Lecture Notes 10
Introduction to PHP

Anand Tripathi

CSci 4131

Internet Programming

Topics

- Introduction to PHP
  - See links on class reference page
  - http://www.php.net/
  - See example from Chapter 19 (Deitel 5th edition)
- Server-side Includes

Introduction to PHP

- Variable are named starting with the $ symbol.
- Data types
  - int, integer
  - double, float
  - boolean
  - PHP supports dynamic typing of variables
  - Data type conversion is done explicitly using settype or through type casting.
    - You can use gettype to find the type of variable
  - It supports arrays that are accessed as key-value pairs
- PHP5 supports objects-oriented programming

Example 1

See this program

This code is strored in a file with extension .php

```
<html>
<body>
  <h2>  Example 1 <h2>
  <?php
  // php comment
    echo  "<h3> Hello World!  <br/>  This is my first PHP code. </h3>"
     ?>
</body>
</html>
```

Integer and Double

- Integer type corresponds to the long type in C.
- Double corresponds to the double type in C.
  - A double type literal can include a decimal, an exponent, or both. Exponent is specified using “E” or “e”.

Some useful functions

- floor defined for double
- ceil defined for double
- round defined for double (returns nearest integer)
- srand takes integer parameter
  - It initializes random number stream generator.
- rand has two integer arguments
  - It returns a random number greater than the first and smaller than the second argument
- min and max have one or more arguments

See data.php example from Chapter 19

Click here to see this example

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>PHP data types</title>
   </head>
   <body>
      <?php
         // declare a string, double and integer
         $testString = "3.5 seconds";
         $testDouble = 79.2;
         $testInteger = 12;
      ?>
```

```
<!-- print each variables value -->
  <?php
      print( "<p>  $testString is " . gettype ($testString) . "</p>" );
      print( "<p>  $testDouble is " . gettype ($testng). "</p>" );
      print( "<p>  $testInteger is " . gettype ($testInteger) . "</p>" );
  ?>
<br />
     Now, converting to other types:<br />
     <?php
       // call function settype to convert variable
       // testString to different data types
       print( "$testString" );
       settype( $testString, "double" );
       print( " as a double is $testString <br />" );
       print( "$testString" );
       settype( $testString, "integer" );
       print( " as an integer is $testString <br />" );
```

```
        settype( $testString, "string" );
        print( "Converting back to a string results in
           $testString <br /><br />" );

        $data = "98.6 degrees";

        // use type casting to cast variables to a
        // different type
        print( "Now using type casting instead: <br />
       As a string - " . (string) $data .
       "<br />As a double - " . (double) $data .
       "<br />As an integer - " . (integer) $data );
     ?>
   </body>
</html>
```

Output:

3.5 seconds is a string.
79.2 is a double.
12 is an integer.

Now, converting to other types:
3.5 seconds as a double is 3.5
3.5 as an integer is 3
Converting back to a string results in 3

Now using type casting instead:
As a string - 98.6 degrees
As a double - 98.6
As an integer - 98

Strings

- Single quoted and double quieted strings as in Perl.
- Single quoted strings do not interpret $variables and special escape sequences as \n.
- Double quoted strings string substitute $variables with their values, and special escape character sequence have

String Operations

- strlen
- strcmp to compare two strings
- strpos takes two string arguments
  - Check if the second string is present in the first:
    - Returns false if not present
    - Otherwise, position in the first string where the second string's first character appears
- chop removes all trailing whitespaces from the end of the given string
- trim removes whitespaces from both ends
- ltrim removes all whitespaces from the beginning
- strtolower strtoupper

Output functions

- print
  - Simple unformatted string
- echo performs a similar function
- printf is used for generating formatted output
- print_r for printing array elements

- nl2br takes one string argument and returns a string such that all newline characters are preceded by HTML newline

Relational and Boolean Operations

- Relational operators == != <= < > >= have their usual meaning.

- Boolean operattors

  and or xor (lower priority)

  && || (higher priority)

Strings Operations

- explode - split string, similar to split in Perl
- explode(separator,string,limit)

```
<?php
  $str = “Thank you very much.";
  print_r (explode(" ",$str));
  ?>
```

Output:

```
Array
    (
    [0] => Thank
    [1] => you
    [2] => very
    [3] =>much
    )
```

- implode function joins the elements of an array

String comparison

Click here to see this example Figure 19.8

```
// create array fruits
$fruits = array( "apple", "orange", "banana" );

// iterate through each array element
for ( $i = 0; $i < count( $fruits ); $i++ ) {

   // call function strcmp to compare the array element
   // to string "banana"
   if ( strcmp( $fruits[ $i ], "banana" ) < 0 )
      print( $fruits[ $i ]." is less than banana " );
   elseif ( strcmp( $fruits[ $i ], "banana" ) > 0 )
      print( $fruits[ $i ]." is greater than banana " );
   else
      print( $fruits[ $i ]." is equal to banana " );
```

String comparison

