Tutorial 6: Working with Web Forms

Computer Science Department
University of Central Florida

COP 3175 – Internet Applications

Objectives

- Explore how Web forms interact with Web servers
- Create form elements
- Create field sets and legends
- Create input boxes and form labels
- Creation option buttons
- Create selection lists
- Create check boxes

Objectives

- Create text area boxes
- Apply styles to Web forms
- Work with form buttons
- Explore image elements and hidden fields
- Work with form actions and methods

Tutorial 6 Website

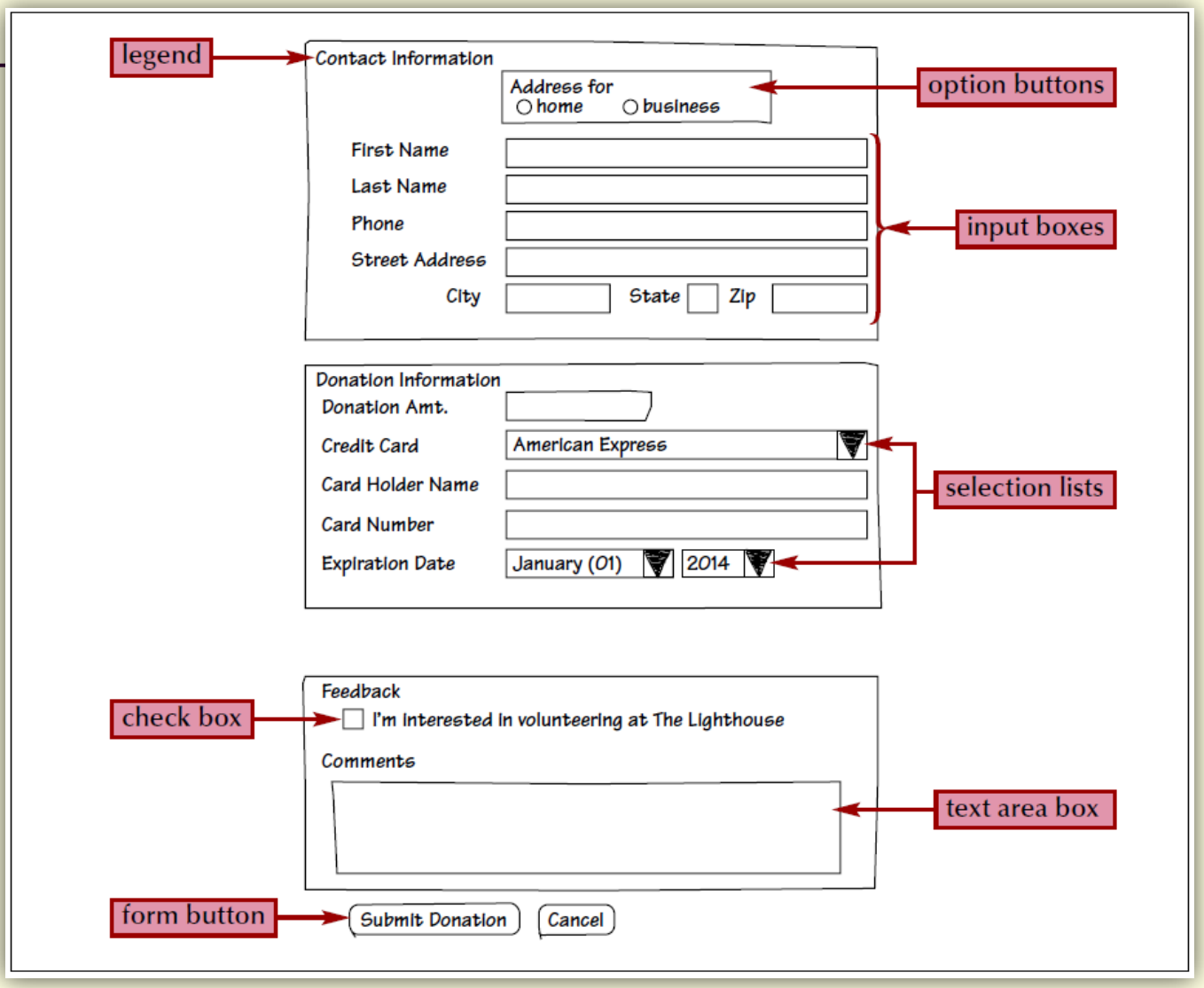
- The Lighthouse
  - A community service center in St. Peters, MI
    - ran by Terry Ives (the director)
  - Provides social services to the community
  - We (CGS 3175) are volunteers at The Lighthouse
    - Helping to upgrade the website!
  - Our current focus is to make a page to receive online donations
  - This means we need to make a Web form!

Introducing Web Forms

- What is a web form?
  - Web forms collect information from Web site visitors
  - These forms will have data fields that the user will fill out
    - Just like filling out a paper form
  - A web form is the “submitted”
    - Which means the data is transmitted to the ISP’s Web server for processing
      - (more details coming)

Tutorial 6 Website

- Terry's proposed donations form



Introducing Web Forms

- Parts of a Web Form:
  - Each piece of information for a form is stored in a **field**
  - And the value itself is known as the **field value**
  - Different types of fields:
    - Some fields are meant for entering data
    - Others are meant for selecting from a set of values
  - Users enter or select a field value using **control elements**
    - such as buttons, boxes, lists, etc.

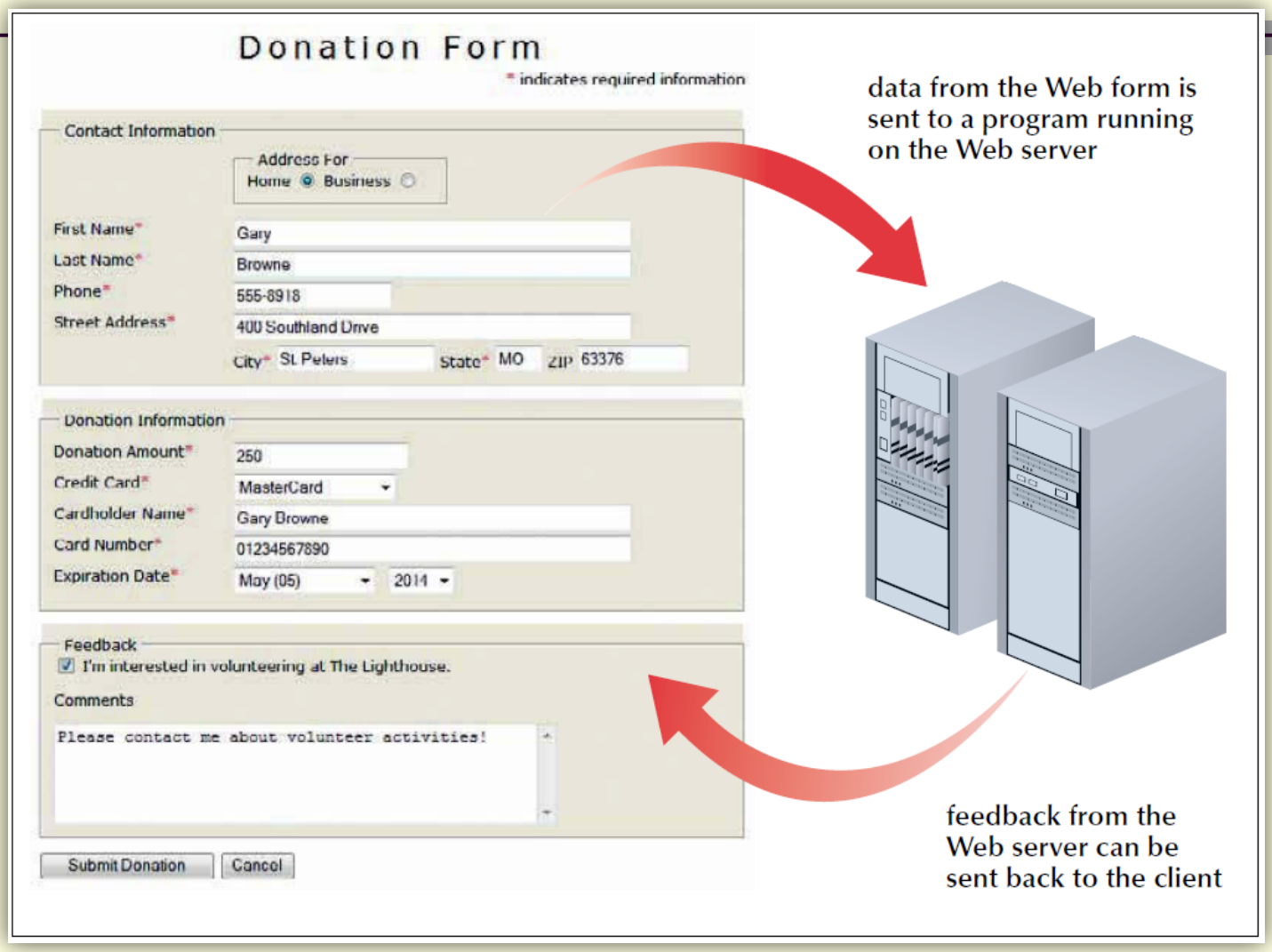
Introducing Web Forms

- Parts of a Web Form:
  - HTML supports the following control elements:
    - **Input boxes** for text and numerical entries
    - **Option buttons** (aka radio buttons) for selecting a single option from a predefined list
    - **Selection lists** for long lists of options
      - usually appearing in **a drop-down list box**
    - **Check boxes** for specifying “yes” or “no”
    - **Text areas** for extended entries that can include several lines of text

Forms and Server-Based Programs

- Processing Web Forms:
  - Web forms are used to collect information
  - But we need to know how forms are processed!
  - The actual data is stored and analyzed by a program that runs on a Web server
  - Examples of server-based programs:
    - Online databases containing customer information
    - Online catalogs for ordering and purchasing products
    - Dynamic Web sites (data driven websites)
    - Message boards for hosting online discussion forums

Forms and Server-Based Programs



Forms and Server-Based Programs

- Processing Web Forms:
  - Access to these form-processing programs:
    - Since these programs run on Web servers, you will most likely not have permission to edit them
    - Rather, you will be given information on how to interact with the program
      - how to send the form data to the program for processing
      - This will usually include a list of fields that are required by the program and a description of those fields
    - Your Web form will need to work in conjunction with the requirements of the server-based processing program

Forms and Server-Based Programs

- Server-based processing programs:
  - These are written in many languages
  - The earliest and most common of these languages are called **Common Gateway Interface (CGI) scripts**
    - which are written in a language called **Perl**.
  - Other popular languages include:
    - ASP
    - ColdFusion
    - C/C++
    - PHP
    - VBScript

Forms and Server-Based Programs

- Server-based processing programs:
  - Which program (and in which language) is your Web server using?
  - You'll have to check with your hosting company to find out!
  - Back to the Tutorial 6 website: The Lighthouse
    - The hosting company has scripts in place to receive the data from the form and process it
    - To be expected, we can't access these scripts
    - But we can make the form properly, so the data, from the form, can be sent over to the scripts on the server

Tutorial 6 Website

Creating a Web Form

- Forms are created using the form element,
  - Syntax:

```
<form attributes>
    elements
</form>
```

    - where *attributes* are the attributes that control how the form is processed and *elements* are elements places within the form
  - Forms contain the typical control elements already mentioned
    - But they can also contain other page elements, such as tables, paragraphs, inline images, headings, etc.

