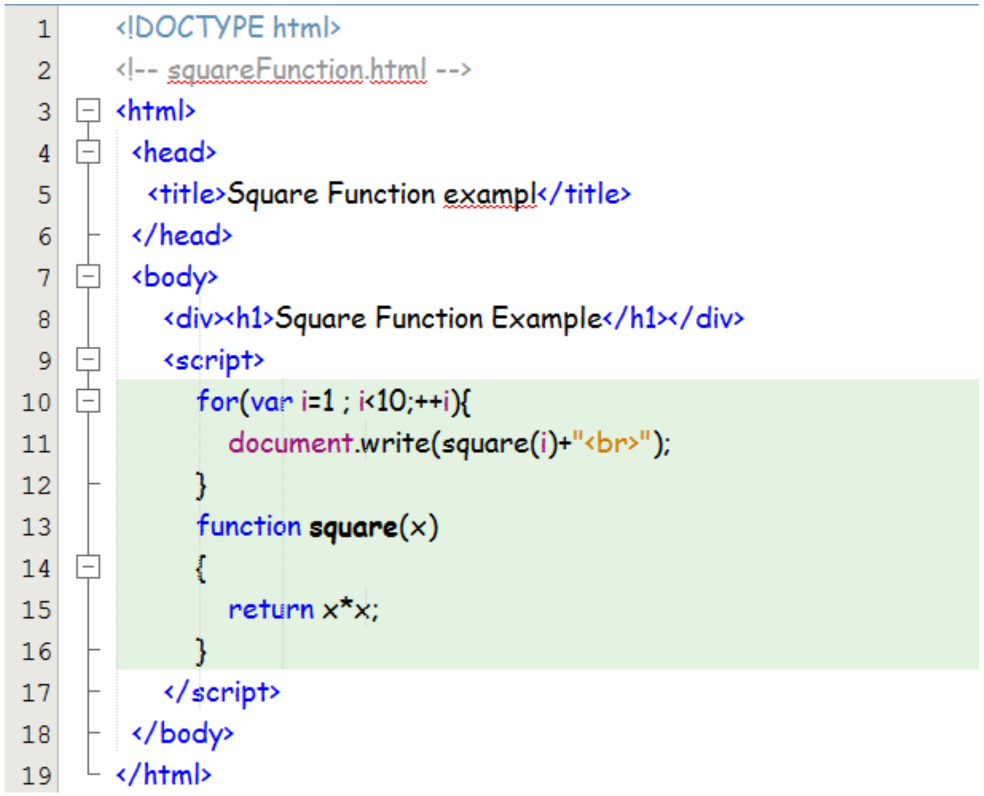
- A function is a block of JavaScript code that is defined once but may be executed, or invoked, any number of times.

- The general syntax for a function is shown here:

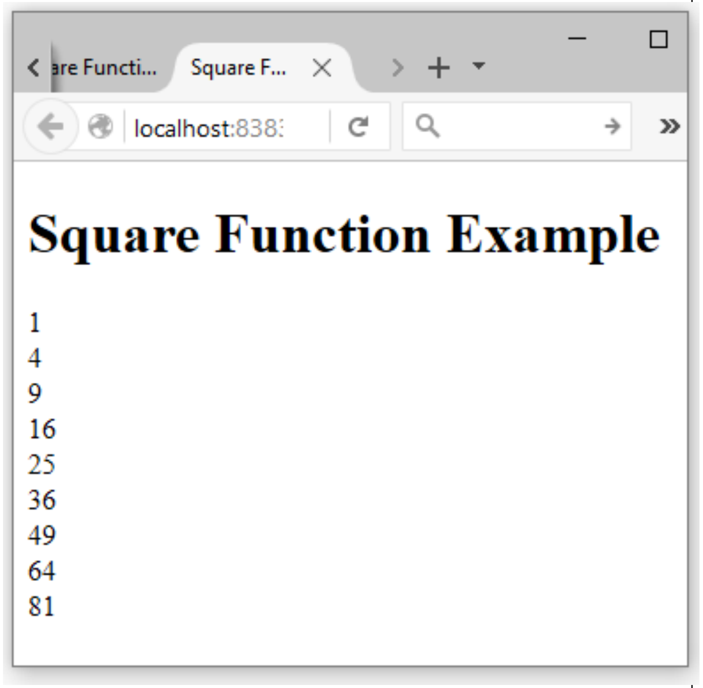
```
function function-name( parameter-list )
{
declarations and statements
}
```

Example-3

- This example uses a square function to square the numbers 1-10



```
<!DOCTYPE html>
<!-- squareFunction.html -->
<html>
 <head>
  <title>Square Function exampl</title>
 </head>
 <body>
   <div><h1>Square Function Example</h1></div>
   <script>
     for(var i=1 ; i<10;++i){
       document.write(square(i)+"<br>");
     }
     function square(x)
     {
       return x*x;
     }
   </script>
 </body>
</html>
```



