

# Web Basics - JavaScript

## Lab Book

---

Copyright © 2011 IGATE Corporation (a part of Capgemini Group). All rights reserved.  
No part of this publication shall be reproduced in any way, including but not limited to photocopy, photographic, magnetic, or other record, without the prior written permission of IGATE Corporation (a part of Capgemini Group).

IGATE Corporation (a part of Capgemini Group) considers information included in this document to be confidential and proprietary.

## Document Revision History

---

Date	Revision No.	Author	Summary of Changes
12-May-2009	0.1D	Pradnya Jagtap	Content Creation
05-Oct-2009	0.1D	CLS Team	Review
24-Jun-2010	2.0	Anu Mitra	Refinements
03-May-2011	3.0	Karthik M/Anu Mitra	Integration Refinements
21-Apr-2015	4.0	Rathnajothi P	Revamp/Refinement

## Table of Contents

Document Revision History .....	2
Table of Contents .....	3
Getting Started .....	5
Overview .....	5
Setup Checklist for JavaScript .....	5
Instructions .....	5
Learning More (Bibliography if applicable) .....	5
Lab 1. Basics Concepts of JavaScript.....	6
1.1: Create a page to display “Welcome to JavaScript”. .....	6
1.2: Create prob2.html to display Formatted Hello World by using JavaScript by embedding Hello World in <H1> tag. ....	7
1.3: Create page to show use of external JavaScript.....	8
1.4: Using Variable in many Script tags .....	10
Lab 2. The JavaScript Language.....	13
2.1: For loop in JavaScript.....	13
2.2: Create a web page to calculate the Compound Interest using the formula given below: .....	14
Lab 3. Working with Predefined core objects.....	15
3.1: Displaying Date using Date Object .....	15
3.2: Using indexOf function of String object .....	15
3.3: Using various String methods .....	16
Lab 4. Working with Arrays .....	17
4.1: Using Array to display values.....	17
Lab 5. Working with Document Object Model(DOM).....	18
5.1: Window object.....	18
Lab 6. Working with Location & History Objects.....	20
6.1: Location Object.....	20
Lab 7. Working with Document and Cookie Object.....	21
7.1: Working with Documents.....	21
Lab 8. Working with Form Object.....	23
8.1: Form Validation .....	23
8.2 Validate Field.....	25
8.3 Validate Field.....	26

---

8.3: Registration Form .....	27
Lab 9. Regular Expressions in JavaScript .....	30
9.1: Regular Expression .....	30
9.2: Form Validation using Regular Expression.....	31
Appendices.....	33
Appendix A: JavaScript Standards.....	33
Appendix B: Coding Best Practices .....	38
JavaScript Best Practice.....	38
Appendix C: Table of Figures .....	43
Appendix D: Table of Examples .....	44

## Getting Started

---

### Overview

These Lab book is a guided tour for Learning JavaScript. It contains solved examples and To Do assignments. Follow the steps provided in the solved examples and then work out the 'to do' Assignments given.

### Setup Checklist for JavaScript

Here is what is expected on your machine in order for the lab to work.

#### Minimum System Requirements

- Hardware: Networked PCs with minimum 64 MB RAM and 60 MB HDD.
- Software: Window based Operating System having the latest version of Internet Explorer (IE) or Netscape Navigator installed.

#### Please ensure that the following is done:

- A text editor like Notepad, Eclipse Luna or Visual Studio 2008 is installed.

### Instructions

- For coding standards refer Appendix – A.
- All Lab assignments should follow the coding standards.
- Create a directory by your name in drive <drive> for JavaScript assignments.
- In this directory, create subdirectory javascript\_assgn.
- For each lab create directory as lab<lab number>.

### Learning More (Bibliography if applicable)

- Beginning JavaScript by Paul Wilton
- JavaScript: The Definitive Guide by David Flanagan
- JavaScript Application Cookbook by Jerry Bradenbaugh

## Lab 1. Basics Concepts of JavaScript

<b>Goals</b>	<ul style="list-style-type: none"><li>• Learn to embed script tags in different parts of the HTML document.</li></ul>
<b>Time</b>	120 minutes

### 1.1: Create a page to display “Welcome to JavaScript”.

#### Solution:

**Step 1:** Complete the following code and save it as **prob1.html**

```
<!DOCTYPE html>
<html>
<head>
<title> Welcome to JavaScript</title>
</head>
<body>
<script type="text/javascript">
  document.write("Welcome to JavaScript - The Scripting Language")
</script>
</body>
</html>
```

Example 1: Lab 1: Prob1.html

**Step 2:** Start the editor to be used.

**Step 3:** Write the JavaScript program.

**Step 4:** Save the file with extension .html or htm.

**Step 5:** Select **Start → Programs → Internet Explorer**.

Alternatively select **Start → Programs → Netscape Navigator**.

**Step 6:** In the Internet Explorer, select **File → Open → Browse**, and select the file you have just saved.

**Step 7:** Click **OK** in the browser pop-up window.

**Step 8:** Verify that you get the output as shown in the figure given below.

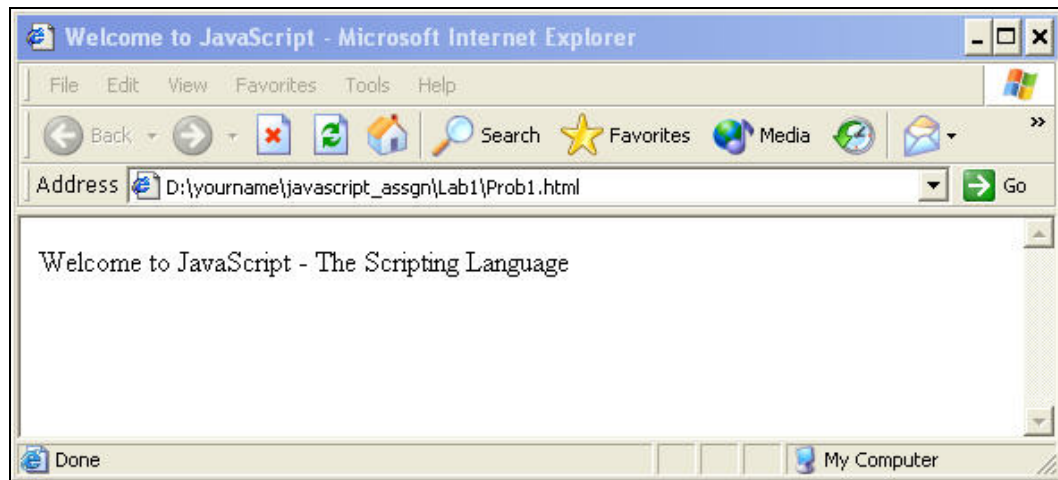


Figure 1: Welcome to JavaScript

**Note:** Follow the above steps (3 - 8) for every Lab problem for verifying the output. You can also use other text editors like editplus, WordPad, MS Visual Interdev (if installed) to create your **html** and **.js** pages.

## 1.2: Create prob2.html to display Formatted Hello World by using JavaScript by embedding Hello World in <H1> tag.

### Solution:

**Step 1:** Create **prob2.html** page to complete the following code and save in lab1 directory.

```
<html>
<head>
<title>Displaying Formatted Text using JavaScript</title>
</head>
<body>

<script type="text/javascript">

//TODO: Display hello world embedded in h1 tag with align attribute value right

</script>

</body>
</html>
```

Example 2: Lab 1: Prob2.html

**Step 2:** Open **prob2.html** page in the browser, and verify that you get the same output as required.



# Hello World!

Figure 2: Formatting Text in JavaScript

### 1.3: Create page to show use of external JavaScript

#### Solution:

**Step 1:** Create **Prob3.html** to complete the following code and save it in lab1 directory.

```
<html>
<head><title>Using External Script file in HTML Document</title>

<script src="HelloWorld.js">
</script>

</head>
<body>
<hr>
<p>The actual script is in external script file called "HelloWorld.js"</p>

<script>
//TODO: Insert the code here to invoke the function sayHello() in the file HelloWorld.js
</script>

<hr>
</body>
</html>
```

Example 3: Lab 1: Prob3.html

**Step 2:** Create a file **HelloWorld.js** which should have a function **sayhello()** that returns a string "Hello World".

```
function sayHello()
{
//TODO:return the string "Hello World"
}
```

Example 4: Lab 1: HelloWorld.js



**Step 3:** Open **prob3.html** page in the browser, and verify that you get the same output as required.

The actual script is in external script file called "HelloWorld.js"

