Web Programming Step by Step

Lecture 9
HTML Forms
Reading: 6.1 - 6.2, 6.4

Except where otherwise noted, the contents of this presentation are Copyright 2009 Marty Stepp and Jessica Miller.

6.1: Form Basics

- 6.1: Form Basics
- 6.2: Form Controls
- 6.3: Submitting Data
- 6.4: Processing Form Data in PHP
Web data

- most interesting web pages revolve around data
  - examples: Google, IMDB, Digg, Facebook, YouTube, Rotten Tomatoes
  - can take many formats: text, HTML, XML, multimedia
- many of them allow us to access their data
- some even allow us to submit our own new data
- most server-side web programs accept **parameters** that guide their execution

Query strings and parameters (6.1.1)

**URL?name=value&name=value...**

http://example.com/student_login.php?**username=stepp&sid=1234567**

- **query string**: a set of parameters passed from a browser to a web server
  - often passed by placing name/value pairs at the end of a URL
  - above, parameter **username** has value stepp, and sid has value 1234567
- PHP code on the server can examine and utilize the value of parameters
HTML forms

- **form**: a group of UI controls that accepts information from the user and sends the information to a web server
- the information is sent to the server as a query string
- JavaScript can be used to create interactive controls (seen later)

```html
<form action="destination URL">
  form controls
</form>
```

- required `action` attribute gives the URL of the page that will process this form's data
- when form has been filled out and **submitted**, its data will be sent to the `action`'s URL
- one page may contain many forms if so desired
Form example

```html
<form action="http://www.google.com/search">
  <div>
    Let's search Google:
    <input name="q" />
    <input type="submit" />
  </div>
</form>
```

• must wrap the form's controls in a block element such as `div`

### 6.2: Form Controls

- 6.1: Form Basics
- **6.2: Form Controls**
- 6.3: Submitting Data
- 6.4: Processing Form Data in PHP
Form controls: `<input>`

- `input` element is used to create many UI controls
  - an inline element that MUST be self-closed
- `name` attribute specifies name of query parameter to pass to server
- `type` can be `button`, `checkbox`, `file`, `hidden`, `password`, `radio`, `reset`, `submit`, `text`, ...
- `value` attribute specifies control's initial text

Text fields: `<input>` (6.2.1)

- `input` attributes: `disabled`, `maxlength`, `readonly`, `size`, `value`
- `size` attribute controls onscreen width of text field
- `maxlength` limits how many characters user is able to type into field
Text boxes: `<textarea>` (6.2.2)

A multi-line text input area (inline)

```html
<textarea rows="4" cols="20">
Type your comments here.
</textarea>
```

- initial text is placed inside `textarea` tag (optional)
- required `rows` and `cols` attributes specify height/width in characters
- optional `readonly` attribute means text cannot be modified

Checkboxes: `<input>` (6.2.3)

Yes/no choices that can be checked and unchecked (inline)

```html
<input type="checkbox" name="lettuce" /> Lettuce
<input type="checkbox" name="tomato" checked="checked" /> Tomato
<input type="checkbox" name="pickles" /> Pickles
```

- none, 1, or many checkboxes can be checked at same time
- when sent to server, any checked boxes will be sent with value `on`:
- use `checked="checked"` attribute in HTML to initially check the box
Radio buttons: `<input>` (6.2.4)

*sets of mutually exclusive choices (inline)*

```html
<input type="radio" name="cc" value="visa" checked="checked" /> Visa
<input type="radio" name="cc" value="mastercard" /> MasterCard
<input type="radio" name="cc" value="amex" /> American Express
```

- grouped by `name` attribute (only one can be checked at a time)
- must specify a `value` for each one or else it will be sent as `value on`

Text labels: `<label>` (6.2.5)

```html
<label><input type="radio" name="cc" value="visa" checked="checked" /> Visa</label>
<label><input type="radio" name="cc" value="mastercard" /> MasterCard</label>
<label><input type="radio" name="cc" value="amex" /> American Express</label>
```

- associates nearby text with control, so you can click text to activate control
- can be used with checkboxes or radio buttons
- `label` element can be targeted by CSS style rules
Drop-down list: `<select>`, `<option>` (6.2.6)

*menus of choices that collapse and expand (inline)*

```html
<select name="favoritecharacter">
  <option>Jerry</option>
  <option>George</option>
  <option selected="selected">Kramer</option>
  <option>Elaine</option>
</select>
```

- option element represents each choice
- select optional attributes: disabled, multiple, size
- optional selected attribute sets which one is initially chosen

Using `<select>` for lists

```html
<select name="favoritecharacter[]" size="3" multiple="multiple">
  <option>Jerry</option>
  <option>George</option>
  <option>Kramer</option>
  <option>Elaine</option>
  <option selected="selected">Newman</option>
</select>
```

- optional multiple attribute allows selecting multiple items with shift- or ctrl-click
  - must declare parameter's name with `[]` if you allow multiple selections
- option tags can be set to be initially selected
• What should we do if we don't like the bold italic?

6.4: Processing Form Data in PHP

• 6.1: Form Basics
• 6.2: Form Controls
• 6.3: Submitting Data
• 6.4: Processing Form Data in PHP
"Superglobal" arrays (6.4.1)

- PHP **superglobal** arrays (global variables) contain information about the current request, server, etc.:

<table>
<thead>
<tr>
<th>Array</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$_GET, $_POST</td>
<td>parameters passed to GET and POST requests</td>
</tr>
<tr>
<td>$_REQUEST</td>
<td>parameters passed to any type of request</td>
</tr>
<tr>
<td>$_SERVER, $_ENV</td>
<td>information about the web server</td>
</tr>
<tr>
<td>$_FILES</td>
<td>files uploaded with the web request</td>
</tr>
<tr>
<td>$_SESSION, $_COOKIE</td>
<td>&quot;cookies&quot; used to identify the user (seen later)</td>
</tr>
</tbody>
</table>

- These are special kinds of arrays called **associative arrays**.

**Example: Exponents**

```php
<?php
ibase = $_REQUEST["base"];  
exp = $_REQUEST["exponent"];  
$result = pow(base, exp);  
?>

3 ^ 4 = 81
```

http://example.com/exponent.php?base=3&exponent=4

output

3 ^ 4 = 81
Example: Print all parameters

```php
<?php
foreach ($_REQUEST as $param => $value) {
    ?>
}
?>
```

http://example.com/print_params.php?name=Marty+Stepp&sid=1234567

Parameter name has value Marty Stepp
Parameter sid has value 1234567

- or call `print_r` or `var_dump` on `$_REQUEST` for debugging