



*NextGen*



*Web*



Session: 12

# *Introduction to JavaScript*



# Objectives

- Explain scripting
- Explain the JavaScript language
- Explain the client-side and server-side JavaScript
- List the variables and data types in JavaScript
- Describe the JavaScript methods to display information
- Explain escape sequences and built in functions in JavaScript
- Explain events and event handling
- Explain jQuery
- Describe how to use the jQuery Mobile

Scripting refers to a series of commands that are interpreted and executed sequentially and immediately on occurrence of an event.

This event is an action generated by a user while interacting with a Web page.

Examples of events include button clicks, selecting a product from a menu, and so on.

A scripting language refers to a set of instructions that provides some functionality when the user interacts with a Web page.

Scripting languages are often embedded in the HTML pages to change the behavior of the Web pages according to the user's requirements.

# Scripting 2-3

- Following figure displays the need for scripting.



# Scripting 3-3

- There are two types of scripting languages. They are as follows:

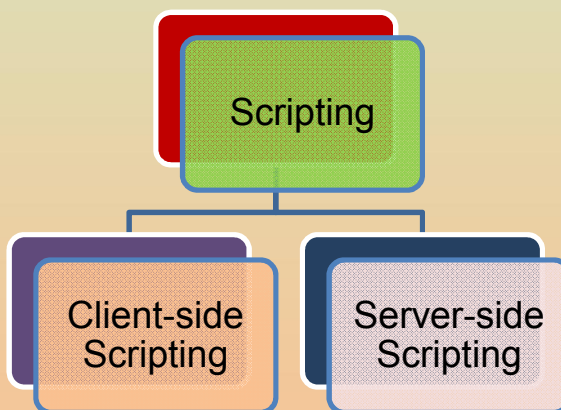
## Client-side Scripting:

- Refers to a script being executed on the client's machine by the browser.

## Server-side Scripting:

- Refers to a script being executed on a Web server to generate dynamic HTML pages.

- Following figure displays the types of scripting.





# JavaScript 1-2

JavaScript is a scripting language that allows building dynamic Web pages by ensuring maximum user interactivity.

JavaScript language is an object-based language, which means that it provides objects for specifying functionalities.

In real life, an object is a visible entity such as a car or a table having some characteristics and capable of performing certain actions.

Similarly, in a scripting language, an object has a unique identity, state, and behavior.

The identity of the object distinguishes it from the other objects of the same type.

The state of the object refers to its characteristics, whereas the behavior of the object consists of its possible actions.

The object stores its identity and state in fields (also called variables) and exposes its behavior through functions (actions).



# JavaScript 2-2

- Following figure displays some real world objects.



Identity: AXA 43 S

State: Color-Red, Wheels-Four

Behavior: Running



Identity: T002

State: Color-Brown

Behavior: Stable

## Objects

# HTML5 Versions of JavaScript

- The first version of JavaScript was developed by Brendan Eich at Netscape in 1995 and was named JavaScript 1.0.
- Following table lists the various versions of JavaScript language.

Version	Description
1.1	Is supported from 3.0 version of the Netscape Navigator and Internet Explorer.
1.2	Is supported by the Internet Explorer from version 4.0.
1.3	Is supported by the Internet Explorer from version 5.0, Netscape Navigator from version 4.0, and Opera from version 5.0.
1.4	Is supported by servers of Netscape and Opera 6.
1.5	Is supported by the Internet Explorer from version 6.0, Netscape Navigator from version 6.0, and Mozilla Firefox from version 1.0.
1.6	Is supported in the latest versions of the Internet Explorer and Netscape Navigator browsers. It is also supported by Mozilla Firefox from version 1.5.
1.7	Is supported in the latest versions of the Internet Explorer and Netscape Navigator browsers. It is also supported by Mozilla Firefox from version 2.0.



# HTML Client-side JavaScript 1-2

A Client-side JavaScript (CSJS) is executed by the browser on the user's workstation.

A client-side script might contain instructions for the browser to handle user interactivity.

These instructions might be to change the look or content of the Web page based on the user inputs.

Examples include displaying a welcome page with the user name, displaying date and time, validating that the required user details are filled, and so on.

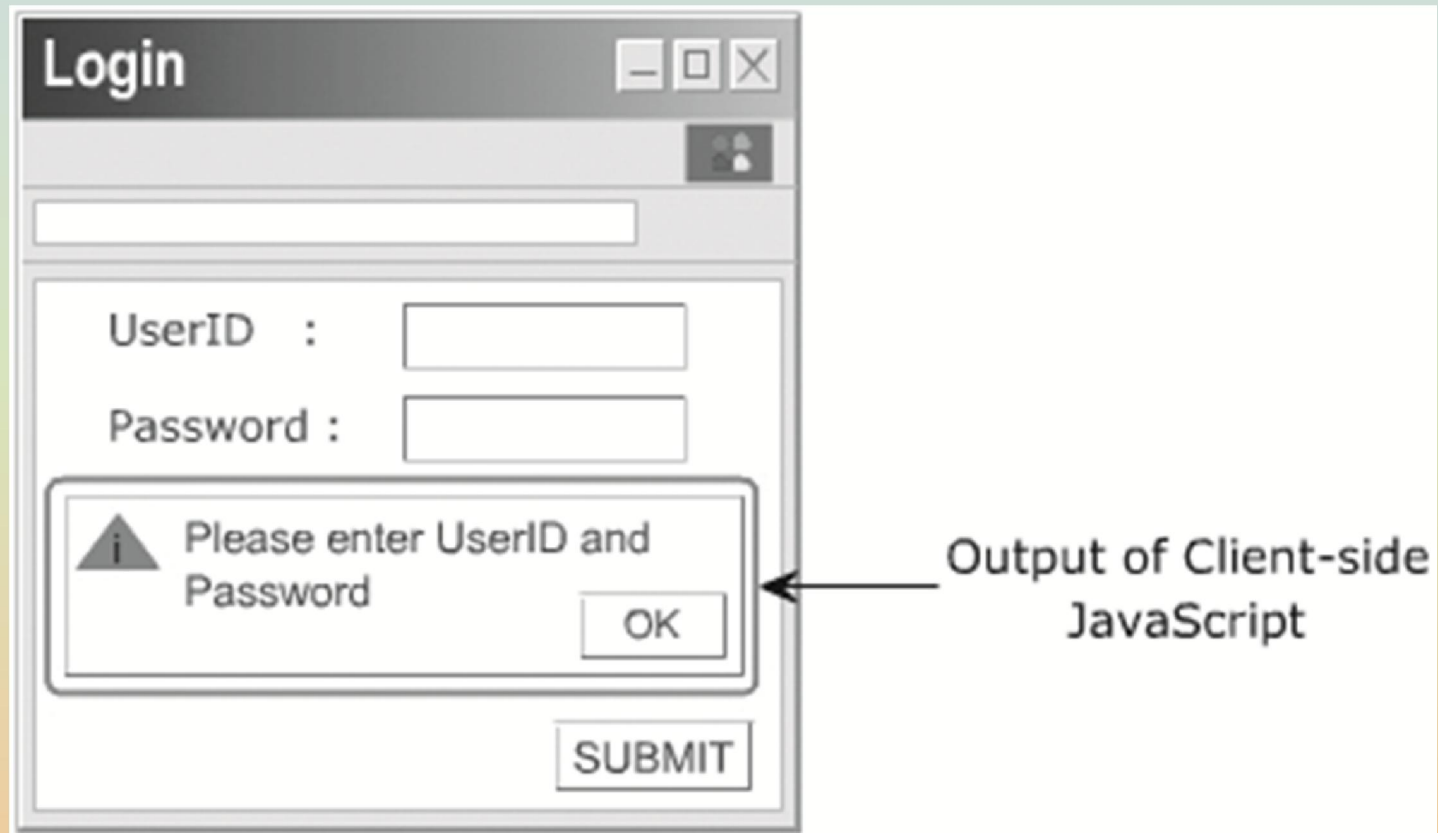
A JavaScript is either embedded in an HTML page or is separately defined in a file, which is saved with .js extension.

