UNIT III: Form & Event Handling

Building A Blocks Of A Form

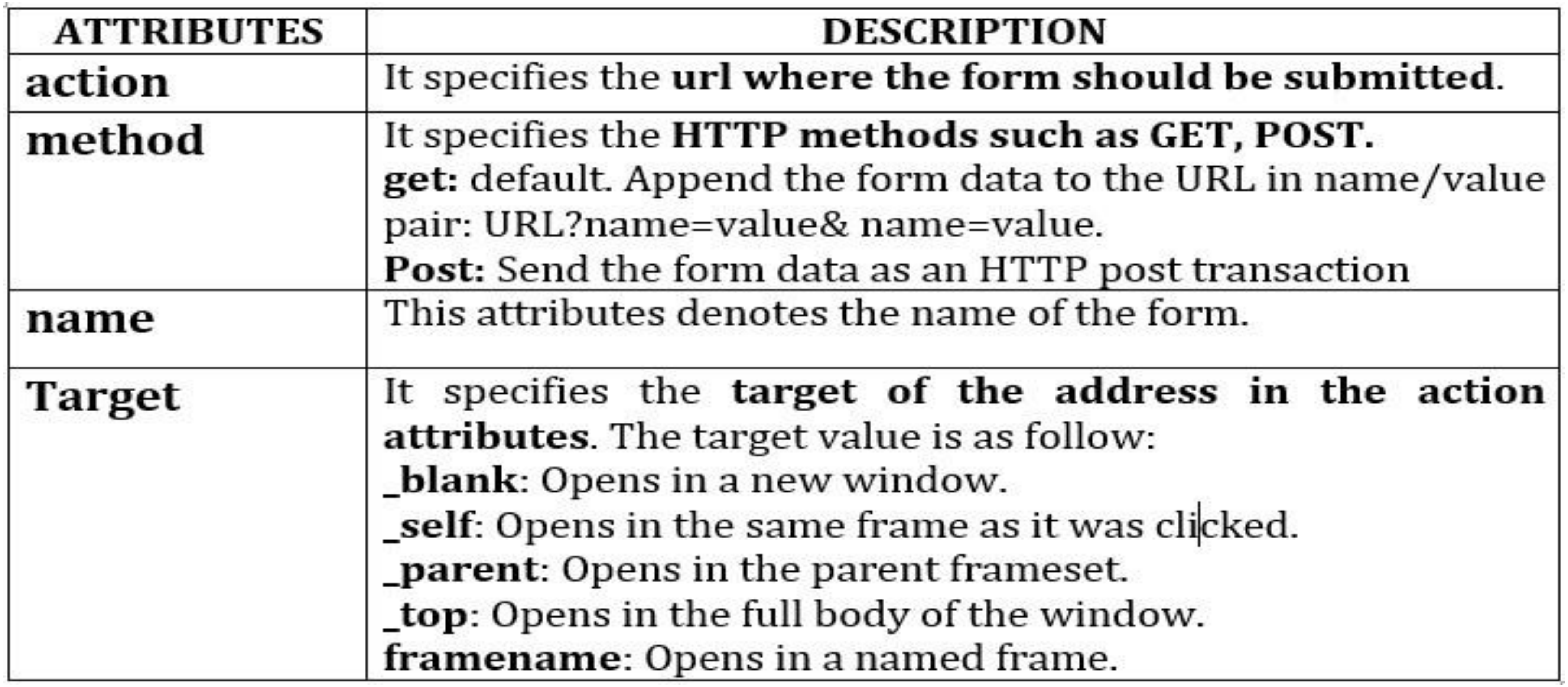
- Form is typical layout on the web page by which a **user can interact with the web page.**
- Typical components of forms are **text, text area, checkboxes, radio buttons & push buttons**. These components of form are also called as **form controls.**
- HTML allows us to place these form components on the web page.
- All these form contents in the **<form> tag**.
- The form has an attributes action which **gets executed user clicks a button on the form**.

Building A Blocks Of A Form

- Uses of form:

1. Forms are used to collect the information from customer for online registration.
2. Forms are used for online survey.
3. Forms are used for conduction online examination.
4. The information present in the forms is submitted to the server for further processing.

Properties & Methods Of Form

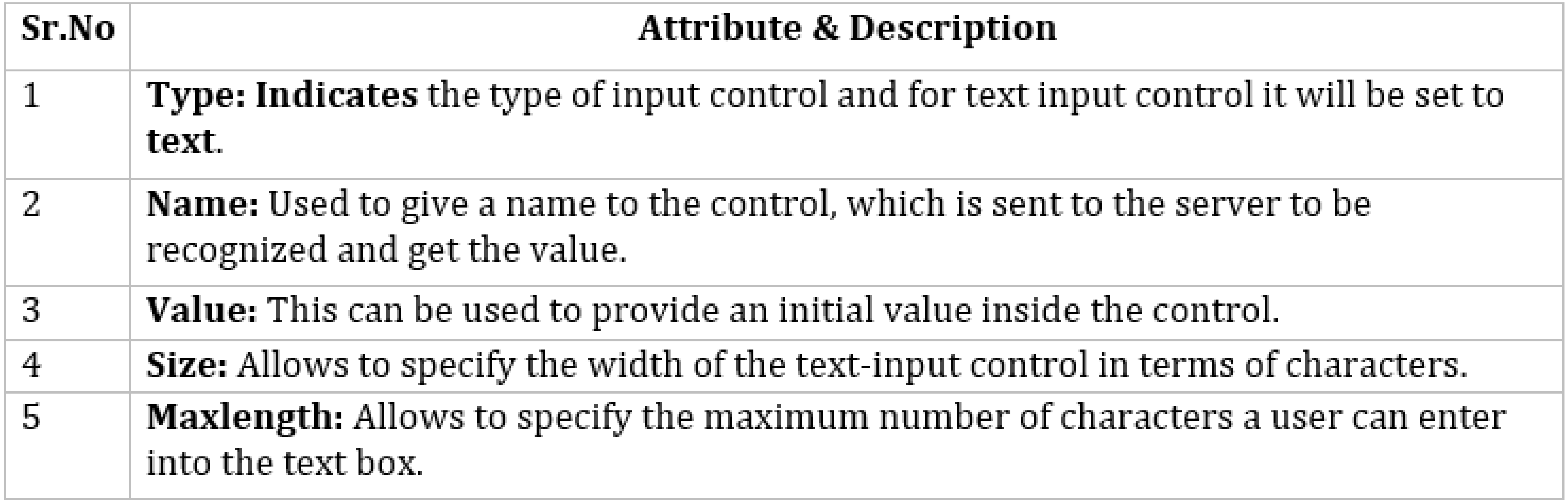


ATTRIBUTES	DESCRIPTION
action	It specifies the **url where the form should be submitted**.
method	It specifies the **HTTP methods such as GET, POST.** **get:** default. Append the form data to the URL in name/value pair: URL?name=value& name=value. **Post:** Send the form data as an HTTP post transaction
name	This attributes denotes the name of the form.
Target	It specifies the **target of the address in the action attributes**. The target value is as follow: **_blank**: Opens in a new window. **_self**: Opens in the same frame as it was clicked. **_parent**: Opens in the parent frameset. **_top**: Opens in the full body of the window. **framename**: Opens in a named frame.

```
<form name=“myform” action=“/myserverPage” method= “GET”
target=“_blank”>
</form>
```

Text

- Text is typically required to place one line text.
- This control is used for items that require only one line of user input is known as Single-line text input controls.
- They are created using HTML <input> tag.

