JavaScript

HTML with JavaScript

JavaScript codes are embedded in an HTML document in two ways:

- As statements and functions using the `<script>` tag.
- As event handlers using HTML tags.

Structural Outline

```
<html>

<head>
<title>New Page 1</title>

<script language="javascript">
      // JavaScript code here: mainly JavaScript functions
</script>

</head>

<body>
      // HTML code here

      // JavaScript code here too: mainly event triggers
</body>

</html>
```

The LANGUAGE attribute specifies the scripting language as follows:

```
<script language="javascript">
      // JavaScript code here
</script>
```

JavaScript Example

The following is an example of the <script> tag embedded in a HTML document using the format:

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    var title = "JavaScript Programming";
    document.write("Welcome to " + title);
</script>

</body>

</html>
```

JavaScript Code Hiding

JavaScripts can be placed inside comment fields to ensure that your code is not displayed by older browsers that do not recognize JavaScript.

```
<script language="javascript">
<!-- Begin to hide script contents from browsers.
         :
     End the hiding here. -->
</script>
```

JavaScript Statements

A JavaScript program consists of one or more statements within <script> tags. A statement may be:

- variable assignment
- procedure call (no return value)
- function call

A function or procedure must be defined before they can be used or called.

Scripts placed within <script> tags are evaluated after the page loads. Functions are stored, but not executed. Functions are invoked by events in the page.

JavaScript Example

Generally, you should define the functions for a page in the <head> portion of a document. Since the <head> section is loaded first, this practice guarantees that functions are loaded before the user has a chance to do anything that might call a function.

```
<html>

<head>
<title>New Page 1</title>

<script language="javascript">
    function square(i)
    {
        return i * i;
    }
</script>

</head>

<body>

<script language="javascript">
    document.write("The function returned "+ square(5));
</script>

</body>

</html>
```

Some Simple Built-in Functions

The page that you are viewing in a Web browser is called a “document”. Some useful functions:

- `document.write("...any msg....");`
- `document.writeln("line1 \n line2 \n line3 \n");`
- `document.writeln("<br>");`
- `document.close();`
- `document.clear();`

You may pop-up windows on the screen using:

- `window.alert("This is a Pop-Up !");`
- `window.prompt("...any msg...", "value");`

Built-In Variables or Properties

Besides functions, JavaScript also has built-in variables. It uses these variables to keep track of things within itself. Built-in variables are also called properties.

Useful built-in variables or properties:

- `navigator.appName`
- `navigator.appCodeName`
- `navigator.appVersion`

More built-in properties:

- `window.location.href`
- `window.status`

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    var myname, reply;
    myname = window.prompt ("What is your name ?", "Your name");
    document.write ("Your name is "+ myname);
    document.write ("<BR>");
    document.write ("You are using "+ navigator.appName);
    document.write ("<BR>");
    window.alert ("Good-bye");
</script>

</body>

</html>
```

Programming Basics

JavaScript is similar in structure to Java and C language. It is placed within a HTML document to be interpreted by the browser. It can be used to create a dynamic web page that responds to user interaction.

Comments

Generally, comments are usually written before the declaration of a variable or function. Comments are ignored by the interpreter.

```
// comment on one line
/* comment on one or more lines */
```

Statement

A statement is always terminated with a semi-colon (;). It can be an assignment, a procedure call or a function call. Examples:

```
Area = PI * 20 * 20;
document.writeln ("Hello World");
calculate ( );
result = square (5);
```

Variables

Variables appear in assignment statements. You use variables to hold values in your application. You name these variables so that you can reference them, and there are certain rules to which the names must follow. A variable name must adhere to the following rules :

- Begin with an alphabet, any subsequent character can be an alphabet, numeric, or underscore.
- No spaces or special characters are allowed within the variable.

Valid variable names are:

- `firstname`
- `last_name`
- `time1`

Keywords

The reserved words in this list cannot be used in naming variables, functions, methods, or objects in JavaScript. Some of these words are keywords used in JavaScript, others are reserved for future use.

abstract	else	int	switch
boolean	extends	interface	synchronized
break	false	long	this
byte	final	native	throw
case	finally	new	throws
catch	float	null	transient
char	for	package	true
class	function	private	try
const	goto	protected	typeof
continue	if	public	var
default	implements	return	void
delete	import	short	while
do	in	static	with
double	instanceof	super	

Data Types

There are different types of data that can be assigned to variables in JavaScript:

- Number
```
age = 28;
salary = 3645.30;
```

- Boolean
```
paid = true;
ans = (7<5);
```

- String
```
dept = "Marketing";
organization = "SUN"+dept+"Dept";
```

Control characters can be embedded in strings. Example: "This is a wonderful \n Song"
Other control characters:

- '\b' indicates a backspace.
- '\f' indicates a a form feed.
- '\n' indicates a new line character.
- '\r' indicates a carriage return.
- '\t' indicates a tab character.

