

Forms

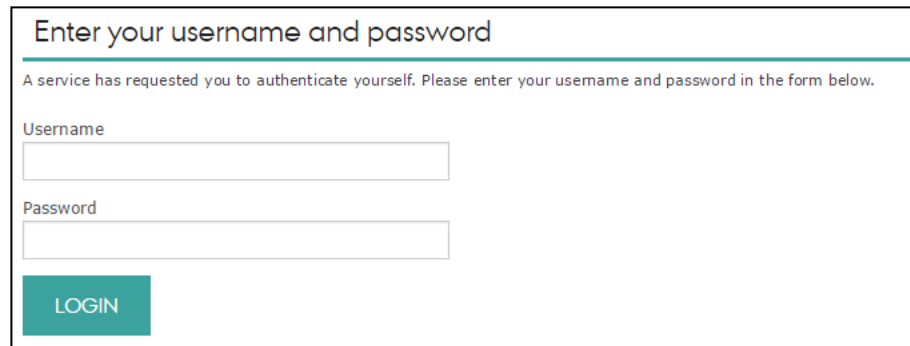
Web Development with HTML5

Agenda

- Overview
- Basic form controls
- Accessibility & Forms
- Client-side validation
- Styling a Form
- Server-Side Processing
- New in HTML5

Overview of Forms

- Forms are used all over the Web to
 - Accept information
 - Provide interactivity



Enter your username and password

A service has requested you to authenticate yourself. Please enter your username and password in the form below.

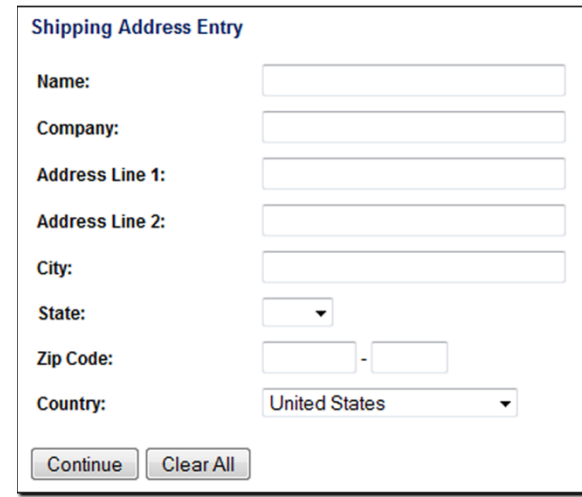
Username

Password

LOGIN

Overview of Forms - 2

- Form
 - An HTML element that contains and organizes form controls
 - such as:
 - text boxes
 - check boxes
 - buttons
 - that can accept information from website visitors



Shipping Address Entry

Name:

Company:

Address Line 1:

Address Line 2:

City:

State:

Zip Code: -

Country:

Two Components of Using Forms

1. The HTML form

- the web page user interface
- accepts input from the user
- sends the data to the server when the user presses the submit button

2. The server-side processing

- When the server receives the form data it calls the specified action (a method or script) that process the data
- E.g. it could:
 - **update a database**
 - write to a file
 - send e-mail
 - or performs some other type of processing on the server

Sample Form HTML

```
<h1>Join Our Newsletter</h1>
<form method="post"
  action="http://mysite.net/newsletter/join.php"
  >
  E-mail: <input type="text" name="email" id="email" >
  <br>
  <br>
  <input type="submit" value="Sign Me Up!"> <input
  type="reset">
</form>
```



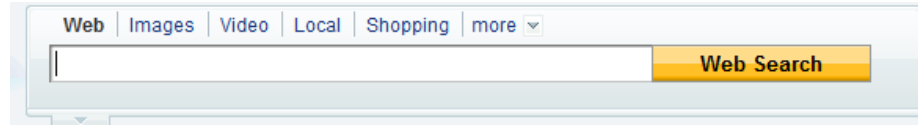
HTML form element

- The form element attributes:
 - action
 - Specifies the server-side program or script that will process your form data
 - method
 - get – default value,
form data passed in URL
 - post – more secure,
form data passed in request Body
 - name
 - Identifies the form
 - id
 - Identifies the form

GET, POST, and HTTP Verb Safety

- An important distinction between these verbs is that a GET operation is not supposed to change anything on the server
 - or to put it in a slightly more abstract way, a GET operation does not result in a change in state on the server
 - You can perform a GET operation on the same resources as many times as you like, and those resources don't change
- A POST request changes something on the server each time you perform the operation

To GET vs. POST



- **GET**

When you perform a search using an engine like Bing or Google, you fill in a form that consists of one text box, and then you click the search button.

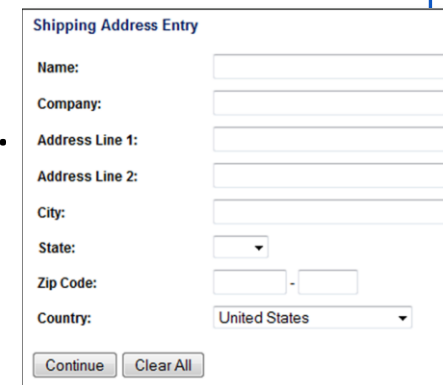
The browser performs a **GET** operation, with the value you entered into the box passed as part of the URL.

Using a GET operation for this type of form is fine, because a search operation doesn't change any resources on the server, it just fetches information.

- **POST**

Now consider the process of ordering something online. You fill in the order details and then click the submit button.