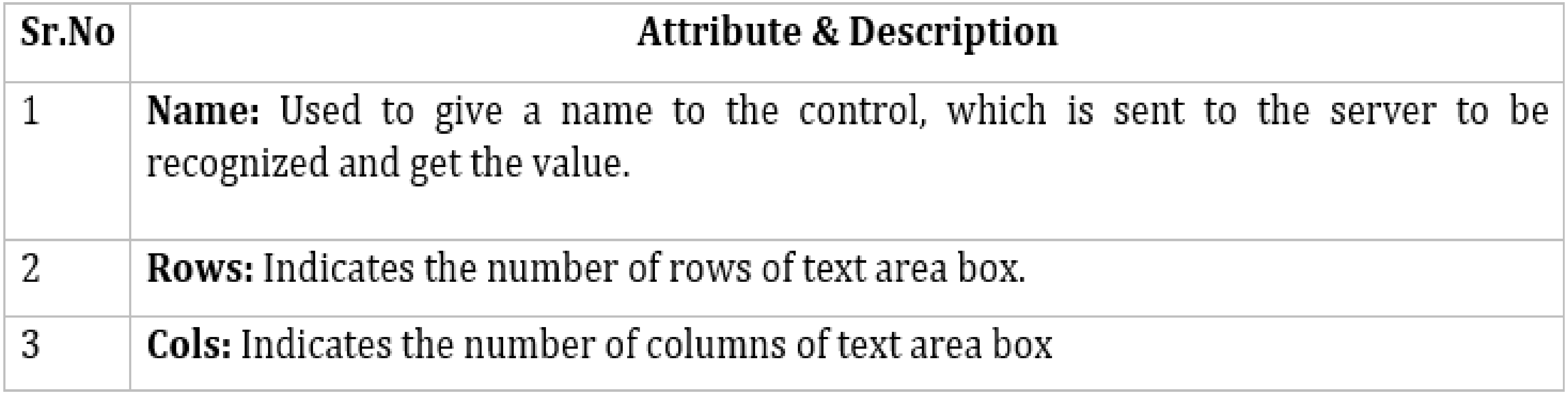


Sr.No	Attribute & Description
1	**Type: Indicates** the type of input control and for text input control it will be set to **text**.
2	**Name:** Used to give a name to the control, which is sent to the server to be recognized and get the value.
3	**Value:** This can be used to provide an initial value inside the control.
4	**Size:** Allows to specify the width of the text-input control in terms of characters.
5	**Maxlength:** Allows to specify the maximum number of characters a user can enter into the text box.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Text Input Control</title>
  </head>
  <body>
    <form >
      First name: <input type = "text" name = "first_name" />
      <br>
      Last name: <input type = "text" name = "last_name" />
    </form>
  </body>
</html>
```

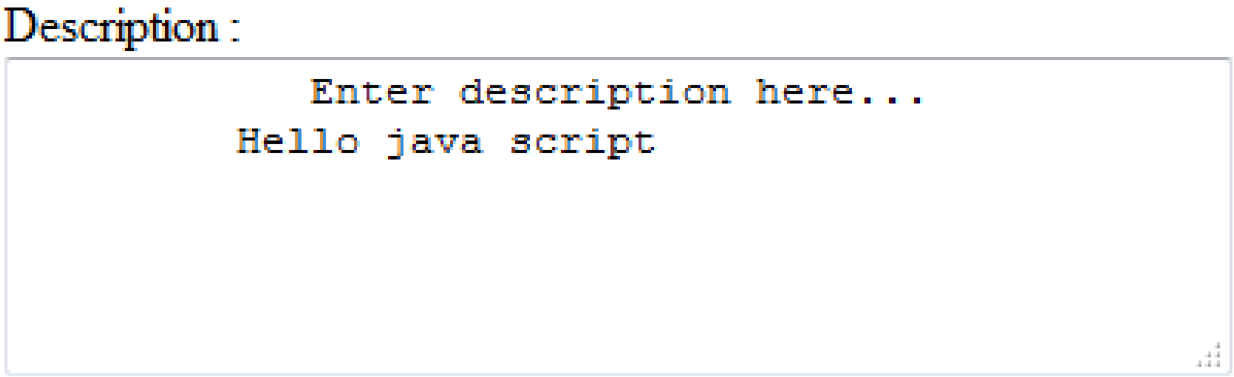
Text Area

- This is used when the user is required to give details that may be **longer than a single sentence**.
- Multi-line input controls are created using HTML **<textarea> tag**.



Sr.No	Attribute & Description
1	**Name:** Used to give a name to the control, which is sent to the server to be recognized and get the value.
2	**Rows:** Indicates the number of rows of text area box.
3	**Cols:** Indicates the number of columns of text area box

```
<!DOCTYPE html>
<html>
   <head>
     <title>Multiple-Line Input Control</title>
   </head>
   <body>
     <form>
        Description : <br />
        <textarea rows = "5" cols = "50" name = "description">
          Enter description here...
        </textarea>
     </form>
   </body>
</html>
```

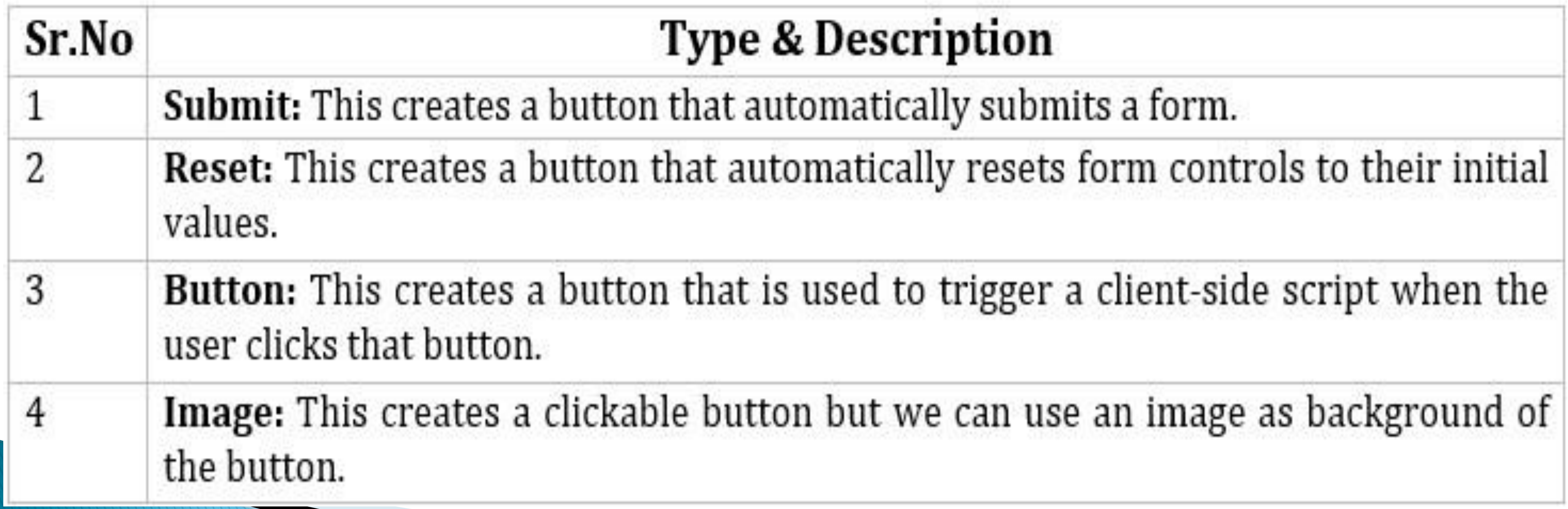


Description :