In client-side scripting, when an HTML is requested, the Web server sends all the required files to the user's computer.

The Web browser executes the script and displays the HTML page to the user along with any tangible output of the script.

# HTML Client-side JavaScript 2-2

- Following figure displays the output of a client-side JavaScript.



# HTML Server-side JavaScript 1-2

A Server-side JavaScript (SSJS) is executed by the Web server when an HTML page is requested by a user and the output is displayed by the browser.

A server-side JavaScript can interact with the database, fetch the required information specific to the user, and display it to the user.

Server-side scripting fulfills the goal of providing dynamic content in Web pages.

Unlike client-side JavaScript, HTML pages using server-side JavaScript are compiled into bytecode files on the server.

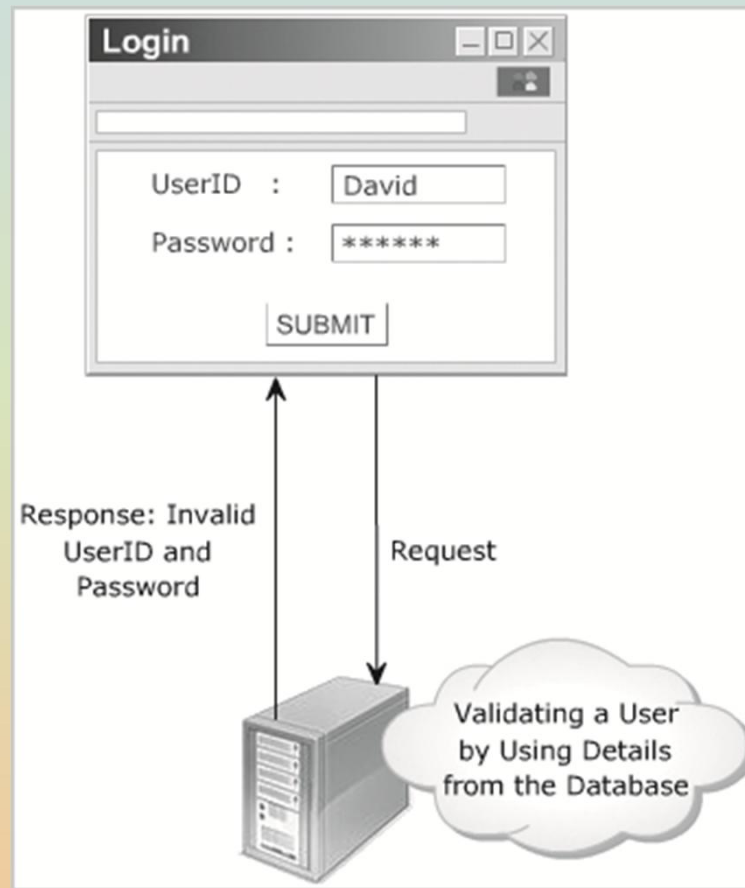
A JavaScript is either embedded in an HTML page or is separately defined in a file, which is saved with .js extension.

Compilation is a process of converting the code into machine-independent code.

This machine-independent code is known as the bytecode, which is an executable file that the Web server runs to generate the desired output.

# HTML5 Server-side JavaScript 2-2

- Following figure displays the output of a client-side JavaScript.





# <Script> Tag 1-2

The <script> tag defines a script for an HTML page to make them interactive.

The browser that supports scripts interprets and executes the script specified under the <script> tag when the page loads in the browser.

You can directly insert a JavaScript code under the <script> tag.

You can define multiple <script> tags either in the <head> or in the <body> elements of an HTML page.

In HTML5, the type attribute specifying the scripting language is no longer required as it is optional.



# <Script> Tag 2-2

- The Code Snippet demonstrates the use of the tag.

```
<!doctype html>
<html>
  <head>
    <script>
      document.write("Welcome to the Digital World");
    </script>
  </head>
  <body>
    .....
  </body>
</html>
```

There are two main purposes of the `<script>` tag, which are as follows:

Identifies a given segment of script in the HTML page.

Loads an external script file.

# HTML5 Variables in JavaScript

A variable refers to a symbolic name that holds a value, which keeps changing.

For example, age of a student and salary of an employee can be treated as variables.

In JavaScript, a variable is a unique location in computer's memory that stores a value and has a unique name.

The name of the variable is used to access and read the value stored in it.

A variable can store different types of data such as a character, a number, or a string.

# Declaring Variables 1-4

- Declaring a variable refers to creating a variable by specifying the variable name.
- For example, one can create a variable named to store the name of a student.

In JavaScript,

the var keyword is used to create a variable by allocating memory to it.

a keyword is a reserved word that holds a special meaning.

the variable can be initialized at the time of creating the variable or later.

initialization refers to the task of assigning a value to a variable.

once the variable is initialized, you can change the value of a variable as required.

variables allow keeping track of data during the execution of the script.

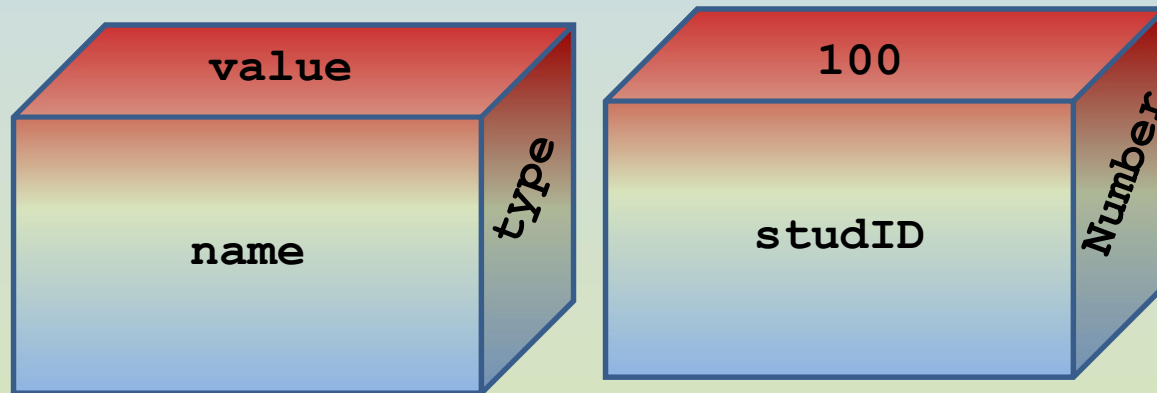
while referring to a variable, you are referring to the value of that variable.

one can declare and initialize multiple variables in a single statement.



# HTML5 Declaring Variables 2-4

- Following figure displays how to declare variables.



- Following syntax demonstrates how to declare variables in JavaScript.

## Syntax:

```
var <variableName>;
```

where,

- `var`: Is the keyword in JavaScript.
- `variableName`: Is a valid variable name.

