Client-Side Scripting Language (22519)

Vidyalankar Polytechnic

Unit 3
Form and Event Handling

Unit Contents

Unit Outcomes

- ✓ Write JavaScript to design a form to accept input values for the given problem.
- ✓ Use JavaScript to implement form events to solve the given problems.
- ✓ Develop JavaScript to dynamically assign specified attribute value to the given form control.
- ✓ Use the given intrinsic functions with specified parameters.

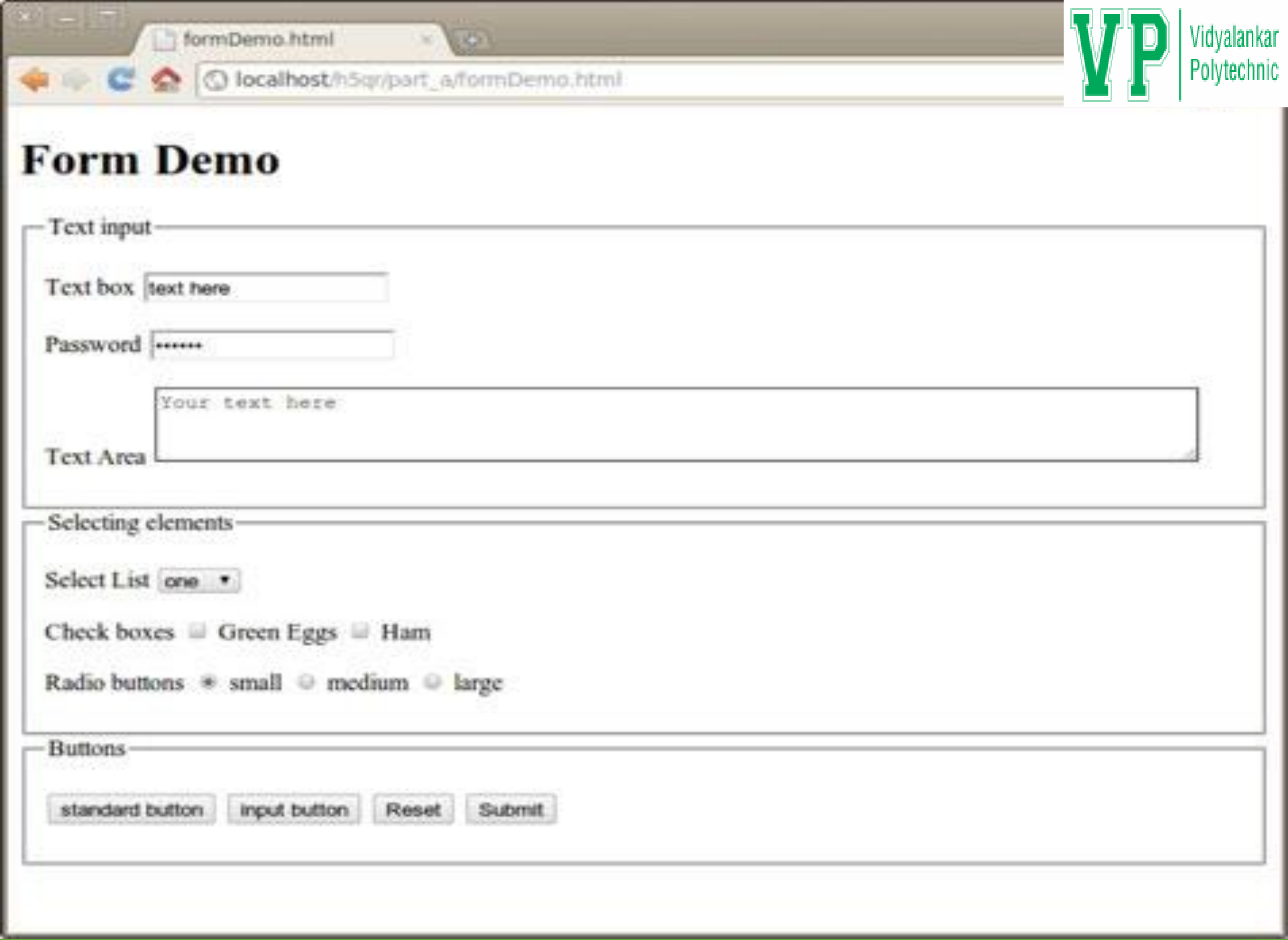
Introduction to Form

- ✓ Forms are one of the most common web page elements used with JavaScript.
- ✓ JavaScript is commonly used with following two reasons:
  - ➢ *To add functionality that makes forms easier for users to fill out*
  - ➢ *To validate or process the data that a user enters before that data is submitted to a server side script.*

Building Blocks of a Form

- A form is a section of an HTML document that contains elements such as radio buttons, text boxes and option lists. HTML form elements are also known as controls.
- Elements are used as an efficient way for a user to enter information into a form. Typical form control objects also called “widgets” includes the following:

1. **Text box** for entering a line of text.
2. **Push button** for selecting an action.
3. **Radio buttons** for making one selection among a group of options.
4. **Check boxes** for selecting or deselecting a single, independent option.


formDemo.html

localhost/h5qr/part_a/formDemo.html

Form Demo

Text input

Text box text here

Password •••••••

Text Area Your text here

Selecting elements

Select List one

Check boxes Green Eggs Ham

Radio buttons [✓ small] medium large

Buttons

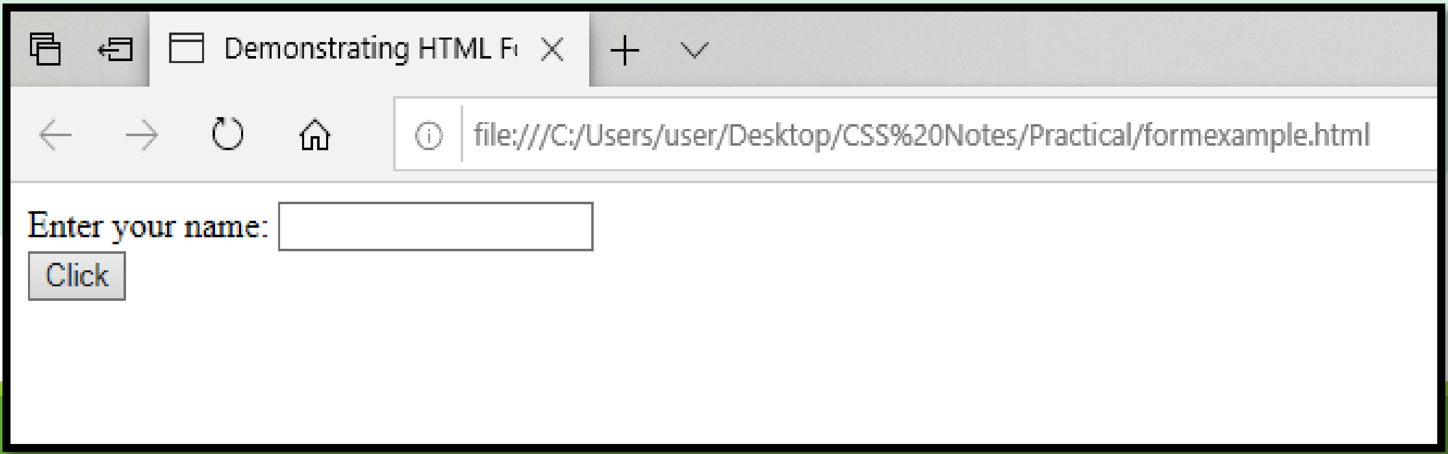
standard button | input button | Reset | Submit

Vidyalankar Polytechnic

Example

```
<html>
<head>
<title> Demonstrating HTML Forms </title>
</head>
<body>
<form name = “DemoForm” action = “ ” method = “GET”>
Enter your name: <input type = “text” name = “text1”> <br>
<input type = “button” name = “button1” value = “Click” onclick = "show()" >
</form>
</body>
</html>
```

Output


Demonstrating HTML Fo
file:///C:/Users/user/Desktop/CSS%20Notes/Practical/formexample.html
Enter your name:
Click

Form Object and Elements

The form object is represented by HTML <form> element.

Syntax

```
<form name = “id=” action = “onSubmit=”>
....
....
....
....
....
</form>
```

1. **id:** The unique identifier.
2. **name:** Its role is equivalent to that of identifier but for the function of DOM, because the method getElementById is based on id.
3. **action:** The name of a script to be loaded when the form is submitted, with submit button.
4. **onSubmit:** Event fired up when the form is submitted and before the execution of the action. It can allow to determine if the action must be executed or not.

Form Object and Elements

Example

```
<form name = “myform” id = “myform” action = “page.html” onSubmit = “test()”>
....
....
....
....
....
</form>
```

HTML Form Elements

Sr. No.	HTML Element	Type Property	Event Handler	Description and Events
1	<input type = “button”> **or** <button type = “button”>	“button”	onclick	A push button
2	<input type = “checkbox”>	“checkbox”	onchange	A toggle button without radio button behaviour
3	<input type = “file”>	“file”	onchange	An input held for entering the name of a file to upload to the web server, value property is read only
4	<input type = “hidden”	“hidden”	none	Data submitted with the form but not visible to the user.
5	<option>	none	none	A single item within a select object, event handlers are an the select object and not on individual option objects.

HTML Form Elements

Vidyalankar Polytechnic

Sr. No.	HTML Element	Type Property	Event Handler	Description and Events
6	<input type = "password">	"password"	onchange	An input field for password entry where typed characters are not visible.
7	<input type = "radio">	"radio"	onchange	A toggle button with radio button behaviour where only one item is selected at a time
8	<input type = "reset"> **or** <button type = "reset">	"reset"	onclick	A push button that resets a form.
9	<select>	"select-one"	onchange	A list or drop down menu from which one item may be selected.
10	<select multiple>	"select-multiple"	onchange	A list from which multiple items are selected.

HTML Form Elements

Sr. No.	HTML Element	Type Property	Event Handler	Description and Events
11	<input type = “submit”> **or** <button type = “submit”>	“submit”	onclick	A push button that submits a form.
12	<input type = “text”>	“text”	onchange	A single line text entry field.
13	<textarea>	“textarea”	onchange	A multi line text entry field.

<input> tag with its parameters

1. name: Can be used so that the value of the element can be processed.

2. type: Can be used to specify the type of input.

3. id: Identification name of element.

4. value: Can be used to specify the initial value. It is required when type is set to checkbox or radio. It should not be used when type is set to file.

5. checked: Can be used when type is set to checkbox or radio to set the initial state of a checkbox or radio button to be selected.

6. maxlength: Can be used to specify the maximum number of characters allowed in a textbox.

7. src: Can be used when type is set to image to specify the location of an image file.

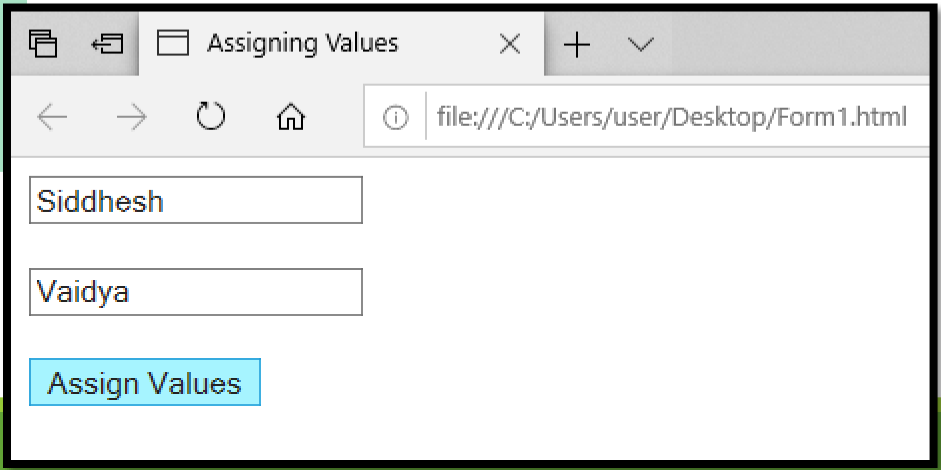
8. alt: Can be used when type is set to image to specify the alternative text of the image, which should be a short description.

