Theme Functions

- What is functions.php?
- Examples
  - Theme Setup
    - Automatic Feed Links
    - Navigation Menus
    - Load Text Domain
    - Post Thumbnails
    - Post Formats
    - Theme support in block themes
    - Initial Setup Example
  - Content Width
  - Other Features
- Your functions.php File

`Functions.php` can be used by both classic themes, block themes, and child themes.

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

What is `functions.php`?

The `functions.php` file behaves like a WordPress plugin, adding features and functionality to a WordPress site. You can use it to call WordPress functions and to define your own functions.

The same result can be produced using either a plugin or `functions.php`. If you are creating new features that should be available no matter what the website looks like, **it is best practice to put them in a plugin**.

There are advantages and tradeoffs to either using a WordPress plugin or using `functions.php`.

A WordPress plugin:

- requires specific, unique header text;
- is stored in wp-content/plugins, usually in a subdirectory;
- only executes on page load when activated;
- applies to all themes; and
- should have a single purpose – for example, offer search engine optimization features or help with backups.

Meanwhile, a `functions.php` file:

- requires no unique header text;
- is stored in theme's subdirectory in wp-content/themes;
- executes only when in the active theme's directory;
- applies only to that theme (if the theme is changed, the features can no longer be used); and
- can have numerous blocks of code used for many different purposes.

Each theme has its own functions file, but only code in the active theme's `functions.php` is actually run. If your theme already has a functions file, you can add code to it. If not, you can create a plain-text file named `functions.php` to add to your theme's directory, as explained below.

A child theme can have its own `functions.php` file. Adding a function to the child functions file is a risk-free way to modify a parent theme. That way, when the parent theme is updated, you don't have to worry about your newly added function disappearing.

Although the child theme's `functions.php` is loaded by WordPress right before the parent theme's `functions.php`, it does not *override* it. The child theme's `functions.php` can be used to augment or replace the parent theme's functions. Similarly, `functions.php` is loaded *after any plugin files have loaded*.

With `functions.php` you can:

- Use WordPress hooks. For example, with the `excerpt_length` filter you can change your post excerpt length (from default of 55 words).
- Enable WordPress features with `add_theme_support()`. For example, turn on post thumbnails, post formats, and navigation menus.
- Define functions you wish to reuse in multiple theme template files.

In WordPress, naming conflicts can occur when two or more functions, classes, or variables have the same name. This can cause errors or unexpected behavior in a WordPress site. It is the responsibility of both the theme developer and plugin developer to avoid naming conflicts in their

respective code.

Theme developers should ensure that their functions, classes, and variables have unique names that do not conflict with those used by WordPress core or other plugins. They should also prefix their function and class names with a unique identifier, such as the theme name or abbreviation, to minimize the chances of a naming conflict.

Examples

Below are a number of examples that you can use in your functions.php file to support various features. Each of these examples are allowed in your theme if you choose to submit it to the WordPress.org theme directory.

Theme Setup

A number of theme features should be included within a “setup” function that runs initially when your theme is activated. As shown below, each of these features can be added to your `functions.php` file to activate recommended WordPress features.

It’s important to namespace your functions with your theme name. All examples below use `myfirsttheme_` as their namespace, which should be customized based on your theme name.

To create this initial function, start a new function entitled `myfirsttheme_setup()`, like so:

```
if ( ! function_exists( 'myfirsttheme_setup' ) ) :
/**
 * Sets up theme defaults and registers support for various WordPress
 * features.
 *
 * It is important to set up these functions before the init hook so
 * that none of these features are lost.
 *
 *  @since MyFirstTheme 1.0
 */
function myfirsttheme_setup() { ... }
```

Note: In the above example, the function myfirsttheme_setup is started but not closed out. Be sure to close out your functions.

Automatic Feed Links

Automatic feed links enables post and comment RSS feeds by default. These feeds will be displayed in `<head>` automatically. They can be called using `add_theme_support()` in classic themes. This feature is automatically enabled for block themes, and does not need to be included during theme setup.

```
add_theme_support( 'automatic-feed-links' );
```

Navigation Menus

In classic themes, custom navigation menus allow users to edit and customize menus in the Menus admin panel, giving users a drag-and-drop interface to edit the various menus in their theme.

You can set up multiple menus in `functions.php`. They can be added using `register_nav_menus()` and inserted into a theme using `wp_nav_menu()`, as discussed later in this handbook. If your theme will allow more than one menu, you should use an array. While some themes will not have custom navigation menus, it is recommended that you allow this feature for easy customization.

```
register_nav_menus( array(
    'primary'   => __( 'Primary Menu', 'myfirsttheme' ),
    'secondary' => __( 'Secondary Menu', 'myfirsttheme' )
) );
```

Each of the menus you define can be called later using `wp_nav_menu()` and using the name assigned (i.e. primary) as the `theme_location` parameter.

In block themes, you use the navigation block instead.