# HTML5 Declaring Variables 3-4

- Following syntax demonstrates how to initialize variables in JavaScript.

## Syntax:

```
<variableName> = <value>;
```

where,

- `=`: Is the assignment operator used to assign values.
  - `value`: Is the data that is to be stored in the variable.
- The syntax demonstrates how to declare and initialize multiple variables in a single statement, which are separated by commas.

## Syntax:

```
var <variableName1> = <value1>, <variableName2> = <value2>;
```

# HTML5 Declaring Variables 4-4

- The Code Snippet declares two variables namely, `studID` and `studName` and assign values to them.

```
var studID;  
var studName;  
studID = 50;  
  
studName = "David Fernando";
```

- The snippet assigns values to `studID` and `studName` variables by using the assignment operator (=).
- The value named `David Fernando` is specified within double quotes.
- The Code Snippet demonstrates how to declare and initialize multiple variables in a single statement in JavaScript.

```
var studName = David, studAge = 15;
```

# Variable Naming Rules

- JavaScript is a case-sensitive language.
- The variables X and x are treated as two different variables.
- JavaScript consists of certain rules for naming a variable as follows:

In JavaScript, a variable name

can consist of digits, underscore, and alphabets.

must begin with a letter or the underscore character.

cannot begin with a number and cannot contain any punctuation marks.

cannot contain any kind of special characters such as +, \*, %, and so on.

cannot contain spaces.

cannot be a JavaScript keyword.

# Data Types in JavaScript 1-3

- To identify the type of data that can be stored in a variable, JavaScript provides different data types.
- Data types in JavaScript are classified into two broad categories namely, primitive and composite data types.
- Primitive data types contain only a single value, whereas the composite data types contain a group of values.

## ➤ PRIMITIVE DATA TYPES

- A primitive data type contains a single literal value such as a number or a string.
- A literal is a static value that you can assign to variables.

# Data Types in JavaScript 2-3

- Following table lists the primitive data types.

Primitive Data Type	Description
<code>boolean</code>	Contains only two values namely, true or false
<code>null</code>	Contains only one value namely, null. A variable of this value specifies that the variable has no value. This null value is a keyword and it is not the same as the value, zero
<code>number</code>	Contains positive and negative numbers and numbers with decimal point. Some of the valid examples include 6, 7.5, -8, 7.5e-3, and so on
<code>string</code>	Contains alphanumeric characters in single or double quotation marks. The single quotes is used to represent a string, which itself consists of quotation marks. A set of quotes without any characters within it is known as the null string

## ➤ COMPOSITE DATA TYPES

- A composite data type stores a collection of multiple related values, unlike primitive data types.
- In JavaScript, all composite data types are treated as objects.
- A composite data type can be either predefined or user-defined in JavaScript.
- Following table lists the composite data types.

Composite Data Type	Description
Objects	Refers to a collection of properties and functions. Properties specify the characteristics and functions determine the behavior of a JavaScript object
Functions	Refers to a collection of statements, which are instructions to achieve a specific task
Arrays	Refers to a collection of values stored in adjacent memory locations



# Methods 1-3

- JavaScript allows you to display information using the methods of the document object.
- The document object is a predefined object in JavaScript, which represents the HTML page and allows managing the page dynamically.
- Each object in JavaScript consists of methods, that fulfill a specific task.
- There are two methods of the document object, that display any type of data in the browser. These methods are as follows:
  - `write()`: Displays any type of data.
  - `writeln()`: Displays any type of data and appends a new line character.
- The syntax demonstrates the use of `document.write()` method, which allows you to display information in the displayed HTML page.

## Syntax:

```
document.write("<data>" + variables);
```

where,

- `data`: Specifies strings enclosed in double quotes.
- `variables`: Specify variable names whose value should be displayed on the HTML page.



# Methods 2-3

- The syntax demonstrates the use of `document.writeln()` method, which appends a new line character.

**Syntax:**

```
document.writeln("<data>" + variables);
```

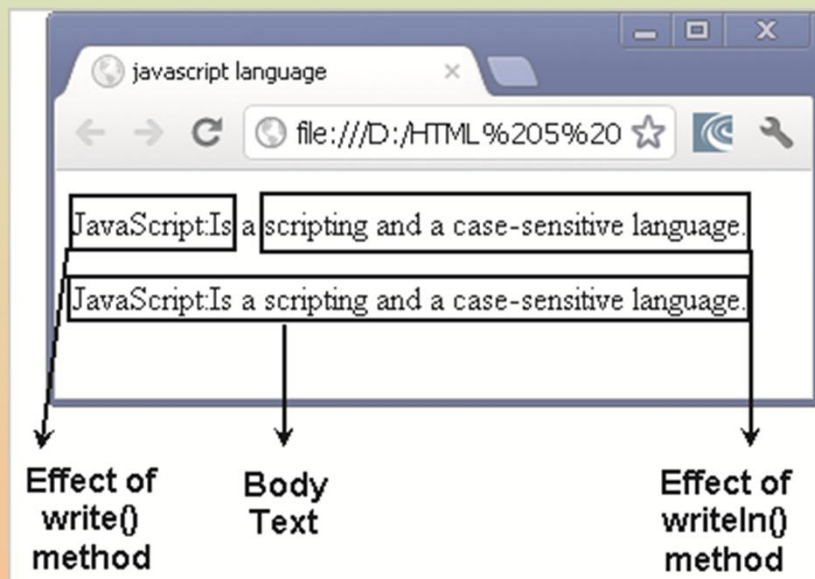
- The Code Snippet demonstrates the use of `write()` method.

```
<!DOCTYPE HTML>
<html>
  <head>
    <title> JavaScript language </title>
    <script>
      document.write("<p> JavaScript:");
      document.writeln("is a scripting");
      document.write("and a case-sensitive language.");
    </script>
  </head>
  <p>
    JavaScript: is a scripting and a case-sensitive language.
  </p>
</html>
```



# Methods 3-3

- The code uses the `writeln()` method to display the text after the colon.
- It appends a new line character after the text. Then, the text within the `write()` method is displayed on the same line after leaving a space.
- The same paragraph is displayed in the body of the HTML page.
- Note that the text in the `<p>` element appears on different lines.
- In HTML, by default the text will not be displayed in the new line in the browser even though the ENTER key is pressed while writing the code.
- Rather, it will be displayed on the same line with a space. The `writeln()` method also follows this same format.
- Following figure displays the use of `write()` and `writeln()` methods.





# Using Comments

- Comments provide information about a piece of code in the script.
- When the script is executed, the browser identifies comments as they are marked with special characters and does not display them.
- JavaScript supports two types of comments. These are as follows:

## ➤ SINGLE LINE COMMENTS

- Single-line comments begin with two forward slashes (//). You can insert single-line comments as follows:

```
// This statement declares a variable named num.  
var num;
```

## ➤ MULTI-LINE COMMENTS

- Multi-line comments begin with a forward slash followed by an asterisk (/\*) and end with an asterisk followed by a forward slash (\*).
- You can insert multiple lines of comments as follows:

```
/* This line of code  
   declares a variable */  
var num;
```

# Escape Sequence Characters 1-2

- An escape sequence character is a special character that is preceded by a backslash (\).
- Escape sequence characters are used to display special non-printing characters such as a tab space, a single space, or a backspace.
- In JavaScript, the escape sequence characters must be enclosed in double quotes.
- Following table lists the escape sequence characters.