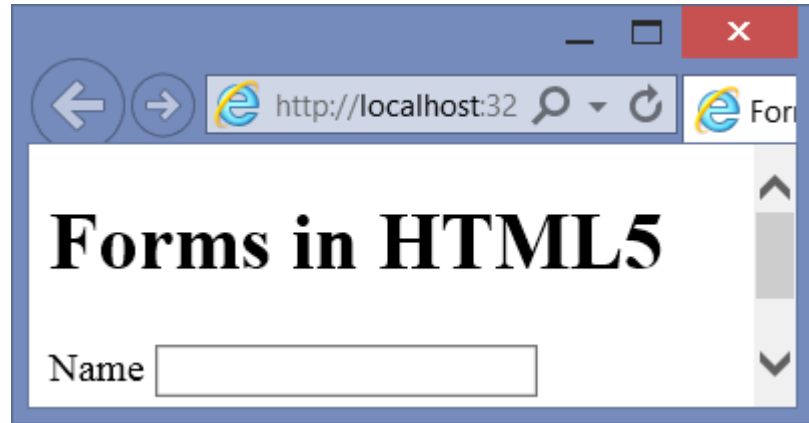
This operation will be a **POST** request, because the operation will result in changes on the server, such as a new order record, a change in your account information, and perhaps many other changes.



BASIC FORM CONTROLS

Input Text box

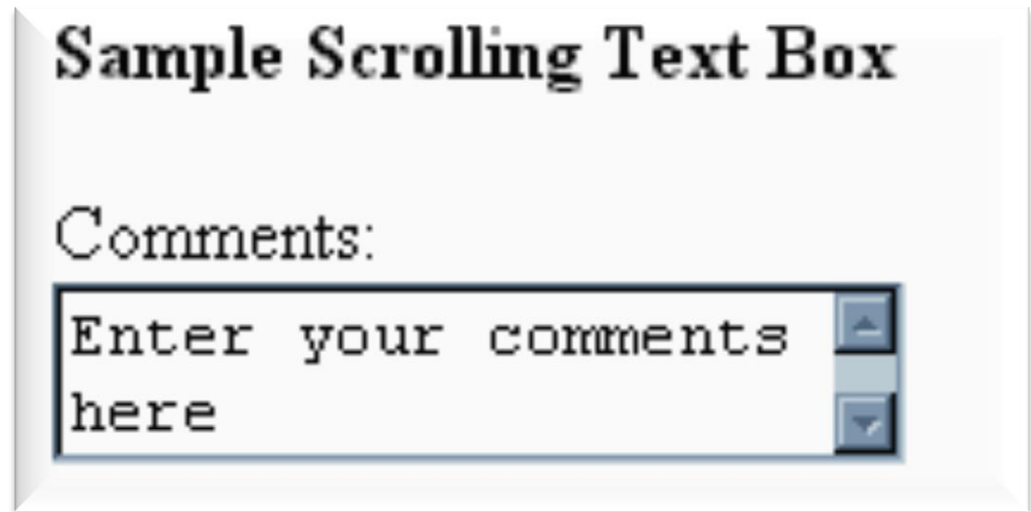
- `<input type="text">`
Accepts text information
- Attributes:
 - name
 - id
 - size
 - maxlength
 - value
 - placeholder



```
<input type="text" id="name" name="name" />
```

textarea - Scrolling Text Box

- `<textarea> </textarea>`
Configures a scrolling text box
- Attributes:
 - name
 - id
 - cols
 - rows



input Submit Button

- `<input type="submit">`
Submits the form information ~ *an enter button*
- When clicked:
 - Triggers the action method on the `<form>` tag
 - Sends the form data (the name=value pair for each form element) to the web server
- Attributes:
 - name
 - id
 - value

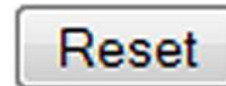
Sample Submit Button



input Reset Button

- `<input type="reset">`
Resets the form fields to their initial values
- Attributes:
 - name
 - id
 - value

Sample Reset Button



input Password box

- `<input type="password">`
Accepts text information that needs to be hidden as it is entered
- Attributes:
 - name
 - id
 - maxlength
 - value



A sample password box interface. It features a title "Sample Password Box" in a bold, serif font. Below the title, the word "Password:" is followed by a rectangular input field. The input field contains a series of asterisks, indicating that the password is hidden. The entire interface is enclosed in a light gray box with a subtle drop shadow.

input Check box

- `<input type="checkbox">`

Allows the user to select one or more of a group of predetermined items

- Attributes:

- name
- id
- checked
- value

Sample Check Box

Choose the browsers you use:

- ☐ Internet Explorer
- ☐ Firefox
- ☐ Opera

input Radio Button

- `<input type="radio">`
Allows the user to select exactly one from a group of predetermined items
- Each radio button in a group is given the same name and a unique value
- Attributes:
 - name
 - id
 - checked
 - value

Sample Radio Buttons

Select your favorite browser:

- ☐ Internet Explorer
- ☐ Firefox
- ☐ Opera

input Hidden form data

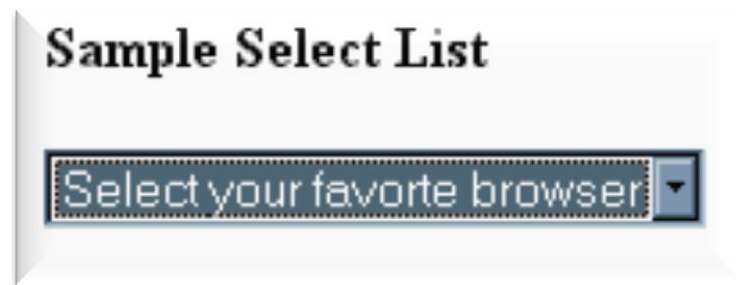
- `<input type="hidden">`

This form control is not displayed on the web page

- Hidden form fields
 - Can be accessed by both client-side and server-side scripting
 - Sometimes used to contain information needed as the visitor moves from page to page
- Attributes:
 - name
 - id
 - value

