Developing Server Side Content Using PHP Technology

CGT 356
Web Programming, Development, & Database Integration
Lecture 3
Resources

- http://www.php.net/docs.php
- And many others… just search for it
What is PHP?

- Hypertext PreProcessor
- When first released... sometimes it was referred to as Paul's Home Page
- PHP is a server-side HTML embedded scripting language. It provides web developers with a full suite of tools for building dynamic websites
About PHP

- PHP can run anywhere
  - Linux
  - Windows
  - Mac

- PHP is Open Source
  - “human knowledge belongs to the world”
    - Antitrust (2001)
About PHP

Capabilities:

- Browser independent
- Access databases, install application logic, accept user input via forms, send email, etc.
- Replacement for CGI programming
Needs met by PHP?

- Dynamic content deployment
  - Links and content that changes multiple times each day [news sites, portal sites]
  - Application logic needed to perform calculations or perform logical functions.
  - Interface to “outside programs”
    - Custom Active X controls
How does it compare to CGI?

- Replacement for CGI
- Pools systems resources more efficiently than CGI. Threads can share.
- Guarantees the presence of a core set of methods and functions.
- Faster than CGI.
How does dynamic content (PHP, ASP, JSP, ASP.NET) work?

- Using specialized tags within HTML code to access methods provided PHP.
- Author constructs templates that are filled in at run time.
Server-Side Includes in PHP

- **Syntax**
  - `<?php include("openDbConn.php"); ?>`
  - *or*
  - `<?php include($absolutePath."xincludesx/openDbConn.php"); ?>`
Server-Side Includes in HTML

- Yes, you can have them in a plain HTML document.
- Virtual includes (still a Server Side Include)
  
  <!-- #include virtual="filename.htm" -->
General Format of PHP

- No general declaration at line 1

- Code segments encapsulated by
  ```php
  echo "hi";
  ```

- Plain HTML cannot appear inside `<?php ?>` alone, but can be alternated
  ```php
  echo "<h1> Hello </h1>";
  
  <h1> echo "Hello"; </h1>
  ```
Variables

- In PHP
  - No dimensioning required:
    - $login = $_POST["login"];
    - In PHP, variables start with a $
echo and print in PHP

- echo “Hello, how are you?”;
- print “I am fine, how are you?”;
- For our purposes, echo and print are interchangeable. There is a difference between them, but it is not necessary to know for this course. As a rule for this course, use echo.
Example: Hello World

- Display Server Time
- Print Hello World increasing size
  - Expression Format
    - expression format
    - helloworld_expression.php
    - http://cgtmm2.tech.purdue.edu/356/rjglotzbach/Lecture03Examples/helloworld_expression.php
  - Encapsulated HTML
    - encapsulated format
    - helloworld_encapsulated.php
    - http://cgtmm2.tech.purdue.edu/356/rjglotzbach/Lecture03Examples/helloworld_encapsulated.php
More General Concepts

- if statements do not need to be contiguous
  - if(login=="page1" && password=="alpha")
    
    `?>
    <a href="http://www.purdue.edu">Purdue</a>
    Place a page of text or more html elements in here… as much as needed OR… use a Redirect here
    `?>php
    }

- Client never sees the code on the server
  - Mix and Match of languages
  - Security
Requesting Data in PHP

- $_POST
  - `echo $_POST["login"];`
    - Writes the value of the variable login to the browser

- $_GET
  - `echo $_GET["id"];`
    - Writes the value of the variable id to the browser
Get vs. Post

- **Get**
  - Encoded into URL
    - Visible to users
    - Not good for passwords
  - Limited by size
  - Retrieval in PHP:
    - `$_GET["name"]`

- **Post**
  - Encoded into page header
    - Not visible to users
  - Not limited by size
  - Retrieval in PHP:
    - `$_POST["name"]`
Logging Into a system

- Hard coded server side security
  - Login.php (form)
  - Login_Action.php (form_handler)
- Examine Get vs. Put
- Add cookies to recognize user
  - Session Object
Design Form

- Use HTML to design the form
- login1.html
  - http://cgtmm2.tech.purdue.edu/356/rjglotzbach/Lecture03Examples/Login1/login1.html

- Get method is the default <watch out>
- Set the Action to the form handler
- Name of each field is used to access on “the other side”
Design Login_Action Page

- login_action.php
  - $_GET or $_POST retrieves FORM information using the assigned name
  - Get is encoded into Query String
    - Retrieved using $_GET["fieldname"]
  - Post is sent in the header
    - Retrieved using $_POST["fieldname"]
Design Form (again)

- Use HTML to design the form
- login2.html
  - http://cgmm2.tech.purdue.edu/356/rjglotzbach/Lecture03Examples/Login2/login2.html

- Method now equals Post
- Keep Action set to the form handler
- Name of each field used on “the other side”
Redirecting in PHP

- `header()`
  - Example: `header("Location: example.php");`

- in PHP, `header()` must occur before any output to the browser (HTML tags, echo statements, etc)
Redirecting the user

- Alter login_action.php
  - Verify user has the proper login information
  - Use header();
  - login3.htm
General Concepts - PHP

- if(($login=="page1") && ($password=="alpha"))
  {
      ?>
      Page 1 html goes here
      <?php
  }

- Comparison operator in PHP is ==
- Logical AND in PHP is &&
- Logical OR in PHP is ||