WordPress Theme Development from Scratch

WordPress for Designers

digitaLinkR

Requirements

Tools/Software

1. Text editor/IDE (preferably Notepad++, Sublime Text, or Visual Studio Code)
2. Web server solution stack (preferably XAMPP or Laragon)
3. A clean WordPress installation

Technical

Learners must have at least basic knowledge about the following:

- HTML
- CSS
- Javascript
- PHP

digitaLinkR

Before we start

After every end of the topics. You will be given time for question/concerns. Raise your hands if you have questions/concerns about the topic/s.

digitaLinkR

Course Outline

1. Introduction
2. Theme Basics
3. Classic Themes
4. Theme Security
5. Theme Privacy
6. Advanced Theme Topics

digitaLinkR

About Me

Name: Darell Duma

- Freelance Web Developer
- Specializes in Platform Development using WordPress & Laravel

Youtube (just started/no contents yet):

- Web Shifu
- WordPress Docs Explained

Facebook Page: Darell Duma::Web Developer

digitaLinkR

Introduction

digitaLinkR

What is a Theme?

- changes the design of your website, often including its layout.
- changing your theme changes
    - how your site looks on the front-end
    - what a visitor sees when they browse to your site on the web
- take the content and data stored by WordPress and display it in the browser
- you decide how that content looks and is displayed

digitaLinkR

WordPress Licensing & the GPL

- you need to get acquainted with the GNU General Public License (GPL) that WordPress uses
- GPL Basic Freedoms
  - Freedom to *run the program* for any purpose.
  - Freedom to *study how the program works* and to change it, so it performs computing as you wish.
  - Freedom to *redistribute copies,* so you can help your neighbor.
  - Freedom to *distribute copies of your modified versions*, giving the community a chance to benefit from your changes.

“free as in speech, not as in beer.”

digitaLinkR

Setting up a Development Environment

- best to do it in an environment identical to the production server
- can either be local or remote
- Benefits
  - build your theme locally without relying on a remote server
  - do not need an Internet connection

digitaLinkR

Setting up a Development Environment

- What you need (at least)
  - Local server stack
    - MAMP
    - XAMPP
    - Docker
    - Laragon (I personally use)
  - Text Editor/IDE
    - Notepad++
    - Sublime Text
    - Atom
    - Brackets
    - Visual Studio Code (I personally use)

Setting up a Development Environment

- Supporting old versions
  - *at least two versions back*
  - *5.8 means you should make sure that it also works in 5.6 and 5.7*

digitaLinkR

Setting up a Development Environment

WP_DEBUG

- used to trigger the built-in “debug” mode on your WordPress installation
- allows you to view errors in your theme

wp-config.php

```
define( 'WP_DEBUG', true );
```

digitaLinkR

Setting up a Development Environment

WP_DEBUG_DISPLAY and WP_DEBUG_LOG

wp-config.php

```
define( 'WP_DEBUG_LOG', true );

define( 'WP_DEBUG_DISPLAY', true );
```

digitaLinkR

Setting up a Development Environment

WP_DEBUG_DISPLAY and WP_DEBUG_LOG

wp-config.php

```
define( 'WP_DEBUG_LOG', true );

define( 'WP_DEBUG_DISPLAY', true );
```

digitaLinkR

Setting up a Development Environment

Test Data

WordPress.org Theme Test Data

WordPress.org Theme Test Data is an XML file containing dummy test data that you can upload to test how themes perform with different types and layouts of content.

WordPress.com Theme Unit Test Data

WordPress.com Theme Unit Test Data is dummy test data that you can upload to a WordPress installation to test your theme, including WordPress.com-specific features.

Setting up a Development Environment

Plugins

- **Debug Bar** - adds an admin bar to your WordPress admin
- **Query Monitor** - allows debugging of database queries, API request and AJAX
- **Log Deprecated Notices -** logs incorrect function usage and the use of deprecated files and functions
- **Monster Widgets -** consolidates the core WordPress widgets into a single widget
- **Theme-Check -** tests your theme for compliance with the latest WordPress standards and practices

Setting up a Development Environment

Plugins

In addition to the above development tools, it's a good idea to stay up to date on the WordPress.org Themes Team's guidelines for theme submission and guidance on meeting WordPress Coding Standards. These guidelines are the "gold standard" for quality theme development and are useful, even if you don't plan on releasing a theme on WordPress.org.

