Session: 10

HTML Forms

Objectives

- Describe HTML5 forms
- Explain the working of new input types in HTML5
- Explain the new Form attributes
- Explain the new Form elements

Introduction to HTML5 Forms

HTML5 Web forms are those sections on the Web page that contain special elements called as controls.

The controls, such as check boxes, radio buttons, and text boxes provide a visual interface to the user to interact with them.

A user provides data through these controls that is sent to the server for further processing.

In HTML5, creation of form is made easier for Web developers by standardizing them with rich form controls.

It also provides client-side validations that are now handled natively by the browsers.

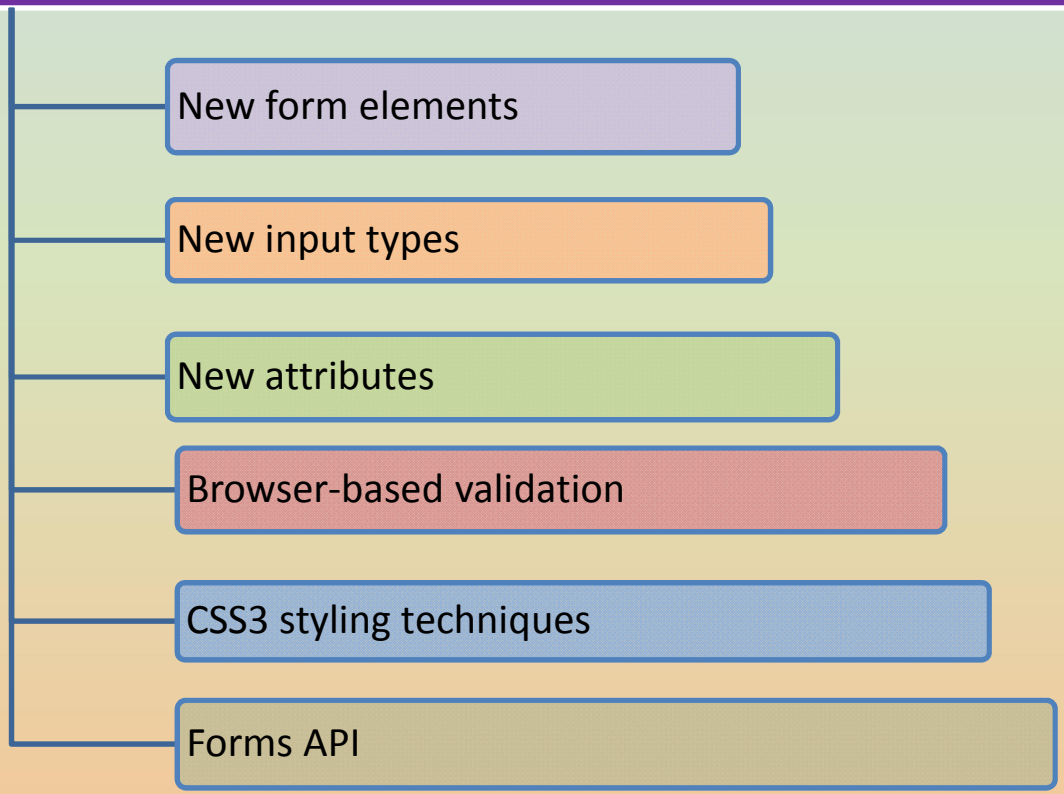
This reduces the load time of the pages and also removes the need of the repetitive JavaScript codes to be included on the page.

Even the visual appearance of the forms is improved when displayed on different devices, such as iPhone, ipad, touch screens, and browsers.

New Features in HTML5 Forms

- HTML5 Web forms bring great improvements related to form creation for the Web developers and also for users interacting with them.

The following are the changes introduced in HTML5 forms:



New Elements

- HTML5 has introduced a range of new elements that are expanding the options for more number of elements related to input on the forms.
- Following table lists the new elements in HTML5.

Data Type	Description
`progress`	Represents the completion progress of a task on the page
`meter`	Represents a scale of known range
`datalist`	Represents a set of options used with list attribute to make a drop-down control
`output`	Represents the result of a calculation

New Input Types 1-2

- The `input` element is a data field that allows the user to edit the data on the form.
- It has an attribute named `type` which controls the data type and characteristics of the input element.
- Following table lists the new input types supported by HTML5.

Type	Description
`email`	Represents the completion progress of a task on the page
`search`	Represents a scale of known range
`url`	Represents a set of options used with list attribute to make a drop-down control
`tel`	Represents the result of a calculation
`number`	Represents a numeric value in the input field

New Input Types 2-2

Type	Description
`range`	Represents a numeric value to be selected from a range of numbers
`date`	Represents a calendar which is shown whenever the field is clicked
`Week`	Represents date in year-week format
`month`	Represents a value with year-month format
`time`	Represents a value in hours and minutes format
`datetime`	Represents a full date and time input field with a time zone
`datetime-local`	Represents a full date and time with no time zone
`color`	Represents a predefined interface for selecting color

New Attributes

- HTML5 has introduced several new attributes that can be used with `form` and `input` elements. Attributes help the elements to perform their tasks.
- Following table lists the new attributes in HTML5.

Type	Description
`placeholder`	Represents a hint that help users to enter the correct data in the field
`required`	A Boolean attribute that validates the entry in the field
`multiple`	A Boolean attribute that allows multiple values to be entered in the field
`autofocus`	Focuses the input element on page load
`pattern`	Represents a regular expression for validating the field's value
`form`	Allows the elements to reference the form by including the form name

Browser-based Validation

HTML4 supported the use of custom JavaScript or libraries to perform validation on the client-side browsers.

These validations ensure that the input fields are checked before the form is submitted to the server for further processing.

The new attributes in HTML5, such as required and pattern can be used with the input elements to perform validation.

This relieves the Web developers from writing the custom JavaScript code for performing client-side validation on the Web pages.

HTML5 also provides advanced validation techniques that can be used with JavaScript to set custom validation rules and messages for the input elements.

CSS Styling Techniques 1-2

- A Web developer can enhance the form elements with the pseudo-class selectors, such as `:required`, `:valid`, and `:invalid`.
- The input fields which cannot be left blank while submitting the form can be displayed with an outline by styling the input field using CSS.
- The Code Snippet shows the CSS code for formatting non-empty and incorrect data input in the input element fields on the form.

```
<style>
     input:required
     {
      outline: 1px red solid;
      color: green ;
     }
     input:required:valid
     {
      background-size:10px 10px;
      background-position: right top;
      background-repeat: no-repeat;
     }
```

CSS Styling Techniques 2-2

```
input:required:invalid
{
    background-size:10px 10px;
    background-position: right top;
    background-repeat: no-repeat;
}
</style>
</head>
<body>
<form method="get" action="try.php">
  Name: <input type="text" name="name" required="true" /><br/>
  Email: <input type="email" name="emailid" required="true" />
 <input type="submit" value="submit" />
</form>

.....
```

Forms API

- HTML5 has introduced JavaScript API for forms to customize validations and processing performed on the forms.
- The new Forms API provides new methods, events, and properties to perform complex validations combining fields or calculations.
- Following table lists the events and methods.

Events and Methods	Description
`setCustomValidity` (message)	Sets the custom error message that is displayed when the form is submitted by the user
`checkValidity()`	Checks the validity of the e-mail address entered by the user
`oninvalid`	Allows script to run only when the element is invalid
`onforminput`	Allows script to run when the form run when a form gets a input from the user
`onformchange`	Represents a regular expression for validating the field's value
`form`	Allows script to run when the form changes

Working with New Input Types

- The type attribute of the input element determines what kind of input will be displayed on the user's browser.
- The default input is `type="text"`.
- The registration form in the session is using the following input types:
  - `text`
  - `label`
  - `radio`
  - `textarea`
  - `checkbox`
  - `submit`
- HTML5 has introduced more data-specific user interface elements.

