Tutorial 6: HTML Forms

An example of an HTML form

1

Form components and elements

text box

This figure shows a form that contains various control elements commonly used in Web page forms.

drop-down list box

group box

radio buttons

check boxes

text area

form button

First Name		Last Name	
Address #1			
Address #2			
City		State	Zip
Country			
Item Purchased	▼	Purchase Date	
		Serial Number	

Used For (check one)
○ Home
○ Business
○ Religious or Charitable Institution
○ Government
○ Educational Institution

Network Operating System (check all that apply)
☐ Netware
☐ Banyan Vines
☐ Windows
☐ IBM Lan Server
☐ PC/NFS

Comments?:

Send Registration   Cancel

Form control elements

❑ Control elements that are commonly used:
- o **text boxes** for text and numerical entries
- o **selection lists** for long lists of options, usually appearing in a **drop-down list box**
- o **radio buttons**, also called **option buttons**, to select a single option from a predefined list
- o **check boxes** to specify an item as either present or absent, can have multiple choices
- o **groups boxes** to organize form elements
- o **text areas** for extended entries that can include several lines of text
- o **buttons** that can be clicked to start processing the form

Form control elements (cont.)

❑ Each control element in which the user can enter information is called a **field**

❑ Information entered into a field is called the **field value**, or simply the **value**

❑ In some fields, users are free to enter anything they choose

❑ Other fields, such as selection lists, limit the user to a predefined list of options

The **<form>** tag

❑ The **<form>** tag identifies the beginning and end of a form

❑ A single page can include several different forms, but you cannot nest one form inside another

❑ The general syntax of the **<form>** tag is:

```
<form attributes>
   form elements and layout tags
</form>
```

❑ Between the **<form>** and **</form>** tags, place the various tags for each of the fields in the form

The <form> tag (cont.)

❑ A single Web page can contain multiple forms, the **<form>** tag includes the **name** or **id** attribute.

❑ name/id attribute identifies each form on the page.

❑ name/id attribute is also needed for programs that retrieve values from the form.

Adding the <form> tag

The **<form>** tag includes attributes that control how the form is processed, including information on what CGI script to use, how the data is to be transferred to the script, and so forth.

```
<html>
<head>
<title>LanGear Registration Form</title>
</head>
<body text="#850000">
<form name="reg">




</form>
</body>

</html>
```

Creating a field set

- A **fieldset** groups similar fields together
- When rendered by browser, a field set appears as a box surrounding the fields
- Example

```
<fieldset id="contact">
  <legend align="value">contact
  information</legend>
</fieldset>
```

- the `<legend>` tag is used to display a legend for a field set
- *legend text* specifies the text for that legend

The `<input>` tag

- `<input>` tag specifies particular input types for a form
- syntax

```
<input type="type">
```

Input types

Type	Description	
type="button"	Display a button which can be clicked to perform an action from a script	button
type="checkbox"	Display a check box	☑
type="file"	Display a browse button to locate and select a file	Browse...
type="hidden"	Create a hidden field, not viewable on the form	
type="image"	Display an inline image which can be clicked to perform an action from a script	
type="password"	Display a text box in which hides text entered by the user	********
type="radio"	Display a radio (option) button	⦿
type="reset"	Display a button which resets the form when clicked	reset
type="submit"	Display a button which submits the form when clicked	submit
type="text"	Display a text box in which displays text entered by the user	LanGear

---

Working with form fields

❑ The general syntax is:

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

o *type* specifies the type of input field

o *name* and *id* attributes identifies the input field for the CGI script

❑ To create a text box, you would enter the tag:

```
<input type="text">
```

❑ If the *type* attribute is not included, the Web browser assumes, *by default*, that you want to create a *text box*.

