Forward looking IT finishing school

Forms

Hypertext Markup Language 5 (HTML5)

Table of Content

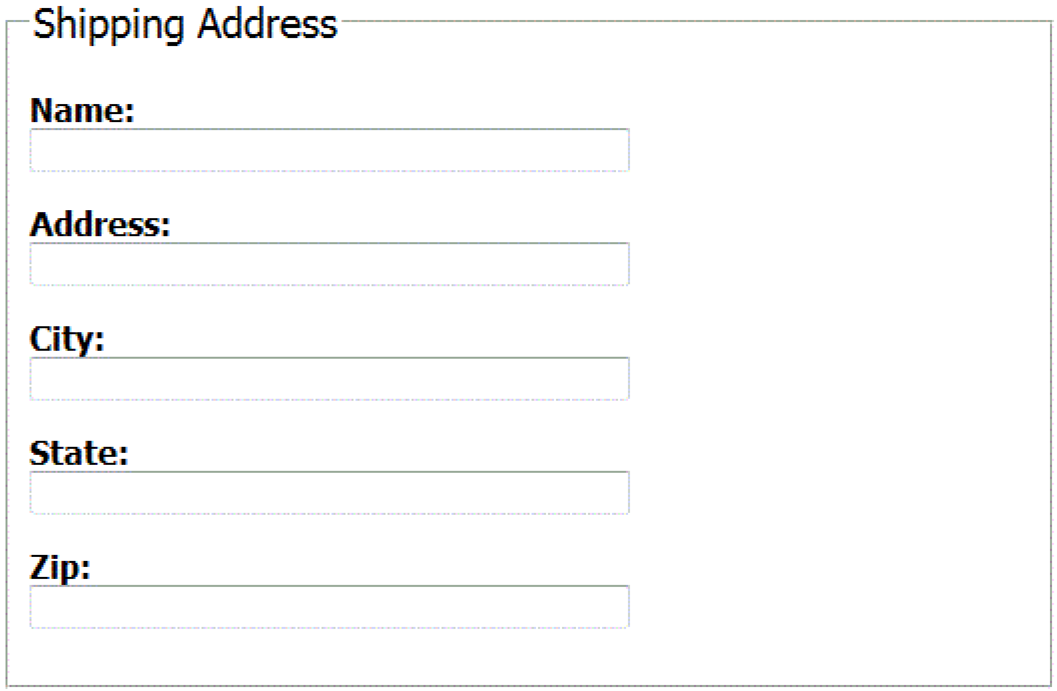
Forms

(Hypertext Markup Language 5)

Forms

- Forms are used to collect information from users
- Forms are defined by <form>......</form> tag
- Form elements are different type of input elements like text fields, check box, radio button, list box, submit buttons etc

Forms


Shipping Address

Name:

Address:

City:

State:

Zip:

Form submission

Syntax :
<form action = “url” method=“post”>

- Action attribute tells browser where to send the data
- Method attribute tells browser how to handle data
- The method attribute have two values (get | post)

Form submission

- GET and POST are used for submission of a form
- These methods are specified inside a FORM element using method attribute
- GET is default value of method attribute
- GET and POST primarily differ in terms of FORM data encoding

Method Attribute
(GET)

- Appends form-data into URL in name/value pairs (URL?name=value&name=value)
- Length of a URL is limited (about 3000 characters)
- DO NOT use GET to send sensitive data (data encoded in URL is visible to others)
- Useful for FORM submissions where a user want to bookmark the result
- GET is better for non-secure data, like query strings

Method Attribute
(POST)

- Appends form-data inside the body of the HTTP request (data is not shown is in URL)
- Has no size limitations
- Form submissions with POST cannot be bookmarked

Method Attribute
(GET vs POST)

GET

- Form data is to be encoded (by a browser) into a URL
- GET shall primarily be used for retrieving/query data
- Data is visible to everyone in URL
- Can be bookmarked

POST

- Form data is to appear within a message body
- POST shall be used for form submission (store/update data)
- Data is not displayed in the URL
- Cannot be bookmarked

Method Attribute
(GET vs POST)

GET

- The parameter data is limited
- GET is less secure compared to POST
- GET request is often cacheable
- Parameters are saved in browser history

