PHP Data Types and Functions

CS174

Chris Pollett

Sep 17, 2008.

Outline

- General Syntax of PHP
- Data Types and Operators
- Outputting to the browser stream
- Control Statements
- Arrays
- Functions

General Syntax

- Recall a section of PHP code is delimited with <?php … code here …?>.
- The command include("filename.php"); can be used to include one document in another.
- All variables in PHP begin with a $ sign. Names of variables are otherwise like in Perl or other common programming languages.
- PHP can use either Perl, C++, or C comments: #, //, /* … */ .

Primitives

- PHP has four scalar types: Boolean, integer, double, and string.
- There are also two compound types: array and object and two special types: resource and NULL.
- As with Javascript and Perl, PHP is dynamically typed: it has no type definitions.
- An unassigned variable is sometimes called an unbound variable and has value NULL. This can be coerced to one of the other types depending on the context.
- To check if a variable is currently bound you can use the IsSet($variable_name) function. This returns TRUE or FALSE (one of the two Boolean values).
- You can also call error_reporting(15), to set PHP's error reporting level so that it prints unbound variables.

Integer, String, and Boolean Type

- Integers in PHP correspond to C long's (signed), so there size depends on the size of a long on a given machine.
- This is usually 32 bits.
- PHP's double type corresponds to C's double and literals follow the format for C (or Perl or Javascript) floating point literals: I.e., .345, 3E-7, etc.
- Characters in PHP are single bytes (No Unicode!).
- String literal are built up out of these characters.
- PHP distinguish between single quote and double quote literals in the same way that Perl does. So the variable in "Bob=$bob" is interpolated but in 'Bob=$bob' is not.
- There are only two values for Boolean's: TRUE or FALSE. As for coercion, the empty string or a "0" or a "0.0" is interpreted as FALSE. Otherwise strings evaluate to TRUE. Integers and double evaluate to TRUE as long as they aren't zero.

Arithmetic Operators,etc

- PHP supports the usual arithmetic operators: +, -, *, /, %, ++, --.
- PHP also does type coercion.
- Division should be assumed to output a double if any fractional value exists.
- Some useful predefined functions are: floor, ceil, round, srand, rand, abs, min, and max.
- String concatenation is done with a “.”
- Some useful string operations are strlen, strcmp, strpos, substr, chop, trim, ltrim, strtolower, strtoupper.
- Type conversion can be done using expressions like (int)$sum or intval($sum) or settype($sum, integer).
- You can also check the type with is_int, is_integer, is_long, etc.
- Assignment operators are like in C, Java, etc.

Output

- You can output text to be inserted into the page using either of the commands: print or echo
- For example,

  print “hi there”;

Control Statements

- PHP supports:
  - if, else if, else: For example,

    ```
    if($a) print “hello”;
    ```
  - switch case:

    ```
    switch($a)
    {
        case 5:
         echo “hello”;
    }
    ```
  - for : for($a = 0; $a<10; $a++){echo “hello $a”;}
  - while: while(!$var) { /*do something*/}
  - do while: do {/*do something */} while (!$var);

Arrays

- Arrays can be declared with the syntax:

  $a = array(“hi”, 1, 2);
- You can nest arrays:

  $b = array(“hi”, array(1,2), 2);
- You can dereference arrays as in most languages:

  echo $a[0];
- You can cycle over elements of an array using foreach:
  foreach($arr as $var){echo $var;}

More on PHP Arrays

- 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

- 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.
- 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){…}