Theme Development Examples

Default Twenty-twenty Themes

- Twenty Twenty-One
- Twenty Twenty
- Twenty Nineteen
- Twenty Seventeen
- Twenty Sixteen (only packaged in WordPress 4.8)

Other Sources: https://wordpress.org/themes/

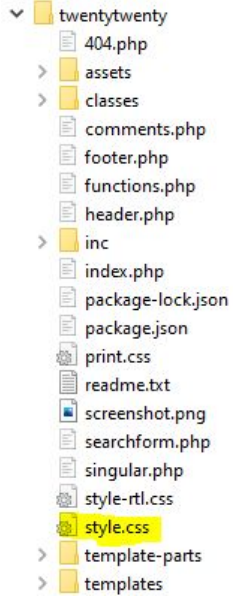
digitaLinkR

Theme Basics

digitaLinkR

Main Stylesheet (style.css)

- required for every WordPress theme
- controls the presentation (visual design and layout) of the website pages
- needs to be located in the root directory of your theme, not a subdirectory



digitaLinkR

Main Stylesheet (style.css)

Basic Structure

WordPress uses the header comment section of a style.css to display information about the theme in the Appearance (Themes) dashboard panel.

Main Stylesheet (style.css)

Example

/
Theme Name: Twenty Twenty
Theme URI: https://wordpress.org/themes/twentytwenty/
Author: the WordPress team
Author URI: https://wordpress.org/
Description: Our default theme for 2020 is designed to take full advantage of the flexibility of the block editor. Organizations and businesses have the ability to create dynamic landing pages with endless layouts using the group and column blocks. The centered content column and fine-tuned typography also makes it perfect for traditional blogs. Complete editor styles give you a good idea of what your content will look like, even before you publish. You can give your site a personal touch by changing the background colors and the accent color in the Customizer. The colors of all elements on your site are automatically calculated based on the colors you pick, ensuring a high, accessible color contrast for your visitors.
Tags: blog, one-column, custom-background, custom-colors, custom-logo, custom-menu, editor-style, featured-images, footer-widgets, full-width-template, rtl-language-support, sticky-post, theme-options, threaded-comments, translation-ready, block-styles, wide-blocks, accessibility-ready
Version: 1.3
Requires at least: 5.0
Tested up to: 5.4
Requires PHP: 7.0
License: GNU General Public License v2 or later
License URI: http://www.gnu.org/licenses/gpl-2.0.html
Text Domain: twentytwenty
This theme, like WordPress, is licensed under the GPL.
Use it to make something cool, have fun, and share what you've learned with others.
*/

Template Files

- exist within a theme and express how your site is displayed
- Template Hierarchy is the logic WordPress uses to decide which theme template file(s) to use, depending on the content being requested.
- Page Templates are those that apply to pages, posts, and custom post types to change their look and feel.
- In classic themes these are PHP files that contain a mixture of HTML, Template Tags, and PHP code.
- In block themes these are HTML files that contain HTML markup representing blocks.

digitaLinkR

Template Files

Template partials

- template part is a piece of a template that is included as a part of another template, such as a site header
- can be embedded in multiple templates, simplifying theme creation.
  - header.php or header.html for generating the site's header
  - footer.php or footer.html for generating the footer
  - sidebar.php or sidebar.html for generating the sidebar

digitaLinkR

Template Files

Common Wordpress Template Files

- index.php - main template file; required
- style.css - main stylesheet; required
- rtl.css - right-to-left stylesheet is included automatically if the website language's text direction is right-to-left
- front-page.php - always used as the site front page if it exists
- home.php - almost the same as front-page.php; used to show latest posts
- singular.php
    - used for posts when `single.php` is not found
    - or for pages when `page.php` are not found
    - If `singular.php` is not found, `index.php` is used
- search.php - used to display a visitor's search results
- 404.php - used when WordPress cannot find a post, page, or other content that matches the visitor's request.

digitaLinkR

Template Files

Using Template Files

In classic themes, within WordPress templates, you can use Template Tags to display information dynamically, include other template files, or otherwise customize your site.

```
<?php get sidebar(); ?>
<?php get template part( 'featured-content' ); ?>
<?php get_footer(); ?>
```

digitaLinkR

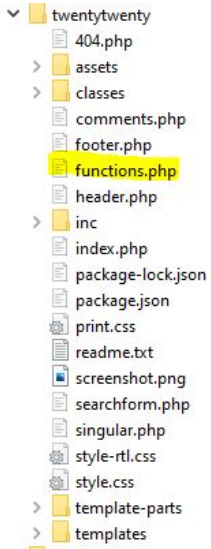
Template Hierarchy

digitaLinkR

Organizing Theme Files

```
assets (dir)
      - css (dir)
      - images (dir)
      - js (dir)
inc (dir)
template-parts (dir)
      - footer (dir)
      - header (dir)
      - navigation (dir)
      - page (dir)
      - post (dir)
404.php
archive.php
comments.php
footer.php
front-page.php
functions.php
header.php
index.php
page.php
README.txt
rtl.css
screenshot.png
search.php
searchform.php
sidebar.php
single.php
style.css
```

