WordPress Child Themes

Version 1.0
Guido Verschoor

Contents

References:

reference for this Pdf:
1) https://codex.wordpress.org/Child_Themes
2) https://developer.wordpress.org/themes/advanced-topics/child-themes/
3) http://wordpress.stackexchange.com/questions/163301/versioning-import-of-parent-themes-style-css
4) http://techblog.kjodle.net/2014/04/12/wordpress-child-themes-the-whys-and-hows/

Resources

be aware that some of these resources recommend using @import from your child theme's stylesheet as the method of importing the parent theme stylesheet. Please use the `wp_enqueue_script()` method described in this Pdf.

Theme Development
How to Modify WordPress Themes the Smart Way (four part series)
How To Make Your Own Child Theme - Includes Screencast
Guide to WordPress Child Theme Development
How to: Create a Child Theme based on Twenty Eleven
Customizing your WordPress theme using Firebug
Tutorial: Child Themes basics and creating Child Themes in WordPress
How to modify the Parent Theme behavior within the Child Theme
Child Theme Plugins
WordPress Child Theme The Why and How

extra reading - gives actual examples of style.css and function.php pages:
How to create Twentythirteen child theme for WordPress

Theme downloads (will add more soon):
Extra Themes
Big Theme

What is a child theme?

a child theme is a theme that inherits the functionality and styling of another theme, called the parent theme. Child themes are the recommended way of modifying an existing theme. It can be as simple or as complex as you need or want it to be.

In actual terms, a child theme is just a directory (i.e. a folder) with a single child theme style sheet in it. You can add other files as you need them, but this is the minimum requirement.

A parent theme is a complete theme which includes all of the required WordPress template files, images, assets, etc. for the theme to work. all themes, excluding child themes, are parent themes. Child themes of child themes, or grandchild themes, are not possible.

Note: If you are making extensive customisation – beyond styles and a few theme files – creating a parent theme might be a better option than a child theme. Creating a parent theme allows you to avoid issues with deprecated code in the future. This needs to be decided on a case-by-case basis.

Why use a Child Theme?

There are a few reasons why you would want to use a child theme:

- If you modify a theme directly and it is updated, then your modifications may be lost. By using a child theme you will ensure that your modifications are preserved.
- using a child theme can speed up development time.
- using a child theme is a great way to to learn about WordPress theme development.

How to Create a Child Theme

A child theme consists of at least one directory (the child theme directory) and two files (style.css and functions.php), which you will need to create:

- The child theme directory
- style.css
- functions.php

The first step in creating a child theme is to create the child theme directory, which will be placed in wp-content/themes. It is recommended (though not required, especially if you're creating a theme for public use) that the name of your child theme directory is appended with '-child'. you will also want to make sure that there are no spaces in your child theme directory name, which may result in errors. In the screenshot above we have called our child theme 'twentyfifteen-child', indicating that the parent theme is the Twenty fifteen theme.

Part 1 - Using the style.css

The next step is to create your child theme's stylesheet (style.css). The stylesheet must begin with the following (the stylesheet 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:  twenty-fifteen-child
*/
```

here is an ***example*** of the stylesheet:

```
/*
Theme Name: Twenty Thirteen Child
Theme URI: http://wordpress.transformnews.com/
Description: Twenty Thirteen Child theme!
Author: M.R.D.A.
Author URI: http://wordpress.transformnews.com/
Template: twentythirteen
Version: 0.1
*/

/* This is must */
@import url("../twentythirteen/style.css");

/* Add some custom fonts */
@font-face {
  font-family: CustomFont;
      src:  url("fonts/Custom_Font-webfont.eot");
      src:  url("fonts/Custom_Font-webfont.eot?#iefix") format("embedded-opentype"),
            url("fonts/Custom_Font-webfont.woff") format("woff"),
            url("fonts/Custom_Font-webfont.ttf") format("truetype"),
            url("fonts/Custom_Font-webfont.svg#CustomFont") format("svg");
      font-weight: normal;
      font-style: normal;
}
```

a couple things to note:

- you will need to replace the example text with the details relevant to your theme.
- The Template line corresponds to the directory name of the parent theme. The parent theme in our example is the Twenty Fifteen theme, so the Template will be twentyfifteen. You may be working with a different theme, so adjust accordingly.
- The only required child theme file is style.css, but functions.php is necessary to enqueue styles correctly (below).

Why not just copy the entire stylesheet over to the child theme?

you really want to avoid doing this. for reasons I don't entirely (yet) understand, this causes a lot of problems. Instead, just copy over those portions that you changed.

for example, if your original theme contained this block of code:

```
.page-title {
    background: #2070B7;
    color: #e3e3e3;
    font-size: 16px;
    margin: 0 auto 10px;
    padding: 8px 0;
    text-align: center;
    text-shadow: 0 -1px 0 #333;
}
```

and you changed only the font size property, so that it looks like this:

```
.page-title {
    background: #2070B7;
    color: #e3e3e3;
    font-size: 24px;
    margin: 0 auto 10px;
    padding: 8px 0;
    text-align: center;
    text-shadow: 0 -1px 0 #333;
}
```

Then the only thing you should include in your child theme would be this:

```
.page-title {font-size: 24px;}
```

The final step is to enqueue the parent and child theme stylesheets. Note that the previous method was to import the parent theme stylesheet using **@import:** this is no longer best practice.

The correct method of enqueuing the parent theme stylesheet is to add a `wp_enqueue_scripts action` and use `wp_enqueue_style()` in your child theme's **functions.php**. you will therefore need to create a functions.php in your child theme directory. The first line of your child theme's functions.php will be an opening PHP tag (<?php), after which you can enqueue your parent and child theme stylesheets. The following example function will only work if your Parent Theme uses only one main style.css to hold all of the css. If your theme has more than one .css file (eg. ie.css, style.css, main.css) then you will have to make sure to maintain all of the Parent Theme dependencies.

```
add_action( 'wp_enqueue_scripts', 'theme_enqueue_styles' );
function theme_enqueue_styles() {
    wp_enqueue_style( 'parent-style', get_template_directory_uri() . '/style.css' );

}
```

Your child theme's stylesheet will usually be loaded automatically. If it is not, you will need to enqueue it as well. setting 'parent-style' as a dependency will ensure that the child theme stylesheet loads after it. see here a more detailed discussion :

```
function theme_enqueue_styles() {

    $parent_style = 'parent-style';

    wp_enqueue_style( $parent_style, get_template_directory_uri() . '/style.css' );
```

```
    wp_enqueue_style( 'child-style',
        get_stylesheet_directory_uri() . '/style.css',
        array( $parent_style )
    );
}
add_action( 'wp_enqueue_scripts', 'theme_enqueue_styles' );
```

your child theme is now ready for activation. log in to your site's administration panel, and go to **Administration Panels > Appearance > Themes.** you should see your child theme listed and ready for activation. (If your WordPress installation is multi-site enabled, then you may need to switch to your network administration panel to enable the theme (within the network admin Themes screen tab). You can then switch back to your site-specific WordPress administration panel to activate your child theme.)
note: you may need to re-save your menu (**Appearance > Menus, or Appearance > Customize > Menus)** and theme options (including background and header images) after activating the child theme.
Template files

If you want to change more than just the stylesheet, your child theme can override any file in the parent theme: simply include a file of the same name in the child theme directory, and it will override the equivalent file in the parent theme directory when your site loads. For instance, if you want to change the PhP code for the site header, you can include a header.php in your child theme's directory, and that file will be used instead of the parent theme's header.php.

You can also include files in the child theme that are not included in the parent theme. For instance, you might want to create a more specific template than is found in your parent theme, such as a template for a specific page or category archive. See the Template Hierarchy for more information about how WordPress decides what template to use.

Part 2 - Using functions.php

here is an ***example*** of the functions.php page:

```
<?php
// in your Child Theme's functions.php
// Use the after_setup_theme hook with a priority of 11 to load after the
// parent theme, which will fire on the default priority of 10

// Remove meta generator (WP version) from site and feed
if ( ! function_exists( 'mywp_remove_version' ) ) {

function mywp_remove_version() {
            return '';
}
add_filter('the_generator', 'mywp_remove_version');
}

// Clean header from unneeded links
if ( ! function_exists( 'mywp_head_cleanup' ) ) {

function mywp_head_cleanup() {
            remove_action('wp_head', 'feed_links', 2);  // Remove Post and Comment Feeds
            remove_action('wp_head', 'feed_links_extra', 3);  // Remove category feeds
            remove_action('wp_head', 'rsd_link'); // Disable link to Really Simple Discovery
service
            remove_action('wp_head', 'wlwmanifest_link'); // Remove link to the Windows
Live Writer manifest file.
            /*remove_action( 'wp_head', 'index_rel_link' ); */ // canonic link
            remove_action( 'wp_head', 'parent_post_rel_link', 10, 0 ); // prev link
            remove_action( 'wp_head', 'start_post_rel_link', 10, 0 ); // start link
            remove_action('wp_head', 'adjacent_posts_rel_link_wp_head', 10, 0);  // Remove
relation links for the posts adjacent to the current post.
            remove_action('wp_head', 'wp_shortlink_wp_head', 10, 0);

