WordPress Plugin Development

Darko Gjorgjijoski

Freelance Web Developer with over 7 years of experience

Interests: Back-end, Databases, DevOps, Security and occasionally gaming

gdarko darkog dg.mk

WordCamp Skopje - October 05-06 2019, FINKI

Introduction to Plugin Development

Based on
https://github.com/gdarko/books-library

Why plugins?

- To add new functionality on your site
- To modify existing functionality on your site
- Plugins are portable - you can re-use them on other WordPress sites
- To reduce your development time. There are over **55 000** free plugins on the official plugins directory and many other that are private or premium.

Development Resources

Lots of resources available on internet. Always use the **codex.***, the **developer.*** portals first.

- https://codex.wordpress.org/
- https://developer.wordpress.org/
- https://developer.wordpress.org/plugins/intro/
- WordPress source code itself https://github.com/wordpress/wordpress
- Stackoverflow https://wordpress.stackexchange.com 😎

Development Environment

At least **PHP 5.6.20**, Web Server (nginx, litespeed, apache, etc.) and MySQL/MariaDB database.

The recommended **PHP version** is always the **latest** stable **PHP version**.

Some popular environments for development are as follows:

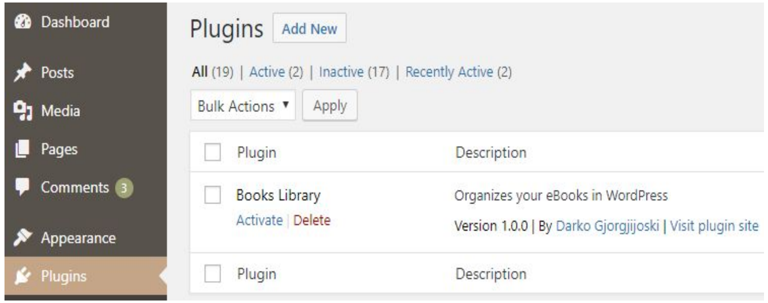
- Xampp
- Bitnami
- Local by flywheel
- Or just FTP/SFTP access to web server

Getting started

The first step is to create the plugin folder in **wp-content/plugins/**. Inside this folder (eg **books-library**) we need one php file that initializes the plugin eg. **books-library.php**.

Plugin declaration is done with PHP comment block in the **books-library/books-library.php** file as follows:

```
<?php
/*
Plugin Name: Books Library
Plugin URI: https://thepluginurl.com
Description: Organizes your eBooks in WordPress
Author: Darko Gjorgjijoski
Version: 1.0.0
Author URI: https://darkog.com/
*/
```



Plugin	Description
Books Library Activate | Delete	Organizes your eBooks in WordPress Version 1.0.0 | By Darko Gjorgjijoski | Visit plugin site
Plugin	Description

Publishing on the plugins directory

In order to successfully publish you are required to meet the following requirements:

- To have valid **wordpress.org** account
- To have valid **readme.txt** file (https://wordpress.org/plugins/developers/readme-validator/)
- To **not use any trademarked word** as first word in your plugin name/slug or
- To **not use any trademarked logo** it in your marketing assets

Plugins must be submitted for review at https://wordpress.org/plugins/developers/add/

If approved you get **SVN** repository access to store your code. First you need to add the code into the **trunk** and after that you need to **create version tag** in order to release version.

https://developer.wordpress.org/plugins/wordpress-org/how-your-readme-txt-works/

Using Hooks (Actions and Filters)

Actions and Filters (aka Hooks)

Hooks are defined at different places in the WordPress core, theme or plugins and allow us to **execute tasks** or **modify values of variables** at some point of time during the page rendering lifecycle when they are invoked.

