More PHP

CS174

Chris Pollett

Nov 27, 2006.

Outline

- More on PHP Arrays
- Functions
- Variable Scope
- Pattern Matching
- Form Handling
- File Handling
- Cookies
- Sessions

More on PHP Arrays

- Last Wednesday, we saw the basics of creating/accessing an array in PHP: $arr=array(1,2,3); echo $arr[2]; /*would print 3 */
- Recall also $carr = array(); //create an empty array
- Arrays in PHP are similar to Perl hashes.
- The above way to create $arr can also be written in PHP as:

  $arr = array(0=> 1, 1=>2, 2=>3);
- Like hashes we can do things like $arr = array("joe"=> 5, "mary" =>6);
- To get the keys and values we can use the functions: $keys = array_keys($arr) and $values = array_values($arr);
- Arrays can also be created by an assignment: $barr[1] = 5; // creates array $barr if doesn't exist
- If did the assignment $barr[] = 6; Then since the argument to [] wasn't specified PHP will assign $barr[2] = 6;

Yet More on PHP Arrays

- As arrays are like Perl hashes, you can call the unset function on the element in an array: $list= array(2,4,6,8); unset($list[2]);
- Some useful array functions: count -- returns the number of elements in an array, is_array, in_array, implode, explode, sort, rsort, asort, arsort, ksort, krsort.
- To see how implode/explode work consider:

  $str="this is a string";

  $words = explode(" ", $str); /*acts like split except here the first argument is a string rather than a regular expression. So words is an array("this", "is", "a", "string"). PHP has a split function but not as fast, since arg might be a regular expression. */

  $str2 = implode(" ", $words); //undoes the explode.

Iterating Through Arrays

- The function current can be used to return a pointer to the current element in an array. The next function can be used to advance this pointer and get its value:

```
$cities = array("San Jose", "San Diego");
echo current($cities); // prints San Jose
$another = next($cities); // $another is now San Diego;
```

- There are also the functions each, prev, end, and reset to facilitate moving through array.
- The function each is similar to next except after advancing the current pointer, it returns the old pointer as a two element array consisting of a key/value pair.
- We saw last day that one can iterate through arrays using foreach($arr as $val){…}
- PHP also supports code like

```
$lows = array("Mon" => 23, "Tue" => 18);
foreach($lows as $day =>$temp )
{echo "$day lows were $temp";}
```

Functions

- The general format of a PHP functions is:

  function *name*([*parameter*]){…}

  For example,

  function inc($i){return $i++;}
- A return value can be sent back using a return call as in many programming languages.
- You can modularize your code by putting several function definitions into a file and then use the include function to include them into any document that needs those functions.
- Parameters are passed by value. So the function call:

$b = inc($a); // leaves the value of $a unchanged

- You can call by reference by using an ampersand:

$b =inc(&$a); //here the value of $a is changed (one is added to it).

- You can also create functions with pass by reference parameters: function inc(&$i){…}

Variable Scope

- The default scope of a variable in PHP is only within the function that it is used. That is local scope:

```
$bob = 5;
function test()
{ $bob=6; echo $bob; //echo's 6}
test();
echo $bob; //echo's 5
```

- In order to access global variables within a local function one would need to use the global declaration:

```
 $bob =5;
function test()
{
   global $bob; # if did not do bob would be NULL
   echo $bob;
}
test();
```

- PHP also supports static local variables. These preserve states between function calls: function addone () {static $count =0; echo $count++;}

Pattern Matching

- PHP supports Perl style regular expression and POSIX regular expressions.
- For example,

$fruits = preg_split(“/:/”, “apples:oranges”);

//would act like Perl’s split

- preg_match acts like acts like Javascript match.

Form Handling

- As we saw last day in the test examples done on my machine, PHP makes available several important global variables which are useful for server side scripts.
- The phpinfo() function can be used to find out all of these globals.
- To process forms the most useful global variables are: $_GET and $_POST.
- For instance, $_GET[“bob”] returns the value of the form variable bob that was HTTP GET’d to the server.

File Handling

- Since PHP is a server side technology it is allowed to create, read, and write file on the server's filesystem.
- To open a file one can do:

```
$fileHandle = fopen("my.dat", "r");
// one can also open as r+, w+, a, a+
$file_string = fread($fileHandle, filesize("my.dat"));
fclose($fileHandle);
```

- Other file I/O functions are: file, fgets, fwrite

Cookies

- Cookies can be set using the setcookie function:

  setcookie(name, value, expires);
- This should be done before output is produced by your script.
- To access the value of a cookie returned from a browser you can use the $_COOKIES array.

Sessions

- Like Java servlets, PHP supports session management.
- To start a session one calls start_session();
- Then to set/get values of the session one uses the global array variable $_SESSION:

  $_SESSION[“test”]=37; /* sets the test session variable*/

  echo $_SESSION[“test”];