UNIT V : REGULAR EXPRESSION, ROLLOVER & FRAMES

Regular Expression

- A regular expression is an object that **describes a pattern of characters.**
- The JavaScript **RegExp** class **represents regular expressions**, and both String and **RegExp define methods that use regular expressions to** perform powerful **pattern-matching and search-and-replace functions on text.**
- Syntax: A regular expression could be defined with the RegExp () constructor, as follows –
- **var pattern = new RegExp(pattern, attributes);**
- **var pattern = /pattern/attributes;**

Regular Expression

- Here is the description of the parameters –
- **pattern –** A string that specifies the **pattern of the regular expression** or another regular expression.
- **attributes –** An optional string containing any of the **"g", "i", and "m" attributes that specify global**, case-insensitive, and multi-line matches, respectively.

```
<html>
   <body>
      <script type = "text/javascript">
      var str = "Javascript is an interesting scripting language";
      var re = new RegExp( "script", "g" );
      var result = re.exec(str);
      document.write("Test 1 - returned value : " +   result);

      re = new RegExp( "pushing", "g" );
      var result = re.exec(str);
      document.write("<br />Test 2 - returned value : " +   result);
   </script>
   </body>
</html>
```

Output:

```
Test 1 - returned value : script
Test 2 - returned value : null
```

Language Of Regular Expression

- The **words of regular expression** are called **special character**.
- Various special characters that can be used in regular expression along with their meaning are shown in following table:

Special Character	Meaning	Special Character	Meaning
.	Any Character Except Newline	*	0 or more
A	The Character a	+	1 or more
ab	The String ab	?	0 or 1
A| B	A or B	{2}	Exactly 2
a*	0 or more A's	{2,5}	Between 2 & 5
\	Escapes a special character	{2 , }	2 or more
[ab-d]	One character of : a, b, c, d	(...)	Group of pattern
[^ab-d]	One character except : a, b, c, d	^	Start of string
[\b]	Backspace character	$	End of string
\d	One digit	\b	Word boundry
\D	One non digit	\n	Newline
\s	One whitespace	\r	Carriage return
\S	One non whitespace	\t	Tab
\w	One word character	\0	Null character
\W	One non word character		

Method	Description
exec	Executes a search for a match in its string parameter.
test	Tests for a match in its string parameter.
match	A string method that executes a search for a match in a string. It return an array of information or null on a mismatch
matchAll	A string method that returns an iterator containing all of the matches, including, capturing groups.
search	A string method that test for a match in a string. It returns the index of the match or -1 if the search fails.
replace	A string method that executes a search for a match in a string, & replaces the matched substring with a replacement substring.
split	A string method that uses a regular expression or a fixed string to break a string into an array of substring.

Finding Non Matching Characters

- We can find the non matching character from the given text by **placing ^ as the first charter with a square[ ].**

```
<html >
<body>
 <script>
   var regex = /^M/;
   var cityes = ["Nashik", "Mumbai", "Pune"];
   for(var city of cityes)
   {
      if(regex.test(city)) {
         document.write("<p>The Name of city: " + city + "</p>")
      }
   }
   </script>
</body>  </html>
```

Output: The Name of city: Mumbai

```
<html >
<body>
   <script>
   var regex = /ca[kf]e/;
   var str = "He was eating cake in the cafe .";
   if(regex.test(str)) {
      document.write("Match found!");
   } else {
      document.write("Match not found.");
   }
   </script>
</body>
 </html>
```

Output: **Match found!**

```
<html >
<body>
 <script>
   var regex =  /fox|dog|cat/;
   var str = "The quick brown fox jumps over the lazy dog.";
   var matches = str.match(regex);
   document.write("Matches the substring: " + matches);
   console.log(matches);
</script>
</body>
 </html>
```

Output: Matches the substring: fox

Entering A Range Of Characters

- For matching any digit we need not have to enter **digit from 0 to 9**. Similarly for matching letters we need not have to test with every single alphabet. We can achieve this by **entering range of characters.**
- **Eg.** To match a digit we must have a **regular expression as [0-9].** By placing the range within a square brackets helps us to evaluate a **complete range of set of characters.**
- Suppose we enter **[k-u] then that match the characters k,l,m,n,o,p,q,r, s, t, and u**

Matching Digits & Non Digits

- Determining whether the string contains **digits or non digits is a common task in any search pattern.**
- In the application of validating telephone numbers this is the most wanted task.
- This can be simplified by using regular expression. If we write **\d then that means search the text for digits** & if we write **\D then that means search the text for non digits.**

Matching Punctuation & Symbols

- The **\w special symbol** tells the browser to **determine whether the text contain a letter, number or underscore.**
- The **\W special symbol** tells the browser to **determine whether the text contain other than a letter, number or underscore.**
- Using **\W is equivalent to using [a-zA-Z0-9_]**. The last _ indicates space charcters.

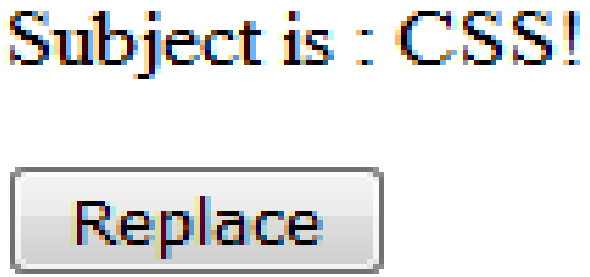
Matching Words

- The words in the text is defined as set of characters. A word is **determined by a word boundary that is the space between two words.**
- The **boundary can be defined** by using **special symbol \b.**
- For example : From the string ‘Cricket’ we can get a match for ‘ket’ by using **regular expression /\bket\b/.**

Replacing A The Text Using Regular Expression

- The **replace**() method we can replace the desired patterns.
- The first parameters in the replace function is the **string which is to be replaced** & second parameters is **replacing string**.

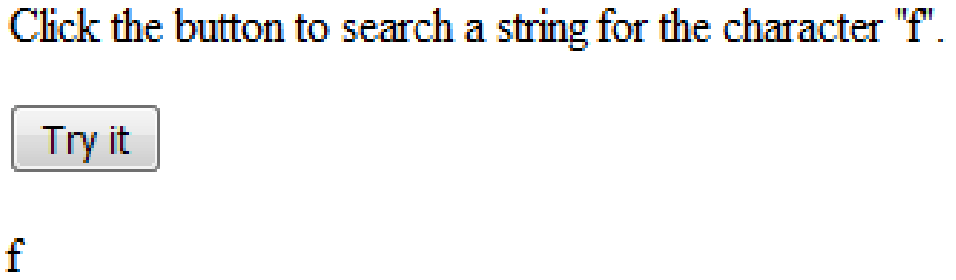
```
<html >
<body>
<p id="demo">Subject is :  CSS!</p>
<button onclick="myFunction()">Replace</button>
<script>
function myFunction()
{
 var str = document.getElementById("demo").innerHTML;
 var res = str.replace("CSS", "AJP");
 document.getElementById("demo").innerHTML = res;
}
</script>
</body> </html>
```