Creating a text box

❑ To create a text box, use the following HTML code:

```
<input type="text" name="name" id="id"
   value="value" size="value"
   maxlength="value">
```

- o *name* and *id* attributes identify the field
- o *value* attribute assigns a default value to the text box
- o *size* attribute defines the width of the text box in number of characters
- o *maxlength* attribute defines the maximum number of characters allowed in the field

HTML - Forms, Math 279, Fall 2013

13

Controlling the size of a text box

❑ By default, all text boxes are 20 characters wide.
❑ The syntax for changing the size of a text box is:

```
<input size="value">
```

- o *value* is the size of the text box in characters
- o Exmaple

```
<input type="text" name="fname" id="fname"
   size="30">
```

HTML - Forms, Math 279, Fall 2013

14

7

Setting the maximum length for text input

- ❑ Setting the width of a text box does not limit the number of characters the box can hold.
  - o if a user enters text longer than the box's width, the text scrolls to the left
  - o the user cannot see the entire text, but all of it is sent to the CGI script for processing
- ❑ The syntax for setting the maximum length for field input is:

  `<input maxlength="value">`
  - o *value* is the maximum number of characters that can be stored in the field

Setting a default value for a field

- ❑ When the same value is entered into a field, it may make sense to define a default value for a field.
- ❑ Default values can save time and increase accuracy for users of a Web site.
- ❑ To define a default value, use the following syntax:

  `<input value="value">`
  - o *value* is the default text or number that is displayed in the field

Working with form labels

❑ HTML allows a label to be linked with an associated text element – using **for** attribute
❑ The syntax for creating a form label is:

`<label for="id">label text</label>`

- o *id* is the value of the *id attribute* for a field on the form
- o label text is the text of the label
- o A form label has to be bound with the id attribute, NOT the name attribute of a field

Example

```
<label for="fname"> First Name </label>
<input type="text" name="fname" id="fname">
<label for="lname">
   Last Name
   <input type="text" name="lname" id="lname">
</label>
```

Working with radio buttons

- **Radio buttons** display a list of choices from which a user makes a selection.
- Only one radio button can be selected at a time.
- The syntax to create a radio button is:

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

  - *name* identifies the field containing the radio button
  - *id* attribute identifies the specific option. Only required if you intend to use a field label with the radio button
  - *value* attribute indicates the value sent to the CGI script, if that radio button is selected by the user

Working with radio buttons (cont.)

- The **name** attribute must be included, because it groups distinct radio buttons together.
  - selecting one radio button in the group automatically deselects all of the other radio buttons in that group
- Insert descriptive text next to the button.
- Enclose text within a label tag to allow the user to select the radio button or label.

Creating a password field

- A **password field** is a text box in which the characters typed by the user are displayed as **bullets** or **asterisks** i.e. ****.
- The syntax for creating a Password field is:
  ```
  <input type="password">
  ```
- Using a password field is NOT equivalent to having a secure connection.
- The password itself is NOT encrypted.
- The password field only acts as a mask for a field entry as it is entered.

Creating a selection list

- a **selection list** is a list box from which a user selects a particular value or set of values
- good for that a fixed set of possible responses is to be selected
- help prevent spelling mistakes and erroneous entries.
- a selection list is created using **<select>** tag.
- the **<option>** tag is used to specify each of the selection items.

Using a Selection List

Product Registration

First Name [   ] Last Name [   ]
Address #1 [   ]
Address #2 [   ]
City [   ] State [   ] Zip [   ]
Country [United States   ]

Item Purchased [LanPass 115 ▼]   Purchase Date [   ]
| LanPass 115 |
| LanPass 125 | Serial Number [   ]
| LanPass 250 |
| FastSwitch 200 |
| FastSwitch 400 |
| LG 10Mpbs |
| LG 10Mpbs/w |
| LG 100Mpbs |
| LG 100Mpbs/w |

Wittlow Way ▪ Farley, SD 85312 ▪ 1 (800) 555-2377

Using **<select>** & **<option>** tags

□ general syntax for **<select>** and **<option>** tags is:

```
<select name="name" id="id">
    <option value="value1"> item1 </option>
    <option value="value2"> item2 </option>
  .
  .
  .
</select>
```

o **name** and **id** attribute identify the selection field
o each **<option>** tag represents an individual item in the selection list
o the text in the selection list is indicated by the text in **item1**, **item2**, and so forth

Modifying the appearance of a selection list

❑ By default, the `<select>` tag displays one option from the selection list, along with a list arrow to view additional selection options

❑ The number of options displayed can be modified with the **size** attribute.