```
           Enter description here...
         Hello java script
```

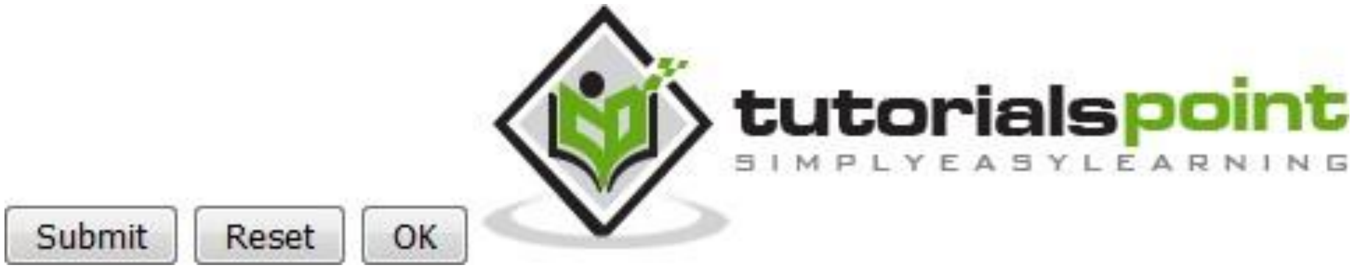
Button

- There are various ways in **HTML to create clickable buttons**.
- You can also **create a clickable button using <input>tag** by setting its type attribute to **button**.



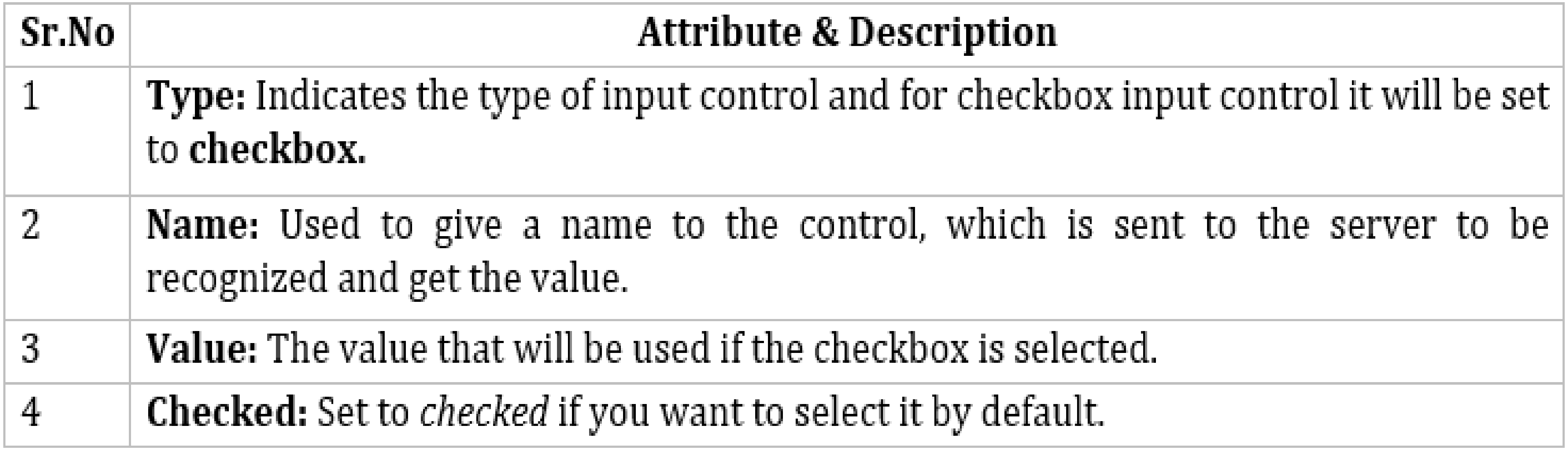
Sr.No	Type & Description
1	**Submit:** This creates a button that automatically submits a form.
2	**Reset:** This creates a button that automatically resets form controls to their initial values.
3	**Button:** This creates a button that is used to trigger a client-side script when the user clicks that button.
4	**Image:** This creates a clickable button but we can use an image as background of the button.

```
<!DOCTYPE html>
<html>
<body>
    <form>
      <input type = "submit" name = "submit" value = "Submit" />
      <input type = "reset" name = "reset"  value = "Reset" />
      <input type = "button" name = "ok" value = "OK" />
      <input type = "image" name = "imagebutton" src =
        "/html/images/logo.png" />
    </form>
  </body>
</html>
```



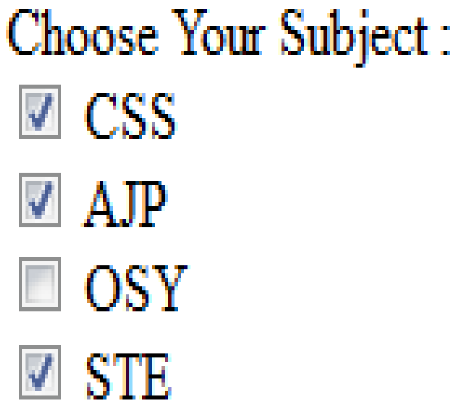
Checkbox

- Checkboxes are used when **more than one option is required to be selected**.
- They are also created using **HTML <input> tag** but type attribute is set to **checkbox.**



Sr.No	Attribute & Description
1	**Type:** Indicates the type of input control and for checkbox input control it will be set to **checkbox.**
2	**Name:** Used to give a name to the control, which is sent to the server to be recognized and get the value.
3	**Value:** The value that will be used if the checkbox is selected.
4	**Checked:** Set to *checked* if you want to select it by default.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Checkbox Control</title>
  </head>
  <body>
    <form>
      Choose Your Subject : <br />
       <input type = "checkbox" name = “Subject" value = "on"> CSS <br />
       <input type = "checkbox" name = " Subject " value = "on"> AJP <br />
       <input type = "checkbox" name = " Subject " value = "on"> OSY <br />
       <input type = "checkbox" name = " Subject " value = "on"> STE <br />
    </form>
  </body>
</html>
```