E-mail Address 1-2

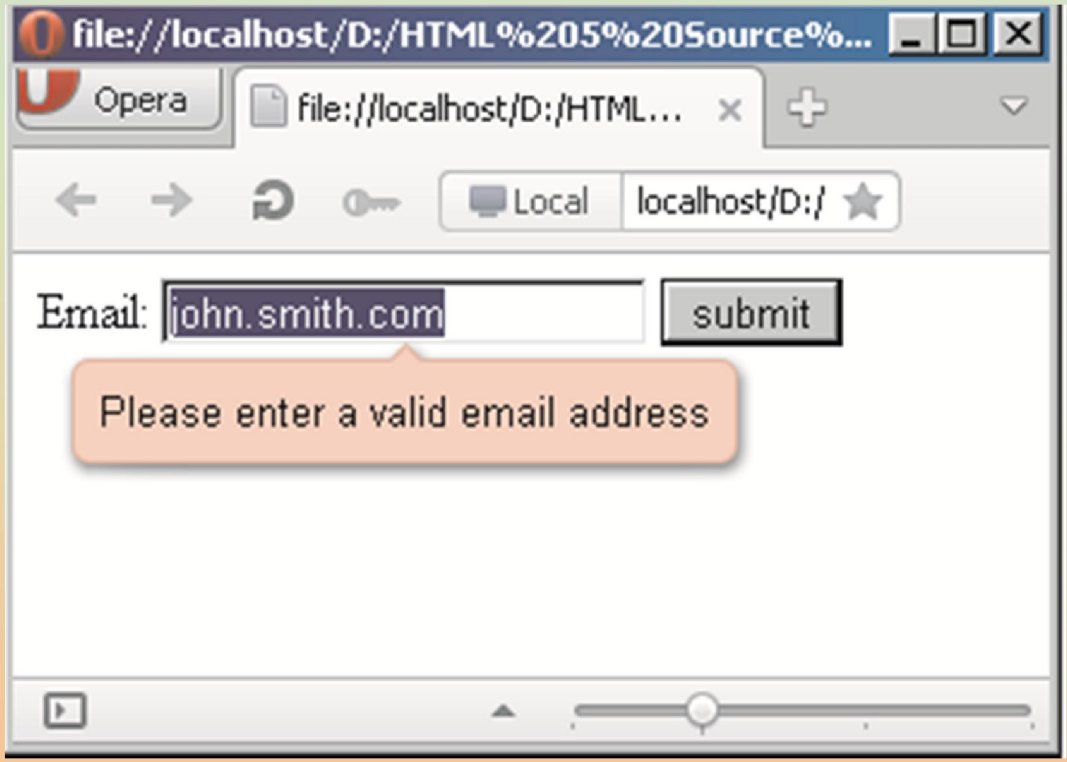
- The `type="email"` is used for specifying one or more e-mail addresses. To allow multiple addresses in the e-mail field, separate each address with comma-separator.
- In the registration form, the input type is changed from `text` to `email` as shown in the following code snippet:

```
<form method="get" action="test.html">
   <label for="emailid">Email:</label>
   <input type="email" value="" id="emailid"
          name="emailaddress" maxlength="255" />
   <input type="submit" value="submit"/>
</form>
```

- In the code snippet, `<label>` tag defines a label for the element on the form.
- The `for` attribute of `<label>` tag binds it with the related element, that is email element, on the form.
- The value of `for` attribute must match with the value of `id` attribute assigned to the element.
- The `email` contains two attributes, namely `id` and `name`.
- The `id` attribute specifies a unique id for the element.
- The value of the `id` attribute should be unique within the document.

E-mail Address 2-2

- The `name` attribute specifies a name for the `input` element.
- The look of the input is still like a plain text field, but changes are applied behind the scenes.
- Browsers, such as Firefox, Chrome, and Opera will display a default error message if user submits the form with some unrecognizable contents.
- Following figure shows the error message for incorrect E-mail Address in Chrome.

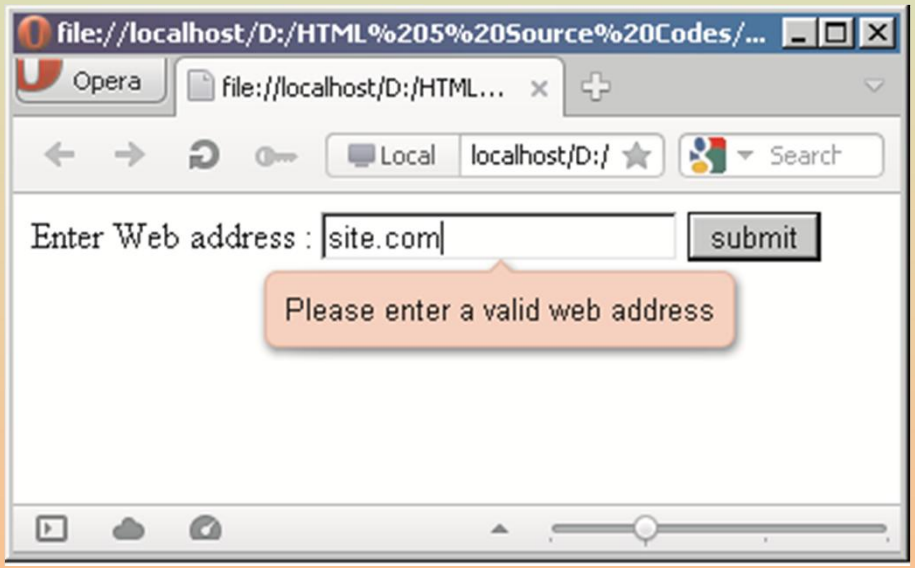


URL

- The `type="url"` input element is used for specifying a Web address.
- The look of the `url` field is a normal text field.
- The Code Snippet shows the code of `url` input type.

```
<label for="url">Enter your Web page address:</label>
 <input type="url" value="" id="urlname" name="urltext"
        maxlength="255" />
<input type="submit" value="submit"/>
```

- Browsers, such as Opera, Firefox, and Chrome support validation for the `url` input type.
- While validating the URL, browsers only checks for entry with forward slash (/).
- Following figure shows the error message for incorrect URL in Chrome.



Telephone Number

- The `type="tel"` input element is used for accepting telephone numbers.
- The `tel` type does not impose a particular pattern.
- It supports characters, numbers, and special characters except new lines or carriage returns.
- A user can enforce a pattern for `tel` input type by using placeholder, pattern attribute, or a JavaScript code for performing client-side validation.
- The Code Snippet shows the code for including input type on the registration form.

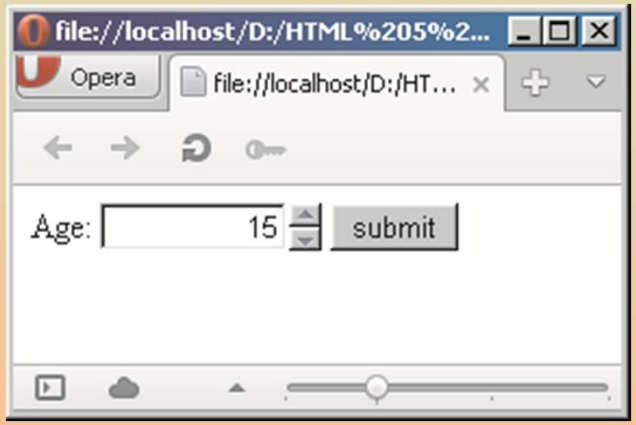
```
<label for="telno">Telephone Number:</label>
<input type="tel" value="" id="telno" name="telephone_no"
     maxlength="10" />
```

Number

- The input `type="number"` is used for accepting only number values.
- The `input` element displayed for `number` type is a spinner box.
- The user can either type a number or click the up or down arrow to select a number in the spinner box.
- The Code Snippet shows the code for including number input type on the form.

```
<label for="stud_age">Age:</label>
<input type="number" value="15" id="stud_age"
       name="studentage" min="15" max="45" step="1" />
<input type="submit" value="submit"/>
```

- In the code snippet, the `number` input type has attributes, such as `min` and `max` to specify the minimum and maximum value for the input.
- Following figure shows the input type in Opera .



Range 1-2

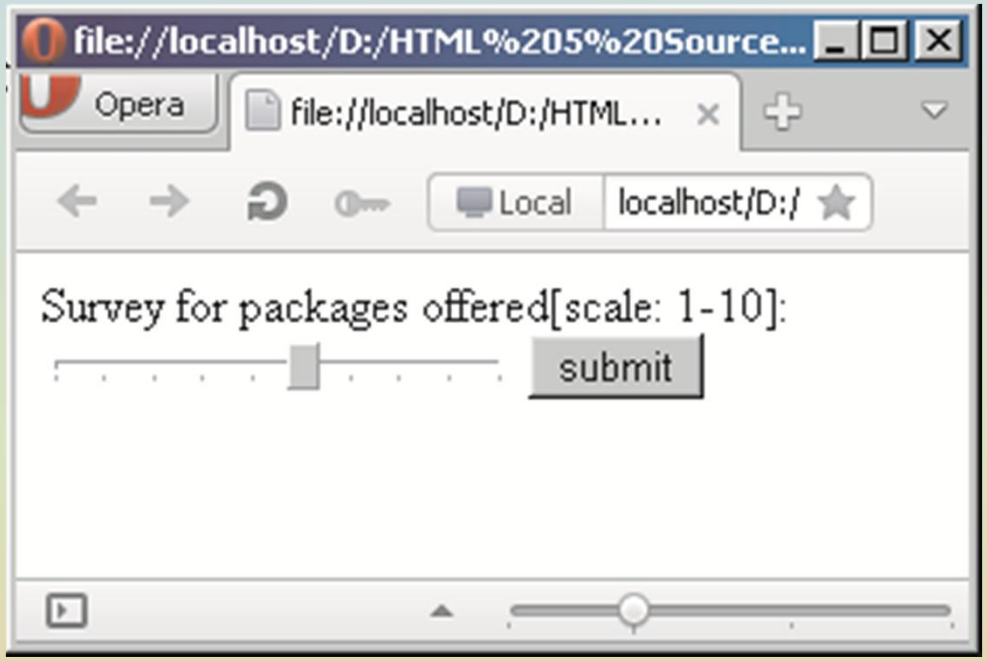
- The input `type="range"` is similar to `number` type and displays a slider control on the page.
- The range type is used when the exact value is not required in the input.
- For example, an online survey form asking the clients to rate the products may not receive exact values in the ratings.
- The Code Snippet shows the code for including range input type with attributes `min` and `max`.

```
<label>Survey for packages offered[scale: 1-10]:</label>
<input type="range" name="rating" min="1" max="10" />
<input type="submit" value="submit"/>
```