Square Function Example

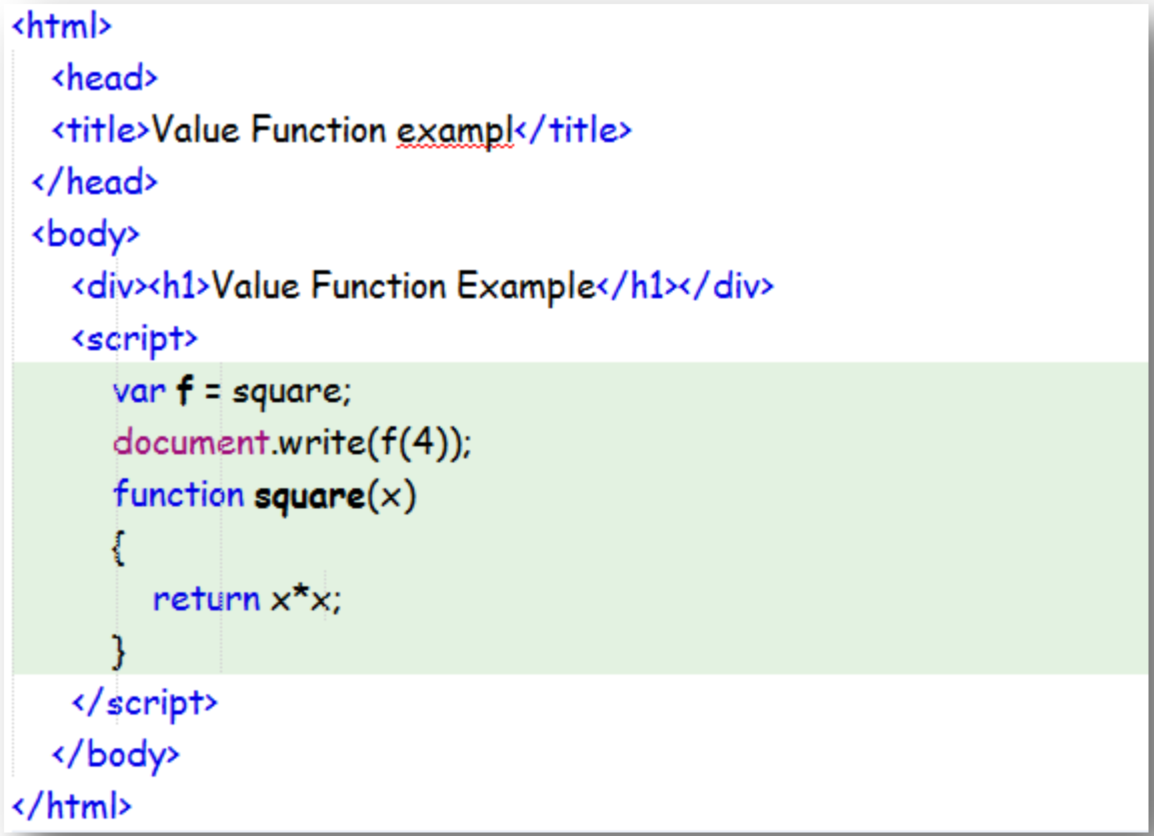
```
1
4
9
16
25
36
49
64
81
```

Functions As Values

- In JavaScript, functions are not only syntax but also values, which means they can be assigned to variables.

```
function square(x) { return x*x; }
var s = square;     // Now s refers to the same function that square does
square(4);        // => 16
s(4);             // => 16
```

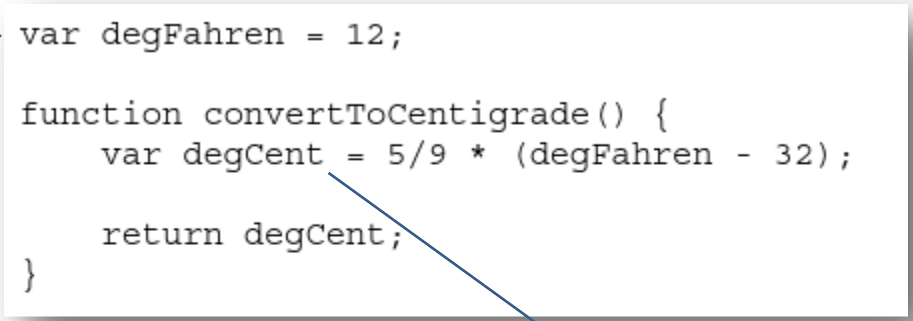
Example-4



```
<html>
  <head>
  <title>Value Function exampl</title>
 </head>
 <body>
    <div><h1>Value Function Example</h1></div>
    <script>
        var f = square;
        document.write(f(4));
        function square(x)
        {
          return x*x;
        }
    </script>
  </body>
</html>
```

Variables Scope

- **Global Variable**
  - Variables declared outside of a function are global variables and are visible everywhere in a JavaScript program.
- **Local Variable**
  - Variables declared inside a function have function scope and are visible only to code that appears inside that function.