There are two types of hooks: **Actions** and **Filters**

Actions are triggered on specific events that take place in WordPress (either in the core, themes or plugins), such as publishing a post and are used to perform specific task when the event occurs.

```
do_action( 'save_post', int $post_ID, WP_Post $post, bool $update )
```

Filters are similar to actions but they are used only to **modify specific variable value**.

```
$favourite_team = apply_filters( 'favourite_team', 'Manchester United')
```

Example 1 (Actions)

Notify site admin when user signed into the site

How/where it is defined? https://developer.wordpress.org/reference/functions/wp_signon/

```
/**
* Fires after the user has successfully logged in.
* @param string  $user_login Username.
* @param WP_User $user       WP_User object of the logged-in user.
*/
do_action( 'wp_login', $user->user_login, $user );
```

Hooking into...

```
function dg_login_notification( $user_login, \WP_User $user ) {
  $subject = __( 'User login' );
  $message = sprintf( __( '%s logged into the site.' ), $user_login );
  $email   = 'info@mycompany.com';
  wp_mail( $email, $subject, $message );
}
add_action( 'wp_login', 'dg_login_notification', 100, 2 );
```

Example 2 (Filters)

Modify post content without editing template files

How/where it is defined?

```
function the_content(...) {
  /// ...
  $content = apply_filters( 'the_content', $content );
  /// ...
  echo $content;
}
```

Hooking into...

```
function dg_the_content( $content ) {
  $content .= '<p>' . __( 'This is the last paragraph' ) . '</p>';
  return $content;
}
add_filter( 'the_content', 'dg_the_content' );
```

https://developer.wordpress.org/reference/hooks/the_content/ (hook)
https://developer.wordpress.org/reference/functions/the_content/ (function)

List of all actions and filters

Actions Documentation
https://codex.wordpress.org/Plugin_API/Action_Reference/

Filters Documentation
https://codex.wordpress.org/Plugin_API/Filter_Reference

Post Types, Taxonomies, Metadata

Post Types

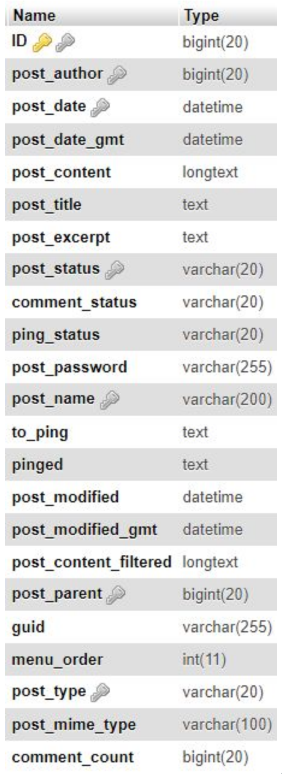
A way to organize your data, eg. Similarly to Pages, Posts for our own purposes we can register Books post type which will allow us to store Books in the database

All the posts are stored in **wp_posts** table regardless of the type

- The type of the post is identified by the **post_type** column.
- **Page** and **Post** are native WordPress post types (and some other that are private)

Post Types can be registered based on your needs
https://codex.wordpress.org/Function_Reference/register_post_type