```
// use relational operators to compare each element
 // to string "apple"
 if ( $fruits[ $i ] < "apple" )
      print( "and less than apple! <br />" );
 elseif ( $fruits[ $i ] > "apple" )
      print( "and greater than apple! <br />" );
 elseif ( $fruits[ $i ] == "apple" )
      print( "and equal to apple! <br />" );

}
```

apple is less than banana and equal to apple!

orange is greater than banana and greater than apple!

banana is equal to banana and greater than apple!

Including files and Inline Data

- Include a file:

```
<?php
  include("xmlheader.txt");
 ?>
```

- Inline data (also called “heredoc” , starts with <<<, create double-quoted strings)

```
<?php  // HTML or XML  safe containers
  $htmlOutput = <<<HTMLOUTPUT
 <?xml version="1.0"?>
 <html>
     <body>
    <p>...all sorts goes here...</p>
   </body>
 </html>
 HTMLOUTPUT;

 echo $htmlOutput;
 ?>
```

Example of including files

See Example/PHP/include.php click here

Include construct is used to include common contents in multiple files. For example inclusion of standard display items, style sheets, or menus.

The contents of the specified file become part of the generated HTML document.

```
<html>
<body>
<?php
  include ("banner.txt");
  print "<h3> This is a test page to show the use of includes in PHP.
    The top two lines are from banner.txt file and the line below is from footer.txt.
    </h3>";
  include ("footer.txt");
?>
</body>
</html>
```

Contents of banner.txt and footer.txt

banner.text

```
<font  color="blue">
<h1> CSCI 4131:  Internet Programming </h1>
<h2> Spring 2012 </h2>
</font>
```

footer.txt

```
<font  color ="red" size="2ps">
<h4> Please contact Anand Tripathi for any questions. </h4>
</font>
```

Arrays

Array stores a list of key-value pairs of items.

If key not specified, then integer valued keys, starting with 0 are used.

```
<?php
    $a=array("a"=>“Apple","b"=>“Banana","c"=>“Cherry");
    print_r($a);
?

<?php
    $a=array(“Apple", “Banana", “Cherry");
    print_r($a);
?>
>
```

Array Operations

- There are close to 50 operations defined for arrays. See php.net site.
- array_push, array_pop similar to Python functions append and pop
  - Add remove items at the end of the array
- array_unshift, array_shift

  Add remove items at the beginning of the array.
- array_keys
  - Returns a list (i.e. an array with integer indices) of all keys
- array_values
  - Returns a list (i.e. an array) of all values
- asort
- arsort (reverse sort)
- array_reverse

Arrays and List

- list () defines a list of scalar variables
- Example from php.net

```
$info = array('coffee', 'brown', 'caffeine');
// Listing all the variables
list($drink, $color, $power) = $info;
echo "$drink is $color and $power makes it special.\n";
```

Array iteration

- Two ways to do iteration:

$fruits = array(“Apple”, “0range”, “Banana”);

1. for ($i = 0; $i < count($fruits); $i++)
   print("Element $i is $fruits[$i]
“

One can also use sizeof($fruits) to get the size of the array.

2. foreach ($fruits as $item) {
   echo “$item”;
   }

Arrays - example arrays.php

Click here to see this example execution This is example 23.6 in Deitel's book.

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Array manipulation</title>
   </head>
   <body>
      <?php
         // create array first
         print( "<strong>Creating the first array</strong>
            <br />" );
         $first[ 0 ] = "zero";
         $first[ 1 ] = "one";
         $first[ 2 ] = "two";
         $first[] = "three";
```

```
// print each elements index and value
  for ( $i = 0; $i < count( $first ); $i++ )
     print( "Element $i is $first[$i] <br />" );
  print( "<br /><strong>Creating the second array
     </strong><br />" );

  // call function array to create array second
  $second = array( "zero", "one", "two", "three" );
  for ( $i = 0; $i < count( $second ); $i++ )
     print( "Element $i is $second[$i] <br />" );

  print( "<br /><strong>Creating the third array
     </strong><br />" );

  // assign values to non-numerical indices
  $third[ “Amy" ] = 21;
  $third[ “Bob" ] = 18;
  $third[ “Carol" ] = 23;
```

```
      // iterate through the array elements and print each
      // elements name and value
      for ( reset( $third ); $element = key( $third );  next( $third ) )
         print( "$element is $third[$element] <br />" );
      print( "<br /><strong>Creating the fourth array
         </strong><br />" );
      // call function array to create array fourth using
      // string indices
      $fourth = array(
         "January"   => "first",   "February" => "second",
         "March"     => "third",   "April"    => "fourth",
         "May"       => "fifth",   "June"     => "sixth",
         "July"      => "seventh", "August"   => "eighth",
         "September" => "ninth",   "October"  => "tenth",
         "November"  => "eleventh","December" => "twelfth"
         );
      // print each elements name and value
      foreach ( $fourth as $element => $value )
         print( "$element is the $value month <br />" )
   ?>
  </body>
</html>
```

```
 Creating the first array
Element 0 is zero
Element 1 is one
Element 2 is two
Element 3 is three

Creating the second array
Element 0 is zero
Element 1 is one
Element 2 is two
Element 3 is three

Creating the third array
Amy is 21
Bob is 18
Carol is 23

Creating the fourth array
January is the first month
February is the second month
March is the third month
April is the fourth month
May is the fifth month
June is the sixth month
July is the seventh month
August is the eighth month
September is the ninth month
October is the tenth month
November is the eleventh month
December is the twelfth month
```