Escape Sequence	Non-Printing Character
\b	Back space
\f	Form feed
\n	New line
\r	Carriage return
\t	Horizontal tab
\'	Single quote
\"	Double quote

# Escape Sequence Characters 2-2

Escape Sequence	Non-Printing Character
\\	Backslash
\\aaa	Matches a Latin-1 encoding character using octal representation, where aaa are three octal numbers. For example, \251 represents the copyright symbol
\\xaa	Matches a Latin-1 encoding character using hexadecimal representation, where aa are two hexadecimal numbers. For example, \x61 represents the character 'a'
\\uaaaa	Represent the Unicode encoding character, where aaaa are four hexadecimal numbers. For example, the character \u0020 represents a space

- The Code Snippet demonstrates the use of escape sequence characters in JavaScript.

```
<script>
    document.write("You need to have a \u0022credit card\u0022, if
                    you want to shop on the \'Internet\'.");
</script>
```

- The code uses a Unicode encoding character namely, \u0022, which represents double quotes that will contain the term `credit card`.
- The word `Internet` is placed in single quotes that are specified using the backslash character.

# HTML5 Built-in Functions 1-4

- A function is a piece of code that performs some operations on variables to fulfill a specific task.
- It takes one or more input values, processes them, and returns an output value.
- Following table lists the built-in JavaScript functions.

Function	Description	Example
<code>alert()</code>	Displays a dialog box with some information and OK button	<code>alert("Please fill all the fields of the form");</code> Displays a message box with the instruction
<code>confirm()</code>	Displays a dialog box with OK and Cancel buttons. It verifies an action, which a user wants to perform	<code>confirm("Are you sure you want to close the page?");</code> Displays a message box with the question
<code>parseInt()</code>	Converts a string value into a numeric value	<code>parseInt("25 years");</code>
<code>parseFloat()</code>	Converts a string into a number with decimal point	<code>parseFloat("10.33");</code> Returns 10.33

# HTML Built-in Functions 2-4

Function	Description	Example
<code>eval()</code>	Evaluates an expression and returns the evaluated result	<code>eval("2+2");</code> Returns 4
<code>isNaN()</code>	Checks whether a value is not a number	<code>isNaN("Hello");</code> Returns true
<code>prompt()</code>	Displays a dialog box that accepts an input value through a text box. It also accepts the default value for the text box	<code>prompt("Enter your name", "Name");</code> Displays the message in the dialog box and Name in the text box.

- An element organizes the content in a Web page hierarchically, which forms the basic HTML structure.

```
<!DOCTYPE HTML>
<html>
  <head>
    <title> JavaScript language </title>
    <script>
      var value = "";
```

## Built-in Functions 3-4

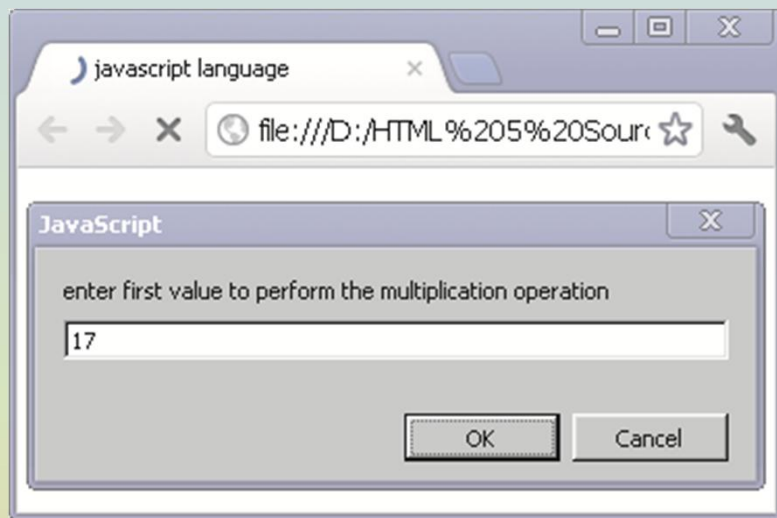
```
var numone = prompt("enter first value to perform the  
multiplication operation", value);  
  
var numtwo = prompt("enter second value to perform the  
multiplication operation", value);  
  
var result = eval(numone * numtwo);  
document.write("The result of multiplying: " + numone +  
"and " + numtwo + " is: " + result + "." );  
  
</script>  
</head>  
</html>
```

- The code snippet, takes the first value from the user and stores in the `numOne` variable.
- Then, it takes the second value from the user and stores in the `numTwo` variable.
- It multiplies the values and stores the output in the `result` variable and then, displays the output on the Web page.

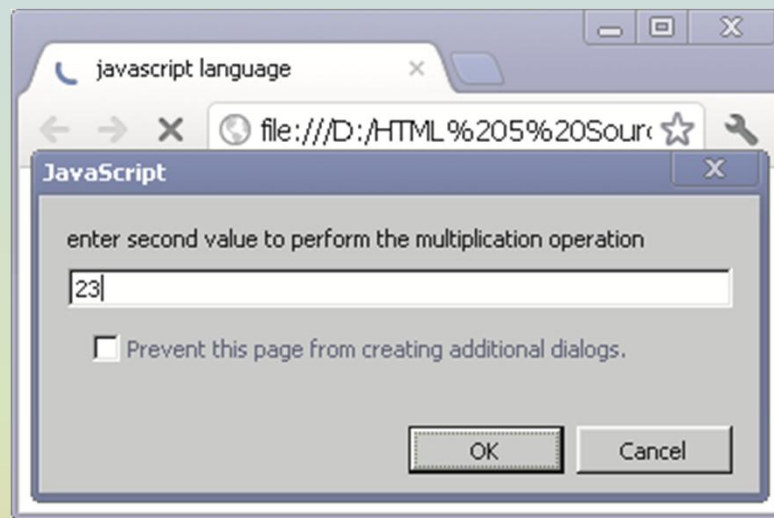


# HTML5 Built-in Functions 4-4

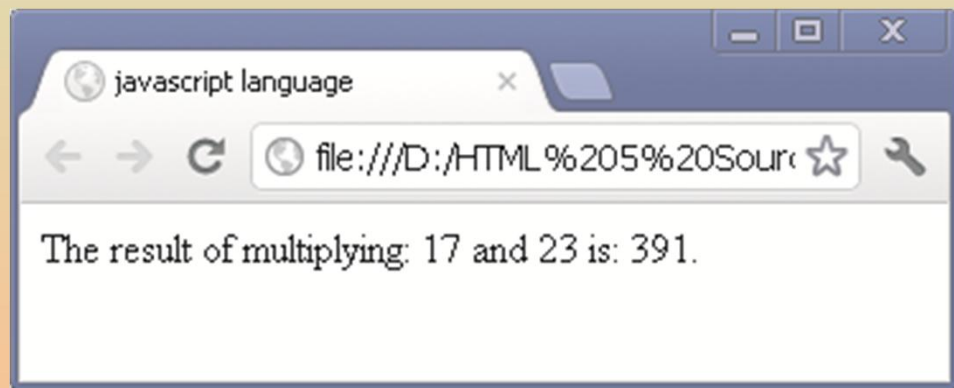
- Following figure displays the input first number.



- Following figure displays the input second number.



- Following figure displays the result.



- An event occurs when a user interacts with the Web page.
- Some of the commonly generated events are mouse clicks, key strokes, and so on.
- The process of handling these events is known as event handling.
- Following figure displays the event.