❑ The syntax of the **size** attribute is:

  `<select size="value">`

  o *value* is the number of *items* that a selection list displays in the form

  o by specifying a value greater than 1, the selection list changes from a drop-down list box to a list box

Selection lists with different size values
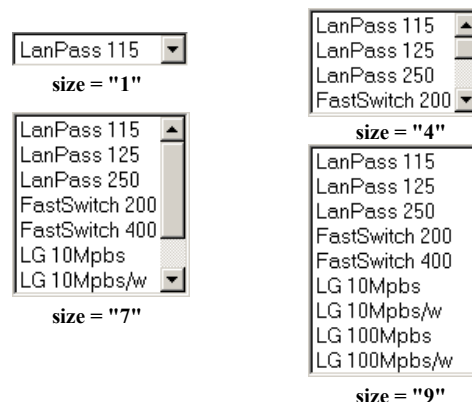
if the *size* value equals to the number of options in the selection list, scroll bar is either not displayed or is dimmed

Making multiple selections

- Adding the **multiple** attribute to the `<select>` tag allows multiple selections from a list.
- The syntax for this attribute is:

    `<select multiple> … </select>`
- A common method to make multiple selections from a selection list is to hold down a specific key while making selections.
    - for **noncontiguous selections**, press and hold the Ctrl key while you make your selections
    - for a **contiguous selection**, select the first item, press and hold the Shift key, and then select the last item in the range

Setting default option

- Specify which of the options should be selected by default, using `selected` attribute
- Syntax for this attribute is:

    `<option selected="selected" value="value">`

    `text`

    `</option>`

Working with option groups

- ❑ the most recent releases of HTML allows you to organize selection lists into distinct groups called **option groups**
- ❑ syntax:

  `<optgroup label="label">`

  - o *label* is the label assigned to the option group, which will appear in the form
  - o the text for the label appears in the selection list above each group of items but is not a selectable item from the list

Option Groups

```
<select name="item" id="item">
   <optgroup label="Routers">
      <option>LanPass 115
      <option>LanPass 125
      <option>LanPass 250
   </optgroup>
   <optgroup label="Switches">
      <option>FastSwitch 200
      <option>FastSwitch 400
   </optgroup>
   <optgroup label="Adapters">
      <option>LG 10Mpbs
      <option>LG 10Mpbs/w
      <option>LG 100Mpbs
      <option>LG 100Mpbs/w
   </optgroup>
</select>
```

a single option group

option group label

a single option group

option group label

LanPass 115

Routers
LanPass 115
LanPass 125
LanPass 250
Switches
FastSwitch 200
FastSwitch 400
Adapters
LG 10Mpbs
LG 10Mpbs/w
LG 100Mpbs

Selection lists vs. radio buttons

❑ Use selection list, if
  o you have a long list of options
  o you want to allow users to select more than one option (use a selection list with the multiple attribute)
❑ Use radio buttons, if
  o you have a short list of options, and only one option is allowed at a time

---

Working with check boxes

❑ A **check box** is either selected or not, there is only one check box per field.
❑ Check boxes are created using the following syntax:
```
<input type="checkbox" name="name" id="id"
   value="value">
```
  o *name* and *id* attribute identify the check box
  o the *value* attribute specifies the value that is sent to the CGI script when the check box is selected
❑ To check a text box by default
```
<input type="checkbox" checked="checked">
```

16

Creating a text area

❑ to create a **text area** for a text box, use tag **<textarea>**

<textarea name="*name*" id="*id*" rows="*value*" cols="*value*> **default text </textarea>**

o **rows** and **cols** attributes define the dimensions of the text box

o **rows** attribute indicates the number of lines in the text box

❑ default text can be specified in the text box when the form is initially displayed.