Data Conversion

JavaScript is a loosely typed language. This means that you do not have to specify the data type of a variable when you declare it, and data types are converted automatically as needed during execution.

So, for example, you could define a variable as follows :

```
var answer = 42;
```

And later, you could assign the same variable with a string value as follows :

```
answer = "This is interesting...";
```

JavaScript provides several special functions for converting string to numeric values:

- eval attempts to evaluate a string representing any JavaScript literals or variables, converting it to a number.
  Example: `ans = eval ("2+3*7");`

- parseInt converts a string to an integer.
  Example: `day = parselnt ("30");`

- parseFloat converts a string to a floating-point number, if possible.
  Example: `length = parseFloat ("1.233");`

Functions

Program statements may be grouped into a function to perform a particular task. A function definition begins with the keyword function followed by the name you assign to the function and a pair of parentheses. Usually you will need to pass certain information to the function as parameters.

For example :

```
function areaOfCircle(radius)
{
      var area = 3.14 * radius * radius;
      return area;
}
```

Operators

There are five different types of operators supported in JavaScript :

- Arithmetic
  `+, -, *, / , %, ++, --`

  Examples:
  ```
  first = 1;
  second = 2;
  sum = first + second + 5;
  sum ++ ;
  remainder = sum %5 ;
  ```

- Comparison
  `==, !=, >, <, >=, <=`

  Used in flow control statements.
  ```
  if (password == "abc") {
        :
  }
  ```

- Logical
  `&&, ||, !`

  Used in flow control statements.
  ```
  if((username == "alex") && (age>12)){
        :
  }
  ```

- Bitwise (Binary)
  `&, |, ^, ~`

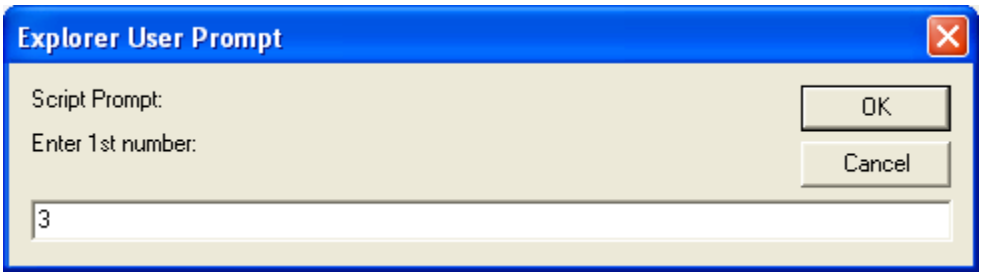
  Examples:
  What is a & 4 ?
  What is a | 7 ?
  What is a ^ 7 ?
  What is ~a ?

