CSC 551: Web Programming

Spring 2004

Server-side programming & PHP

- server-side includes
- Netscape LiveWire
- Active Server Pages (ASP)
- Java Servlets
- PHP
  - variables, control statements, file access
  - form handling
  - file uploading, email redirection

Server-side alternatives

CGI is one approach to server-side programming

- general-purpose, can use any language (e.g., C++, perl)
- CGI program runs as a separate process, spawned by the Web server

other server-side alternatives exist

- server-side includes
- Netscape LiveWire
- Active Server Pages (ASP)
- Java Servlets
- PHP

Accessing CGI programs via server-side includes

CGI programs can be called

- directly using the full URL (as in hello, fortune)
- via form submission (as in helloEcho, emailDB, grades, NCAA)

- using embedded server-side includes

```
<!-- #exec cgi="URL" -->
```

must store page with extension .shtml:
tells server to scan for #exec command, calls CGI program, replaces with output
Web server must be configured to allow server-side includes

```
<html>
<!-- fortune.shtml -->

<head>
  <title>embedded CGI call</title>
</head>

<body>
  <table border=1 align="center">
    <tr><td>
      <!-- #exec cgi="/cgi-bin/fortune.cgi" -->
  </table>
</body>
</html>
```


Server-side development (server-specific)

Netscape and Microsoft both have established platforms for Web development and Web programming

- Netscape ONE
  group of technologies packaged for *crossware support*, including
  - Enterprise server, Netscape Communicator
  - Java and JavaScript client-side programming
    Internet Foundation Classes, now part of Java Foundation Classes
  - LiveWire (server-side JavaScript) for server-side programming
  - Dynamic HTML with Cascading Style Sheets
  - component communication using CORBA & JavaBeans model

- Microsoft DNA (Distributed Net Applications architecture)
  group of Microsoft-specific technologies, including
  - Internet Information Server (IIS), Internet Explorer
  - Java, JScript & Visual Basic for client-side programming
    Application Framework Classes
  - Active Server Pages for server-side programming
  - Dynamic HTML with Cascading Style Sheets
  - component communication using COM & ActiveX

```
<html>
<head>
  <title>Server-side Fortune</title>
</head>
<server>
  list = ["Live long and prosper",
          "Save for tomorrow",
          "You will meet someone"];
  fortune = list[Math.floor(Math.random()*list.length)];
</server>
<body>
  <table border=1 align="center">
  <tr><td>
    <server> write(fortune); </server>
  </table>
</body>
</html>
```

server-side JavaScript via Netscape's LiveWire

- code is processed by Web server before downloading
- must compile the page using the Netscape compiler

  fortune.html → fortune.web
- must add the application to the Web server Application Manager

```
<%@ language=javascript %>
<html>
<head>
  <title>Server-side Fortune</title>
</head>
<%
  list = ["Live long and prosper",
          "Save for tomorrow",
          "You will meet someone"];
  fortune = list[Math.floor(Math.random()*list.length)];
%>
<body>
  <table border=1 align="center">
    <tr><td>
      <% = fortune %>
  </table>
</body>
</html>
```

Microsoft's Active Server Pages (ASP's)

- does not require any special compilation or installation (must use .asp extension)
- must specify JavaScript (VBScript is the default)
- code is embedded in <% %>
- access values using <% = %>

Server-side development (server independent)

Java servlets are the server-side counterparts of applets

- servlet API defines input/output behavior, similar to CGI
- must install servlet plug-in for server (but versions exist for diff. servers)
- each servlet must be compiled (e.g., with Java SDK) and stored in special directory
- unlike CGI, servlet is loaded and executed as part of the Web server
  servlet is loaded & initialized once, each call spawns new threads
  - servlets tend to be faster than CGI since run in the Web server process
    reduces overhead since don't spawn new process each execution
  - since servlets continue to run within the server, they are capable of saving status information

Servlet example

servlet inherits from HttpServlet class

doGet method handles HTTP GET requests
- arguments are objects corresponding to input/output