Creating a text area (cont.)

```
<label for="comments">Comments</label><br>
<textarea name="comments" id="comments" rows="5"
cols="50">
     Enter comments here.
</textarea>
```

default text

dimensions of text area

Comments

Enter comments here.

resulting text area

Using **wrap** attribute (cont.)

❑ wrap user's comments to the next line in the text area

❑ value of **wrap** attribute is **soft, hard** or **off**
- o **soft** value automatically wraps the text to the next line when it exceeds the width of the box
- o **hard** value also wraps text automatically, in the meantime, preserves any line wrapping that takes place in the text box when sending it to CGI
- o **off** value sets all the text to be displayed in a single line

❑ default value for wrap attribute is **soft**

Creating form buttons

❑ Form buttons can be clicked to
- o run programs
- o submit forms
- o reset the form to its original state

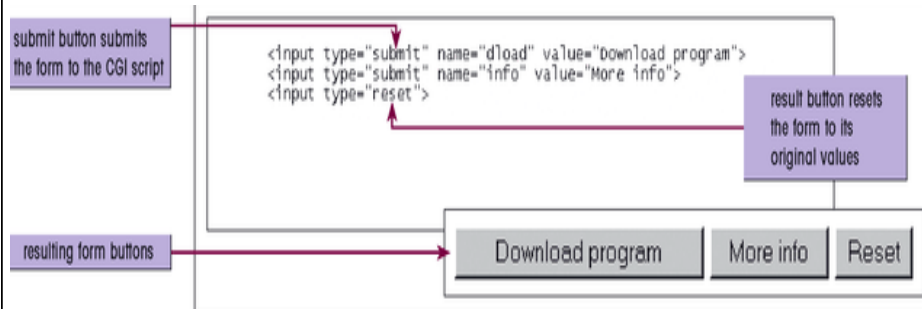# Creating push buttons

- syntax for creating a **push button**

  `<input type="button" value="text">`

  - o **text** is the text that appears on the button
- push buttons perform no actions by themselves
- need a script to associate an action with a push button

Creating submit/reset buttons

- a **submit button** is a button that submits the form to a *CGI* script for processing
- a **reset button** erases users' input and resets form to its original (default) values
- the syntax for creating these two buttons is:

  `<input type="submit" value="text">`

  `<input type="reset" value="text">`

  - o **value** attribute defines the text that appears on the button

Using **name** and **value** attributes

❑ use **name** and **value** attributes when a form contains multiple buttons

submit button submits the form to the CGI script	`<input type="submit" name="dload" value="Download program">`
	`<input type="submit" name="info" value="More info">`
	`<input type="reset">`

	result button resets the form to its original values
resulting form buttons	Download program More info Reset

Creating buttons using the <button> tag

❑ **<input>** tag does not allow the Web page designer to control the appearance of a button

❑ **<button>** tag enables artistic control over the appearance of form buttons

❑ The syntax of the **<button>** tag is:

<button name="name" value="value" type="option">
```
button text and HTML tags
```
</button>

o **name** attribute specifies the name of the button

o **value** attribute sends to a CGI script

o **type** attribute specifies the button type (*submit, reset*, or *button*)

Example

The figure shows how to create a button that contains formatted text and an inline image.