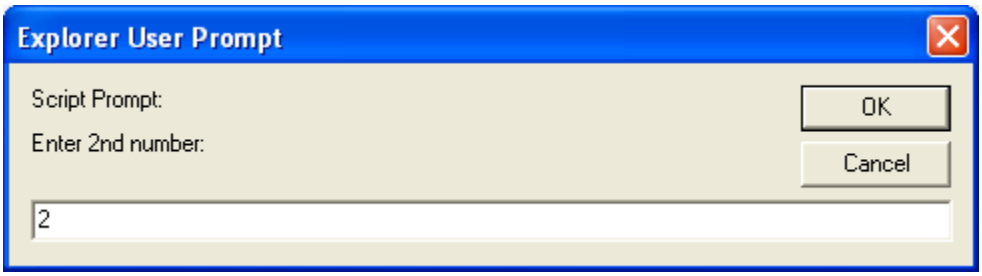
- Assignment
  `-=, +=, \=, *=, %=`

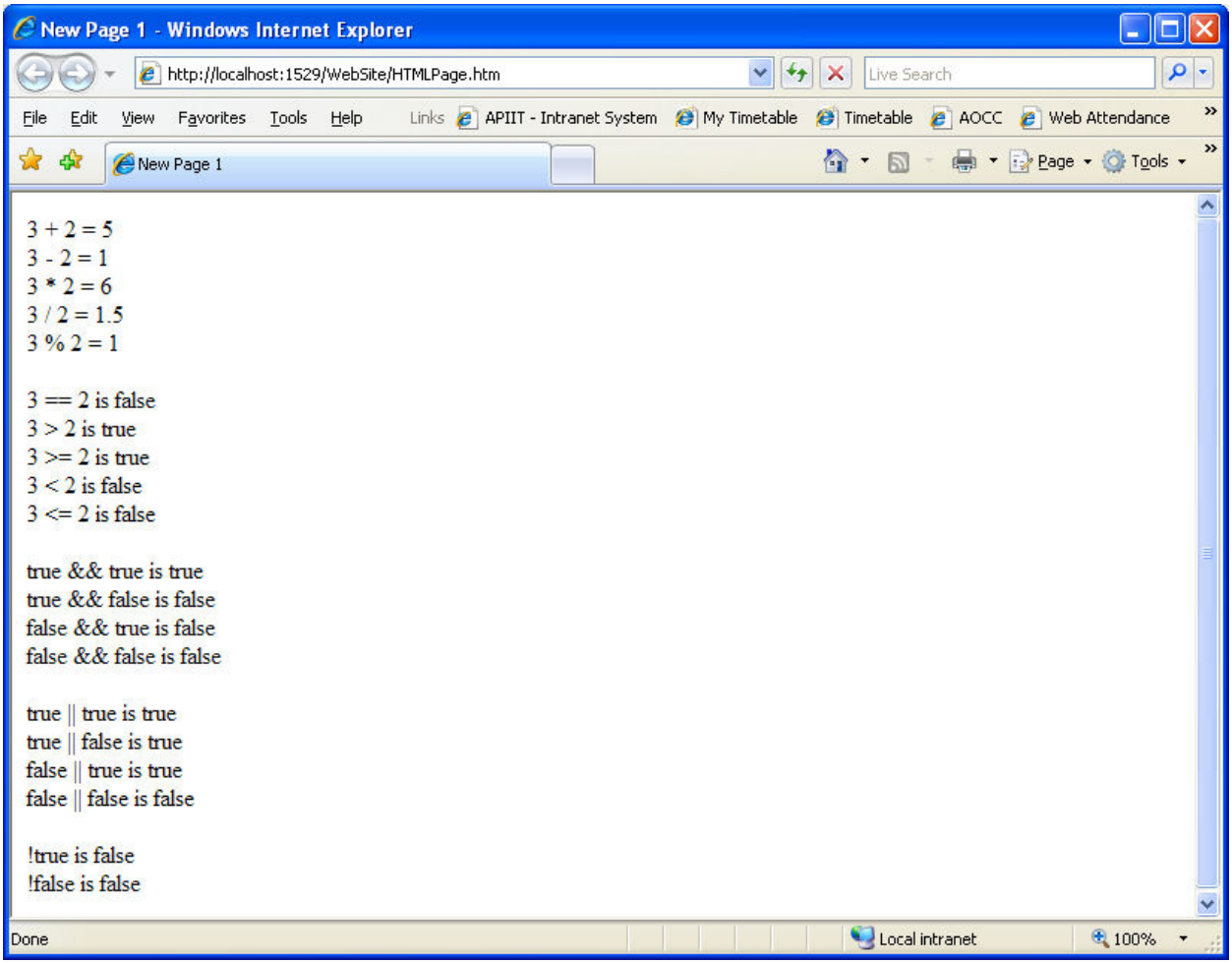
  Examples:
  ```
  a = 5;
  a += 1;
  a -= 2;
  a *= 3;
  ```