Choose Your Subject :

☑ CSS

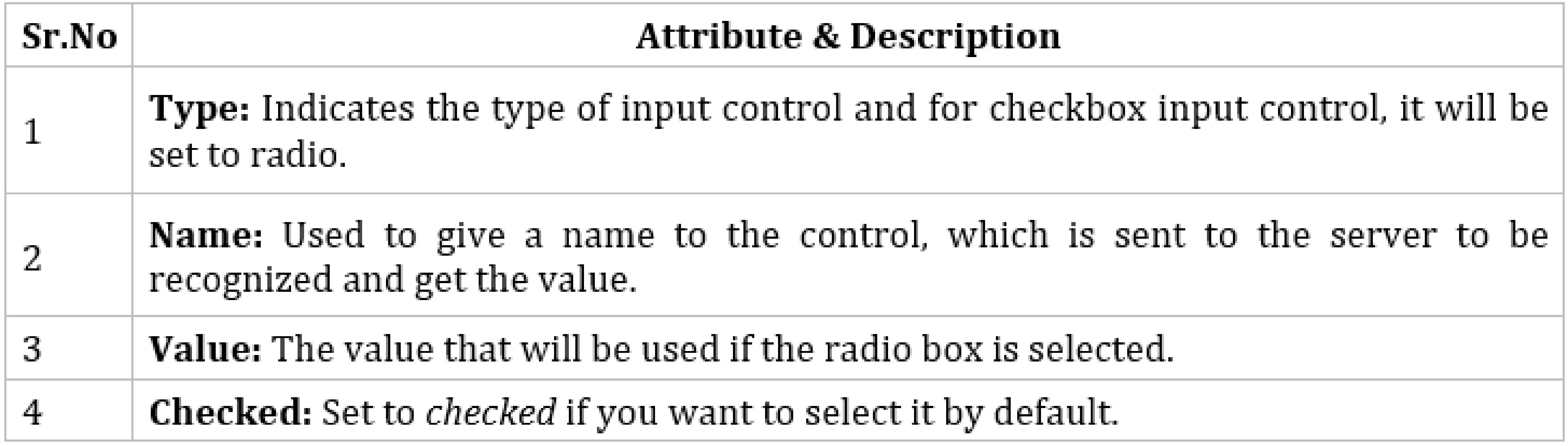
☑ AJP

□ OSY

☑ STE

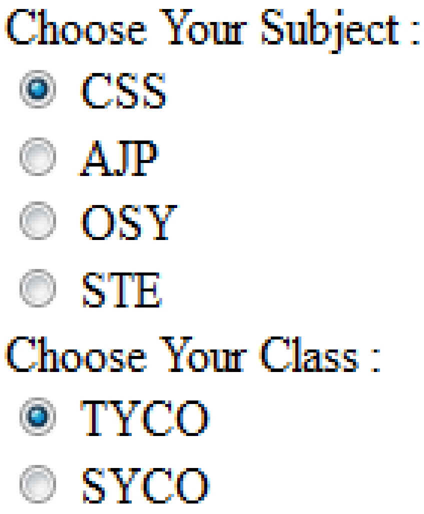
Radio Button

- Radio buttons are used when **out of many options, just one option is required to be selected.**
- They are also created using **HTML <input> tag but type attribute is set to radio.**



Sr.No	Attribute & Description
1	**Type:** Indicates the type of input control and for checkbox input control, it will be set to radio.
2	**Name:** Used to give a name to the control, which is sent to the server to be recognized and get the value.
3	**Value:** The value that will be used if the radio box is selected.
4	**Checked:** Set to *checked* if you want to select it by default.

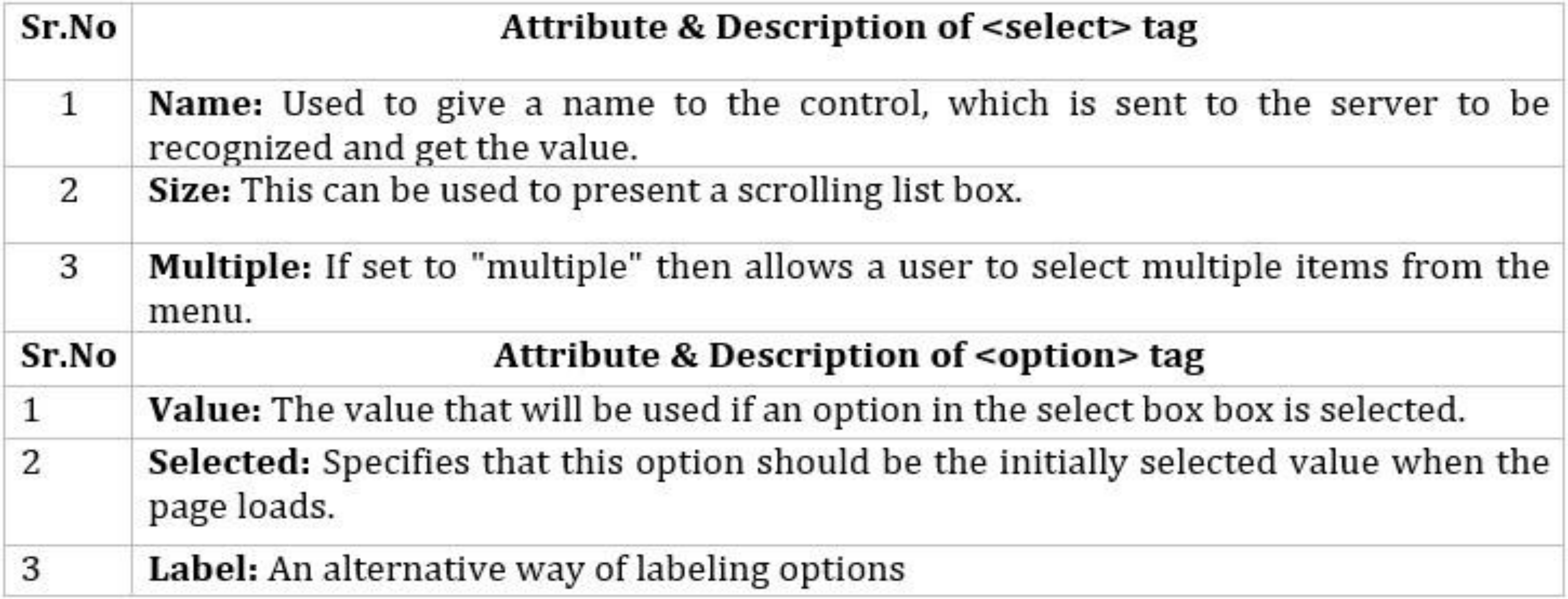
```
<!DOCTYPE html>
<html>
 <head>
   <title>Checkbox Control</title>
  </head>
  <body>
   <form>
    Choose Your Subject : <br />
     <input type = "radio" name = "Subject" value = "on"> CSS <br />
     <input type = "radio" name = "Subject" value = "on"> AJP <br />
     <input type = "radio" name = "Subject" value = "on"> OSY <br />
     <input type = "radio" name = "Subject" value = "on"> STE <br />
     Choose Your Class : <br />
     <input type = "radio" name = "Class" value = "on"> TYCO <br />
     <input type = "radio" name = "Class" value = "on"> SYCO <br />
   </form>
  </body>
</html>
```