File Operations

fopen to open a file

$handle = fopen(“filepathstring”, “mode”);

$handle = fopen (“/home/testinput”, “r”);

Modes:

‘r’ open for reading

‘r+’ open for reading and writing

‘w’ open for writing, pointer is at the beginning of the file, truncate length to 0

‘w+’ As above, but creates file if it does not exist

‘a’ open for append

‘x’ Create and open for writing only. If it exists, fopen returns false

See manual pages for other options such as ‘x+’, “c’. ‘c+’

See fclose

File Operations

- **fgets** reads from an open file, from the open file handle
- string **fgets** (resource $handle [, int $length])
  - Option argument indicates max number of characters or till new line is encountered.
- string **fread** (resource $handle , int $length)
  - Reads up to specified number of bytes
- int **fwrite** (resource $handle , string $string [, int $length])
  - Write the specified number of bytes or till the end of the string is reached.
- bool **flock** (resource $handle , int $operation [, int &$wouldblock])

  $operation can be LOCK_SH (reader lock, shared), LOCK_EX (exclusive lock, write), LOCK_UN (release lock)
- function **file** (filepath) open the specified file by path for reading, and returns an array containing the lines.

Example for flock

See PHP documentation page

This example shows how to get an exclusive lock on a file for writing.

```
<?php
$fp = fopen("/tmp/lock.txt", "r+");

if (flock($fp, LOCK_EX)) {  // acquire an exclusive lock
    ftruncate($fp, 0);            // truncate file
    fwrite($fp, "Write something here\n");
    fflush($fp);                  // flush output before releasing the lock
    flock($fp, LOCK_UN);    // release the lock
} else {
    echo "Couldn't get the lock!";
}
fclose($fp);
?>
```

Example

See Example/PHP/fileread.php

Click here to execute this example

```
<?php
// set file to read
    $file = 'fruits.txt' ;
// read file into array
    $data = file($file) or die('Could not read file!');
// loop through array and print each line
   foreach ($data as $line) {
         echo $line;
}
?>
```

PHP functions

- A function can return a scalar or an array.
- Parameters can be passed by value or reference.
  - Comma separated parameter list
- By default function arguments are passed by value.
- To pass parameter by reference, prepend ampersdan (&) to the argument
  - E.g. &$param (C-like notation)

PHP functions

- Default arguments:

```
function sayHi ( $param = “World” ) {
    return “Hello $param”
}
echo sayHi( “Sam” );
echo sayHi( );
```

- PHP versions higher than 5.1 also allow function definition to specify required type of parameter such as int, bool, float, string, array.
  - See php.net for more details

PHP Objects

Example from php.net Click here to see this execution

```
<?php
class A {
   public $foo = 1;
}
$a = new A;
$b = $a;     // $a and $b are copies of the same identifier
          // ($a) = ($b) = <id>
$b->foo = 2;
echo $a->foo."\n";

$c = new A;
$d = &$c;    // $c and $d are references
          // ($c,$d) = <id>
$d->foo = 2;
echo $c->foo."\n";
$e = new A;
function foo($obj) {
   // ($obj) = ($e) = <id>
   $obj->foo = 2;
}
foo($e);
echo $e->foo."\n";
?>
```

Parsing JSON data

- Parse JSON data to get either PHP object-based or an associate array-based representation of data.

```
<?php
$json = '{"employees": [
    {"Name":"John","tel":"111-3333"},
    {"Name":"Anna","tel":"111-222-3333"},
    {"Name":"Peter","tel":"111-222-3333"}
    ]  }';
```

Parsing JSON data

```
echo "JSON decoded as objects <br/>";
var_dump( json_decode( $json ) );
echo "JSON decoded as Associatve Array <br/>";
var_dump( json_decode( $json, true ) );
?>
```

See this example

Fetching data from web and parsing

```
$json = file_get_contents(
            "http://www-users.cs.umn.edu/~tripathi/Internet-Examples/json/exmpl.json");
```

- exmpl.json contains the following data:

{"employees":[{"Name":"John","tel":"111-222-3333"},{"Name":"Anna","tel":"111-222-3333"},{"Name":"Peter","tel":"111-222-3333"}]}