This text is displayed by Calling external function : **Hello World**

**Figure 3: Using external JavaScript File**

**Step 4:** Create **Prob4.html** page and complete the following code and save it in lab1 directory.

```
<html>
<head><title>Embedding Script tag in HTML Document</title>

<script>

//TODO:use write method in document object to display the desired output

</script>
<hr>
<script src="Hello.js">
</script>

</head>
<body>

<script>
//TODO: use write method in document object to display desired the output
</script>

<hr>

<p><code>The actual script is in external script file called "Hello.js"</code></p>

<script>
//TODO: Insert your code here to call the function Disp_Hello() from the Hello.js file
</script>

<hr>

</body>
</html>
```

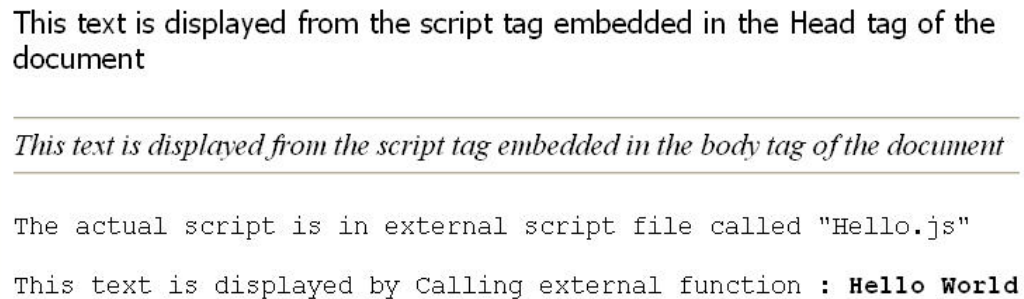
**Figure 4: Lab 1: Prob4.html**

**Step 5:** Create a file **Hello.js** which should have a function **Disp\_Hello()** that returns a string "Hello World".

```
function Disp_Hello()
{
//TODO:return the string "Hello World"
}
```

Example 5: Lab 1: Hello.js

**Step 6:** Open **prob4.html** page in the browser, and verify that you get the same output as required.



This text is displayed from the script tag embedded in the Head tag of the document

---

*This text is displayed from the script tag embedded in the body tag of the document*

---

The actual script is in external script file called "Hello.js"

---

This text is displayed by Calling external function : **Hello World**

Figure 5: Embedding Script tags in HTML document

#### 1.4: Using Variable in many Script tags

##### Solution:

**Step 1:** Create **Prob5.html** page, and complete the following code and save it in lab1 directory.

```
<html>
<head><title>Embedding Script tag in HTML Document</title>

<script>
/*
TODO:define variable headVar and initialize it to some integer value and display the value as
shown in the Fig 6
*/
</script>

<hr>
</head>
<body>

<script>
/*
```

TODO:define variable bodyVar and initialize it to some integer value and display the value as shown in the Fig 6

```
*/
```

```
</script>
```

```
<hr>
```

```
<script src="common.js">
```

```
</script>
```

```
<script>
```

```
/*
```

TODO: Invoke the method add\_nos(headVar,bodyVar) defined in common.js file and pass the two variables headVar and bodyVar defined in the head and the body script tag and display the added result as shown in the Fig 6

```
*/
```

```
</script>
```

```
<hr>
```

```
</body>
```

```
</html>
```

Example 6: Lab 1: Prob5.html

**Step 2:** Create a file **common.js** which has a function **add\_nos()** that adds two numbers and returns the addition of two numbers.

```
var msg;
msg="<p><code>The actual script is in external script file called common.js</code></p>";

function add_nos(headVar,bodyVar)
{

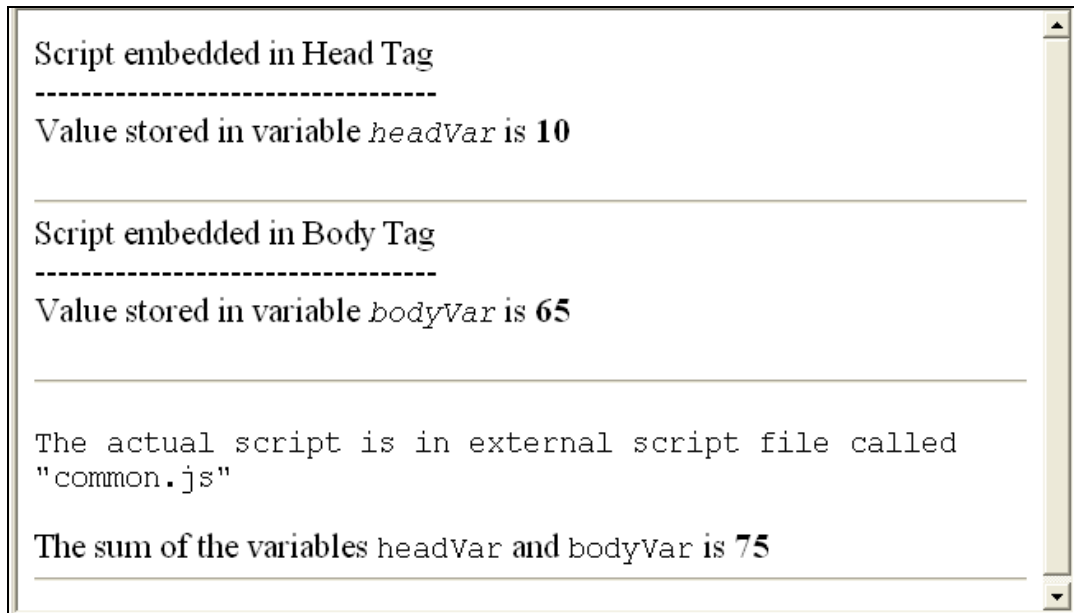
//TODO: display the contents of the variable "msg"

//TODO: display the addition of two numbers

}
```

Example 7: Lab 1: common.js

**Step 3:** Open **prob5.html** page in the browser, and verify that you get the same output as required.



**Figure 6:** Using Variable in many Script tags

## Lab 2. The JavaScript Language

<b>Goals</b>	<ul style="list-style-type: none"> <li>Learn to use looping structures and operators in JavaScript.</li> </ul>
<b>Time</b>	20 minutes

### 2.1: For loop in JavaScript

Create a web page containing a heading “Layout is here” followed by a horizontal rule and a table with a single row as shown in the figure given below.

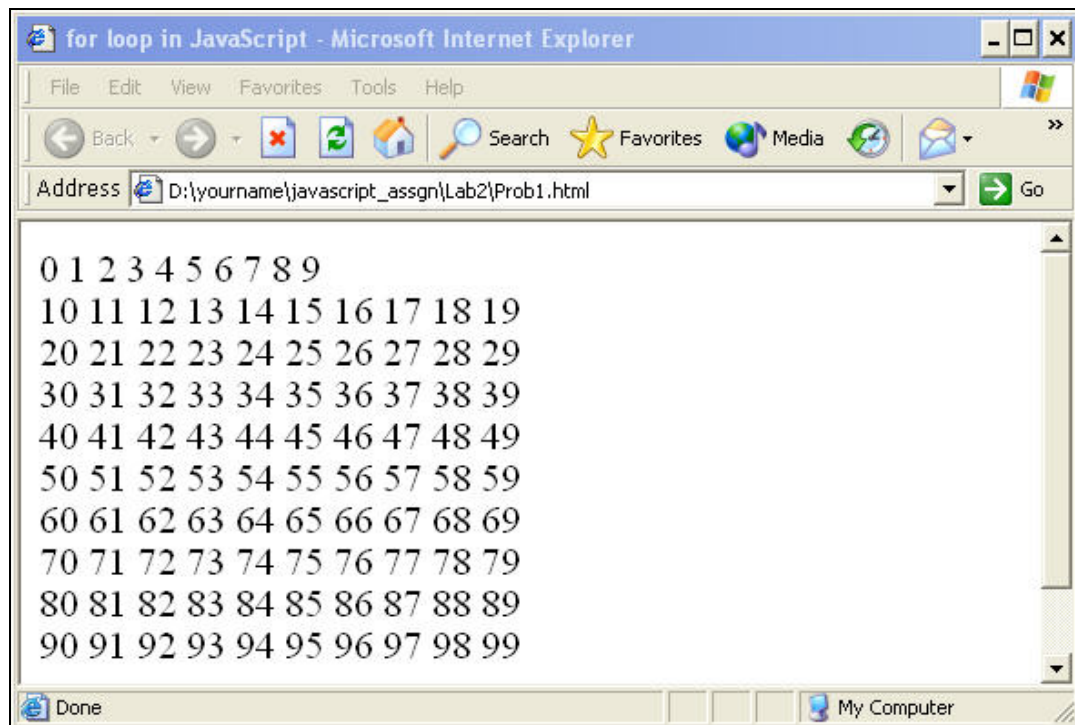


Figure 7: For loop in JavaScript

After completing the loop, the variable used, that is “*i*”, should be equal to **100**.

