Customizing WordPress

How do we customize WordPress? How do we make themse our own and add functionality that is not part of a theme or that you want to use regardless of the theme you have installed?

This post will explore the basics of creating custom WordPress elements and will cover plugins, child themes, and custom themes.

> ## Note
>
> The specifics of your situation will be different than mine. You should always analyze the your needs and requirements before making a decision.

Planning

The first thing to do when planning to add things to a theme is decide what's the best alternative to get it done.

Do you use an existing plugin to work with your theme? WordPress has a huge collection of plugins to accomplish a variety of tasks; it may just pay to evaluate existing plugins to see if they work.

Same things with theme. Evaluate existing plugins and test the existing capabilities and expand from there. Only if you evaluate existing themes and none do what you want without extensive modifications you should consider build a brand new theme using a starter kit like _s or the themes listed in articles like 10 Best WordPress Starter Themes for Developers in 2020 or 19 Best WordPress Starter Themes for Developers in 2020. **Test the themes and don't take anything for granted**.

Plugins

I recommend using plugins when you have a single-puropose task or script that you want to use across multiple themes or site.

For example, in a [previous post](#) I discussed how to add the scripts for Quotebacks to my WordPress installation. I want to make sure that this functionality works on every theme I may want to use so, rather than copying the enqueue function to whatever theme I choose to use, I create a simple plugin to do it once and leave it in the site regardless of the theme

I won't go into the details of creating a plugin, we could spend entire articles covering just that. Instead I will refer you to [Plugin Basics](#) from the [Plugin Developer Handbook](#) and [WordPress Essentials: How To Create A WordPress Plugin](#) as good starting points to use when creating your own plugins.

Themes

Themes are the look and feel of your WordPress site. They can also bundle functionality that the developer considered essential for the theme to work although the bundling is not required, you can give users a list of plugins to install on their own.

We'll discuss two ways to work with WordPress themes: Child Themes and Custom themes from scratch.

Child Themes

Child themes are the easiest way to customize a theme without writing it from scratch. Some of the advantages of using child themes:

- You're building on something that already exists, thus speeding up development time
- You can leverage the functionality of sophisticated frameworks and parent themes, while customizing the design to your needs
- Upgrade the parent theme without losing your customizations
- Since you didn't modify the parent theme you can disable the child theme and everything will be as it was before
- It's a good way to learn

Assuming that you've already uploaded the theme that you want to use as a parent; the idea is to create an empty folder and add two required files.

To create a child theme follow these steps:

1. Create a folder for your theme (in this case we'll call it `Twenty Twenty Rivendellweb`)
2. In the folder create the following files
   1. a `style.css` stylesheet
   2. a `functions.php` file

For now don't worry that the files are empty. We'll populate them as we move forward in the tutorial.

In `styles.css` add the following comment.

```
/*
 Theme Name:   Twenty Twenty Rivendellweb
 Theme URI:    https://github.com/caraya/2020-child
 Description:  Twenty Twenty Child Theme
 Author:       Carlos Araya
 Author URI:   https://publishing-project.rivendellweb.net/
 Template:     twentytwenty
 Version:      1.0.0
 License:      GPL2.0 or later
 License URI:  https://opensource.org/licenses/GPL-2.0
 Tags:         light, dark, two-columns, right-sidebar, responsive-layout
 Text Domain:  twentytwentyrivendellweb
*/
```

You can add further css customizations to make the site look the way you want to but it's not a requirement, the comment is enough.

In the PHP file, we need to add the parent theme's stylesheet and the style that we just created.

To add the style sheets in a WordPress safe way we create a function, `my_theme_enqueue_styles` to put the code we want to run.

```php
<?php
function my_theme_enqueue_styles() {

  // This is 'twentytwenty-style'
```

```
  // for the Twenty Twenty theme.
  $parent_style = 'parent-style';

  wp_enqueue_style( $parent_style,
    get_template_directory_uri() . '/style.css' );
  wp_enq'parent-style'hild-style',
    get_stylesheet_directory_uri() . '/style.css',
    array( $parent_style ),
    wp_get_theme()->get('Version')
  );
}
add_action( 'wp_enqueue_scripts', 'my_theme_enqueue_styles' );
```

The first call to wp_enqueue_styles will add the parent theme's stylesheet by using [get_template_directory()](#)to build a URI pointing to the parent theme.

The second wp_enqueue_styles will load the child theme's stylesheet by using [get_stylesheet_directory_uri()](#) to retrieve the locaation of the child theme's stylesheet.

The final steep is to call the wp_enqueue_scripts action with the function we created as the callback.

This is the basic child theme.

If we want to modify a template we copy it from the parent theme and modify the copy in the child theme.

Same with new templates. We create them in the child theme and edit them there.

You can find more information on child themes in [How To Create And Customize A WordPress Child Theme](#)

The Twenty Twenty child theme I created using these techniques is in a [Github Repo](#)

Custom Themes

As good as child themes are there are times when the number of changes make the child theme strategy too cumbersome.

That's where we create a brand new theme, either completely from scratch or using a starter theme.

There are plenty of starter themes developers to choose from. [19 Best WordPress Starter Themes for Developers in 2020](#) lists starter themes for you to consider.

There are themes that are bundled with libraries and designed for specific workflows. These starter themes include:

- [WP Rig](#) (starter theme with a build process baked in)
- [Bones](#) (a mobile-first HTML5 starter theme)
- [FoundationPress](#) (a WordPress starter theme for [Foundation](#)
- [UnderStrap](#) – an [Underscores](#) based starter theme incorporating [Bootstrap 4](#)

I selected [Underscores](#) (without Bootstrap) for my theme development. It gives me the best combination of ready-to-use features and ease of creating tooling around the elements we create.

Custom themes are way harder to conceptualize what we want to do and how we want to do it. Because of these difficulties, it's hard to prescribe what to do with custom themes. I would suggest starting with the following questions:

- What type of site am I designing?
- Do I need to add any additional templates?
- How am I planning to modify the existing templates?
- Who is the audience?
- What sites am I competing with?
- Do I have analytics to confirm my audience? Is there a comparable site to evaluate?
- What browsers do I have to support
    - How does browser support limit the technologies I want to use?
- How am I handling responsive design?

These are some basic questions, there are others that may become relevant as you

build the site.

I learned a lot about Underscores from Morten Rand-Hendriksen's [WordPress: Building Themes from Scratch Using Underscores](). Although the course is from 2017, the code hasn't changed enough, in my opinion, to not use it.