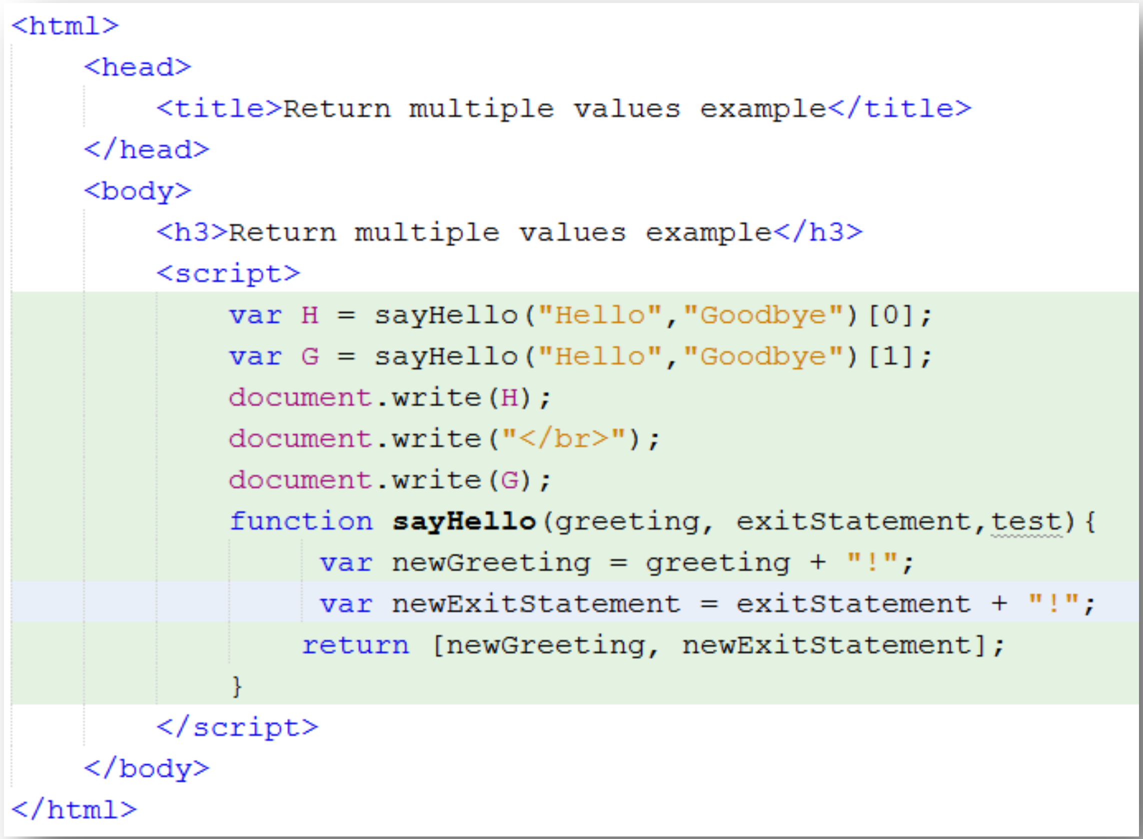
```
var degFahren = 12;

function convertToCentigrade() {
    var degCent = 5/9 * (degFahren - 32);

    return degCent;
}
```

Global variable

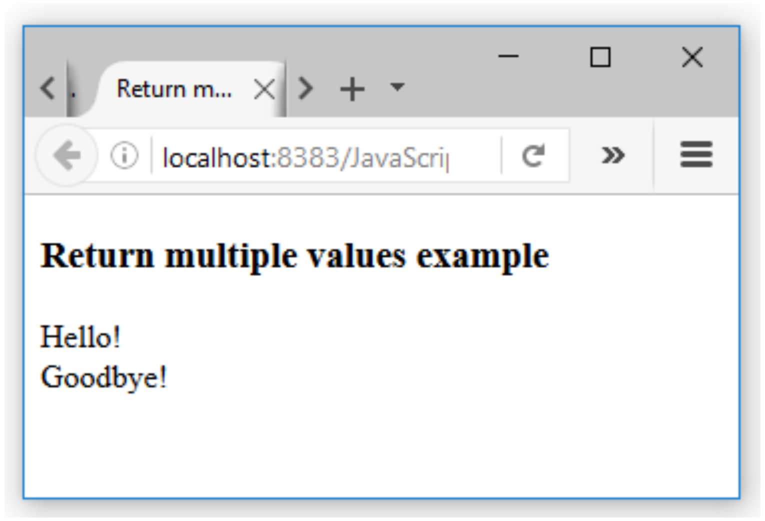
Local variable

Returning Multiple Data Values with a Function

- In JavaScript a function can return multiple values;



```
<html>
    <head>
        <title>Return multiple values example</title>
    </head>
    <body>
        <h3>Return multiple values example</h3>
        <script>
            var H = sayHello("Hello","Goodbye")[0];
            var G = sayHello("Hello","Goodbye")[1];
            document.write(H);
            document.write("</br>");
            document.write(G);
            function sayHello(greeting, exitStatement,test){
                 var newGreeting = greeting + "!";
                 var newExitStatement = exitStatement + "!";
                return [newGreeting, newExitStatement];
            }
        </script>
    </body>
</html>
```



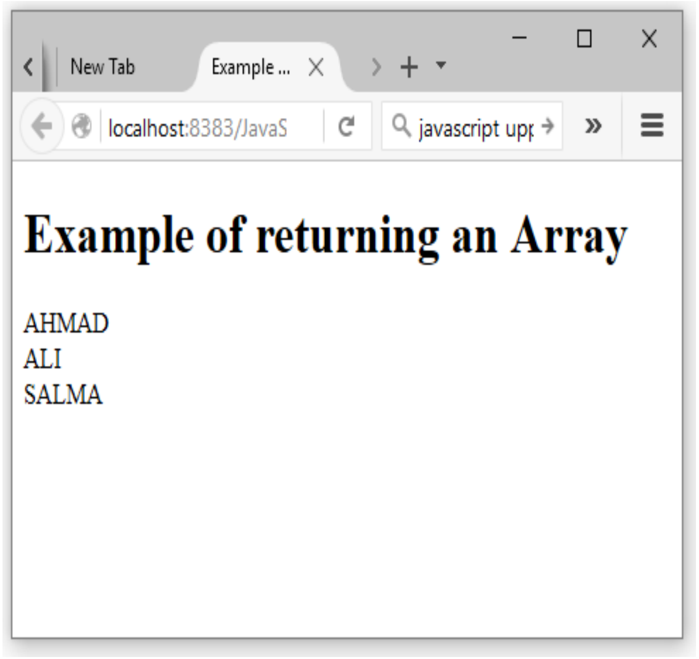
Return multiple values example

```
Hello!
Goodbye!
```