#### Solution:

**Step 1:** Write the code and save it as **Prob1.html** in lab2 directory.

**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

**Step 3:** Create **prob1\_dowhile.html** and **prob1\_whiledo.html** page using **do...while** and **while...do** control statements respectively to display similar output as shown in the figure given above.

**2.2: Create a web page to calculate the Compound Interest using the formula given below:**

$$\text{Compound Interest} = \left[ P * \left( 1 + \frac{r}{100} \right)^n \right] - P$$

Where:

p = Principal,

r = Rate of Interest,

n = period in years

The values used in the example in the following figure are as follows:

P = 1000, n = 1, r = 10

```

-----
*****Calculate Compound Interest*****
-----

Prinicipal          -      1000 rs
Rate of Interest    -      10%
Period              -      1 yr
Comp Interest       -      100
  
```

**Figure 8: Operators and Arithmetic Expression**

**Solution:**

**Step 1:** Write the code, and save it in lab2 directory.

**Step 2:** Open page in the browser, and verify that you get the same output as required.

## Lab 3. Working with Predefined core objects

<b>Goals</b>	<ul style="list-style-type: none"> <li>Understand date, String Object</li> <li>Learn to use Date and String objects in html pages</li> </ul>
<b>Time</b>	45 minutes

### 3.1: Displaying Date using Date Object

Create a web page **Prob1.html**. In this web page, create a **date** object and use the **getXXXX** functions of the date object to display today's date in the format as shown below in the figure and also greet the user depending on the time the user visits the page. The message to be displayed is given in the following table. The time column shows the current date hour value.

Time	Msg to be displayed
< 12	Good Morning
>= 12 and <= 17	Good Afternoon
> 17	Good Evening



Figure 9: Displaying Date using Date Object

#### Solution:

**Step 1:** Write the Code, and save it as **Prob1.html** in lab4 directory.

**Step 2:** Open prob1.html page in the browser, and verify that you get the same output as required.

### 3.2: Using indexOf function of String object

Create a web page **prob2.html**, which uses the **indexOf** method of string object and displays the index number of the substring searched for within the string.

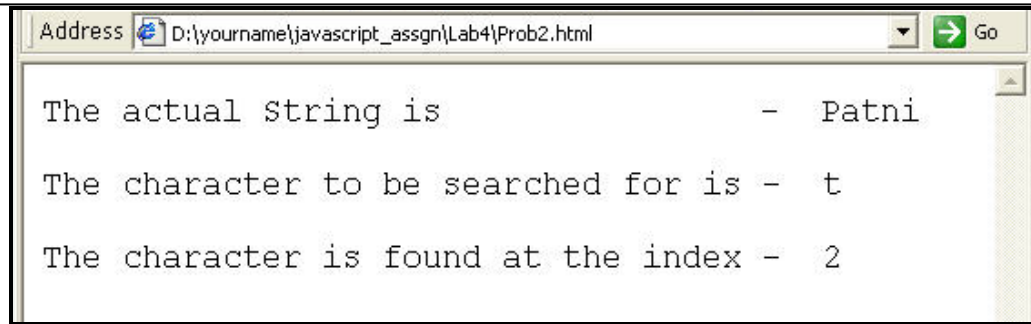


Figure 10: Using indexOf function of String object

**Solution:**

**Step 1:** Write the Code and save it as **Prob2.html**.

**Step 2:** Open **prob2.html** page in the browser, and verify that you get the same output as required.

### 3.3: Using various String methods

Write **prob3.html** page by completing the following code that demonstrates some of the methods of the String objects like **match**, **substr**, **LowerCase**, and **UpperCase** to produce the output as shown in the figure given below:

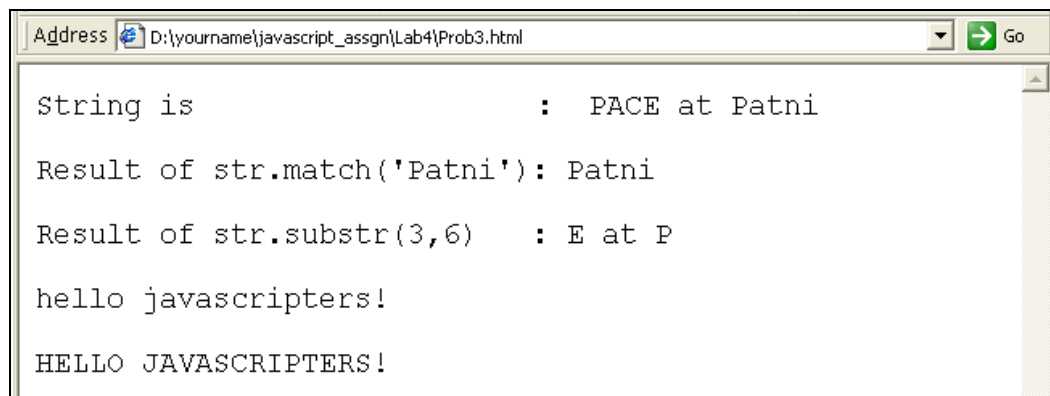


Figure 11: Using various String methods

**Solution:**

**Step 1:** Write the Code and save it as **Prob3.html**.

**Step 2:** Open **prob3.html** page in the browser, and verify that you get the same output as required.



## Lab 4. Working with Arrays

Goals	<ul style="list-style-type: none"><li>• Work with Array Object</li></ul>
Time	10 minutes

### 4.1: Using Array to display values

Create a **prob1.html** web page containing script. In this script, declare an array of 6 employee names and display it in the browser as shown below:



Figure 12: Using Array to display values

#### Solution:

**Step 1:** Write the Code, and save it as **Prob1.html**.

**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

## Lab 5. Working with Document Object Model(DOM)

<b>Goals</b>	<ul style="list-style-type: none"> <li>Understand Window, Frame Object.</li> <li>Dynamically create windows and frames.</li> <li>Handle window and frame objects events.</li> </ul>
<b>Time</b>	90 minutes

### 5.1: Window object

Create a **prob1.html** web page which has the following items as shown in the figure given below:

- a form that accepts window parameters width, height, title, left and top parameters from text field, and
- two buttons with the labels **New Window** and **Reset** to the web page

The screenshot shows a web browser window with the address bar displaying 'D:\yourname\javascript\_assgn\Lab6\Prob1.html'. The main content area contains a form titled 'Create a new window'. The form is organized into sections: 'Window Size' with 'Width:' and 'Height:' labels and corresponding text input fields; 'POSITION' with 'Left:' and 'Top:' labels and corresponding text input fields. At the bottom of the form are two buttons: 'New Window' and 'Reset'. The browser's status bar at the bottom shows 'Done' and 'My Computer'.

Figure 13: Interface to accept window coordinates

If **Reset** button is clicked, then clear all text fields. If **New Window** button is clicked, then open a new window with specifications entered in the text fields as shown in the figure given below.

**Note:** By default, the new window opens at the top left corner of the screen.

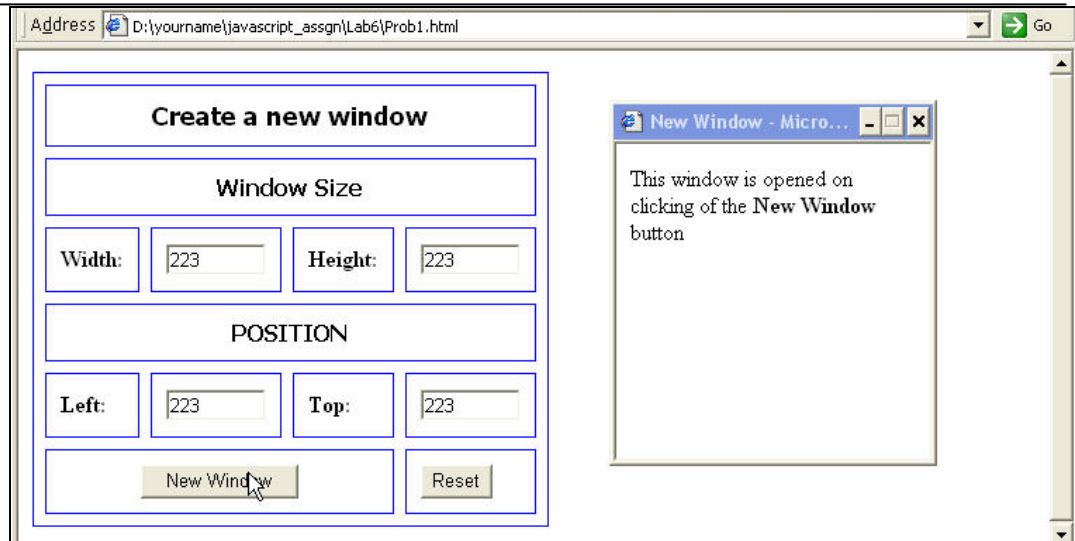


Figure 14: Opening a window

**Solution:**

**Step 1:** Complete the following Code and save it as **Prob1.html**.

```
<html>
<head>
<title> window example </title>
</head>
<script language="javascript" >
function nwindow()
{
/*TODO: get the height, width, left and top from the form object and pass the values to
open method of window along with the name of the html file to be opened in the new
window.*/
}
</script>
<body >
<form id="frmlab">
<table border="1" cellspacing="8" cellpadding="10" bordercolor="blue">

// Create Table as shown in fig 6.2
</table>
</form>
</body>
</html>
```