- In code snippet, the `range` input type contains attributes, such as `min`, `max`, `step`, and `value`.
- The `min` and `max` attributes are used to specify the minimum and maximum value allowed for a range and are set to `1` and `10` respectively.
- The `step` attribute specify the intervals for incrementing the value.
- The value of `step` attribute is `1` by default.
- The `value` attribute specifies the default value for the range.
- The default value is the midpoint of the range specified.

Range 2-2

- Following figure shows the range input type in Opera.



- Following figure shows the value for the range input type in the URL after the form is submitted by the user.
- The rating selected by the user can be seen in the Address Bar of the browser.

Date and Time 1-7

- HTML5 has introduced several new input types for date and time.
- The format for all these date and time types is according to the ISO standards.
- At present only Opera provides the support for date element by displaying a calendar control.

➢ Date

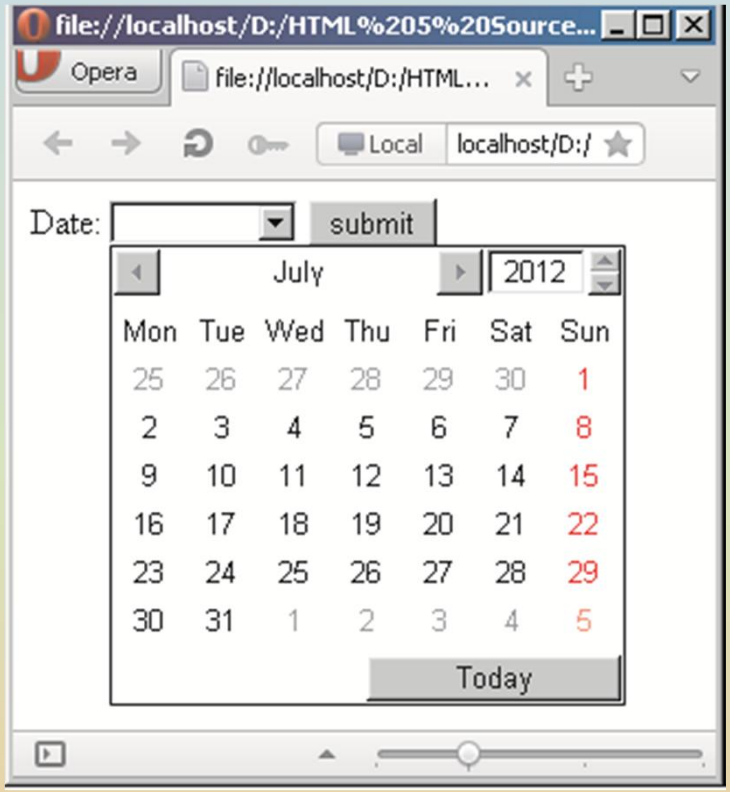
- This input type contains only date in year, month, and day format. The time part is not support by date type.
- The Code Snippet shows the code of the date input type.

```
<label for="stdate">Date:</label>
<input type="date" id="stdate" name="startdate"
   min="2000-01-01"/>
<input type="submit" value="Submit" id="btnSubmit"></input>
```

- In the code snippet, all date input types have `min` and `max` attributes to set the range for the dates.
- The default value for `date` input type is based on the browsers.
- Thus, it is advisable to set the minimum and maximum value for the `date` type.

Date and Time 2-7

- Following figure shows the input type.



- Following figure shows the value sent for the date input type after the form is submitted by the user.

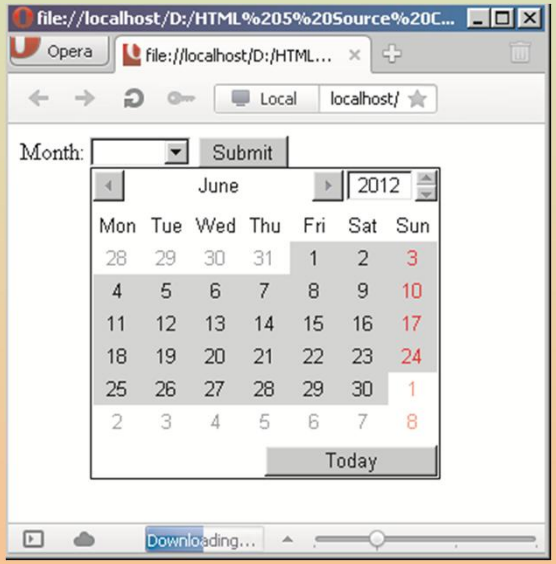
Date and Time 3-7

Month

- The `type="month"` includes only the year and month in the input.
- The Code Snippet shows the syntax of month input type.

```
<label for="stmonth">Month:</label>
<input type="month"  id="stmonth" name="startmonth" />
<input type="submit" value="submit"/>
```

- Browser such as Opera will display the date picker for selecting month.
- On selecting any day from the calendar, the whole month is selected.
- Following figure shows the date picker for the month input type.



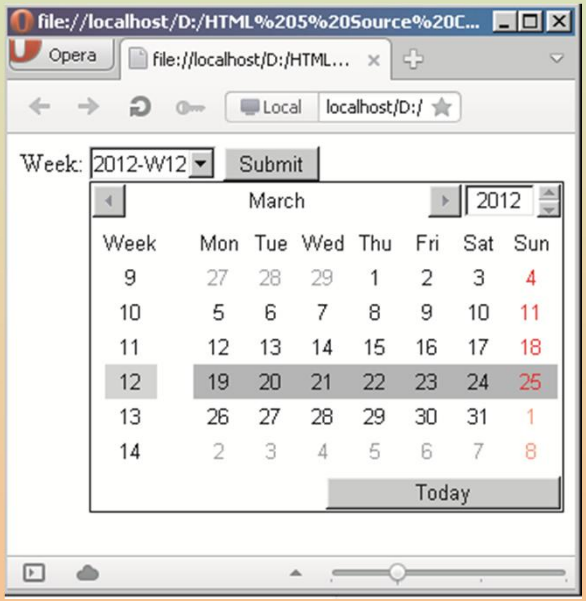
Date and Time 4-7

Week

- The input `type="week"` provides a similar interface as displayed for `date` type and selects the entire week.
- The Code Snippet shows the syntax of the week input type.

```
<label>Week:</label>
<input type="week" name="week" />
<input type="submit" value="submit"/>
```

- Following figure shows the week input type in Opera.



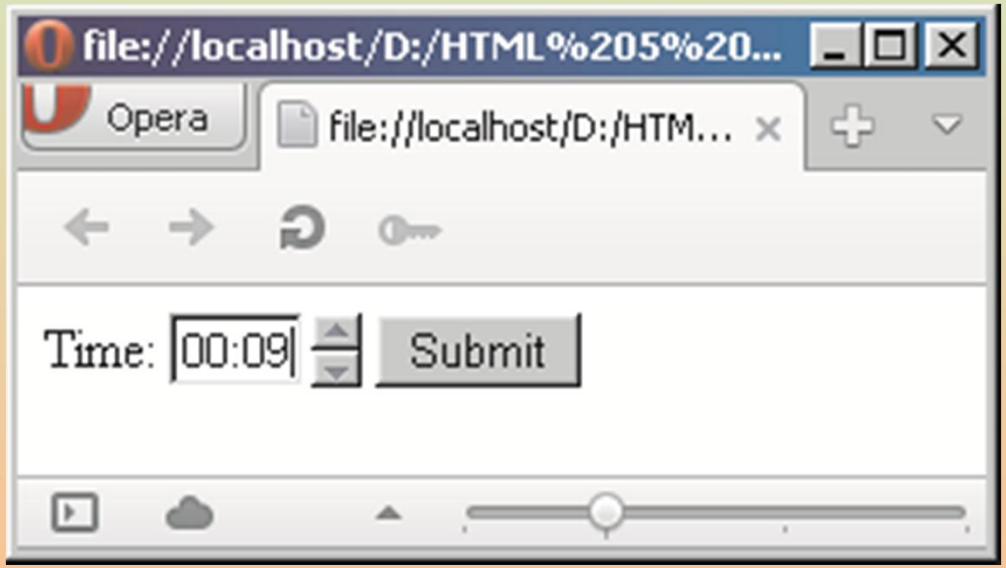
Date and Time 5-7

Time

- The input `type="time"` displays a time of day in hours and minutes format (24-hour clock).
- The Code Snippet shows the syntax of the time input type.

```
<label>Time:</label>
<input type="time" name="time" />
<input type="submit" value="submit"/>
```

- Following figure shows the time input type in Opera.