Creating a Web Form

- Form attributes usually tell the browser several things:
  - the location of the server-based program to be applied to the form's data
  - how the data is to be transferred to the script
  - and other important information
- For now, these things are not needed for the purposes of designing the form
  - actually useful to omit/ignore them at first
  - Once the form is complete, we can then add the attributes needed for the server program
    - We'll do that at the end of this tutorial

Creating a Web Form

- Forms are identified with two attributes:
  - the id attribute and the name attribute
  - Why name a form?
    - Useful when a page has multiple forms, allowing you to differentiate one from another
    - Also the name is sometimes required by sever programs
  - Syntax:
    - **`<form name="name" id="id">…</form>`**
      - where name is the name of the form, and id is the id of the form
    - The name attribute is the older standard of identification
    - The id attribute is the current standard
      - For maximum compatibility, use both and set to **same value**

Tutorial 6 Website

- Adding the basic form element:
  - So in the id, “rightcolumn”, we begin adding our form information

```
<div id="rightColumn">
    <h1>Donation Form</h1>
    <p><span>*</span> indicates required information</p>

    <form name="donationForm" id="donationForm">
    </form>

</div>
```

Creating a Field Set

- Web forms can have many fields
  - HTML and XHTML allow you to organize fields into groups, called called field sets
    - Browsers usually render field sets by placing a box around fields in the group
  - Syntax:

```
<fieldset id="id">
   controls
</fieldset>
```

    - where id identifies the field set and controls are the control elements associated with fields within the field set

Tutorial 6 Website

- Partitioning the form into three field sets
  - Terry wants three sections of the form:
    - contact, donation, and feedback
    - So we make a field set for each

```
<form name="donationForm" id="donationForm">
    <fieldset id="contact">
    </fieldset>

    <fieldset id="donation">
    </fieldset>

    <fieldset id="feedback">
    </fieldset>
</form>
```

Creating a Field Set

- To add a caption to a field set, add the following tag after the opening <fieldset> tag:

  <legend>*text*</legend>

  - where *text* is the text of the field set caption

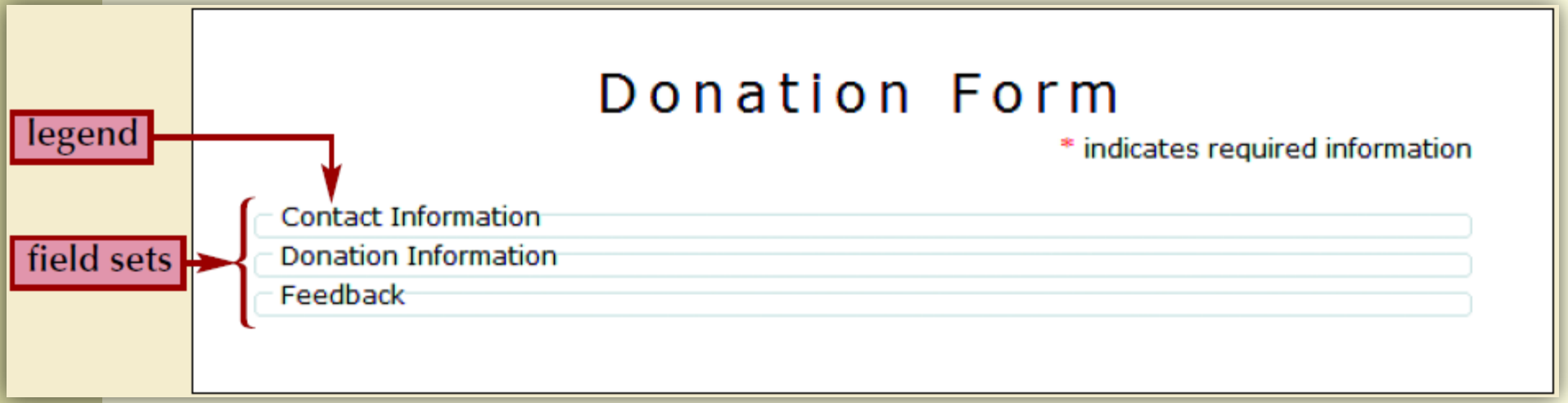
- Note: legend elements can only contain text and not other page elements

Tutorial 6 Website

- Adding legends for the three field sets:
  - Based off of the sketch Terry gave us, we make legends for the three field sets

```
<form name="donationForm" id="donationForm">
   <fieldset id="contact">
      <legend>Contact Information</legend>
   </fieldset>

   <fieldset id="donation">
      <legend>Donation Information</legend>
   </fieldset>

   <fieldset id="feedback">
      <legend>Feedback</legend>
   </fieldset>
</form>
```

Tutorial 6 Website – Result



- Field sets are block level elements
  - The boxes around them are small and narrow
    - cause we haven't added other fields in them yet
  - Legends appear in the upper-left corner
    - can use CSS to move and style the legend

Creating Input Boxes

- Most of the control elements in which users type or select a data value are marked as input elements
  - Syntax:

    <input type="*type*" name="*name*" id="*id*" />

    - where ***type*** specifies the type of input control, and the name and id attributes provide the control's name and id
    - You should give BOTH the name and id attributes, and they should both have the SAME value
      - ensures compatibility with older browsers
  - There are 10 different input types shown in the following graphic…

Creating Input Boxes

Type Value	Description	General Appearance
button	Displays a button that can be clicked to perform an action from a script	Run Program
checkbox	Displays a check box	
file	Displays a Browse button to locate and select a file	donations.htm Browse...
hidden	Creates a hidden field, not viewable on the form	
image	Displays an inline image that can be clicked to perform an action from a script	
password	Displays an input box that hides text entered by the user	•••••••••
radio	Displays an option button	
reset	Displays a button that resets the form when clicked	Cancel Donation
submit	Displays a button that submits the form when clicked	Submit Donation
text	Displays an input box that displays text entered by the user	Terry Ives

Tutorial 6 Website

- Adding the first couple control elements:
  - Terry wants input boxes for the first and last name
  - Syntax:

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

  - Note:
    - We have to ask Terry for the expected id names of the first and last name fields
      - Meaning, the ids required by the ISP server-based program
    - She tells us the ids are firstName and lastName
    - So we make the two text boxes and we put the words "First Name" and "Last Name" in front of them

Tutorial 6 Website – Result

```
<form name="donationForm" id="donationForm">
   <fieldset id="contact">
      <legend>Contact Information</legend>
      First Name
      <input type="text" id="firstName" name="firstName" />
      Last Name
      <input type="text" id="lastName" name="lastName" />
</fieldset>
```

text type indicates an input text box

Contact Information

First Name Terry Last Name Ives

Donation Information

Feedback

- Notice that the input boxes are on the same line
  - HTML treats all form control elements as inline elements

Working with Field Labels

- Adding Field Labels
  - Notice that we typed the words “First Name” directly before the firstName field
    - This descriptive text simply indicates to the user the purpose of the field
  - However, nothing in that HTML code explicitly associates that test (“First Name”) with the firstName input box
  - Ideally, when we’d like this descriptive text to be “linked” to the field

Working with Field Labels

- Adding Field Labels
  - We can make field labels, which associate text with a control element
  - Syntax:

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

    - where id is the value of the id attribute for a field's control element, and label text is the text of the label
    - The for attribute associates the text of the label with the control element id

Working with Field Labels

- Adding Field Labels
  - Example:
    - The following code associates the label text “First Name” with the firstName control element

```
<label for=“firstName”>First Name</label>
<input type=“text” id=“firstName” />
```

    - This for attribute explicitly associates the label with the control element

Working with Field Labels

- Adding Field Labels
  - Example:
    - You can also make the association implicitly by nesting the control element within the label

```
<label>
    First Name
    <input type="text" id="firstName" />
</label>
```

    - Note: we do not need the for attribute when we nest the control element within the label element
  - The method you choose, explicit or implicit association, is up to you

Tutorial 6 Website

- Adding field labels:
  - We use the implicit association
    - Which is nice also since it treats the control element and its label as a single object

```
<fieldset id="contact">
   <legend>Contact Information</legend>

   <label>
      First Name
      <input type="text" id="firstName" name="firstName" />
   </label>
   <label>
      Last Name
      <input type="text" id="lastName" name="lastName" />
   </label>

</fieldset>
```

Tutorial 6 Website

- Terry wants the labels in one column and the input boxes in another column
  - We use a CSS sheet (forms.css) for layout
  - We start by simply assigning the labels to a class that we will style

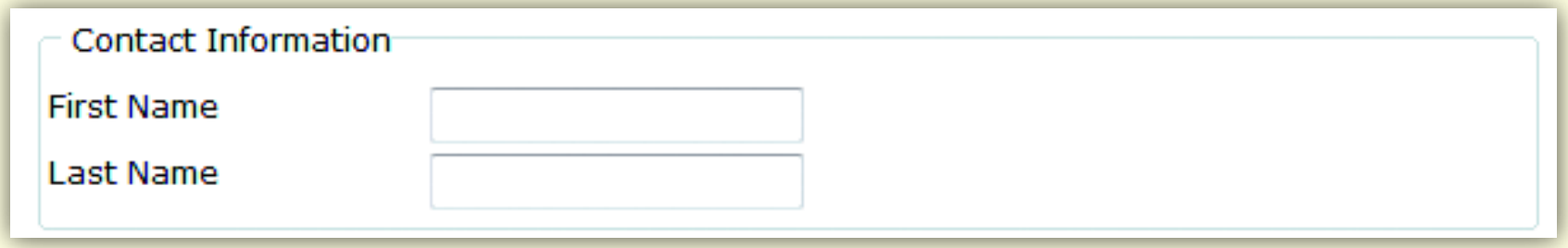
```
<label class="blockLabel">
   First Name
   <input type="text" id="firstName" name="firstName" />
</label>
<label class="blockLabel">
   Last Name
   <input type="text" id="lastName" name="lastName" />>
</label>
```

Tutorial 6 Website

- Terry wants the labels in one column and the input boxes in another column
  - We style this blockLabel class:
    - display property should be set to block
    - Margin is 12px above and below the label and 0 px to the left and right
    - We also use relative positioning
    - For each input element, within the label, we give it an absolute position of 150 px from the left margin of label

```
label.blockLabel       {display: block; position: relative; margin: 12px 0px}
label.blockLabel input {position: absolute; left: 150px}
```

Tutorial 6 Website – Result



Tutorial 6 Website

- Marking some fields as required:
  - Some fields are required by the server program
    - Such as the firstName and lastName fields
  - Terry wants them marked with a red asterik

```
<label class="blockLabel">
   First Name<span>*</span>
   <input type="text" id="firstName" name="firstName" />
</label>

<label class="blockLabel">
   Last Name<span>*</span>
   <input type="text" id="lastName" name="lastName" />
</label>
```

  - We then add the following style to the CSS file:

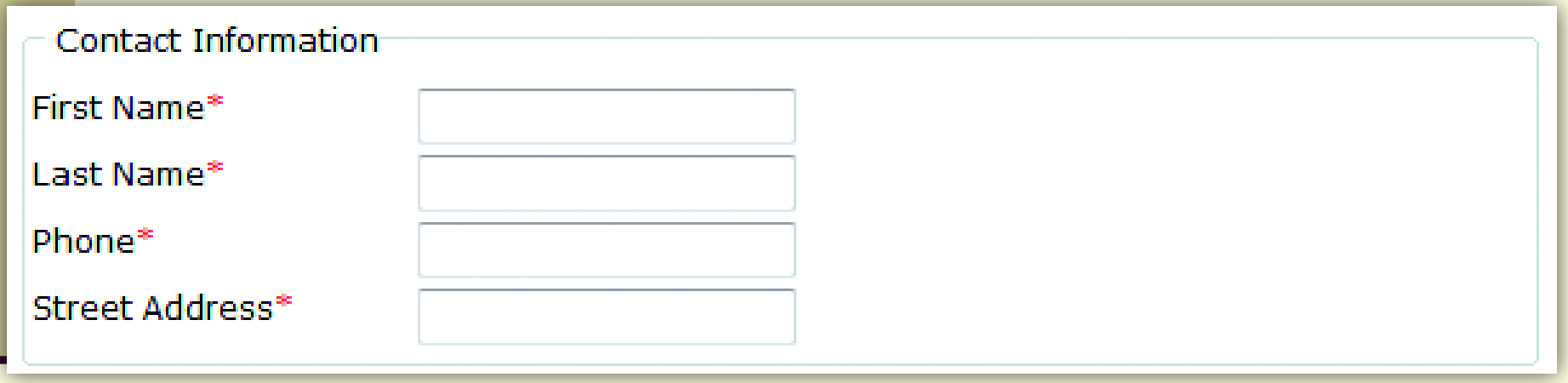
```
#donationForm span {color: red}
```

Tutorial 6 Website

- Adding more fields to the form
  - Terry wants the form and street fields added

```
<fieldset id="contact">
   <legend>Contact Information</legend>

   <label class="blockLabel">
      First Name<span>*</span>
      <input type="text" id="firstName" name="firstName" />
   </label>

   <label class="blockLabel">
      Last Name<span>*</span>
      <input type="text" id="lastName" name="lastName" />
   </label>

   <label class="blockLabel">
      Phone<span>*</span>
      <input type="text" id="phone" name="phone" />
   </label>

   <label class="blockLabel">
      Street Address<span>*</span>
      <input type="text" id="street" name="street" />
   </label>

</fieldset>
```

Tutorial 6 Website – Result

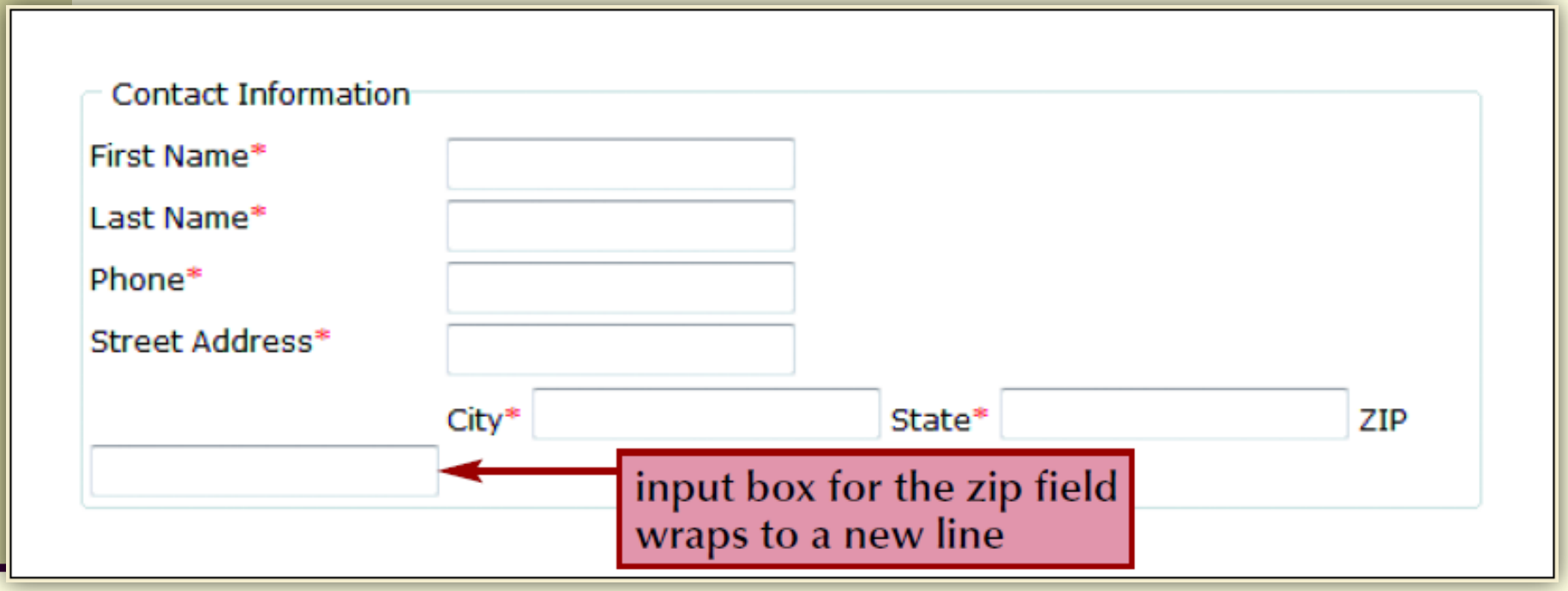


Tutorial 6 Website

- Adding city, state, and zip fields
  - Terry wants them on the same line
    - So we don't use the blockLabel class attribute
    - We add a new style to the CSS file:
    - **`label.indentLabel {margin-left: 150px}`**

```
    <label class="blockLabel">
       Street Address<span>*</span>
       <input type="text" id="street" name="street" />
    </label>

    <label class="indentLabel">
       City<span>*</span>
       <input type="text" id="city" name="city" />
    </label>
    <label>
       State<span>*</span>
       <input type="text" id="state" name="state" />
    </label>
    <label>
       ZIP
       <input type="text" id="zip" name="zip" />
    </label>

</fieldset>
```

Tutorial 6 Website – Result



- Problem: city, state, zip don't fit on one line
  - Browsers give input boxes a default width
    - We can change the width using the CSS width style

Setting the Width of an Input Box

- To change the width of an input box, use the width attribute, which is displayed as follows:

```
#id {width: value}
```

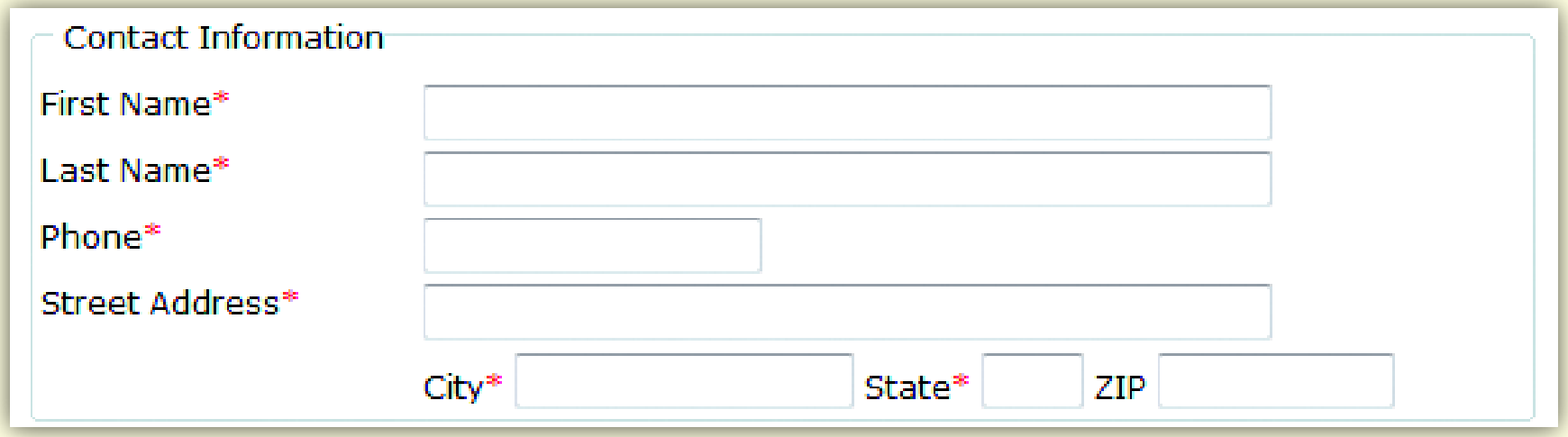
  - where *id* is the id of the control and *value* is the width you want to apply to the input box

Tutorial 6 Website

- Setting the widths of the input boxes
  - State input box uses only a two-letter abbreviation
    - So it can be set to 3em
  - Zip code can be 7em
  - Finally, Terry wants city and phone boxes set to 10em and other boxes set to 25em

```
#donationForm span      {color: red}

label.indentLabel       {margin-left: 150px}

#firstName, #lastName, #street {width: 25em}
#phone, #city                  {width: 10em}
#state                         {width: 3em}
#zip                           {width: 7em}
```

Tutorial 6 Website – Result



- Changing size of boxes in HTML
  - See page 362 for older way of doing this
  - uses a size attribute of the input element

Setting the Maximum Width of an Input Box

- Using the maxlength attribute
  - We just saw how to change the visual width of the input box
    - However, this does NOT limit the number of characters the box can hold
    - If a user types more text than the box's width, the text simply scrolls to the left
  - To actually limit the number of characters, you add the attribute:
    - **`<input type="text" maxlength="chars" />`**
      - where chars is the maximum number of characters that can be stored in the field

Tutorial 6 Website

- Setting the maximum width of certain fields
  - Terry wants us to limit the state field to two chars
  - Zip code should be limited to 10 characters, allowing a nine-digit zip code that could incorporate a hyphen

```
<label>
   State<span>*</span>
   <input type="text" id="state" name="state" maxlength="2" />
</label>
<label>
   ZIP
   <input type="text" id="zip" name="zip" maxlength="10" />
</label>
```

values of the state field are limited to two characters

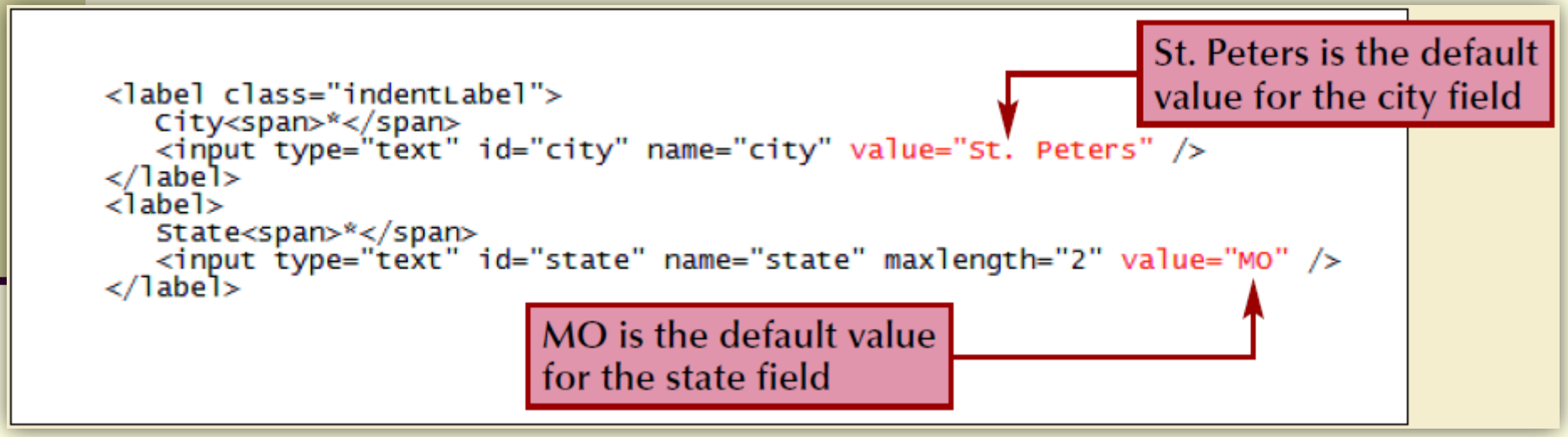
values of the zip field are limited to 10 characters