Returning The Matched Characters

- The exec() method searches **string for the text that matches an array of results**, otherwise it returns null.
- If we want search particular pattern from a text then exec() method can be used as follow:
- **pattern.exec(text)**

```
<html>  <body>
<p>Click the button to search a string for the character
"f".</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
  var str = "The best things in life are free";
  var patt = new RegExp("f");
  var res = patt.exec(str);
  document.getElementById("demo").innerHTML = res;
}
</script> </body> </html>
```



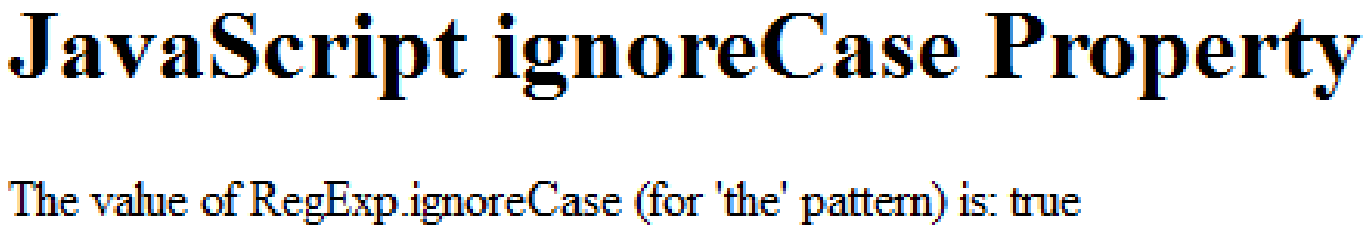
Regular Expression Object Properties.

- There are various regular expression object properties that help in **matching particular word, character, last character, index at which to start the next match & so on.**

Regular Expression Object	Description
$1	Parenthesized substring matches
S_	Same an input
S*	Same as multiline
S&	Same as lastMatch
S+	Same as lastParen
$`	Same as leftContext
S'	Same as rightContext
global	Whether or not to test with the regular expression or not.
ignoreCase	Whether case is to be ignored during pattern matching in a string
input	The string against which a regular expression is matched.
lastIndex	Specifies the index at which to start the next match.
lastMatch	The last matched characters.
multiline	Whether search in strings should be performed across multiple lines.
prototype	Use to add new properties and methods to all instances of a class.
rightContext	The substring following the most recent match.
source	A read-only property that contains the text of the pattern.

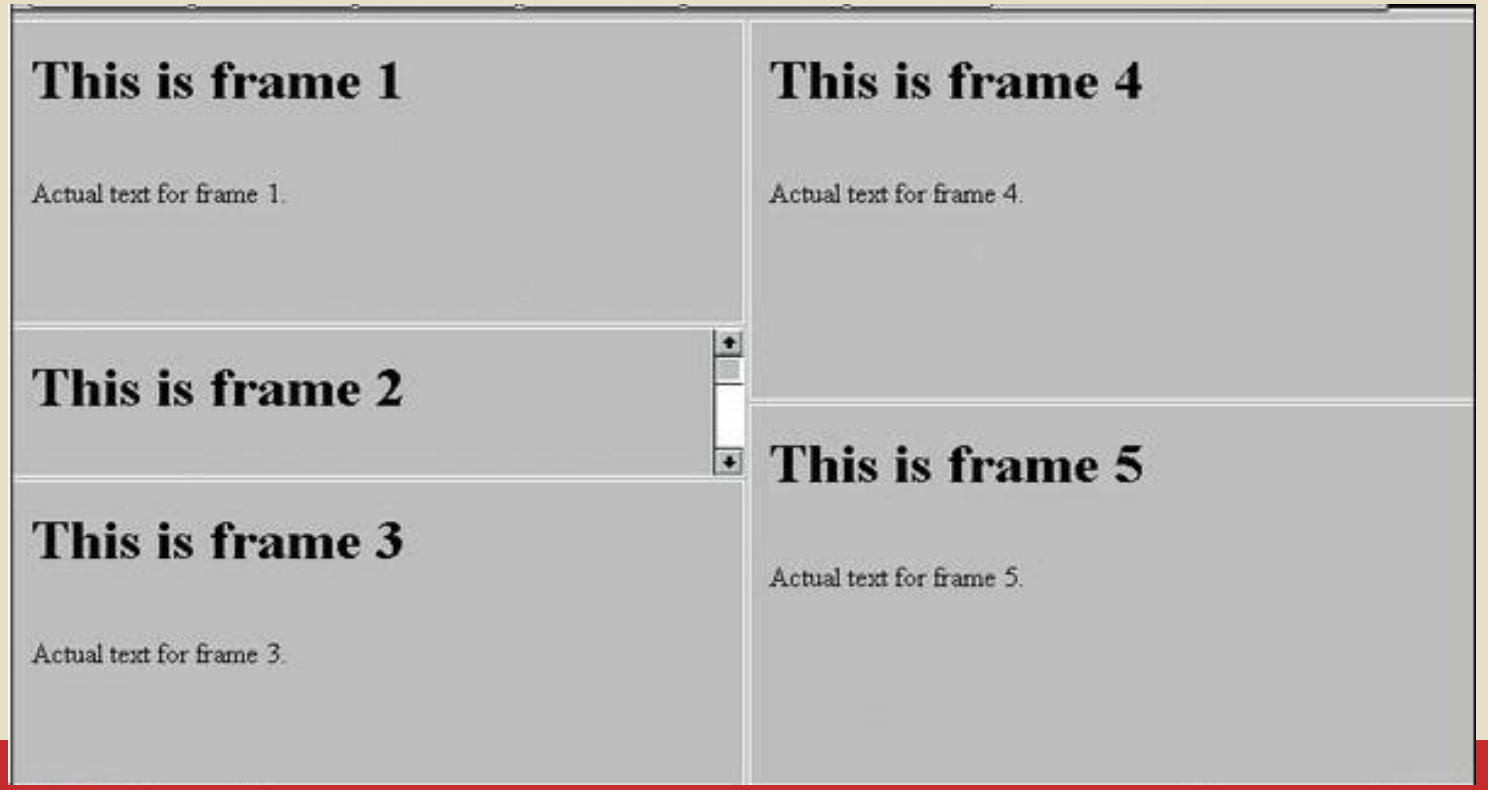
```
<html >
<head>
<body>
<h1 >JavaScript lastIndex constructor Property</h1>
<script type="text/javascript">
rexp=/The*/g;
str1 = "The Quick Brown Fox Jumps Over The Lazy Dog";
array1 = rexp.exec(str1);
document.write("The pattern '"+array1[0] + "' is matched Next
match starts at index : " + rexp.lastIndex);
</script>
</body>
</html>
```

```
<html >
<head>
<body>
<h1 >JavaScript ignoreCase Property</h1>
<script type="text/javascript">
strpat1 = new RegExp("the","i");str1 = "The Quick Brown Fox
Jumps Over The Lazy Dog";
array1 = strpat1.exec(str1);document.write("The value of
RegExp.ignoreCase (for 'the' pattern) is: " + strpat1.ignoreCase);
</script>
</body>
</html>
```