# Event Handling 1-2

Event handling is a process of specifying actions to be performed when an event occurs. This is done by using an event handler.

An event handler is a scripting code or a function that defines the actions to be performed when the event is triggered.

When an event occurs, an event handler function that is associated with the specific event is invoked.

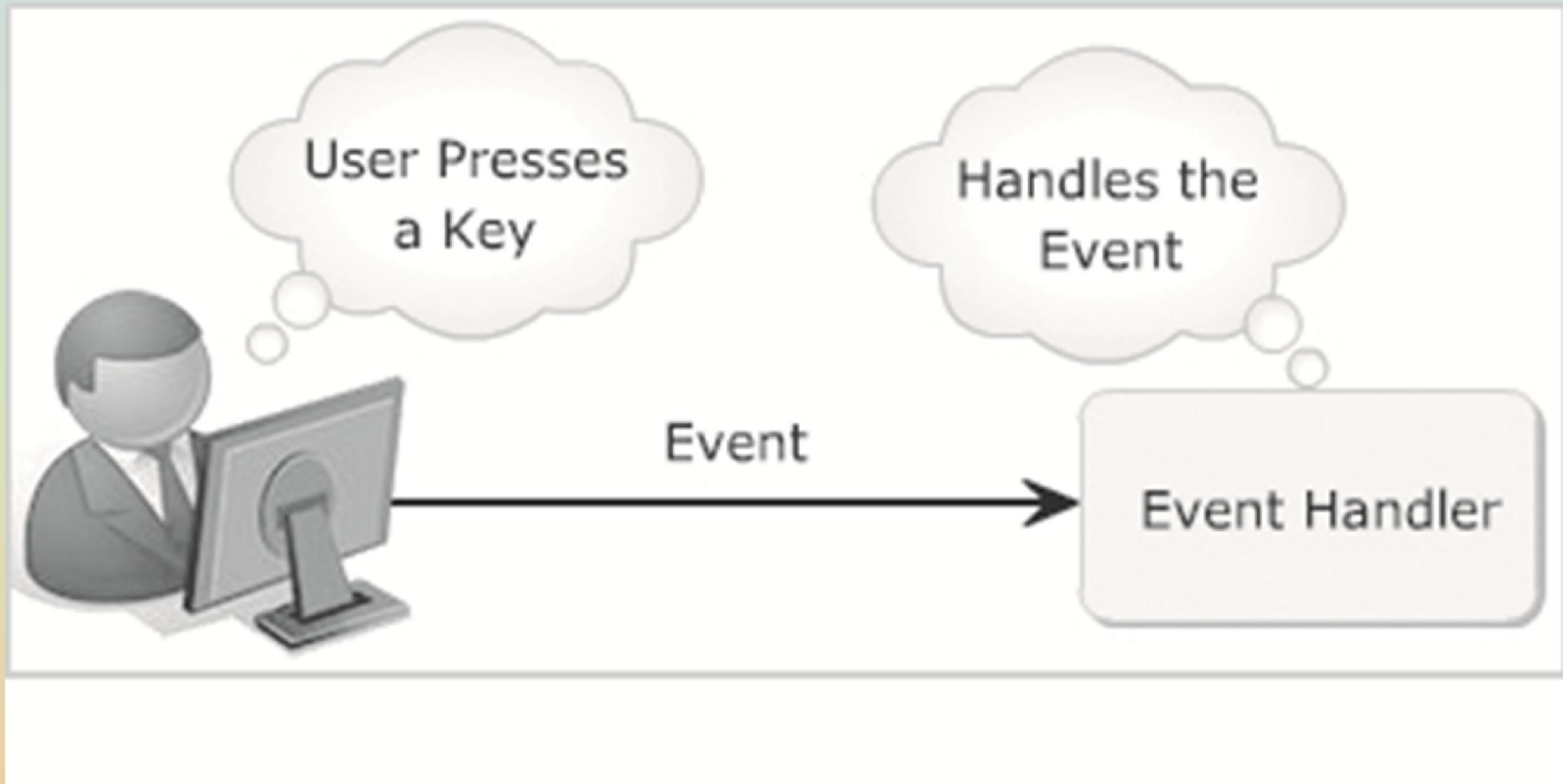
The information about this generated event is updated on the `event` object.

The `event` object is a built-in object, which can be accessed through the `window` object.

It specifies the event state, including information such as the location of mouse cursor, element on which an event occurred, and state of the keys in a keyboard.

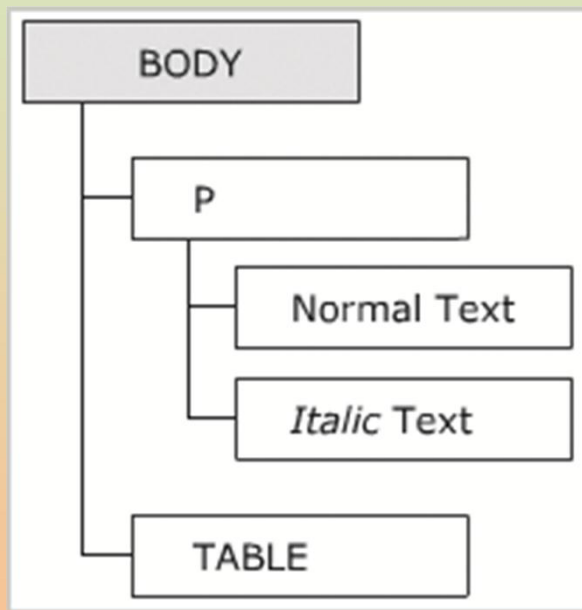
# Event Handling 2-2

- Following figure displays event handling.



# Event Bubbling

- Event bubbling is a mechanism that allows a user to specify a common event handler for all child elements.
- This means that the parent element handles all the events generated by the child elements.
- For example, consider a Web page that consists of a paragraph and a table. The paragraph consists of multiple occurrences of italic text.
- To change the color of each italic text of a paragraph when the user clicks a particular button, instead of declaring an event handler for each italic text, one can declare it within the <P> element.
- Following figure displays the event bubbling.



# Life Cycle of an Event

- An event's life starts when the user performs an action to interact with the Web page.
- It finally ends when the event handler provides a response to the user's action.
- The steps involved in the life cycle of an event are as follows:

1. The user performs an action to raise an event.

2. The event object is updated to determine the event state.

3. The event is fired.

4. The event bubbling occurs as the event bubbles through the elements of the hierarchy.

5. The event handler is invoked that performs the specified actions.

# Keyboard Events 1-3

- Keyboard events are the events that occur when a key or a combination of keys are pressed or released from a keyboard.
- These events occur for all keys of a keyboard.
- The different keyboard events are as follows:

## Onkeydown

- Occurs when a key is pressed down.

## Onkeyup

- Occurs when the key is released.

## Onkeypress

- Occurs when a key is pressed and released.

# Keyboard Events 2-3

- The Code Snippet demonstrates how to create a JavaScript code that defines the event handlers.

```
function numericonly()
{
    if(!event.keyCode >=48 && event.keyCode<=57))
        event.returnValue=false;
}

function countWords()
{
    var message = document.getElementById('txtMessage').value;
    message= message.replace(/\s+/g, ' ');
    var numberOfWords = message.split(' ').length;
    document.getElementById('txtTrack').value = words
    Remaining:  ` + eval(50 - numberOfWords);
    if(numberOfWords > 50)
        alert("too many words.");
}
```



# Keyboard Events 3-3

- In the code snippet, the function `numericOnly()` declares an event handler function, `numericOnly()`.
- The `event.keyCode` checks if the Unicode character of the entered key is greater than 48 and less than 57.
- This checks that only numeric values are entered. It also declares an event handler function, `countWords()`.
- It retrieves the text specified in the `txtMessage` control. `split()` function splits the specified string when a space is encountered and returns the length after splitting.
- It also calculates and displays the number of remaining words to complete the count of 50 words.
- If the number of words is greater than 50, an alert box is displayed.