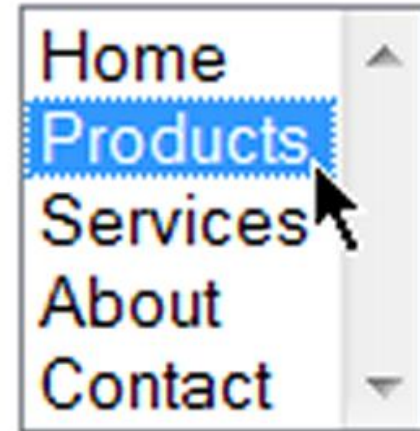
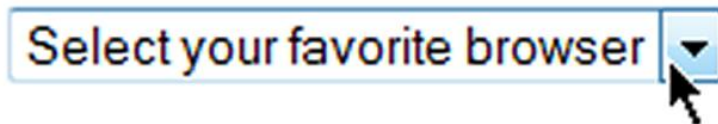
Select List

- `<select></select>`
Configures a select list
- Also known as: Select Box, Drop-Down List, Drop-Down Box, and Option Box
- Allows the user to select one or more items from a list of predetermined choices.
- Attributes:
 - name
 - id
 - size
 - multiple



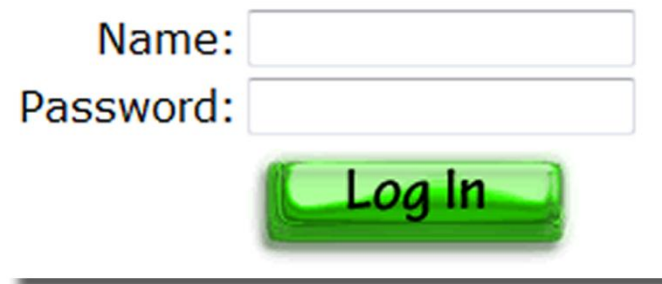
Options in a Select List

- `<option></option>`
Configures the options in a Select List
- Attributes:
 - value
 - selected



Input Image Button

- `<input type="image">`
Submits the form
- When clicked:
 - Triggers the action method on the form tag
 - Sends the form data (the name=value pair for each form element) to the web server
- Attributes:
 - name
 - id
 - src



A login form with the following elements:

- A label "Name:" followed by a text input field.
- A label "Password:" followed by a text input field.
- A green, rounded rectangular button with the text "Log In" in black.

Button Element

- `<button type="button"></button>`
A container tag
- When clicked, its function depends on the value of the type attribute
- Can contain a combination of text, images, and media
- Attributes:
 - `type="submit", "reset", or "button"`
 - `name`
 - `id`
 - `alt`
 - `value`

Name:

E-mail:



Sign up for free newsletter

ACCESSIBILITY & FORMS

Label Element
Fieldset Element
Legend Element
Placeholder attribute
TabIndex Attribute
Accesskey Attribute

Label element

- `<label></label>`

Associates a text label with a form control

- Two Different Formats:

```
<label>Email: <input type="text" name="email"
  id ="email"></label>
```

– Or

```
<label for="email">Email: </label>
<input type="text" name="CustEmail" id= "email">
```


Fieldset and Legend Elements

- The Fieldset Element
 - Container tag
 - Creates a visual group of form elements on a web page
- The Legend Element
 - Container tag
 - Creates a text label within the fieldset

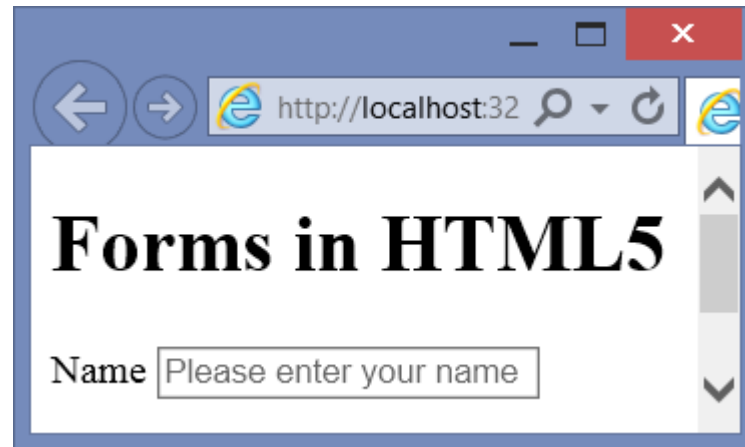


```
<fieldset><legend>Customer Information</legend>
  <label>Name:
  <input type="text" name="Name" id="Name"></label>
  <br><br>
  <label>Email:
  <input type="text" name="Email" id="Email"></label>
</fieldset>
```

placeholder attribute

- The placeholder attribute on <input> and <textarea> elements provides a hint to the user of what can be entered in the field
 - The placeholder text must not contain carriage returns or line-feeds

```
<input type="text" id="name" name="name"  
      placeholder="Please enter your name"/>
```



tabindex attribute

- Attribute that can be used on form controls and anchor tags
- Modifies the default tab order
- Assign a numeric value

```
<input type="text"  
       name="CustEmail"  
       id="CustEmail"  
       tabindex="1"
```

```
>
```