Frames

- HTML frames allow us to present documents in **multiple views.**
- Using multiple views we can keep **certain information visible & at the same time other view are scrolled.**



Create a Frames

- To use frames on a page we use **<frameset> tag instead of <body> tag.**
- The **<frameset> tag defines, how to divide the window into frames.**
- The **rows** attribute of <frameset> tag defines **horizontal frames** and **cols** attribute defines **vertical frames.**
- Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.
- **<frameset rows = "10%,80%,10%">**
- **<frameset cols="150, *">**

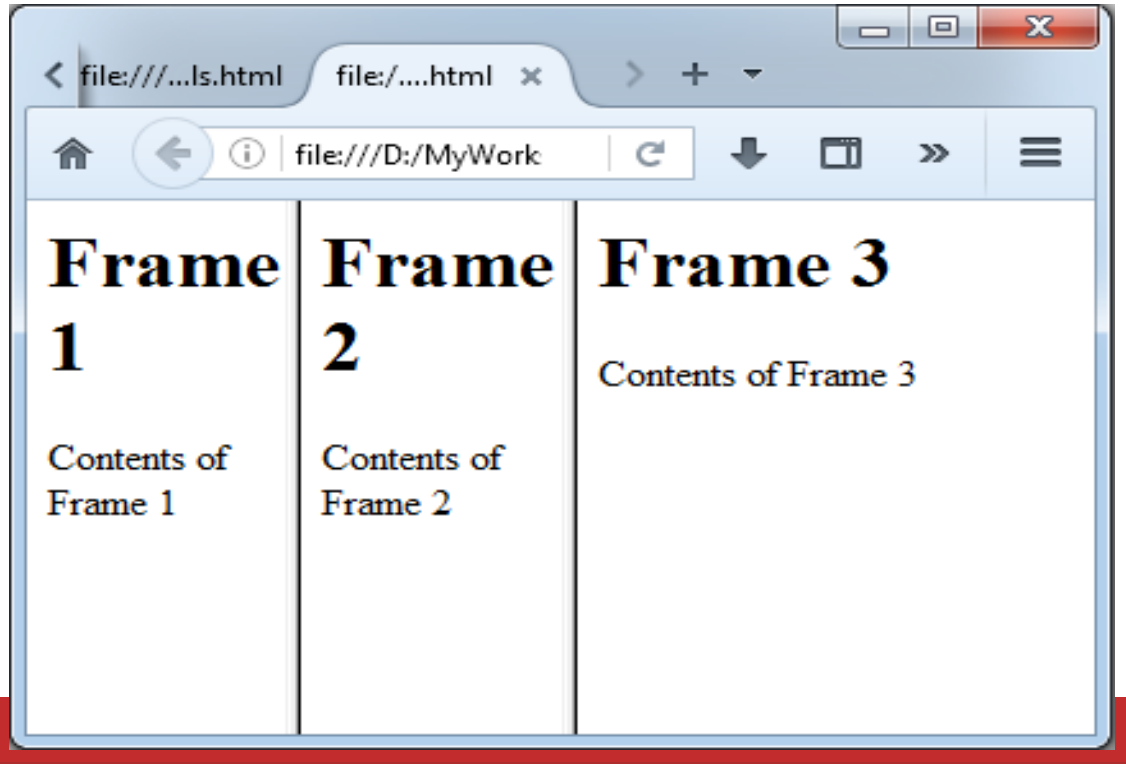
ATTRIBUTES IN FRAMESET TAG

Attribute	Value	Description
cols	pixels % *	Specifies the number and size of columns in a frameset
rows	pixels % *	Specifies the number and size of rows in a frameset

ATTRIBUTES OF FRAME TAG

Attribute	Value	Description
frameborder	0 1	Specifies whether or not to display a border around a frame
name	text	Specifies the name of a frame
noresize	noresize	Specifies that a frame is not resizable
scrolling	yes no auto	Specifies whether or not to display scrollbars in a frame
src	URL	Specifies the URL of the document to show in a frame

```
<html>
<frameset cols="25%,*,50%">
   <frame src="frame1.html">
   <frame src="frame2.html">
   <frame src="frame3.html">
</frameset>
</html>
```



frame1.html

```
<html>
   <body>
      <h1>Frame 1</h1>
      <p>Contents of Frame 1</p>
   </body> </html>
```

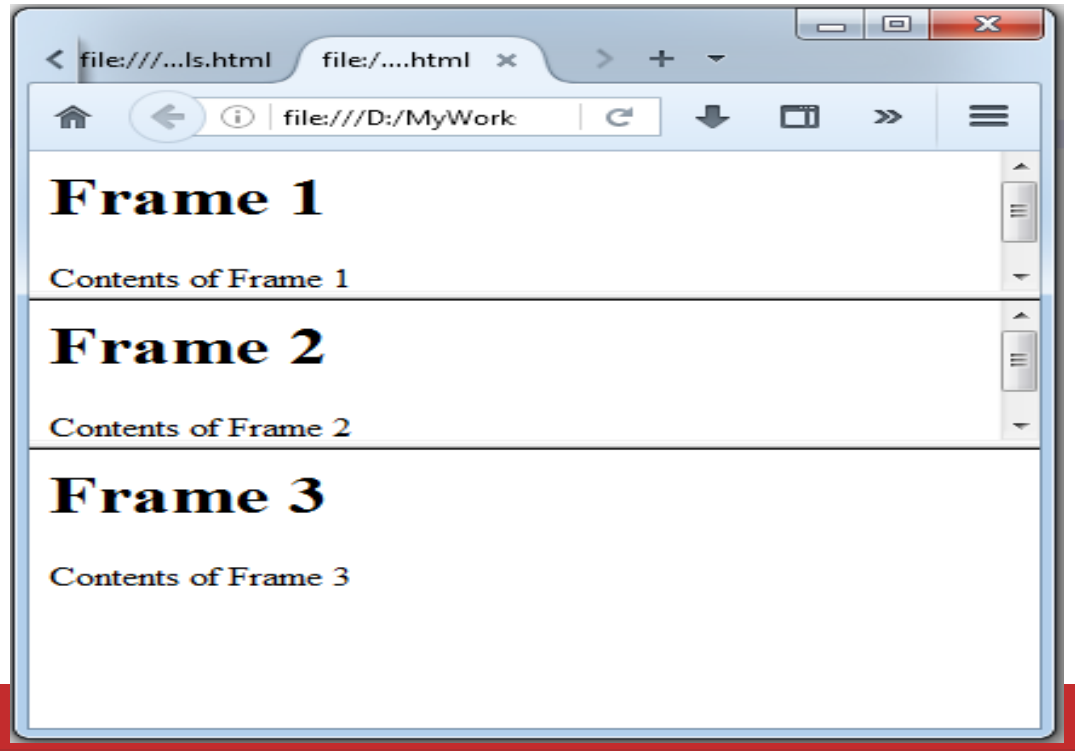
frame2.html

```
<html>
   <body>
      <h1>Frame 2</h1>
      <p>Contents of Frame 2</p>
   </body> </html>
```