Returning n Array

- The following example shows how to return an Array.

```
<!DOCTYPE html>
<!-- returnArray.html -->
<html>
  <head>
    <title>Example of returning an Array</title>
  </head>
  <body>
    <div><h1>Example of returning an Array</h1></div>
    <script>
      var names = ["Ahmad", "Ali", "Salma"];
      var uNames = UpperCaseNames(names);
      for (j = 0; j < uNames.length; ++j)
        document.write(uNames[j] + "<br>");
          function UpperCaseNames(names)
          {
            var s = new Array();
            for (j = 0; j < names.length; ++j)
           s[j]=names[j].toUpperCase();
            return s;
          }
    </script>
  </body>
</html>
```



Example of returning an Array

```
AHMAD
ALI
SALMA
```

JavaScript Objects

- Objects are variables can contain many values.

```
Var  car = {type:” Kia”, model:”Picanto”,color:”white”}
```

```
var  student = {
      firstName:"Ali",
      lastName:"Salim",
      age:20,
      eyeColor:"brown" };
```

Accessing Object Properties

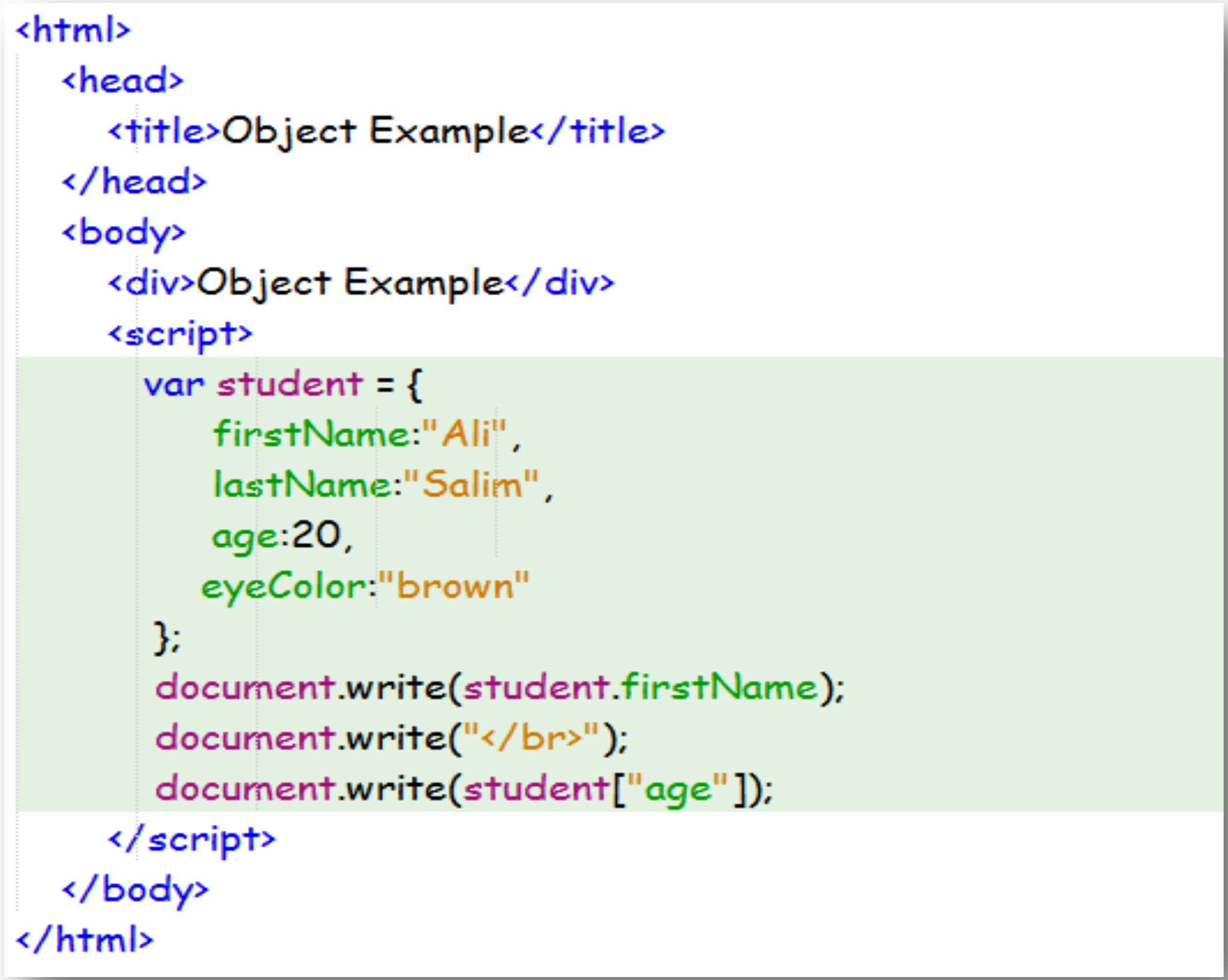
- object properties can be accessed in two ways:
  - objectName.propertyName