accesskey attribute

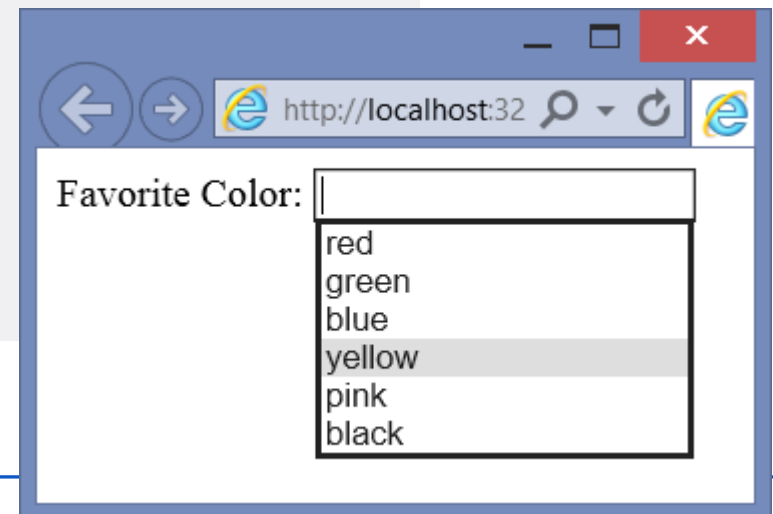
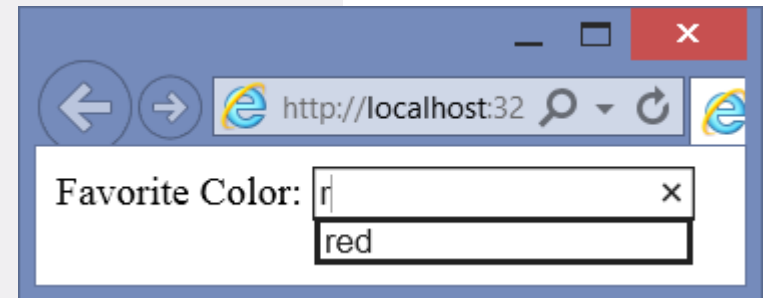
- Attribute that can be used on form controls and anchor tags
- Create a “hot-key” combination to place the focus on the component
- Assign a value of a keyboard letter
- On Windows use the CTRL and the “hot-key” to move the cursor

```
<input type="text" name="CustEmail" id="CustEmail"  
accesskey="E" />
```

<datalist> element

- The <datalist> element represents the list of <option> elements to suggest when filling an <input> field

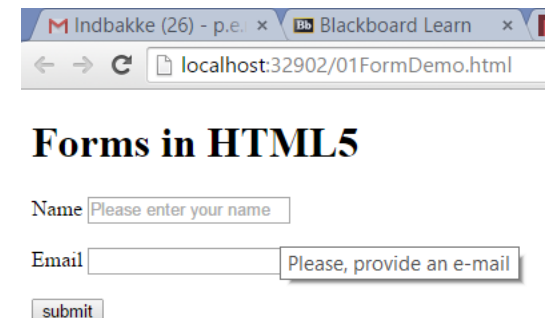
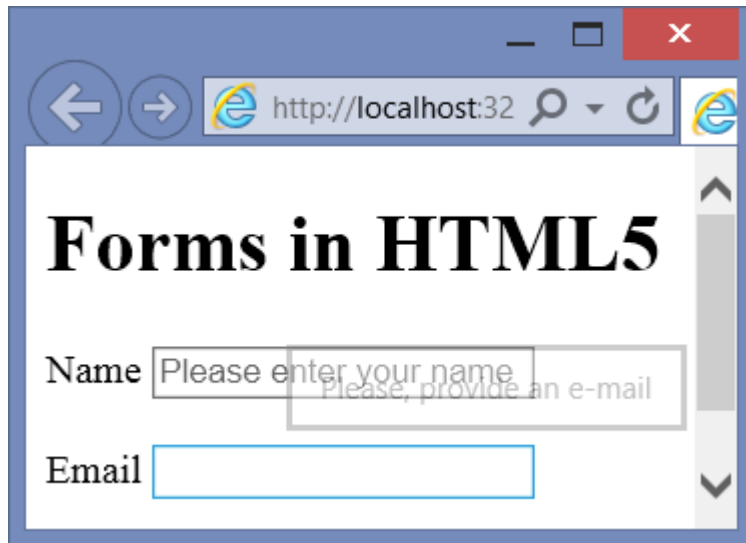
```
<label for="color">Favorite Color:</label>
<input type="text" name="color" id="color"
list="colors" >
<datalist id="colors">
  <option value="red">
  <option value="green">
  <option value="blue">
  <option value="yellow">
  <option value="pink">
  <option value="black">
</datalist>
```



title attribute

- If the title attribute is set on the <input> element, its value is used as tooltip
- If the validation fail, this tooltip text will be replaced with the associate error message

```
<input type="email" title="Please, provide an e-mail" />
```



CLIENT-SIDE VALIDATION

While this functionality does not replace server-side validation, which is still necessary for security and data integrity, client-side validation can support a better user experience

required attribute

- The required attribute on the <input>, <select>, and <textarea> elements indicates that a value must be supplied