POST

- Can send parameters including uploading files, to the server
- POST is little safer than GET
- POST usually not cached
- Parameters are not saved in browser history

Forms Example

```
<form action="url" method="get">
<label for="firstname">First Name : </label>
<input type="text" id="firstname" /> <br />
<label for="lastname">Last Name : </label>
<input type="text" id="lastname" /><br />
<input type="submit" value="Submit" />
</form>
```

Forms Example

```
<form action="url" method="get">
<label >First Name : <input type="text" /></label>
<br />
<label>Last Name : <input type="text" /></label>
<br />
<input type="submit" value="Submit" />
</form>
```

The <input> element

- The input fields are defined by the <input> element
- The type attribute defines which input state you are using
- The <input> element is an empty element, therefore, designed to be self-closing

The <label> element

- The <label> tag defines a label for an <input> element
- The <label> element does not render as anything special for the user
- But, improves usability by toggling the control, if user clicks on the text within the <label> element
- The “for” attribute of the <label> tag should be equal to the id attribute of the related element to bind them together
- A label can be bound to an element either by using the "for" attribute, or by placing the element inside the <label> element

The type attribute

- <input type="text">
  - Defines a one-line text input field
- <input type="radio">
  - Defines a radio button (to select one of many choices)
- <input type="submit">
  - Defines a submit button (to submit the form)

The type attribute

- Defining a password field
  - <input type=“password”>
- Defining a reset button that will reset all form values to their default values
  - <input type=“reset”>
- CheckBox is suitable choice when one or more items to be selected by user
  - <input type =”checkbox”>

Input Type CheckBox
(Example)

```
<form method="post">
<fieldset>
<legend>What is Your Favourite Pet?</legend>
<input type="checkbox" name="animal" value="Cat" />Cats <br />
<input type="checkbox" name="animal" value="Dog" />Dogs<br />
<input type="checkbox" name="animal" value="Bird" />Birds <br />
<input type="submit" value="Submit" />
<input type="reset" value="Reset"/>
</fieldset>
</form>
```

Input Type Radio
(Example)

```
<form>
<input type=“radio” name=“gender” value=“male” checked /> Male <br/>
<input type=“radio” name=“gender” value=“female”/> Female <br/>
</form>
```

The <select> element

- The <select> element defines a drop-down list
- The <option> element defines an option that can be selected
- The selected attribute is added to define a pre-selected option

The <select> element
(Example)

```
<select name=“Ice Cream Flavours”>
<option value=“Butter Scotch” selected>Butter Scotch</option>
<option value=“Vanilla”>Vanilla</option>
<option value=“Strawberry”>Strawberry</option>
</select>
```

The <textarea> element

- The <textarea> defines a multi-line input
- The rows attribute specifies the visible number of lines in a text area
- The cols attribute specifies the visible width of a text area

```
<textarea  rows=“5” cols=“20”>

</textarea>
```

Grouping Form Data

- The <fieldset> element is used to group related data in form
- The <legend> element defines a caption for the fieldset element

Grouping Form Data
(Example)

```
<form>
<fieldset>
<legend>User info</legend>
<label for="username">Username:</label>
<input type="text" id=”username”><br />
<label for="password">Password:</label>
<input type="password" id=”username”><br /></fieldset>
<input type="submit" value="Submit" />
</form>
```

The <button> element

- The button element defines a clickable button

Example :

```
<button type=“button” onclick=“alert('welcome to html5 world')”>click me
</button>
```

Form Elements

- HTML5 added some new Form Elements
  - datalist
  - keygen
  - output

The <datalist> Element

- The <datalist> element specifies a list of pre-defined option for an <input> element
- These pre-defined option user can see in drop-down list
- The list attribute of the <input> element, must refer to the id attribute of the <datalist> element
- The <datalist> element bind together by using <input> element

The <datalist> Element
(Example)

```
<input list=“states”/>
<datalist id=“states”>
<option value=“Jharkhand”>
<option value =“Karnataka”>
<option value =“Uttar Pradesh”>
<option value=“Tamilnadu”>
</datalist>
```