Date and Time 6-7

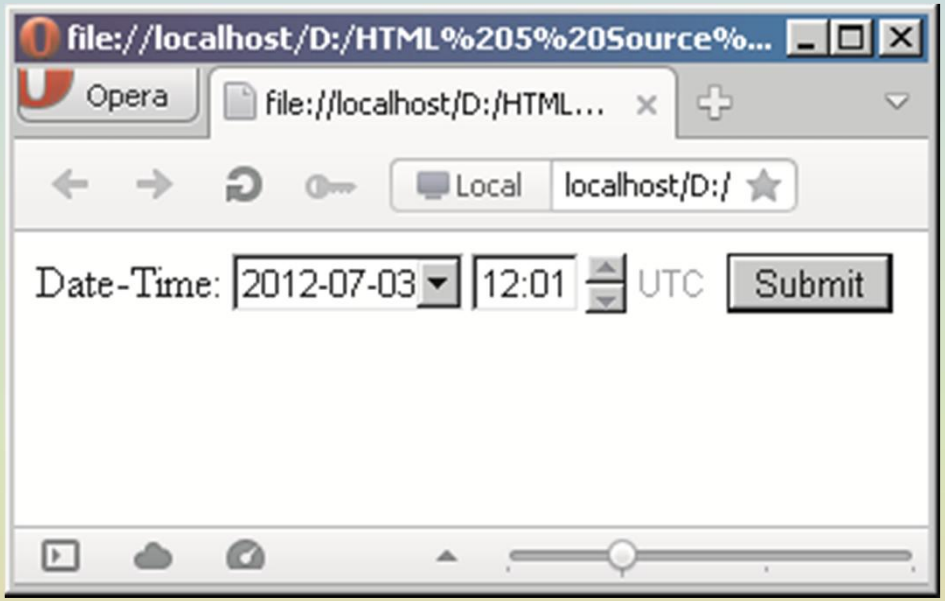
Datetime

- The input `type="datetime"` includes full date and time in the input.
- The input includes a date part and a time part which is represented as Coordinated Universal Time (UTC).
- Thus, UTC time will be displayed with 'T' followed by a 'Z'.
- The Code Snippet shows the syntax of datetime input type.

```
<label for="mydatetime">Date-Time:</label>
<input type="datetime" name="mydatetime" />
<input type="submit" value="submit"/>
```

Date and Time 7-7

- Following figure shows the datetime input type in Opera.



Datetime-local

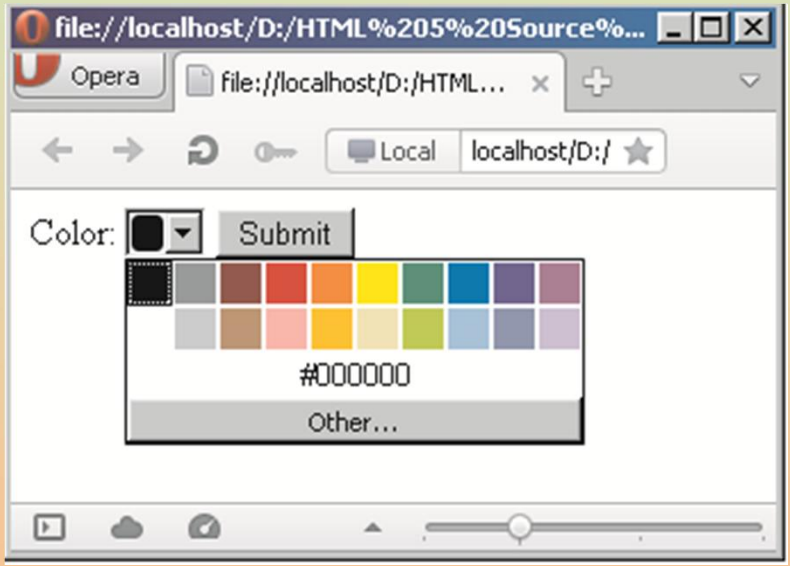
- This input type is similar to `datetime` input type, except that the time zone is omitted for input `type="datetime-local"`.

Color

- HTML5 provides a predefined interface for selecting the colors using input `type="color"`.
- The input value from the color input field is a hexadecimal number.
- For example, `#00FF00` represents a hexadecimal RGB color value.
- The Code Snippet shows the usage of color input type to display a color picker on the Web page.

```
<label>Color:</label>
<input type="color" name="mycolor" />
<input type="submit" value="submit"/>
```

- Following figure shows the color input type in Opera.



New Form Attributes

- HTML5 has provided several new attributes that perform the validations without writing JavaScript snippets for them.
- These attributes perform the following tasks:
  - Check data provided by users with the regular expression pattern assigned to the fields
  - Inform users with appropriate errors
  - Check that the required fields are not left empty by the users
  - Enable multiple values for the fields, if provided

- These attributes can be used to support scripting drawbacks, without actually hard coding them in the Web pages.
- Browsers that do not understand these new attributes will ignore them.

Required 1-2

- This is a `boolean` attribute that informs the browser to submit the form only when the required fields are not left empty by the users.
- The input type elements, such as `button`, `range`, and `color` cannot be set for `required` attribute as they have a default value.
- Different Web browsers such as Opera, Chrome, and Firefox provide different error messages, such as 'This is a required field', or 'Please fill out this field' for `required` attribute.
- The Code Snippet shows assignment of required attribute to the name field on the registration form.

```
<label>Name: <em> <img src="star.jpg" width="9" height="10"
             alt="" border="0"> </em>
</label> <br>
<input type="text" value="" name="first" size="8" tabindex="1"
   required ="true"/>
<input type="text" value="" name="last" size="14" tabindex="2"
   required="true"/>
<input type="submit" value="submit"/>
```

Required 2-2

- Following figure shows the message of required attribute in Opera.