Example

```
<html>
<head>
<title> Assigning Values </title>
<script type="text/javascript">
function assign()
{
document.forms.fullname.name.value="Siddhesh";
document.forms.fullname.surname.value="Vaidya";
}
</script>  </head>
```

```
<body>
<form id="fullname">
<input id="name"> <br> <br>
<input id="surname"> <br> <br>
<input type="button" id="btn" value="Assign
Values" onclick="assign()">
</form>
</body></html>
```

Output



Properties and Methods of Form

- The Form object represents a <form> element in an HTML document. The elements property is an HTML collection that provides convenient access to all elements of the form. The submit() and reset() methods allow a form to be submitted or reset under program control.
- Each form in a document is represented as an element of the documents.forms[] array. The elements of a form are collected in the array like object Form.elements.

Properties of Form

Sr. No.	Properties	Description
1	action	Read/Write string that reflects the action attribute of the form.
2	elements[]	An array containing all of the elements of the form. Use it to loop through form easily.
3	encoding	Read/Write string that specifies how the form data is encoded.
4	length	The number of elements in the form.
5	method	Read/Write string that specifies how the method the form is submitted.
6	name	The name of the form.
7	target	The name of the target frame or window form is to be submitted to.

Methods of Form

Sr. No.	Methods	Description
1	reset()	Resets the form
2	submit()	Submits a form

Methods of Events

Sr. No.	Methods	Description
1	onReset	Code is executed when the form is reset.
2	onSubmit	Code is executed when form is submitted.

- Forms and the elements they contain can be selected from a document using standard methods like:
  1. getElementById()
  2. getElementsByName()
  3. getElementsByTagName()
  4. innerHTML Property

getElementById()

- It returns a reference to the elements by its ID.

Syntax:

```
element = document.getElementById(id);
```

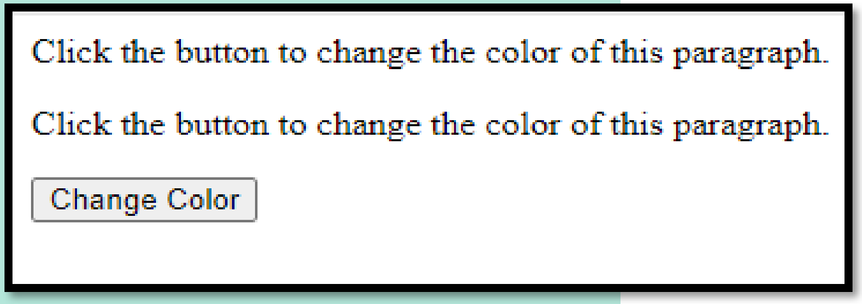
where,

- element is a reference to an element object, or null if an element with the specified ID is not in the document.
- id is a case sensitive string representing the unique ID of the element being sought.

Example

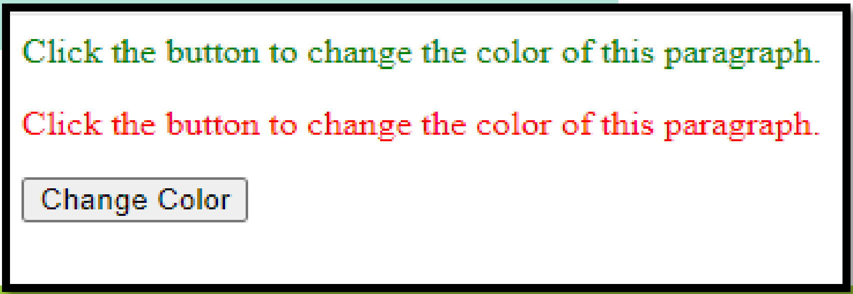
```
<html>
<body>
<p id="demo">Click the button to change the color of this paragraph.</p>
<p id="demo1">Click the button to change the color of this paragraph.</p>
<button onclick="myFunction()">Change Color</button>
<script>
function myFunction()
{
var x = document.getElementById("demo");
var y = document.getElementById("demo1");
x.style.color = "green";
y.style.color = "red";
}
</script>
</body>
</html>
```

Output



Click the button to change the color of this paragraph.

Click the button to change the color of this paragraph.



Click the button to change the color of this paragraph.

Click the button to change the color of this paragraph.

getElementsByName()

- It returns a collection of all elements in the document with the specified name (the **value** of the name attribute).

Syntax:

- element = document.getElementsByName(name);

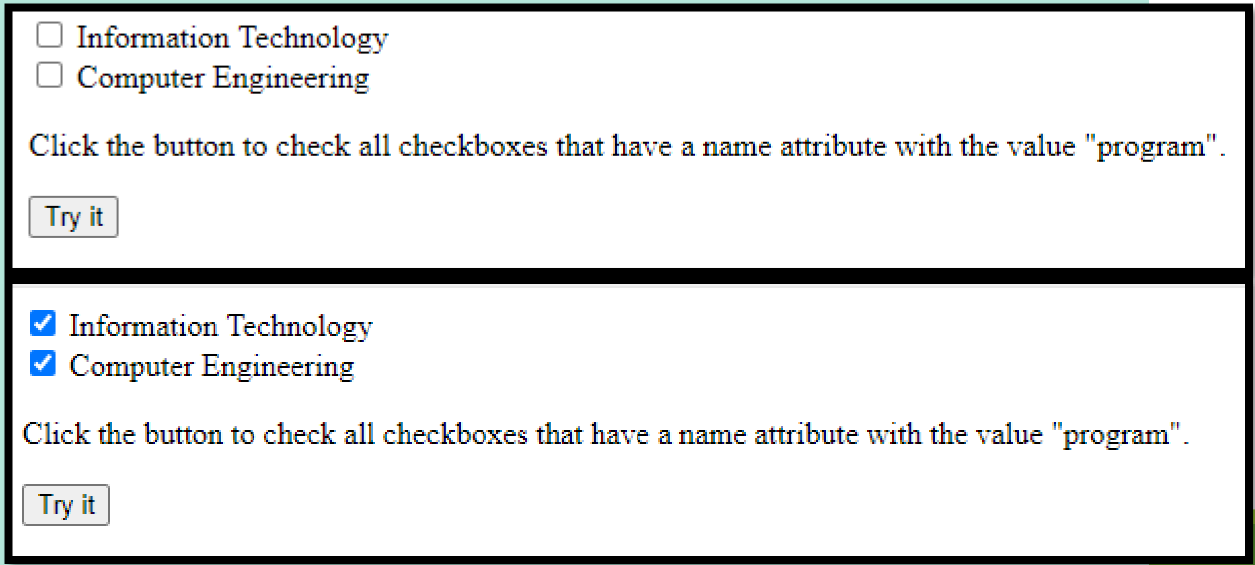
where,

- element is a reference to an element object.
- name is a case sensitive string representing the name of the element being sought.

Example

```
<html>
<body>
<input name="program" type="checkbox" value="IF"> Information Technology <br>
<input name="program" type="checkbox" value="CO"> Computer Engineering <br>
<p>Click the button to check all checkboxes that have a name attribute with the value
"program".</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction()
{
var x = document.getElementsByName("program");
for (var i = 0; i < x.length; i++)
{
if (x[i].type == "checkbox")
{
x[i].checked = true;
}
}
}
</script>
</body>
</html>
```

Output



```
☐ Information Technology
☐ Computer Engineering

Click the button to check all checkboxes that have a name attribute with the value "program".

[Try it]
```

```
☑ Information Technology
☑ Computer Engineering

Click the button to check all checkboxes that have a name attribute with the value "program".

[Try it]
```

getElementsByTagName()

- It returns an HTMLCollection of elements with the given tag name. The complete document is searched, including the root.
- The returned HTMLCollection is live, meaning that it updates itself automatically to stay in sync with the DOM tree without having to all **document.getElementsByTagName()** again.

Syntax:

```
elements = document.getElementsByTagName(tagname);
```

where,

elements is a live HTMLCollection of found elements in the order they appear in the tree.

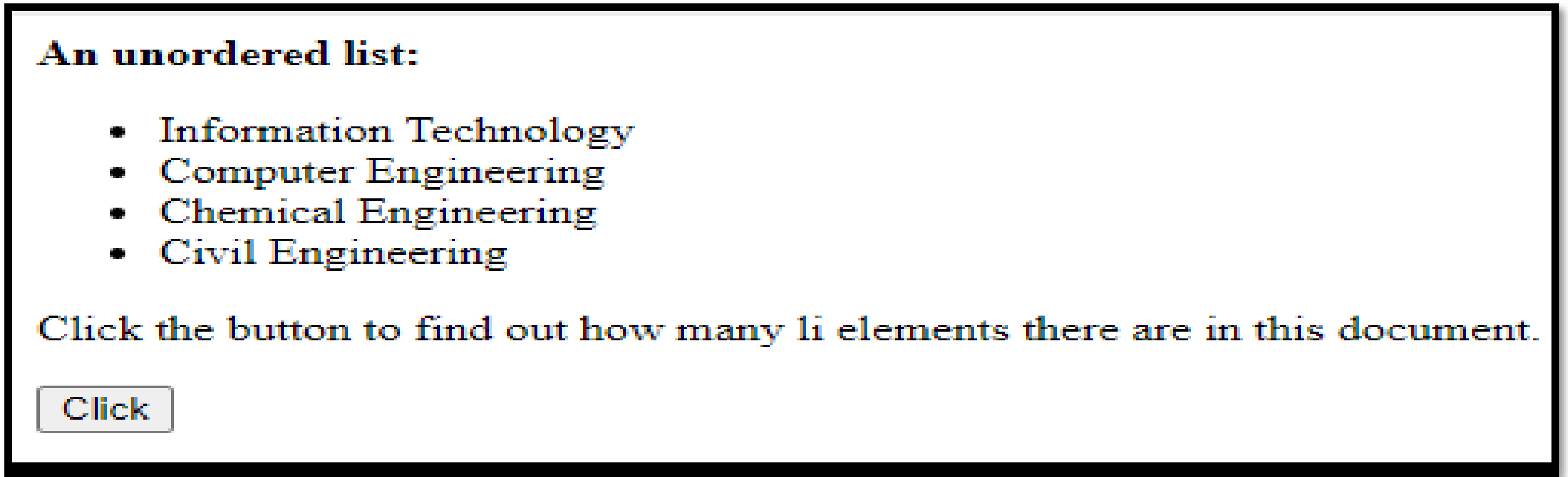
name is a string representing the name of the elements.

The special string **"*"** represents all elements.

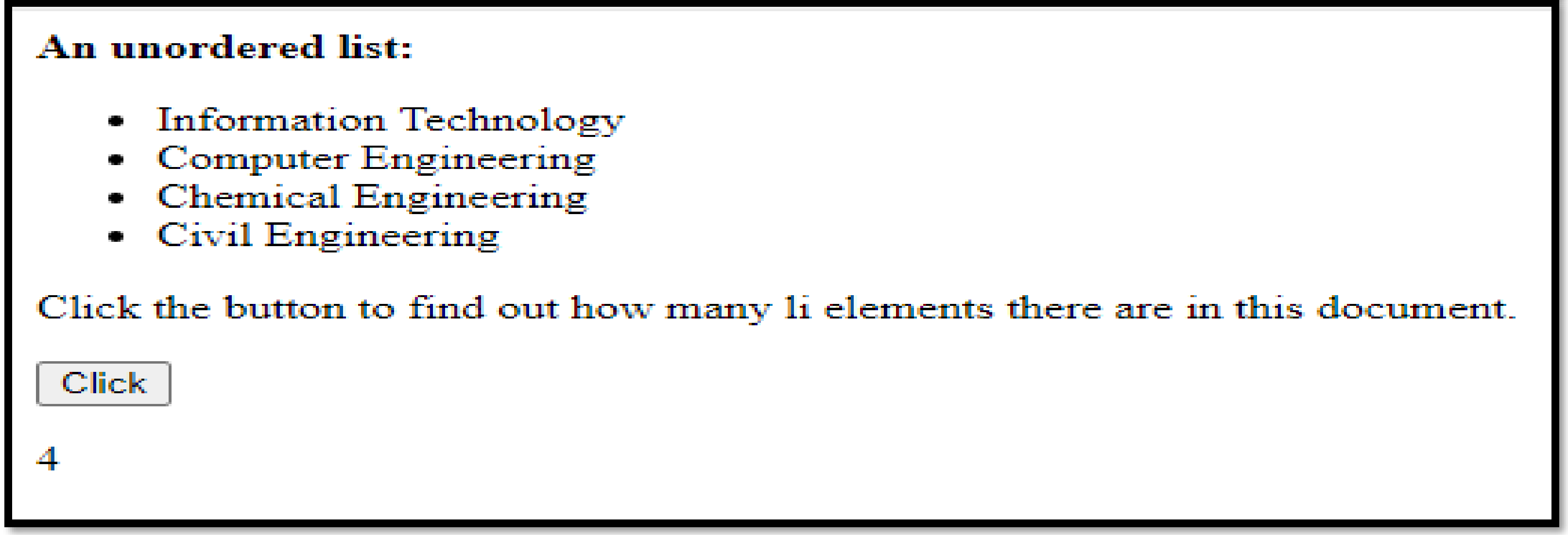
Example 1

```
<html>
<body>
<strong><p>An unordered list:</p></strong>
<ul>
<li>Information Technology</li>
<li>Computer Engineering</li>
<li>Chemical Engineering</li>
<li>Civil Engineering</li>
</ul>
<p>Click the button to find out how many li elements there are in this
document.</p>
<button onclick="myFunction()">Click</button>
<p id="demo"></p>
<script>
function myFunction()
{
var x = document.getElementsByTagName("li");
document.getElementById("demo").innerHTML = x.length;
}
</script>
</body>
</html>
```