frame3.html

```
<html>
   <body>
      <h1>Frame 3</h1>
      <p>Contents of Frame 3</p>
   </body> </html>
```

```
<html>
<frameset rows="25%,*,50%">
   <frame src="frame1.html">
   <frame src="frame2.html">
   <frame src="frame3.html">
</frameset>
</html>
```



frame1.html

```
<html>
   <body>
      <h1>Frame 1</h1>
      <p>Contents of Frame 1</p>
   </body> </html>
```

frame2.html

```
<html>
   <body>
      <h1>Frame 2</h1>
      <p>Contents of Frame 2</p>
   </body> </html>
```

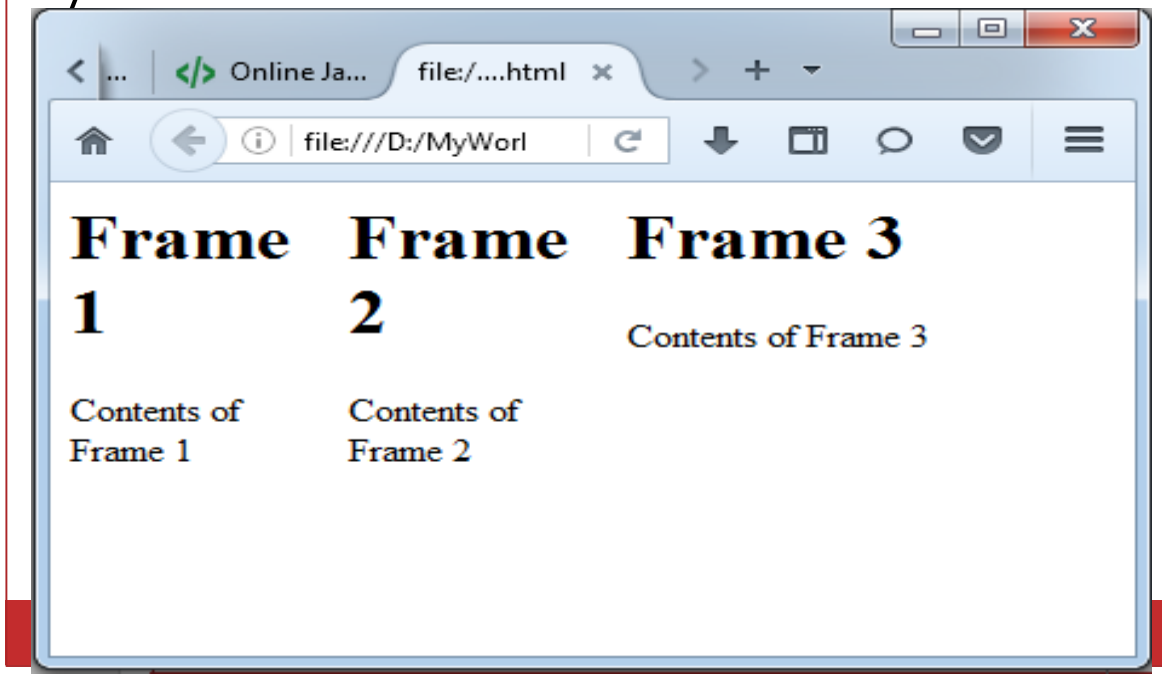
frame3.html

```
<html>
   <body>
      <h1>Frame 3</h1>
      <p>Contents of Frame 3</p>
   </body> </html>
```

Invisible Border Of Frame

- The borders of the frames can be made **invisible by setting the attributes “frameborder” & “border” =0**

```
<html>
<frameset cols="25%,25%,50%">
 <frame src="frame1.html"
frameborder="0" & border="0">
   <frame src="frame2.html"
frameborder="0" & border="0">
   <frame src="frame3.html"
frameborder="0" & border="0">
</frameset>
</html>
```



frame1.html

```
<html>
  <body>
    <h1>Frame 1</h1>
    <p>Contents of Frame 1</p>
  </body> </html>
```

frame2.html

```
<html>
  <body>
    <h1>Frame 2</h1>
    <p>Contents of Frame 2</p>
  </body> </html>
```

frame3.html

```
<html>
  <body>
    <h1>Frame 3</h1>
    <p>Contents of Frame 3</p>
  </body> </html>
```

Calling Child Windows

- The borders of the frames can be made **invisible by setting the attributes “frameborder” & “border” =0**

Test.html

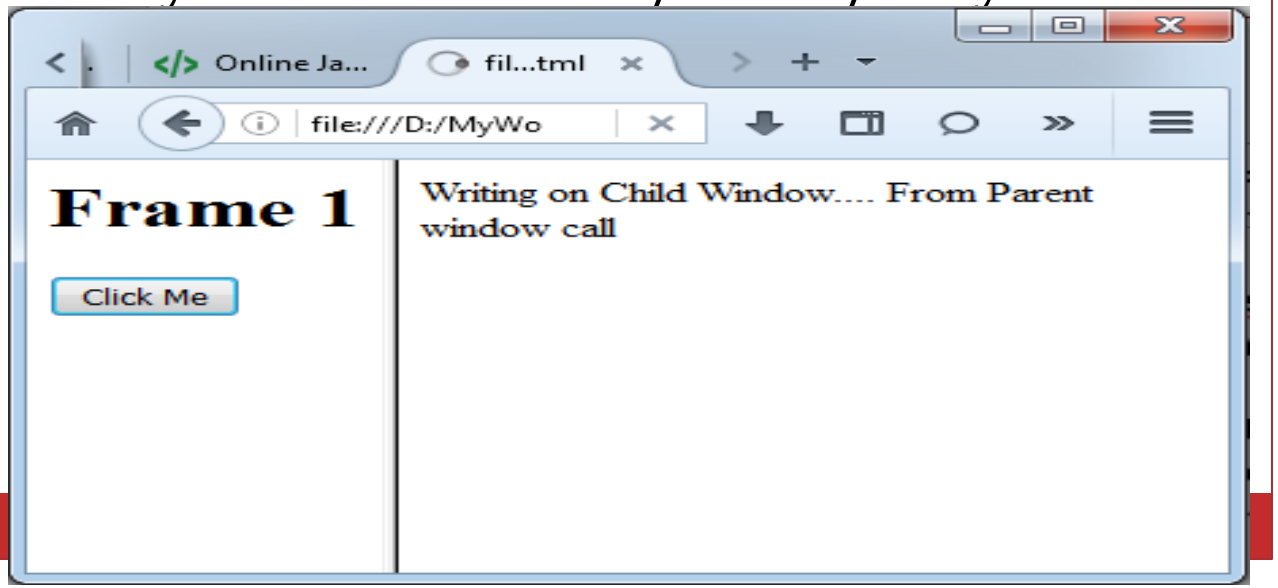
```
<html>
<frameset cols="30%,70%">
   <frame src="f1.html" name=
LeftPage>
   <frame src="f2.html" name=
RightPage>
</frameset>
</html>
```