- See this example

Accessing JSON data fields as associative array decoding

```
<?php
        $json = file_get_contents(
          "http://www-users.cs.umn.edu/~tripathi/Internet-Examples/json/exmpl.json");
        $json_data = json_decode($json, true);
        echo "<br/>";
        $result = $json_data[employees];

        print count($result);

        for ($i =0; $i < count($result); $i++) {
          echo "<br/>";
          print_r ($result[$i] );
        }
 ?>
```

See this example

Accessing JSON data fields as object-based decoding

```
<?php
        $json= '{"employees":[{"Name":"John","tel":"111-222-3333"},{"Name":"Anna","tel":"111-222-3333"},{"Name":"Peter","tel":"111-222-3333"}]}';
        $json_data = json_decode($json );
        $result = $json_data->employees;
        for ($i =0; $i < count($result); $i++) {
            echo "<br/>";
            print_r ($result[$i] );
            echo "<br/>";
            print ($result[$i]->Name);
        }
?>
```

See this example

PHP with JavaScript

- You can emebd PHP code within script tag to generate contents of JavaScript code.

```
<head>
   <title>A simple PHP document</title>
   <script>
     var  name =  <?php  echo “Hello" ?>
   </script>
</head>
```

Regular Expressions

See Section 19.7 of Deitels book

We will be learning Perl regular expressions.

Regular expressions for searching for a pattern of characters in a given string

Perl Regular Expressions

A pattern can be specified containing the following:

C Non-special character C
\C Turn-off special meaning of C
. Any single character
^ Beginning of string
$ End of string
[…] Any one of the characters enclosed in the square brackets
[a-zA-Z]
[^…] Not any character in the given set of characters
[^a-zA-Z]
C* Zero or more occurrences of C
C+ One or more occurrences of C
C? Zero or one occurrence of C
C{m,n} At least m and no more than n occurrences of C
PQ Pattern P followed by Q

Patterns are constructed from repetitions of character sequences.

* zero or more

+ one or more

? zero or one

c{n,m} means at least n and at most m occurrences of c

. matches any character other than newline \n.

[aXcd] matches any character that is a, X, c, d.

[^aXcd] does not matches any character that is a, X, c, d.

Examples of regular expressions

/p*/

/pq*/

/^[eE]xample/

/[a-zA-Z]/

/[A-Z][a-zA-Z]*/

Examples of regular expressions

p*

Matches empty sequence, p, pp, ppp, pppp, ...

pq*

Matches p, pq, pqq, pqqq, pqqqq, …

‘^[eE]xample’

Beginning of line, string example or Example

[a-zA-Z]

Matches any lower case or upper case letter, including empty string

[a-zA-Z]+

At least one or more letters

\d* zero or more digit characters

Anchors and Patterns

Some useful "anchor" and patterns:

- ^ means beginning of the string being searched
- $ end of the string
- \s a space character, \n, \r, \t, \f
- \S non-space character
- \b word boundary
- \w word character [a-zA-Z0-9_]
- \W non-word character [^a-zA-Z0-9_]
- \d digit,
- \D non-digit
- \n newline (LF)
- \r return (CR)
- \t tab

Character Classes

- [:alnum:] represents alphanumeric characters [a-zA-Z] and [0-9]
- [:alpha:] represents letters [a-zA-Z]
- [:word:] Just like alnum but also includes _
- [:digit:] Digits
- [:space:] White space (spaces,tabs, and newline)
- [:lower:] Lowercase letters
- [:upper:] Uppercase letters

Examples of regular expressions

/a{1, 3} b/
matches ab, aab, aaab
/ab{3}c/
matches abbbc
/(cats){3} z{5}/
Matches catscatscats zzzzz
/0\d+/
Matches an octal digit

/\b”\w+”\b/
Matches a double-quoted word or one or more [A-Za-z]characters.

Examples of regular expressions

/\$?[:digit:]+\.[:digit:]{2}/

Another way to write it
/\$?\d+\.\d\d/

Pattern Modifier:
/pattern/i ignore case

Examples of regular expressions

/^ [ ]*[eE]xample/

/Chapter[0-2]/

/\bh.*d\b/

/^[^:]*::/

Examples of regular expressions

/^ [ ]*[eE]xample/

One or more spaces in the beginning of a string, followed by Example or example

/\bh.*d\b/

Will match a word such as handed, hounded, hampered.

/chap[0-2]/ will match chap0, chap1, but not chap3

/^[^:]*::/

At the beginning of a line, any sequence of characters not containing : , and ending with two colons.

- Matched patterns can be remembered in variables such as **$match array** if the pattern is specified in (..).

- For example: a(m+)i(n+) The string of one or more m's will be remembered in **$match[1]**, and similarly the string of one or more n's will be stored in **$match[2].**

- If you do not wish a parenthesized pattern to be memorized use (?:pattern).