Setting a Default Value for a Field

- Setting default values:
  - Some fields will often receive the same input
    - Such as a state field, if most of your orders come from a certain state
  - It makes sense to give this field a default value
    - Easier for customers
    - Increases the accuracy of the data entered into the web form
  - Syntax:
    - **`<input value="value" />`**
      - where value is the default text or number that is displayed in the field

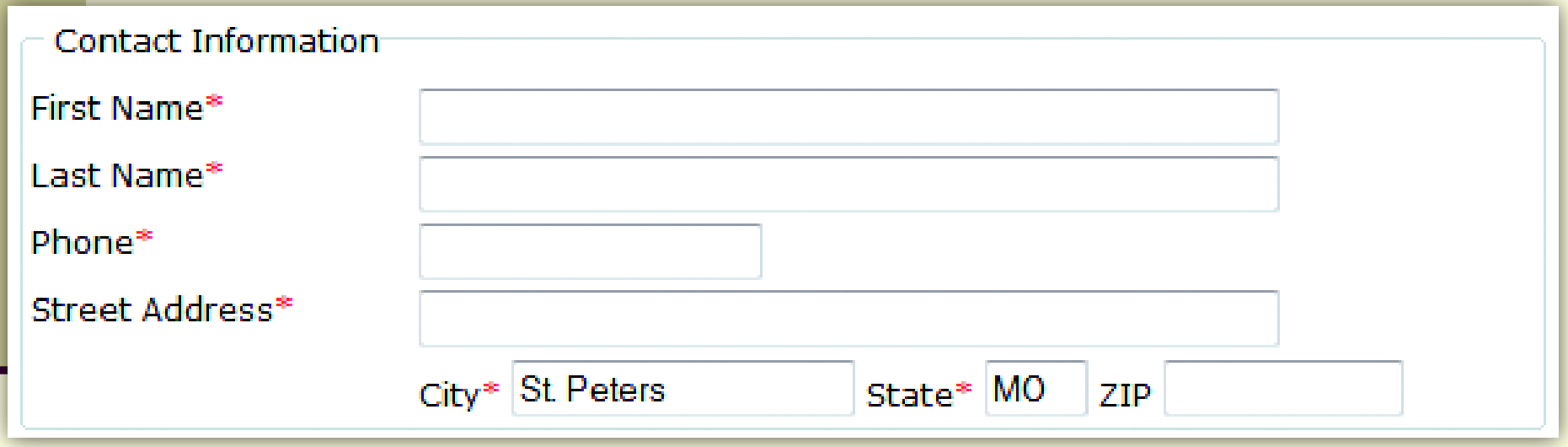
Tutorial 6 Website

- Setting default values
  - Terry says most donations come from St. Peters, Missouri and wants that set as the default




```
<label class="indentLabel">
   City<span>*</span>
   <input type="text" id="city" name="city" value="St. Peters" />
</label>
<label>
   State<span>*</span>
   <input type="text" id="state" name="state" maxlength="2" value="MO" />
</label>
```

Tutorial 6 Website – Result



Brief Interlude: FAIL Pics

Creating Option Buttons

- Other Parts of a Web Form:
  - Thus far, we've only looked at text boxes
  - Another common control element is known as an option button (aka radio button)
    - These buttons allow users to select a data value from a limited set of possible values
  - Note: users can only select one button at a time from a group of buttons

Creating Option Buttons

- Option (Radio) Buttons:
  - Syntax to create a group of buttons

```
<input type="radio" name="name" id="id1" value="value1" />
<input type="radio" name="name" id="id2" value="value2" />
<input type="radio" name="name" id="id3" value="value3" />
...
```

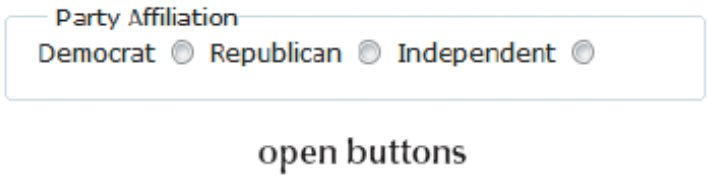
    - Notice that all options within the group have the same **name** value
      - This places those radio buttons in the same group
    - The **id** is really only required if you plan to use field labels
      - Or if you need some way of distinguishing one button from another
    - **value1**, **value2**, **value3**, etc. are field values associated with each option

Creating Option Buttons

- Option (Radio) Buttons:
  - Example Web form:

```
<fieldset>
   <legend>Party Affiliation</legend>

   <label for="demOption">Democrat</label>
   <input type="radio" name="party" id="demOption" value="dem" />

   <label for="gopOption">Republican</label>
   <input type="radio" name="party" id="gopOption" value="gop" />

   <label for="indOption">Independent</label>
   <input type="radio" name="party" id="indOption" value="ind" />

</fieldset>
```

HTML code



open buttons

Creating Option Buttons

- Option (Radio) Buttons:
  - Notice in last example:
    - None of the buttons were initially selected
    - By default, option buttons are unselected
    - You can set an option button to be selected by adding the checked attribute to the input element:
    - Syntax:

      `<input type="radio" checked="checked" />`

      - or in older Web pages, you may see:

      `<input type="radio" checked />`

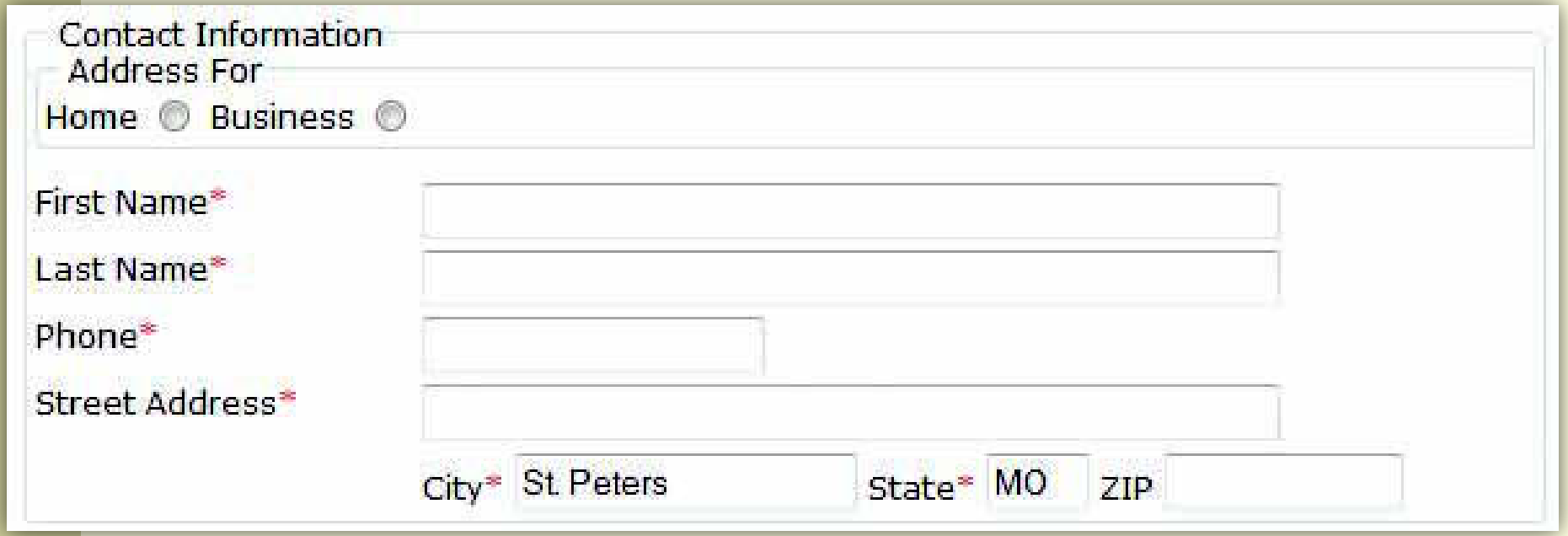
      - Note: this format is NOT supported in the official specifications of HTML and XHTML

Tutorial 6 Website

- Adding Option/Radio Buttons
  - Terry wants option buttons to distinguish donations from Home users or Business users

```
<form name="donationForm" id="donationForm">

   <fieldset id="contact">
      <legend>Contact Information</legend>

      <fieldset id="addressOptions">
         <legend>Address For</legend>

         <label for="homeType">Home</label>
         <input type="radio" id="homeType" name="addressType" value="home" />
         <label for="busType">Business</label>
         <input type="radio" id="busType" name="addressType" value="business" />

      </fieldset>

      <label class="blockLabel">
         First Name<span>*</span>
         <input type="text" id="firstName" name="firstName" />
      </label>
```

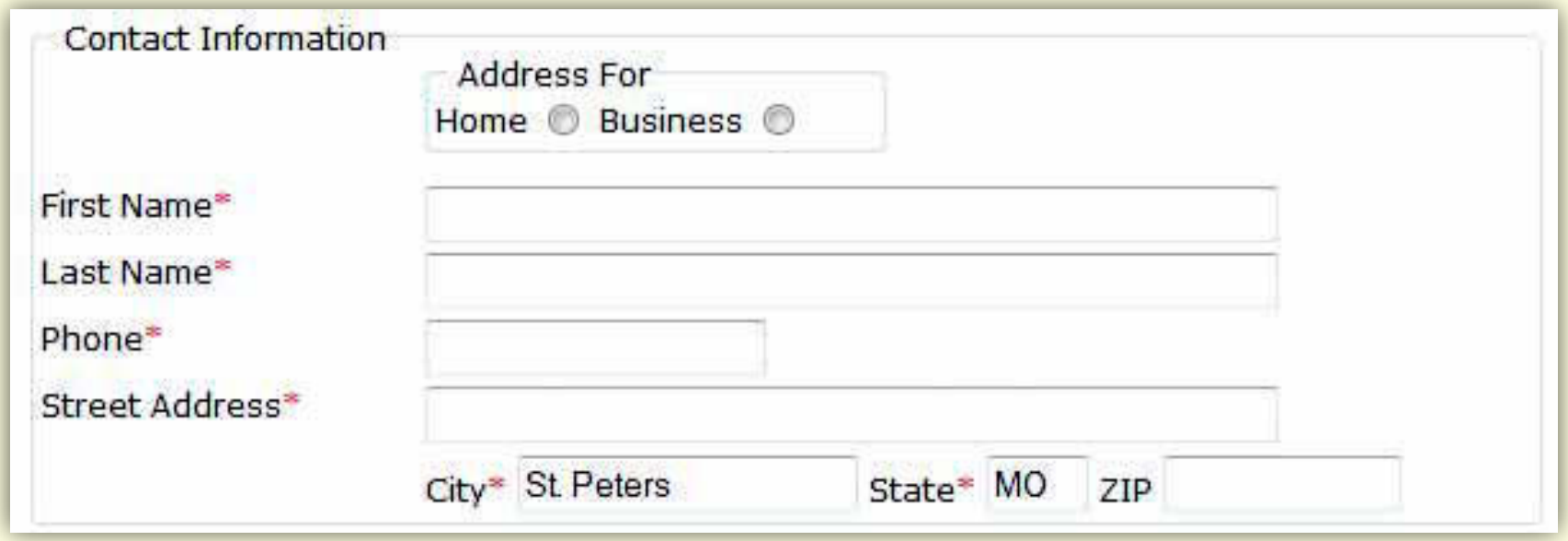
Tutorial 6 Website – Result



- Option button group looks too wide
  - Terry wants the option group to be lined up with the other elements in the form

Tutorial 6 Website – Result

```
#firstName, #lastName, #street {width: 25em}
#phone, #city                  {width: 10em}
#state                         {width: 3em}
#zip                           {width: 7em}

#addressOptions                {width: 180px; margin-left: 150px}
```