Example 8: Lab 6: Prob1.html

**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

**Step 3:** Open **prob2.html** page in the browser, and verify that you get the same output as required.

## Lab 6. Working with Location & History Objects

Goals	<ul style="list-style-type: none"><li>• Understand and use Location Object.</li><li>• Understand and use History Object.</li></ul>
Time	10 minutes

### 6.1: Location Object

Create a web page which will display the properties **href**, **protocol**, and the **pathname** of the location object of your current file.

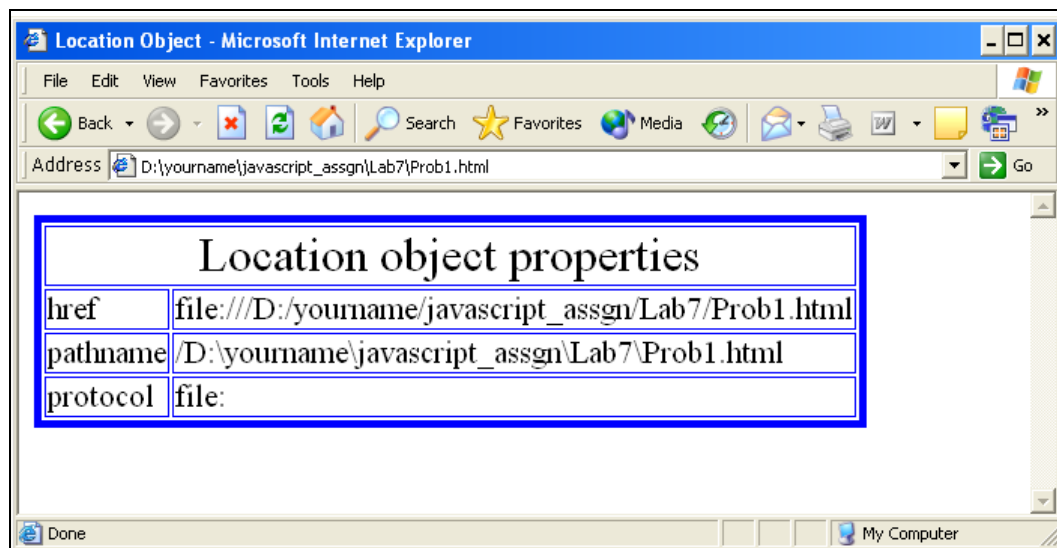


Figure 15: Location Object Properties

#### Solution:

**Step 1:** Write the code and save it as **Prob1.html**.

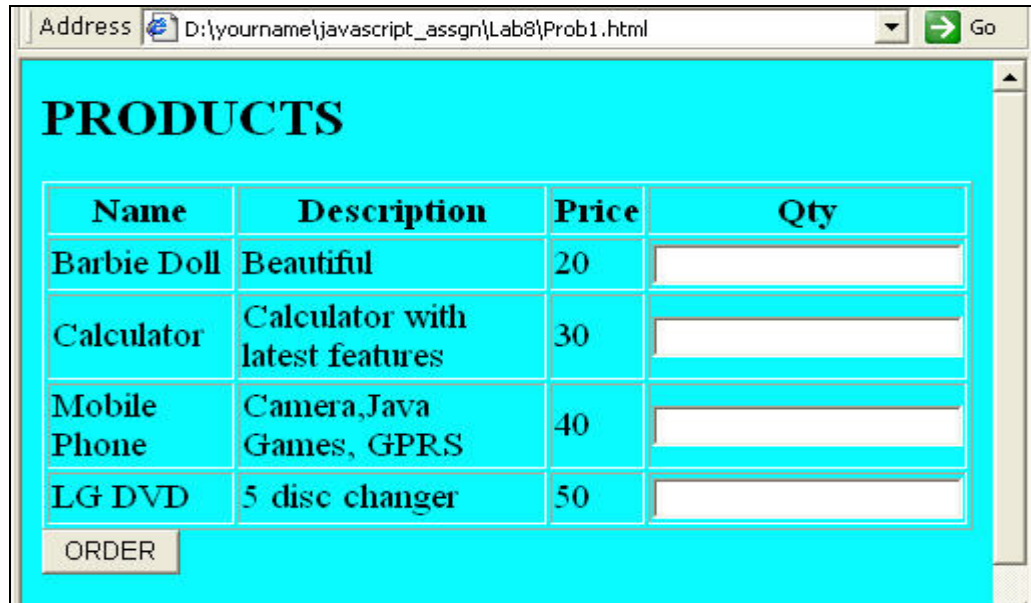
**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

## Lab 7. Working with Document and Cookie Object

Goals	<ul style="list-style-type: none"><li>Understand Document and Cookie Object</li></ul>
Time	60 minutes

### 7.1: Working with Documents

Create a **prob1.html** web page which displays products available as shown in the following figure. The product details comprise Product Name, Product description, and its price.



Name	Description	Price	Qty
Barbie Doll	Beautiful	20	<input type="text"/>
Calculator	Calculator with latest features	30	<input type="text"/>
Mobile Phone	Camera, Java Games, GPRS	40	<input type="text"/>
LG DVD	5 disc changer	50	<input type="text"/>

ORDER

Figure 16: Displaying Products

Users can place orders specifying the quantity of each product. If the user does not enter quantity in any of the text fields, then an error message should be displayed as shown in the figure given below:

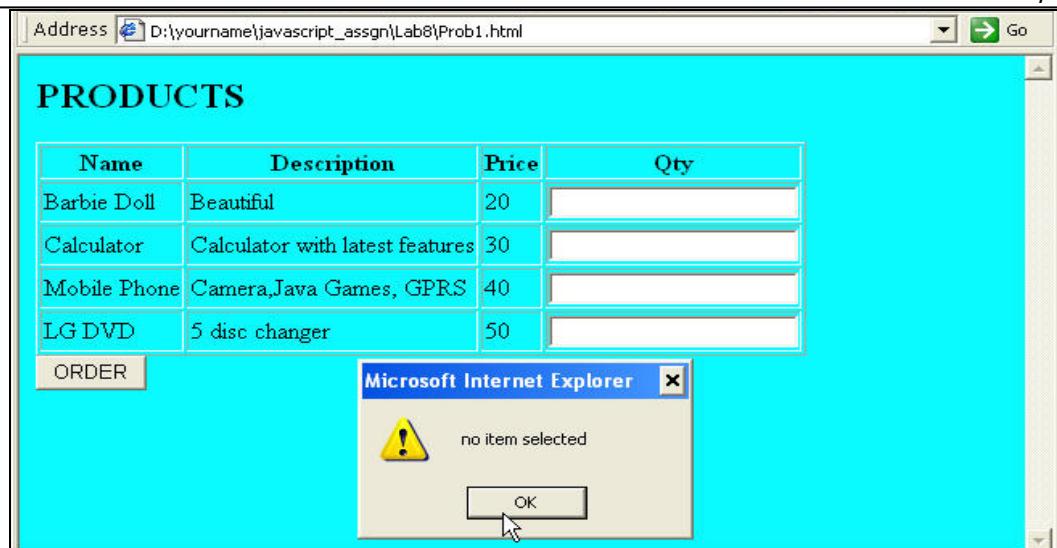


Figure 17: Validating Products

When the user clicks the **Order** button, the invoice for the current products transaction showing the product name, quantity ordered, price and total amount is displayed in a new window as shown in the figure given below:

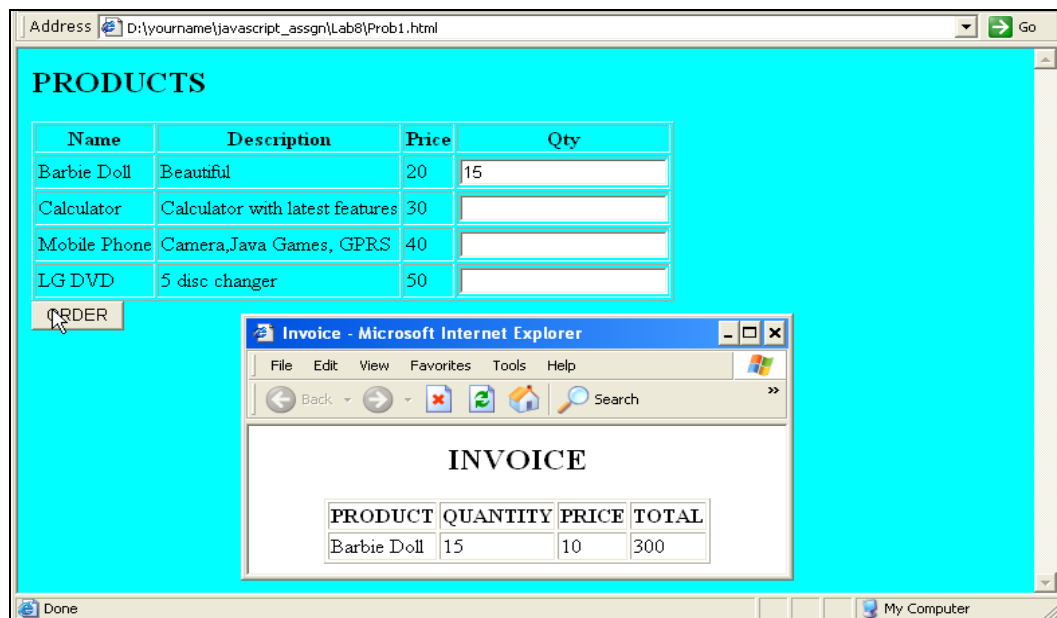


Figure 18: Displaying Invoice details in a new window

#### Solution:

**Step 1:** Write the code and save it as **Prob1.html**.

**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

## Lab 8. Working with Form Object

<b>Goals</b>	<ul style="list-style-type: none"> <li>Understand and use Form Object.</li> </ul>
<b>Time</b>	90 minutes

### 8.1: Form Validation

Create a **prob1.html** web page, as shown below, and calculate **Payment Information** based on **Loan Information**. Validate **Loan information** textfields for numbers. The **Payment Information** textfields should be uneditable. The other constraints are as follows:

- Amount of Loan should not be more than 15 lakhs.
- Repayment period should be between 7 yrs to 15 yrs.

The screenshot shows a Microsoft Internet Explorer window with the title 'Computing Payment and Loan Information - Microsoft Internet Explorer'. The address bar shows 'D:\yourname\javascript\_assgn\Lab9\Prob1.html'. The form contains two sections:

**Enter Loan Information:**

- 1) Amount of the loan (any currency):
- 2) Annual percentage rate of interest:
- 3) Repayment period in years:

Below these fields is a 'Compute' button.

**Payment Information:**

- 4) Your monthly payment will be:
- 5) Your total payment will be:
- 6) Your total interest payments will be:

The status bar at the bottom shows 'Done' and 'My Computer'.

Figure 19: Validating Form elements

If the repayment period is not between 7 and 15, then an error message should be displayed as shown below:

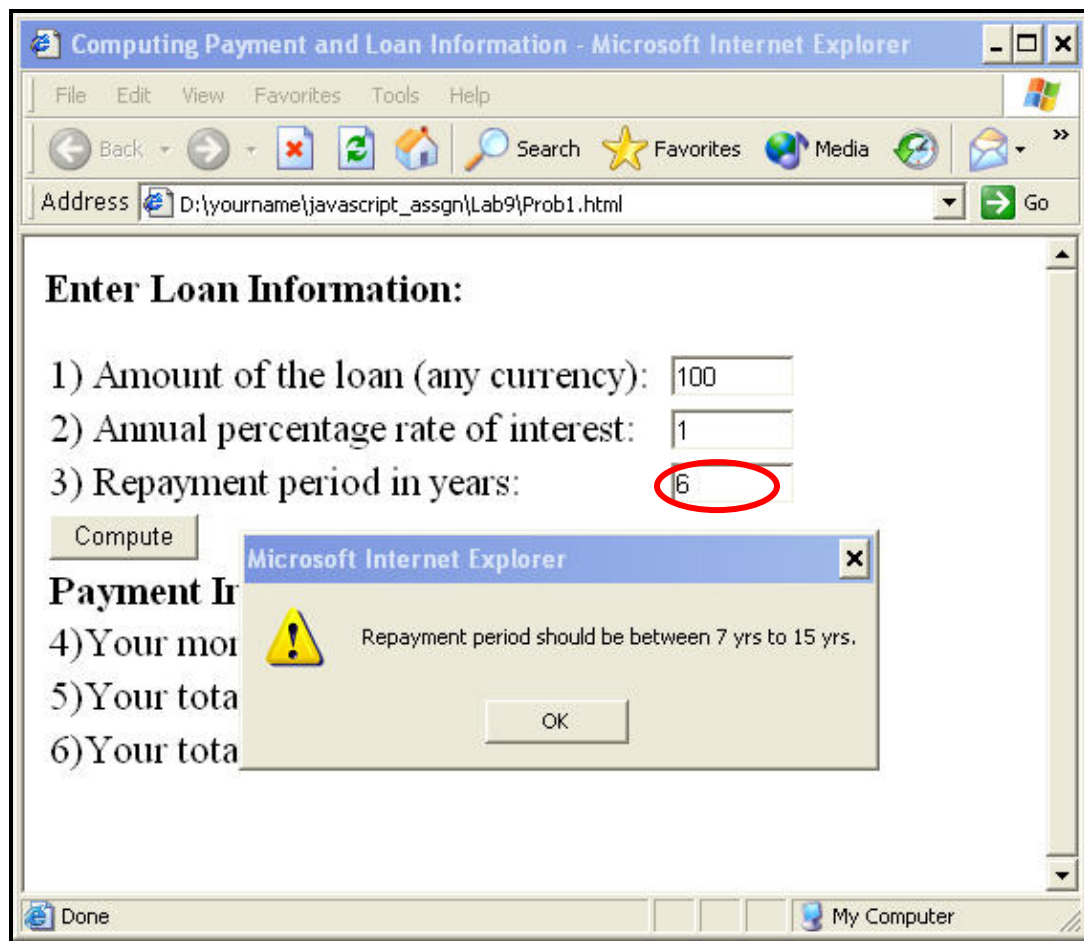


Figure 20: Validating Form elements

Similar kind of error message should be displayed if the amount of loan exceeds 15 lakh.

**Solution:**

**Step 1:** Write the Code, and save it as **Prob1.html**.

Some tips

In the function validloan(field)

/\*

TODO:

the value of the field loan is passed as a argument to this function validloan onblur event of the loan text field. perform validations for empty, nan and the validations specified in the problem statement

\*/

In the function validrate(field)



```

/*
TODO:
the value of the field rate is passed as a argument to this function validate onblur event of
the loan text field. perform validations for empty, nan and the validations specified in the
problem statement
*/

In the function validyrs(field)
/*
TODO:
the value of the field years is passed as a argument to this function validyrs onblur event of
the loan text field. perform validations for empty, nan and the validations specified in the
problem statement
*/

In the function calculate()
/*
TODO:
calculate the monthly payment, total payment, total interest payment on click of the button
with label "compute"
*/

```

Example 9: Lab 8: Prob1.html

**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

## 8.2 Validate Field

Create a **prob2.html** page as shown in the below figure.

The image shows a web form titled "Product Details" with a blue header. Below the header, there is a white box containing the form fields. The fields are: "Category:" with a dropdown menu showing "Electronics", "Product:" with a dropdown menu showing "-----", "Quantity:" with a text input field, and "Total Price:" with a text input field. At the bottom of the form, there are two buttons: "Submit" and "Clear".

Figure 21: Lab 8.2 Product Details

Data should be prepopulated in category list box (Electronics, Grocery). Based on selection of category, product list need to be populated automatically with values as given in the below

table. Also Total price need to be calculated for the entered quantity as per the data in the below table. Total price field should be non-editable field.