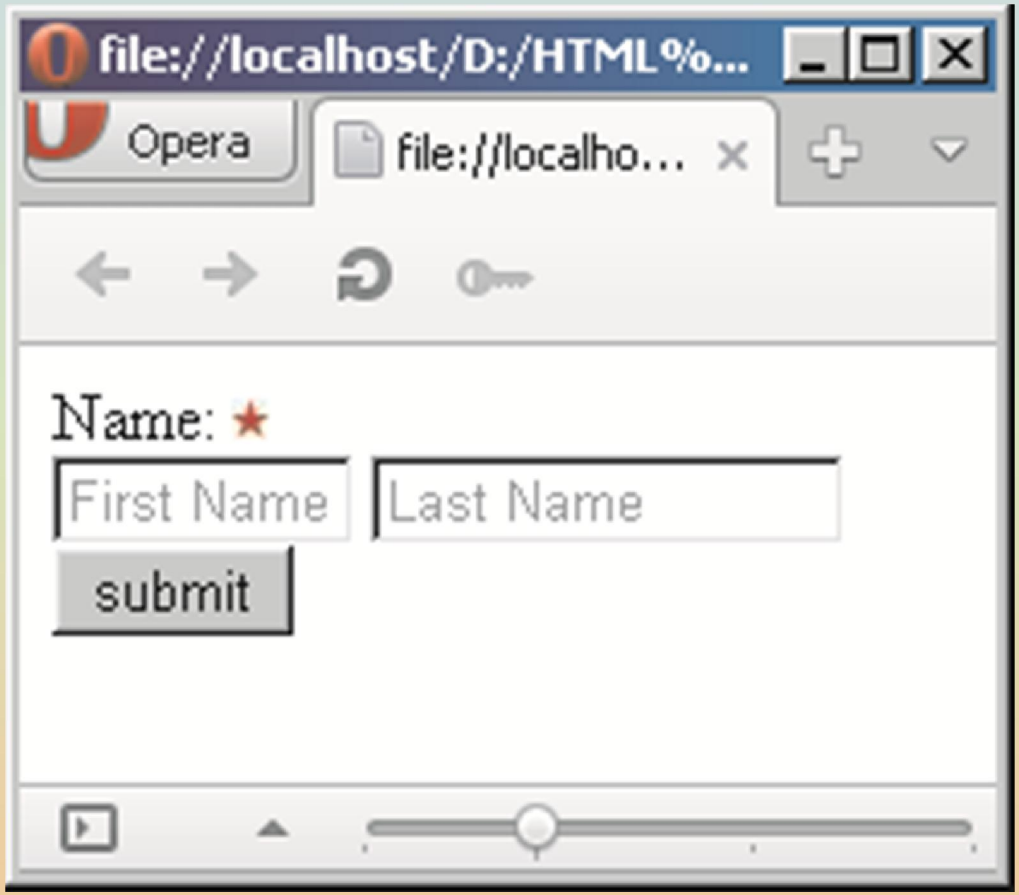
Placeholder 1-2

- This attribute displays a short hint or text inside a form element informing the user about what data needs to be entered in that field.
- The `placeholder` text toggles, which means it appears in the field and disappears when the user clicks inside the field.
- If the size of the hint exceeds the field size, then use `title` attribute to describe text for the field.
- The Code Snippet shows the assignment of placeholder attribute to the name field on the registration form.

```
<label>Name: <img src="required_star.gif" height="10px"
  width="10px"/></label> <br>
  <input type="text" value="" name="first" size="8"
    tabindex="1" required="true" placeholder="First Name"/>
  <input type="text" value="" name="last" size="14"
    tabindex="2" required="true" placeholder="Last Name"/>
<br/>
```

Placeholder 2-2

- Following figure shows the message of placeholder attribute in Opera.



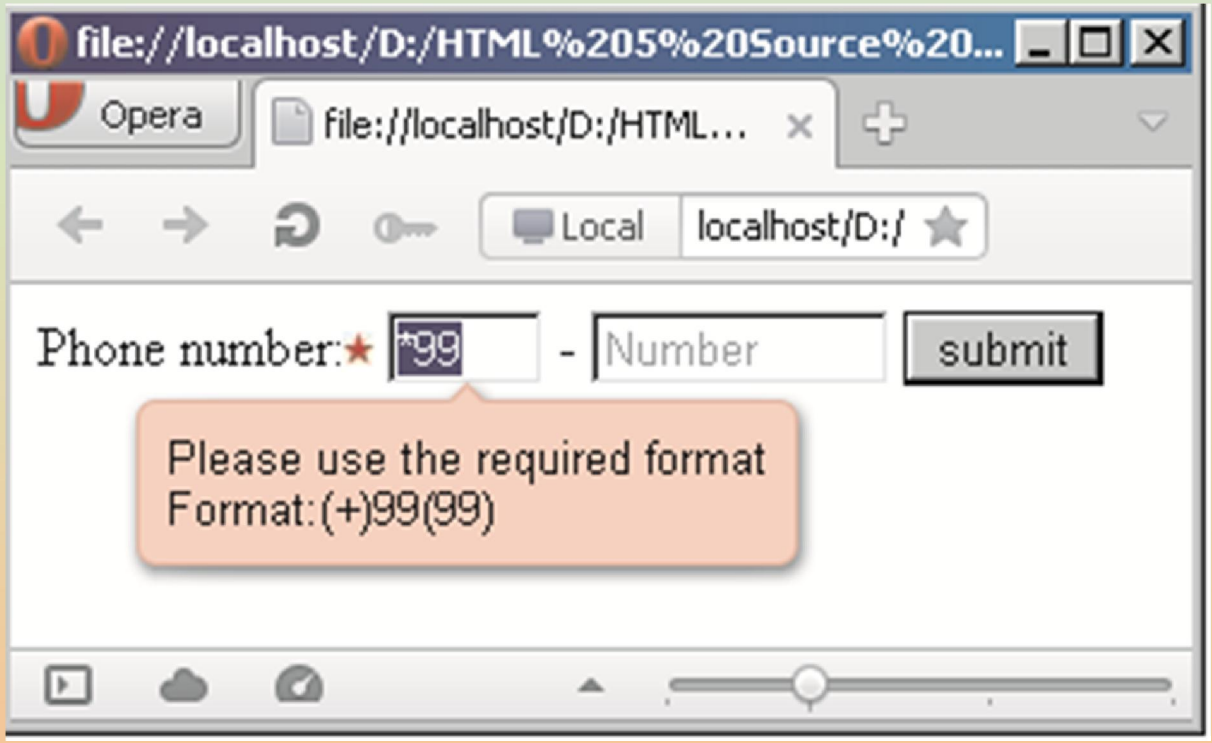
Pattern 1-2

- The `pattern` attribute uses regular expressions for validating the fields.
- The data entered by the user must match with the pattern specified in the regular expression.
- This helps to limit the input accepted from the user.
- To inform the users about the excepted pattern for the data, use the `title` attribute, which is displayed as a tool tip when pointer is moved over the field.
- The Code Snippet shows the assignment of `pattern` attribute to the phone number field on the registration form.

```
<label>Phone number:<img src="required_star.gif" height="10px"
   width="10px"/></label>
 <input type="tel" value="" size="4" maxlength="5"
   tabindex="11" required="true" placeholder =
   "Code"pattern="[+0-9]{1,4}" title="Format:(+)99(99)"/>
 <label>-</label>
 <input type="tel" value="" size="10" maxlength="12"
   tabindex="13" required="true" placeholder="Number"
   pattern="[0-9]{8,}"  title="Minimum 8 numbers"/>
```

Pattern 2-2

- In the code snippet, `[+0-9]` pattern indicates that only special character '+' as well as numbers are allowed.
- Also, `{1, 4}` refers to the length of the numbers, that is between `1` and `4`.
- Similarly, `{8,}` means minimum eight numbers are allowed in the `tel` input type field.
- Following figure shows the message of pattern attribute in Opera.



Multiple 1-2

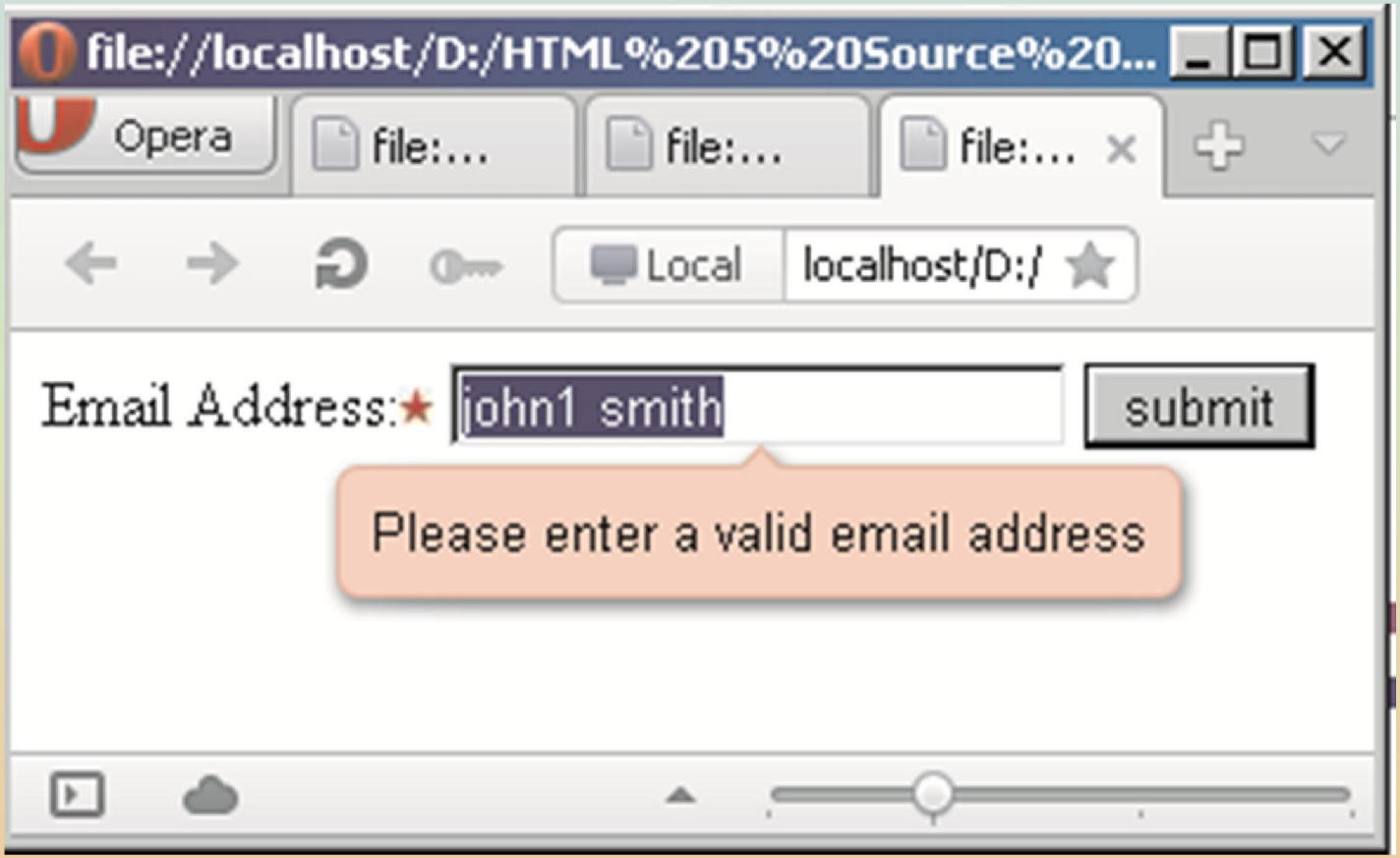
- This is a `boolean` attribute that allows multiple values for some input types.
- This was available only for `select` input type in the earlier version of HTML.
- HTML5 allows multiple attribute with input types, such as email and file.
- If assigned, it allows selection of multiple files, or include several e-mail addresses in the email field separated by comma separator.
- The Code Snippet shows the assignment of `multiple` attribute to the e-mail address field on the registration form.

```
<label>Email Address:<img src="required_star.gif" height="10px"
width="10px"/></label>
<input type="email" value="" name="emailid" maxlength="255"
tabindex="5" required="true" placeholder="Email Address"
multiple/>
```