Load Text Domain

Themes can be translated into multiple languages by making the strings in your theme available for translation. To do so, you must use `load_theme_textdomain()`. For more information on making your theme available for translation, read the internationalization section.

```
load_theme_textdomain( 'myfirsttheme', get_template_directory() . '/languages' );
```

Post Thumbnails

Post thumbnails and featured images allow your users to choose an image to represent their post. Your theme can decide how to display them, depending on its design. For example, you may choose to display a post thumbnail with each post in an archive view. Or, you may want to use a large featured image on your homepage. This feature is automatically enabled for block themes, and does not need to be included during theme setup.

```
add_theme_support( 'post-thumbnails' );
```

Post Formats

Post formats allow users to format their posts in different ways. This is useful for allowing bloggers to choose different formats and templates based on the content of the post. `add_theme_support()` is also used for Post Formats. This is **recommended**.

```
add_theme_support( 'post-formats',  array( 'aside', 'gallery', 'quote', 'image', 'video' ) );
```

Learn more about post formats.

Theme support in block themes

In block themes, HTML5 markup as well as the following theme supports are enabled automatically:

```
add_theme_support( 'post-thumbnails' );
add_theme_support( 'responsive-embeds' );
add_theme_support( 'editor-styles' );
add_theme_support( 'automatic-feed-links' );
```

Initial Setup Example

Including all of the above features will give you a `functions.php` file like the one below. Code comments have been added for future clarity.

As shown at the bottom of this example, you must add the required `add_action()` statement to ensure the `myfirsttheme_setup` function is loaded.

```
if ( ! function_exists( 'myfirsttheme_setup' ) ) :
    /**
     * Sets up theme defaults and registers support for various
     * WordPress features.
     *
     * Note that this function is hooked into the after_setup_theme
     * hook, which runs before the init hook. The init hook is too late
     * for some features, such as indicating support post thumbnails.
     */
    function myfirsttheme_setup() {

    /**
```

```
     * Make theme available for translation.
     * Translations can be placed in the /languages/ directory.
     */
        load_theme_textdomain( 'myfirsttheme', get_template_directory() . '/languages' );

        /**
         * Add default posts and comments RSS feed links to <head>.
         */
        add_theme_support( 'automatic-feed-links' );

        /**
         * Enable support for post thumbnails and featured images.
         */
        add_theme_support( 'post-thumbnails' );

        /**
         * Add support for two custom navigation menus.
         */
        register_nav_menus( array(
            'primary'   => __( 'Primary Menu', 'myfirsttheme' ),
            'secondary' => __( 'Secondary Menu', 'myfirsttheme' ),
        ) );

        /**
         * Enable support for the following post formats:
         * aside, gallery, quote, image, and video
         */
        add_theme_support( 'post-formats', array( 'aside', 'gallery', 'quote', 'image',
'video' ) );
    }
endif; // myfirsttheme_setup
add_action( 'after_setup_theme', 'myfirsttheme_setup' );
```

Content Width

In classic themes, a content width is added to your `functions.php` file to ensure that no content or assets break the container of the site. The content width sets the maximum allowed width for any content added to your site, including uploaded images. In the example below, the content area has a maximum width of 800 pixels. No content will be larger than that.

```
if ( ! isset ( $content_width) ) {
    $content_width = 800;
}
```

Themes that include a theme.json configuration file does not need to include the variable in functions.php. Instead, the content width is added to the layout setting in theme.json. You can learn more about using theme.json in the advanced section.

Other Features

There are other common features you can include in `functions.php`. Listed below are some of the most common features. Click through and learn more about each of these features.

- Custom Headers **-Classic themes**
- Sidebars (widget areas) **-Classic themes**
- Custom Background **-Classic themes**
- Title tag **-Classic themes**
- Add Editor Styles
- HTML5 **-Classic themes**

Your *functions.php* File

If you choose to include all the functions listed above, this is what your *functions.php* might look like. It has been commented with references to above.

```
/**
 * MyFirstTheme's functions and definitions
 *
 * @package MyFirstTheme
 * @since MyFirstTheme 1.0
 */

/**
 * First, let's set the maximum content width based on the theme's
 * design and stylesheet.
 * This will limit the width of all uploaded images and embeds.
 */
if ( ! isset( $content_width ) ) {
    $content_width = 800; /* pixels */
```

```
}


if ( ! function_exists( 'myfirsttheme_setup' ) ) :

    /**
     * Sets up theme defaults and registers support for various
     * WordPress features.
     *
     * Note that this function is hooked into the after_setup_theme
     * hook, which runs before the init hook. The init hook is too late
     * for some features, such as indicating support post thumbnails.
     */
    function myfirsttheme_setup() {

        /**
         * Make theme available for translation.
         * Translations can be placed in the /languages/ directory.
         */
        load_theme_textdomain( 'myfirsttheme', get_template_directory() . '/languages' );

        /**
         * Add default posts and comments RSS feed links to <head>.
         */
        add_theme_support( 'automatic-feed-links' );

        /**
         * Enable support for post thumbnails and featured images.
         */
        add_theme_support( 'post-thumbnails' );

        /**
         * Add support for two custom navigation menus.
         */
        register_nav_menus( array(
            'primary'   => __( 'Primary Menu', 'myfirsttheme' ),
            'secondary' => __( 'Secondary Menu', 'myfirsttheme' ),
        ) );

        /**
         * Enable support for the following post formats:
```

```
         * aside, gallery, quote, image, and video
         */
        add_theme_support( 'post-formats', array( 'aside', 'gallery', 'quote', 'image',
'video' ) );
    }
endif; // myfirsttheme_setup
add_action( 'after_setup_theme', 'myfirsttheme_setup' );
```

□□□ #2
□ □□□ □□□ 2026-10-31 18:35:36 UTC
□ □□□ □□□ 2026-10-31 18:35:38 UTC