wp_posts

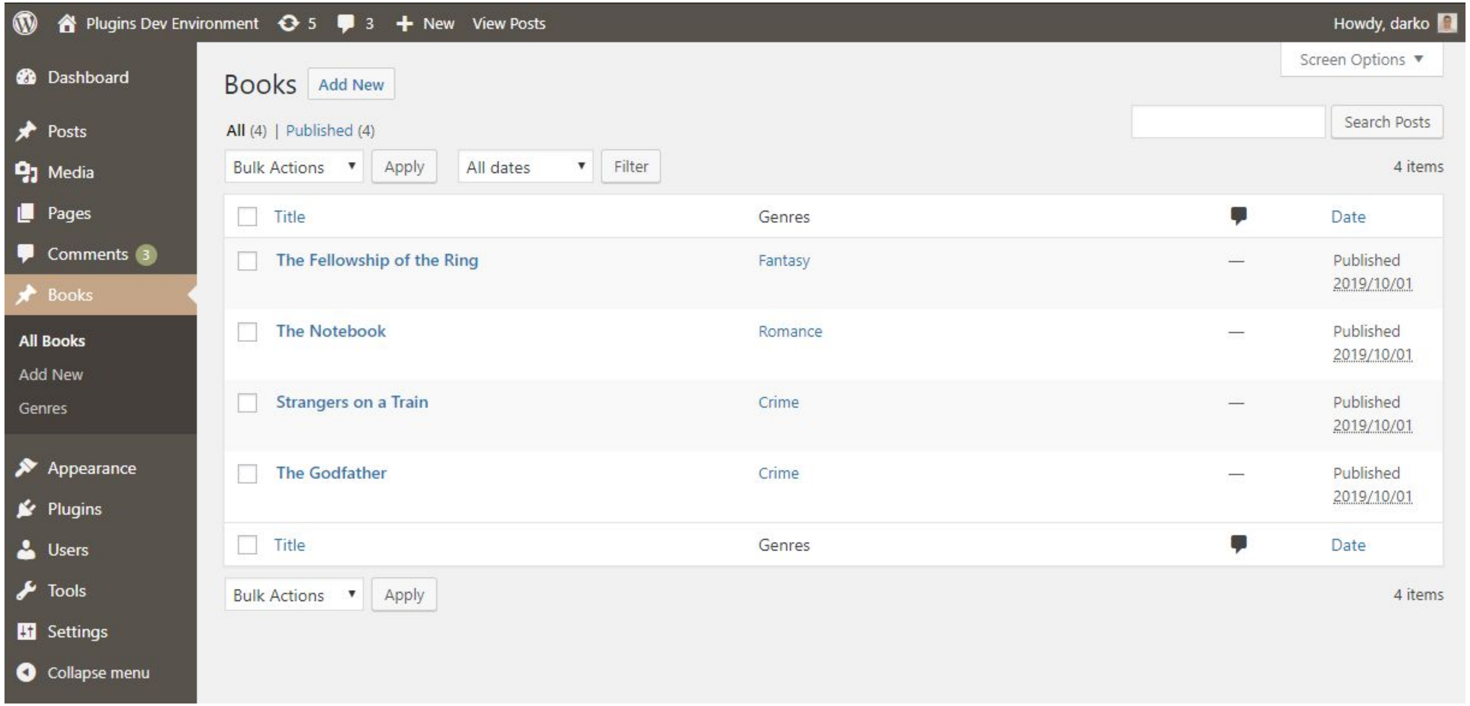


Name	Type
ID	bigint(20)
post_author	bigint(20)
post_date	datetime
post_date_gmt	datetime
post_content	longtext
post_title	text
post_excerpt	text
post_status	varchar(20)
comment_status	varchar(20)
ping_status	varchar(20)
post_password	varchar(255)
post_name	varchar(200)
to_ping	text
pinged	text
post_modified	datetime
post_modified_gmt	datetime
post_content_filtered	longtext
post_parent	bigint(20)
guid	varchar(255)
menu_order	int(11)
post_type	varchar(20)
post_mime_type	varchar(100)
comment_count	bigint(20)

```
add_action( 'init', 'bl_register_books' );

function bl_register_books() {
  $labels = array(
      'name'                => _x( 'Books', 'post type general name', 'books-library' ),
      'singular_name'       => _x( 'Book', 'post type singular name', 'books-library' ),
      'menu_name'           => _x( 'Books', 'admin menu', 'books-library' ),
      'name_admin_bar'      => _x( 'Book', 'add new on admin bar', 'books-library' ),
      'add_new'             => _x( 'Add New', 'book', 'books-library' ),
      'add_new_item'        => __( 'Add New Book', 'books-library' ),
      'new_item'            => __( 'New Book', 'books-library' ),
      'edit_item'           => __( 'Edit Book', 'books-library' ),
      'view_item'           => __( 'View Book', 'books-library' ),
      'all_items'           => __( 'All Books', 'books-library' ),
  );
  $args = array(
      'labels'              => $labels,
      'public'              => true,
      'publicly_queryable'  => true,
      'show_ui'             => true,
      'show_in_menu'        => true,
      'query_var'           => true,
      'rewrite'             => array( 'slug' => 'book' ),
      'capability_type'     => 'post',
      'has_archive'         => true,
      'hierarchical'        => false,
      'menu_position'       => null,
      'supports'            => array( 'title', 'editor', 'thumbnail', 'excerpt', 'comments' )
  );
  register_post_type( 'book', $args );
}
```

Example of the Books post type



	Title	Genres		Date
☐	The Fellowship of the Ring	Fantasy	—	Published 2019/10/01
☐	The Notebook	Romance	—	Published 2019/10/01
☐	Strangers on a Train	Crime	—	Published 2019/10/01
☐	The Godfather	Crime	—	Published 2019/10/01
☐	Title	Genres		Date

Taxonomies

A **way to group the data**, they can be registered like the post types with some differences.