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io

public class Greeting extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse response)
                                                  throws ServletException, IOException {
      PrintWriter returnHTML;

      response.setContentType("text/html");

      returnHTML = response.getWriter();
      returnHTML.println("<html><head><title>A simple GET servlet</title></head>");
      returnHTML.println("<body><h2>This is your servlet answering</h2></body></html>");
      returnHTML.close();
  }
}
```

can write to response object

Servlet is called from HTML using form action:

```
<form action="servlets/Greeting" method="get">

</form>
```


PHP

developed in 1995 by Rasmus Lerdorf (member of the Apache Group)

- originally designed as a tool for tracking visitors at Lerdorf's Web site
- within 2 years, widely used in conjunction with the Apache server

- developed into full-featured, scripting language for server-side programming
- free, open-source
- server plug-ins exist for various servers

PHP is similar to JavaScript, only server-side

- PHP code is embedded in HTML using tags
- when a page request arrives, the server recognizes PHP content via the file extension (`.php`, `.php3`, or `.phtml`)
- the server executes the PHP code, substitutes output into the HTML
- the resulting page is then downloaded to the client

- user never sees the PHP code, only the output in the page

PHP execution

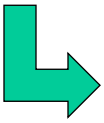
PHP code can be embedded within a `<?php...?>` tag

- output is displayed using `print`

```
<!--  hello.php  -->
<html>
<head>
  <title>Server-side Hello</title>
</head>
<body>
  <table border=1 align="center">
    <tr><td>
      <?php print("Hello and welcome to <i>my</i> page!"); ?>
  </table>
</body>
</html>
```

view page in browser

the server executes the print statement, substitutes output, downloads resulting page



```
<!--  hello.php  -->
<html>
<head>
  <title>Server-side Hello</title>
</head>
<body>
  <table border=1 align="center">
    <tr><td>
      Hello and welcome to <i>my</i> page!
  </table>
</body>
</html>
```


PHP variables

similar to JavaScript, PHP variables are not declared, dynamically typed

- scalar data types: Boolean, integer, double, string
- compound data types: array, object
- special types: resource, NULL

- all variable names begin with `$`

variables are flexible

- any unassigned variable has value NULL
- can test if NULL using IsSet e.g., `IsSet($name)`
- can set mode so that unbound access is reported, or automatically coerced to default values

```
<html>
<head>
  <title>Server-side Fortune</title>
  <?php
    $list = array("Live long and prosper",
                  "Save for tomorrow",
                  "You will meet someone");

    $fortune = $list[rand(0, count($list)-1)];
  ?>
</head>
<body>
  <table border=1 align="center">
    <tr><td>
      <?php print($fortune); ?>
  </table>
</body>
</html>
```

view page in browser

PHP: handling form data

can write server-side programs in PHP as an alternative to CGI

- no need for CGIinput class, can use either GET or POST
- form data is automatically accessible via variable with form element name
- similar to Perl, '.' is used for string concatenation

```
<html>  <!-- helloNice.php -->
<head>  <title>Server-side Greeting</title> </head>
<body>
  <?php
    $yourName = $_POST['yourName'];

    print("Hello and welcome to my page <i>$yourName</i>.<br />\n");
    print("If you like it, " .
          "<a href='mailto:davereed@creighton.edu'>email me</a>!\n");
  ?>
</body>
</html>
```

```
<html>  <!-- helloNicePHP.html -->
<head>  <title>PHP Form Handling</title> </head>
<body>
<form action="http://empirical.cs.creighton.edu/~davereed/helloNice.php"
      method="post">
  Enter your name: <input type="text" name="yourName"/>
  <br /><br />
  <input type="submit" value="click for greeting" />
</form>
</body>
</html>
```

view page in browser

PHP: email database example

```
<html>
<head> <title>PHP Email DB</title>
  <?php
    $emailDB = array("Jim Carlson", "carlsn@creighton.edu",
                     "Davendar Malik", "malik@creighton.edu",
                     "Prem Nair", "psnair@creighton.edu",
                     "Dave Reed", "davereed@creighton.edu",
                     "Mark Wierman", "wierman@creighton.edu");
  ?>
</head>
<body>
  <?php
    $person = $_POST['person'];
    print("Search results for: " . $person . "<br /><br />\n");