The <output> element

- The <output> element contains result of any calculation

The <output> element

(Example)

```
<form oninput="y.value=parseInt(b.value)
+parseInt(c.value)">0
<input type="range" name="b" value="55" />150
+<input type="number" name="c" value="55" />=<output
name="y" for="b c"></output>
</form>
```

Date

- Date element provides a handy drop down calendar to pick the date from
- Date element provides six different ways of defining dates
  - Date
  - Month
  - Week
  - Time
  - date+time
  - date +time-time zone

Date

- The <input type=“date”> is used for input fields that should contain a date
- **Example**

```
<form>
Enter a date after  2005-01-01
<input type=“date” name=“joining-date” min=“2005-02-02”>
</form>
```

Time

- <input type =”time”> allows user to select a time (no time zone)
- **Example**

```
<form>
Select Time :
<input type=“time” name=“user-time”>
<input type=”submit”>
</form>
```

Color

- The <input type= “color”> is used for input fields that should contain a color
- Example

```
<form>

Select Your favorite Color :

<input type=“color” name=“colorpicker”>

</form>
```

Email

- Email can be automatically validated at submission
- The <input type=”email”> is used for input fields that should contain an email address
- **Example**

```
<form>
Email : <input type=“email” name=“email”>
<input type=“submit”>
</form>
```

Validation

(Hypertext Markup Language 5)

Form Validation

- The form validation means ensuring
  - Required/mandatory fields are filled
  - Inputs such as date, email, phone number, password are filled in correct format with expected characters
- After validating all input data, form is submitted to server and saved in database
- Client-side validation is validation that occurs in the browser

Form Validation

- Client-side validation is sub-divided as
  - JavaScript validation
  - Built-in form validation
- JavaScript validation is coded using JavaScript
- Built-in form validation is done with HTML5 form validation features

Form Validation Features

- Input validation can be applied by choosing appropriate type attribute value
  - Example : <input type=“email”>
- Specifying validation related attributes
  - pattern, min, max, required, maxlength
- Type mismatch constraint violation is triggered for invalid input data

Number

- The <input type=“number”> defines a numeric input field
- **Example**

```
<form>
Marks (between 50 and 90):
<input type=“number” name=“Marks”  min=“50” max=“90”>
</form>
```

***Here min and max are input restrictions**

Attributes

(Hypertext Markup Language 5)

The autofocus attribute

- The autofocus attribute specifies the input field should get automatically get focus when the page loads
- Example

  <input type=“text” name=“search” autofocus/>

The Required Attribute

- The Required attribute specifies that an input field must be filled out before submitting the form
- Example

  <input type=“text” name=“search” required/>

The placeholder Attribute

- The placeholder attribute specifies a hint that describes the expected value of an input field
- The hint is displayed in the input field before the user enters a value
- Placeholder is only applicable to email, number, password, search, tel, text or URL

The placeholder Attribute

```
<form>
First name : <input type="text" name="firstname" placeholder="Jane"/>
<br>
Surname : <input type="text" name="surname" placeholder="Doe"/>
<br>
<input type="submit" value="Submit" />
</form>
```

The autocomplete Attribute

- The autocomplete Attribute specifies whether a form or input field should have autocomplete on or off
- When autocomplete is on, the browser automatically complete the input values based on values that the user has entered before

The autocomplete Attribute

```
<form method=“post” autocomplete=“on”>
User Name : <input type=“text” name=“username”> <br>
Password : <input type =“password” name=“passwd”
autocomplete=“off”><br>
<input type =“submit”>
</form>
```

The Accept Attribute

- It is used to specify types of files that the server accepts
- It is usually used to upload files
- Accept attribute works with input file (<input type="file">)
- More than one values are separated with comma
  - Example <input accept="audio/*,video/*,image/*" />

Syntax :

<input accept="file_extension|audio/*|video/*|image/*|media_type">

The Accept Attribute

```
<form>
            <label> Select file :
                <input type="file" id="user-file" accept=".pdf, .doc">
            </label>
            <br /> <br />
            <input type="submit" value="Submit">
</form>
```

WSA | Forward looking IT finishing school

Thank you

Web Stack Academy (P) Ltd

#83, Farah Towers,

1st floor,MG Road,

Bangalore – 560001

M: +91-80-4128 9576

T: +91-98862 69112

E: info@www.webstackacademy.com