By default WordPress registers **category(Categories)** and **post_tag(Tags)** taxonomies

- Items in the specific taxonomy are called **terms**. Eg. **Crime** is term in the **Genres** taxonomy
- Once the taxonomy is registered successfully it will appear in the Post Type submenu and the term editor will be available out of the box. (No need to code the functionality for creating or deleting terms or assigning posts to specific terms in the taxonomy.)
  https://codex.wordpress.org/Function_Reference/register_taxonomy

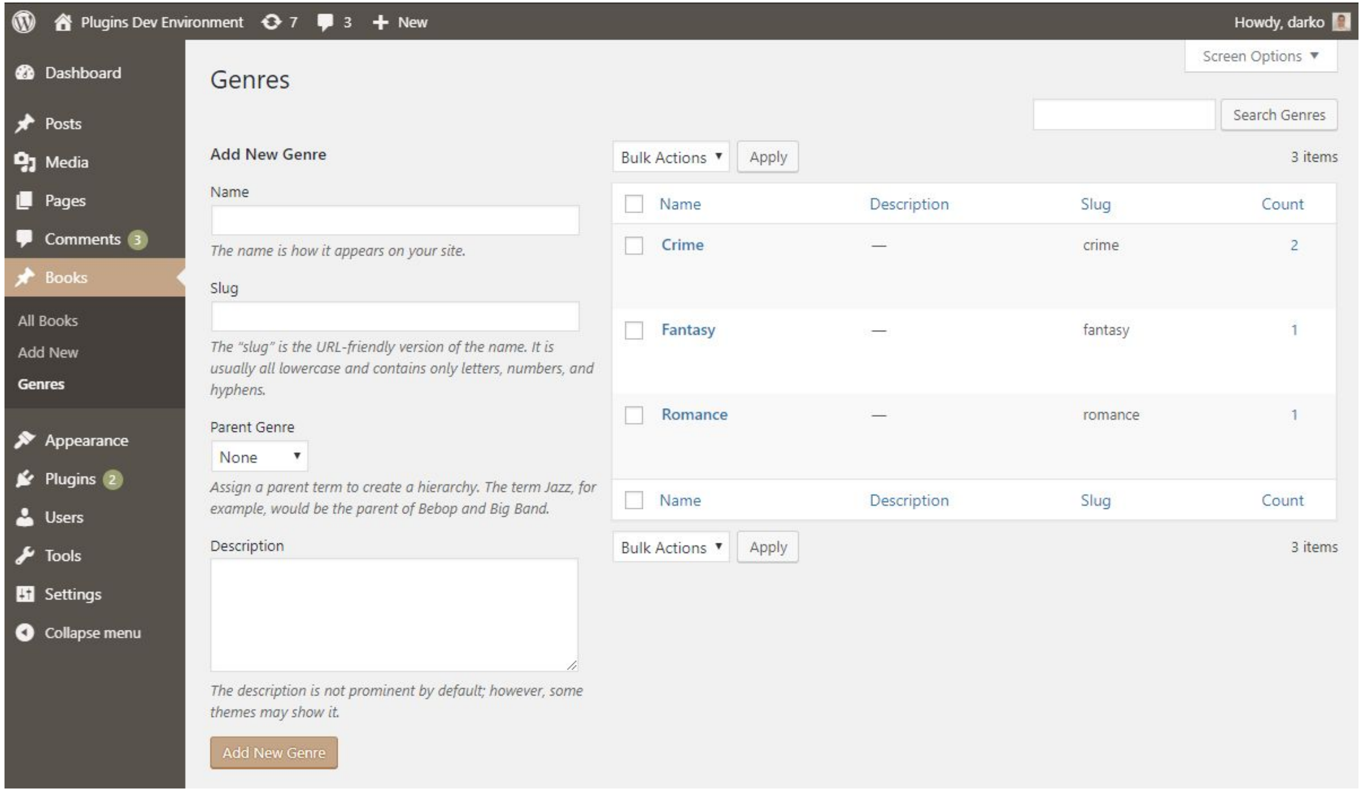
Example: **Books** can be grouped by **Genre**. In the next example we will see how we can register the **Genre** taxonomy to the **Books** post type.