Category	Product	Price per quantity in Rupees
Electronics	Television	20000
	Laptop	30000
	Phone	10000
Grocery	Soap	40
	Powder	90

While clicking on submit button, if all the text fields contains valid values then display the filled details in a popup window.

### 8.3 Validate Field

Create a **prob3.html** page with three text fields and a button. Validate all the text fields in the form and submit the form only after it is validated. All the text fields should be validated for emptiness.

Subject of message:

At Patni, We believe we are one Team

Send cc to:

anil.patil@patni.com

Send blind cc to:

krishnan.villivakkam@patni.com

Submit Query

Example 10: Validating Form elements

On clicking the **Submit Query** button, the message should be sent to the email ids specified.

#### Solution:

**Step 1:** Write the code and save it as **Prob2.html**.

##### Some Tips

In the function mailme(form)

/\*

TODO: validate the fields and display error message if it is empty. if the form has valid values concatenate the email ids and return the value to the onsubmit event invoked from form tag.

use mailto:toemailid@site.com?subject= "+" variable which holds the value of the to field if outlook is configured, the above line will open the outlook mail window

\*/

**Step 2:** Open **prob2.html** page in the browser and verify that you get the same output as required.

## <<Stretched Assignments>>

### 8.3: Registration Form

Create a **prob4.html** page containing **Email Registration Form** containing various HTML fields. The following fields are to be validated:

- Login name
- Password
- Re-Type password
- All the fields under “Password Reminder Information”
- FirstName and LastName in “Tell us about yourself”

**Note:** Since the email form is too large to be displayed using a single screen shot, the screen shot of the email form is divided into two halves.

The screenshot shows a web browser window with the title "Email Registration Form". The form is divided into two main sections. The first section, "Email Account Details", includes fields for "Choose your Login Name", "Choose Password", and "Re-type Password", each marked with an asterisk to indicate it is required. The email field is followed by "@patni.com". To the right of these fields, there are two lines of instructional text: "User name should contain only alphabets (a-z), numbers (0-9) and underscore (\_)" and "Password must be atleast 6 characters to ensure better security." The second section, "Password Reminder Information", is marked with an asterisk and the note "(All the Fields are required)". It includes a "Hint Question" dropdown menu, a text field for "OR Create your Question", a "Hint Answer" text field, a "Birthdate" section with dropdowns for "Month", "Day", and "Year", a "Country" dropdown menu (currently showing "India"), and a "City" dropdown menu (currently showing "[Select One]"). To the right of these fields, there is a note: "Choose a Hint Question whose answer only you will know." The browser window's status bar at the bottom shows "Done" on the left and "My Computer" on the right.

Figure 22: Email Registration Form

**Tell us about yourself**

First Name: \*  Last Name: \*

Gender:

State:

Pin Code:

Education:

Occupation:

Alternate Email Address:  (optional)

---

**Tell us your interests (optional)**

Based on your selection we will periodically send you information.

<input type="checkbox"/> Entertainment	<input type="checkbox"/> Computer & Technology	<input type="checkbox"/> Real Estate
<input type="checkbox"/> News	<input type="checkbox"/> Education	<input type="checkbox"/> Credit Card
<input type="checkbox"/> Home & Family	<input type="checkbox"/> Careers	<input type="checkbox"/> Insurance
<input type="checkbox"/> Health & Nutrition	<input type="checkbox"/> Shopping	<input type="checkbox"/> Banking
<input type="checkbox"/> Beauty & Fashion	<input type="checkbox"/> Automobiles	<input type="checkbox"/> Special Offers
<input type="checkbox"/> Sports & Games	<input type="checkbox"/> Investing	
<input type="checkbox"/> Travel	<input type="checkbox"/> Loans	

Figure 23: Email registration form

**Solution:****Step 1:** Write the Code, and save it as **Prob4.html**.**Some Tips**

In the function isDate()

/\*

TODO:

Retrieve the month,date and year values from the drop down list box and validate the fields. At least one item should be selected or else display error message

\*/

}

In the function isName()

/\*

TODO:

Validate Login name, if login name field is blank display an error message. Also login name should not start with number or special characters

\*/

In the function isPass()

/\*

TODO:

Validate the password field. Password field should not be empty, should not be less than 6 characters and this field should match with re-type password field

\*/

```
In the function isFullName()  
/*  
TODO:  
Validate the first name and the last name field. Both the fields should not be empty and  
should not contain numbers.  
*/  
  
In the function isHintQuesAns()  
/*  
TODO:  
Validate hint question. If nothing is selected in hint question field then ensure that the  
create a question field is not empty and vice versa.Also ensure that hint answer is not  
empty.  
*/  
  
In the function isCountry()  
/*  
TODO:  
Validate the country dropdown list and ensure that atleast one item is selected or else  
display an error message  
*/  
  
In the function isCity()  
/*  
TODO:  
Validate the city dropdown list and ensure that atleast one item is selected or else display  
an error message  
*/  
  
In the function validate()  
/*  
TODO:  
Use all the methods above here to validate all the required fields.  
Invoke this method onclick event of the "Register Me" button.  
If all the validation are successful, it should display alter message "Validation Successful:  
*/
```

Example 12: Lab 8: Prob4.html

**Step 2:** Open **prob4.html** page in the browser, and verify that you get the same output as required.

## Lab 9. Regular Expressions in JavaScript

<b>Goals</b>	<ul style="list-style-type: none"> <li>• Understand Regular Expression object</li> <li>• Use Regular Expression object for validating</li> </ul>
<b>Time</b>	90 minutes

### 9.1: Regular Expression

Create a **prob1.html** page which has two text fields – one for the Regular Expression search pattern and the other for the string in which the pattern has to be checked.

The form has two buttons one “**Test Match**” and the other “**Show Match**”.

**Test Match** will test the regular expression against the string. **Show Match** will show the matching part of the string.

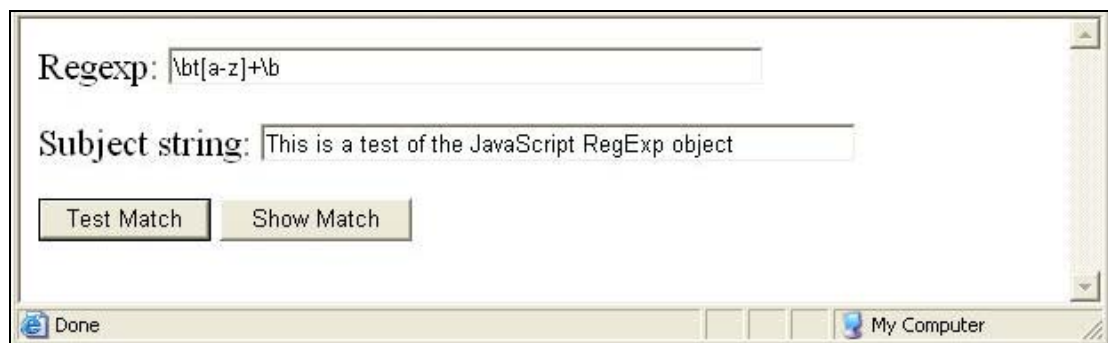


Figure 24: Regular Expression Pattern

Type the regular expression in the first textbox and the string in the second text box. Click the “**Test Match**” button.

If the text in the second text box matches the Regular Expression in the first text box, then it should display a message as shown below.