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    var n1, n2, result;
    n1 = window.prompt("Enter 1st number: ", "0");
    n2 = window.prompt("Enter 2nd number: ", "0");

    result = parseFloat(n1) + parseFloat(n2);
    document.write(n1, " + ", n2, " = ", result, "<br>");

    result = parseFloat(n1) - parseFloat(n2);
    document.write(n1, " - ", n2, " = ", result, "<br>");

    result = parseFloat(n1) * parseFloat(n2);
    document.write(n1, " * ", n2, " = ", result, "<br>");

    result = parseFloat(n1) / parseFloat(n2);
    document.write(n1, " / ", n2, " = ", result, "<br>");

    result = parseFloat(n1) % parseFloat(n2);
    document.write(n1, " % ", n2, " = ", result, "<br><br>");

    document.write(n1, " == ", n2, " is ", (n1 == n2), "<br>");
    document.write(n1, " > ", n2, " is ", (n1 > n2), "<br>");
    document.write(n1, " >= ", n2, " is ", (n1 >= n2), "<br>");
    document.write(n1, " < ", n2, " is ", (n1 < n2), "<br>");
    document.write(n1, " <= ", n2, " is ", (n1 <= n2), "<br><br>");

    document.write(true, " && ", true, " is ", (true && true), "<br>");
    document.write(true, " && ", false, " is ", (true && false), "<br>");
    document.write(false, " && ", true, " is ", (false && true), "<br>");
    document.write(false, " && ", false, " is ", (false && false),
"<br><br>");

    document.write(true, " || ", true, " is ", (true || true), "<br>");
    document.write(true, " || ", false, " is ", (true || false), "<br>");
    document.write(false, " || ", true, " is ", (false || true), "<br>");
    document.write(false, " || ", false, " is ", (false || false),
"<br><br>");

    document.write("!true is ", !true, "<br>");
    document.write("!false is ", !true, "<br>");

</script>

</body>

</html>
```


Explorer User Prompt
Script Prompt:
Enter 1st number:
OK
Cancel
3


Explorer User Prompt
Script Prompt:
Enter 2nd number:
OK
Cancel
2


New Page 1 - Windows Internet Explorer
http://localhost:1529/WebSite/HTMLPage.htm
Live Search
File Edit View Favorites Tools Help
Links APIIT - Intranet System My Timetable Timetable AOCC Web Attendance
New Page 1
Page Tools
3 + 2 = 5
3 - 2 = 1
3 * 2 = 6
3 / 2 = 1.5
3 % 2 = 1
3 == 2 is false
3 > 2 is true
3 >= 2 is true
3 < 2 is false
3 <= 2 is false
true && true is true
true && false is false
false && true is false
false && false is false
true || true is true
true || false is true
false || true is true
false || false is false
!true is false
!false is false
Done
Local intranet
100%

Exercises:

Part 1

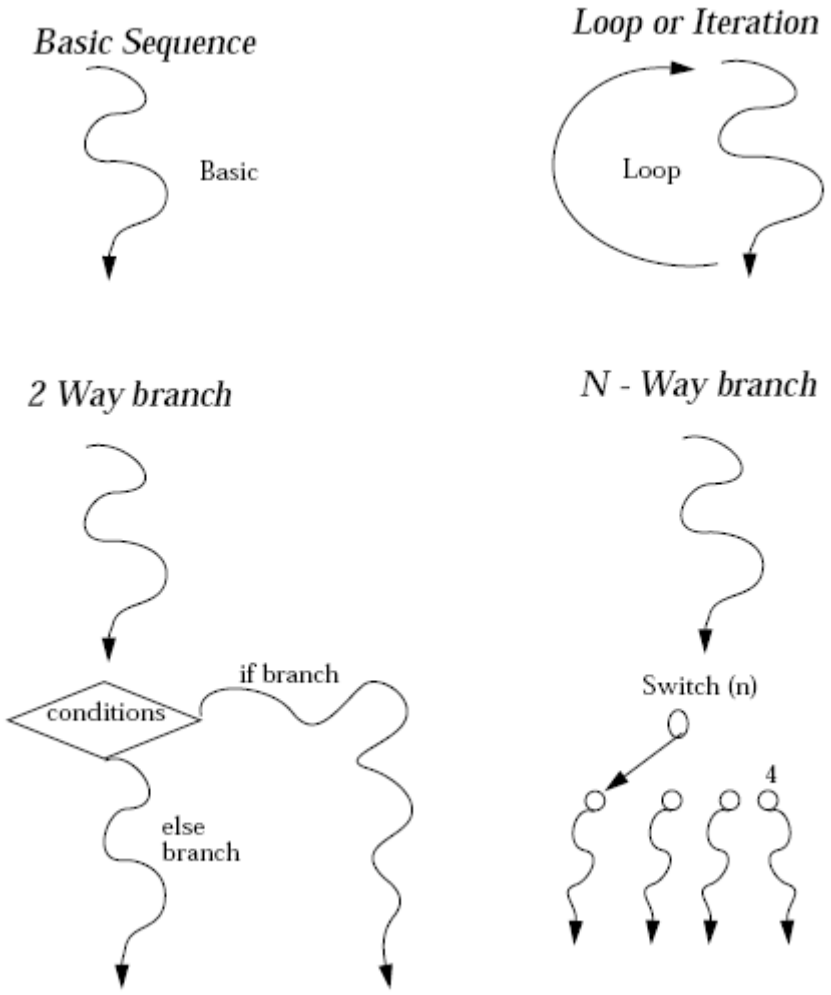
Try the sample JavaScript 01 -03 attached with this document.

Part 2

Write a JavaScript program for each of the following

1. Compute and output the total of three integer number entered from keyboard.
2. Compute and output the average of two floating-point numbers entered from keyboard.
3. Input person's name and year born to the web form and display the current age.

Program Sequencing / Flow Control



In the life of a program or script, decisions will be made, branches taken. Sometimes the same steps need to be repeated. Flow control is about how sequences of statements are executed.

Flow Control in JavaScript

JavaScript supports a set of statements that you can use to incorporate a great deal of interactivity in web pages. The statements are:

- Conditional statements
  `if...else`
  `switch`

- Loop statements
  `for`
  `while`

Conditional Statements

Conditional statements control the sequence in which JavaScript executes the statements. If the logical condition is true, one branch is taken. Else another branch is taken.

- if, else