Output



```
An unordered list:

    • Information Technology
    • Computer Engineering
    • Chemical Engineering
    • Civil Engineering

Click the button to find out how many li elements there are in this document.

[Click]
```

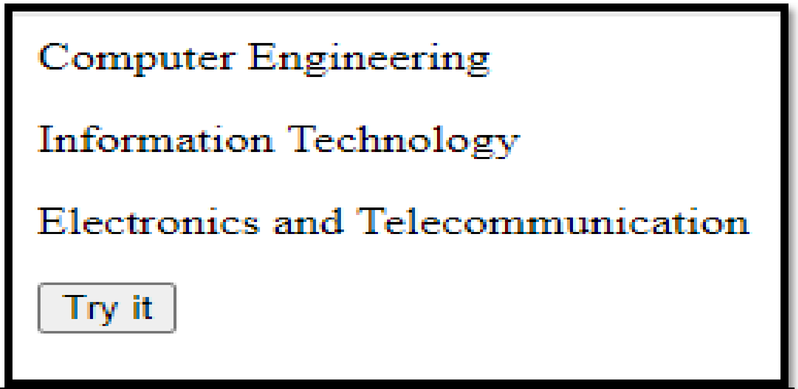


```
An unordered list:

    • Information Technology
    • Computer Engineering
    • Chemical Engineering
    • Civil Engineering

Click the button to find out how many li elements there are in this document.

[Click]

4
```

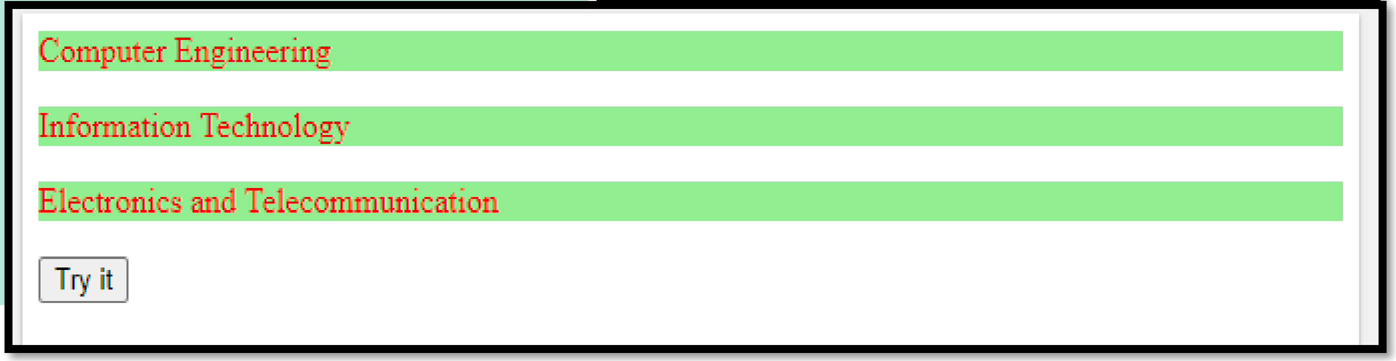
Example 2

```
<html>
<body>
<p>Computer Engineering</p>
<p>Information Technology</p>
<p>Electronics and Telecommunication</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction()
{
var x = document.getElementsByTagName("p");
for (var i = 0; i < x.length; i++)
{
x[i].style.backgroundColor = "lightgreen";
x[i].style.color = "red";
}
}
</script>
</body>
</html>
```

Output



```
Computer Engineering

Information Technology

Electronics and Telecommunication

Try it
```



```
Computer Engineering

Information Technology

Electronics and Telecommunication

Try it
```

innerHTML Property

- The easiest way to modify the content of an HTML element is by using the innerHTML property.

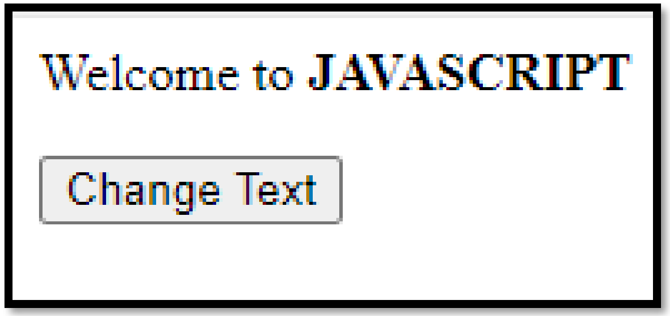
Syntax:

document.getElementById(*id*).innerHTML = *new HTML;*

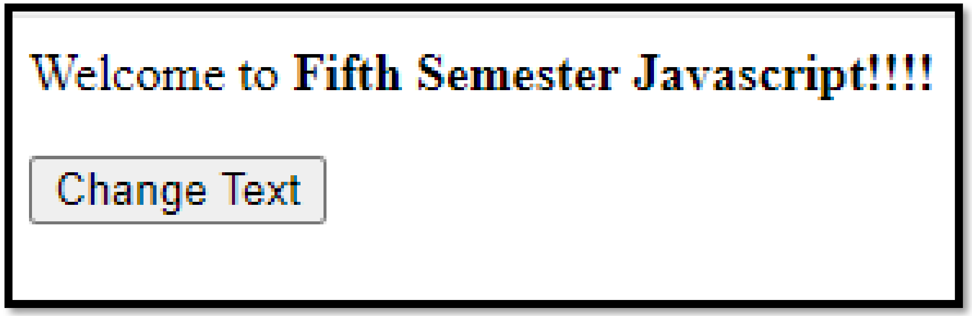
Example 1: To change the text by using innerHTML property

```
<html>
<body>
<script type="text/javascript">
function changeText()
{
document.getElementById('js').innerHTML = "Fifth Semester Javascript!!!!";
}
</script>
<p>Welcome to <b id="js">JAVASCRIPT</b> </p>
<input type="button" onclick="changeText()" value="Change Text"/>
</body>
</html>
```

Output



Welcome to **JAVASCRIPT**

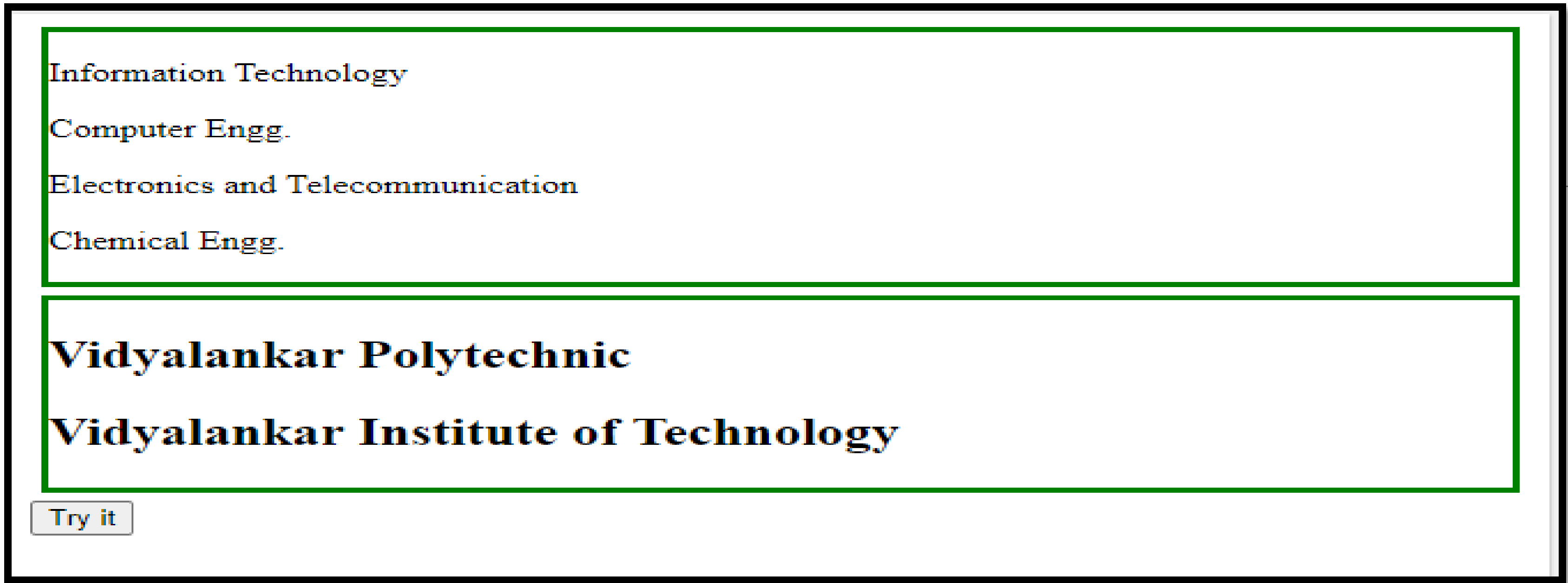


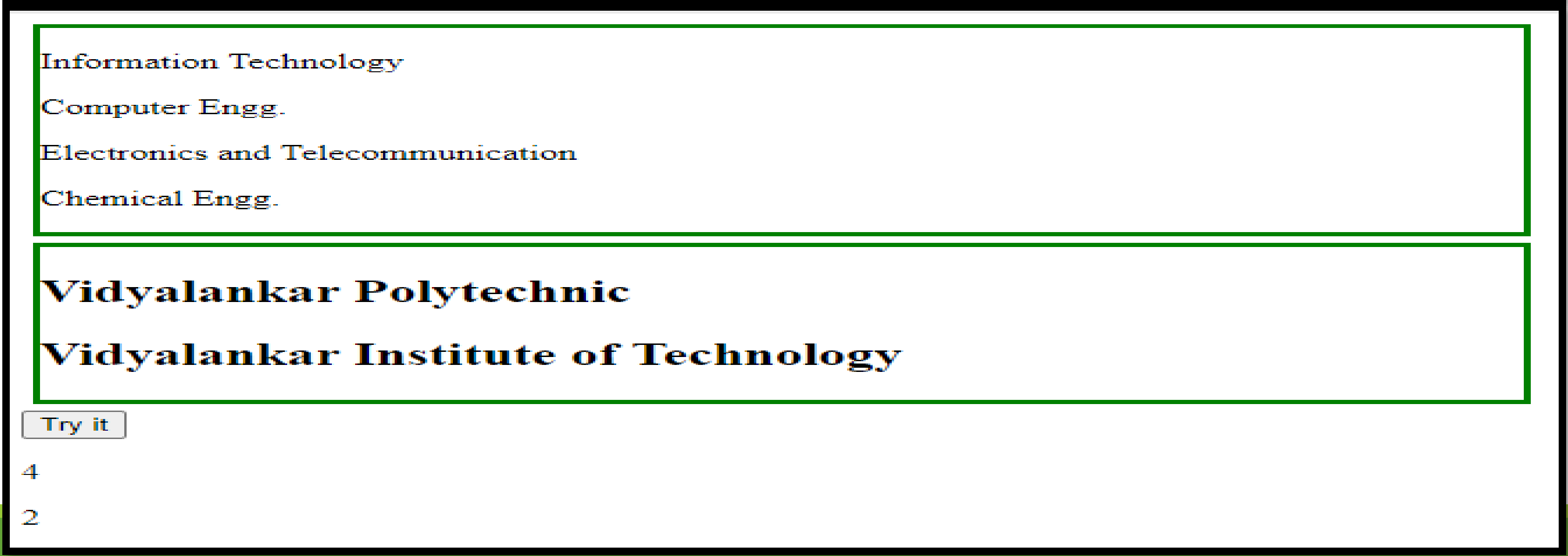
Welcome to **Fifth Semester Javascript!!!!**