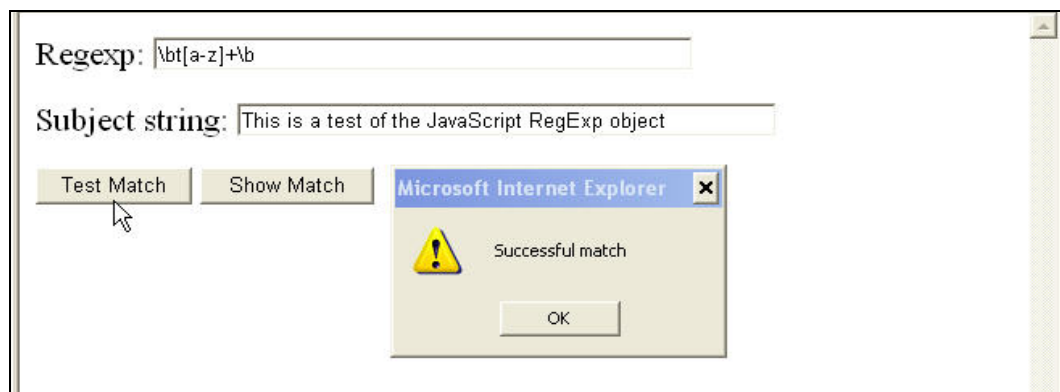


Figure 25: Validating Regular Expression Pattern

If you click the “**Show Match**” button, then it should display the match as shown in the figure given below:

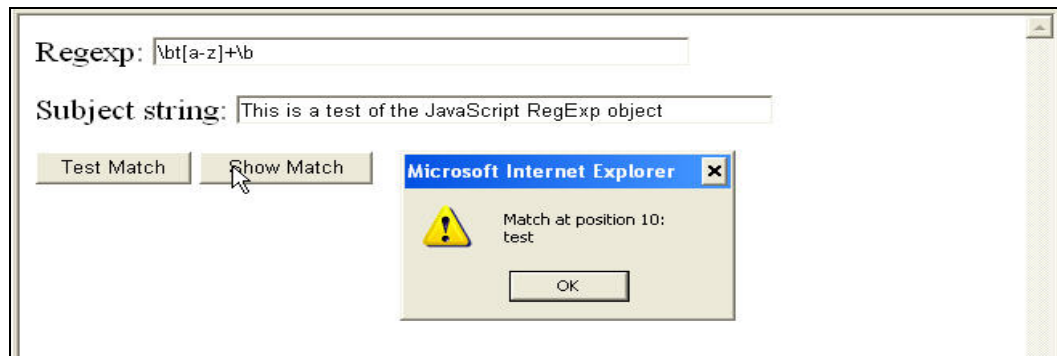


Figure 26: Displaying text matching Regular Expression Pattern

#### Solution:

**Step 1:** Write the Code and save it as **Prob1.html**.

##### Some Tips

In the function demomatchclick()

/\*

TODO:

define a variable re which is the regular expression object, to which the regular expression pattern is passed as an argument. define a variable str which holds the string from subject field match the "str" with "re" using the match method of the string object and display appropriate messages

\*/

In the function demoshowmatchclick()

/\*

TODO:

Store the regular expression pattern in a variable "re" and invoke exec method of regular expression object which takes the matching string as argument and returns the index of the matching string found

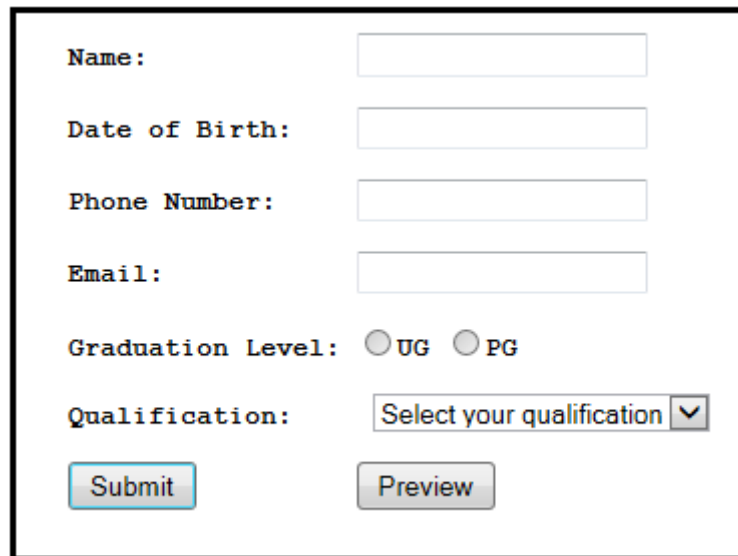
\*/

Example 13: Prob1.html

**Step 2:** Open **prob1.html** page in the browser, and verify that you get the same output as required.

## 9.2: Form Validation using Regular Expression

Create a **prob3.html** page as shown in the below figure. Use CSS for designing page The page should be submitted on clicking the **Submit** button when all the form fields are properly validated.



Name:

Date of Birth:

Phone Number:

Email:

Graduation Level: ☐ UG ☐ PG

Qualification:

**Figure 27: Form Validation using Regular Expression**

1. None of the fields should be empty
2. Name field should be between 3 to 10 characters
3. Date of Birth format can be either (DD/MM/YY or DD/MM/YYYY)
4. Date of Birth should be lesser than current date.
5. Phone Number should be in xxx-xxxx-xxxx format (use Regular Expression)
6. Email ID should be valid.
7. Based on graduation level selected, qualification need to be populated automatically. For an example, if graduation level selected is UG, then qualification should be B.Sc, B.A, B.Com, etc... If graduation level selected is PG, then qualification should be M.A, M.Tech, MCA, MBA, etc...
8. Calculate age of the person and display all the details in a new popup window when "Preview" button is clicked. Details should be printed in the specified format as given below:

Name:

Age:

Phone Number:

Email:

Graduation Level:

Qualification:

Display the appropriate error message adjacent to the fields when the condition gets fails.



## Appendices

### Appendix A: JavaScript Standards

#### 1: Naming conventions for variables in JavaScript:

- Variables must begin with prefix indicating the type of the variable.
  - All integer must start with "int".
  - All floating data types must start with "flt".
  - All string must start with "str".
  - All object name must start with "obj".
  - All Boolean variables must start with "bln".
  - All variables that store date must start with "dt".
  - All constants must be in upper case with different words separated by underscore ( \_ ).
  - All array variables must start with "arr".
  - Apart from these guidelines all variable name must be sensible enough, so that it's purpose can be identified from it's name.

Type	Example
String	strStringName
Boolean	blnPresent
Array	arrArrayName
Object	objObjectName
Date	dtDateName
Integer	intValueInteger
Float	fltValueFloat
Constants	STRING_CONSTANT ARRAY_CONSTANT NUMERIC_CONSTANT

- All HTML elements must be prefixed with appropriate types.
  - TextBox "txt"
  - Image "img"
  - Image map Area "img"
  - option button "opt"
  - CheckBox "chk"
  - DropDown List "lst"
  - Form Name "frm"

- Buttons "btn"
- All div tags "div"
- All class names must start with "cls"
- All user-defined objects must start with "u"
- First letter of each variable/function name must be in upper case. Rest all letters must be in lowercase.
- Use of underscore and digits for naming variables must be avoided.

Tag	Example
Div	divContent
Class	clsInterest
Form	frmContainer
Image	imgMapThis
Button	btnOk
TextBox	txtInterestRate
CheckBox	chkAllow
Option Button	rdbRate
DropDownList	lstState

It must be noted that this naming style does not apply to HTML elements. However, when these elements are accessed in the JavaScript functions, these naming conventions must be followed. This document describes coding convention only for JavaScript.

## 2: Commenting

- Comments related to a particular line of code should be on the same line after the statement gets over.

```

If (dtToday == "15/07/99") {           // Is date birthdate?
    alert ("Happy Birthday");           // Give birthday message
} else {
    alert ("Happy Day");                 // Give standard message
}

```

- Over all commenting should consist of two parts – Comment header and Comment footer. Comment header must precede the block of code and Comment footer must follow the block of code.

```
//Function Name: calculateInterest
//Description: This function calculates the interest. It accepts the initial investment
//              and period for which the amount is invested. Rate of interest is
//              fixed. Formula is
//              fltInterest = fltAmount * fltPeriod *fltRATE/100
//              Dhrumil Dalal
//              15/07/1999
//Author:       fltAmount – indicated the amount invested
//Start Date:   fltPeriod – Indicates the period of investment
//Input        Calculated interest
Parameters:
//
//Return Value:
Function calculateInterest(fltAmount,fltPeriod){
}

//End of function for calculating the rate of interest
```

### 3: Documentation

- All variables used in the function must be declared in brief.
- Only one variable declaration per line.
- Describe each variable on the same line and description should not be more than one line.
- All functions must be preceded by comments. Comments must describe the following:
  - Input parameters.
  - Return value.
  - Function logic in brief.
  - Starting date.
  - Name of the author.
  - Revision history.
- After the end of function, there must be block of comment indicating the end of function.

```
//Function Name: calculateInterest
//Description: This function calculates the interest. It accepts the initial
               investment and period for which the amount is invested.
```

```
//
//      Rate of interest is fixed. Formula is
//      fltInterest = fltAmount * fltPeriod      *fltRATE/100
//      Dhrumil Dalal
//      15/07/1999
//      fltAmount – indicated the amount invested
//Author:      fltPeriod – Indicates the period of investment
//Start Date:      Calculated interest
//Input Parameters:
//
//Return Value:
Function calculateInterest(fltAmount,fltPeriod){
Var fltRATE = 12.5; // fixed rate of interest
Var fltInterest; // The variable to store calculated interest
fltInterest = fltAmount * fltPeriod * fltRATE/100 ;
return fltInterest;
}

//End of function for calculating the rate of interest
```

#### 4: Coding Styles

- For statements which may have block of code enclosed in {}, the opening brace "{" must immediately follow the statement and the closing brace "}" must be below the statement. That is to say, the closing brace and first letter of the statement must be same in the column.

```
If (condition) {
...
} else {
  if (condition) {
    ...
  } else {
    ...
  }
}

for(intCounter=0; intCounter <= 5; intcounter++){
  //Perform calculation.
  //Display Result}
```

- All statements within corresponding opening and closing brace must be indented. Indentations must be in odd columns.