Register the Genre Taxonomy

```
add_action( 'init', 'bl_register_genres', 0 );

function bl_register_genres() {
  $labels = array(
      'name'              => _x( 'Genres', 'taxonomy general name', 'books-library' ),
      'singular_name'     => _x( 'Genre', 'taxonomy singular name', 'books-library' ),
      'search_items'      => __( 'Search Genres', 'books-library' ),
      'all_items'         => __( 'All Genres', 'books-library' ),
      'parent_item'       => __( 'Parent Genre', 'books-library' ),
      'parent_item_colon' => __( 'Parent Genre:', 'books-library' ),
      'edit_item'         => __( 'Edit Genre', 'books-library' ),
      'update_item'       => __( 'Update Genre', 'books-library' ),
      'add_new_item'      => __( 'Add New Genre', 'books-library' ),
      'new_item_name'     => __( 'New Genre Name', 'books-library' ),
      'menu_name'         => __( 'Genre', 'books-library' ),
  );
  $args = array(
      'hierarchical'      => true,
      'labels'            => $labels,
      'show_ui'           => true,
      'show_admin_column' => true,
      'query_var'         => true,
      'rewrite'           => array( 'slug' => 'genre' ),
  );
  register_taxonomy( 'genre', array( 'book' ), $args );
}
```

Example of the Genres editor



Genres

Add New Genre

Name

The name is how it appears on your site.

Slug

The "slug" is the URL-friendly version of the name. It is usually all lowercase and contains only letters, numbers, and hyphens.

Parent Genre

None

Assign a parent term to create a hierarchy. The term Jazz, for example, would be the parent of Bebop and Big Band.

Description

The description is not prominent by default; however, some themes may show it.

Name	Description	Slug	Count
Crime	—	crime	2
Fantasy	—	fantasy	1
Romance	—	romance	1

Posts Metadata

What is metadata and how it works?

Metadata in WordPress is way to store additional information about the **posts** that are stored in **wp_posts** table. For example if we have post of the type 'book' (wp_posts.post_type=book) we can add meta data like number of pages, isbn, etc.

Where is the post metadata stored?

