WordPress Custom Themes

Domenic J. Licata, Instructional Support Technician
University at Buffalo Department of Visual Studies
djlicata@buffalo.edu
• Contain CSS files and PHP template files that are used to generate requested pages, output to the viewer as HTML.

• Template files are made up of:
  • HTML, PHP, WordPress Template Tags, the Loop

• The parts of a page (structure and content) that remain static no matter what page you’re on can be hardcoded into the template files.

• Dynamic content appears through the loop.

• The complete list of template files recognized by WordPress can be found at codex.wordpress.org.

yoast.com/wordpress-theme-anatomy/
Static vs Changeable Content
Static vs Changeable Content

- Content that stays the same from page to page is “hard-coded” into the site theme templates.
Static vs Changeable Content

• Content that stays the same from page to page is “hard-coded” into the site theme templates.

• Content that changes from page to page is stored in a MySQL database and loaded into the page templates using a php “Loop”.
The Loop

- The Loop is PHP code used by WordPress to display posts (titles and content.) Note that pages behave like posts as far as retrieving content through the loop.

- A PHP if/else statement: WordPress asks the database if there are any posts. If and while there are if, display some information about each one; otherwise display an error message (or show nothing at all.)

```php
<?php
if ( have_posts() ) {
  while have_posts() {
    the_post();
    the_title();
    the_content();
  } // end while
} // end if
?>`
The Loop

- The Loop is PHP code used by WordPress to display posts (titles and content.) Note that pages behave like posts as far as retrieving content through the loop.

- A PHP if/else statement: WordPress asks the database if there are any posts. If and while there are if, display some information about each one; otherwise display an error message (or show nothing at all.)

```php
<?php
if ( have_posts() ) {
    while have_posts() {
        the_post();
        the_title();
        the_content();
    } // end while
} // end if
?>
```
The Loop can be limited to displaying only certain posts, for example those files in specific categories or with certain tags.
Common Theme Files

- **style.css—Required.** Contains special code so the theme can be chosen in the Dashboard.
- **index.php—Required.** Contains the loop which shows a list of recent blog posts. This is the default home page setting of a new WordPress installation.
- **header.php**—global file that displays headers and navigation, and usually the HTML head code and opening #wrapper and body tags.
- **footer.php**—global file that displays footers and also closing #wrapper, body and html tags.
- **sidebar.php**—global file that displays secondary content or nav. If no file is present but the call is made, WordPress will insert a default template.
- **home.php**—By default, used to render the Blog Posts Index.
- **front-page.php**—used for the landing page, if the it uses a different layout from inside pages.
- **page.php**—If present, controls what static pages look like. If wp is set to display a static home page instead of posts, page.php is used for home, and index.php is used for the posts page.
- **single.php**—Controls the display of individual posts.
Custom Template Pages

<?php

/*

* Template Name: [Name]

* Description: [Description]

*/

get_header(); ?>

• If you require a template that has a design structure unique from other pages, you can create your own, and assign it to any page you create in the dashboard.

• Custom Template pages must begin with this code.
Template Calls

• Template calls are special PHP code embedded along with the HTML in your template pages, that instruct Wordpress to bring in other template pages, or to perform special functions.
Required Template Calls

These calls are used (usually in index.php) to join two or more template files:

- `<php get_header(); ?>`
  - Placed at the very top of a template page, contains opening HTML elements `<head>` and `<body>` through `<header>`
- `<php get_sidebar(); ?>`
  - if a sidebar is needed
- `<php get_footer(); ?>`
  - Contains footer elements and closing `<body>` and `<html>` tags
These two calls import WordPress code necessary to run plugins and javascript:

- `<?php wp_head(); ?>`
  - Placed before the closing `</head>` tag.

- `<?php wp_footer(); ?>`
  - Placed before the closing `</body>` tag.
PHP for Dynamic Elements
Use Dynamic PHP Instead of Static Links When Possible

Dynamic Link to Embedded Images

```html
<img src="images/banner.png" alt="banner" />
```

becomes

```html
<img src="<?php echo get_template_directory_uri(); ?>/images/banner.png" alt="banner" />
```
Dynamic PHP instead of static URIs

Dynamic Link to CSS

```html
<link href="style.css" rel="stylesheet">
```

becomes

```html
<link href="<?php bloginfo('stylesheet_url');?>" rel="stylesheet" />
```
Dynamic Page Titles (for browser windows and bookmarks)

<title>Title of Page</title>

becomes

<title><?php global $page, $paged; wp_title( '|', true, 'right' ); bloginfo( 'name' ); ?></title>
Required CSS

/*

Theme Name: theme_name_here
Author: your_name
Author URI: website_where_to_find_you
Description: This is a theme ...
Version: 1.0
*/
The Loop

- Displays contents of the main area of a site
- Multiple loops can appear on one page
- Starts with a query (which posts or pages to display), and ends with a PHP “endwhile” statement. In between, all or selected contents of a post are displayed.

```php
<?php
if ( have_posts() ) {
    while ( have_posts() ) {
        the_post();
        the_title();
        the_content();
    } // end while
} // end if
?>
```
The Loop

- Query post or page
  - Start Loop
    - the_title (outputs the title of the post)
    - the_excerpt (outputs the post excerpt)
    - the_content (outputs the full post content)
    - the_category (outputs the post categories)
    - the_author (outputs the post author)
    - the_date (outputs the post date)
    - other tags (there is a variety of other tags you can use in the loop)
  - endwhile;
- Exit the loop
Custom Loop Queries

• Template tags can call title of the post/page the_title(); the content of the post the_content(); categories or tags associated with the post the_category(); and the_tags(); and so on.

• The Loop can be limited to displaying only certain posts, for example those files in specific categories or with certain tags. The Query is added to the second line of the loop.

• For example, this query would only show posts filed in the "Spotlight" category:
  • $query = new WP_Query( 'category_name=spotlight' );
functions.php

• You can change the behavior and add dynamic features to a site by adding a functions.php file to your theme. You can call functions that WordPress has built-in, or write your own.

• Two common uses of functions are to add custom navigation Menus and Widgets (small applications) areas to your theme.
Menus

• Custom nav menus can be easily controlled and modified in the Dashboard under Appearance>Menus. As new pages and child pages are added, menus can be updated dynamically without having to edit HTML.
Widgets

- Widgets are small applications that add content and features to Sidebars, most commonly, or to any widgetized area of your theme. Many widgets come with WordPress: for post categories, tag clouds, navigation, search, etc. Plugins can also add their own widgets.
Advanced Custom Fields

• If you require more than one container on a single page that will be filled with dynamic content, you can create a custom field for each of those areas, then add content to those fields from within the Dashboard. For example, you might have boxes in two or more different locations on a page, each with its own content.

• Install and activate the Advanced Custom Fields plugin.

• Within the “Custom Fields” section of your WordPress Dashboard, create the fields that you need, organized into “Field Groups”, and adjust the appropriate Field Settings.

• On the pages that will contain the fields, add content into the field through the Dashboard.

• Within your Theme template files, determine where the field content should appear. Place php code into your template, and add the necessary CSS to your style sheet. Example: `<div id="CustomFieldName"><php the_field('field_name'); ?></div>`
CSS Conditionals

- You can automatically load dynamic content based on the page that is being viewed.