The default value for the type attribute is "button". Within the `<button>` tags you can place whatever HTML tags you wish to format the button's appearance. This includes inline images.

the button type is a simple push button

```
<button name="back" type="button">
    <img src="back.gif" align="absmiddle">
    Go to the
    <b style="color:blue">Home Page</b>
</button>
```

contents of the button

Go to the **Home Page**

button image

Creating file buttons

- a **file button** is used to select files
- contents of the file are not displayed--only the file's location is
- syntax for creating a file button

  `<input type="file" name="file_name">`

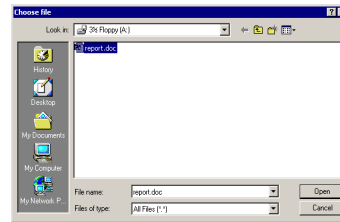 here, type attribute's value is **"file"**

Using a file button, example

`<input type="file" name="file_name">`

1. User clicks the Browse button

2. Selects a file from the Choose File dialog box

3. The filename and location are automatically placed in the text box

A:\report.doc Browse...

HTML - Forms, Math 279, Fall 2013

43

Working with form attributes

❑ attributes to the **`<form>`** tag
- o specify where to send the form data
- o specify how to send

❑ syntax

`<form action="URL" method="option"`
`enctype="text">`
- o **action** specifies the filename and location of the CGI script that process the form
- o **method** specifies how your Web browser sends data to the CGI script
- o **enctype** specifies the format of the data stored in the form's field

HTML - Forms, Math 279, Fall 2013

44

22

The **method** attribute

- ❑ two possible values for the method attribute
    - o **get** or **post**
- ❑ **get** method (default value for method attribute)
    - o packages the form data by appending it to the end of the URL specified in the action attribute
    - o some Web servers limit the amount of data sent via *get* method and will cut off valuable information
- ❑ **post** method
    - o sends form data in a separate data stream, allowing the Web server to receive the data through what is called "standard input"
    - o is preferred for sending data to a Web server
    - o Web servers do not limit the amount of data sent via *post* method

---

The **enctype** attribute

- ❑ defines the technique used to encrypt form data
- ❑ default **enctype** value is "**application/x-www-form-urlencoded**."
- ❑ another **enctype** value that is often used is "**multipart/form-data**,"
- ❑ The most basic way of encoding data is to use "**text/plain**," which encodes the data as simple text.

Specifying where and how to send form data

This following example shows the CGI script that processes the form is located at the URL http://www.langear.com/cgi/mailer (a fictional address) and uses the "post" method.

```
<html>
<head>
<title>LanGear Registration Form</title>
</head>
<body text="#850000">
<form name="reg" action="http://www.langear.com/cgi/mailer"
method="post">
```

Form values

The Web browser presents a page, an example of which is shown in this figure, displaying the name of each field in the form and the value assigned to it. At the same time, the CGI script formats a mail message to be sent to the address you entered.

LanGear Test Registration Form Values

Below is what you submitted to adavis@langear.com on Thursday, March 11, 2004 at 15:33:57

address1: Room 634

address2: 211 Hawkins Avenue

city: Lawrence

comments: How do I access the internal settings of the router?

country: United States

date: 2/14/2004

fname: Andrew

item: LanPass 250

lname: Davis

nw: yes

snumber: LG100-78711

state: WI

use: edu

win: yes

zip: 53701

Using **mailto** action

- **mailto** action sends form information via e-mail without using a CGI script
- it mails form information to a specified e-mail address, bypassing the need for using CGI scripts on a Web server.
- the syntax of *mailto* action is:

  ```
  <form action="mailto:e-mail_address"
    method="post" enctype="text/plain">
  ```

  - *e-mail_address* is the e-mail address of the recipient of the form

Working with hidden fields

- A **hidden field** is added to the form but not displayed in the Web page
- Because the field is hidden, it can be placed anywhere between the opening and closing `<form>` tags
- The syntax for creating a hidden field is:

  ```
  <input type="hidden" name="name"
    value="value>
  ```

- Place all hidden fields in one location to make it easier to read and interpret the HTML code
- Include a comment describing the purpose of the field

Homework

- ❑ READ Tutorial 6
- ❑ Review assignment
- ❑ Case Problem 1
- ❑ Case Problem 2