f1.html

```
<html> <body>
      <h1>Frame 1</h1>
<form>
<input
type="button"name="Frame 1"
Value="Click Me"
onclick="parent.RightPage.MyFu
nction()"/>
</form> </body> </html>
```

f2.html

```
<html>    <head>
      <title>Frame 2</title>
<script>
function MyFunction()
{
document.write("Writing on Child
Window.... From Parent window call")
}
</script> </head>
<body>  <h1> Frame 2</h1>  </body>
```



Changing A Content & Focus Of A Child Window

- We can change the content of a child window from a JavaScript function by modifying the source web page for the child window.

MainWindow.html

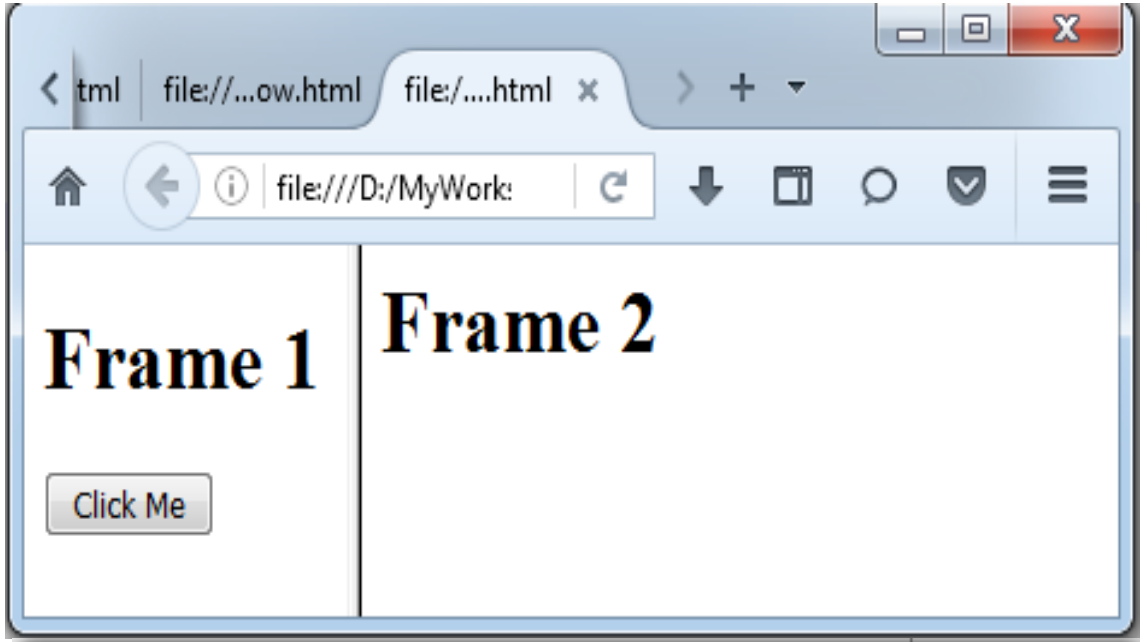
```
<!DOCTYPE html>
<html>
<frameset cols="30%,70%">
   <frame src="fw1.html" name=
"LeftPage"/>
   <frame src="fw2.html" name=
"RightPage"/>
</frameset>
</html>
```

fw1.html

```
<!DOCTYPE html>
<html>
<head>
<script>
function MyFunction()
{
        parent.RightPage.location.href='fw
3.html';
}
</script>  </head>
   <body>
      <h1>Frame 1</h1>
      <form>
<input type="button"name="Frame 1"
Value="Click Me" onclick="MyFunction()"/>
</form>    </body>  </html>
```

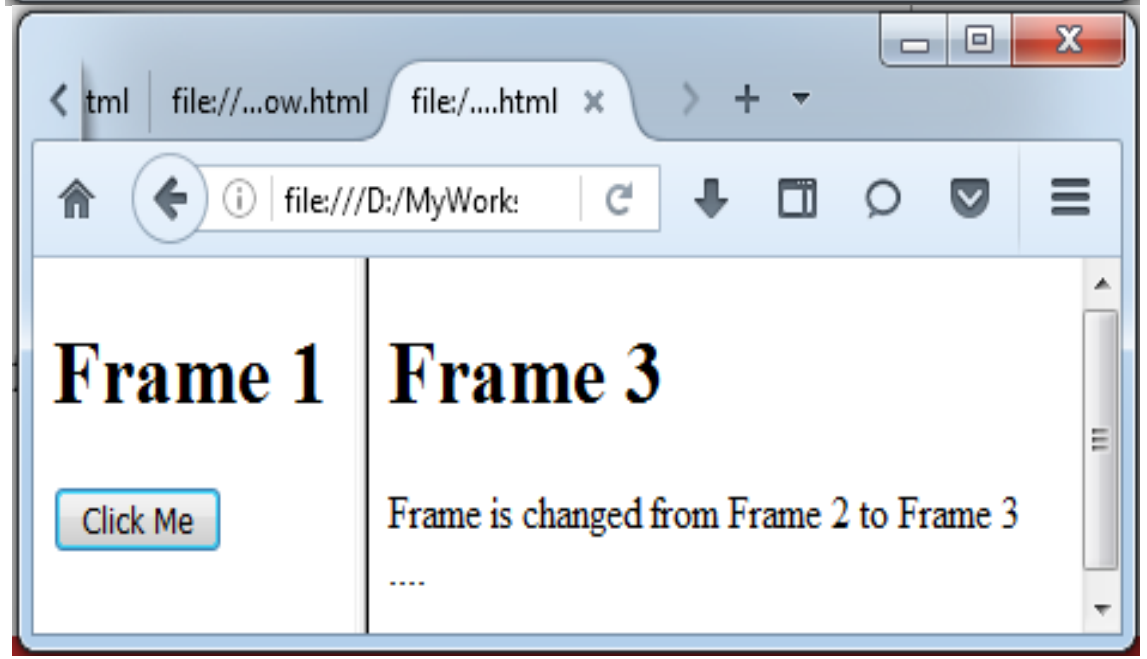
fw2.html

```
<html>
   <head>
      <title>Frame 2</title>
</head>
      <body>
         <h1> Frame 2</h1>
   </body>   </html>
```



fw3.html

```
<html>
<body>
<h1> Frame 3 </h1>
<p>Frame is changed from Frame 2
       to Frame 3 .... </p>
</body>  </html>
```