Theme Functions



- The `functions.php` file is where you add unique features to your WordPress theme
- can be used to hook into the core functions of WordPress to make your theme more modular, extensible, and functional

Post Types

Default Post Types

- Post (Post Type: ‘post’)
- Page (Post Type: ‘page’)
- Attachment (Post Type: ‘attachment’)
- Revision (Post Type: ‘revision’)
- Navigation menu (Post Type: ‘nav_menu_item’)
- Block templates (Post Type: ‘wp_template’)
- Template parts (Post Type: ‘wp_template_part’)

digitaLinkR

Including CSS and Javascript

```
wp_enqueue_style( $handle, $src, $deps, $ver, $media);

wp_enqueue_script( $handle, $src, $deps, $ver, $in_footer);
```

digitaLinkR

Including CSS and Javascript

Combining Enqueue Functions

```
function add theme scripts() {
  wp_enqueue_style( 'style', get_stylesheet_uri() );

  wp_enqueue_style( 'slider', get_template_directory_uri() . '/css/slider.css', array(), '1.1', 'all');

  wp_enqueue_script( 'script', get_template_directory_uri() . '/js/script.js', array ( 'jquery' ), 1.1, true);

    if ( is singular() && comments open() && get_option( 'thread_comments' ) ) {
      wp_enqueue_script( 'comment-reply' );
    }
}
add_action( 'wp_enqueue_scripts', 'add_theme_scripts' );
```

digitaLinkR

Categories, Tags and Custom Taxonomies

Default Taxonomies

The default taxonomies in WordPress are:

- categories: a hierarchical taxonomy that organizes content in the *post* Post Type
- tags: a non-hierarchical taxonomy that organizes content in the *post* Post Type
- post formats: a method for creating formats for your posts. You can learn more about these on the Post Formats page.

Terms

Terms are items within your taxonomy. So, for example, if you have the *Animal* taxonomy you would have the terms, dogs, cats, and sheep. Terms can be created via the WordPress admin, or you can use the wp_insert_term() function.

Categories, Tags and Custom Taxonomies

Custom Taxonomies

It is possible to create new taxonomies in WordPress. You may, for example, want to create an *author* taxonomy on a book review website, or an *actor* taxonomy on a film site. As with custom post type **it is recommended that you put this functionality in a plugin**. This ensures that when the user changes their website's design, their content is preserved in the plugin.

Classic Themes

digitaLinkR

Linking Theme Files and Directories

```
get_header( 'your_custom_template' );

get_footer( 'your_custom_template' );

get_sidebar( 'your_custom_template' );

get_template_part( 'content', 'product');
```

```
get_template_part( 'content-templates/content', 'location' );

get_template_part( 'content-templates/content', 'product' );

get_template_part( 'content-templates/content', 'profile' );
```

digitaLinkR

Linking Theme Files and Directories

Linking to Theme Directories

```
echo get_theme_file_uri( 'images/logo.png' );
```

Linking Theme Files and Directories

Dynamic Linking to Templates

```
<a href="<?php echo get_permalink($ID); ?>">This is a link</a>
```

digitaLinkR

Theme Loop

- default mechanism WordPress uses for outputting posts through a theme's template files.

```
if ( have_posts() ) :
    while ( have_posts() ) : the_post();
        the_content();
    endwhile;
else :
    _e( 'Sorry, no posts matched your criteria.', 'textdomain' );
endif;
```

digitaLinkR

Theme Loop

What the loop can display

- next_post_link() – a link to the post published chronologically *after* the current post
- previous_post_link() – a link to the post published chronologically *before* the current post
- the_category() – the category or categories associated with the post or page being viewed
- the_author() – the author of the post or page
- the_content() – the main content for a post or page
- the_excerpt() – the first 55 words of a post's main content followed by an ellipsis (…) or read more link that goes to the full post. You may also use the "Excerpt" field of a post to customize the length of a particular excerpt.
- the_ID() – the ID for the post or page
- the_meta() – the custom fields associated with the post or page
- the_shortlink() – a link to the page or post using the url of the site and the ID of the post or page
- the_tags() – the tag or tags associated with the post
- the_title() – the title of the post or page
- the_time() – the time or date for the post or page. This can be customized using standard php date function formatting.

