Study Unit 10: PHP

Outline

- Parameterized Pages
- Form Basics
- Form Controls
- Processing Form data in PHP

Learning Outcomes of Study Unit 10

You will learn the basic concepts query string and parameters, Form fields, text, select, text area, checkbox and Form submission and controls in PHP.
1.1: Query String & Parameters
1.2: Form Basics,
1.3: Form Controls
Input, type: text, password, textarea, checkbox, radio, label, select, option
1.4: Processing Form data in PHP
Superglobal Arrays

Outline

- **1.1: Parameterized Pages**
- **Form Basics**
- **Form Controls**
- **Processing Form data in PHP**

Web Data

- Most interesting web pages revolve around data
  - Examples: Google, Baidu, IMDB, Digg, Facebook, YouTube, renren
  - 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

```
URL?name=value&name=value...
```

```
http://www.google.com/search?q=Obama
http://example.com/student_login.php?username=stepp&id=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 id has value 1234567
- PHP code on the server can examine and utilize the value of parameters
- A way for PHP code to produce different output based on values passed by the user

Query Parameters: $_REQUEST

```php
$user_name = $_REQUEST["username"];
$id_number = (int) $_REQUEST["id"];
$eats_meat = FALSE;
if (isset($_REQUEST["meat"])) {
  $eats_meat = TRUE;
}
```

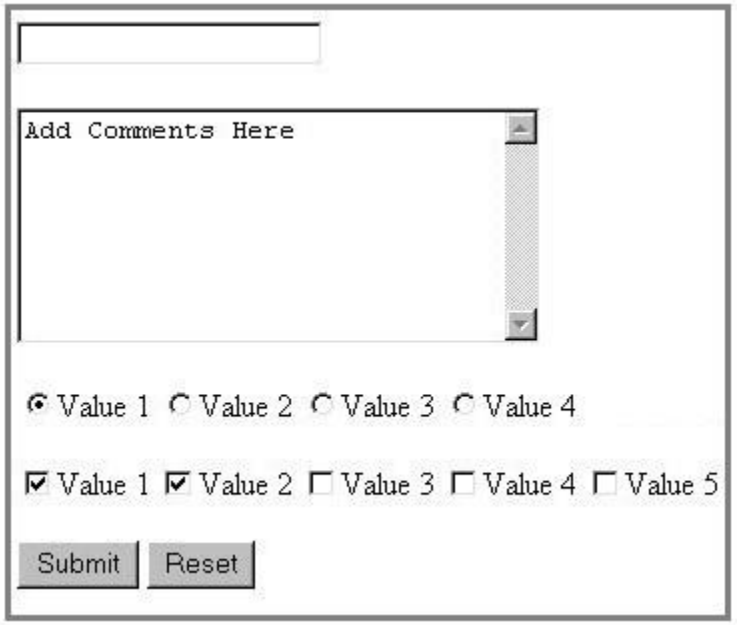
- $_REQUEST["*parameter name*"] returns a parameter's value as a string
- Test whether a given parameter was passed with isset

Outline

- Parameterized Pages
- **1.2: Form Basics**
- **Form Controls**
- **Processing Form data in PHP**

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**



HTML Form: <form>

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

- 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

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

HTML

output

- Must wrap the form's controls in a block element such as div, fieldset, etc.

Outline

- Parameterized Pages
- Form Basics
- **1.3: Form Controls**
- **Processing Form data in PHP**

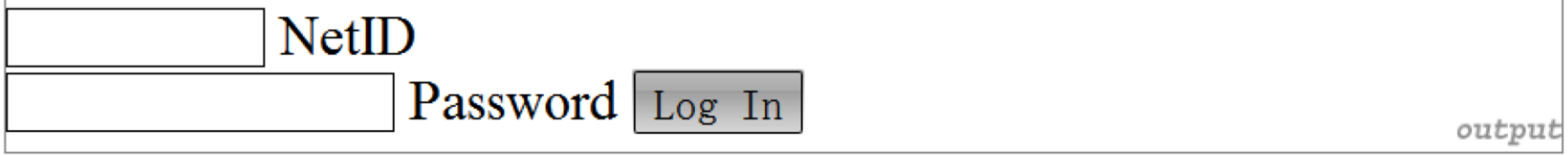
```HTML
<!-- 'q' happens to be the name of Google's required parameter -->
<input type="text" name="q" value="Colbert Report" />
<input type="submit" value="Booyah!" />
```

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>