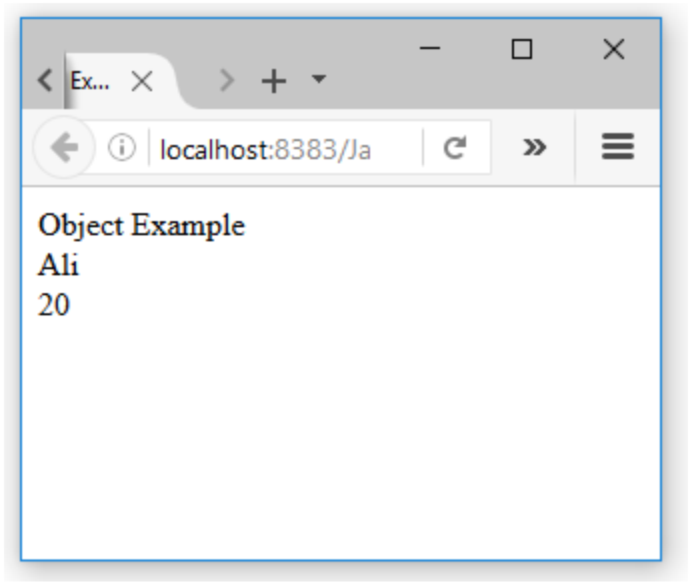
    car.type

  - *objectName["propertyName"]*

    car[“type”]

Example


```
<html>
   <head>
      <title>Object Example</title>
   </head>
   <body>
      <div>Object Example</div>
      <script>
         var student = {
               firstName:"Ali",
               lastName:"Salim",
               age:20,
              eyeColor:"brown"
         };
          document.write(student.firstName);
          document.write("</br>");
          document.write(student["age"]);
      </script>
   </body>
</html>
```



```
Object Example
Ali
20
```

JavaScript
(Event Handling)

Events

- Events are the actions that occur as a result of browser activities or user interactions with the web pages.
  - An HTML web page has finished loading
  - Such as the user performs an action
    - mouse click
    - enters data

Event Handlers

- When an event occurs, a code is executed in response to a specific event is called *"event handler".*
- Event handler names are quite similar to the name of events they handle.

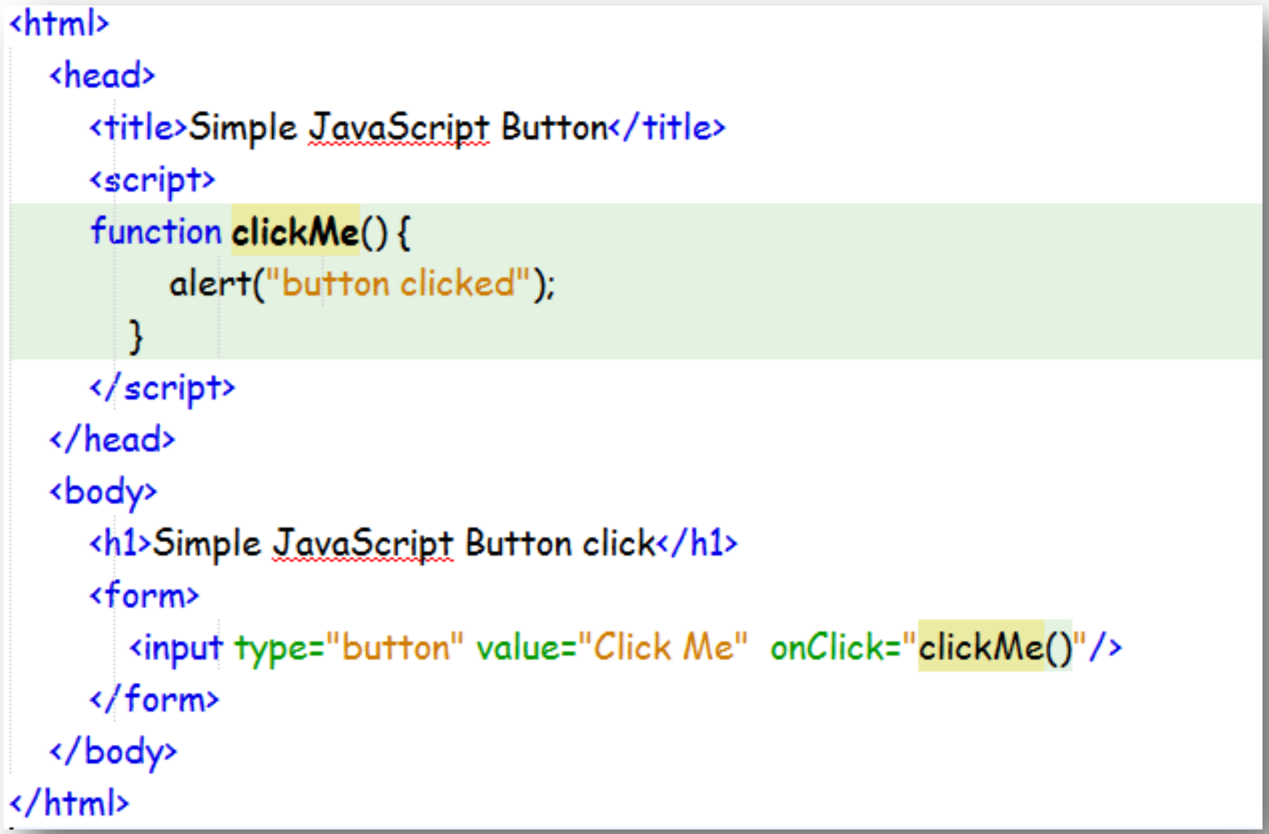
  E.g the event handler for the "click" event is "onClick".