Conditional Tags

- `is_home()` – Returns true if the current page is the homepage
- `is_admin()` – Returns true if inside Administration Screen, false otherwise
- `is_single()` – Returns true if the page is currently displaying a single post
- `is_page()` – Returns true if the page is currently displaying a single page
- `is_page_template()` – Can be used to determine if a page is using a specific template, for example:

  ```
  is_page_template('about-page.php')
  ```
- `is_category()` – Returns true if page or post has the specified category, for example: `is_category('news')`
- `is_tag()` – Returns true if a page or post has the specified tag
- `is_author()` – Returns true if inside author's archive page
- `is_search()` – Returns true if the current page is a search results page
- `is_404()` – Returns true if the current page does not exist
- `has_excerpt()` – Returns true if the post or page has an excerpt

digitaLinkR

Template Tags

- used within themes to **retrieve content from your database**.
- could be anything from a blog title to a complete sidebar.
- preferred method to pull content
    i. can print dynamic content;
    ii. can be used in multiple theme files
    iii. separate the theme into smaller, more understandable, sections.

digitaLinkR

Template Tags

What are Template Tags

- a piece of code that tells WordPress to get something from the database

Three Components

- A PHP code tag
- A WordPress function
- Optional parameters

digitaLinkR

Template Tags

What are Template Tags

- a piece of code that tells WordPress to get something from the database

Three Components

- A PHP code tag
- A WordPress function
- Optional parameters

digitaLinkR

Examples

`get_header()`

`get_footer()`

`the_title()`

`bloginfo( 'name' )`

digitaLinkR

Theme Functionality

digitaLinkR

Theme Functionality

Register Menu(s)

```
function register_my_menus() {
  register_nav_menus(
    array(
      'header-menu' => __( 'Header Menu' ),
      'extra-menu' => __( 'Extra Menu' )
     )
   );
 }
 add_action( 'init', 'register_my_menus' );
```

Display Menu

```
wp_nav_menu( array( 'theme_location'=> 'header-menu' ) );
```

digitaLinkR

Theme Functionality

Pagination

- `posts_nav_link();`
- `next_posts_link(); / previous_posts_link();`
- `the_posts_pagination(); / echo paginate_links();`

digitaLinkR

Theme Security

digitaLinkR

Data Sanitation

Securing Input

- sanitize_email()
- sanitize_file_name()
- sanitize_html_class()
- sanitize_key()
- sanitize_meta()
- sanitize_mime_type()
- sanitize_option()
- sanitize_sql_orderby()
- sanitize_text_field()
- sanitize_title()
- sanitize_title_for_query()
- sanitize_title_with_dashes()
- sanitize_user()
- esc_url_raw()
- wp_filter_post_kses()
- wp_filter_nohtml_kses()

digitaLinkR

Data Sanitation

Securing Output

- esc_html() – Use this function anytime an HTML element encloses a section of data being displayed.

```
<?php echo esc_html( $title ); ?>
```

- esc_url() – Use this function on all URLs, including those in the `src` and `href` attributes of an HTML element.

```
<img src="<?php echo esc_url( $great_user_picture_url ); ?>" />
```

- esc_js() – Use this function for inline Javascript.

```
<a href="#" onclick="<?php echo esc_js( $custom_js ); ?>">Click me</a>
```

- esc_attr() – Use this function on everything else that's printed into an HTML element's attribute.

```
<ul class="<?php echo esc_attr( $stored_class ); ?>"> </ul>
```

- esc_textarea() – encodes text for use inside a textarea element.

```
<textarea><?php echo esc_textarea( $text ); ?></textarea>
```

Data Validation

Securing Output

- esc_html() – Use this function anytime an HTML element encloses a section of data being displayed.

```
<?php echo esc_html( $title ); ?>
```

- esc_url() – Use this function on all URLs, including those in the `src` and `href` attributes of an HTML element.

```
<img src="<?php echo esc_url( $great_user_picture_url ); ?>" />
```

- esc_js() – Use this function for inline Javascript.

```
<a href="#" onclick="<?php echo esc_js( $custom_js ); ?>">Click me</a>
```

- esc_attr() – Use this function on everything else that's printed into an HTML element's attribute.

```
<ul class="<?php echo esc_attr( $stored_class ); ?>"> </ul>
```

- esc_textarea() – encodes text for use inside a textarea element.

```
<textarea><?php echo esc_textarea( $text ); ?></textarea>
```