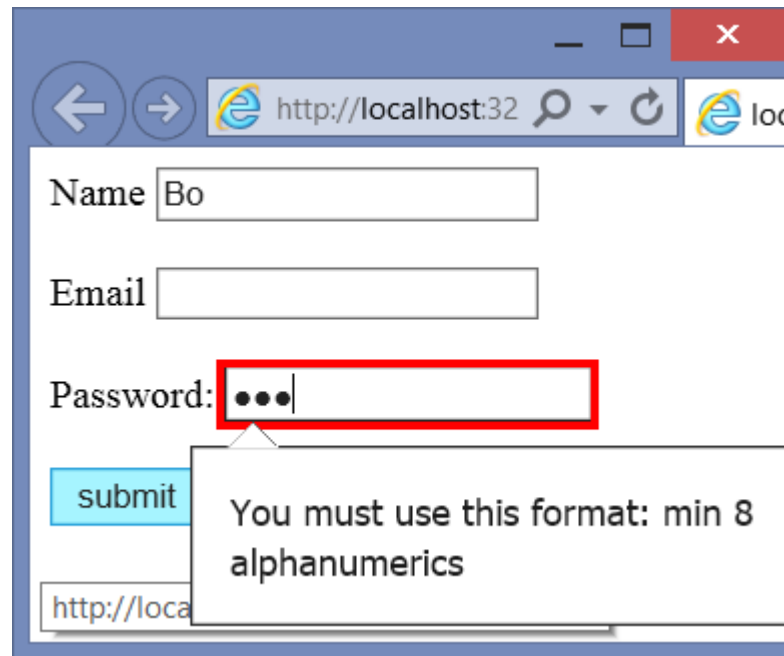
```
<input type="email" required />
```

The screenshot shows a web browser window with the title "Forms in HTML5". The browser's address bar displays "http://localhost:32". The page content includes a form with a "Name" label and an input field. The input field has a red border, indicating it is a required field. Below the "Name" field is a "submit" button. A tooltip message "This is a required field" is displayed over the "Name" input field. The browser's status bar at the bottom shows the URL "http://localhost:32902/acknowledge.html".

pattern attribute

- The pattern attribute on the <input> element constrains the value to match a specific regular expression

```
<input type="password" id="password" name="password"
      required pattern="[\w]{8,}" />
```



The screenshot shows a web browser window with a form. The form has three input fields: 'Name' (containing 'Bo'), 'Email' (empty), and 'Password' (containing three dots). The 'Password' field is highlighted with a red rectangular border. Below the 'Password' field, there is a blue 'submit' button. A tooltip message is displayed over the 'submit' button, stating: 'You must use this format: min 8 alphanumerics'. The browser's address bar shows 'http://localhost:32'.

maxlength attribute

- The maxlength attribute of the <input> and <textarea> elements constrains the maximum number of characters that the user can enter

min, max, and step attributes

- The min and max attributes of the <input> element constrain the minimum and maximum values that can be entered
- The step attribute of the <input> element constrains the granularity of values that can be entered
 - Can only be used together with min and max

STYLING A FORM

```
</table>
</form>
```

Name:

E-mail:

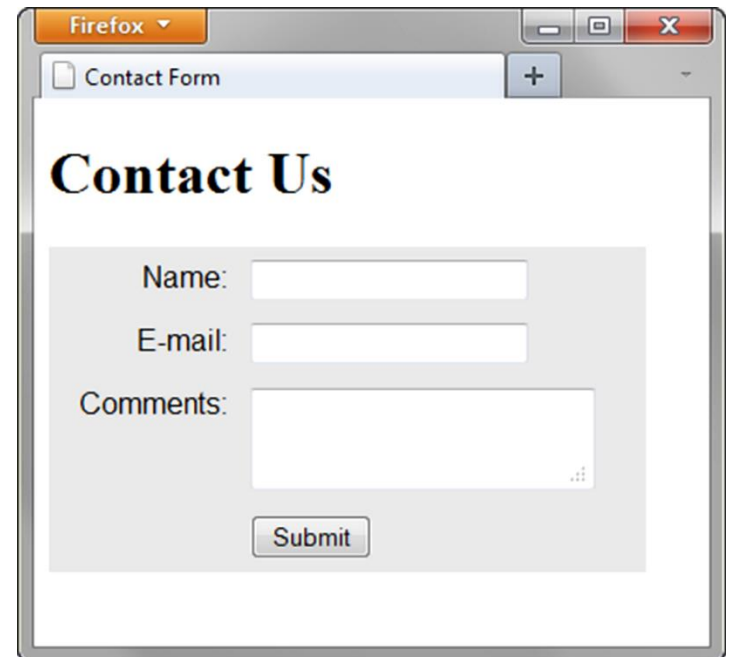
Comments:

Old approach - Do not use!

Using CSS to Style a Form

- Transitional Approach
 - Use a table to format the form but configure styles instead of HTML table attributes

```
table { background-color: #eaeaea;  
        border-style: none;  
        width: 20em;  
        font-family: Arial, sans-serif; }  
th { font-weight: normal;  
      text-align: right;  
      vertical-align: top; }  
td, th {padding: 5px; }
```



The screenshot shows a Firefox browser window with a tab titled 'Contact Form'. The page content features a heading 'Contact Us' in a bold, serif font. Below the heading is a light gray rectangular box containing the form fields. The form has three input fields: 'Name:', 'E-mail:', and 'Comments:'. The 'Comments:' field is a larger text area. At the bottom of the form box is a 'Submit' button. The browser's address bar and window controls are visible at the top.

Using CSS to Style a Form

form

label text box

label text box

label scrolling text box

submit button

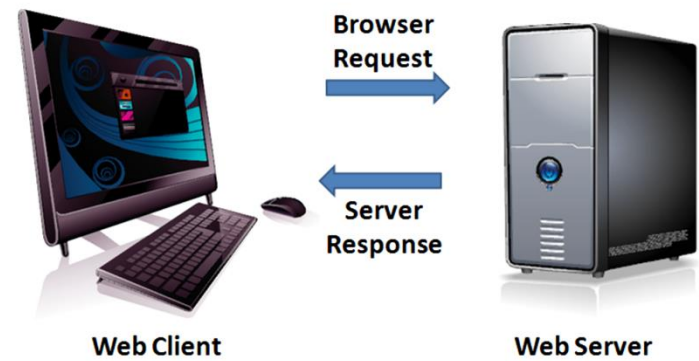
- “Pure” CSS Approach
 - Do not use a table to format the form
 - Use CSS float and display: block

```
form { background-color:#eaeaea; width: 350px;
      font-family: Arial, sans-serif; padding: 10px;
}
label { float: left; width: 100px; display: block;
       clear: left; text-align: right; padding-right: 10px;
       margin-top: 10px;
}
input, textarea { margin-top: 10px; display: block; }
#mySubmit { margin-left: 110px; }
```