  - `<HTMLtag eventhandler="JavaScript Code">`

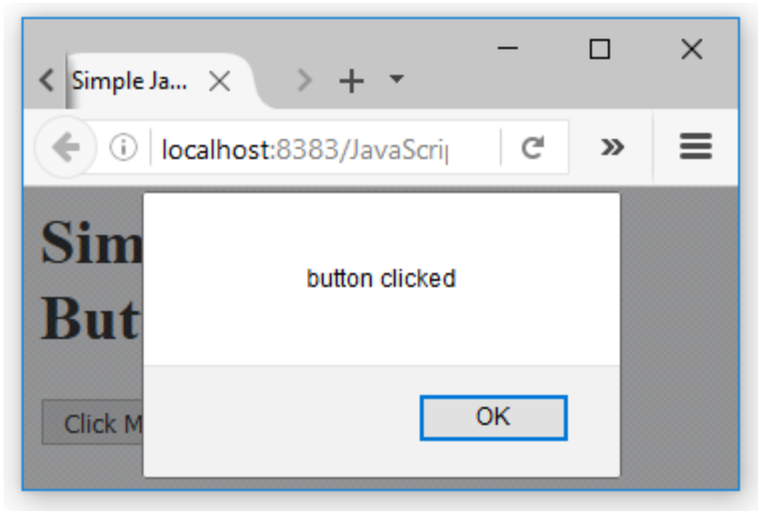
```
<body onload="alert('Welcome User');">
```

Event Handlers	Triggered when
onChange	The value of the text field, textarea, or a drop down list is modified
onClick	A link, an image or a form element is clicked once
onDblClick	The element is double-clicked
onMouseDown	The user presses the mouse button
onLoad	A document or an image is loaded
onSubmit	A user submits a form
onReset	The form is reset
onUnLoad	The user closes a document or a frame
onResize	A form is resized by the user

Onclick Example

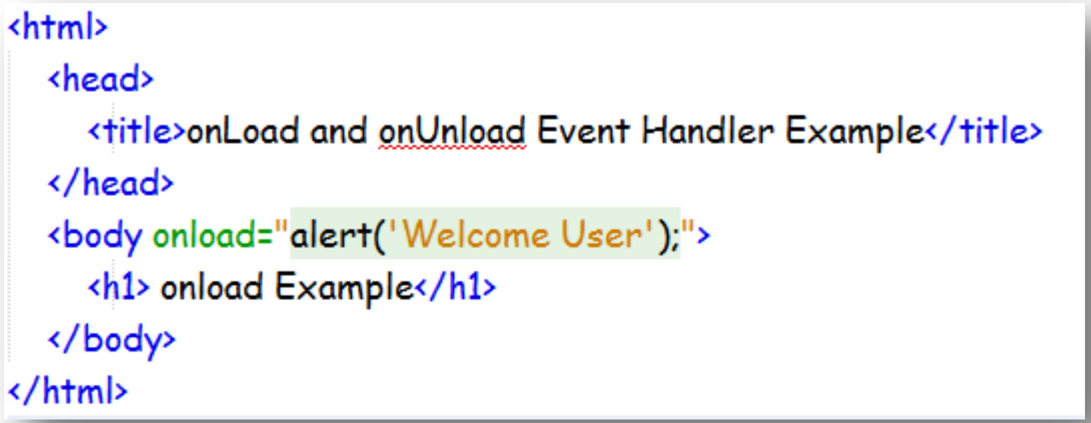


```
<html>
  <head>
    <title>Simple JavaScript Button</title>
    <script>
    function clickMe() {
        alert("button clicked");
      }
    </script>
  </head>
  <body>
    <h1>Simple JavaScript Button click</h1>
    <form>
      <input type="button" value="Click Me"  onClick="clickMe()"/>
    </form>
  </body>
</html>
```