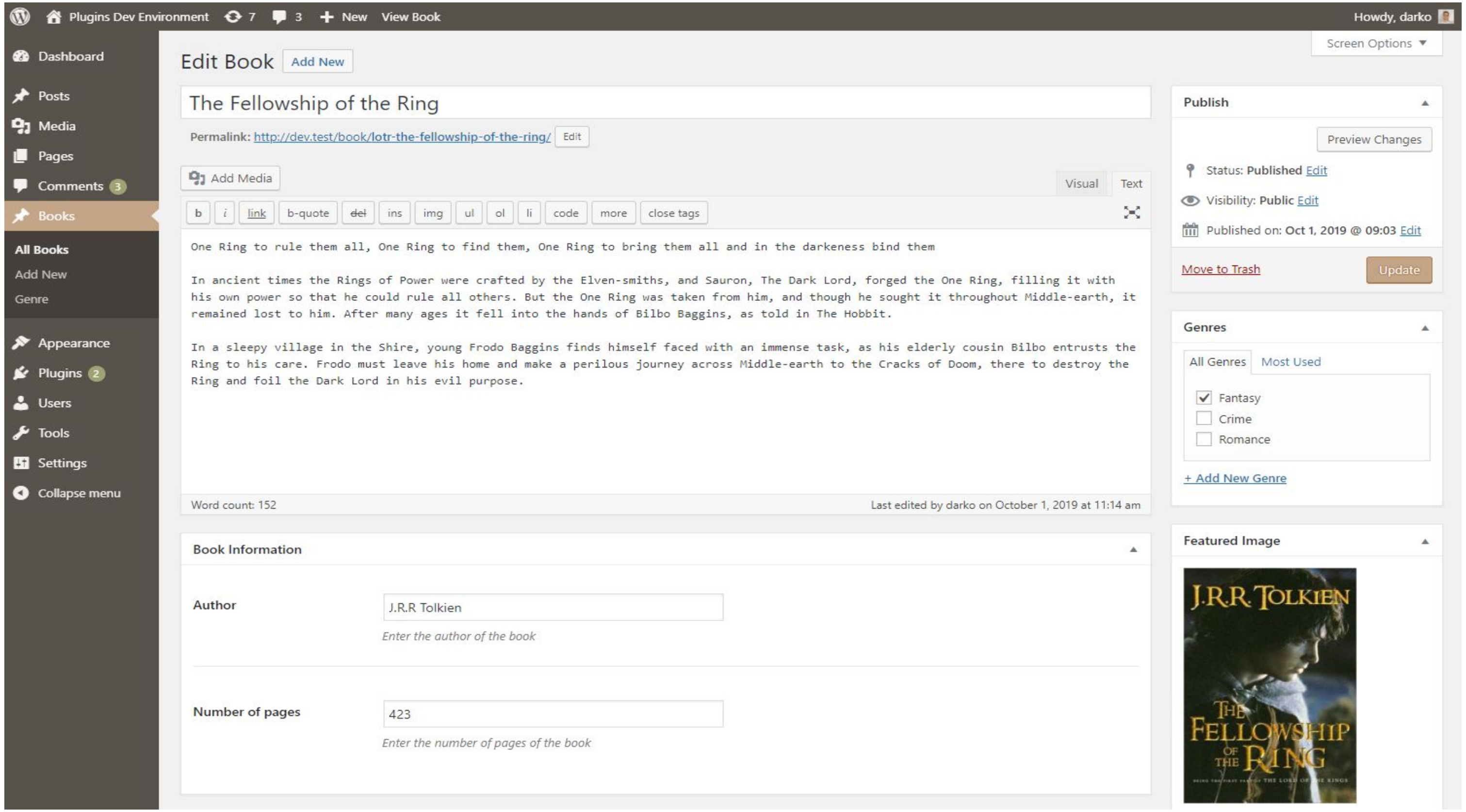
The metadata is stored in **wp_postmeta(meta_id, post_id, meta_key, meta_value)** table

How to manage the post metadata?

The metadata is managed in the editor. There are multiple ways to add metaboxes, including:

- Official https://developer.wordpress.org/plugins/metadata/custom-meta-boxes/
- Carbon Fields / https://github.com/htmlburger/carbon-fields
- CMB2 / https://github.com/CMB2/CMB2
- … a lot others like ACF, etc.

Example using CMB2 Framework



Edit Book Add New

The Fellowship of the Ring

Permalink: http://dev.test/book/lotr-the-fellowship-of-the-ring/ Edit

```
One Ring to rule them all, One Ring to find them, One Ring to bring them all and in the darkeness bind them

In ancient times the Rings of Power were crafted by the Elven-smiths, and Sauron, The Dark Lord, forged the One Ring, filling it with
his own power so that he could rule all others. But the One Ring was taken from him, and though he sought it throughout Middle-earth, it
remained lost to him. After many ages it fell into the hands of Bilbo Baggins, as told in The Hobbit.

In a sleepy village in the Shire, young Frodo Baggins finds himself faced with an immense task, as his elderly cousin Bilbo entrusts the
Ring to his care. Frodo must leave his home and make a perilous journey across Middle-earth to the Cracks of Doom, there to destroy the
Ring and foil the Dark Lord in his evil purpose.
```

Word count: 152 Last edited by darko on October 1, 2019 at 11:14 am

Book Information

Author	J.R.R Tolkien
	Enter the author of the book
Number of pages	423
	Enter the number of pages of the book

Publish

Status: Published Edit

Visibility: Public Edit

Published on: Oct 1, 2019 @ 09:03 Edit

Genres

- [✓] Fantasy
- [ ] Crime
- [ ] Romance

Featured Image

Including CSS / JS Files the right way

How to include css/js files from plugin?