Accessing Elements Of Child Window

- It is possible to **change the elements of one frame from another frame.**

MainWindowAccessChild.html

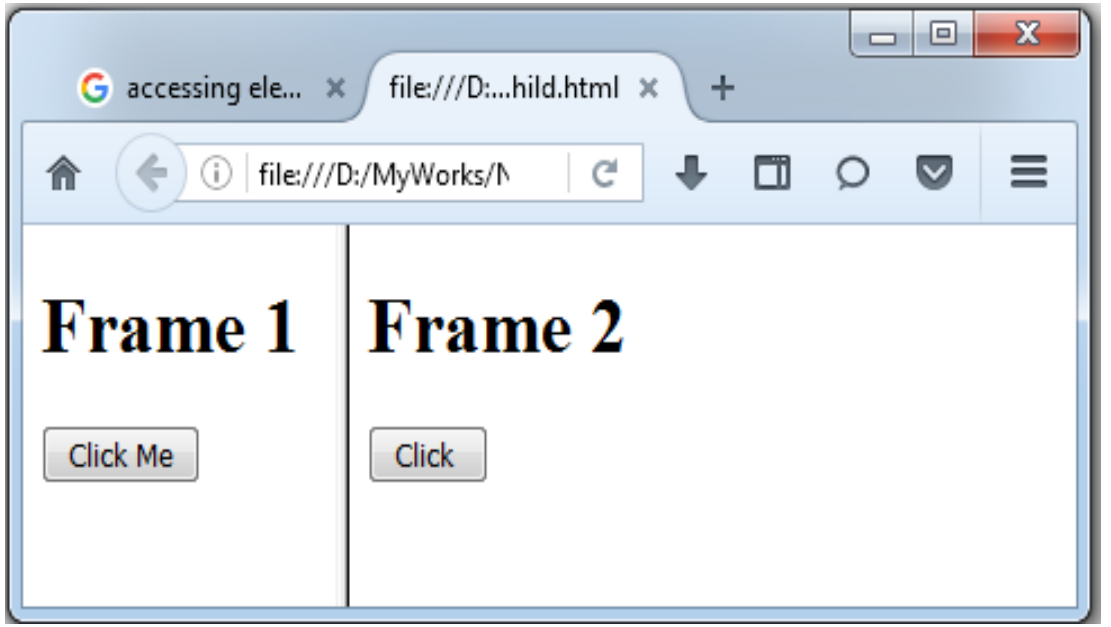
```
<!DOCTYPE html>
<html>
<frameset cols="30%,70%">
   <frame src="Fac1.html"
name= "LeftPage"/>
   <frame src="Fac2.html"
name= "RightPage"/>
</frameset>
</html>
```

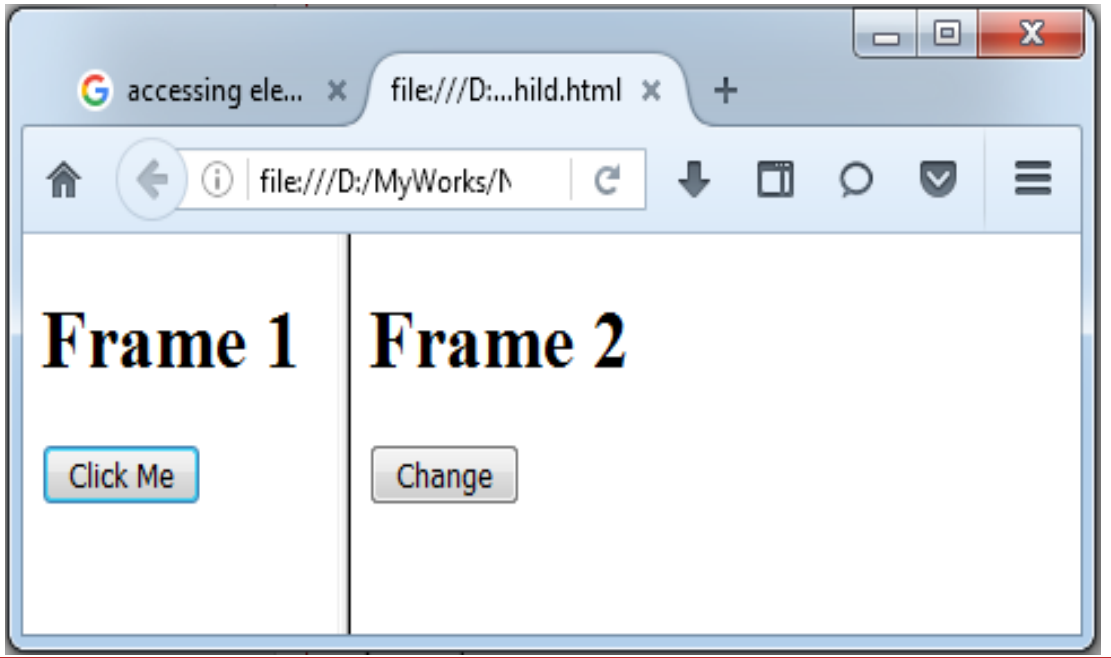
Fac1.html

```
<!DOCTYPE html>
<html><head>
<script language="JavaScript"
type="text/javascript">
function NewFunction()
{
parent.RightPage.form2.Frame2.value="Chan
ge";
}
</script>   </head>
   <body>
      <h1>Frame 1</h1>
      <form name="form1">
<input type="button" name="Frame1"
Value="Click Me" onclick="NewFunction()"/>
</form>    </body>  </html>
```

Fac2.html

```
<html>
<head>
        <title> Frame 2 </title>
</head>
  <body>
    <h1> Frame 2</h1>
    <form name = "form2">
<input type="button" name="Frame2"
Value="Click "/>
</form>
</body>
</html>
```





Rollover

What is rollover?

- Appears when the user place his or her mouse over the text area and the rollover text changes to "**Rollover means a webpage changes when the user moves his or her mouse over an object on the page**" when the user moves his or her mouse away from the text area.