- In the code snippet, `multiple` attribute will allow insertion of multiple e-mail addresses in the field.
- Every e-mail address will be validated individually by the browser.

Multiple 2-2

- Following figure shows the validation of multiple e-mail address.



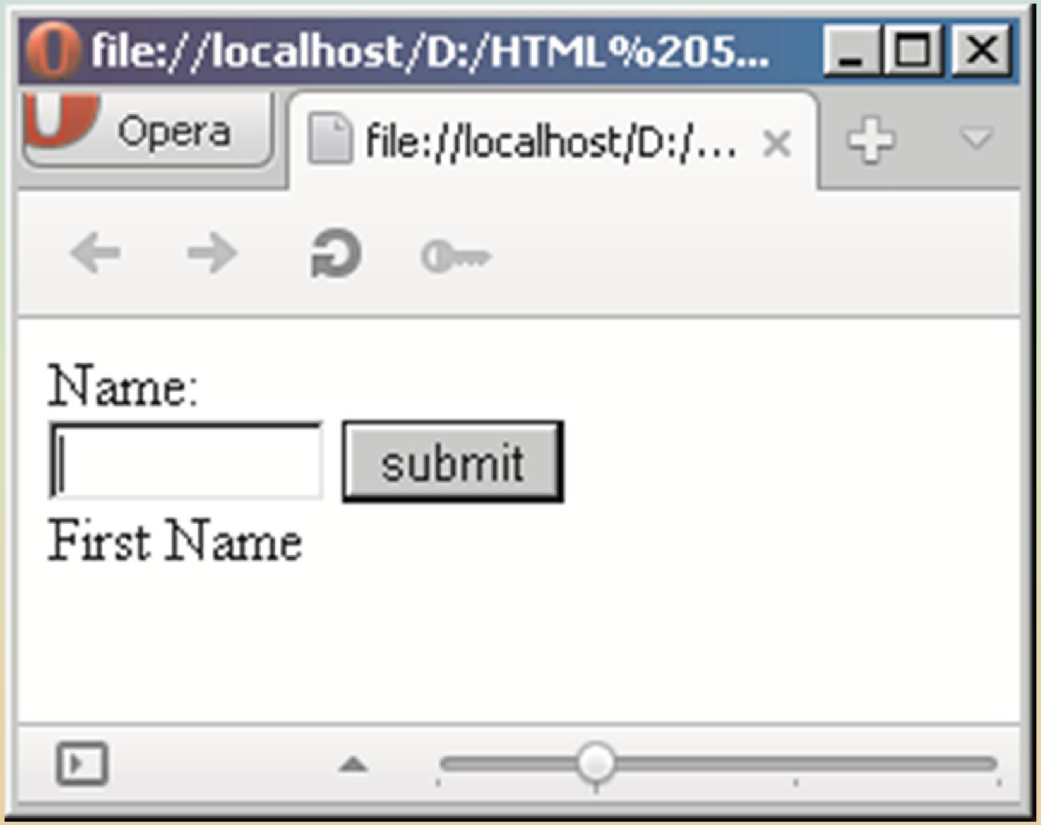
Autofocus 1-2

- The autofocus attribute will focus on the input field on page load.
- However, depending upon the situation, it will not move the focus away if the user has selected some other field.
- Only one element can be focused with autofocus attribute on a particular page while loading.
- The Code Snippet shows the assignment of autofocus attribute to the first name field on the registration form.

```
<label>Name:</label>
 <br>
  <input type=”text” value=”” name=”first” size=”8”
    tabindex=”1” placeholder =”First Name” autofocus/>
  <input type=”submit” value=”submit”/>
  <br>
<label>First Name</label>
```

Autofocus 2-2

- Following figure shows the behavior of `autofocus` attribute.



Form

- Earlier, all the form controls need to be provided between the opening and closing `<form>` tag.
- In HTML5, elements can be inserted at any place in the document and they can reference the form using the form attribute.
- The Code Snippet shows the association of an element with the form on the Web page.

```
<body>
 <input type="text" name="mytext" id="mytext" form="myform"/>
 . . .
 . . .
  <form id="myform">
 . . .
 . . .
  </form>
</body>
```

- In the code snippet, the form is declared with an `id` attribute.
- The value of the `id` attribute is assigned to the input element using `form` attribute.

Autocomplete Attribute 1-2

HTML5 offers an `autocomplete` attribute which provides control on prefilled values displayed in the fields.

It must be specified on the form element which applies for all input fields or on particular input fields.

The input element that can support autocomplete are text, url, tel, password, datepickers, range, and color.

The autocomplete feature comprises two states namely, on and off. The on state indicates that the data that is not sensitive can be remembered by the browser.

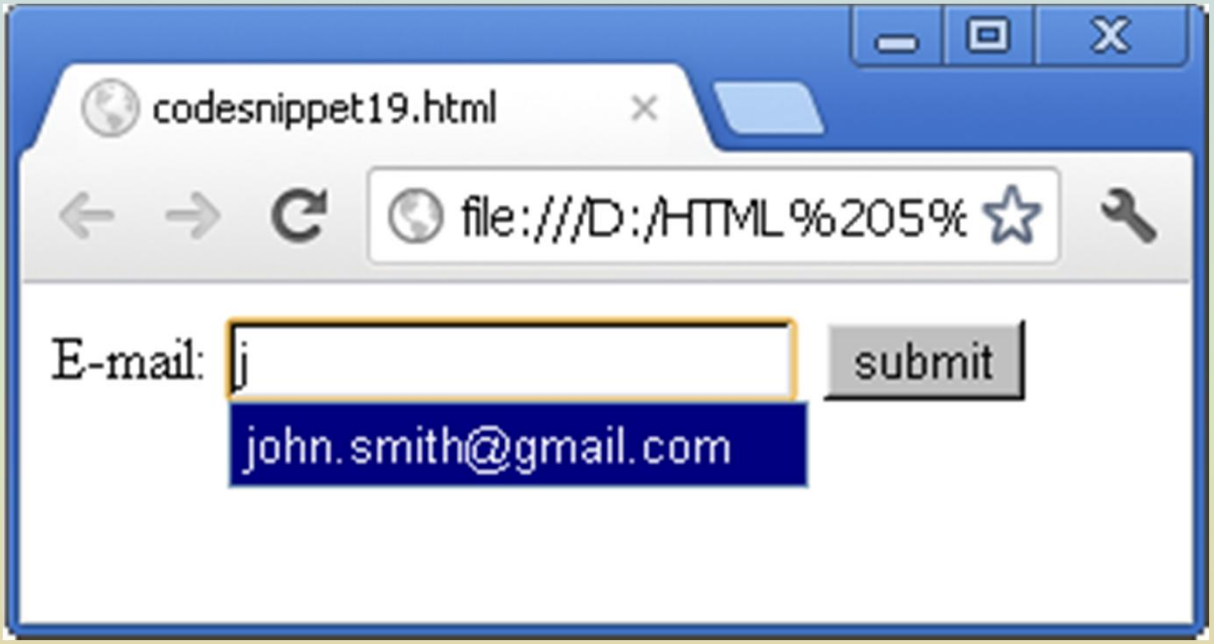
The off state indicates that the data will not be remembered. Such data may be sensitive and not safe for storing with the browsers.