# Mouse Events 1-4

- Mouse events occur when the user clicks the mouse button.
- Following table lists the mouse events.

Event	Description
<code>onmousedown</code>	Occurs when the mouse button is pressed
<code>onmouseup</code>	Occurs when the mouse button is released
<code>onclick</code>	Occurs when the mouse button is pressed and released
<code>ondblclick</code>	Occurs when the mouse button is double-clicked
<code>onmousemove</code>	Occurs when the mouse pointer is moved from one location to other
<code>onmouseover</code>	Occurs when the mouse pointer is moved over the element
<code>onmouseout</code>	Occurs when the mouse pointer is moved out of the element

# Mouse Events 2-4

- The Code Snippet demonstrates the use of different mouse events.

```

<tr>
  <td> Arrival Date: </td>
  <td> <input id="txtArrival" type="text" /></td>
</tr>
</table>
<table>
  <tr>
    <td> Departure Date: </td>
    <td> <input id="txtDeparture" type="text" /></td>
  </tr>
</table>
<script src="form.js">
</script>
</head>
<body>
  <table>
    <tr>
      <td> Number of Person: </td>
      <td> <input id="txtPerson" type="text" maxlength="3"
        size="3"></td>
    </tr>
  </table>
  <table>
    <tr>
      <td>
        
            submitdown.jpg');">
      </td>
      <td>
        <input id="txtName" type="text" />
        onmouseup="showImage(this, 'submit.jpg');",
        onclick="frmReservation.submit();"/>
      </td>
    </tr>
  </table>

```

# Mouse Events 3-4

```

<td>
    
</td>
</tr>
</table>
</form>
</body>
</html>

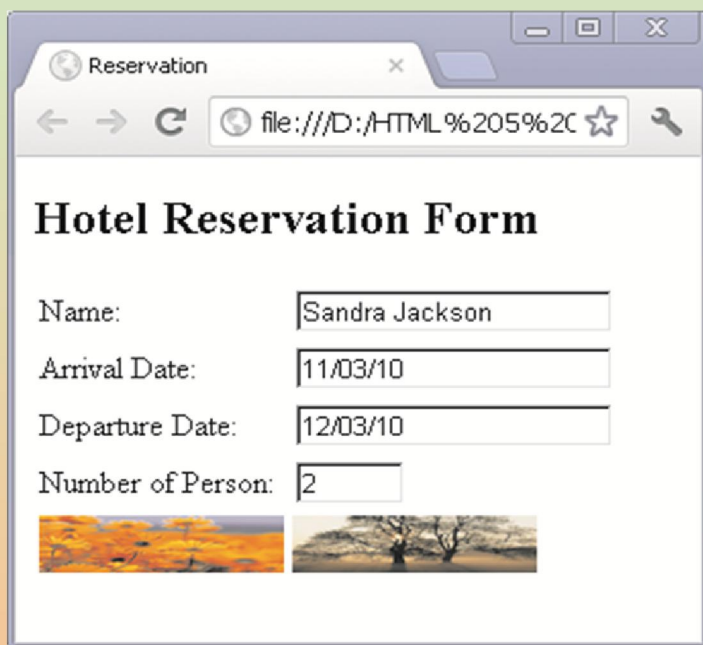
```

- In the code snippet, an image is displayed when Submit button is clicked.
- It displays the `submit.jpg` image when the mouse is released from Submit button.
- It submits the form data when the Submit button is clicked.
- It displays the image when Reset button is clicked and it displays the `reset.jpg` image when the mouse is released from Reset button.
- It resets the form data when the Reset button is clicked.

- The Code Snippet demonstrates the loading of images in a JavaScript file.

```
function showImage(object,url)
{
    object.src=url;
}
```

- Following figure displays the output of mouseup.



Reservation

file:///D:/HTML%205%2C



## Hotel Reservation Form

Name:

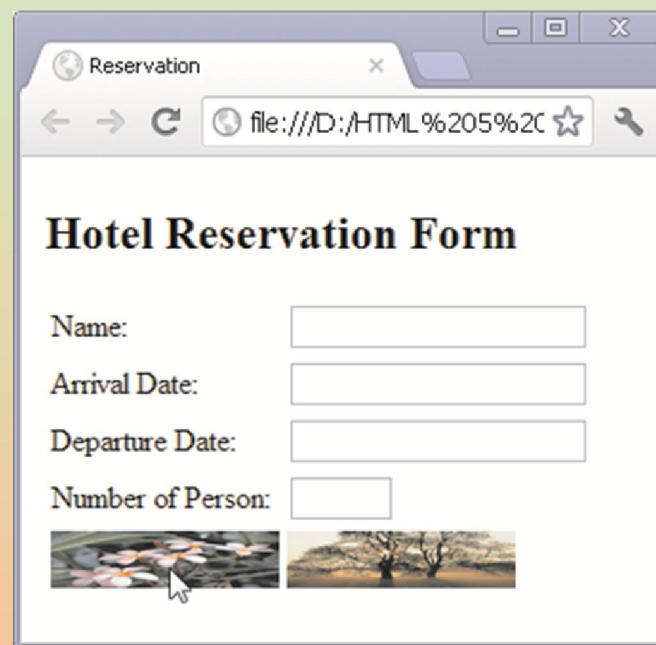
Arrival Date:

Departure Date:

Number of Person:



- Following figure displays the output on mousedown.



Reservation

file:///D:/HTML%205%2C



## Hotel Reservation Form

Name:

Arrival Date:

Departure Date:

Number of Person:



# Focus and Selection Events 1-4

- The focus events determine the activation of various elements that uses the element.
- It allows a user to set or reset focus for different elements.
- The selection events occur when an element or a part of an element within a Web page is selected.
- Following table lists the focus and selection events.

Data Type	Description
<code>onfocus</code>	Occurs when an element receives focus
<code>onblur</code>	Occurs when an element loses focus
<code>onselectstart</code>	Occurs when the selection of an element starts
<code>onselect</code>	Occurs when the present selection changes
<code>ondragstart</code>	Occurs when the selected element is moved

# Focus and Selection Events 2-4

- The Code Snippet demonstrates the use of focus and selection events.

```

</body>
<!DOCTYPE HTML>
<h2>Feedback Form</h2>
<html>
  <form id="frmreservation">
    <head>
      <table>
        <tr>
          <td>
            <label for="txtName">Name:</label></td>
            <td>
              <input id="txtName" type="text"
                {
                  onfocus="showStyle(this);"
                  field.style.backgroundColor = '#FFFFCC';
                  onblur="hideStyle(this);"
                  onselect=setFontStyle(this); />
              </td>
            </tr>
            <tr>
              <td>
                <label for="txtEmail">E-mail:</label></td>
                <td>
                  <input id="txtEmail" type="text"
                    {
                      onfocus="showStyle(this);"
                      field.style.fontWeight = 'bold';
                      onblur="hideStyle(this);"
                      field.style.fontFamily = 'Arial';
                      onselect=setFontStyle(this); />
                  </td>
                </tr>
              </script>
            </tr>
          </tr>
        </table>
      </form>
    </html>
  </body>
</html>