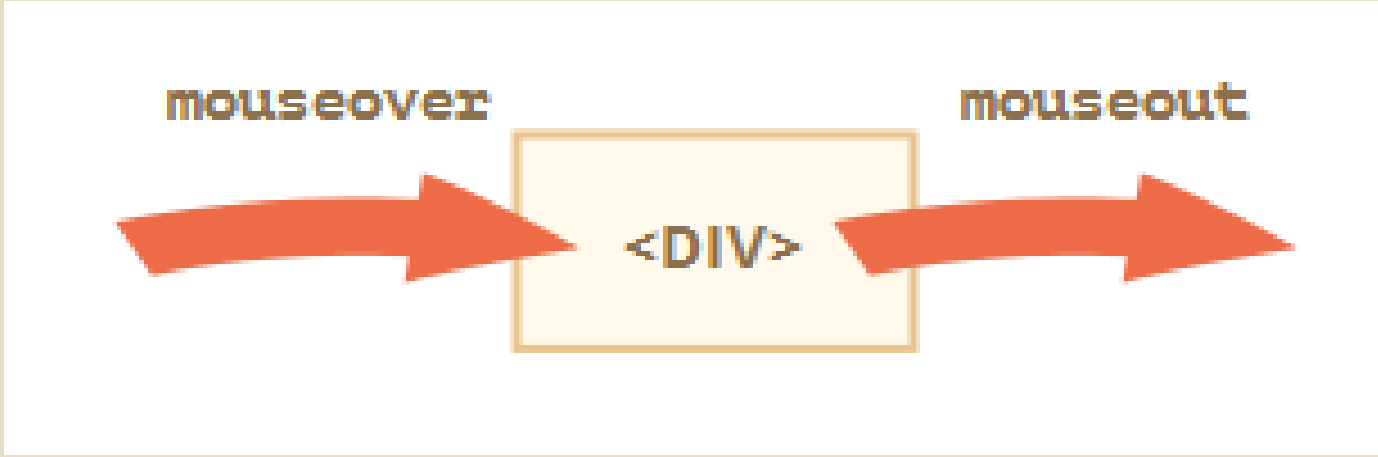
Rollover

- **Rollover** is a **JavaScript** technique used by Web developers to produce an effect in which **the appearance of a graphical image changes when the user rolls the mouse pointer over it.**
- **Rollover** also refers to a button on a Web page that allows **interactivity between the user and the Web page.**
- Rollover effect is mainly used in web page designing for **advertising purpose.**
- Rollover is triggered when the **mouse moves over the primary image, causing the secondary image to appear**. The primary **image reappears when the mouse is moved away.**

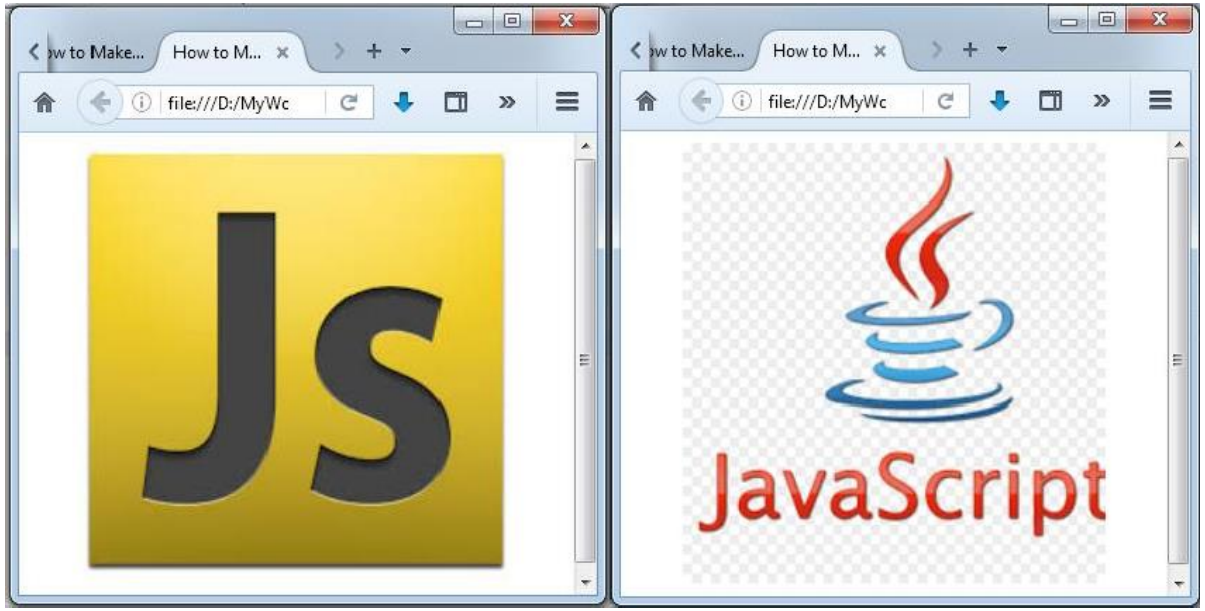
Creating Rollover

- On many web pages JavaScript rollovers are handled by adding an on**mouseover and onmouseout event on image.**

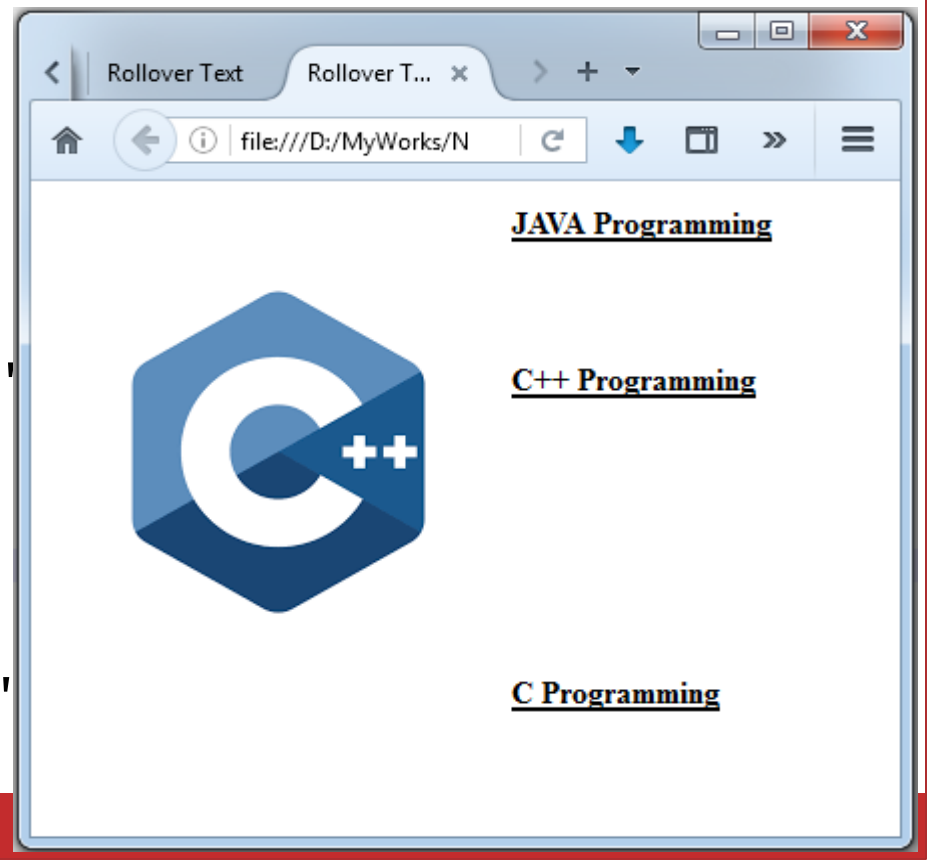
1. **onmouseover :** event occurs when a **mouse pointer comes over an element.**
2. **onmouseout –** event occurs when a **mouse pointer leaves an element.**



```
html>   <head>
<title>How to Make a JavaScript Image Rollover</title>
<script language="javascript">
function MouseRollover(MyImage)
{
        MyImage.src = "image1.jpg";
}
function MouseOut(MyImage)
{
        MyImage.src = "image2.jpg";
}
</script>  </head>
<body>
<div align="center">
<img src="image1.jpg" boarder="0px" width="300px" height="300px"
onMouseOver="MouseRollover(this)"  onMouseOut="MouseOut(this)" />
</div>  </body>  </html>
```