Example 2: Script to count the number of <p> tag and <h2> tag

```
<html>
<style>
div
{
border: 1px solid black;
margin: 5px;
}
</style>
<body>
<div id="myDIV">
<p>Information Technology</p>
<p>Computer Engg.</p>
<p>Electronics and Telecommunication</p>
<p>Chemical Engg.</p>
</div>
<div id="myh">
<h2>Vidyalankar Polytechnic</h2>
<h2>Vidyalankar Institute of Technology </h2>
</div>
<button onclick="myFunction()">Try
it</button>
<p id="demo"></p>
<p id="demo1"></p>
<script>
function myFunction()
{
var x =
document.getElementById("myDIV").getEleme
ntsByTagName(“p");
document.getElementById("demo").innerHTML
= x.length;
var y =
document.getElementById("myh").getElements
ByTagName("h2");
document.getElementById("demo1").innerHTM
L = y.length;
}
</script>
</body>
</html>
```

Output


```
Information Technology

Computer Engg.

Electronics and Telecommunication

Chemical Engg.

Vidyalankar Polytechnic

Vidyalankar Institute of Technology
```

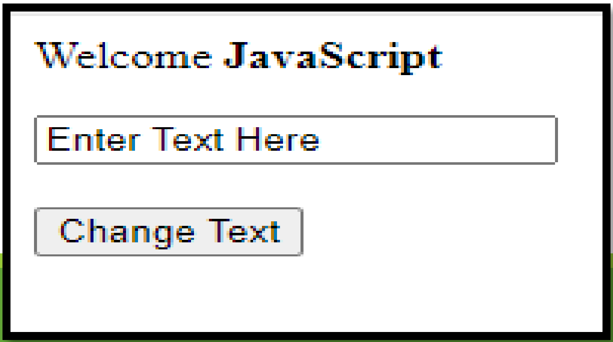

```
Information Technology

Computer Engg.

Electronics and Telecommunication

Chemical Engg.

Vidyalankar Polytechnic

Vidyalankar Institute of Technology
```

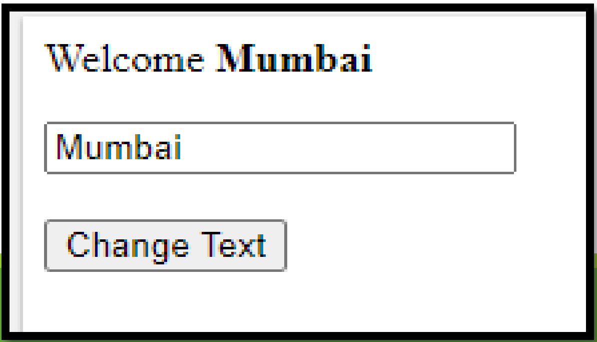
```
4

2
```

Example 3: Change content by providing user input

```
<html>
<body>
<p>Welcome  <b id="vp">JavaScript</b> </p>
<input type="text" id="userInput" value="Enter Text Here"> <br> <br>
<input type="button" onclick="changeText()" value="Change Text">
<script>
function changeText()
{
var userInput = document.getElementById("userInput").value;
document.getElementById("vp").innerHTML = userInput;
}
</script>
</body>
</html>
```

Output

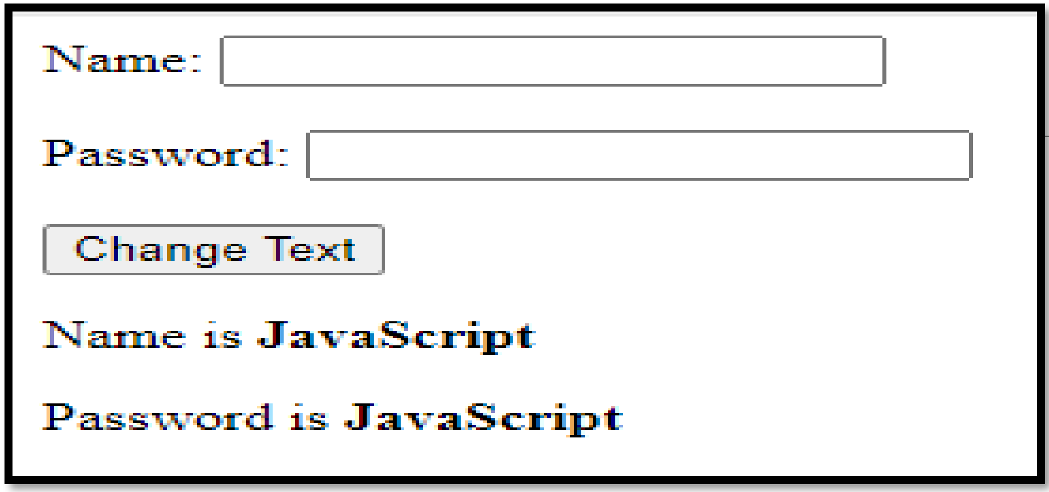

Welcome **JavaScript**


Welcome **Mumbai**

Example 4: Displaying user input by clicking on button

```
<html>
<body>
Name: <input type="text" id="userInputName" value=""> <br> <br>
Password: <input type="password" id="userInputPwd" value=""> <br> <br>
<input type="button" onclick="changeText()" value="Change Text">
<p>Name is <b id="vp">JavaScript</b> </p>
<p>Password is <b id="vp1">JavaScript</b> </p>
<script>
function changeText()
{
var userInputName = document.getElementById("userInputName").value;
document.getElementById("vp").innerHTML = userInputName;
var userInputPwd = document.getElementById("userInputPwd").value;
document.getElementById("vp1").innerHTML = userInputPwd;
}
</script>
</body>
</html>
```

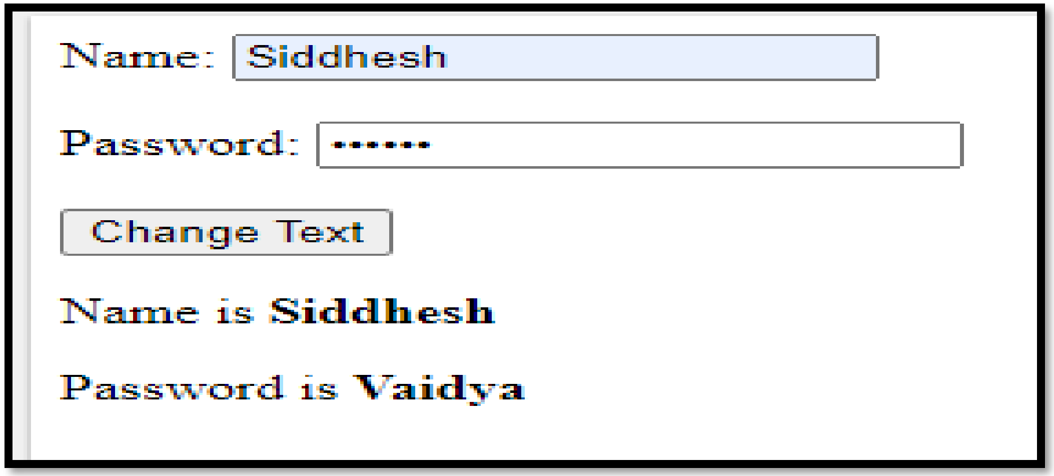
Output


Name:

Password:

Name is **JavaScript**

Password is **JavaScript**


Name: Siddhesh

Password: •••••••

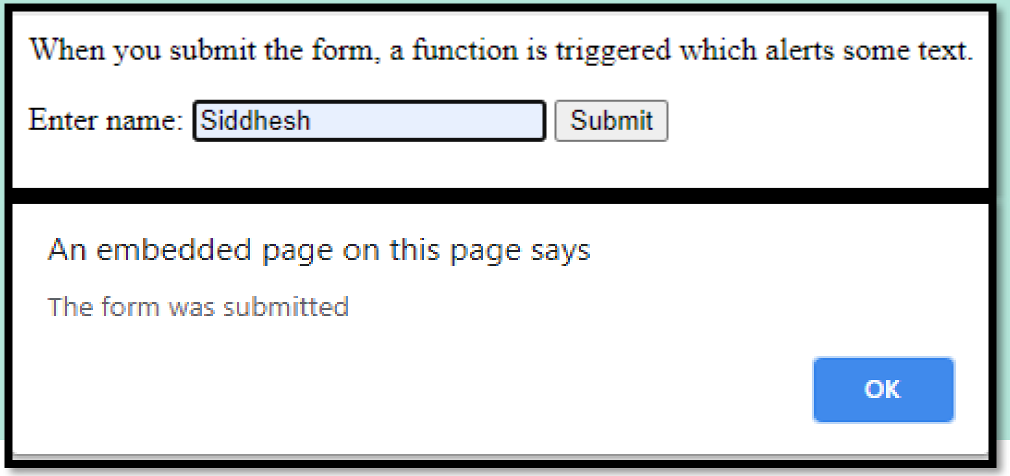
Name is **Siddhesh**

Password is **Vaidya**

Example 5: Generate alert after clicking on submit button

```
<html>
<body>
<p>When you submit the form, a function is triggered which alerts some text.</p>
<form action="" onsubmit="myFunction()">
Enter name: <input type="text" name="fname" value="">
<input type="submit" value="Submit">
</form>
<script>
function myFunction()
{
alert("The form was submitted");
}
</script>
</body>
</html>
```

Output


When you submit the form, a function is triggered which alerts some text.

Enter name: Siddhesh

An embedded page on this page says

The form was submitted

Button

- Button is created by using following code:

```
<form method = “GET” action = “”>
<input type = “button” name = “MyButton” value = “Click” onclick = “msg()”>
<form>
```

A Button object also represents an HTML <button> element which is specified as follows:

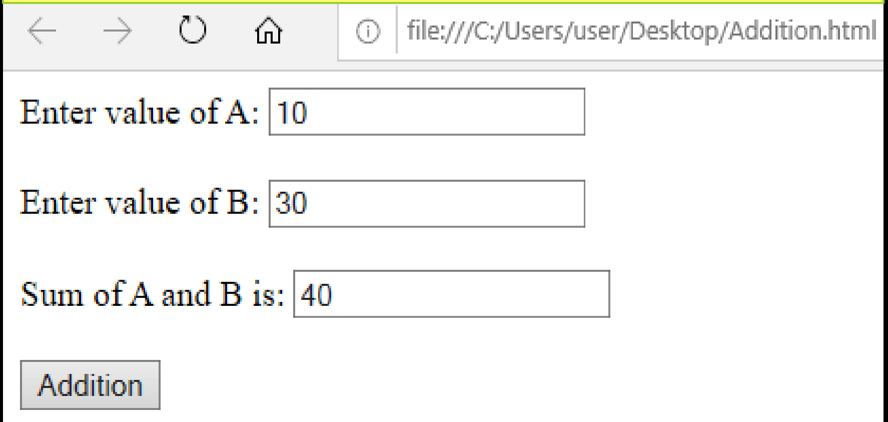
```
<button name = “btn” value = “MyButton” onclick = “msg()”>
```

- Inside a <button> element you can put content, like text or images. But, this is not the case with the buttons created with <input> tag.

Example

```
<html>

<body>

Enter value of A: <input type = "text" id =
"txt1"> <br> <br>

Enter value of B: <input type = "text" id =
"txt2"> <br> <br>

Sum of A and B is: <input type = "text" id =
"txt3"> <br> <br>

<input type = "button" onclick = "add()" value
= "Addition">

<script>

function add()

{

var a =

document.getElementById("txt1").value

var b =

document.getElementById("txt2").value

var c = parseInt(a) + parseInt(b);

document.getElementById("txt3").value = c

}

</script>

</body>

</html>
```

Output


file:///C:/Users/user/Desktop/Addition.html

```
Enter value of A: 10

Enter value of B: 30

Sum of A and B is: 40

Addition
```

Text

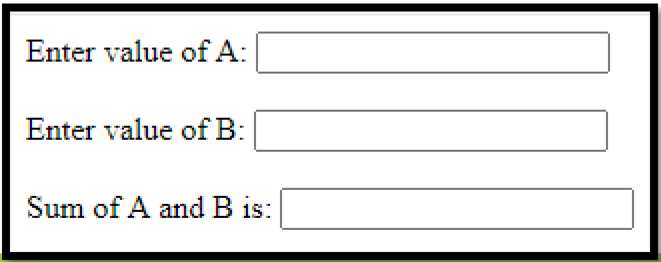
Input “text” is an object to enter a single line of text whose element will be part of form data. In HTML text is created by using following code:

<input type = “text” name = “textname” id = “textid” value = “”>

Example:

```
<body>
Enter value of A: <input type = "text" id = "txt1"> <br> <br>
Enter value of B: <input type = "text" id = "txt2"> <br> <br>
Sum of A and B is: <input type = "text" id = "txt3">
</body>
```

Output



Enter value of A:

Enter value of B:

Sum of A and B is:

Textarea

- A TextArea object represents an HTML <textarea> elements that creates a multiline text input field, often with an HTML form.
- The initial content of the text area is specified as plain text between <textarea> and </textarea> tags.
- You can query and set the displayed text with the value property.
- The <textarea> tag indicates a form field where the user can enter a large amount of text.