```

# Focus and Selection Events 3-4

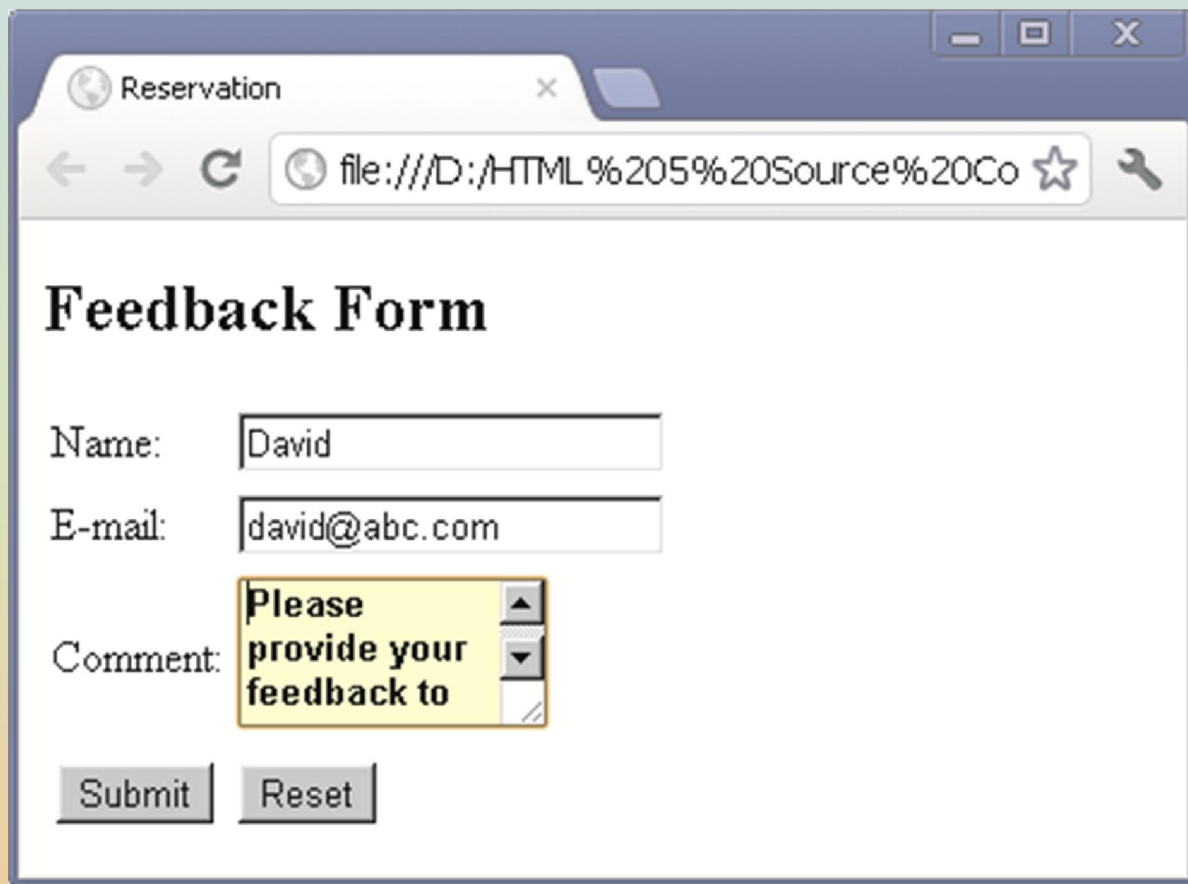
```
<tr>
  <td> <label for="txtComment">Comment:</label></td>
  <td> <textarea id="txtComment" cols="15" rows="3"
        onfocus="showStyle(this);"
        onblur="hideStyle(this);"
        onselect=setFontStyle(this);> </textarea>
  </td>
</tr>
<tr>
  <td> <input id="btnSubmit" type="button" type="button"
        value="Submit" /></td>
  <td> <input id="btnReset" type="reset" /></td>
</tr>
</table>
</form>
</body>
</html>
```

- In the code snippet, a specified style is displayed when the element receives and loses focus.
- It displays the specified font style when the element is selected.
- It declares an event handler function and specifies the background color for the field.
- It sets the font style for text to bold and the text should appear in `Arial` font.



# Focus and Selection Events 4-4

- Following figure displays the focus and selection events.



The screenshot shows a web browser window titled "Reservation" with a single tab. The address bar displays the file path: `file:///D:/HTML%205%20Source%20Co`. The page content is a "Feedback Form" with the following elements:

- Name:** A text input field containing the text "David".
- E-mail:** A text input field containing the text "david@abc.com".
- Comment:** A text area containing the text "Please provide your feedback to". The text area has a yellow background and a border. To the right of the text area are two small buttons with up and down arrows, and a small icon at the bottom right.
- Submit:** A button labeled "Submit".
- Reset:** A button labeled "Reset".



# jQuery 1-2

- jQuery is a short and fast JavaScript library developed by John Resig in 2006 with a wonderful slogan: **Write less and do more.**
- jQuery simplifies client-side scripting, HTML files animation, event handling, traversing, and developing AJAX based Web applications.
- It helps in rapid Web application development by writing lesser code.
- Following are the key features supported by jQuery.

## Event Handling

- jQuery has a smart way to capture a wide range of events, such as user clicks a link, without making the HTML code complex with event handlers.

## Animations

- jQuery has many built-in animation effects that the user can use while developing their Web sites.

## DOM Manipulation

- jQuery easily selects, traverses, and modifies DOM by using the cross-browser open source selector engine named Sizzle.

## Cross Browser Support

- jQuery has a support for cross-browser and works well with the following browsers:
  - Internet Explorer 6 and above
  - Firefox 2.0 and above
  - Safari 3.0 and above
  - Chrome
  - Opera 9.0 and above

## Lightweight

- jQuery has a lightweight library of 19 KB size.

## AJAX Support

- jQuery helps you to develop feature-rich and responsive Web sites by using AJAX technologies.

## Latest Technology

- jQuery supports basic XPath syntax and CSS3 selectors.

# HTML Using jQuery Library

- To work with jQuery perform the following steps:

1. Download the jQuery library from the <http://jquery.com/> Web site.

2. Place the jquery-1.7.2.min.js file in the current directory of the Web site.

- The user can include jQuery library in their file.
- The Code Snippet shows how to use a jQuery library.