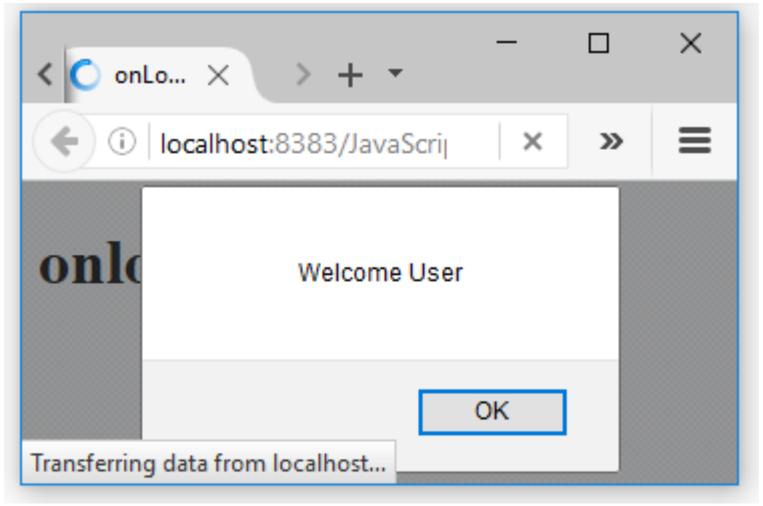


button clicked

Onload Example



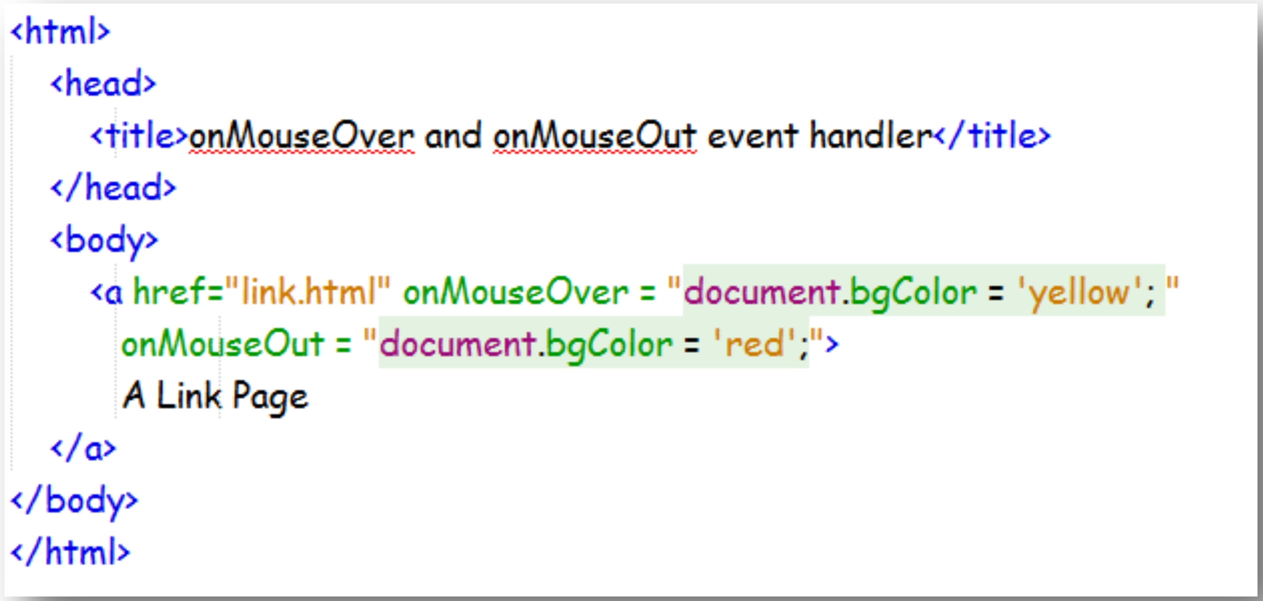
```
<html>
  <head>
    <title>onLoad and onUnload Event Handler Example</title>
  </head>
  <body onload="alert('Welcome User');">
    <h1> onload Example</h1>
  </body>
</html>
```



Welcome User

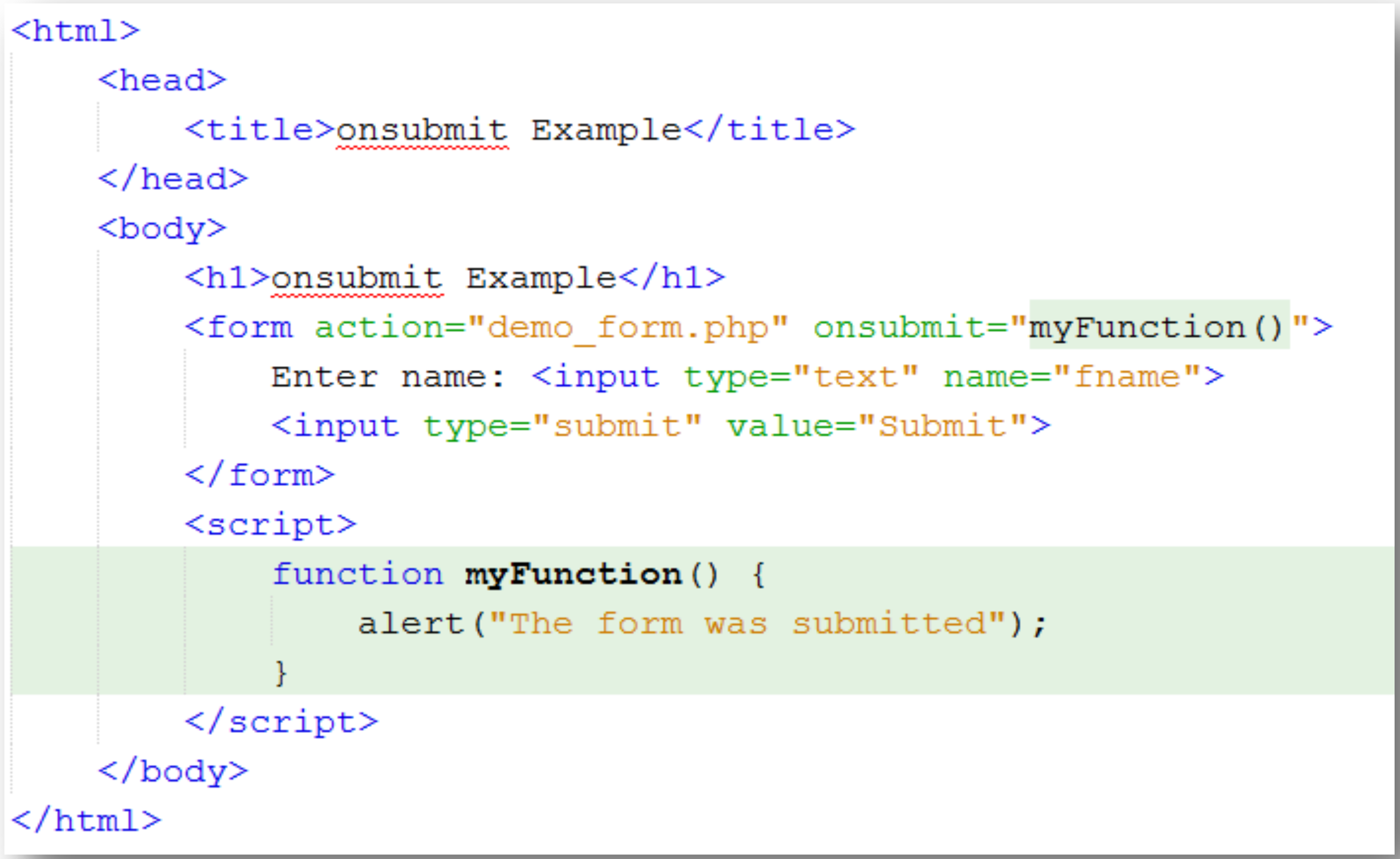
Transferring data from localhost...