- Attributes of <textarea> tag are:
  - **name:** It sets the name of the form field
  - **cols and rows:** Cols indicates how many characters wide the TextArea should be.
    Rows indicates how many rows should be present in text area.
  - **wrap:** It describes how the text area wraps at the end of lines.
    - **Soft:** It wraps long lines in the TextArea for easy editing. It does not return carriage to the server.
    - **Hard:** It looks similar to soft, but it sends the carriage return to the server.
    - **Off:** It does not wrap at all. It just displays and sends the text exactly as typed in.

Textarea

Use:

1. name:

<textarea name = “comments”></textarea>

2. cols and rows:

<textarea name = “comments” cols = 30 rows = 7></textarea>

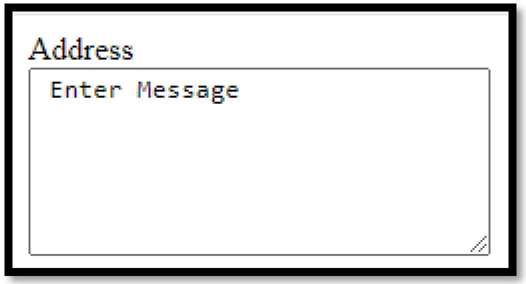
3. wrap:

<textarea name = “comments” wrap = soft></textarea>

Example

```
<body>
Address <br>
<textarea name = "message" cols = "30" rows = “7"> Enter Message </textarea>
</body>
```

Output


Address

Enter Message

Checkbox

- Radio and checkboxes are an integral part of forms that allows user to make a selection with a simple click of the mouse.
- You can prompt users for specific input by using list of checkboxes. More than one checkboxes can be checked by the user according to the information.
- Syntax for creating checkbox is:

```
<input type="checkbox" id="myCheck" onclick="myFunction()">
```

- A checkbox can have only two states:
    1. Checked
    2. Unchecked

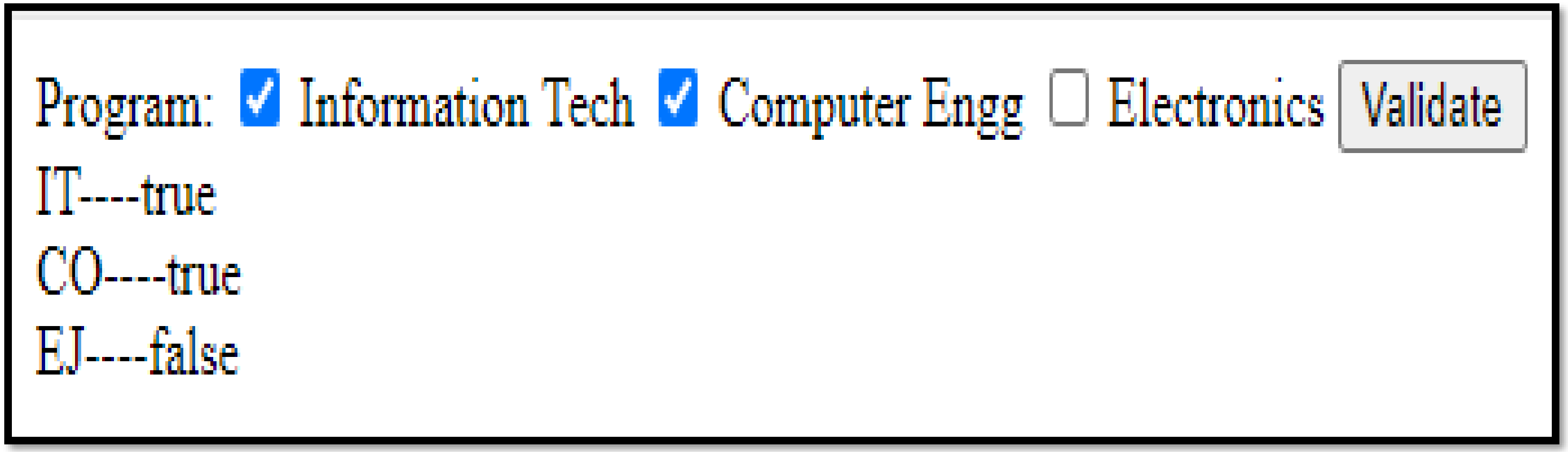
Example

Vidyalankar Polytechnic

```
<html> <body>
<div>
Program:
   <input type="checkbox"
name="program" id="it" value="IT">
   <label for="it">Information
Tech</label>
   <input type="checkbox"
name="program" id="co" value="CO"
checked>
   <label for="co">Computer
Engg</label>
   <input type="checkbox"
name="program" id="ej"  value="EJ">
   <label for="ej">Electronics</label>
   <button
onclick="javascript:validate();">Validat
e</button>
</div>
<div id="status"> </div>
```

```
<script>
function validate()
{
   var elements =
document.getElementsByName("program");
   var statusText = " ";
   for (var index=0;index <
elements.length;index++)
  {
  statusText = statusText +
elements[index].value+"-
"+elements[index].checked+"<br>";
   }

document.getElementById("status").innerHTML
= statusText;
}
</script>  </body> </html>
```

Output



```
IT----true
CO----true
EJ----false
```

Radiobutton

- ✓The radiobutton allows the user to choose one of a predefined set of options. You can define groups with the name property of the radio buttons.
- ✓Radio buttons with the same name belong to the same group. Radio buttons with different names belongs to the different groups. At most one radio button can be checked in a group.

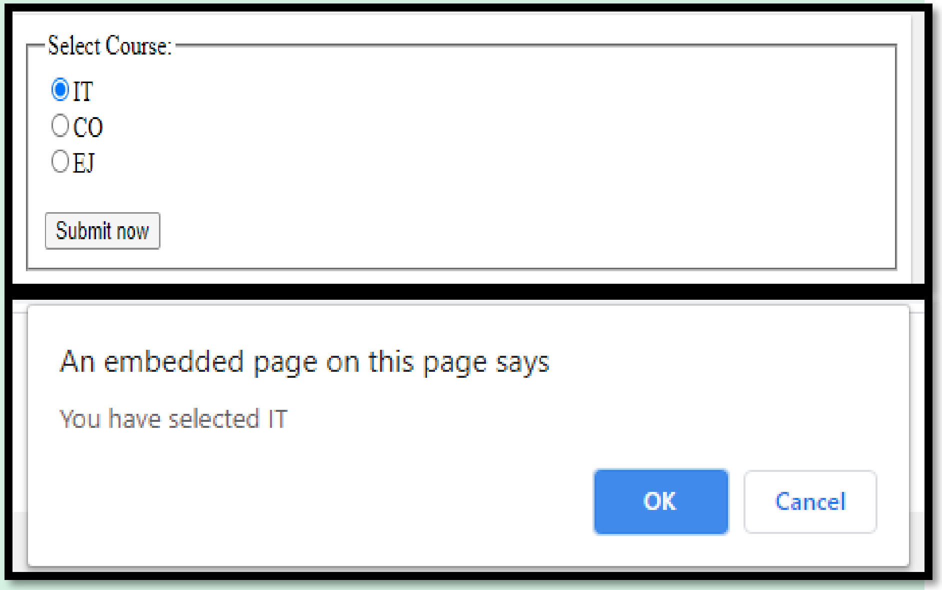
Syntax

```
<input type="radio" id="male" name="gender" value="male">
```

Example

```
<form method="post" action=" " onsubmit="return ValidateForm();">
<fieldset>
<legend>Select Course:</legend>
<input type="radio" name="br" value="IT" checked>IT<br>
<input type="radio" name="br" value="CO">CO<br>
<input type="radio" name="br" value="EJ">EJ<br><br>
<input type="submit" value="Submit now">
</fieldset>
</form>
<script type="text/javascript">
function ValidateForm()
{
var obj = document.getElementsByName("br");
for(var i = 0; i < obj.length; i++)
{
 if(obj[i].checked == true)
{
if(confirm("You have selected " + obj[i].value))
return true;
else
return false;
} } }  </script>
```

Output


Select Course:

● IT
○ CO
○ EJ

An embedded page on this page says

You have selected IT

Select

Form SELECT elements (<select>) within your form can be accessed and manipulated in JavaScript via the corresponding Select object.

To access a SELECT element in JavaScript, use the syntax:

document.myform.selectname //where myform and selectname are names of your form/element.

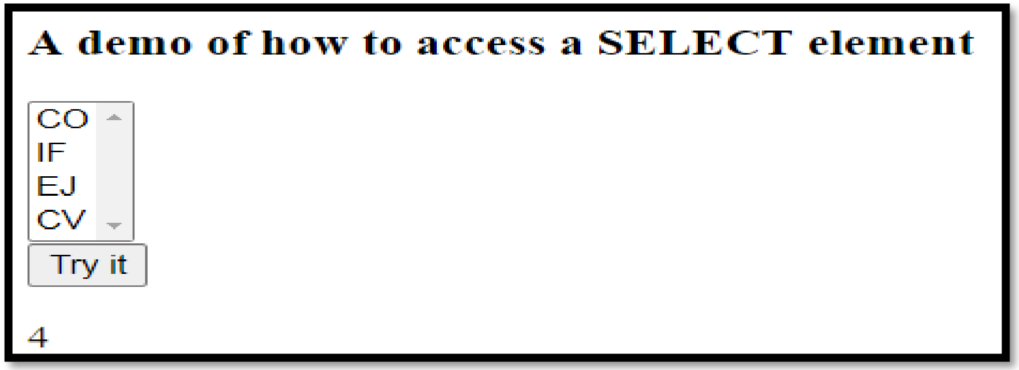
document.myform.elements[i] //where i is the position of the select element within form

document.getElementById("selectid") //where "selectid" is the ID of the SELECT element on the page

Example

```
<html>
<body>
<h3>A demo of how to access a SELECT element</h3>
<select id="mySelect" size="4">
<option>CO</option>
<option>IF</option>
<option>EJ</option>
<option>CV</option>
</select> <br>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction()
{
var x =
document.getElementById("mySelect").length;
document.getElementById("demo").innerHTML = x;
}
</script>  </body>  </html>
```

Output



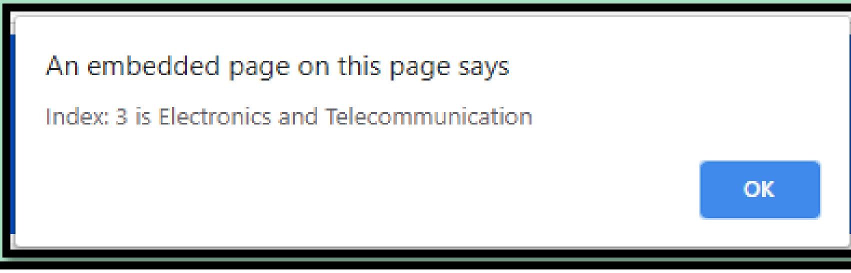
A demo of how to access a SELECT element

```
CO
IF
EJ
CV
Try it

4
```

Example

```
<html>

<body>

Select a fruit and click the button:

<select id="mySelect">s

<option>Information Technology</option>

  <option>Computer Engg</option>

  <option>Civil Engg</option>

  <option>Electronics and Telecommunication</option>

</select>

<button type="button" onclick="myFunction()">

Display index</button>
```

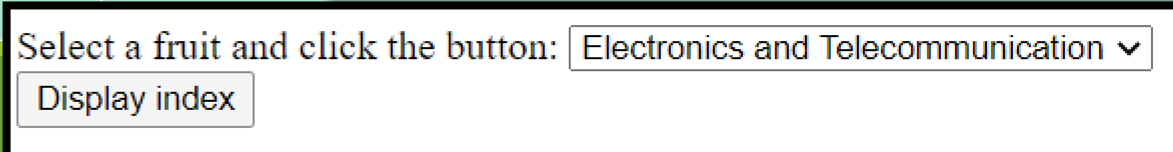
```
<script>

function myFunction()

{

var x =

document.getElementById("mySelect").selectedIndex

;

var y =

document.getElementById("mySelect").options;

alert("Index: " + y[x].index + " is " + y[x].text);

}

</script>

</body>

</html>
```



An embedded page on this page says

Index: 3 is Electronics and Telecommunication

Output



Select a fruit and click the button: Electronics and Telecommunication

Form Events

- ✓One of the primary ways in which JavaScript is executed on a Web Pages is through events. An event is a specific circumstance that is monitored by JavaScript and that the script can respond in some way.
- ✓JavaScript enabled web pages are typically event driven. Events are actions that occur on the webpage.
- ✓Events are signals generated when specific action occurs.
- ✓In order to make a web page more interactive, the script needs to be able to access the contents of the document and know when the user is interacting with it.