Choose Your Subject :
● CSS
○ AJP
○ OSY
○ STE
Choose Your Class :
● TYCO
○ SYCO

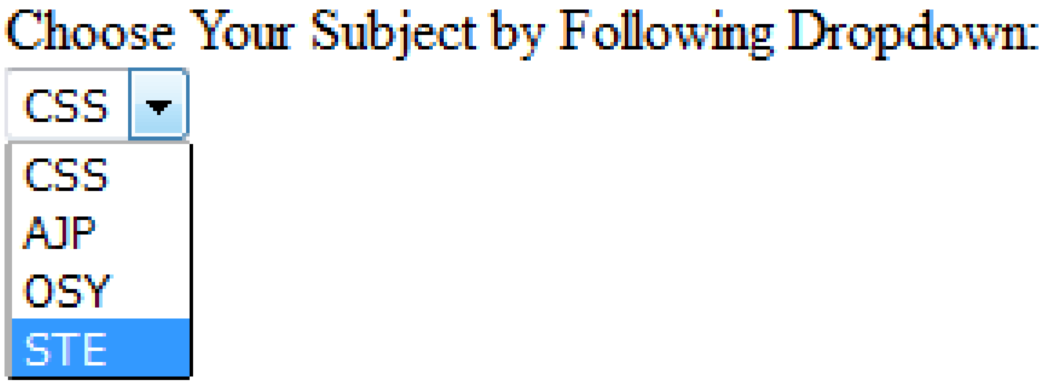
Select Elements

- A select box, also called **drop down box which provides option to list down various options** in the form of drop down list, from where a **user can select one or more options**.



Sr.No	Attribute & Description of <select> tag
1	**Name:** Used to give a name to the control, which is sent to the server to be recognized and get the value.
2	**Size:** This can be used to present a scrolling list box.
3	**Multiple:** If set to "multiple" then allows a user to select multiple items from the menu.
Sr.No	**Attribute & Description of <option> tag**
1	**Value:** The value that will be used if an option in the select box box is selected.
2	**Selected:** Specifies that this option should be the initially selected value when the page loads.
3	**Label:** An alternative way of labeling options

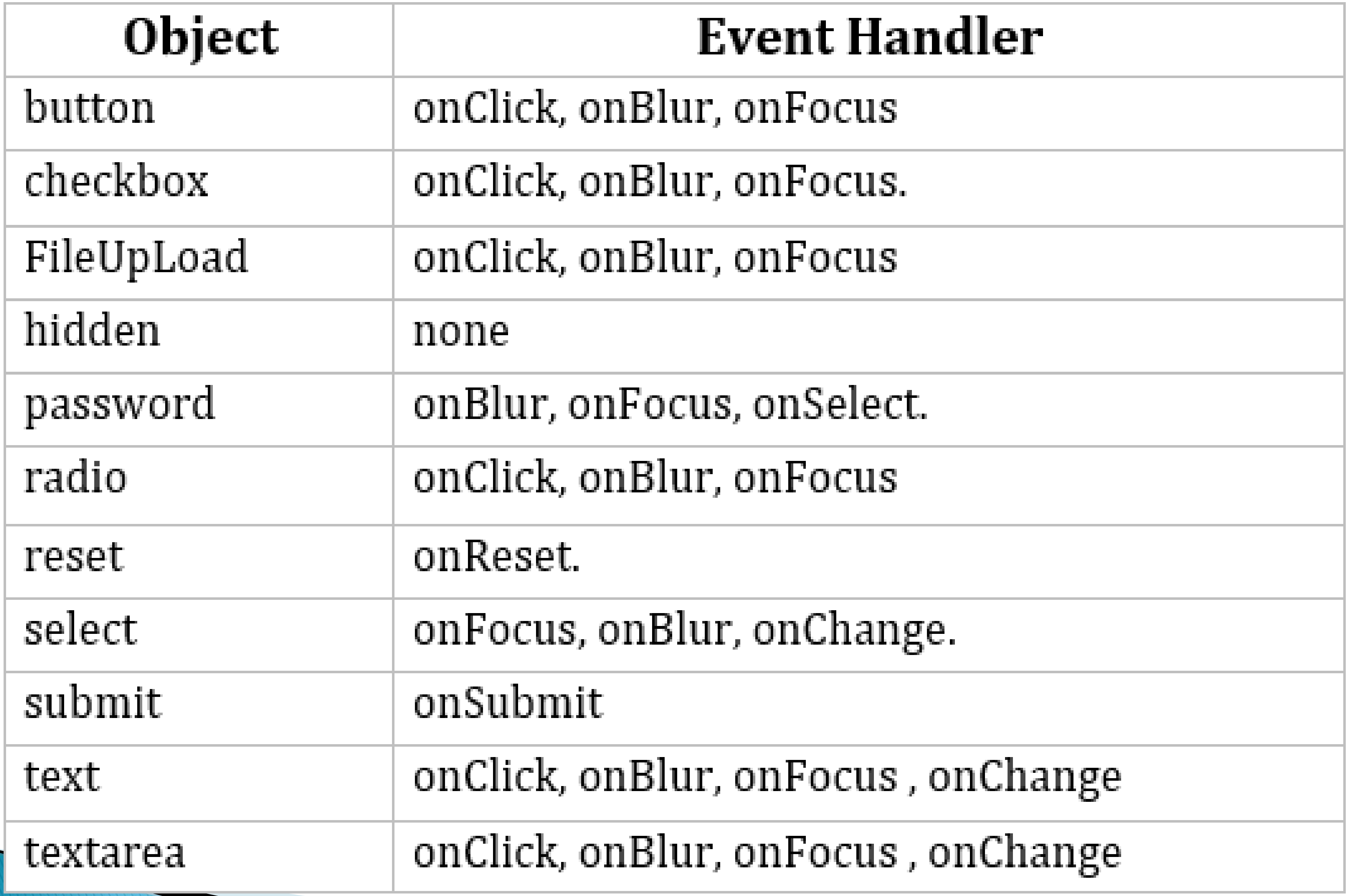
```
<!DOCTYPE html>
<html>
  <head>
    <title>Select Box Control</title>
  </head>
  <body>
    <form>
       Choose Your Subject by Following Dropdown: <br />
        <select name = "dropdown">
         <option value = "Subject" selected>CSS</option>
         <option value = "Subject">AJP</option>
         <option value = "Subject">OSY</option>
         <option value = "Subject">STE</option>
      </select>
    </form>
  </body>
</html>
```



Choose Your Subject by Following Dropdown:

Form Events

- Event is an activity that **represent a change in the environment.**
- A JavaScript event is an **action that can be detected by JavaScript**. Many of them are **initiated by user action** but some are **generated by the browser**.
- Event is triggered & then it **can be caught by JavaScript functions, which then do something response.**
- Event handler is a **script that get executed in response to these events**. Event handler enables the web documents to **respond the user activities through the browser window.**
- Event are specified **in lowercase & these are case sensitive**.