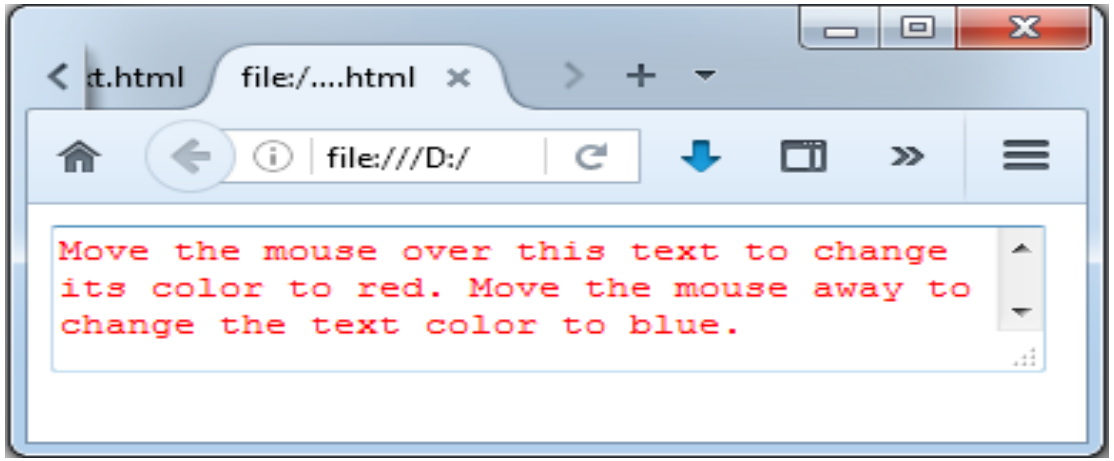
```
<html>
<head>
<title> Rollover Text </title>
</head>
<Body>
<table>
<tr>  <td>
<a>
<IMG src="Java.jpg" name= "lang">
</a> </td>
<td>
<a onmouseover="document.lang.src='Java.jpg' "
<b> <u> JAVA Programming </u> </b>
</a>
<br/><br/><br/><br/>
<a onmouseover="document.lang.src='C++.jpg' "
<b> <u> C++ Programming </u> </b>
</a>
<br/><br/><br/><br/>
<br/><br/><br/><br/>
<a onmouseover="document.lang.src='C.jpg'">
<b> <u> C Programming </u> </b>
</a>
</td> </tr></table> </body> </html>
```

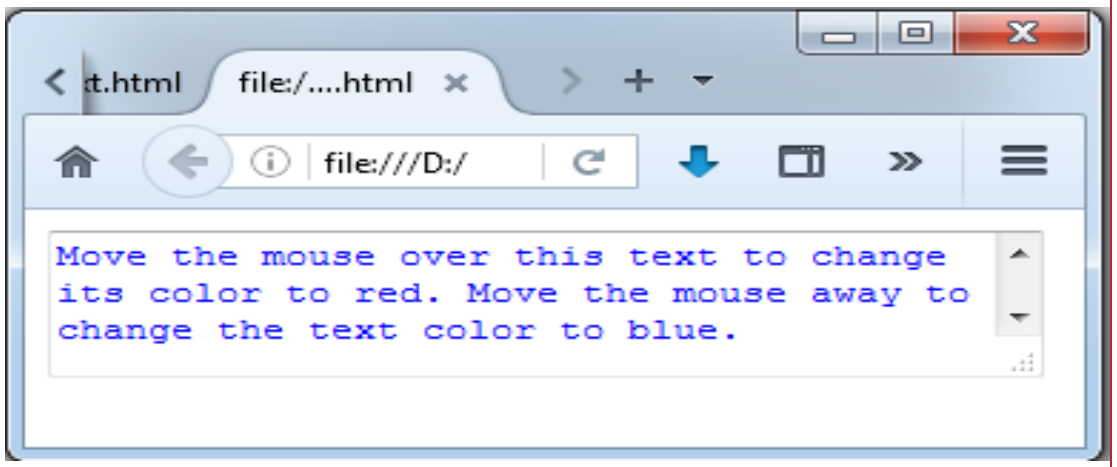


Text Rollover

- Text rollover is a technique in which whenever user rollover the text, JavaScript allows to change the page element usually some graphics.

```
<HTML>
<Body>
<Textarea rows="2" cols="100" onmouseover="this.style.color='red'"
onmouseout="this.style.color='blue'">
Move the mouse over this text to change its color to red. Move the mouse  away
to change the text color to blue.
</textarea>
</body>
</html>
```





Multiple Actions For Rollover

- Suppose user is rolling the cursor over the text, then instead of simply **changing the image we can display more window displaying some features about the item on which mouse is rolling over**. This process is called as **Multiple Actions For Rollover.**
- Duo to this effect visitor gets more information at a glance.
- We can open additional window using function **Open().** This function is invoking using the object window.
- The **open() method open a new browser window. The close() window close the window**.

Multiple Actions For Rollover JavaScript

Multiple Actions For Rollover.docx

More Efficient Rollover

- For more efficient use of rollover, the images can be stored in an **array & required images are displayed when the web page is loaded.**
- This makes the rollover action efficient because the image are **already collected & loaded in an array.** The required image is **displayed when user rollover particular** text.

```
<html> <head>
<title> Efficient Rollover </title>
<script language="Javascript">
   Rollimage= new Array()
   Rollimage[0]= new Image(100,100)
   Rollimage[0].src='Java.jpg'
   Rollimage[1]= new Image(100,100)
   Rollimage[1].src='C++.jpg'
   Rollimage[2]= new Image(100,100)
   Rollimage[2].src='C.jpg'
</script> </head>
<Body>
<table> <tr> <td><a>
<IMG src="Java.jpg" name= "lang">
</a> </td><td>
<a
onmouseover="document.lang.src=Rolli
mage[0].src">
<b> <u> JAVA Programming </u> </b>
</a><br/><br/><br/><br/>
<a
onmouseover="document.lang.src=Rollimage[1].src"
>
<b> <u> C++ Programming </u> </b>
</a><br/><br/><br/><br/>
<a
onmouseover="document.lang.src=Rollimage[2].src"
>
<b> <u> C Programming </u> </b>
</a></td></tr></table></body> </html>
```


Thank you