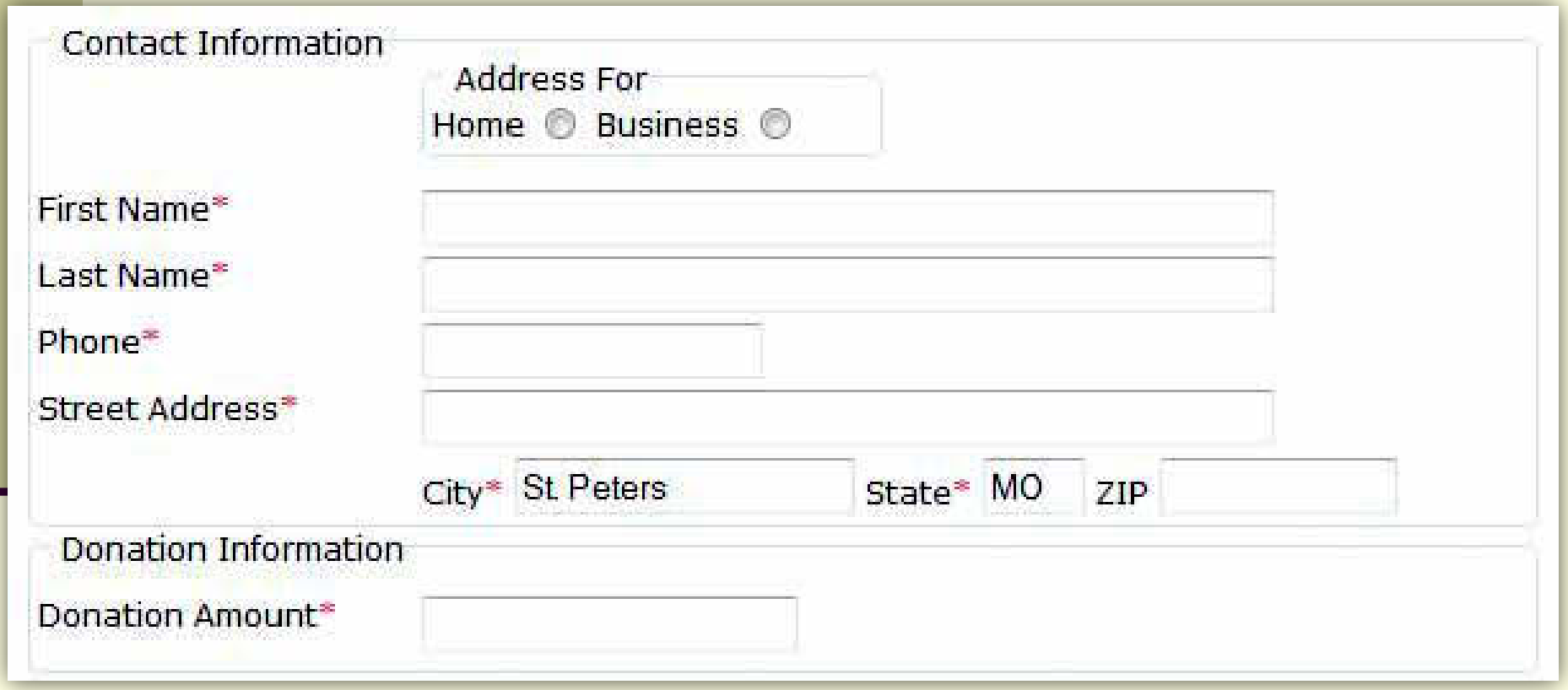


Tutorial 6 Website

- Adding control elements for Donation portion of the form
  - will store the amount of the donation
  - and the method of payment

```
<fieldset id="donation">
   <legend>Donation Information</legend>

   <label class="blockLabel">
      Donation Amount<span>*</span>
      <input type="text" id="amount" name="amount" />
   </label>
</fieldset>
```

Tutorial 6 Website – Result



Creating a Selection List

- Another common control element is known as the **selection list**
  - A selection list is a list box from which a user selects a particular field value **or** set of field values
  - Fulfills the same role as a group of option buttons
    - But used when there are too many options to be shown in the form by way of option buttons
  - So when are they useful?
    - when there is a large, yet fixed set of possible responses from the user

Creating a Selection List

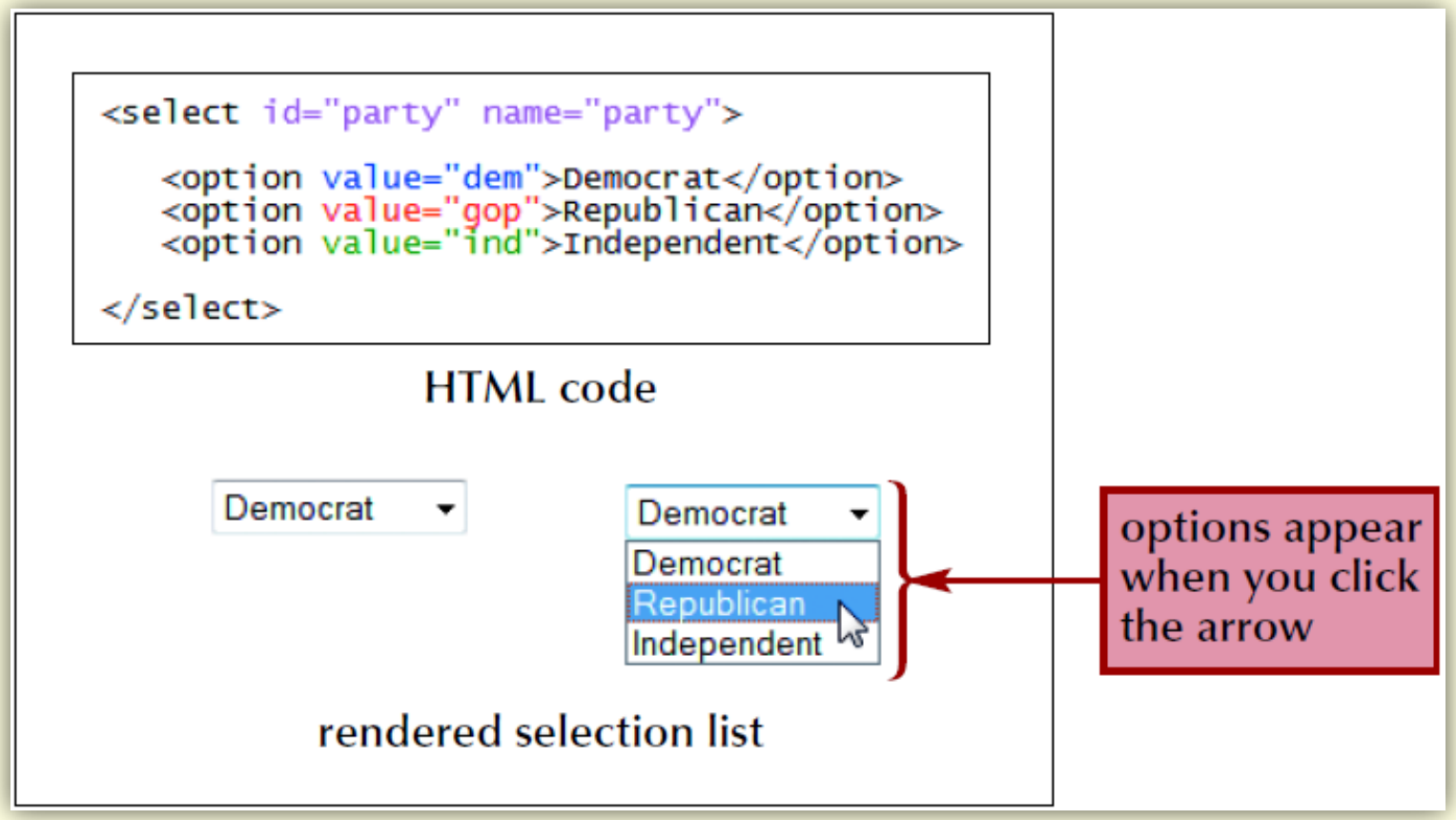
- Another common control element is known as the **selection list**
  - You can create a selection list using the <select> element
  - and you can specify each individual selection item using the <option> element

```
<select name="name" id="id">
     <option value="value1">text1</option>
     <option value="value2">text2</option>
     ...
</select>
```

    - where name and id identify the field associated with the selection list

Creating a Selection List

- Example:



Creating a Selection List

- Grouping Selection Options
  - For longer lists, it may be helpful to group list options together
  - To do this, place list options in option groups:

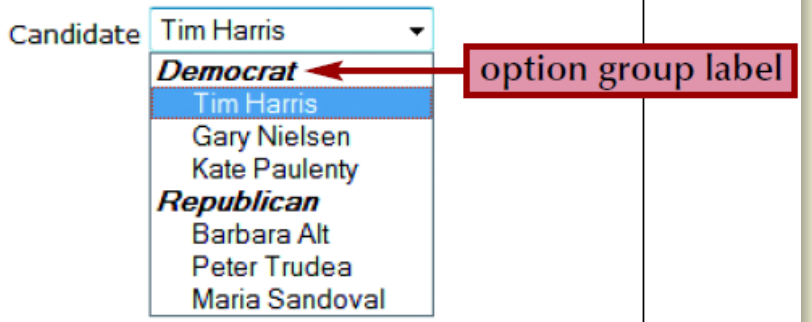
```
<select>
   <optgroup label="label1">
      <option>text1</option>
      <option>text2</option>
      ...
   </optgroup>
   <optgroup label="label2">
      <option>text1</option>
      <option>text2</option>
      ...
   </optgroup>
</select>
```

Creating a Selection List

- Example of Option Groups:

```
<label for="candidate">Candidate</label>
<select id="candidate" name="candidate">

   <optgroup label="Democrat">
      <option>Tim Harris</option>
      <option>Gary Nielsen</option>
      <option>Kate Paulenty</option>
   </optgroup>

   <optgroup label="Republican">
      <option>Barbara Alt</option>
      <option>Peter Trudea</option>
      <option>Maria Sandoval</option>
   </optgroup>

</select>
```

HTML code



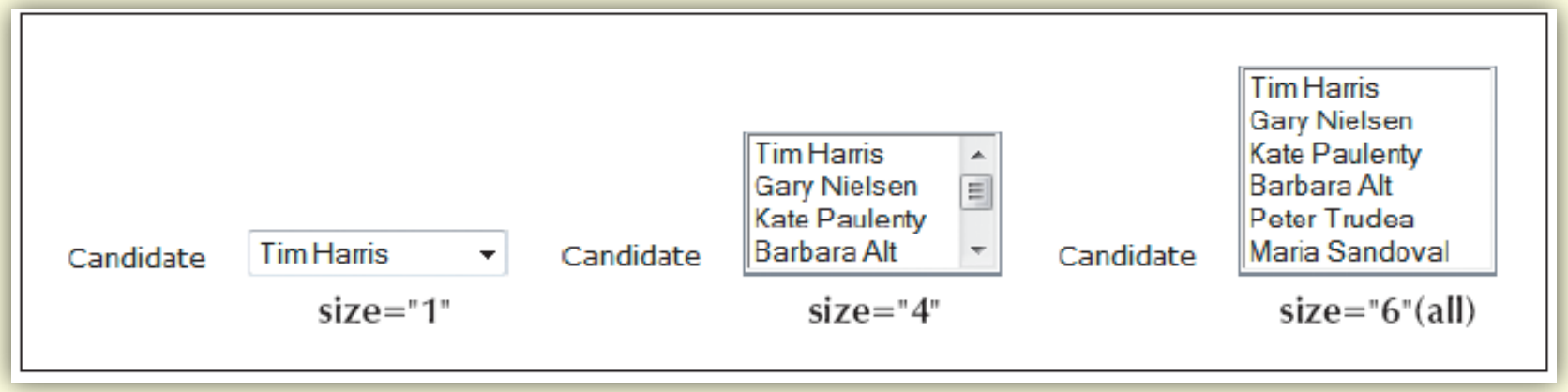
selection list option group

Setting the Selection List Size

- You can change the number of options displayed in the selection list by modifying the size attribute
  - Syntax:
    - **`<select size= “value”>...</select>`**
    - where *value* is the number of items that the selection list displays in the form
  - When value is set to greater than 1, the selection list changes from a drop-down list box to a list box with a scroll bar
    - If the size is set to be equal to the number of options in the list, the scroll bar is dimmed or not displayed at all

