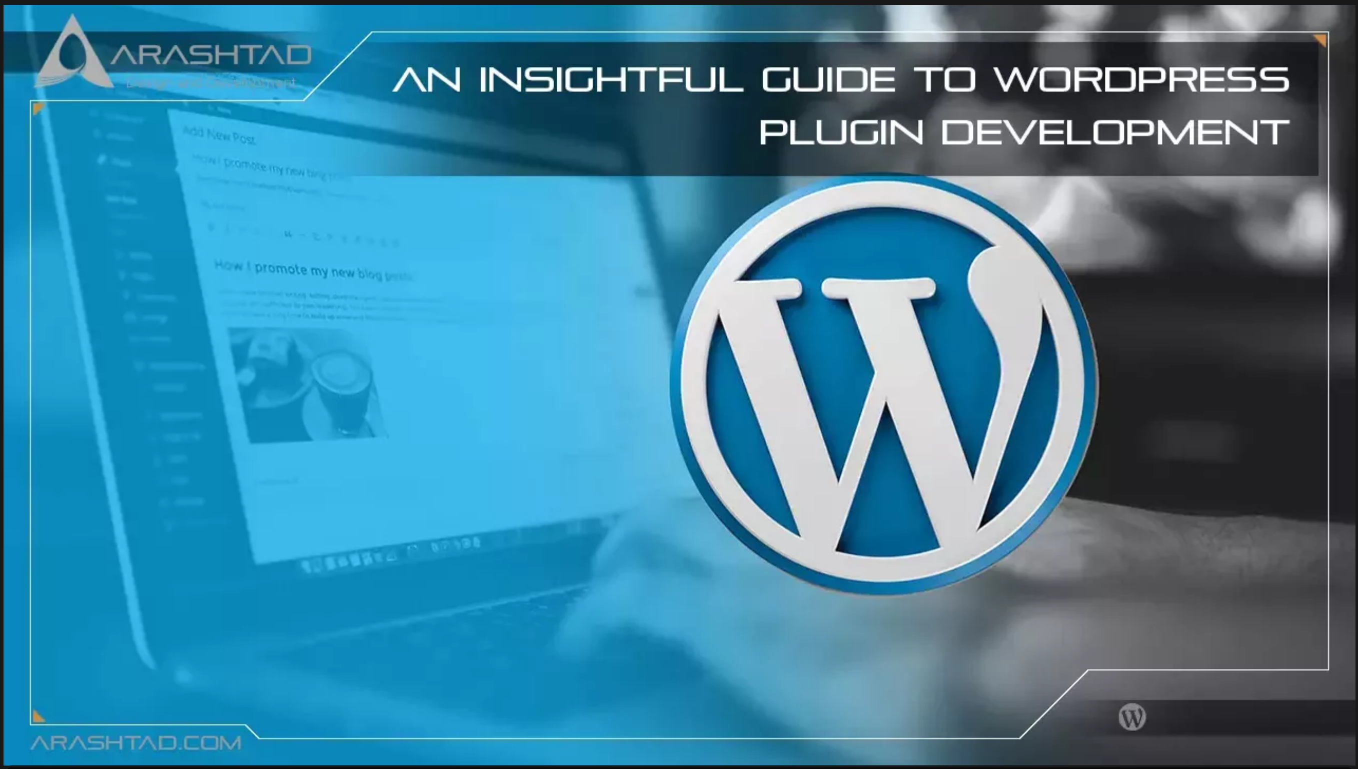
An Insightful Guide to WordPress Plugin Development

No comments



W3Techs estimates that WordPress will power 39.1% of all websites by 2020. This is because WordPress has so many built-in functionalities and is highly customizable through plugins. Plugins are an integral component of the WordPress platform, allowing you to easily extend functionality that goes beyond the core without having to modify it. Developers can list their plugins on the web for others to use, but these plugins may not meet your specific needs. The official WordPress plugin directory contains almost sixty thousand plugins, so you should check if one matches your requirements before you start designing one.

You might want to consider creating a custom WordPress plugin if you don't find one that fits your needs. WordPress is powered by scripting and query languages like PHP and SQL in the background. You need expertise in these technologies and deployment experience to create a WordPress plugin. This article will guide you on how to develop a WordPress plugin.

Basic Concepts of WordPress Plugin Development

The core files of WordPress are overridden when it gets updated to a new version. As a result, you may need to modify the core to add custom functionality to a WordPress site. As a result, one of WordPress' core development concepts is to add or modify functionality using plugins. Plugins are essential functions in PHP files, as PHP is the main scripting language behind WordPress. A plugin can also include hooks, shortcodes, and widgets. These three main elements make up a WordPress plugin.