By default, many browsers have the autocomplete feature enabled in them.

The browsers that do not support autocompletion, can be turned on or off for this behavior by specifying `autocomplete` attribute.

Autocomplete Attribute 2-2

- Following figure shows the behavior of `autocomplete` attribute in Chrome.



- The Code Snippet demonstrates disabling the default behavior of `autocomplete` attribute.

```
E-mail: <input type="email" name="email" autocomplete="off" />
        <input type="submit" value="submit"/>
```

New Form Elements

- HTML5 has introduced some new elements that can be incorporated in the Web pages.
- These new elements are specifically designed to be used with the JavaScript.
- When combined with JavaScript, these new elements can be more functional.
- At present, all the browsers do not provide the support for these new elements.
- If the control is not supported by the browser, then it displays element as a text field.
- Opera provides the support for all the new form elements.

  - `Datalist`
  - `Progress`
  - `Meter`
  - `Output`

Datalist 1-3

Datalist is a form-specific element. It provides a text field with a set of predefined list of options that are displayed in a drop-down list.

When the text field receives focus, a list of options is displayed to the user.

The <datalist> element is very similar to standard <select> element available in earlier HTML.

The only difference in datalist is that it allows the user to enter data of their choice or select from the suggested list of options.

The lists of options for the <datalist> element are placed using the option element.

Then, the datalist is associated with an input element using the list attribute.

The value of the list attribute is the value of id attribute provided with the <datalist> element.

Datalist 2-3

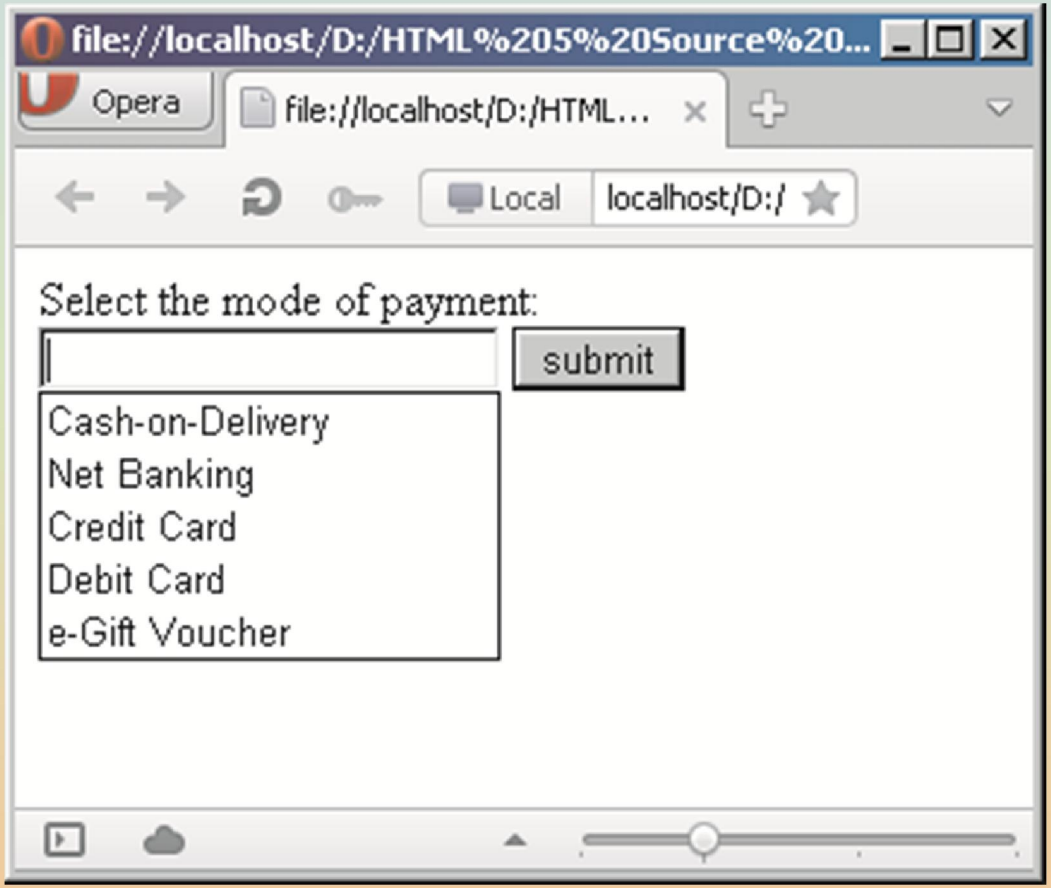
- At present, only Opera browser provides the support for the `datalist`.
- The Code Snippet shows the syntax of providing the `<datalist>` element on the form.

```
<label> Select the mode of payment: </label>
<input type="text"  name="payment" list="paymentlist" />
<datalist id="paymentlist">
  <option value="Cash-on-Delivery">
  <option value="Net Banking">
  <option value="Credit Card">
  <option value="Debit Card">
  <option value="e-Gift Voucher">
</datalist>
<input type="submit" value="submit"/>
```

- As shown in the code snippet, a datalist requires value attribute to be added with the `<option>` tag.
- Values nested between the opening and closing `<option>` tag will not be displayed in the datalist menu.

Datalist 3-3

- Following figure shows the `<datalist>` element in Opera.



Progress 1-2

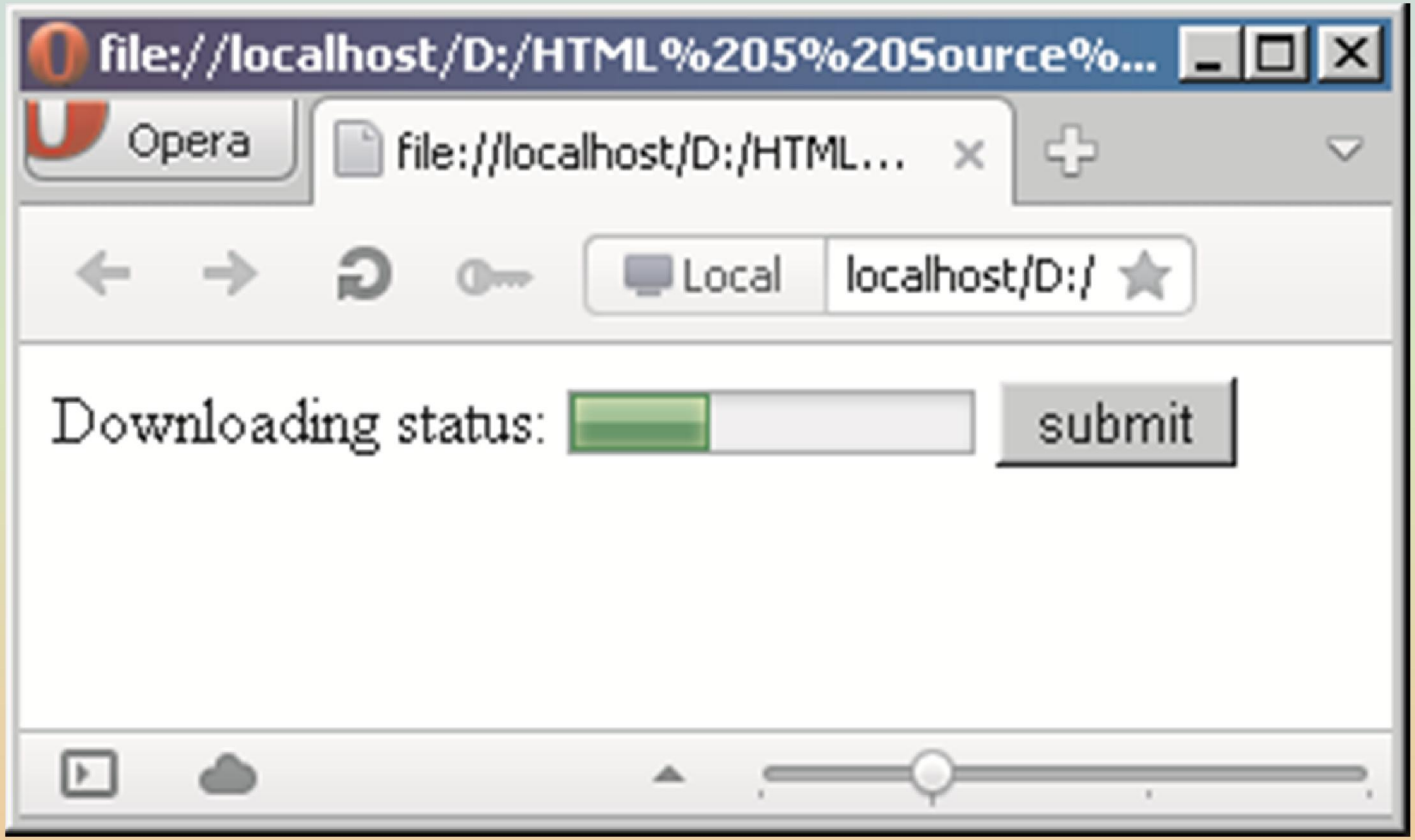
- The `progress` element represents the current status of a task, which gradually changes as the task heads for completion.
- This is not a form-specific element.
- For example, when the user downloads any file from a particular Web page, the download task is represented as a progress bar.
- The Code Snippet shows the syntax for providing `progress` element on the form.

```
<label> Downloading status: </label>
<progress value=”35” max=”100” ></progress>
<input type=”submit” value=”submit”/>
```