- The kinds of events that might occur are due to:
  1. A document loading
  2. The user clicking a mouse button
  3. The browser screen changing size

Following are the HTML intrinsic event attributes

onload

onunload

onclick

ondblclick

onmousedown

onmouseup

onmouseover

onmousemove

onmouseout

onfocus

onblur

onkeypress

onkeydown

onkeyup

onsubmit

onreset

onselect

onchange

Event Handler

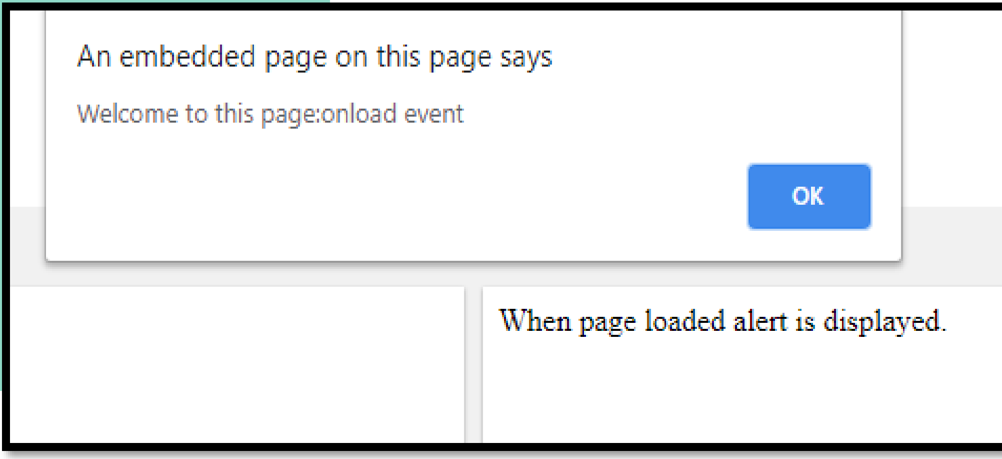
- When a function is assigned to an event handler, that function is executed when that event occurs.
- A handler that is assigned from a script used.
  - Syntax: '[element].[event] = [function];'

  where,
  - **[element]** is a page element.
  - **[event]** is the name of the selected event.
  - **[function]** is the name of the function that occurs when the event takes place.

Example: document.onclick = clickHandler();

Note: This handler will cause the function clickHandler() to be executed whenever the user clicks the mouse anywhere on the screen.

```
<html>
<head>
<script type="text/javascript">
 function message()
{
alert("Welcome to this page:onload event");
}
</script>
</head>
<body onload="message()">
When page loaded alert is displayed.
</body>
</html>
```


An embedded page on this page says

Welcome to this page:onload event

When page loaded alert is displayed.

Mouse Events

- JavaScript’s interaction with HTML is handled through events that occur when the user or browser manipulates a page.
- When the page loads, that is an event. When the user clicks a button, that click too, too, is an event.
- Another example of events are like click mouse button, pressing any key, closing window , resizing window etc.
- **Mouse related events can be categorized as following ways:**
  - 1. Simple Events
  - 2. Complex Events

Simple Events

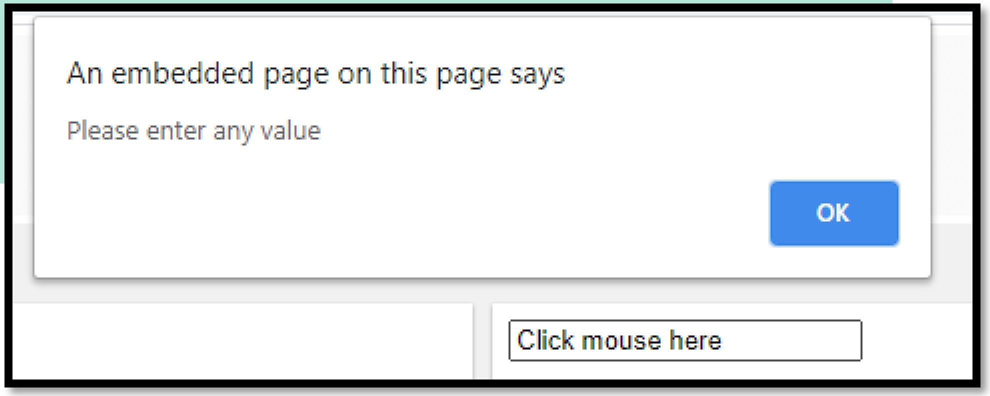
1. **Mousedown:** Triggered by an element when a mouse button is pressed down over it.
2. **Mouseup:** Triggered by an element when a mouse button is released over it.
3. **Mouseover:** Triggered by an element when a mouse comes over it.
4. **Mouseout:** Triggered by an element when a mouse goes out of it.
5. **Mousemove:** Triggered by an element on every mouse move over it.

Complex Events

1. **Click:** Triggered by a mouse click: mousedown and then mouseup over an element.
2. **Contextmenu:** Triggered by a right button mouse click over an element.
3. **Dblclick:** Triggered by two clicks within a short time over an element.

Example

```
<html>
<head>
<title>Events</title>
<script>
function react()
{
alert("Please enter any value");
}
</script></head>
<body>
<form name = "myform" action = "" method = "post" onmousedown = "react()">
<input value = "Click mouse here">
</form>
</body>
</html>
```


An embedded page on this page says

Please enter any value

Event Object & Event Listeners

- An **event object** provides information about the element that has generated an event and enables to retrieve it in the script. An **event listener** is essentially a mechanism that instructs an object to respond to a particular kind of event.
- In the JavaScript version of the DOM, an event listener is simply a function that takes a single argument that is an instance of Event. A call to the addEventListener() method on a node object associates an event listener with a type of event occurring on that node.

- The syntax of **addEventListener()** method is as follows:

```
document.addEventListener(event, function, useCapture)
```

Where,

- The first parameter is case insensitive string which specifies the type of event like “click” or “mousedown”.
- The second parameter is the function we want to call when the event occurs.
- The third parameter is a boolean value specifying whether to use event capturing or event bubbling. This parameter is optional.

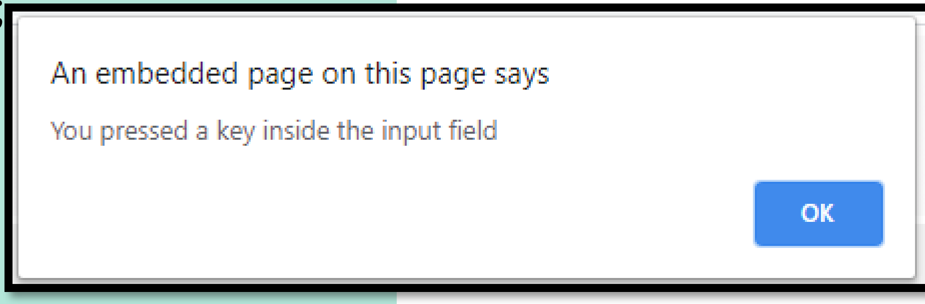
Key Events

- Key related events can be categorized as following ways:
  1. **Keydown:** A key is pressed down.
  2. **Keypress:** A character key is pressed.
  3. **Keyup:** A key is released.

- There is fundamental difference between keypress and keydown:
  1. **Keydown** triggers on any key press and gives scan code.
  2. **Keypress** triggers after keydown and gives char code, but it is guaranteed for character keys only.

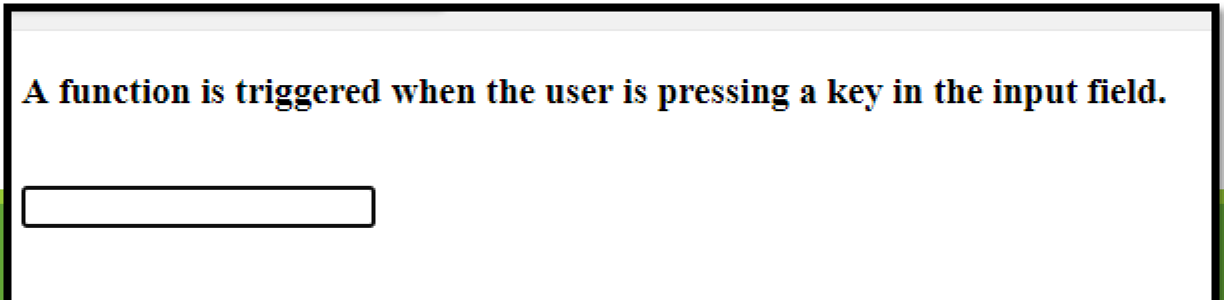
- The onkeypress event occurs when the user presses a key (on the keyboard). The order of events related to the onkeypress event:
  1. onkeydown
  2. onkeypress
  3. onkeyup

```
<html>
<body>

<h3>A function is triggered when the user is pressing a key in the
input field.</h3>

<br>
<input type="text" onkeypress="myFunction()">

<script>
function myFunction()
{
  alert("You pressed a key inside the input field");
}
</script>

</body>
</html>
```



An embedded page on this page says

You pressed a key inside the input field



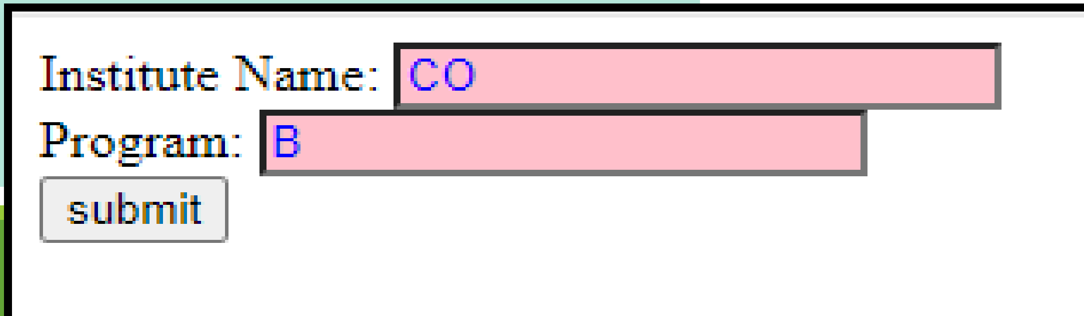
A function is triggered when the user is pressing a key in the input field.