Theme Privacy

digitaLinkR

Theme Privacy

- In WordPress 4.9.6, several tools were introduced to help sites meet the requirements of the new European Union's new GDPR (General Data Protection Regulation) laws.
- Theme authors should test their themes to confirm that there are no design conflicts between the features and their themes detailed below.
- WordPress 4.9.6 introduced the ability to select a page as a privacy policy for a site in the Settings > Privacy section of the admin area. For new sites, a privacy policy template page will automatically be created in draft status.
- To link to the selected page in plugins and themes, three template tags have been added:
  - get_privacy_policy_url() – Retrieves the URL to the privacy policy page.
  - the_privacy_policy_link() – Displays the privacy policy link with formatting, when applicable.
  - get_the_privacy_policy_link() – Returns the privacy policy link with formatting, when applicable.

digitaLinkR

Advanced Theme Topics

digitaLinkR

Child Themes

- allows you to change small aspects of your site's appearance yet still preserve your theme's look and functionality
- make your modifications portable and replicable
- keep customization separate from parent theme functions
- allow parent themes to be updated without destroying your modifications
- allow you to take advantage of the effort and testing put into parent theme
- save on development time since you are not recreating the wheel
- are a *great way* to start learning about theme development

digitaLinkR

How to Create a Child Theme

- Create a child theme folder
- Create a stylesheet style.css
- Enqueue Style
- Install/Activate Child Theme

```
add action( 'wp enqueue scripts', 'my_theme_enqueue_styles' );
function my theme enqueue styles() {
    wp enqueue style( 'child-style', get_stylesheet_uri(),
        array( 'parenthandle' ),
        wp get theme()->get('Version') // this only works if
you have Version in the style header
    );
}
```

```
/*
 Theme Name:   Twenty Fifteen Child
 Theme URI:
http://example.com/twenty-fifteen-child/
 Description:  Twenty Fifteen Child Theme
 Author:    John Doe
 Author URI:   http://example.com
 Template:  twentyfifteen
 Version:   1.0.0
 License:   GNU General Public License v2 or
later
 License URI:
http://www.gnu.org/licenses/gpl-2.0.html
 Tags:      light, dark, two-columns,
right-sidebar, responsive-layout,
accessibility-ready
 Text Domain:  twentyfifteenchild
*/
```

digitaLinkR

UI Best Practices

- logo at the top each page should send the user to the homepage of your site
- Good link text should give the reader an idea of the action that will take place when clicking it.

A bad example:

> The best way to learn WordPress is to start using it. To Download WordPress, click here.

A better example:

> Download WordPress and start using it. That's the best way to learn.

digitaLinkR

UI Best Practices

- Style links with underlines
- Different link colors
- Color contrast
- Sufficient font size
- Associate label with inputs

```
<label for="username">Username</label>
<input type="text" id="username" name="login" />
```

- Placeholder text in forms

```
<label for="name">Name</label>
<input type="text" id="name" name="name" placeholder="John Smith" />
```

- Descriptive buttons

```
<button>Reserve Seat</button>
```

digitaLinkR

Standard Javascript

- Avoid Global Variables
- Keep your JavaScript unobtrusive
- Use closures and the module pattern
- Stick to a coding style. Use The WordPress Javascript Coding Standard.
- Validate and Lint Your Code – ESLint.com
- Ensure your site still works without JavaScript first – then add JavaScript to provide additional capabilities. This is a form of Progressive Enhancement.
- Lazy load assets that aren't immediately required.
- Don't use jQuery *if you don't need to* — There's a great site that shows you how to do some common tasks with plain old JavaScript – you might not need jQuery

Theme Testing

1. Fix PHP and WordPress errors. Add the following debug setting to your `wp-config.php`file to see deprecated function calls and other WordPress-related errors:

   ```
   define('WP_DEBUG', true);
   ```

   See Deprecated Functions Hook for more information.
2. Check template files against Template File Checklist (see above).
3. Do a run-through using the Theme Unit Test.
4. Validate HTML and CSS. See Validating a Website.
5. Check for JavaScript errors.
6. Test in all your target browsers. For example Safari, Chrome, Opera, Firefox and Microsoft Edge.
7. Clean up any extraneous comments, debug settings or TODO items.
8. See Theme Review if you are publicly releasing the Theme by submitting it to the Themes Directory.

Plugin API Hooks

wp_head()

Goes at the end of the `<head>` element of a theme's *header.php* template file.

wp_body_open()

Goes at the beginning of the `<body>` element of a theme's *header.php* template file.

wp_footer()

Goes in *footer.php*, just before the closing `</body>` tag.

wp_meta()

Typically goes in the `<li>Meta</li>`section of a Theme's menu or sidebar.

comment_form()

Goes in *comments.php* directly before the file's closing tag (`</div>`).

What’s Next?

Questions?

Training Schedules

Thank You