    $found = false;
    for ($i = 0; $i < count($emailDB); $i+=2) {
      if ($person == "" ||
          strpos("?".strtolower($emailDB[$i]), strtolower($person))) {
            $entry1 = $emailDB[$i];
            $entry2 = $emailDB[$i+1];
            print("$entry1: <a href=\'mailto:$entry2\'>$entry2</a><br />\n");
            $found = true;
      }
    }

    if (!$found) {
      print("No matching names were found.  Please try again.<br />\n");
    }
  ?>
</body>
</html>
```

since PHP source is not seen by user, can embed protected data safely

string functions include:

- strlen
- strcmp
- strpos
- substr
- strtolower
- strtoupper
- trim

view page in browser

PHP: file example

```
<html>
<head>
  <title>PHP Email DB</title>
  <?php
    $emailDB = file("email.txt");
  ?>
</head>
<body>
  <?php
    $person = $_POST['person'];
    print("Search results for: " . $person . "<br /><br />\n");

    $found = false;
    for ($i = 0; $i < count($emailDB); $i+=2) {
      if ($person == "" ||
          strpos("?".strtolower($emailDB[$i]), strtolower($person))) {
            $entry1 = $emailDB[$i];
            $entry2 = $emailDB[$i+1];
            print("$entry1: <a href=\'mailto:$entry2\'>$entry2</a><br />\n");
            $found = true;
      }
    }

    if (!$found) {
      print("No matching names were found.  Please try again.<br />\n");
    }
  ?>
</body>
</html>
```

various routines exist for reading/writing files

simplest is file, which reads a file into an array of strings (one per line)

view page in browser

Word ladder in PHP

```
<html>
<head>
  <title>Word Ladder Challenge</title>
  <?php
    $dictionary = file("words5.txt");
    $start = $dictionary[rand(0, count($dictionary)-1)];
    $end = $dictionary[rand(0, count($dictionary)-1)];
  ?>
</head>

<body>
  <div style="text-align:center">
  <h2>Dave's Word Ladder Challenge</h2>

  <p>Create a word ladder between <b> <?php print($start); ?> </b>
     and <b> <?php print($end); ?> </b>

  <form name="ladderForm"
        action="http://empirical.cs.creighton.edu/~davereed/ladderCheck.php"
        method="post">
    <textarea name="ladder" rows=10 cols=8 wrap="virtual"></textarea>
    <br /><br />
    <input type="hidden" name="start" value="<?php print($start); ?>">
    <input type="hidden" name="end" value="<?php print($end); ?>">
    <input type="submit" value="Submit Ladder">
  </form>
  </div>
</body>
</html>
```

`ladder.php` contains start of game:

- PHP displays start & end words
- static HTML gives form with text area & submit button

view page in browser

Word ladder in PHP

```
<html>
<head>
  <title>Word Ladder Challenge</title>
  <?php
    function binsearch($needle, $haystack)
    {
      $high = count($haystack)-1;
      $low = 0;

      while ($low <= $high){
        $middle = floor(($high + $low) / 2);
        if (trim($haystack[$middle]) == trim($needle)){
           return true;
        }
        else if (trim($haystack[$middle]) < trim($needle)){
           $low = $middle+1;
        }
        else{
           $high = $middle-1;
        }
      }
      return false;
    }

    function LetterDiff($word1, $word2)
    {
        $diffCount = 0;
        for ($z = 0; $z < strlen(word1); $z++) {
            if ($word1{$z} != $word2{$z}) {
                $diffCount++;
            }
        }
        return $diffCount;
    }
  ?>
</head>
```

`ladderCheck.php` page

- defines functions for checking a ladder
- static HTML gives form with text area & submit button


```
<body>
  <div style="text-align:center">
  <h2>Dave's Word Ladder Challenge</h2>

  <?php
    $dictionary = file("words5.txt");

    $start = trim($_POST['start']);
    $end = trim($_POST['end']);
    $ladder = trim($_POST['ladder']);