Changing Attribute Values Dynamically

- ✓ JavaScript provides us facility to change an attribute of an element by assigning a new value to the attribute at runtime.
- ✓We can write the code of change in attribute value in a user defined function and can call that function when an appropriate event occurs.
- ✓The change in any attribute value can be reflected to the user by highlighting the value or text by some colour.
- ✓The ***onchange*** event is associated with many elements (<input>, <select>) of a form object and helpful to make call to a function where the change of attribute value code is written.
- ✓The onchange event occurs when the value of an element has been changed or the cursor is moved away from the element. For radiobuttons and checkboxes, the onchange event occurs when the checked state has been changed

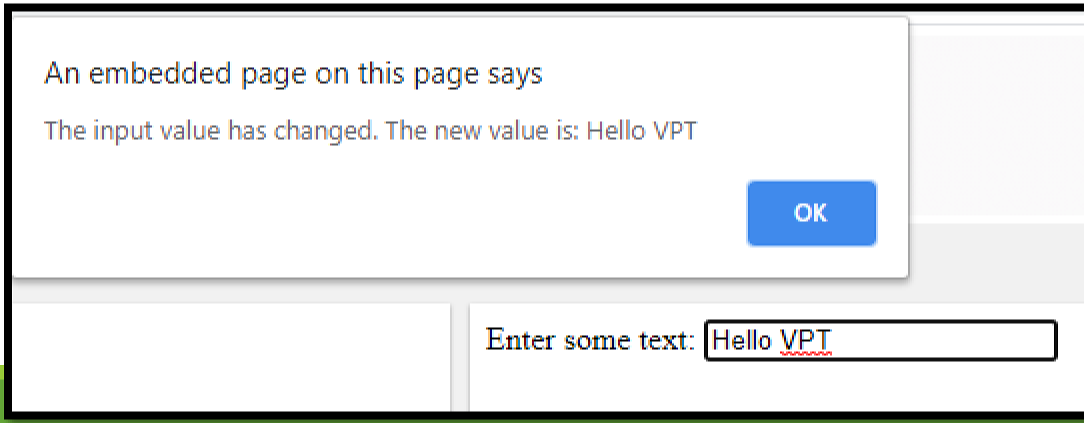
Example 1

```
<html>
<script type="text/javascript">
function highlight(x)
{
x.style.color="blue";
x.style.backgroundColor="pink";
}
</script>
<body>
<form name="myform" action=" " method="post">
Institute Name:
<input type="text" name="iname" onchange="highlight(this)"/>
<br>
Program:
<input type="text"  name="infotech" onchange="highlight(this)"/>
<br>
<input type="submit" value="submit" name="submit">
</form>
</body>
</html>
```


Institute Name: CO
Program: B

Example 2

```
<html>
<body>
Enter some text:
<input type="text" name="txt" value="Hello“ onchange="myFunction(this.value)">
<script>
function myFunction(val)
{
alert("The input value has changed. The new value is: " + val);
}
</script>
</body>
</html>
```


An embedded page on this page says

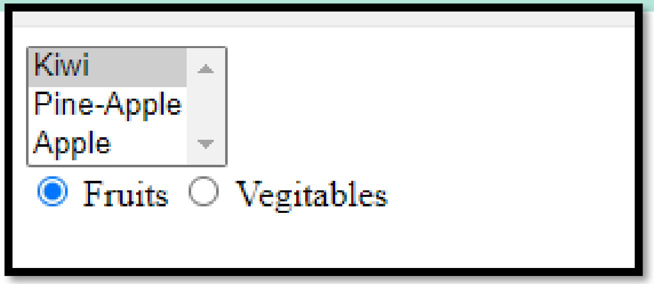
The input value has changed. The new value is: Hello VPT

Changing Option List Dynamically

✓We can change options in the option list at runtime by writing the code in function.

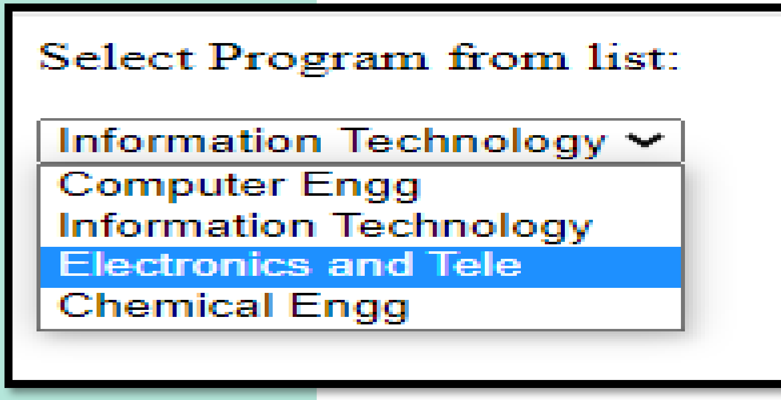
✓The purpose of an option list is to present a user two or more items for choice. Elements in option list are usually set when the option list is created. However, you can change the content of an option list at runtime by using a JavaScript function.

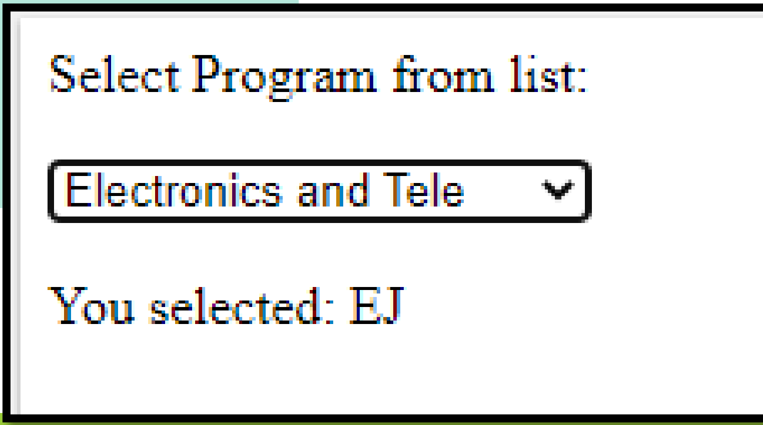
```
<html>
<body>
<html>
<script type="text/javascript">
function modifyList(x)
{
with(document.forms.myform)
{
if(x ==1)
{
optionList[0].text="Kiwi";
optionList[0].value=1;
optionList[1].text="Pine-Apple ";
optionList[1].value=2;
optionList[2].text="Apple";
optionList[2].value=3;
}
if(x == 2)
{
optionList[0].text="Tomato";
optionList[0].value=1;
optionList[1].text="Onion ";
optionList[1].value=2;
optionList[2].text="Cabbage ";
optionList[2].value=3;
}}}  </script>
<body>
<form name="myform" action=" " method="post">
<select name="optionList" size="3">
<option value=1>Kiwi
<option value=1>Pine-Apple
<option value=1>Apple
</select><br>
<input type="radio" name="grp1" value=1 checked="true"
onclick="modifyList(this.value)"> Fruits
<input type="radio" name="grp1" value=2
onclick="modifyList(this.value)"> Vegitables       </form>
</body>
</html>
```



```
Kiwi
Pine-Apple
Apple
● Fruits ○ Vegitables
```

```
<html>
<body>
<p>Select Program from list:</p>
<select id="mySelect" onchange="myFunction()">
<option value="CO">Computer Engg</option>
<option value="IF">Information Technology</option>
<option value="EJ">Electronics and Tele</option>
<option value="CE">Chemical Engg</option>
</select>
<p id="demo"></p>
<script>
function myFunction()
{
var x = document.getElementById("mySelect").value;
document.getElementById("demo").innerHTML = "You selected: " + x;
}
</script>
</body>
</html>
```


Select Program from list:

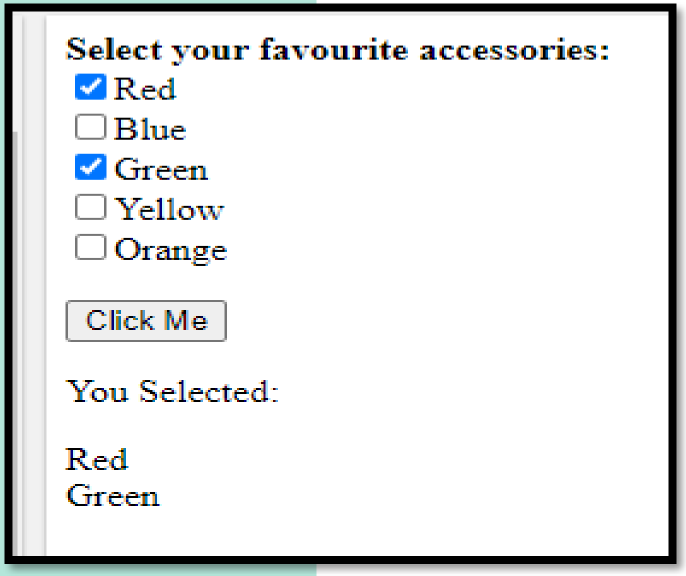

Select Program from list:

You selected: EJ

Evaluating Checkbox Selection

- A checkbox is created by using the input element with the type=”checkbox” attribute-value pair.
- A checkbox in a form has only two states (checked or un-checked). Checkboxes can be grouped together under a common name.

```
<html>
<script>
function printChecked()
{
var items=document.getElementsByName('check_print');
var selectedItems="";
for(var i=0; i<items.length; i++)
{
if(items[i].type=='checkbox' && items[i].checked==true)
selectedItems+=items[i].value+"<br>";
}
document.getElementById("y").innerHTML =selectedItems;
}
</script>
<body> <b>Select your favourite accessories: </b><br>
<input type="checkbox" name="check_print" value="Red">Red<br>
<input type="checkbox" name="check_print" value="Blue">Blue<br>
<input type="checkbox" name="check_print" value="Green">Green<br>
<input type="checkbox" name="check_print" value="Yellow">Yellow<br>
<input type="checkbox" name="check_print" value="Orange">Orange<br>
<p><input type="button" onclick='printChecked()' value="Click Me"/></p>
You Selected:
<p id="y"></p> </body> </html>
```



Select your favourite accessories:
- [✓ Red]
- Blue
- [✓ Green]
- Yellow
- Orange

You Selected:

Red
Green

Changing Labels Dynamically

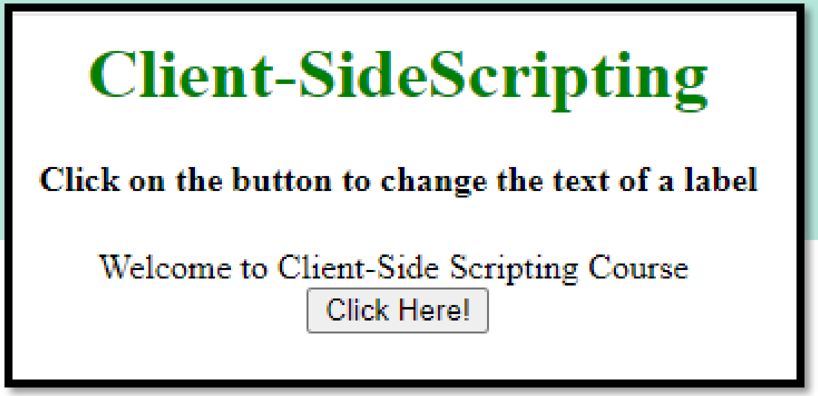
Label

The <label>tag is used to provide a usability improvement for mouse users i.e, if a user clicks on the text within the <label> element, it toggles the control.

Approach:

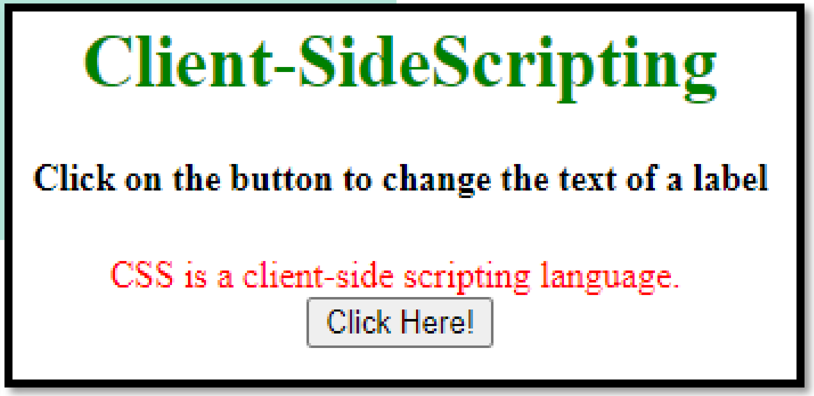
- Create a label element and assign an id to that element.
- Define a button that is used to call a function. It acts as a switch to change the text in the label element.
- Define a javaScript function, that will update the label text.
- Use the innerHTML property to change the text inside the label. The innerHTML property sets or returns the HTML content of an element.

```
<html>
<head>
</head>
<body style="text-align:center;">
<h1 style="color:green;"> Client-SideScripting </h1>
<h4> Click on the button to change the text of a label </h4>
<label id = "aaa"> Welcome to Client-Side Scripting Course </label>
<br>
<button onclick="change_L()"> Click Here! </button>
<script>
function change_L()
{
document.getElementById('aaa').innerHTML = "CSS is a client-side
scripting language.";
document.getElementById('aaa').style.color = "red";
}
</script>
</body>
</html>
```


Client-SideScripting

Click on the button to change the text of a label

Welcome to Client-Side Scripting Course


Client-SideScripting

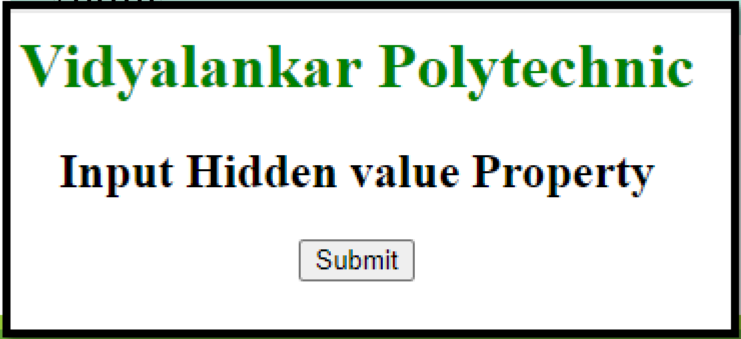
Click on the button to change the text of a label

CSS is a client-side scripting language.