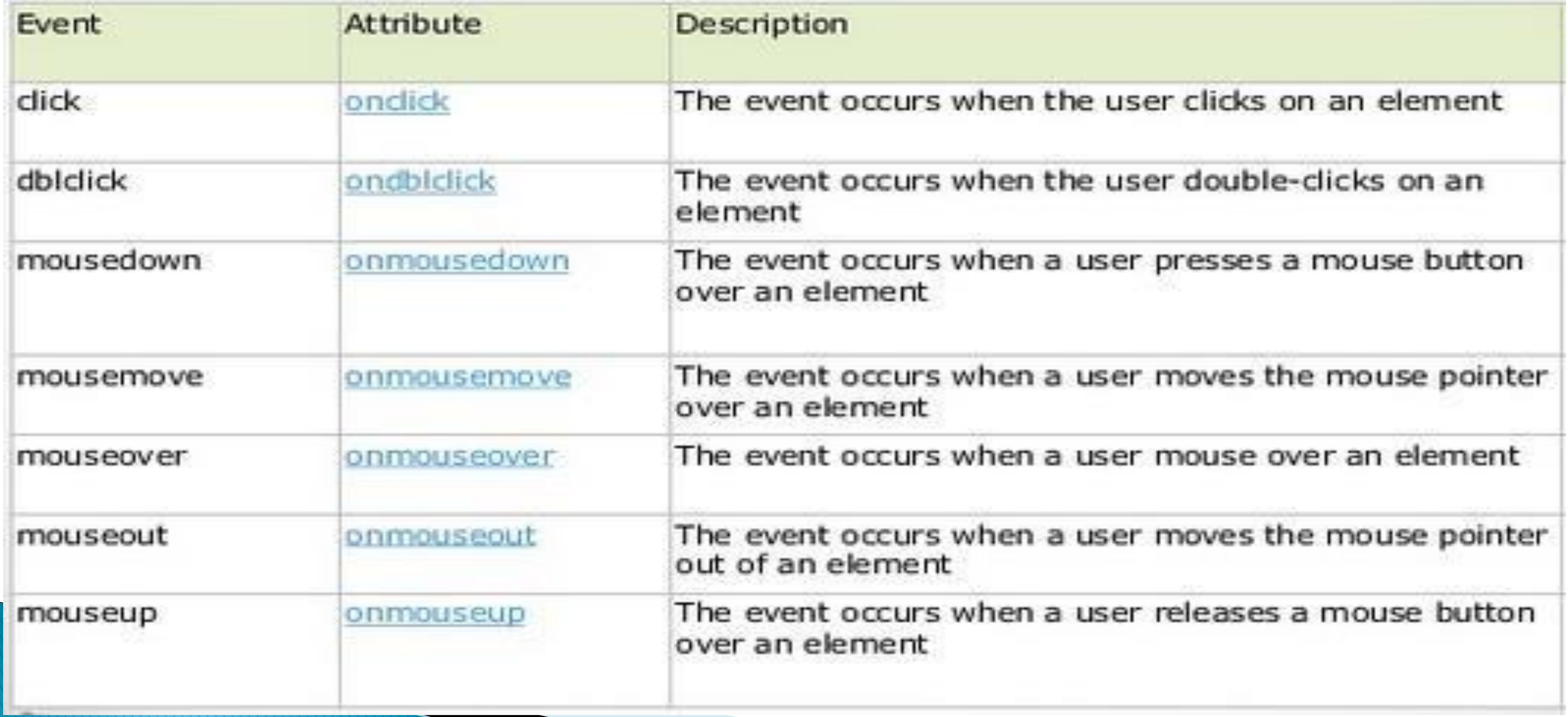


Object	Event Handler
button	onClick, onBlur, onFocus
checkbox	onClick, onBlur, onFocus.
FileUpLoad	onClick, onBlur, onFocus
hidden	none
password	onBlur, onFocus, onSelect.
radio	onClick, onBlur, onFocus
reset	onReset.
select	onFocus, onBlur, onChange.
submit	onSubmit
text	onClick, onBlur, onFocus , onChange
textarea	onClick, onBlur, onFocus , onChange

```
<!DOCTYPE html>
<html>
<body onload="myFunction()">
<h1>Hello World!</h1>
<script>
        function myFunction()
        {
        alert("Page is loaded");
        }
</script>
</body>
</html>
```

Mouse Events

- Mouse Event are **used to capture the interaction made by the user by using mouse.**