    $ladderArr = preg_split("/[\n\s]+/", $ladder);

    if ($start != $ladderArr[0]) {
        print("INVALID LADDER: starting word should be $start<br />");
    }
    else if ($end != $ladderArr[count($ladderArr)-1]) {
       print("INVALID LADDER: ending word should be $end<br />");
    }
    else {
       $OK = true;
       for ($i = 1; $i < count($ladderArr) and $OK; $i++) {
         if (!binsearch($ladderArr[$i], $dictionary)) {
           print("INVALID LADDER: ".$ladderArr[$i]." is not a legal word.<br />");
           $OK = false;
         }
         else if (LetterDiff($ladderArr[$i], $ladderArr[$i-1]) != 1) {
           print("INVALID LADDER: ".$ladderArr[$i]." is not connected to" .
                 $ladderArr[$i-1].".<br />");
           $OK = false;
         }
       }

      if ($OK) {
        print("$ladder is a valid ladder. Congratulations!<br />");
      }
    }
  ?>

  <form name="ladderForm"
        action="http://empirical.cs.creighton.edu/~davereed/ladder.php" method="post">
    <input type="submit" value="Play Again?">
  </form>
  </div></body></html>
```

rest of `ladderCheck.php`

- gets data from form elements
- verifies the correctness of the ladder

view page in browser

Standalone PHP

previous examples have shown how PHP code can be nested in HTML

- browsers don't really require HTML, HEAD, BODY tags
- if no static content in the page, can ignore HTML elements
  - PHP output text will be treated as the body of a default Web page

```
<?php
    $emailDB = file("email.txt");

    $person = $_POST['person'];
    print("Search results for: " . $person . "<br /><br />\n");

    $found = false;
    for ($i = 0; $i < count($emailDB); $i+=2) {
      if ($person == "" ||
          strpos("?".strtolower($emailDB[$i]), strtolower($person))) {
            $entry1 = $emailDB[$i];
            $entry2 = $emailDB[$i+1];
            print("$entry1: <a href=\'mailto:$entry2\'>" .
                  "$entry2</a><br />\n");
            $found = true;
      }
    }

    if (!$found) {
      print("No matching names were found.  Please try again.<br />\n");
    }
 ?>
```

view page in browser

Another example: file uploading

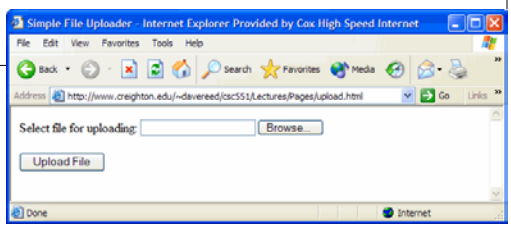
the HTML `file` input element allows the user to browse for a file

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

- once the user selects a file, can use a submit button to call a CGI or PHP program to process that file

```
<html>
<head>
  <title>Simple File Uploader</title>
</head>

<body>
  <form name="uploader" action="http://empirical.cs.creighton.edu/~davereed/upload.php"
        enctype="multipart/form-data" method="post">
    Select file for uploading: <input type="file" name="userfile">
    <br /><br />
    <input type="submit" value="Upload File">
  </form>
</body>
</html>
```



PHP file manipulations

PHP provides extensive support for file/directory manipulation

`$_FILES[$FORM_ELE_NAME]['name']` — original name of the file uploaded via the specified form input element

`$_FILES[$FORM_ELE_NAME]['tmp_name']` — temporary name of the file where it is uploaded onto the server

`move_uploaded_file($_FILES[$FORM_ELE_NAME]['tmp_name'], $FILE_PATH_NAME)` — copies uploaded file to specified loc.

```
<?php
$BASEDIR = "/var/www/davereed/files/";

if (!file_exists($BASEDIR.$_FILES['userfile']['name'])) {
    move_uploaded_file($_FILES['userfile']['tmp_name'],
                       $BASEDIR.$_FILES['userfile']['name']);
    print("File uploaded successfully");
}
else {
    print("File already exists - no upload performed.");
}
?>
```

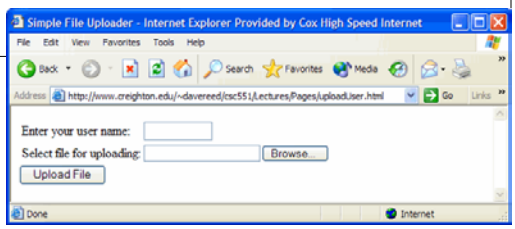

Robust file uploading

could utilize other PHP features to make file uploading more robust

- allow multiple students to submit same assignment
- each student specifies a user name, file is uploaded into a subdirectory