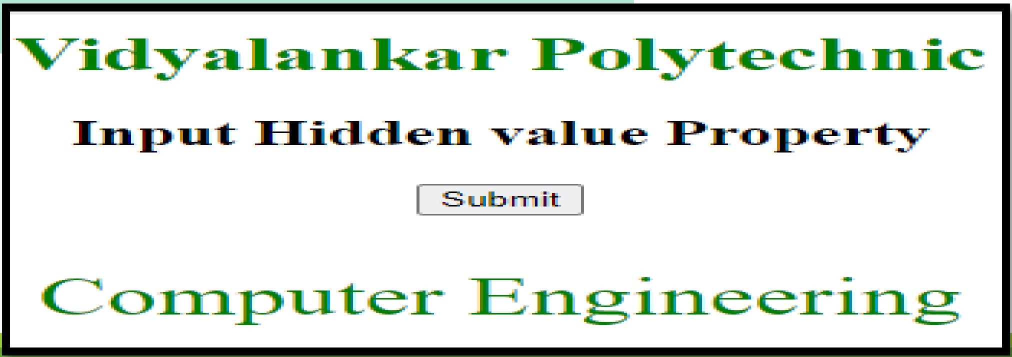
Manipulating Form Elements

Javascript make it possible with help of hidden element which is similar to any html element except it does not appear on screen.

```
<html>
<body style="text-align:center;">
<h1 style="color:green;"> Vidyalankar Polytechnic </h1>
<h2>Input Hidden value Property</h2>
<input type="hidden" id="it" value="Computer Engineering">
<button onclick="disp_hidden_Text()"> Submit </button>
<p id="demo" style="color:green;font-size:35px;"></p>
<script>
function disp_hidden_Text()
{
var x = document.getElementById("it").value;
document.getElementById("demo").innerHTML = x;
}
</script>
</body>
</html>
```


Vidyalankar Polytechnic

Input Hidden value Property


Vidyalankar Polytechnic

Input Hidden value Property

Computer Engineering

Intrinsic JavaScript Functions

The **HTML <input> src Attribute** is used to *specify the URL of the image to be used as a submit Button*. This attribute is not used with **<input type="image">**

Syntax:

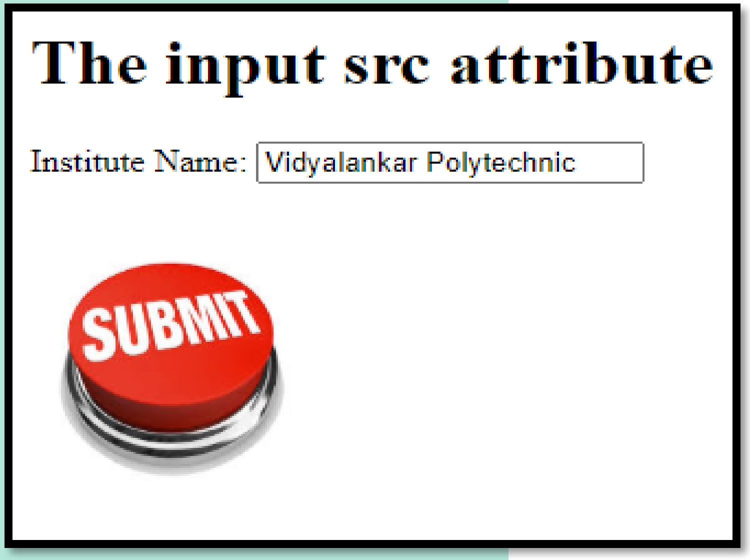
<input src="URL">

Attribute Values: It contains a single value URL which specifies the link of source image. There are two types of URL link which are listed below:

Absolute URL: It points to another webpage.

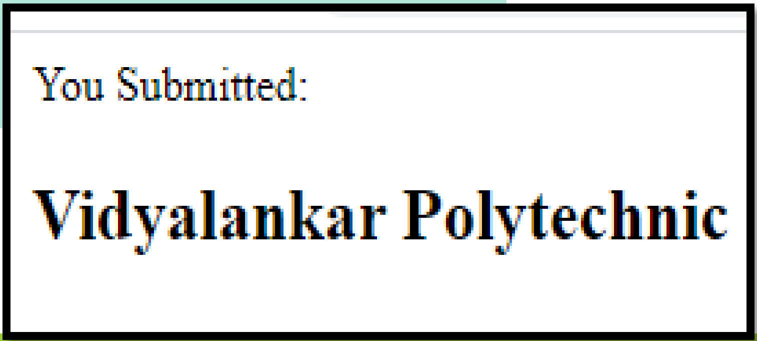
Relative URL: It points to other files of the same web page.

```
<html>
<body>
<h1>The input src attribute</h1>
<form action=" ">
<label for="fname">Institute Name:</label>
<input type="text" id="name" name="name"><br><br>
<input type="image" src="Submit.jpg" alt="Submit" width="130" height="150"
onclick="myFunction()">
</form>
<p id="demo"></p>
<script>
function myFunction()
{
var x = document.getElementById("name").value;
document.write("You Submitted:<h2>" +x +"</h2>");
}
</script>
</body>
</html>
```



The input src attribute

Institute Name: Vidyalankar Polytechnic



You Submitted:

Vidyalankar Polytechnic

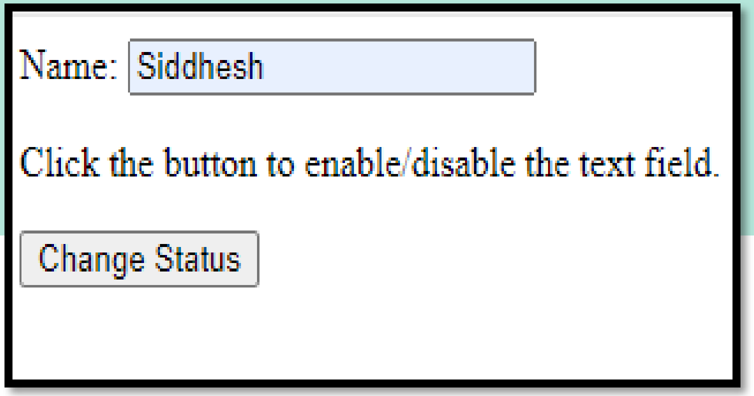
Disabling Elements

It is common to display a form with some elements disabled, which prevents the user from entering information into the element.

Vidyalankar Polytechnic

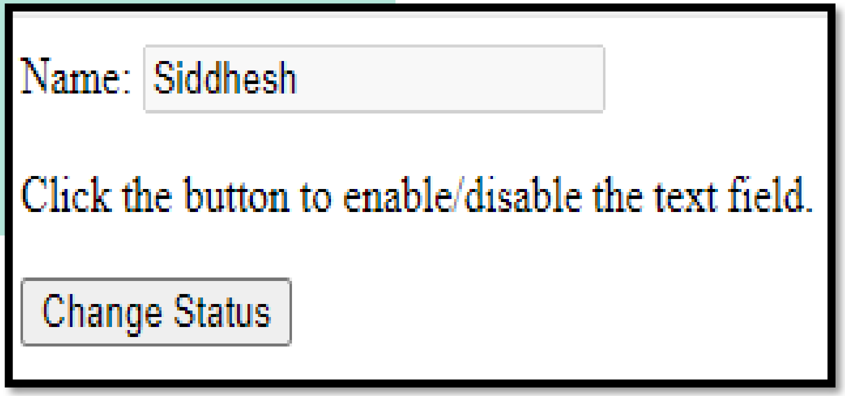
Method 1

```
<html>
<body>
Name: <input type="text" id="myText">
<p>Click the button to enable/disable the text field.</p>
<button onclick="myFunction()"> Change Status </button>
<script>
function myFunction()
{
var txt=document.getElementById("myText")
if ('disabled' in txt)
{
txt.disabled=!txt.disabled;
}
}
</script>
</body>
</html>
```



Name: Siddhesh

Click the button to enable/disable the text field.

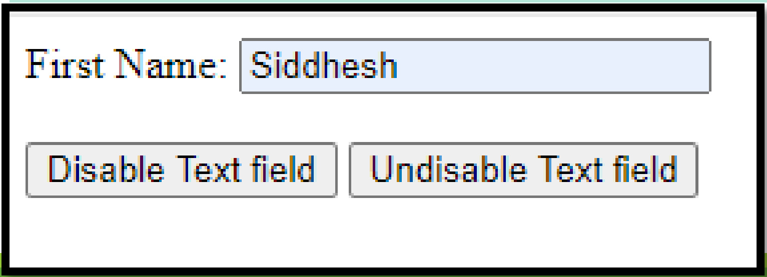


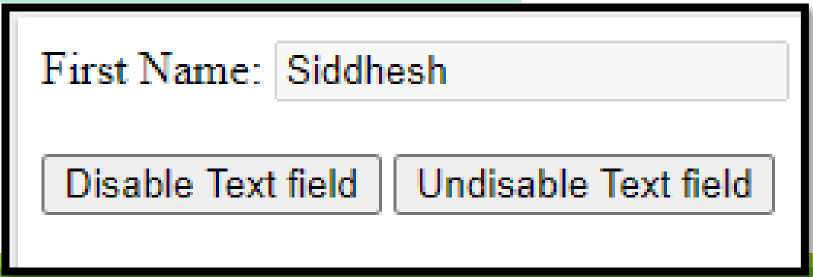
Name: Siddhesh

Click the button to enable/disable the text field.

Method 2

```
<html>
<body>
First Name: <input type="text" id="myText"><br><br>
<button onclick="disableTxt()">Disable Text field</button>
<button onclick="undisableTxt()">Undisable Text field</button>
<script>
function disableTxt()
{
document.getElementById("myText").disabled = true;
}
function undisableTxt()
{
document.getElementById("myText").disabled = false;
}
</script>
</body>
</html>
```


First Name: Siddhesh


First Name: Siddhesh

Read Only Elements

- The readOnly property sets or returns whether a text field is read-only, or not.
- A read-only field cannot be modified. However, a user can tab to it, highlight it, and copy the text from it.
- Set a text field to read-only:

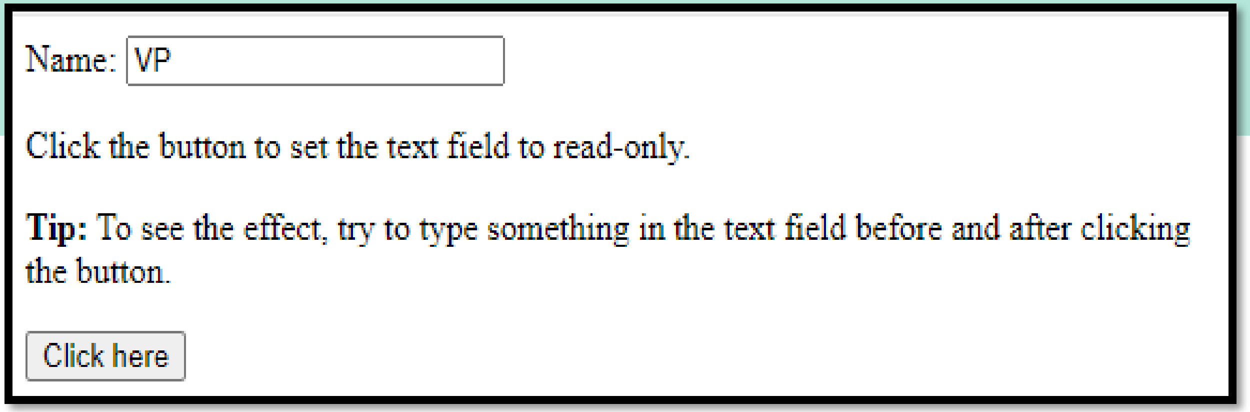
```
document.getElementById("myText").readOnly = true;
```

Syntax:

To return the readOnly property: *textObject*.readOnly

To Set the readOnly property: *textObject*.readOnly = true|false

```
<html>
<body>
Name: <input type="text" id="myText" value="VP">
<p>Click the button to set the text field to read-only.</p>
<p><strong>Tip:</strong> To see the effect, try to type something in the text field
before and after clicking the button.</p>
<button onclick="myFunction()">Click here </button>
<script>
function myFunction()
{
document.getElementById("myText").readOnly = true;
}
</script>
</body>
</html>
```



Name: VP

Click the button to set the text field to read-only.

Tip: To see the effect, try to type something in the text field before and after clicking the button.

Thank
You