onMouseOver and onMouseOut

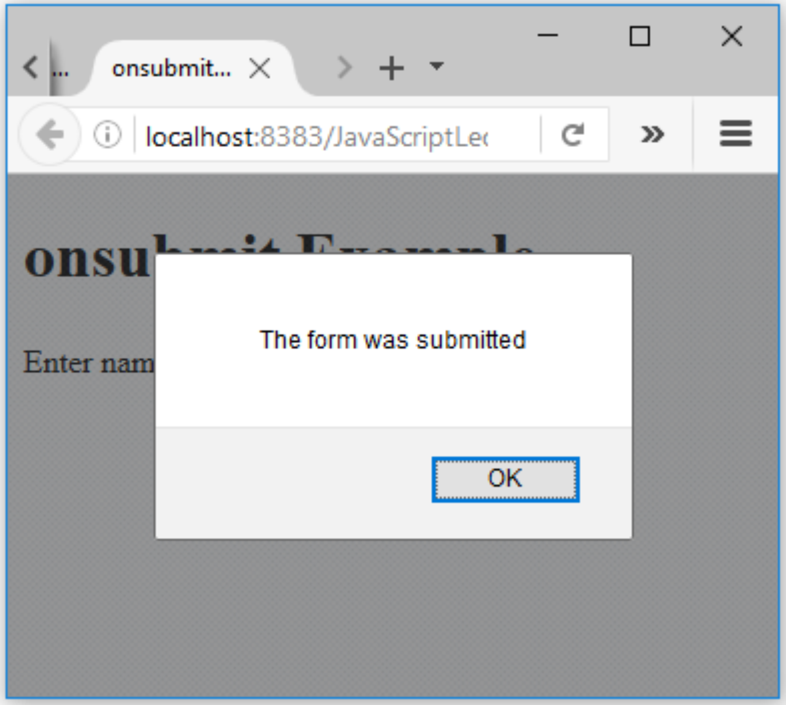


```
<html>
   <head>
      <title>onMouseOver and onMouseOut event handler</title>
   </head>
   <body>
      <a href="link.html" onMouseOver = "document.bgColor = 'yellow'; "
        onMouseOut = "document.bgColor = 'red';">
        A Link Page
   </a>
</body>
</html>
```

Onsubmit Example



```
<html>
    <head>
        <title>onsubmit Example</title>
    </head>
    <body>
        <h1>onsubmit Example</h1>
        <form action="demo_form.php" onsubmit="myFunction()">
            Enter name: <input type="text" name="fname">
            <input type="submit" value="Submit">
        </form>
        <script>
            function myFunction() {
                alert("The form was submitted");
            }
        </script>
    </body>
</html>
```



The form was submitted

The following tables show the events and their event handlers:

Mouse Events

Event	Fires When
`onclick`	mouse button is clicked
`ondblclick`	mouse button is double-clicked
`onmousedown`	mouse button is pressed down
`onmouseup`	mouse button is released
`onmousemove`	mouse moves
`onmouseover`	mouse enters an element
`onmouseout`	mouse leaves an element

Keyboard Events

Event	Fires When The User
`onkeypress`	presses then releases a key
`onkeydown`	pushes down a key
`onkeyup`	releases a key

Selection and Focus Events

Event	Fires When
`onselect`	text selection begins (inside either <input type="text"> or <textarea>)
`onchange`	when a text input is changed and the element loses focus, or new choice is made in a select element
`onfocus`	form element gains focus
`onblur`	form element loses focus

Other Events

Event	Fires When
`onresize`	user resizes a window or a frame
`onsubmit`	form is submitted, i.e., the user clicks the reset button
`onreset`	form is reset, i.e., the user clicks the reset button

The following Link conations the HTML events.

- List of HTML Events