PHP Regular Expressions

- Two functions for pattern matching
  - ereg POSIX compatible
    - **Deprecated in PHP 5.3**
  - eregi is for case insensitive matching

  - preg_match Perl compatible

- Click here to see example in Figure 19.9
  (This is in Section 19.7)

```
$search = "Now is the time";
 print( "Test string is: '$search'<br /><br />" );

 // call function ereg to search for pattern Now
 // in variable search
 if ( preg_match( “/Now/”, $search ) )
    print( "String 'Now' was found.<br />" );

 // search for pattern Now in the beginning of
 // the string
 if ( preg_match( “/^Now/”, $search ) )
    print( "String 'Now' found at beginning
       of the line.<br />" );

 // search for pattern Now at the end of the string
 if ( preg_match( “/Now$/”, $search ) )
    print( "String 'Now' was found at the end
```

```
// search for any word ending in ow
if ( preg_match( “/\b([a-zA-Z]*ow)\b/i", $search,   $match ) )
   print( "Word found ending in 'ow': " .  $match[ 1 ] . "<br />" );

// search for any words beginning with t
print( "Words beginning with 't' found: ");

while ( preg_match(“/\b(t[[:alpha:]]+)\b/i", $search, $match ) ) {
   print( $match[ 1 ] . “ " );

  // remove the first occurrence of a word beginning
  // with t to find other instances in the string
  $search = preg_replace( “/”.$match[ 1 ].”/”, "", $search );
}
print( "<br />" );
```

b

Test string is: 'Now is the time'

String 'Now' was found.
String 'Now' found at beginning of the line.
Word found ending in 'ow': Now
Words beginning with 't' found: the time

Environment Variables PHP and Form Processing in PHP

- PHP stores environment variables and their values in $_ENV
- $_SERVER contains data about server
- $_GET contains data sent to server using HTTP GET
- $_POST contains data sent to server using HTTP POST
- $_COOKIES Data send as cookies by the client
- $GLOBAL array of global variables

Environemnt Variables PHP

- Example of $_SERVER

```
<form action="<?php echo $_SERVER['PHP_SELF']; ?> method="POST">
```

- Extract function on $_POST creates for each form variable, a scalar starting with $varName, which can be used to access the value submitted through form submission.
- See example env.php

env.php

```
<table border = "0" cellpadding = "2" cellspacing = "0"
      width = "100%">
   <?php
      // print the key and value for each element
      // in the $_SERVER array
      foreach ( $_SERVER as $key => $value )
         print( "<tr><td bgcolor = \"#11bbff\">
            <strong>$key</strong></td>
            <td>$value</td></tr>" );
   ?>
   </table>
```

env.php

In PHP, you must do getenv(“Environment Var Name”) to explicitly add it the $_ENV

For example:

```
getenv("REMOTE_ADDR" );
getenv("USER" );
getenv("HTTP_HOST" );
getenv("REQUEST_URI" );
```

Form Processing in PHP

- Extract function on $_POST creates for each form variable, a scalar starting with $varName, which can be used to access the value submitted through form submission.

- For processing example from Chapter 23

  - form.html which POSTs data to form.php
  - Click here to see form.html execution

form.html

```
<form method = "post" action = "form.php">
        <img src = "images/user.gif" alt = "User" /><br />
        <span style = "color: blue">
          Please fill out the fields below.<br />
        </span>
        <!-- create four text boxes for user input -->
        <img src = "images/fname.gif" alt = "First Name" />
        <input type = "text" name = "fname" /><br />

        <img src = "images/lname.gif" alt = "Last Name" />
        <input type = "text" name = "lname" /><br />
        <img src = "images/email.gif" alt = "Email" />
        <input type = "text" name = "email" /><br />

        <img src = "images/phone.gif" alt = "Phone" />
        <input type = "text" name = "phone" /><br />
        <span style = "font-size: 10pt">
          Must be in the form (555)555-5555</span>
        <br /><br />
```

```
<img src = "images/downloads.gif"
   alt = "Publications" /> <br />
<span style = "color: blue">
   Which book would you like information about?
</span><br />
<!-- create drop-down list containing book names -->
<select name = "book">
   <option>Internet and WWW How to Program 3e</option>
   <option>C++ How to Program 4e</option>
   <option>Java How to Program 5e</option>
   <option>XML How to Program 1e</option>
</select>

<br /><br />
<img src = "images/os.gif" alt = "Operating System" />
<br /><span style = "color: blue">
   Which operating system are you currently using?
<br /></span>
```

```
<!-- create five radio buttons -->
      <input type = "radio" name = "os" value = "Windows XP"
        checked = "checked" />
        Windows XP

      <input type = "radio" name = "os" value =
        "Windows 2000" />
        Windows 2000
     <input type = "radio" name = "os" value =
        "Windows 98" />
        Windows 98<br />
      <input type = "radio" name = "os" value = "Linux" />
        Linux
      <input type = "radio" name = "os" value = "Other" />
        Other<br />

       <!-- create a submit button -->
      <input type = "submit" value = "Register" />
   </form>
```