- As shown in the code snippet, the `progress` element contains two attributes namely, `max` and `value`.
- The `max` attribute declares the maximum value for the task to be processed for its completion.
- The `value` attribute indicates how much task has been processed so far.

Progress 2-2

- Following figure shows the `progress` element in Opera.



Meter 1-2

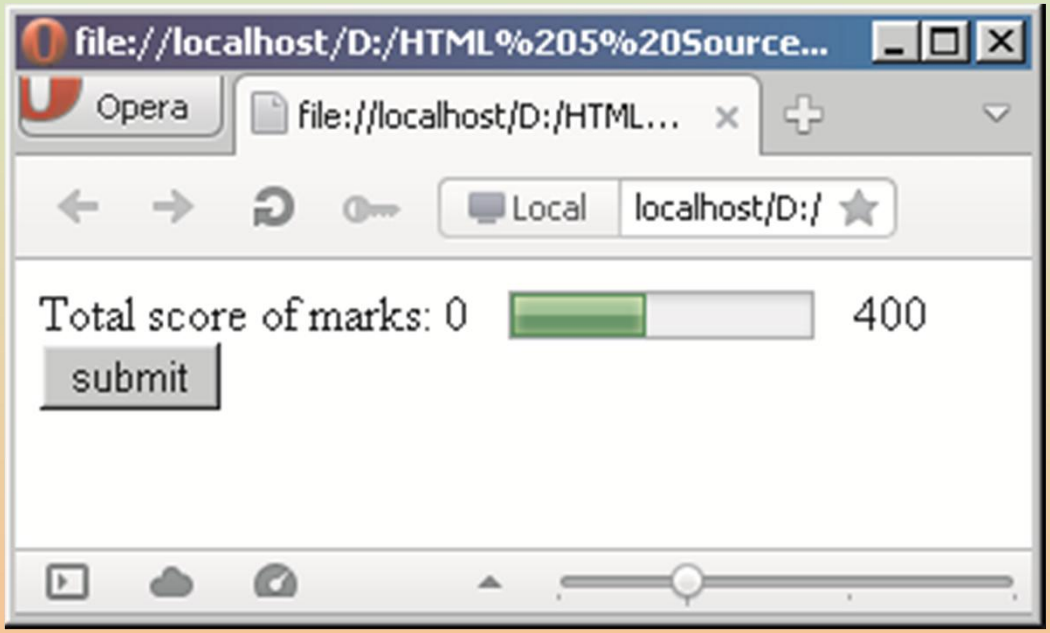
- The `meter` element represents a measurement scale for a known range.
- The known range has a definite minimum and maximum values to measure the data on the scale.
- For example, a meter element can be used to represent measurements, such as disk usage space, fraction value, or significance of a query result.
- All these have a known maximum value defined for them.
- The `meter` element cannot indicate age, height, or weight, as maximum values for them cannot be specified.
- The Code Snippet shows the code of the meter element.

```
<label> Total score of marks: </label>
0   <meter min="0" max="400" value="180"
       title="numbers scored" low="120" high="300">
      </meter>   400<br/>
<input type="submit" value="submit"/>
```

- In the code snippet, the meter element contains six attributes that are used to determine the measurements in the known range.
- The `min` and `max` attribute specifies the minimum and maximum value that sets bounds for the range.

Meter 2-2

- The default value for the max attribute is `1`.
- The value attribute specifies the current measured value.
- The high and low attributes specifies the range of values that can be considered as `high` or `low` for the given range.
- For example, in the given range of scores, the range of values below 120 will be considered `low`, but anything above 300 will be considered as `high`.
- There is another attribute named `optimum` which refers to the ideal value for the measurement.
- Following figure shows the `meter` element in Opera.



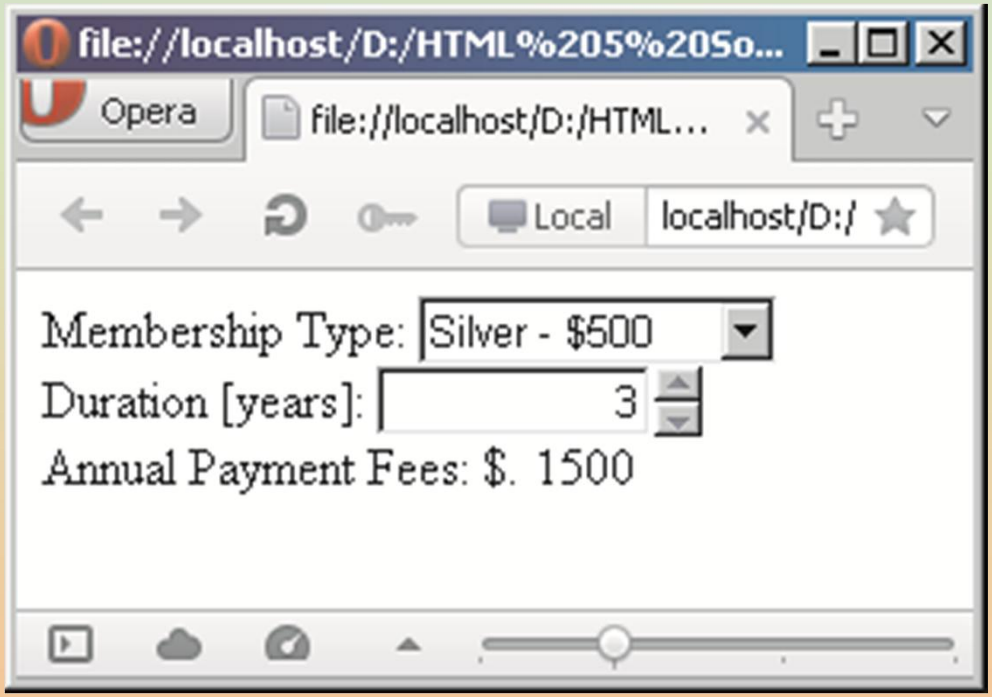
Output 1-1

- The `output` element displays the results of a calculation on a form.
- The result values displayed in the output element are processed from the other form elements.
- For example, the `output` element might be used to display the total cost on the purchase items after calculating discount amount in a registration form or purchase order form.
- The Code Snippet shows the calculation of data from other form elements to be displayed in the `output` element.

```
<form oninput="x.value = parseInt(type.value)*
 parseInt(duration.value)">
 <label>Membership Type:</label>
  <select name="type">
   <option value="400">Gold - $400</option>
   <option value="500">Silver - $500</option>
   <option value="600">Platinum - $600</option>
  </select>
 <label>Duration [years]:</label>
  <input type="number" value="0" name="duration"
      min="1"max="5" step="1" />
 <label> Annual Payment Fees: $.</label>
 <output name="x" for="type duration"></output>
```

Output 2-2

- In the code snippet, `for` attribute relates the `output` element with the elements whose values are taken for calculation.
- The form `oninput` event handles the `input` event which gets fired whenever the value of the elements change on receiving input from the user.
- A JavaScript code can also be written to update the values for the `output` element.
- Following figure shows the result of calculation for `output` element.



Summary

- HTML5 provides a great enhancement to Web forms.
- Creation of form is made easier for Web developers by standardizing them with rich form controls.
- HTML5 introduces new form elements such as new input types, new attributes, browser-based validation, CSS3 styling techniques, and forms API.
- HTML5 provides new input types that are data-specific user interface elements such as email, url, number, range, date, tel, and color.
- The new form elements introduced in HTML5 are datalist, progress, meter, and output.
- HTML5 has provided several new attributes that performs the validations without writing JavaScript snippets for them.
- In HTML5, one can use the submit input type for form submission.