SERVER-SIDE PROCESSING

Server-Side Processing

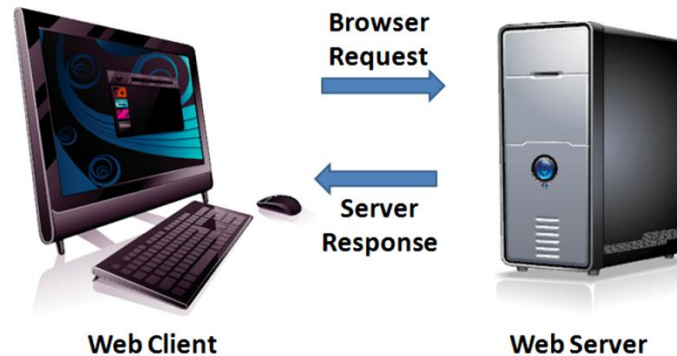
- Your web browser requests web pages and their related files from a web server
- The web server locates the files and sends them to your web browser
- The web browser then renders the returned files and displays the requested web pages for you to use



Server-Side Scripting

- One of many technologies in which a server-side script is embedded within a Web page document saved with a file extension such as:

- .php (PHP)
- .asp (Active Server Pages)
- .cfm (Adobe ColdFusion)
- .jsp (JavaServer Pages)
- .aspx (ASP.Net - forms)



- Uses direct execution — the script is run either by the web server itself or by an extension module to the web server

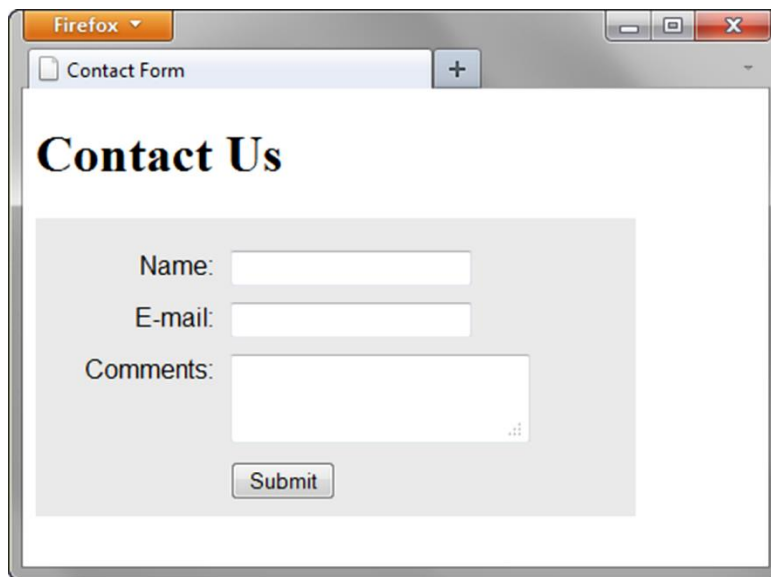
Steps in Utilizing Server-Side Processing

1. Web page invokes server-side processing by a form or hyperlink
2. Web server executes a server-side script
3. Server-side script accesses requested database, file, or process
4. Web server returns web page with requested information or confirmation of action

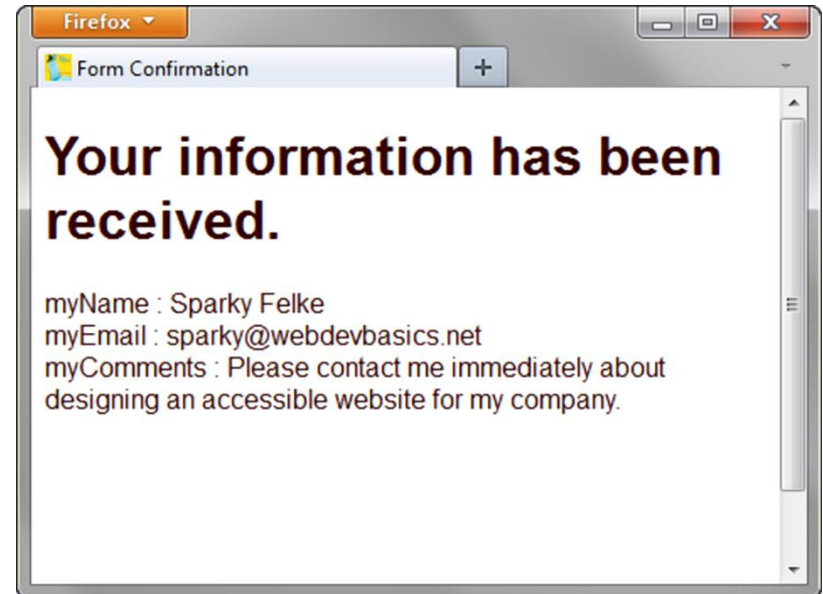


Sending information to a Server-side Script

```
<form method="post"
      action="http://webdevbasics.net/scripts/demo.php ">
```



A screenshot of a Firefox browser window titled "Contact Form". The page has a heading "Contact Us" and a form with three input fields: "Name:", "E-mail:", and "Comments:". Below the fields is a "Submit" button. The form is set against a light gray background.



A screenshot of a Firefox browser window titled "Form Confirmation". The page displays a large heading "Your information has been received." followed by a confirmation message: "myName : Sparky Felke", "myEmail : sparky@webdevbasics.net", and "myComments : Please contact me immediately about designing an accessible website for my company."