form.php

```
<?php
   extract($_POST);   // Extract all submitted values by the
                      // variables' names
   // determine whether phone number is valid and print
   // an error message if not
   if ( !ereg( "^\([0-9]{3}\)[0-9]{3}-[0-9]{4}$",
      $phone ) ){

      print( "<p><span style = \"color: red;
         font-size: 2em\">
         INVALID PHONE NUMBER:</span><br />
         A valid phone number must be in the form
         <strong>(555)555-5555</strong><br />
         <span style = \"color: blue\">
         Click the Back button, enter a valid phone
         number and resubmit.<br /><br />
         Thank You.</span></p></body></html>" );
      die(); // terminate script execution
   }
?>
```

form.php

```
<p>Hi
      <span style = "color: blue">
        <strong>
          <?php print( "$fname" ); ?>
        </strong>
      </span>.
      Thank you for completing the survey.<br />

      You have been added to the
      <span style = "color: blue">
        <strong>
          <?php print( "$book " ); ?>
        </strong>
      </span>
      mailing list.
    </p>
```

form.php

```
<strong>The following information has been saved
      in our database:</strong><br />
  <table border = "0" cellpadding = "0" cellspacing = "10">
    <tr>
      <td bgcolor = "#ffffaa">Name </td>
      <td bgcolor = "#ffffbb">Email</td>
      <td bgcolor = "#ffffcc">Phone</td>
      <td bgcolor = "#ffffdd">OS</td>
    </tr>
    <tr>
      <?php
        // print each form fields value
        print( "<td>$fname $lname</td>
          <td>$email</td>
          <td>$phone</td>
          <td>$os</td>" );
      ?>
    </tr>
  </table>
```

How to make an HTTP request in PHP?

- Use -- file_get_contnets(URL)
- See this Example

```
<html>
<body>
  <?php
    $data = file_get_contents("http://www.cnn.com");
    print $data;
  ?>
</body>
</html>
```

Generation of Cookies

- This example is from Deitel's Book (4th edition – available online)
- See Section 23.6
- Three programs:
  - cookies.html displays a from with three fields
    - It is submitted to cookies.php
  - cookies.php
    - When form is submitted, three cookies are generated by the PHP program and sent to the client
  - readCookies.php
    - Displays the cookies submitted with the request

setcookie function

- PHP code on the server side can execute setcookie function to send cookies to the client (browser)
- setcookie(name, value, expiration-time, dirpath)
  - name - required
  - value – optional
  - expiration time - in seconds (optional)
    - Specifed as time() + number-of-seconds
    - If not specified or set to 0, then the cookie will expire at the end of the session when the browser closes
  - dirpath – optional
    - If not specified then it is for the current directory

“define” function

- Using define, one can define a constant in a PHP program.

- Example:

```
define( “FIVE_DAYS”,  60*60*24*5)
setcookie( "Name", “Sam”, time() + FIVE_DAYS);
```

cookies.html

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <body>
     <form method = "post" action = "cookies.php“>
        <input type = "text" name = "NAME" /><br />
         <input type = "text" name = "HEIGHT" /><br />
         <input type = "text" name = "COLOR" /><br />
         <input type = "submit" value = "Write Cookie“ />
      </form>
   </body>
</html>
```

cookies.php

```
<?php
  extract( $_POST );
  // write each form field'ss value to a cookie and set the cookie's expiration date
  setcookie( "Name", $NAME, time() + 60 * 60 * 24 * 5 );
  setcookie( "Height", $HEIGHT, time() + 60 * 60 * 24 * 5 );
  setcookie( "Color", $COLOR, time() + 60 * 60 * 24 * 5 );
?>
<html>
<head>   ....    </head>
<body>
     <p>The cookie has been set with the following data:</p>
     <br /><span style = "color: blue">Name:</span> <?php print( $NAME ) ?><br />
     <span style = "color: blue">Height:</span>  <?php print( $HEIGHT ) ?> <br />
     <span style = "color: blue">Favorite Color:</span>
     <span style = "color: <?php print( "$COLOR\">$COLOR" ) ?>  </span><br />

    <p>Click <a href = "readCookies.php">here</a>  to read the saved cookie.</p>
</body>  </html>
```

readCookies.php

Figure 26.24 from Deitels book

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head><title>Read Cookies</title></head>
   <body style = "font-family: arial, sans-serif">
      <p> <strong> The following data is saved in a cookie on your computer. </strong> </p>
      <table border = "5" cellspacing = "0" cellpadding = "10">
         <?php
            // iterate through array $_COOKIE and print
            // name and value of each cookie
            foreach (   $_COOKIE     as      $key => $value    )
               print( "<tr>
                  <td bgcolor=\"#F0E68C\">  $key    </td>
                  <td bgcolor=\"#FFA500\">   $value   </td>
                  </tr>" );
         ?>
      </table>
 </body> </html>
```

File Upload using PHP

- Uploaded files are available in an array variable $_FILES
- Suppose that a file is uploaded with form element as below:

<input type=file name="uploadedFile" …..>

On the server side, this file can be accessed using its temporary name with which it would be stored on the server:

$_FILES['uploadedFile']['tmp_name']