```
if (condition) {
      // perform task if condition is true
}
else {
      // perform task if condition is false
}
```

The condition may be any JavaScript expression that evaluates to true or false. Multiple statements must be enclosed in braces.

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    var score;
    score = window.prompt("Enter your score: ", "0");
    score = parseFloat(score);

    if(score > 50)
        document.write("Passed");
    else
        document.write("Failed");
</script>

</body>

</html>
```

- Switch

```
switch (var_name) {
      case value1 : // branch 1 statements
                    break;
      case value2 : // branch 2 statements
                    break;
      default :     // default branch
}
```

value1, value2 should be of the same data type as var_name.

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    var str;
    str = window.prompt("Enter your mobilephone first 3 digits: ", "XXX");

    switch(str)
    {
        case "012": document.write("You are Maxis user.");
                    break;
        case "013": document.write("You are Celcom user.");
                    break;
        case "016": document.write("You are Digi user.");
                    break;
        case "017": document.write("You are Maxis user.");
                    break;
        case "019": document.write("You are Celcom user.");
                    break;
        default:    document.write("Your mobile network service can not be
                    identified!");
    }
</script>

</body>

</html>
```

Loop Statements

Loop statements allow a number of statements to be executed repeatedly, based on a condition. JavaScript supports two loop structures: for and while.

- for

```
for ( init_expression; condition; update_expression )
{
      // statements
}
```

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    for (var i = 1; i <=20; i++)
    {
        document.write(i);
        document.write("<br>");
    }
</script>

</body>

</html>
```

- `while`

  A while statement repeats a loop as long as a specified condition evaluates to true. A while statement looks as follows:

```
while ( condition )
{
      // statements
}
```

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    var i = 0;

    while(i <=20)
    {
        document.write(i);
        document.write("<br>");
        i++;
    }
</script>

</body>

</html>
```

More PopupWindow Functions

- `window.alert("Any user message");`
- `reply_string = window.prompt("prompt", "default ans");`
- `boolean_flag = window.confirm("Do you want to quit ?");`
- `window.open("html file", "windowname", "width =200, height=100");`
- `window.close( );`

JavaScript Example

```
<html>

<head>
<title>New Page 1</title>
</head>

<body>

<script language="javascript">
    ans = window.prompt("How many times have you tried?", "0");

    if(parseInt(ans) < 3)
    {
        helpflag = window.confirm ("Do you want some help ?");

        //if (helpflag == true)
            window.open("http://www.google.com", "Google");
    }

    quitflag = window.confirm ("Click OK when you are ready to exit.");
    if (quitflag) window.close ( );
</script>