Sources of Free Server-Side Processing

- Many web host providers offer free scripts for their clients. Contact their support area or FAQ to learn more about their services
- Some web sites that offer FREE remotely hosted scripts (in return for displaying an ad)
 - <http://formbuddy.com>
 - <http://response-o-matic.com>
 - <http://master.com>
 - <http://www.formmail.com>
 - <http://wufoo.com>
 - <http://icebrrg.com>
 - <http://formassembly.com>

Server-Side Scripting Technologies

- ASP.Net - Active Server Pages

<http://www.asp.net/>

Tutorial: <http://www.asp.net/web-pages/tutorials/introducing-aspnet-web-pages-2/getting-started>

- PHP

<http://www.php.net>

- Java Server Pages

<http://java.sun.com/products/jsp>

- Ruby on Rails

<http://www.rubyonrails.org> or <http://tryruby.hobix.com>

NEW IN HTML5

HTML5: Email Text Box

- `<input type="email">`

Accepts text information in e-mail address format – and validates

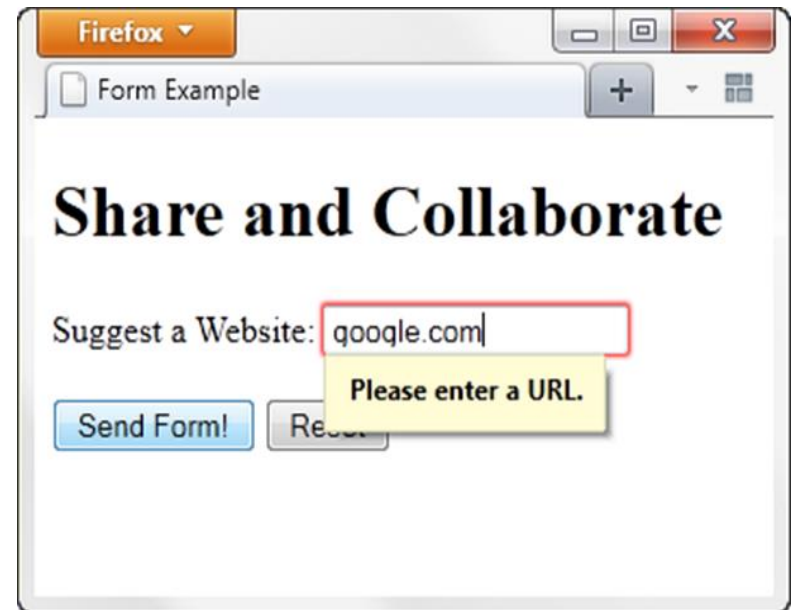
- Common Attributes:

- name
- id
- size
- maxlength
- value
- placeholder
- required



HTML5: URL Text Box

- `<input type="url">`
Accepts text information in URL format – and validates
- Common Attributes:
 - name
 - id
 - size
 - maxlength
 - value
 - placeholder
 - required



HTML5: Telephone Number Text Box

- `<input type="tel">`

Accepts text information in telephone number format – and validates

- Common Attributes:

- name
- id
- size
- maxlength
- value
- placeholder
- required

HTML5: Search Text Box

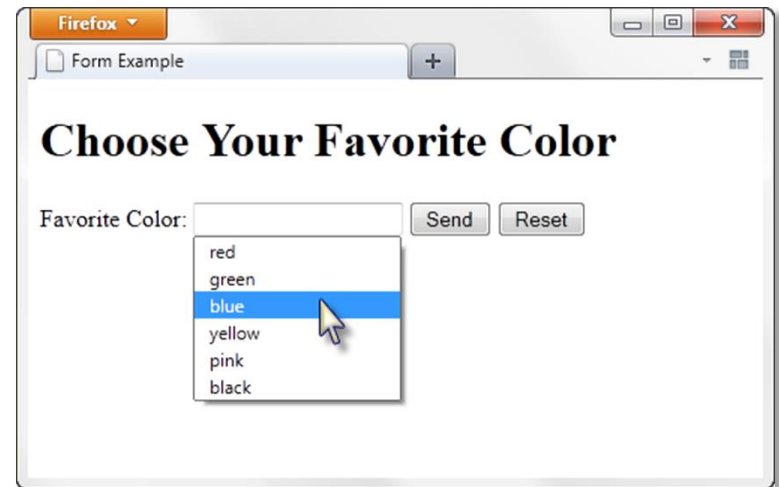
- `<input type="search">`

Accepts search terms

- Common Attributes:

- name
- id
- size
- maxlength
- value
- placeholder
- required

HTML5: Datalist Control

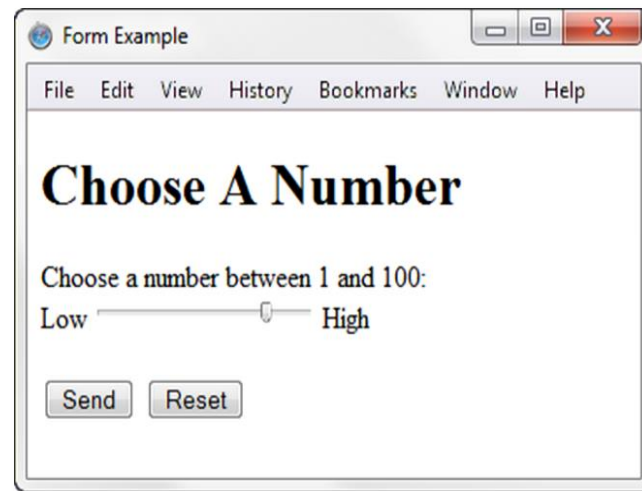


HTML5: Slider Control

`<label for="myChoice">`

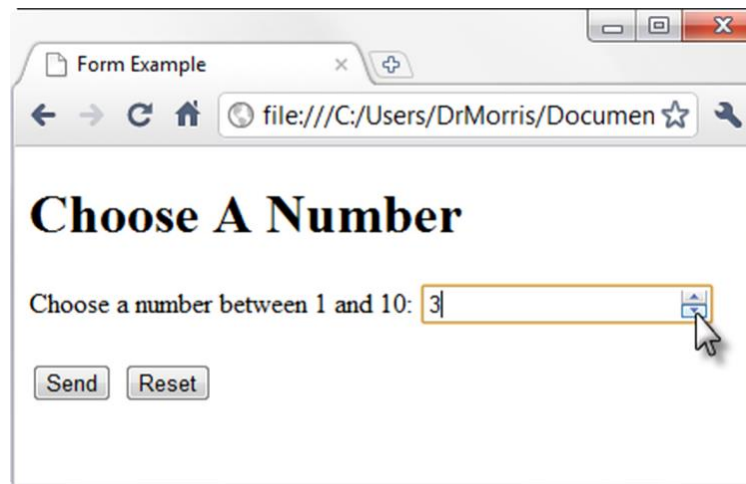
Choose a number between 1 and 100:`</label>
`

Low `<input type="range" name="myChoice" id="myChoice">` High



HTML5: Spinner Control

```
<label for="myChoice">Choose a number between 1 and 10:</label>  
<input type="number" name="myChoice" id="myChoice"  
  min="1" max="10">
```

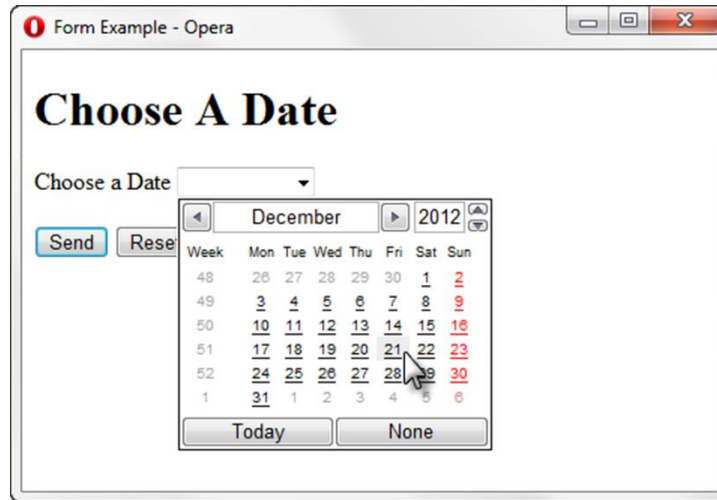


The screenshot shows a web browser window with a single tab titled 'Form Example'. The address bar shows a local file path: 'file:///C:/Users/DrMorris/Documen'. The main content area has a heading 'Choose A Number' in bold. Below the heading is a form with the text 'Choose a number between 1 and 10:' followed by a text input field containing the number '3'. To the right of the input field is a small blue spinner control. Below the input field are two buttons: 'Send' and 'Reset'. A mouse cursor is pointing at the spinner control.

HTML5: Calendar Control

`<label for="myDate">Choose a Date</label>`

`<input type="date" name="myDate" id="myDate">`



Practice with an HTML5 Form

- The form display and functioning varies with browser support.



Comment Form - Opera

Send Us Your Comments

Required fields are marked with an asterisk *.

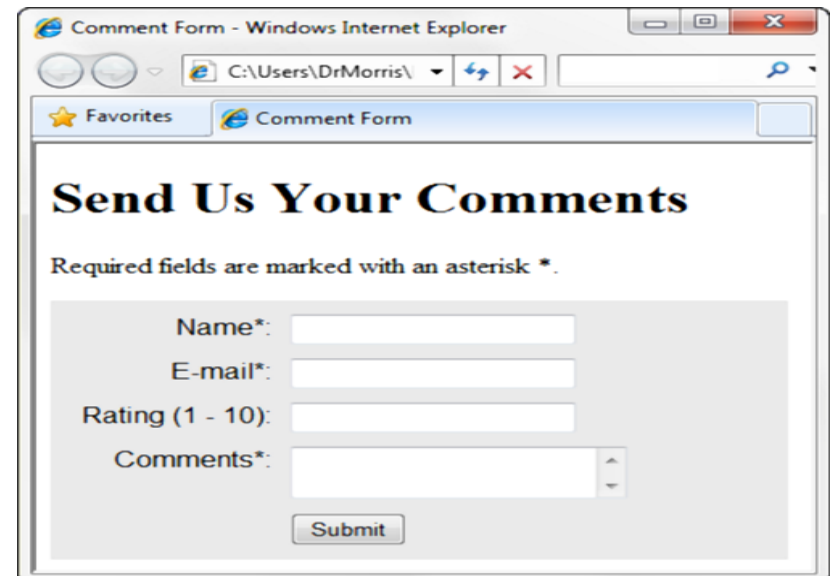
Name*: your first and last name

E-mail*: you@yourdomain.com

Rating (1 - 10):

Comments*:

Submit



Comment Form - Windows Internet Explorer

Send Us Your Comments

Required fields are marked with an asterisk *.

Name*:

E-mail*:

Rating (1 - 10):

Comments*:

Submit

References and Links

- **"Web Development and Design Foundations with HTML5"** by Terry Felke-Morris, sixth edition, isbn13: 9780273774501
- Forms in HTML
[https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms in HTML](https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms_in_HTML)
- Beginner's Guide to HTML5 & CSS3 - Formidable Forms with HTML5
<http://www.codeproject.com/Articles/761123/Beginners-Guide-to-HTML-CSS-Formidable-Forms-with>
- HTML Input Types
http://www.w3schools.com/html/html_form_input_types.asp
- **How To Build An Awesome Form**
<https://medium.com/@kubachrzecijanek/how-to-build-an-awesome-form-1e9b2c1bd00d>