

PHP syntax template

HTML content

```
<?php
```

PHP code

```
?>
```

HTML content

```
<?php
```

PHP code

```
?>
```

HTML content ...

PHP

- any contents of a .php file between `<?php` and `?>` are executed as PHP code
- all other contents are output as pure HTML



String type

```
$favorite_food = "Lebanese";  
    print $favorite_food[2];
```

b

PHP

- zero-based indexing using bracket notation
- string concatenation operator is . (period), not +
 - 5 . "2 turtle doves" produces "52 turtle doves"
- can be specified with "" or ''



String functions

```
# index 0123456789012345
$name = "Austin Weale";
$length = strlen($name);           # 16
$cmp = strcmp($name, "Linda Guo"); # > 0
$index = strpos($name, "s");        # 2
$first = substr($name, 7, 4);       # "Weal"
$name = strtoupper($name);          # "AUSTIN WEALE" PHP
```

Name	Java Equivalent
<u>strlen</u>	length
<u>strpos</u>	indexOf
<u>substr</u>	substring
<u>strtolower</u> , <u>strtoupper</u>	toLowerCase, toUpperCase
<u>trim</u>	trim
<u>explode</u> , <u>implode</u>	split, join



Interpreted strings

```
$age = 16;  
print "You are " . $age . " years old.\n";  
print "You are $age years old.\n";      # You are 16 years old. PHP
```

- strings inside " " are interpreted
 - variables that appear inside them will have their values inserted into the string
- strings inside ' ' are not interpreted:

```
print 'You are $age years old.\n';      # You are $age years old.\n PHP
```



Arrays

```
$name = array(); # create
$name = array(value0, value1, ..., valueN);

$name[index] # get element value
$name[index] = value; # set element value
$name[] = value; # append PHP

$a = array(); # empty array (length 0)
$a[0] = 23; # stores 23 at index 0 (length 1)
$a2 = array("some", "strings", "in", "an", "array");
$a2[] = "Ooh!"; # add string to end (at index 5) PHP
```

- to append, use bracket notation without specifying an index
- element type is not specified; can mix types



Array functions

function name(s)	description
<u>count</u>	number of elements in the array
<u>print_r</u>	print array's contents
<u>array_pop</u> , <u>array_push</u> , <u>array_shift</u> , <u>array_unshift</u>	using array as a stack/queue
<u>in_array</u> , <u>array_search</u> , <u>array_reverse</u> , <u>sort</u> , <u>rsort</u> , <u>shuffle</u>	searching and reordering
<u>array_fill</u> , <u>array_merge</u> , <u>array_intersect</u> , <u>array_diff</u> , <u>array_slice</u> , <u>range</u>	creating, filling, filtering
<u>array_sum</u> , <u>array_product</u> , <u>array_unique</u> , <u>array_filter</u> , <u>array_reduce</u>	processing elements



Array function example

```
$tas = array("MD", "BH", "KK", "HM", "JP");  
for ($i = 0; $i < count($tas); $i++) {  
    $tas[$i] = strtolower($tas[$i]);  
}  
$morgan = array_shift($tas);  
array_pop($tas);  
array_push($tas, "ms");  
array_reverse($tas);  
sort($tas);  
$best = array_slice($tas, 1, 2);
```

	# ("md", "bh", "kk", "hm", "jp")
	# ("bh", "kk", "hm", "jp")
	# ("bh", "kk", "hm")
	# ("bh", "kk", "hm", "ms")
	# ("ms", "hm", "kk", "bh")
	# ("bh", "hm", "kk", "ms")
	# ("hm", "kk")

- the array in PHP replaces many other collections in Java
 - list, stack, queue, set, map, ...



bool (Boolean) type

```
$feels_like_summer = FALSE;  
$php_is_rad = TRUE;  
  
$student_count = 217;  
$nonzero = (bool) $student_count;      # TRUE
```

PHP

- the following values are considered to be FALSE (all others are TRUE):
 - 0 and 0.0
 - "", "0", and NULL (includes unset variables)
 - arrays with 0 elements
- can cast to boolean using (bool)
- FALSE prints as an empty string (no output); TRUE prints as a 1



Math operations

```
$a = 3;  
$b = 4;  
$c = sqrt(pow($a, 2) + pow($b, 2));
```

PHP

<u>abs</u>	<u>ceil</u>	<u>cos</u>	<u>floor</u>	<u>log</u>	<u>log10</u>	<u>max</u>
<u>min</u>	<u>pow</u>	<u>rand</u>	<u>round</u>	<u>sin</u>	<u>sqrt</u>	<u>tan</u>

math functions

- the syntax for method calls, parameters, returns is the same as Java



NULL

```
$name = "Victoria";  
$name = NULL;  
if (isset($name)) {  
    print "This line isn't going to be reached.\n";  
}
```

- a variable is NULL if
 - it has not been set to any value (undefined variables)
 - it has been assigned the constant NULL
 - it has been deleted using the unset function
- can test if a variable is NULL using the isset function
- NULL prints as an empty string (no output)



Printing HTML tags in PHP = bad style

```
<?php
print "<!DOCTYPE html>\n";
print "<html>\n";
print "    <head>\n";
print "        <title>Geneva's web page</title>\n";
...
for ($i = 1; $i <= 10; $i++) {
    print "<p class=\"count\"> I can count to $i! </p>\n";
}
?>
```

PHP

- printing HTML tags with print statements is bad style and error-prone:
 - must quote the HTML and escape special characters, e.g. \"
- but without print, how do we insert dynamic content into the page?



PHP expression blocks

<code><?= expression ?></code>	PHP
<code><h2> The answer is <?= 6 * 7 ?> </h2></code>	PHP
The answer is 42	output

- **PHP expression block:** evaluates and embeds an expression's value into HTML
- `<?= expr ?>` is equivalent to `<?php print expr; ?>`



Expression block example

```
<!DOCTYPE html>
<html>
  <head><title> Embedded PHP</title></head>
  <body>
    <?php for ($i = 99; $i >= 1; $i--) { ?>
      <p> <?= $i ?> bottles of beer on the wall, <br />
        <?= $i ?> bottles of beer. <br />
        Take one down, pass it around, <br />
        <?= $i - 1 ?> bottles of beer on the wall. </p>
    <?php } ?>
  </body>
</html>
```

PHP



Common errors: unclosed braces, missing = sign

```
<body>
  <p>Watch how high I can count:
    <?php for ($i = 1; $i <= 10; $i++) { ?>
      <? $i ?>
    </p>
  </body>
</html>
```

PHP

- </body> and </html> above are inside the for loop, which is never closed
- if you forget to close your braces, you'll see an error about 'unexpected \$end'
- if you forget = in <?=?, the expression does not produce any output



Complex expression blocks

```
<body>
  <?php for ($i = 1; $i <= 3; $i++) { ?>
    <h<?= $i ?>>This is a level <?= $i ?> heading.</h<?= $i ?>>
  <?php } ?>
</body>
```

PHP

This is a level 1 heading.

This is a level 2 heading.

This is a level 3 heading.

output

- expression blocks can even go inside HTML tags and attributes