```
<html>
<head>
  <title>Simple File Uploader</title>
</head>

<body>
  <form name="uploader" action="http://empirical.cs.creighton.edu/~davereed/upload.php"
        enctype="multipart/form-data" method="post">
    <table>
    <tr><td>Enter your user name:
        <td><input type="text" name="userID" size=10 value="">
    <tr><td>Select file for uploading:
        <td><input type="file" name="userfile">
    </table>
    <input type="submit" value="Upload File">
  </form>
</body>
</html>
```



Robust file uploading

```
<?php
$userID = $_POST['userID'];
$BASEDIR = "/var/www/davereed/files/";

$_FILES['userfile']['name'] = explode(' ', $_FILES['userfile']['name']);
$_FILES['userfile']['name'] = implode('_', $_FILES['userfile']['name']);

if (IsSet($userID)) {
  $BASEDIR = $BASEDIR.$userID."/";
  if (!file_exists($BASEDIR)) {
    mkdir($BASEDIR, 755);
  }
}

if (!file_exists($BASEDIR.$_FILES['userfile']['name'])) {
    move_uploaded_file($_FILES['userfile']['tmp_name'],
                       $BASEDIR.$_FILES['userfile']['name']);
    print("File uploaded successfully");
}
else {
    print("File already exists - no upload performed.");
}
?>
```

get the user ID from text box

replace ' ' with '_' in file name

if user ID is entered, extend path & create directory if deosn't already exist

Homework submission program

Joel van Brandwijk has extended this program to create a generic homework submission utility

- student interface is password driven
  connects with Active Directory to verify password for given user name
- student can select among current classes and assignments
  course/assignment info is stored in an SQL database, accessed via PHP
- student can specify multiple files, even structure into subdirectories for uploading

- instructor interface is also password driven
  checks database to make sure a legit instructor, then checks password
- instructor can add/remove class, add/remove students, add/remove assignments
  info is stored in SQL database, accessed via PHP
- instructor can download submissions to local computer for grading

empirical.cs.creighton.edu/submissions

Example: email redirection

most Web sites give visitor option of sending feedback

- can be accomplished using `mailto:` link

```
<a href=mailto:davereed@creighton.edu>Feedback?</a>
```

potential problem: not all client computers are email-enabled

- can instead use a PHP program as a front-end for email
- user enters address, subject, message
- PHP program receives & bundles these pieces into an email message, sends on

- if desired, can be used for anonymous email redirection (no return address supplied)

Example: email redirection

```
<?php function show_form($email="",$message="",$subject="") {  ?>

<h2>Send Me an E-mail</h2>

<form action="http://empirical.cs.creighton.edu/~davereed/email.php"
      method="post">

Your E-mail address:<br>
<input type=text name=email size=30 value="<?php print($email); ?>"><br>

The Subject:<br>
<input type=text name=subject size=30 value="<?php print($subject); ?>"><br>

Your Message:<br>
<textarea rows=10 cols=50 name=message><?php print($message); ?></textarea><br>
<input type=submit value="Send E-mail">
</form>

<?php }

.
.
.
```

this part of the page defines a function for generating the input page

Example: email redirection (cont.)

```
$email = $_POST['email'];
$subject = $_POST['subject'];
$message = $_POST['message'];

if (!isset($email) or !isset($message)) {
     show_form();
}
else {
  if (empty($message)) {
      print("Please write something and resend.");
      show_form($email,$message,$subject);
  }
  else {
     if (empty($email)) {
         $email="anonymous";
     }
     if (empty($subject)) {
         $subject="stuff";
     }

     $sent = mail( "davereed@creighton.edu", $subject, $message, "From: $email" );

     if ($sent) {
        print("<H1>Your Message Has Been Sent.</H1>");
     }
     else {
        print("<p>The server was unable to send your mail.");
     }
  }
}
?>
```

if no inputs to page (not called from form), show input page

if message field is empty, warn user and show the input page again

if email or subject fields are empty, give default values

send the email message and report status