global $wp_widget_factory;
            remove_action('wp_head', array($wp_widget_factory->widgets['WP_Widget_Recent_
Comments'], 'recent_comments_style'));
add_filter('use_default_gallery_style',  '__return_null');
}
add_action('init', 'mywp_head_cleanup');
}
// Hook into the 'widgets_init' action
add_action( 'widgets_init', 'custom_sidebar' );
?>
```

unlike style.css, the functions.php of a child theme does not override its counterpart from the parent. Instead, it is loaded in addition to the parent's functions.php. (Specifically, it is loaded right before the parent's file.)

In that way, the functions.php of a child theme provides a smart, trouble-free method of modifying the functionality of a parent theme. say that you want to add a PhP function to your theme. The fastest way would be to open its functions.php file and put the function there. But that's not smart: The next time your theme is updated, your function will disappear. but there is an alternative way which is the smart way: you can create a child theme, add a functions.php file in it, and add your function to that file. The function will do the exact same job from there too, with the advantage that it will not be affected by future updates of the parent theme. do not copy the full content of functions.php of the

parent theme into functions.php in the child theme.
The structure of functions.php is simple: an opening PhP tag at the top, and below it, your bits of PhP. In it you can put as many or as few functions as you wish. The example below shows an elementary functions.php file that does one simple thing: Adds a favicon link to the head element of HTML pages.

```
<?php // Opening PHP tag - nothing should be before this, not even whitespace

// Custom Function to Include
function favicon_link() {
    echo '<link rel="shortcut icon" type="image/x-icon" href="/favicon.ico" />' . "\n";
}
add_action( 'wp_head', 'favicon_link' );

TIP FOR THEME DEVELOPERS. The fact that a child theme's functions.php is loaded first means
that you can make the user functions of your theme pluggable —that is, replaceable by a
child theme— by declaring them conditionally. E.g.:

if ( ! function_exists( 'theme_special_nav' ) ) {
    function theme_special_nav() {
        //  Do something.
    }
}
```

In that way, a child theme can replace a PhP function of the parent by simply declaring it beforehand.

Referencing / Including Files in Your Child Theme

When you need to include files that reside within your child theme's directory structure, you will use `get_stylesheet_directory()`. because the parent template's style.css is replaced by your child theme's style.css, and your style.css resides in the root of your child theme's subdirectory, `get_stylesheet_directory()` points to your child theme's directory (not the parent theme's directory).

Here's an example, using require_once, that shows how you can use get_stylesheet_directory when referencing a file stored within your child theme's directory structure.

```
require_once( get_stylesheet_directory() . '/my_included_file.php' );
```

Part 3 - Other Useful Information

Using Post Formats

A child theme inherits post formats as defined by the parent theme. When creating child themes, be aware that using `add_theme_support('post-formats')` will override the formats defined by the parent theme, not add to it.
RTl support

To support RTL languages, add rtl.css file to your child theme, containing:

```
/*
Theme Name: Twenty Fourteen Child
Template: twentyfourteen
*/
```

rtl.css is only loaded by WordPress if is_rtl() returns true.

It's recommended to add the rtl.css file to your child theme even if the parent theme has no rtl.css file.

What if I've already made changes to my original theme?

now you have a bit of a problem, but don't worry. It's perfectly manageable, but it's going to take some time.

First, FTP to your web site. Go into your WordPress folder, find a folder called "**wp-content**". Inside that folder is another folder labeled "**themes**". Inside that folder is the folder that contains your theme, the one you've been making changes to. find the stylesheet and download it. (If you've made changes to other theme files, you'll want to download those, as well.)

now, head over to the WordPress theme repository and download an unaltered copy of your theme. If you bought a theme from a commercial theme developer, you'll need to check with them about getting a fresh copy.

What you now need is a way to quickly and easily compare your changed files with the unchanged ones you just downloaded. There are at least four sites where you can do this online:

diffnow allows for copying and pasting of text files, or a simple upload.
Text Compare! allows for copying and pasting of files.
diff Checker allows for copying and pasting of files.
Quickdiff allows for copying and pasting of files.
Winmerge is another useful program, which is a program you download and use locally.

With the tool of your choice, find the difference between the original stylesheet and the one you've altered. Copy those changes over to your child theme stylesheet, saving as you go. once you've done that, you can upload your child theme, upload a fresh copy of your original theme, and then select your child theme as the current theme. (because child themes are, from WordPress's point of view, different themes, you will need to reset your header and background.)

If you have made changes to the theme's functions.php file, you'll want to do the same thing.

What if I've edited other files besides the stylesheet?

Then things become a bit more complicated. What you'll need to is to create copies of those files in

your child theme. I don't have a lot of experience with this, so I don't want to get into this too deeply, as I don't want to give you bad advice, and some of this depends on the way your theme is structured. For a start, you could simply copy the files you have altered over to your child theme folder, and you should be okay. But see the resources in the Codex about this first.

Summary

as you can see, WordPress child themes can be as simple or as complicated as you need them to be. They are not difficult to set up, and can help you learn the basics of HTML, CSS, and even PHP. You may have so much fun with them that you decide to create your own WordPress theme or plugin. enjoy!