</body>

</html>
```

Exercises:

Part 1

Try the sample JavaScript 04 -11 attached with this document.

Part 2

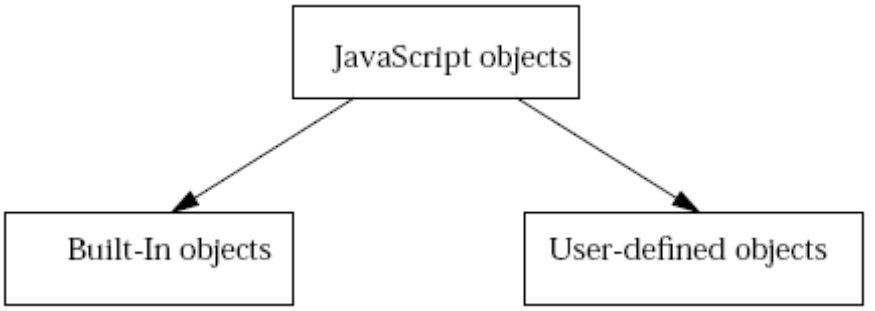
1. Write a script that determines the integer entered is either odd or even.

2. Write a script that finds the smallest and the largest numbers of three numbers entered.

2. Write a script that calculates the total of the odd integers from 1 – 99 and then output HTML text that display the result.

Objects in JavaScript

- An object is a “THING”. It is usually visible on the screen. However, it may also be intangible.



- You have already worked with the document, window and navigator objects. Focus on “Form” objects in this module.

Inside an object : Properties&Methods

- An object contains 2 important elements within itself. Technically, this is called encapsulation.

  Variables or “Properties”
  Examples:
  “`appName`” is a property of “`Navigator`” object
  (we write : “`navigator.appName`”)

  “`href`” is a property of “`window.location`” object
  (we write : “`window.location.href`”)

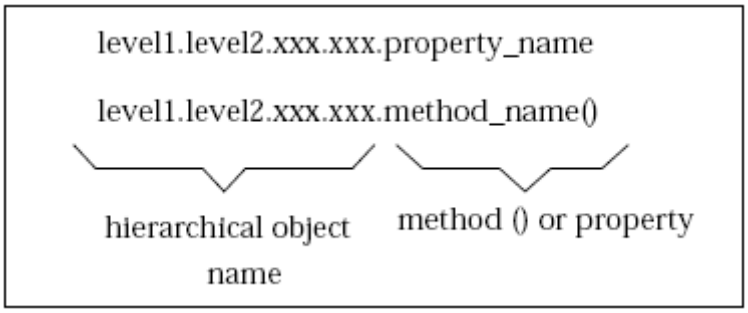
  Functions or “ methods”
  Examples:
  “`alert ()`” is a method of “window” object
  (we write: “`window.alert ()`”)

  “`write ()`” is a method of “document” object
  (we write: “`document.write ()`”)

Object Names

- Every object must have a name
- Object name is used by JavaScript code to manipulate the object
- If an object is contained within another object, it will have a hierarchical name.

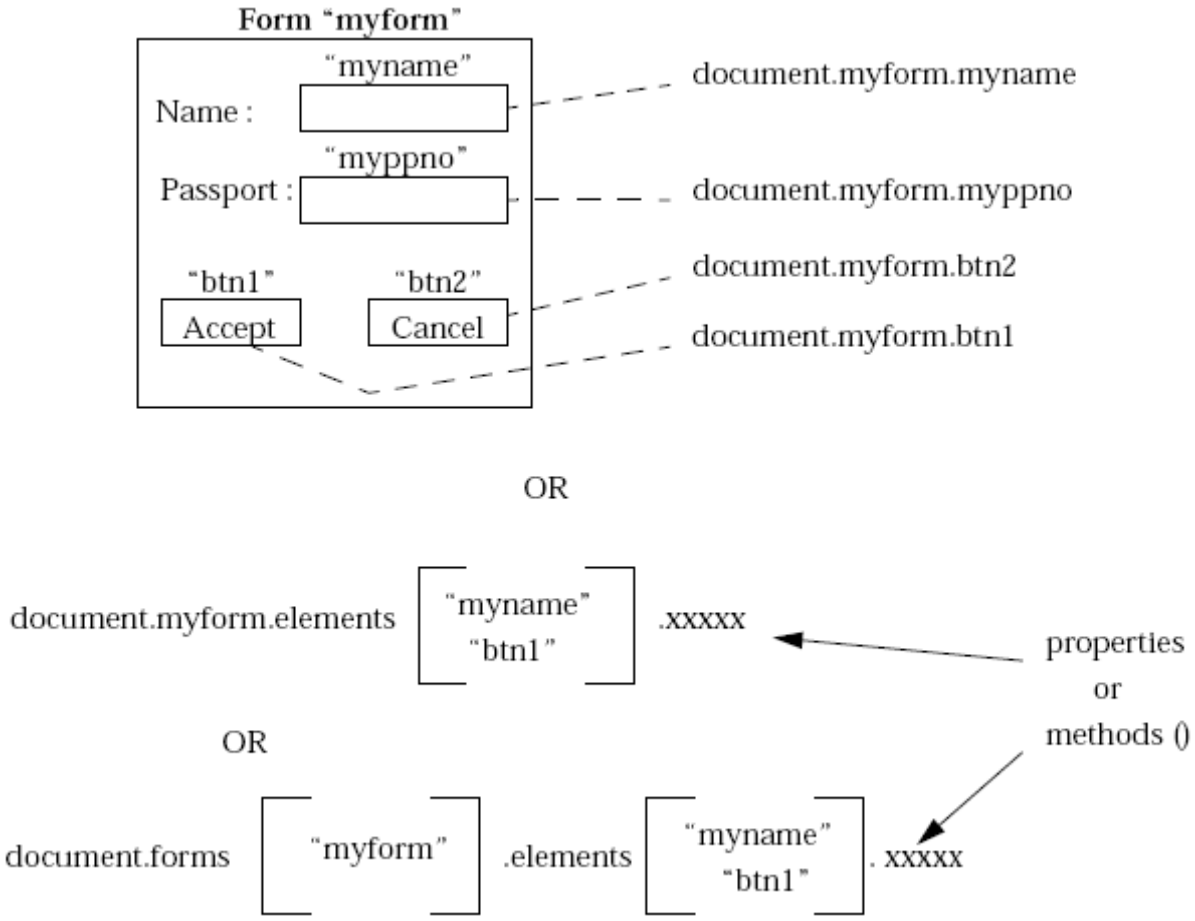
  Convention



JavaScript Example