Setting the Selection List Size

- Example of setting the size of a selection list:



Tutorial 6 Website

- Adding selection list (for credit card options)
  - Terry wants a list of credit cards that she accepts

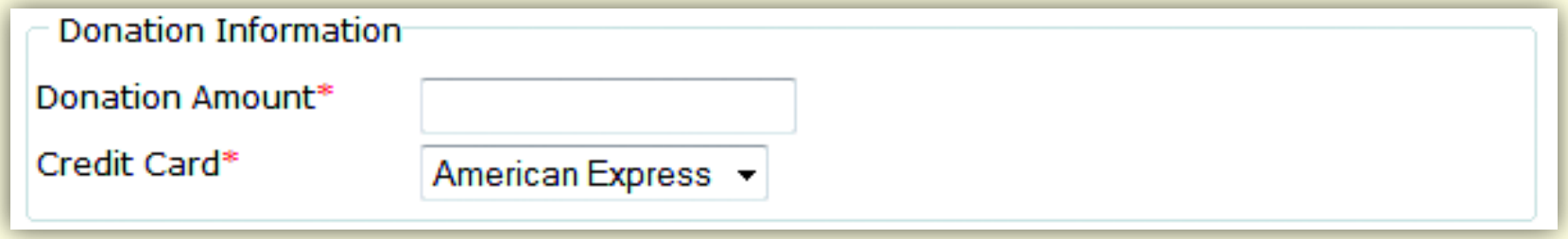
```
<fieldset id="donation">
   <legend>Donation Information</legend>

   <label class="blockLabel">
      Donation Amount<span>*</span>
      <input type="text" id="amount" name="amount" />
   </label>

   <label class="blockLabel">
      Credit Card<span>*</span>
      <select id="creditCard" name="creditCard">
         <option value="Amex">American Express</option>
         <option value="Disc">Discover</option>
         <option value="MC">MasterCard</option>
         <option value="Visa">Visa</option>
      </select>
   </label>

</fieldset>
```

Tutorial 6 Website – Result

- Formatting the selection list:

```
#firstName, #lastName, #street {width: 25em}
#phone, #city                  {width: 10em}
#state                         {width: 3em}
#zip                           {width: 7em}

#addressOptions          {width: 180px; margin-left: 150px}

#creditCard              {position:absolute; left: 150px}
```

- Result:

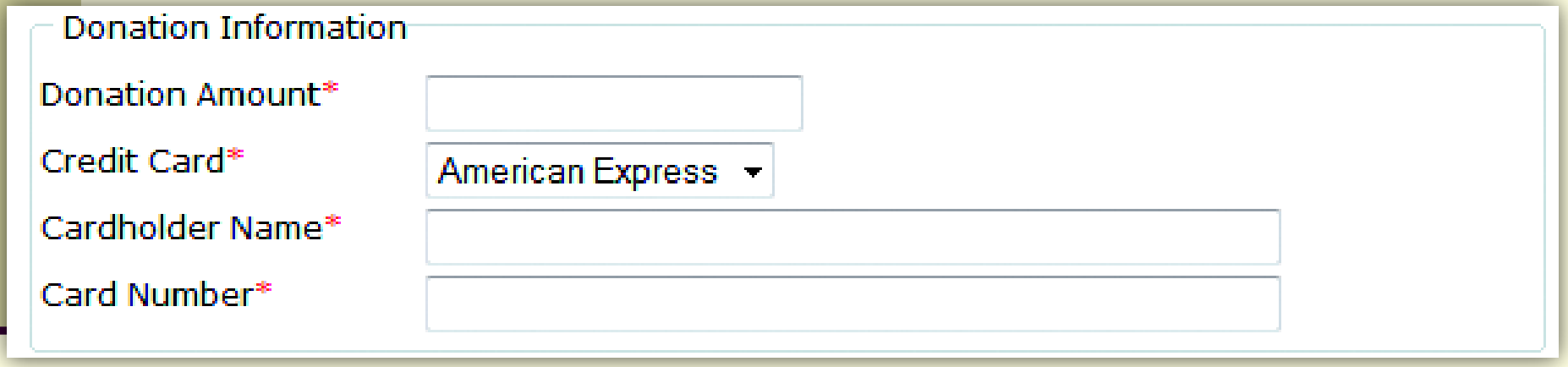


Tutorial 6 Website

- Adding boxes for card holder and card number:

```
<label class="blockLabel">
   Credit Card<span>*</span>
   <select id="creditCard" name="creditCard">
      <option value="Amex">American Express</option>
      <option value="Disc">Discover</option>
      <option value="MC">MasterCard</option>
      <option value="Visa">Visa</option>
   </select>
</label>

<label class="blockLabel">
   Cardholder Name<span>*</span>
   <input type="text" id="cardHolder" name="cardHolder" />
</label>

<label class="blockLabel">
   Card Number<span>*</span>
   <input type="text" id="cardNumber" name="cardNumber" />
</label>

</fieldset>
```

Tutorial 6 Website – Result



Tutorial 6 Website

- Adding selection list (for expiration date)

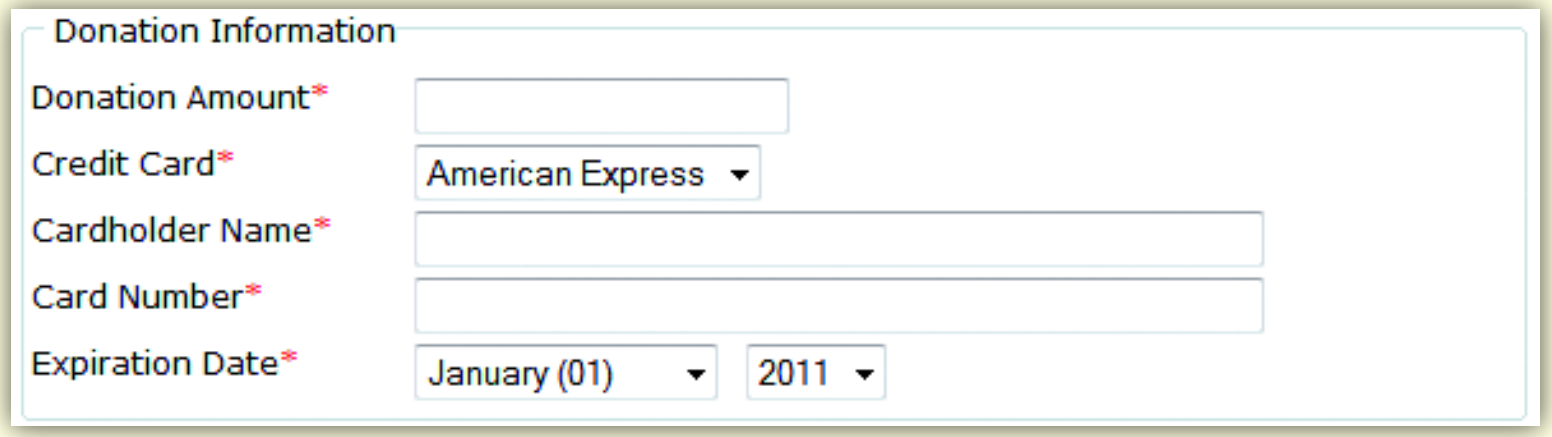
```
<label class="blockLabel">
   Card Number<span>*</span>
   <input type="text" id="cardNumber" name="cardNumber" />
</label>

<label class="blockLabel">
   Expiration Date<span>*</span>
   <select id="expMonth" name="expMonth">
      <option value="01">January (01)</option>
      <option value="02">February (02)</option>
      <option value="03">March (03)</option>
      <option value="04">April (04)</option>
      <option value="05">May (05)</option>
      <option value="06">June (06)</option>
      <option value="07">July (07)</option>
      <option value="08">August (08)</option>
      <option value="09">September (09)</option>
      <option value="10">October (10)</option>
      <option value="11">November (11)</option>
      <option value="12">December (12)</option>
   </select>
   <select id="expYear" name="expYear">
      <option value="2011">2011</option>
      <option value="2012">2012</option>
      <option value="2013">2013</option>
      <option value="2014">2014</option>
      <option value="2015">2015</option>
   </select>
</label>

</fieldset>
```

Tutorial 6 Website – Result

- Formatting the selection list:

```
#creditCard                {position:absolute; left: 150px}

#cardHolder, #cardNumber {width: 25em}

#expMonth                  {position: absolute; left: 150px}
#expYear                   {position: absolute; left: 280px}
```

- Result:



Making Multiple Selections

- Allowing for Multiple Selections
  - Selection lists allow the user to make multiple selections by adding the attribute:

    ```
    <select multiple="multiple">...</select>
    ```

  - Note:
    - Be aware that the form sends a name/value pair to the server for each option the user selects from the list
    - So the server program must be able to handle this
    - You must verify that the server program can handle multiple selections before using a multiple selection list
    - Otherwise, you are better served using check boxes
      - which are coming up next…

Working with Check Boxes

- Another common control element is known as a **check box**
  - To create a check box, use:

```
<input type="checkbox" name="name" id="id"
value="value" />
```

    - where the name and id attributes identify the check box controls, and the value attribute specifies the value sent to the server if the check box is selected
    - You do not need to use the value attribute
      - In this case, the value "On" is used by default

Working with Check Boxes

- Another common control element is known as a **check box**
  - Example:
    - The following code determines if user is a Democrat:

```
<label for="dem">Democrat?</label>
<input type="checkbox" name="dem" id="dem"
  value="yes" />
```

  - By default, check boxes are not selected
  - To specify that a check box be selected by default, use the checked attribute as follows:

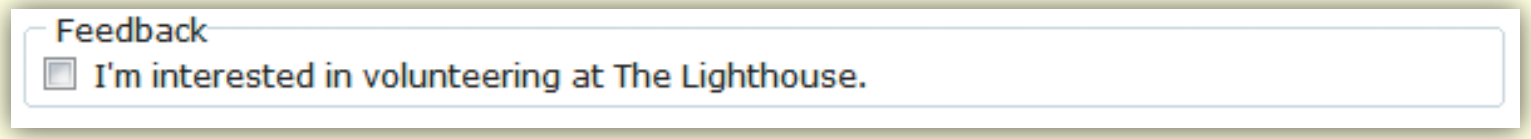
```
<input type="checkbox" checked="checked" />
```

Tutorial 6 Website

- Adding a check box:

```
  <fieldset id="feedback">
     <legend>Feedback</legend>

     <label>
        <input type="checkbox" id="volunteer" name="volunteer" />
        I'm interested in volunteering at The Lighthouse.
     </label>

  </fieldset>

</form>
```

- Result:



Specifying the Tab Order

- Users typically navigate through a form with the tab key.
- You can specify an alternate tab order by adding the tabindex attribute to any control element in your form.
- The syntax is as follows:

```
<input name="fname" id="firstName"
  tabindex="1" />
```

  This syntax assigns the tab index number "1" to the fname field from the registration form.