File's name on the client side can be obtained as:

$_FILES['uploadedFile']['name']

File Upload using PHP

Other attributes can be accessed as follows:

Size of the uploaded file:

$_FILES['uploadedFile']['size']

Type of the uploaded file:

$_FILES['uploadedFile']['type']

Form to upload a file – PHP example

```
<html>
<head>
</head>
<body>
<form enctype="multipart/form-data" action="uploader.php"
    method="POST">
  Choose a file to upload: <input name="uploadedFile" type="file" /><br />
<input type="submit" value="Upload File" />
</form>
</body>
```

PHP program - uploader.php

```
<html>
<head>   ….  </head>
<body>
<?php
   $target_path = "Uploads/"; /* Stores files in this directory */
   /* Add the original filename to our target path.   */
   $target_path = $target_path.basename( $_FILES['uploadedfile']['name']);

   if (move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
         echo  "The file ".  basename( $_FILES['uploadedFile']['name'])." has been
     uploaded";
   } else{
   echo "There was an error uploading the file, please try again!";
}
?>
</body>  </html>
```

Example of Connecting to Database from PHP

Click here to see this example.

```
<!– File  data.html     -->
<!-- Querying a MySQL Database -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Sample Database Query</title>
   </head>

   <body style = "background-color: #F0E68C">
      <h2 style = "font-family: arial color: blue">
         Querying a MySQL database.
      </h2>
```

```
<form method = "post" action = "database1.php">
        <p>Select a field to display:

          <!-- add a select box containing options -->
          <!-- for SELECT query                    -->
          <select name = “tableSelection">
            <option selected = "selected">*</option>
            <option>ID</option>
            <option>Title</option>
            <option>Category</option>
            <option>ISBN</option>
          </select>
        </p>

        <input type = "submit" value = "Send Query"
          style = "background-color: blue;
          color: yellow; font-weight: bold" />
      </form>
   </body>
</html>
```

PART of database1.php file:

```
<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Search Results</title>
   </head>

   <body style = "font-family: arial, sans-serif"
      style = "background-color: #F0E68C">

    <?php
        extract( $_POST );

        // build SELECT query
       $query = "SELECT * from $tableSelection";
```

```
// Connect to MySQL
• if ( !( $database = mysql_connect( “db.cselabs.umn.edu:3307",
        “C4131S12U1", "password" ) ) )`
        die( "Could not connect to database" );

    // open Products database
    if ( !mysql_select_db( “C4131S14U1", $database ) )
        die( "Could not open 4131Examples database" );

    // query Products database
    if ( !( $result = mysql_query( $query, $database ) ) ) {
        print( "Could not execute query! <br />" );
        die( mysql_error() );
    }
?>
<h3 style = "color: blue">
Search Results</h3>
```

```
<table border = "1" cellpadding = "3" cellspacing = "2"
        style = "background-color: #ADD8E6">
        <?php
          // fetch each record in result set
          for ( $counter = 0;  $row = mysql_fetch_row( $result ); $counter++)
          {
             // print a table row to display results
             print( "<tr>" );
                 foreach ( $row as $key => $value )
                             print( "<td>$value</td>" );
             print( "</tr>" );
          }
          mysql_close( $database );
        ?>
 </table>
   <br />Your search yielded <strong>
      <?php print( "$counter" ) ?> results.<br /><br /></strong>
</body> </html>
```

Server Side Includes

Server Side Includes

- These are commands embedded in an HTML document.
- These commands are directive for the server to dynamically generate parts of the contents of the HTML document.
  - Include date
  - Include some document
  - Include output of some command execution
    - Some systems admin disallow this for security
  - Generate conditional actions to generate document contents

Server Side Includes

Commands that can be specified:

- exec to execute a system command or program
- include to include a file
- set declare and set a variable
- echo to echo environment variables
- if – then-else for conditional operations

These are put in the HTML document as comments:

```
<!--#include file="banner-page.html" -->
<!--#echo var="REMOTE_ADDR" -->
```

Echo variables

See this example program

```
<HTML>
<HEAD>
<TITLE>S Echo Example- Server Side Includes </TITLE>
</HEAD>
<BODY>
<H1>Just a simple example Generating SSI!!!</H1>
<B>
   <h3 style = "text-align: center">
       Using Server-Side Includes -  Examples of echo var
    </h3>
   Time in  Minneapolis is:
     <span style = "color: blue">
      <!--#echo var="DATE_LOCAL" -->
    </span><br />
```

Echo variables

```
The Greenwich Mean Time is
 <span style = "color: blue">
    <!--#ECHO VAR="DATE_GMT" -->.
 </span><br />

 The name of this document is
 <span style = "color: blue">
    <!--#ECHO VAR="DOCUMENT_NAME" -->.
 </span><br />

  This  document was last modified on
 <span style = "color: blue">
    <!--#ECHO VAR="LAST_MODIFIED" -->.
 </span><br />
```