```
<html>

<head>
<title>New Page</title>
<script language="javascript">
function changeBgColor()
{
      document.body.bgColor = "orange";
}
</script>

</head>

<body>

<p><input type="button" value="Change Backgound Color"
name="buttonChangeBgColor" onclick="changeBgColor()"></p>

</body>

</html>
```

Hierarchical Naming Convention in Forms



- Hierarchical object name is "Top-Down".
- Last component is always a property or a method.
- Same object can be referenced in different ways.

Form Manipulation Examples

```
<html>

<head>
<title>New Page</title>
<script language="javascript">
      function toUpper()
      {
            document.myForm.firstName.value
                  = document.myForm.firstName.value.toUpperCase();
            document.myForm.lastName.value
                  = document.myForm.lastName.value.toUpperCase();
      }
      function toLower()
      {
            document.myForm.firstName.value
                  = document.myForm.firstName.value.toLowerCase();
            document.myForm.lastName.value
                  = document.myForm.lastName.value.toLowerCase();
      }
</script>
</head>

<body>

<form action="#" name="myForm">
      <p>First Name: <input type="text" name="firstName" size="20"></p>
      <p>Last Name: <input type="text" name="lastName" size="20"></p>
      <p><input type="button" value="To Upper" name="buttonToUpper"
onclick="toUpper()">
      <input type="button" value="To Lower" name="buttonToLower"
onclick="toLower()"></p>
</form>