Working with Text Area Control

- Another common control element is known as a **text area** control
  - This is essentially a text box that allows for multiple lines of text (comments, feedback, etc.)
  - To create a text area, use:

```
<textarea name="name" id="id">
     text
</textarea>
```

    - where text is the default text that is placed in the text box
    - You do not have to specify default text
      - If you don't the text box will display as an empty box

Working with Text Area Control

- Another common control element is known as a **text area** control
  - The size of the box is determined by the browser
    - Most browsers make a box that is about 20 characters long and two to three lines high
    - But you can change the dimensions of the box
    - Syntax:

      `<textarea rows="value" cols="value">…</textarea>`

      - where the rows value specifies the number of lines and the cols value specifies the number of characters per line
    - Text automatically wraps to a new line as it extends beyond a box's width, and scrollbars are added as needed

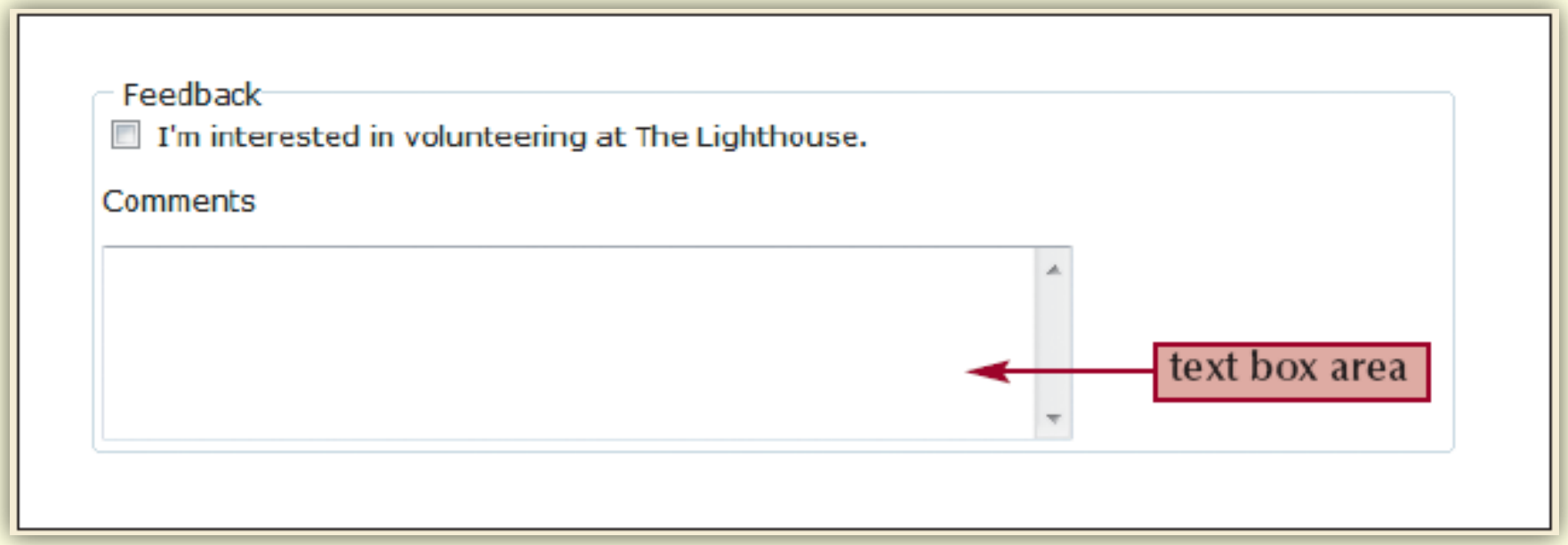
Tutorial 6 Website

- Adding text area box:
  - 50 characters wide by 5 lines high

```
<fieldset id="feedback">
   <legend>Feedback</legend>

   <label>
      <input type="checkbox" id="volunteer" name="volunteer" />
      I'm interested in volunteering at The Lighthouse.
   </label>

   <label for="comments" class="blockLabel">Comments</label>
   <textarea id="comments" name="comments" rows="5" cols="50"></textarea>

</fieldset>
```

the text area box will have five lines of 50 characters each

Tutorial 6 Website – Result

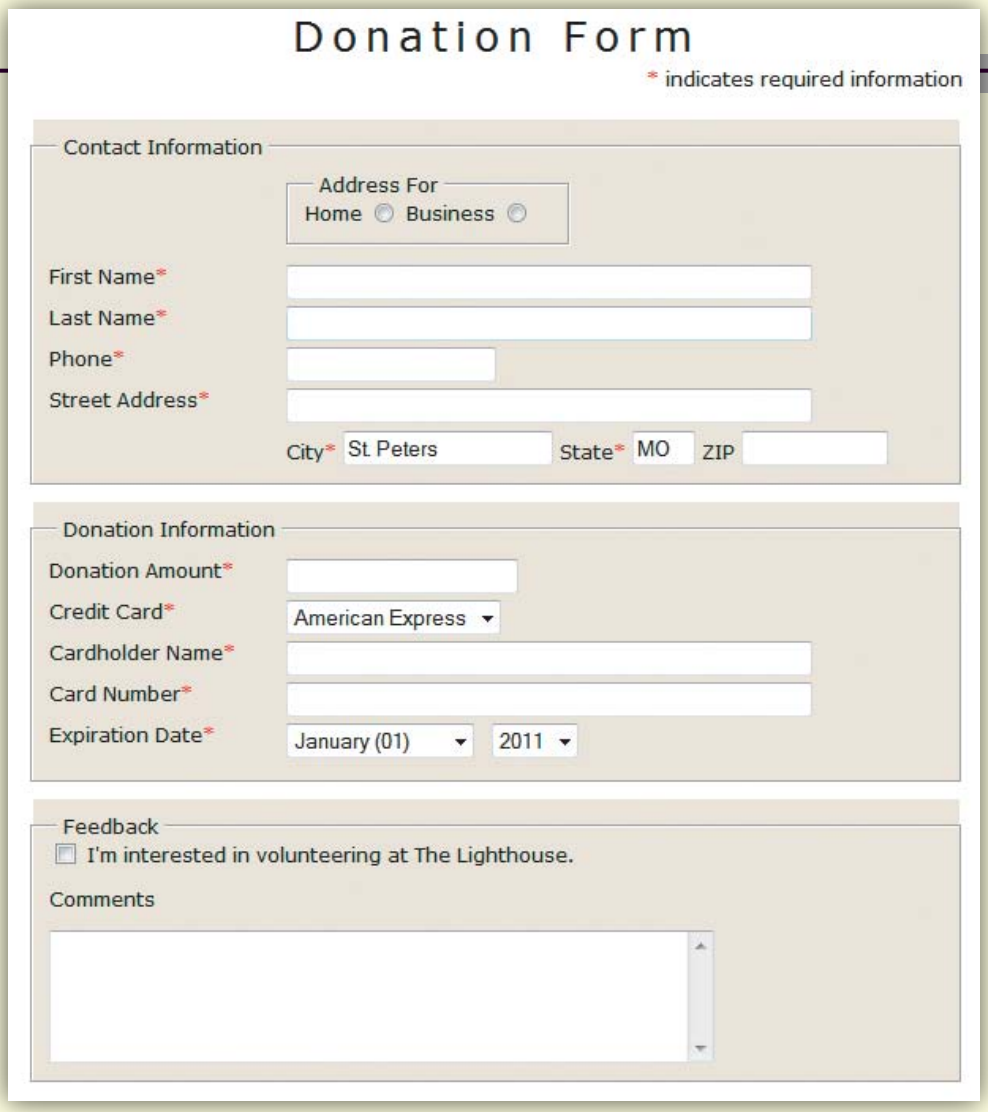


Tutorial 6 Website

- We're almost done with the layout!
  - Terry wants a light brown background
  - And she wants to increase the space between and within each of the field set boxes

```
fieldset                {margin-bottom: 10px; padding: 10px; background-color: rgb(237,233,223)}
label.blockLabel        {display: block; position: relative; margin: 12px 0px}
label.blockLabel input  {position: absolute; left: 150px}
```

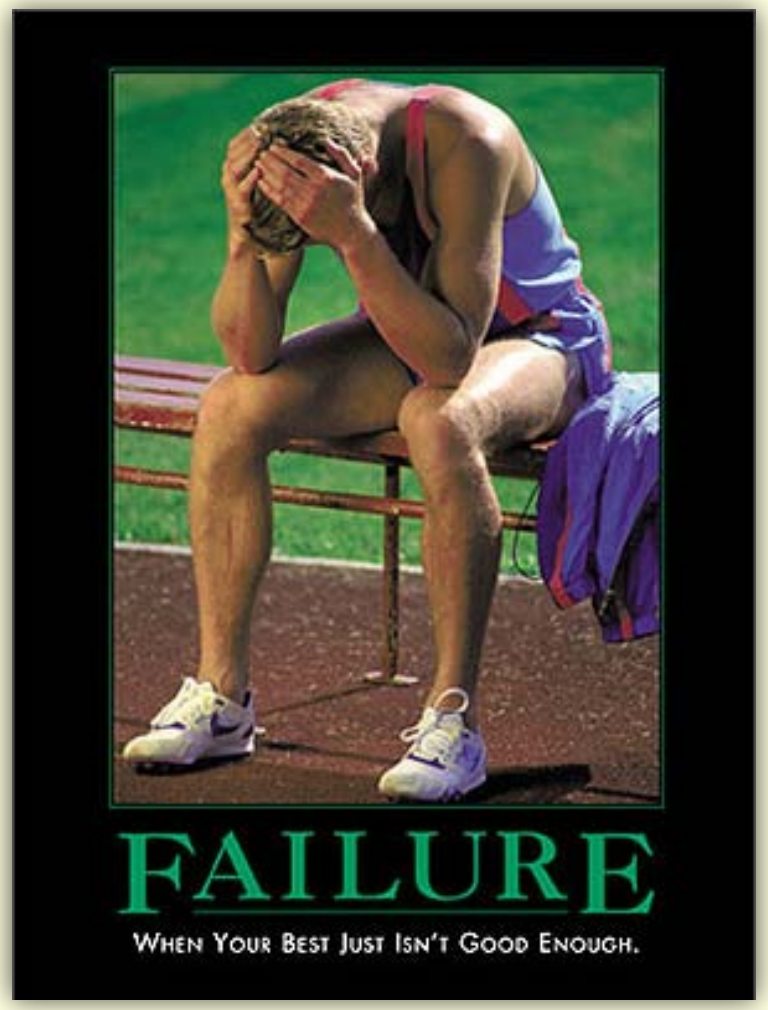
Tutorial 6 Website – Result



Tutorial 6: Working with Web Forms

WASN'T THAT EXCEPTIONAL!