```HTML
<input type="text" size="10" maxlength="8" /> NetID <br />
<input type="password" size="16" /> Password
<input type="submit" value="Log In" />
```



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

Type your comments here.

output

- 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>

- 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

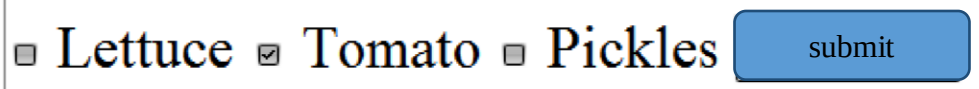
Checkboxes: <input>

- None, 1, or many checkboxes can be checked at same time
- When sent to server, any checked boxes will be sent with value on:
  - http://ssw2p.3322.org/public/params.php?tomato=on&pickles=on

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

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

HTML



output

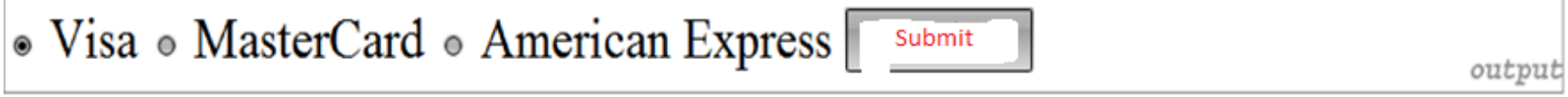
- Use checked="checked" attribute in HTML to initially check the box

Radio Buttons: <input>

sets of mutually exclusive choices (inline)

```
<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
```

HTML



output

- 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**

Think of <input>

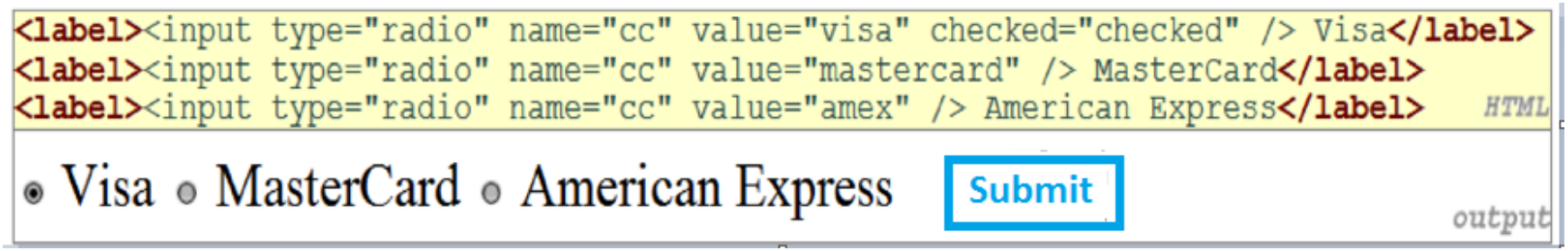
- So many types of input, why **NOT** use elements instead?
- <input type="text" … /> ➔ <text/> or < text></text > <input type="checkbox" … /> ➔ < checkbox …/ >

- In fact, it is just a bad design decision when form was firstly designed and introduced into html in 1996, and we follow it so far…,
- Another flaw: checked="checked" …, is it weird?

- Lessons:
- Reality is never, ever perfect
- **BUT** we will try out best to make it perfect

```
<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>
```

HTML



Text Labels: <label>

- Associates nearby text with control, so you can click text to activate control
- Can be used with checkboxes or radio buttons
- Either wrap the input elements or target input elements with id specified via the "for" attribute
- label element can be targeted by CSS style rules
- Reasons for preferring label than text:

- **Functionality**: can be directly clicked on
- **Styling**: can be styled by CSS rules
- **Accessibility**: screen reader will read it when selected

Drop-down List: <select>, <option>

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>
```

HTML

output

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

Using <select> for Lists

```
<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>
```