Column no

```
123456789.....  
if (condition) {  
    ...  
} else {  
    if (condition) {  
        ...  
    } else {  
        ...  
    }  
}
```

- Also the code should not extend past the 80th column so that it is required to scroll to the right or left to edit a particular line. In the case of strings which do not fit on one line, it is recommended that temporary variables be used with the string concatenation operator (+) to construct strings of longer lengths. The following example illustrates this:

```
Column no  
123456789.....80  
strMessage = "Demonstrating the use of ... ";  
strMessage += "prepared on 15-07-1999";
```

## Appendix B: Coding Best Practices

### JavaScript Best Practice

The following demonstrate the best practices that should be followed while writing JavaScript code.

#### 1: Inline JavaScript source code

Any JavaScript code that does not write out to the document should be placed within the head of the document.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
<!--
function functionName() {
    alert(text);
}

var text = 'Hello World';
//-->
</SCRIPT>
</HEAD>
```

Example 14: Sample Code

This ensures that the browser has loaded the JavaScript function definitions before it is required. It also makes it slightly easier to maintain the JavaScript code if it can always be found in the head of the document.

#### 2: JavaScript Links

Avoid using the **javascript:** protocol as a default URL within a link.

If JavaScript is disabled, then the link will not work. Do not use the following:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function functionName() {
    alert('Hello world');
}
//--></SCRIPT>
<A HREF="javascript:functionName()">text link</A>
```

Example 15: Sample Code

Instead, use JavaScript itself to override the **href** property of the link:

```

<SCRIPT LANGUAGE="JavaScript">
<!--
function functionName() {
    alert('Hello world');
}
//-->
</SCRIPT>

<A HREF="default.htm" onClick="this.href='javascript:functionName()'">text link</A>

```

Example 16: Sample Code

### 3: Avoid Using Void

All browsers do not support the **void** function. Create your own **void** function.

The in built **void()** function is supported since JavaScript 1.1. Therefore it is best to create your own void function rather than rely on JavaScript 1.1 being available.

```

<SCRIPT LANGUAGE="JavaScript">
<!--
function myVoid() { } // create a void function
//-->
</SCRIPT>

<A HREF="#" onClick="this.href='javascript:myVoid()'">non functional text link</A>

```

Example 17: Sample Code

### 4: JavaScript Performance

Avoid writing output multiple times to the document, concatenate the data, and then write all in one go.

With the introduction of Netscape Navigator 4, the rendering of JavaScript generated HTML slowed down considerably.

The following writes the HTML output to the document in one go:

```

<SCRIPT LANGUAGE="JavaScript">
<!--
var output = '<P>';
output += 'Last modified: ';
output += document.lastModified;
output += '<\P>'
document.write(output);
//-->
</SCRIPT>

```

Example 18: Sample Code

---

## 5: Select Form Fields

Use the Netscape method to correctly navigate select field properties.

The following technique works in Microsoft Internet Explorer. However it should be avoided.

```
<SCRIPT LANGUAGE="JavaScript">
<!--
var property = document.formName.selectName.propertyName
//-->
</SCRIPT>
```

Example 19: Sample Code

Whereas the following will work correctly in all browsers:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
var property =
document.formName.selectName.options[document.formName.selectName.selectedIndex].propertyName
//-->
</SCRIPT>
```

Example 20: Sample Code

## 6: Changing Location

Do not use the following:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
location = 'page.htm';
//-->
</SCRIPT>
```

Example 21: Sample Code

The later approach is confusing as it is not clear whether you are changing the location property of the “window” or the “document object”.

Changing the location using the document is deprecated and causes problems on later browsers. Use the following:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
window.location.href = 'page.htm';
//-->
</SCRIPT>
```

Example 22: Sample Code



---

## 7: Opening Windows

While opening a new popup window using JavaScript, there are several points to bear in mind.

To be able to control the popup window from the **opener** window, always retain the returned reference from the window's open method:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
var windowHandle = window.open('page.htm','windowName','width=600,height=320');
//-->
</SCRIPT>
```

Example 23: Sample Code

To avoid errors while referring to the **opener** window from the **popup** window, always check for the in-built browser support for the **opener** property. If necessary, provide your own:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
var windowHandle = window.open('page.htm','windowName','width=600,height=320');
if (!windowHandle.opener)
    windowHandle.opener = self;
//-->
</SCRIPT>
```

Example 24: Sample Code

While updating the contents of a newly opened window, give the browser time to open the window and to load the initial contents:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
function update() {
    windowHandle.document.open();
    windowHandle.document.write('<H1>Hello World</H1>');
    windowHandle.document.close();
}

var windowHandle = window.open('page.htm','windowName','width=600,height=320');
if (!windowHandle.opener)
    windowHandle.opener = self;
setTimeout('update()',2000);
//-->
</SCRIPT>
```

Example 25: Sample Code

### 8: JavaScript Entities

JavaScript Entities are only supported by Netscape Navigator. Avoid their use.

The following will cause errors in other browsers:

```
<HR WIDTH="{barWidth};">
```

Example 26: Sample Code

Instead the following can be used.

```
<SCRIPT LANGUAGE="JavaScript">
<!--
document.write('<HR WIDTH="' + barWidth + '%">');
//-->
</script>
```

Example 27: Sample Code

## Appendix C: Table of Figures

Figure 1: Welcome to JavaScript .....	7
Figure 2: Formatting Text in JavaScript.....	8
Figure 3: Using external JavaScript File.....	9
Figure 4: Lab 1: Prob4.html .....	9
Figure 5: Embedding Script tags in HTML document .....	10
Figure 6: Using Variable in many Script tags .....	12
Figure 7: For loop in JavaScript.....	13
Figure 8: Operators and Arithmetic Expression.....	14
Figure 9: Displaying Date using Date Object.....	15
Figure 10: Using indexOf function of String object.....	16
Figure 11: Using various String methods.....	16
Figure 12: Using Array to display values .....	17
Figure 13: Interface to accept window coordinates .....	18
Figure 14: Opening a window.....	19
Figure 15: Location Object Properties .....	20
Figure 16: Displaying Products .....	21
Figure 17: Validating Products.....	22
Figure 18: Displaying Invoice details in a new window .....	22
Figure 19: Validating Form elements .....	23
Figure 20: Validating Form elements .....	24
Figure 21: Lab 8.2 Product Details .....	25
Figure 22: Email Registration Form.....	27
Figure 23: Email registration form .....	28
Figure 24: Regular Expression Pattern .....	30
Figure 25: Validating Regular Expression Pattern.....	30
Figure 26: Displaying text matching Regular Expression Pattern.....	31
Figure 27: Form Validation using Regular Expression.....	32

---

**Appendix D: Table of Examples**

Example 1: Lab 1: Prob1.html .....	6
Example 2: Lab 1: Prob2.html .....	7
Example 3: Lab 1: Prob3.html .....	8
Example 4: Lab 1: HelloWorld.js .....	8
Example 5: Lab 1: Hello.js .....	10
Example 6: Lab 1: Prob5.html .....	11
Example 7: Lab 1: common.js .....	11
Example 9: Lab 6: Prob1.html .....	19
Example 10: Lab 8: Prob1.html .....	25
Example 11: Validating Form elements .....	26
Example 12: Lab 8: Prob2.html .....	27
Example 13: Lab 8: Prob4.html .....	29
Example 14: Prob1.html .....	31
Example 15: Sample Code .....	38
Example 16: Sample Code .....	38
Example 17: Sample Code .....	39
Example 18: Sample Code .....	39
Example 19: Sample Code .....	39
Example 20: Sample Code .....	40
Example 21: Sample Code .....	40
Example 22: Sample Code .....	40
Example 23: Sample Code .....	40
Example 24: Sample Code .....	41
Example 25: Sample Code .....	41
Example 26: Sample Code .....	42
Example 27: Sample Code .....	42
Example 28: Sample Code .....	42