Daily Demotivator



Tutorial 6: Working with Web Forms

Computer Science Department
University of Central Florida

COP 3175 – Internet Applications

Working with Form Buttons

- Until now, we've looked at control elements used to enter field values
- Another type of control element is one that performs an action
  - For forms, this is usually done with **form buttons**
- There are three main types of buttons:
  - Command button
  - Submit button
    - To submit the for to the server-based program
  - Reset button

Creating a Command button

- A command button runs a command on the page
  - Command buttons are created using the <input> tag:

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

    - where text is the text that appears on the button itself
  - To create an action for the command button, you have to write a script (program) that will run when the button is clicked
  - This is clearly beyond the scope of this course
  - Submit buttons submit forms to the server for processing when clicked. Syntax is as follows:

Creating Submit & Reset Buttons

- Submit buttons submit forms to the server for processing when clicked
- Reset buttons reset a form, changing all fields to their original, default values
  - Syntax for the two buttons:

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

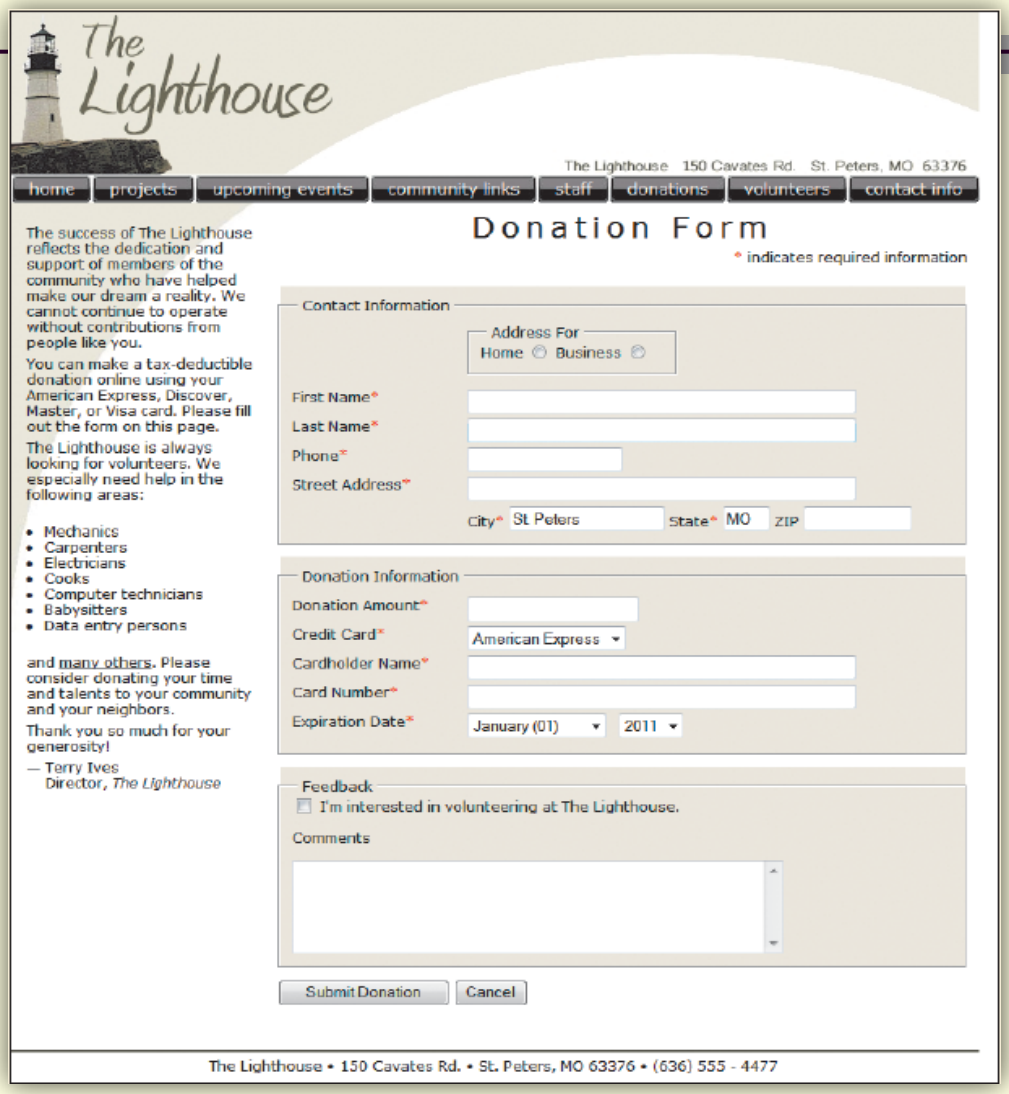
    - where value defines the text that appears on the button
    - You can also specify the name and id attributes
      - But it is not required
      - Only really needed if a form has multiple buttons that you need to distinguish from one another

Tutorial 6 Website

- Adding buttons
  - Submit button should be labeled “Send Donation”
  - Reset button should be labeled “Cancel”

```
        </fieldset>

        <input type="submit" value="Submit Donation" />
        <input type="reset" value="Cancel" />

    </form>

</div>

<address>
    The Lighthouse &bull;
    150 Cavates Rd. &bull;
    St. Peters, MO  63376 &bull;
    (636) 555 - 4477
</address>
```

Tutorial 6 Website – Result



Designing a Custom Button

- The text of command, submit, or reset buttons is determined by the value attribute
  - There's really no other customization allowed
    - Such as adding an image
- For more control over a button's appearance, you can use the actual button element:

```
<button name="name" id="id" value="value"
  type="type">
  content
</button>
```

where the *name* and *value* attributes specify the name of the button and the value sent to a server-based program; the *id* attribute specifies the button's id, the *type* attribute specifies the button type (submit, reset, or button), and the *content* is page content displayed within the button.

Designing a Custom Button

- Example

```
<button name="home" id="home" type="button">
   <img src="homeicon.gif" alt="" style="vertical-align: middle" />
   <span style="color: blue; font-weight: bold; font-style: italic">
     Return to the Home Page
   </span>
</button>
```

HTML code

custom button

Creating File Buttons

- Another type of button is the **file button**
  - Used to select files, allowing them to be submitted for the form
  - Syntax:

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

  - Most browsers render this as input boxes accompanied by a Browse button
    - You cannot change the label of this button
    - But you can increase the size of the input box using CSS or the HTML size attribute

Creating File Buttons

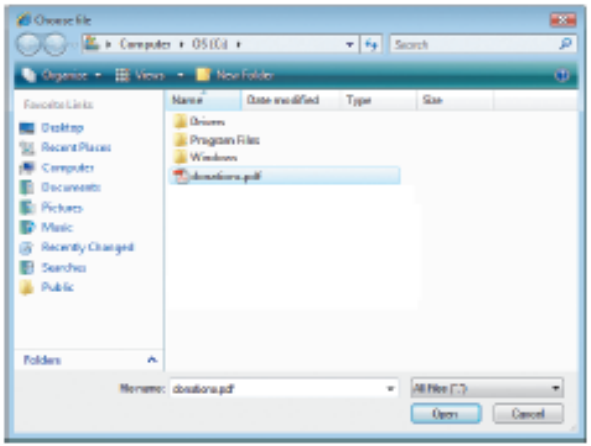
- Example

```
<input type="file" id="filename" name="filename" />
```

HTML code

1. Click the Browse button

2. Select a file from the Choose file window



3. The file name and location are placed in the input box

Working with Hidden Fields

- There are some fields that you don't want the end users to see
  - Example:
    - the form has to be submitted to the server-based program
    - Then this program will process the form and EMAIL it to you
    - This means that the server-program will need to know the email address to send the processed form to
      - which, in turn, means that the form itself must actually include this email address (your address)
      - But there's no need for the end-user to see the address
    - So we use **hidden fields** on the form

Working with Hidden Fields

- There are some fields that you don't want the end users to see
  - **Hidden fields** are added to a form
    - but they are not displayed in the Web page
  - Syntax:

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

  - Note:
    - Since this field is required by the server-based program, you will need to know the exact name of the field, as required by the program

Tutorial 6 Website

- Adding hidden field:
  - Terry checked with the hosting company, and the field with the email should be named eMail
    - Hidden fields can be placed anywhere in the form
    - Common practice is to place them all at the top, making them easier to locate and read
      - which is what we do here…

```
<form name="donationForm" id="donationForm">

  <input type="hidden" name="eMail" id="eMail"
         value="donations@thelighthouse.org" />

  <fieldset id="contact">
     <legend>Contact Information</legend>
```

Brief Interlude: FAIL Pics

Working with Form Attributes

- Graphically, the form is now done!
  - The final task is to specify where to send the form data and how to send it
  - So we go back and modify the form element:

    `<form action="`*`url`*`" method="`*`type`*`" enctype="`*`type`*`">...</form>`
    - where *url* specifies the filename and location of the program that processes the form, the *method* attribute specifies how your Web browser sends data to the server, and the *enctype* attribute specifies the format of the data stored in the form's field
  - We now examine the method attribute in detail

Working with Form Attributes

- The method attribute
  - can have one of two values:
  1) Post
  2) Get
  - The get method is the default; get appends the form data to the end of the URL specified in the action attribute.
  - The post method sends form data in a separate data stream, allowing the Web server to receive the data through “standard input.”
    - This is more common for a variety of reasons

Tutorial 6 Website

- Adding action and method attributes:
  - Terry tells us the URL of the server-based program and that we should use the post method

```
<form name="donationForm" id="donationForm"
      action="http://www.thelighthouse.org/cgi-bin/donation"
      method="post">

  <input type="hidden" name="eMail" id="eMail"
         value="katherinehayes@thelighthouse.org" />
```

Tips for Creating Effective Forms

- Final Tips:
  - Mark fields that are required, but also limit the number of unrequired fields. Don’t overwhelm your users with requests for information that is not really essential. Keep your forms short and to the point.
  - If you need to collect a lot of information, break the form into manageable sections spread out over several pages. Allow users to easily move backward and forward through the forms without losing data.

Tips for Creating Effective Forms

- Final Tips:
  - Provide detailed instructions about what users are expected to do. Don't assume that your form is self-explanatory.
  - If you ask for personal data and financial information, provide clear assurances that the data will be secure. If possible, provide a link to a Web page describing your security practices.
  - Clearly indicate what users will receive once the form is submitted, and provide feedback on the Web site and through e-mail that tells them when their data has been successfully submitted.

From a Hosting Perspective

- You are now equipped to make a form
  - But what “action” and method attributes do you use?
    - Meaning, where do you send this form?
  - This is dependent on the web hosting you have
  - Assuming you administrative interface has cPanel (which we do), you have a couple of common options that are pre-installed:
    - CGI Email
    - FormMail Clone

From a Hosting Perspective

- CGI Email
  - You create a text file that resembles the look of the email that you expect to receive
  - This text file will include the various “fields” from the form
  - Then when the user submits the form, CGI Email, converts the form into an email message, using the format specified, by you (in the text file), and then sends the email to you

From a Hosting Perspective

- FormMail
  - This is a bit more common, if for no other reason that the ease of use
  - You simply provide the necessary action (url) and method attributes
  - And you provide the recipient field
    - Which is the ONLY required filed by this server-based program

      <input type="hidden" name="recipient" value="youremail@yourdomain.com">
    - The form contents are then automatically emailed to you

From a Hosting Perspective

- FormMail
  - There are many different hidden fields that the form can use
  - See the following page for more info:
    - http://www.scriptarchive.com/readme/formmail.html
  - Note: the first half of the page can be very confusing
  - Scroll down about halfway to the section called “Form Configuration”
    - Here it goes through the various preset fields and how they operate

Example Working Form

```
<form name="testform" id="testform"
   action="http://www.mamaluv.us/cgi-sys/formmail.pl" method="post" />
   <input type="hidden" name="recipient" value="cgs3175@mamaluv.us" />
   <input type="hidden" name="subject" value="CGS3175 Form Email" />
   <input type="hidden" name="redirect"
   value="http://www.mamaluv.us/pages/formredirect.html" />
   <input type="hidden" name="required" value="email,name,comment">
   <fieldset id="contact">
        fields here...
   </fieldset>
   <fieldset id="feedback">
        fields here...
   </fieldset>
   <input type="submit" name="submit" value="E-Mail Me!" />
</form>
```

Tutorial 6: Working with Web Forms

WASN'T THAT FABULOUS!

Daily Demotivator

Tutorial 6: Working with Web Forms

Computer Science Department
University of Central Florida

COP 3175 – Internet Applications