HTML

output

- 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

Option Groups: <optgroup> *Web 2.0 Programming*

```HTML
<select name="favoritecharacter">
  <optgroup label="Major Characters">
    <option>Jerry</option>
    <option>George</option>
    <option>Kramer</option>
    <option>Elaine</option>
  </optgroup>
  <optgroup label="Minor Characters">
    <option>Newman</option>
    <option>Susan</option>
  </optgroup>
</select>
```

output

Outline

- Parameterized Pages
- Form Basics
- Form Controls
- **1.4: Processing Form data in PHP**

"Superglobal" Arrays

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

Array	Description
$_GET, $_POST	parameters passed to GET and POST requests
$_REQUEST	parameters passed to any type of request
$_SERVER, $_ENV	information about the web server
$_FILES	files uploaded with the web request
$_SESSION, $_COOKIE	"cookies" used to identify the user (seen later)

- These are special kinds of arrays called associative arrays.

Questions For Study Session 10

Now that you have completed this study unit, you can assess how well you have achieved its Learning Outcomes by answering these questions. Write your answers in your Study Diary and discuss them with your Tutor at the next Study Support Meeting or Online interactive sessions. You can also check your answers at the Self-Review Answers section which is located at the end of this Module.

1: Convert it to a PHP page on your Web server, which shows data you submitted at the top of this page.	**PHP Form Validation Example** * required field Name: * E-mail: * Website: Comment: Gender: Female Male Other * Submit **Your Input:**

Self-Review Answers (SRA) for Study Unit 10

1: Convert it to a PHP page on your Web server, which shows data you submitted at the bottom of this page.

```
<!DOCTYPE HTML>
<html>
<head>
<style>
.error {color: #FF0000;}
</style>
</head>
<body>

<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
    // check if name only contains letters and whitespace
    if (!preg_match("/^[a-zA-Z-' ]*$/",$name)) {
      $nameErr = "Only letters and white space allowed";
    }
  }
```

```
  if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else {
    $email = test_input($_POST["email"]);
    // check if e-mail address is well-formed
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
      $emailErr = "Invalid email format";
    }
  }

  if (empty($_POST["website"])) {
    $website = "";
  } else {
    $website = test_input($_POST["website"]);
    // check if URL address syntax is valid (this regular expression also allows dashes in the URL)
    if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
9+&@#\/%=~_|]/i",$website)) {
      $websiteErr = "Invalid URL";
    }
  }

  if (empty($_POST["comment"])) {
    $comment = "";
  } else {
    $comment = test_input($_POST["comment"]);
  }

  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
```

```
  } else {
    $gender = test_input($_POST["gender"]);
  }
}

function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
}
?>

<h2>PHP Form Validation Example</h2>
<p><span class="error">* required field</span></p>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
  Name: <input type="text" name="name" value="<?php echo $name;?>">
  <span class="error">* <?php echo $nameErr;?></span>
  <br><br>
  E-mail: <input type="text" name="email" value="<?php echo $email;?>">
  <span class="error">* <?php echo $emailErr;?></span>
  <br><br>
  Website: <input type="text" name="website" value="<?php echo $website;?>">
  <span class="error"><?php echo $websiteErr;?></span>
  <br><br>
  Comment: <textarea name="comment" rows="5" cols="40"><?php echo $comment;?></textarea>
  <br><br>
  Gender:
  <input type="radio" name="gender" <?php if (isset($gender) &&
```

```
$gender=="female") echo "checked";?> value="female">Female
  <input type="radio" name="gender" <?php if (isset($gender) &&
$gender=="male") echo "checked";?> value="male">Male
  <input type="radio" name="gender" <?php if (isset($gender) &&
$gender=="other") echo "checked";?> value="other">Other
  <span class="error">* <?php echo $genderErr;?></span>
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>

<?php
echo "<h2>Your Input:</h2>";
echo $name;
echo "<br>";
echo $email;
echo "<br>";
echo $website;
echo "<br>";
echo $comment;
echo "<br>";
echo $gender;
?>

</body>
</html>
```

References and Additional Reading Materials

- PHP home page: http://www.php.net/
- W3Schools PHP tutorial: http://www.w3schools.com/PHP/