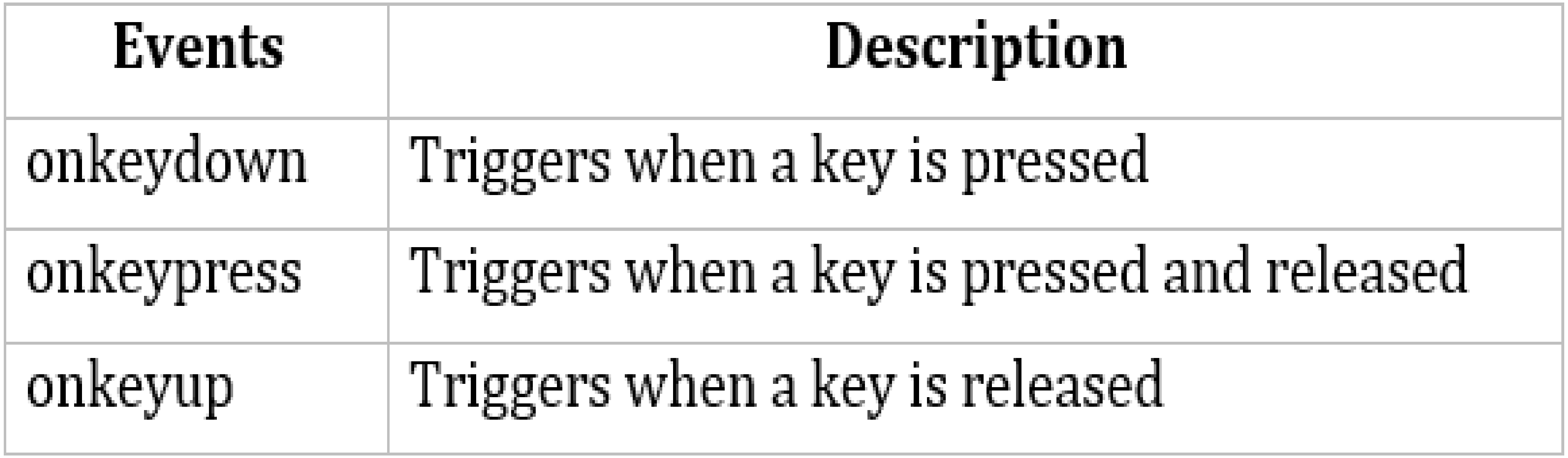


Event	Attribute	Description
click	onclick	The event occurs when the user clicks on an element
dblclick	ondblclick	The event occurs when the user double-clicks on an element
mousedown	onmousedown	The event occurs when a user presses a mouse button over an element
mousemove	onmousemove	The event occurs when a user moves the mouse pointer over an element
mouseover	onmouseover	The event occurs when a user mouse over an element
mouseout	onmouseout	The event occurs when a user moves the mouse pointer out of an element
mouseup	onmouseup	The event occurs when a user releases a mouse button over an element

```
<html>
  <head>
    <script>
        function over() {
          document.write ("Mouse Over");
        }
        function out() {
          document.write ("Mouse Out");
        }
    </script>
  </head>
  <body>
    <p>Bring your mouse inside the division to see the result....</p>
      <div onmouseover="over()" onmouseout="out()">
      <h2> This is inside the division </h2>
    </div>
  </body>
</html>
```

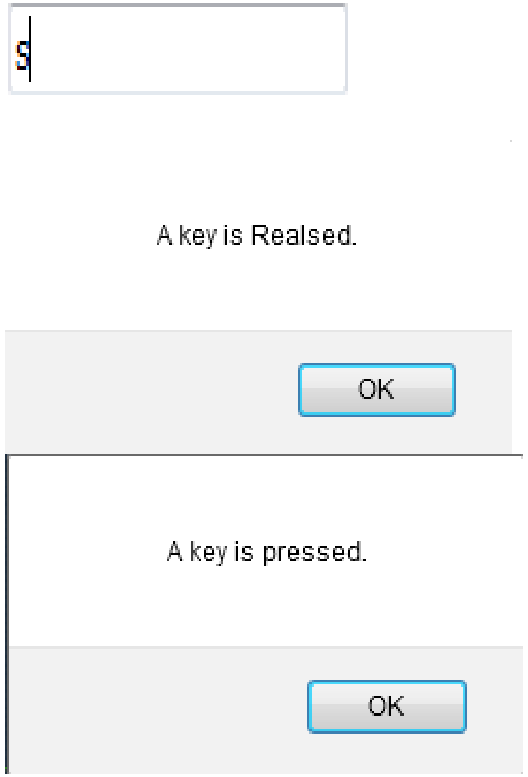
Key Events

- Key Event are **used to capture the interaction made by the user by using key.**



Events	Description
onkeydown	Triggers when a key is pressed
onkeypress	Triggers when a key is pressed and released
onkeyup	Triggers when a key is released

```
<html>
  <head>
    <script>
      <!--
        function fun1() {
          alert("A key is pressed.")
        }
       function fun2() {
          alert("A key is Realsed.")
        }
    </script>
  </head>
  <body>
     <input type = "text" onkeypress = "fun1()" onkeyup="fun2()">
  </body>
</html>
```


s

A key is Realsed.

A key is pressed.

Form Objects & Elements

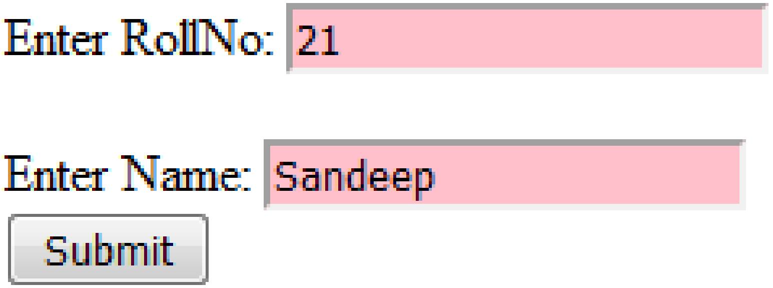
- form object is a Browser **object of JavaScript used to access an HTML form.**
- If a user wants to access all forms within a document then he can **use the forms array**.
- This array includes an item for each **form element, indexed starting with 0.**
- The form object is actually a property of document object that is uniquely created by the browser for **each form present in a document.**
- The properties and methods associated with **form object are used to access the form fields, attributes and controls associated with forms**.

```
<html>
<head>
 <script language="javascript">
  function validate() {
    var method = document.forms[0].method;
    var action = document.forms[0].action;
    var value = document.forms[0].elements[0].value;
    if(value != "CSS"){
     document.forms[0].reset();
    }
   else {
    alert("Hi CSS....!!");
   }
  }
</script>
</head>
<body>
 <form method="post">
  <input type="text" name="user" size=32>
  <input type="submit" value="Press Me!"
        onClick="validate()">
  </form>
</body>
</html>
```

Changing Attribute Value Dynamically

- It is possible to **change the attributes of the form elements dynamically.**
- During form filling process itself , **the color or font of the text field can be changed.**
- The dynamic change **helps the user to notify the importance change in the form fields.**

```
<html>
  <head>
     <script type = "text/javascript">
       function fun(e)
       {
         e.style.background= 'pink';
       }
    </script>   </head>
    <body>
    <form name = "myform">
       Enter RollNo: <input type ="text" name= "roll" onchange="fun(this)"/>
       <br/> <br/>
       Enter Name: <input type ="text" name= "name" onchange="fun(this)"/>
       <br/>
       <input type="submit" value ="Submit">
      </form>
  </body>  </html>
```



Enter RollNo: 21

Enter Name: Sandeep

Changing Option List Dynamically

- Option list represents the list of **one or more than one items which can be chosen by the user.**
- In a web application it is a common practice to change the content of the option list base on some category chosen.
- JavaScript allows to **change the item present in the list dynamically.**

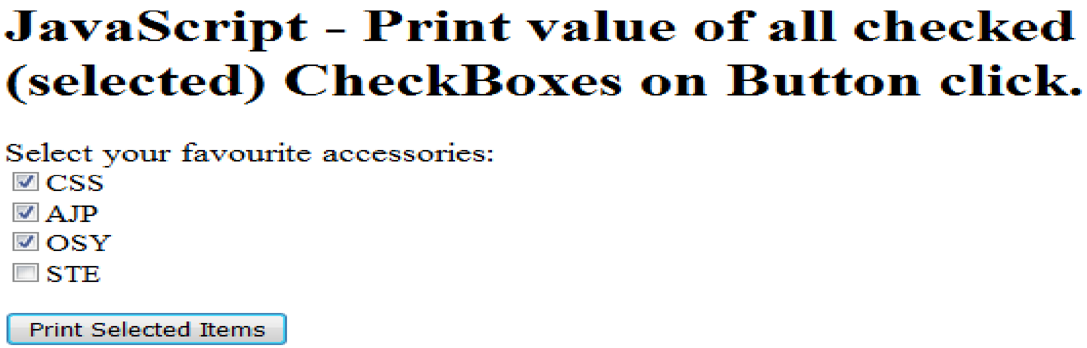
Java Script Program:

Java Script on Changing Option List Dynamically.docx

Evaluating Checkbox Selection

- Evaluating Checkbox Selection is a simple technique using which we **can display the named of check boxes that are selected**.

```
<html> <head>
<script type="text/javascript">
function printChecked(){
        var items=document.getElementsByName('sub');
        var selectedItems="";
        for(var i=0; i<items.length; i++){
                if(items[i].type=='checkbox' && items[i].checked==true)
                        selectedItems+=items[i].value+"\n";
        }
        alert(selectedItems);
        }
        </script>
        </head>
<body style="text-align: left;">
        <h1>JavaScript - Print value of all checked (selected) CheckBoxes on Button click.</h1>
        <big>Select your favourite accessories: </big><br>
        <input type="checkbox" name="sub" value="CSS">CSS<br>
        <input type="checkbox" name="sub" value="AJP">AJP<br>
        <input type="checkbox" name="sub" value="OSY">OSY<br>
        <input type="checkbox" name="sub" value="STE">STE<br>
        <p> <input type="button" onclick='printChecked()' value="Print Selected Items"/>    </p>
</body> </html>
```



JavaScript - Print value of all checked (selected) CheckBoxes on Button click.

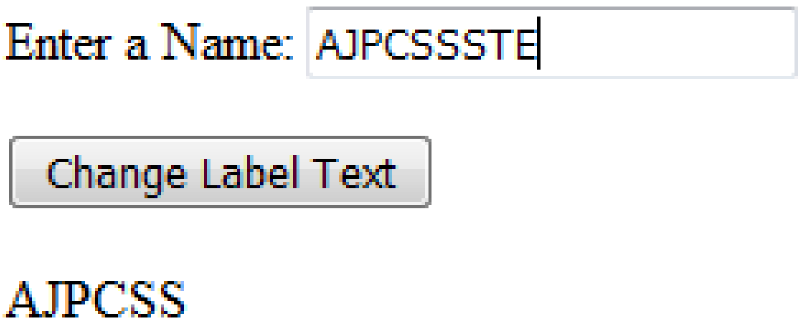
Select your favourite accessories:

☑ CSS
☑ AJP
☑ OSY
□ STE

Changing A Label Dynamically

- We can change the **label of any from element dynamically**.
- The same element can be used for multiple purpose by simply changing the label.

```
<html>
<body>
  Enter a Name: <input type="text" id="emp" value="" />
  <p>
    <input type="button" id="bt" value="Change Label Text"
onclick="changeLabel()" />
  </p>
  <label id="lblEmp">N/A</label>
</body>
<script>
  function changeLabel() {
    let lbl = document.getElementById('lblEmp');
    let empName = document.getElementById('emp').value;
    lbl.innerText = empName;
  }
</script>
</html>
```

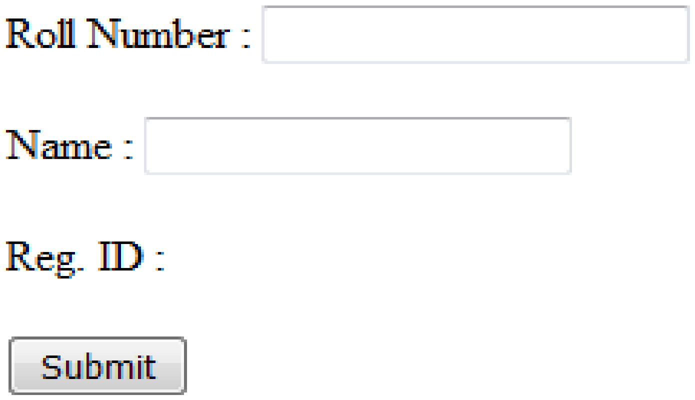

Enter a Name: AJPCSSSTE

AJPCSS

Manipulating Form Elements

- We can manipulating form elements **before submitting it to the web server.**
- For that purpose we can **keep some field hidden & at the time of submitting the form**, the desired value can be set to the hidden field so that the assigned value for the hidden can be submitted.

```
<html>  <head>    <script>
    function MyFunction()
   {
      with(document.forms.myform)
      {
         if(name.value.length>0 && roll.value.length>0)
         regid.value=name.value.charAt(0)+name.value.charAt(1)+ roll.value;
      }
     }
</script> </head>
   <body>
   <form name="myform">
      Roll Number : <input type= "text" name="roll"/><br><br/>
      Name : <input type= "text" name="name"/><br><br/>
      Reg. ID : <input type= "hidden" name="regid"/><br><br/>
      <input type="submit" name="Submit" value="Submit"
        onclick="MyFunction()"/>
   </form>
</body> </html>
```



Roll Number :

Name :

Reg. ID :

Intrinsic JavaScript Functions

- **Intrinsic** function means the built in functions that are provided by JavaScript.
- The JavaScript provides the **Intrinsic** function for **Submit & Reset Button.** It can be used while submitting the form or resetting the form fields.
- The submit() method of the form object can be used to send the form to the server in exactly same way as if the user has pressed the submit button.

```
<html>
  <body>
    <form name="myform">
        Roll Number : <input type= "text" name="roll"/><br><br/>
        Name : <input type= "text" name="name"/><br><br/>
        <img src= "submit.gif" onclick
="javascript:document.forms.myforms.submit()"/><br><br/>

   </form>
</body>
</html>
```

Disabling Elements

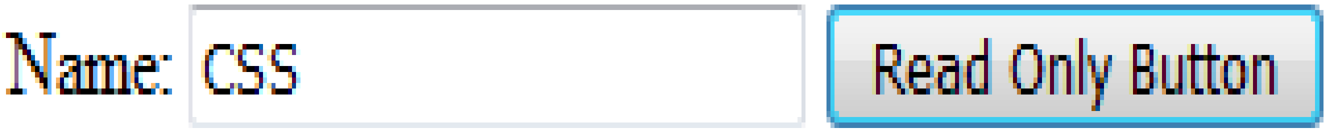
- We can restrict some fields on the **form by using disabled**.
- If disabled property of particular form element is set to true then user can not edit that element. Similarly on setting property to false we can edit the field.

```
<html>
<head>
<script>
     function EnableFunction()
     {
       documents.forms.myform.name.disabled=false;
     }
     function DisableFunction()
     {
       documents.forms.myform.name.disabled= true;
     }
</script>
<form name="myform">
           User Name : <input type= "text" name="name"/><br><br/>
          <input type= "button" value="Disable Name Field"
          onclick="DisableFunction()"/><br><br/>
          <input type= "button" value="Enable Name Field"
          onclick="EnableFunction()"/><br><br/>
</form>      </body>     </html>
```

Read Only Elements

- Some time we need to set some value to **a field which user should not change.**
- To restrict user form changing the value of particular field we make the **element readonly by setting readonly= true.**

```
<html>
<body>
Name: <input type="text" id="myText">
<button onclick="myFunction()">Read Only Button</button>
<script>
function myFunction()
{
 document.getElementById("myText").readOnly = true;
}
</script>
</body>
</html>
```



Name: CSS

Thank you