</body>

</html>
```

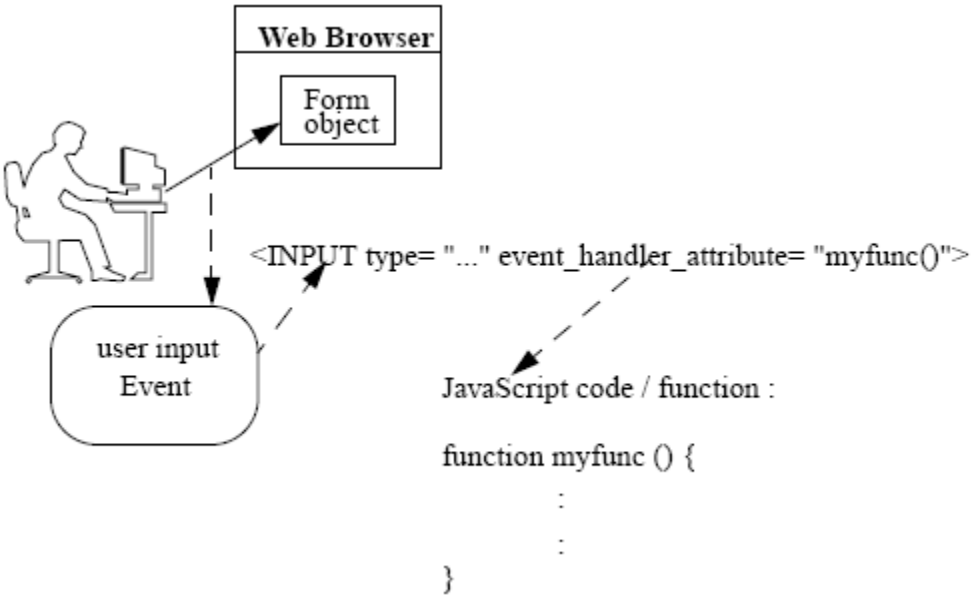
JavaScript Event

- JavaScript is an event driven language.
- Events are actions that take place in a document, usually are the result of user activity, such as clicking on a button or selecting text in a field.
- They are mainly related to user interaction with `<FORM>` elements.

JavaScript Event Handlers

- Events take place as follows. User interact with `<FORM>`/`<INPUT>` object.
- The object in turn causes some user defined JavaScript function or code to be executed.
- Event handlers provide the link between `<FORM>`/`<INPUT>` elements and the JavaScript function.
- Event handlers are JavaScript-related HTML attributes that modify the behavior of `<FORM>`/`<INPUT>` tags.
- JavaScript event handlers can be divided into the following categories: system events, mouse events, keyboard events, data-entry events.

How Event Handling Works



System Events

- System events don't require user interaction in order to be activated.
- For example, the loading and unloading of web page.

`Onload`	`OnLoad` event is activated after your HTML page is completely loaded. Attribute of `<BODY>` tag.
`OnUnload`	`OnUnLoad` event is useful for final cleanup before the HMTL page is unloaded.
`OnAbort`	When user stops loading images on a Web page by clicking on browser "Stop" button.Attribute of `<IMG>`

JavaScript Example

```
<html>

<head>
<title>New Page</title>
<script language="javascript">
    function welcome()
    {
        window.alert("Welcome!!!");
    }
    function bye()
    {
        window.alert("See you again.");
    }
</script>
</head>

<body onload="welcome()" onunload="bye()">

</body>

</html>
```

Mouse Events

- Mouse events will require user interaction in order to be triggered.
- They have to do with mouse movements and mouse clicks.

OnClick	▪ The most basic of the mouse events is the OnClick handler. ▪ This event is activated whenever you click an object that accepts such an event. Object accepting an OnClick event are links, checkboxes and the buttons (including submit, reset and radio buttons). ▪ For links, OnClick can be used to control whether a page jump will occur when the user clicks on a hot link. ▪ Attribute of <INPUT>, <A> tags.
OnMouseOut	▪ OnMouseOver event occurs when the mouse crosses over an object. You can ▪ use the OnMouseOver event to explain a link or an image area. ▪ Attribute of <A> and <AREA>tags.
OnMouseDown	▪ Introduced at Netscape Navigator/Communicator 4.0. Occurs when user holds down a mouse button over a button, link or area object. ▪ Attribute of <INPUT type= "button">, <A> and <AREA> tags. ▪ JavaScript code similar to OnMouseOver.
OnMouseUp	▪ Counterpart to OnMouseDown with similar attributes and JavaScript coding. ▪ Occurs when user releases a mouse button.

JavaScript Example

```
<html>

<head>
<title>New Page</title>
</head>

<body>

    <img src="http://us.i1.yimg.com/us.yimg.com/i/ww/beta/y3.gif"
    onmouseover="javascript:
this.src='http://www.google.com.my/intl/en_com/images/logo_plain.png'"
    onmouseout="javascript:
this.src='http://us.i1.yimg.com/us.yimg.com/i/ww/beta/y3.gif'"></img>

</body>

</html>
```