A Beginners Guide to PHP in WordPress

WordCamp Baltimore 2014
Zane M. Konik

Founder, Zane Matthew, inc.
http://zanematthew.com/
@zanematthew on twitter

@zkolnik on twitter
zane@zanematthew.com
Intro to coding, non-developers, what PHP do I need to know for WordPress?
How do I get it & what version?
Apple Customers

- MAMP
- MAMP Pro

How do I get it & what version?
Window Customers

- WAMP
- WAMP Pro

How do I get it & what version?
MAMP Pro/WAMP Pro

- Supports PHP versions
  - 5.2.17
  - 5.3.2
  - 5.4.1

How do I get it & what version?
PHP Versions

Source http://wordpress.org/about/stats/
If developing to contribute plugins or themes use 5.3.
Specifically *what* PHP code do I need to know?
1. Conditional Statements
2. Printing
3. Arrays (Ordered Maps)
4. Loops
5. Functions
6. File includes
If, else, elseif, switch
Conditional Statements

If, else, elseif, switch

Normally used with operators:

&& and
|| or
== equal
! false
!= not equal
=== exactly equal, i.e., ‘0’, 0
!== exactly not equal
<, >, <=, >= greater/less than
If, else, elseif, switch

...and type comparisons:

Conditional Statements
If, else, elseif, switch

...and type comparisons:

empty()

is_null()

isset()
## Conditional Statements

<table>
<thead>
<tr>
<th>Expression</th>
<th>gettype()</th>
<th>empty()</th>
<th>is_null()</th>
<th>isset()</th>
<th>boolean : if($x)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$x = &quot;&quot;;</code></td>
<td>string</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td><code>$x = null;</code></td>
<td>NULL</td>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td><code>var $x;</code></td>
<td>NULL</td>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td><code>$x</code> is undefined</td>
<td>NULL</td>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>FALSE</td>
</tr>
<tr>
<td><code>$x = array();</code></td>
<td>array</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td><code>$x = false;</code></td>
<td>boolean</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td><code>$x = true;</code></td>
<td>boolean</td>
<td>FALSE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td><code>$x = 1;</code></td>
<td>integer</td>
<td>FALSE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td><code>$x = 42;</code></td>
<td>integer</td>
<td>FALSE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td><code>$x = 0;</code></td>
<td>integer</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>
echo, print, printf(), sprintf()
echo vs. print?

Simply used to display content, echo is a language construct (its built into the language).

Just use one and stick with it.
printf() – Displays content

sprintf() – Returns content

*Most commonly used in translations.*
printf( "<strong>%s</strong>",
    __(‘Name’, ‘foo’)
);

sprintf( "%$2s, <strong>%$1s</strong>",
    __(‘Name’, ‘foo’),
    __(‘First’, ‘foo’)
);
Arrays

Indexed
Associative
Multi-dimensional
$pages_by_slug = array('about', 'blog', 'contact-me');

How would we traverse this?
$pages_by_slug = array('about', 'blog', 'contact-me');

How would we traverse this?

foreach( $pages_by_slug as $page ){
    echo $page;  // What does this display?
}

Indexed
$pages_by_slug = array(‘about’, ‘blog’, ‘contact-me’);

How would we traverse this?

foreach( $pages_by_slug as $page ){
    echo $page;  // Displays, about, blog, contact-me
}

Indexed
$pages = array(
    'page_name' => 'Contact',
    'page_id' => '89',
    'page_slug' => 'contact-me'
);

How would we traverse this?

Associative
$pages = array(
    ['page_name'] => 'Contact',
    ['page_id'] => '89',
    ['page_slug'] => 'contact-me'
);

How would we traverse this?

foreach( $pages as $page ){
    echo $page['page_name']; // about, blog, contact-me
}

Associative
$items = array(
    'pages' => array(
        'name' => 'Contact',
        'id' => 89,
        'slug' => contact-me
    ),
    'pages' => array(
        ...
    )
);

How would we traverse this?

Multi-dimensional
foreach( $items as $k => $v ){
    echo $k . ' ' . $v['slug']; // What does this display?
}

Multi-dimensional
How would we traverse this?

```php
foreach( $items as $k => $v ){
    echo $k . ' ' . $v['slug']; // pages about...
}
```

Multi-dimensional
Loops
$i = 1;
while( $i <= 10 ){
    echo $i++;
}

Loops
$i = 1;
while( $i <= 10 ){
    echo $i++;
}

WordPress has “the_loop”
Loops

```php
<?php if ( have_posts() ) : ?>
    <?php while ( have_posts() ) : the_post(); ?>
        <h1 class="entry-title"><?php the_title(); ?></h1>
    <?php endwhile; ?>
<?php endif; ?>
```
Functions
foo( ‘name: ‘, ‘tim’ );

function foo( $a=null, $b=null ){
    echo $a . ‘ ‘ . $b;
}

Functions
$a = array('name' => 'tim');

function foo( $a=array() ){
    extract( $a );
    echo $name;
}

Functions
1. require_once – fatal error
2. require – fatal error
3. include_once – warning error
4. include – warning error
WordPress has its own set of file includes
- **get_header();** // includes header.php
- **get_sidebar();** // includes sidebar.php
- **get_footer();** // includes footer.php
- **get_template_part( $path, $slug );** // includes 'partials/content-image.php'
- **load_template();** // includes $file
- **locate_template();** // returns a path to a template
Debugging – Why do you care?
“Debugging allows you to be accurate as an owl, that is approaching its first night meal.”
1. Notice – Found something that could indicate an error
2. Warning – Run-time errors, the code still works
3. Fatal – Code will not run, syntax error, memory allocation, etc.
1. Software
2. PHP config
3. wp-config.php
Just enable it
Via PHP

$error_reporting(-1);
ini_set('display_errors', 'On');
Via .htaccess

php_flag display_startup_errors on
php_flag display_errors on
php_flag log_errors on
php_value error_log /my/path/PHP_errors.log

ini_set & error_reporting
define('WP_DEBUG', true);

Shows errors, notices, warnings, and deprecated functions.
define('WP_DEBUG_LOG', true);

define('WP_DEBUG_DISPLAY', false);

Errors will not be shown, but can still be logged via define('WP_DEBUG_LOG', true);
define('SCRIPT_DEBUG', true);

Forces WordPress to use the "dev" versions of core CSS and Javascript files rather than the minified versions that are normally loaded.
define('SAVEQUERIES', true);

Stores queries in the global $wpdb->queries.
Note *all* of these methods should only be done in a *development* environment.
1 Obtaining PHP
2 Components – Conditional, printing, arrays (order maps), loops, file includes
3 Debugging

Conclusion
Questions
References