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>The jQuery Example</title>
    // Using jQuery library
    <script src="jquery-1.7.2.min.js">
      // The user can add our JavaScript code here
    </script>
  </head>
  <body>
    </body>
</html>
```



# Calling jQuery Library Functions 1-2

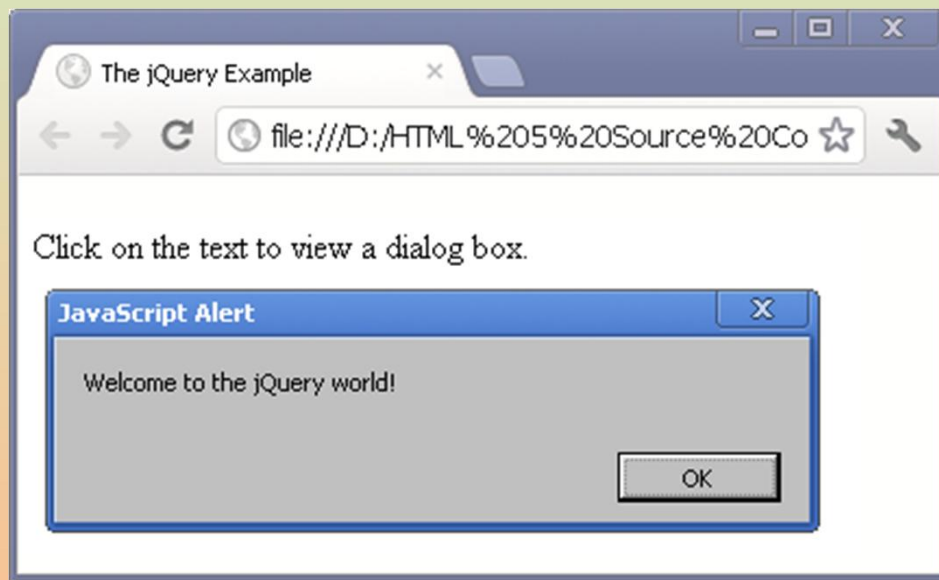
- While jQuery is reading or manipulating the Document Object Model (DOM) object, the users can add the events when the DOM object is ready.
- If the user wants the event on their page then the user has to call the event in the `$(document).ready()` function.
- The users also register the ready event for the document.
- Place the `jquery-1.7.2.min.js` file in the current directory and specify the location of this file in the `src` attribute.
- The Code Snippet shows how to call jQuery library function and ready event in DOM.

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>The jQuery Example</title>
    <script src=" jquery-1.7.2.min.js">
    </script>
    <script>
      $(document).ready(function() {
        $("div").click(function() {
          alert("Welcome to the jQuery world!");
        });
      });
    </script>
```

# Calling jQuery Library Functions 2-2

```
</head>
<body>
  <div id="firstdiv">
    Click on the text to view a dialog box.
  </div>
</body>
</html>
```

- The code snippet includes the jQuery library and also registers the ready event for the document.
- The ready event contains the click function that calls the click event.
- Following figure displays the output of jQuery.



# jQuery Mobile 1-6

- jQuery mobile is a Web User Interface (UI) development framework that allows the user to build mobile Web applications that work on tablets and smartphones.
- The jQuery mobile framework provides many facilities that include XML DOM and HTML manipulation and traversing, performing server communication, handling events, image effects, and animation for Web pages.
- The basic features of jQuery mobile are as follows:

## Simplicity

- This framework is easy to use and allows developing Web pages by using markup driven with minimum or no JavaScript.

## Accessibility

- The framework supports Accessible Rich Internet Applications (ARIA) that helps to develop Web pages accessible to visitors with disabilities.

## Enhancements and Degradation

- The jQuery mobile is influenced by the latest HTML5, JavaScript, and CSS3.





# jQuery Mobile 2-6

## Themes

- This framework provides themes that allow the user to provide their own styling.

## Smaller Size

- The size for jQuery mobile framework is smaller for CSS it is 6KB and for JavaScript library it is 12KB.

- The Code Snippet shows an example of a jQuery mobile.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
    <link rel="stylesheet" href="jquery.mobile-1.0a3.min.css" />
    <script src="jquery-1.5.min.js"></script>
    <script src="jquery.mobile-1.0a3.min.js"></script>
  </head>
  <body>
    <div data-role="page">
      <div data-role="header">
        <h1>Car Rental</h1>
      </div>
```





# jQuery Mobile 3-6

```
<div data-role="content">
  <p>Choose from the listed car models</p>
  <ul data-role="listview" data-inset="true">
    <li><a href="#">Ford</a></li>
    <li><a href="#">Ferrari</a></li>
    <li><a href="#">BMW</a></li>
    <li><a href="#">Toyota</a></li>
    <li><a href="#">Mercedes-Benz</a></li>
  </ul>
</div>
<div data-role="footer">
  <h4>&copy; DriveCars 2012.</h4>
</div>
</div>
</body>
</html>
```

- The jQuery mobile application should have the following three files:
  - CSS file
  - jQuery library
  - jQuery Mobile library



# jQuery Mobile 4-6

- In this code snippet, three files are included, the CSS (`jquery.mobile-1.0a3.min.css`), jQuery library (`jquery-1.5.min.js`), and the jQuery mobile library (`jquery.mobile-1.0a3.min.js`).
- A user can also download the jQuery libraries from <http://code.jquery.com/> Web site.

The jQuery Mobile takes HTML tags and renders them on mobile devices. To work with this, HTML has to make use of data attributes.

jQuery uses these attributes as indicators for rendering it on the Web pages.

jQuery also looks for div using a particular data-role values such as page, content, header, and footer.

There are multiple div blocks added to the code for page, content, header, and footer.

To display the different car models a data-role listview is added to enhance the look and feel of the mobile Web page.



# jQuery Mobile 5-6

- A user need to install the Opera Mobile Emulator from the Opera Web site.
- After installing the **Opera Mobile Emulator**, perform the following steps to apply settings to the emulator:

1. Select **All Programs → Opera Mobile Emulator → Opera Mobile Emulator**.The **Opera Mobile Emulator** dialog box will be displayed.

2. In the **Profile** tab, select the **Samsung Galaxy Tab**.

3. In the **Resolution** drop-down, select the **WVGA Portrait(480x800)**.

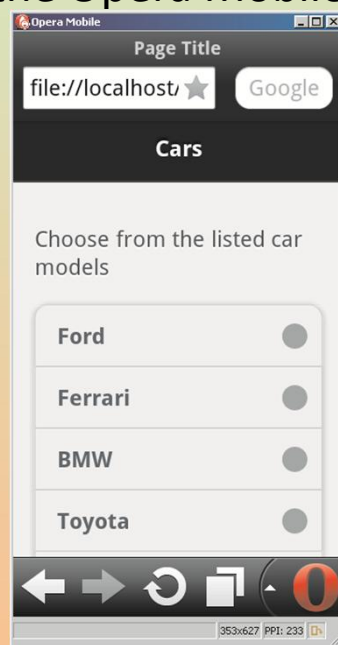
4. Click **Update**.

5. Click **Launch**. The **Samsung Galaxy** tab is displayed.



# jQuery Mobile 6-6

- A user needs to install the Opera Mobile Emulator from the Opera Web site and perform the following steps to apply settings to the emulator:
  1. Add the Opera Mobile Emulator in the CoffeeCup editor by clicking **Tools** → **Additional Browsers** → **Test with Additional Browser 1** and give the location of the Opera Mobile Emulator installed on your system.
  2. Open the jQuery file in the **CoffeeCup** editor and save.
  3. Click **Tools** → **Additional Browser** → **Test with Additional Browser 1**.
- Following figure displays the Opera Mobile Emulator.



# Summary

- Scripting refers to a series of commands that are interpreted and executed sequentially and immediately on an occurrence of an event.
- JavaScript is a scripting language, which can be executed on the client-side and on the server-side.
- A variable refers to a symbolic name that holds a value, which keeps changing.
- A primitive data type contains a single literal value such as a number or a string.
- A function is a piece of code that performs some operations on variables to fulfill a specific task.
- Event handling is a process of specifying actions to be performed when an event occurs.
- Event bubbling is a mechanism that allows you to specify a common event handler for all child elements.
- jQuery mobile is a Web User Interface development framework that allows the user to build mobile Web applications that works on tablets and smartphones.