Echo variables

```
Your current IP Address is
    <span style = "color: blue">
      <!--#ECHO VAR="REMOTE_ADDR" -->.
    </span><br />

Your client HOST is
    <span style = "color: blue">
      <!--#ECHO VAR="REMOTE_HOST" -->.
    </span><br />

    My server name is
    <span style = "color: blue">
      <!--#ECHO VAR="SERVER_NAME" -->.
    </span><br />
```

Echo variables

```
And I am using the
  <span style = "color: blue">
    <!--#ECHO VAR="SERVER_SOFTWARE" -->
    Web Server.
  </span><br />
  You are using
  <span style = "color: blue">
    <!--#ECHO VAR="HTTP_USER_AGENT" -->.
  </span><br />
This server is using
  <span style = "color: blue">
    <!--#ECHO VAR="GATEWAY_INTERFACE" -->.
  </span><br />
This document was last modified on
    <!--#ECHO VAR="LAST_MODIFIED" -->.
```

Declare and set a variable

<--!#set var=“name” value=“sam” -->

<--!--#set var=“age” value=“10” -->

<--!--#set var=“datetime” value=“$DATE_GMT” -->

Just a simple example Generating SSI!!!

Using Server-Side Includes - Examples of echo var

Time in Minneapolis is: Monday, 21-Feb-2005 11:26:35 CST

The Greenwich Mean Time is Monday, 21-Feb-2005 17:26:35 GMT.
The name of this document is echo.shtml.
This document was last modified on Monday, 21-Feb-2005 10:21:30 CST.
Your current IP Address is 128.101.35.12.
Your client HOST is (none).
My server name is www-users.cs.umn.edu.
And I am using the Apache/2.0.52 (Unix) mod_ssl/2.0.52 OpenSSL/0.9.7d PHP/4.3.10 Web Server.
You are using Mozilla/5.0 (Windows; U; Windows NT 5.0; rv:1.7.3) Gecko/20041001 Firefox/0.10.1.
This server is using CGI/1.1.

Just a simple example Generating SSI!!!

Using Server-Side Includes - Examples of echo var

Time in Minneapolis is: Monday, 21-Feb-2005 11:26:35 CST

The Greenwich Mean Time is Monday, 21-Feb-2005 17:26:35 GMT.

The name of this document is echo.shtml.

This document was last modified on Monday, 21-Feb-2005 10:21:30 CST.

Your current IP Address is 128.101.35.12.

Your client HOST is (none).

My server name is www-users.cs.umn.edu.

And I am using the Apache/2.0.52 (Unix) mod_ssl/2.0.52

OpenSSL/0.9.7d PHP/4.3.10 Web Server.

You are using Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0).

This server is using CGI/1.1.

includes

To see this example working – click here

```
<HTML>
<HEAD>
<TITLE>Server Side Includes </TITLE>
</HEAD>
<BODY>
<H1>Just a simple example Generating SSI!!!</H1>
Including file include-file1.shtml: <br>
    <!--#include file =  "include-file1.shtml" -->
Including file include-file2.shtml: <br>
    <!--#include file =  "include-file2.shtml" -->
</BODY>
</HTML>
```

Include files in the example

Include-file1.shtml

```
The Greenwich Mean Time is
    <span style = "color: blue">
      <!--#ECHO VAR="DATE_GMT" -->
    </span><br />
```

Include-file2.shtml

```
Time in  Minneapolis is:
    <span style = "color: blue">
      <!--#echo var="DATE_LOCAL" -->
    </span><br />
```

Conditional generation of contents

Click here to see this example working

```
<head>
<head>
<title>Server Side Includes </title>
</head>
<body>
<h1>Just a simple example Generating SSI!!!</H1>
<h3> This is an example of conditional operations. </H3>
<br>
 <!--#set  var="remote" value="$REMOTE_ADDR" -->
    Remote IP Address is
      <!--#ECHO VAR="remote" -->.

    Your current IP Address is
    <span style = "color: blue">
      <!--#ECHO VAR="REMOTE_ADDR" -->.
    </span><br>
```

Conditional generation of contents

```
 Conditional example<br>
<!--#if expr="$REMOTE_ADDR=/128.101.36.64/" -->
  Welkomme!!! <br>
  <!--#include file =  "include-file2.shtml" -->
<!--#else -->
  Sorry,  you do not have access to this page!  <br>
<!--#endif -->
<P>
These are general contents open to all. <br>
<!--#include file =  "include-file1.shtml" -->
<!--#if expr="$DATE_LOCAL=/Feb/"-->
  It is February <br>
<!--#else -->
  It is not February <br>
<!--#endif -->
</BODY>
</HTML>
```

Two included files

Include_file2

```
 Time in  Minneapolis is:
    <!--#config timefmt="%A %B %d, %Y" -->
      <span style = "color: blue">
        <!--#echo var="DATE_LOCAL" -->
      </span><br />
```

timefmt has several options:

%A	Full weekday
%B	Full month name
%d	day of month as digit
%Y	Four digit year