How it works

Multiple plugins you have may use **jQuery** and other shared scripts. If each plugin linked to these assets separately, chaos would ensue and all your JavaScript could stop working.

Using the wp_enqueue_script/wp_enqueue_style to register styles we are telling WordPress about the assets we want to add and it will take care of actually linking to them in the header and footer.

Example

```
add_action( 'wp_enqueue_scripts', 'bl_enqueue_scripts', 15 );
function bl_enqueue_scripts() {
   wp_enqueue_style( 'books-library', BL_URI . 'assets/style.css', null, BL_VERSION );
   wp_enqueue_script( 'books-library', BL_URI . 'assets/script.js', array( 'jquery' ), BL_VERSION, true );
}
```

Shortcodes

Shortcodes

A **shortcode** is WordPress specific code that in background generates dynamic content.

By default WordPress registers the **[gallery]** shortcode that can be used to display the images uploaded to the post

In the following example we will see how to define shortcode called **[books_library]** that displays list of books.

```
[books_library posts_per_page=6 genre=action]
```

Creating our first shortcode [books_library]

```php
add_shortcode( 'books_library', 'books_library' );

function books_library( $atts ) {
  // Setup the default parameters
  $atts = shortcode_atts( array(
     'posts_per_page' => 5,
  ), $atts );
  // Retrieve the Book posts
  $books = get_posts( array(
     'posts_per_page' => $atts['posts_per_page'],
     'post_type'      => 'book',
     'post_status'    => 'publish',
     'orderby'        => 'date',
     'order'          => 'DESC',
  ) );
  // Output the books
  if ( count( $books ) > 0 ) {
     $output = '<ul>';
     foreach ( $books as $book ) {
        $output .= '<li><a href="' . get_permalink( $book ) . '">' . $book->post_title . '</a>';
     }
     $output .= '</ul>';
  } else {
     $output = __( 'No books found', 'books-library' );
  }

  return $output;
}
```

Internationalization (I18n)

Translating your plugin

1. Define Text Domain

```
/*
Plugin Name: Books Library
Plugin URI: https://thepluginurl.com
Description: Organizes your eBooks in WordPress
Author: Darko Gjorgjijoski
Version: 1.0.0
Author URI: https://darkog.com/
Text Domain: books-library
Domain Path: /languages
*/
```

2. When printing, do it as follows

```
echo '<h2>' . __('My Title', 'books-library') . '</h2>';
```

Instead of

```
echo '<h2>My Title</h2>';
```

3. Prepare your .pot

Pot files are used to store the words that can be translated. To generate pot file you can use Loco Translate or POEdit https://codex.wordpress.org/User:Skippy/Creating_POT_Files

4. Reference

For reference always check the codex https://codex.wordpress.org/I18n_for_WordPress_Developers

GOOD
CODING PRACTICES

Don’t just do it, Do it right!

1. **Do not trust the user input. Always: validate, sanitize, escape!**

   WordPress comes with pre-made functions for those purposes:
   https://developer.wordpress.org/themes/theme-security/data-sanitization-escaping/

2. **Make use the of the WordPress built-in apis for better compatibility.**

   - **HTTP API** instead of plain CURL requests with `curl_init()` function https://developer.wordpress.org/plugins/http-api/
   - **Object Cache** for caching (useful when using Redis in combination with the Redis plugin - https://codex.wordpress.org/Class_Reference/WP_Object_Cache
   - **Transients API** for persistent caching https://codex.wordpress.org/Transients_API
   - **Settings API** for creating admin screens https://codex.wordpress.org/Settings_API
   - **Options API** for storing key/value options persistently in the db https://codex.wordpress.org/Options_API

Complete list of all native APIs: https://codex.wordpress.org/WordPress_API%27s

Thanks for your attention!
Any questions?

Plugin and presentation available on
dg.mk/wcskp2019