Hooks

Without changing WordPress core files, hooks allow you to manipulate a process at a particular point in the process. Therefore, hooks allow your plugin to interact with WordPress core. Code snippets or functions can be associated with hooks and executed at various intervals. Hooks can be applied to either an action (action hook) or a filter (filter hook).

Action

The action hook allows you to add a WordPress process. A common example of a WordPress action is creating, reading, or saving a post. You can use the add_action() function to work with it. PHP functions or code snippets can be associated with actions. Even your own action can be created and associated with code. An action enables you to add functionality to a plugin. You can hook on to action and run custom functions. For instance, you can associate your function, custom_function(), with the action of saving a post. use the add_action() function.

Filter

Filters are hooks that modify processes. They allow you to manipulate data without altering its source of it. Applying a filter hook is as simple as using the apply_filters() function. It requires two arguments: the filter's name and the value to be filtered.

Shortcodes

Plugins use shortcodes to communicate with WordPress themes and display information to users. Shortcodes can be used to insert HTML elements into posts and pages.

Widgets

Plugin widgets allow developers to display the content of their plugins to end-users in another way. To create a widget for your plugin, you need to extend the WP_widget class in PHP.

Key Steps of WordPress Plugin Development

1. Define the Requirements

The first step in WordPress plugin development is clearly defining your development needs. Before you start, ensure that you have a clear idea of the objective of the Plugin. When you have an accurate picture of the issue to resolve, you are able to execute your idea into an efficient plugin. There are many factors that you can consider in this step. What are the features of this Plugin? How are you going to customize it? What will the design look like? Make sure to answer these questions because this step is linked to all the other steps of the process.

2. Create a WordPress Plugin Directory Structure

The default WP directory for storing plugin code in the back end is /wp-content/plugins/. The way you structure your Plugin will depend on how complex it is. The directory's name is the same as the Plugin's name, in lowercase with dashes instead of spaces. A simple plugin that serves a small purpose should have a single PHP file containing all the code (/wp-content/plugins/my-plugin/my-plugin.php).

To organize a plugin with many assets, you can organize its code and PHP files according to its function. For CSS and JavaScript files, create a folder called assets. for template and widget files, create a folder called i18n. You can create an MVC view with a model, view, and controller directory in the my-plugin directory for more complex plugins. This helps to debug later.

3. Configure your Plugin

In the WordPress codex, you can find a sample file header with the contents of a sample plugin directory. After creating your plugin directory and adding files within it, you need to add the file header. The file header is a PHP comment block that contains information about the Plugin. The Plugin will appear in your WordPress admin after you add the file header.

4. Add Functionality to your Plugin

Having created an empty plugin, you haven't accomplished anything yet. You need to add functionality to it now. Use the WordPress plugin handbook as a guide. Here you are bringing your idea to life.

5. Package your Plugin

A developer may work on a WordPress plugin in a development environment. to transfer the plugin to your production site, you would need to compress the plugin directory in WordPress Admin and upload the zipped plugin file there.

WordPress Plugin Development Services By Arashtad

CMS Development

Whether you are looking for a custom CMS or you want to add a custom feature as a plugin for your existing WordPress or Joomla website, you can expect us to deliver a comprehensive service that covers any type of task you may need. We create custom CMS with different technology stacks according to your project requirements and your ideas or develop your WordPress and Joomla web applications in a way that they are fully customized and cover your needs.

Product Types

Custom CMS

WordPress Plugin Development

Joomla Extension Development

WordPress and Joomla Development

We have over a decade of experience in developing websites, themes/templates, and plugins/extensions for both WordPress and Joomla content management systems. Also, we have a number of products that we have shared with the WordPress and Joomla community that you can find on our services pages.

Product Types

WordPress Websites

WordPress Plugins

WordPress Themes

Joomla Websites

Joomla Extensions

Joomla Templates

Final Thoughts on WordPress Plugin Development

During this tutorial on WordPress plugin development, we explored how to build your own plugin from scratch. Even though it requires technical expertise and patience, you can accomplish your goal of launching a plugin with the right skills, knowledge, and willingness to do the work. If you don't have any fundamental skills and don't have the time to learn the basics yourself, you can consider hiring a professional WordPress developer.

Join Arashtad Community

Follow Arashtad on Social Media

We provide variety of content, products, services, tools, tutorials, etc. Each social profile according to its features and purpose can cover only one or few parts of our updates. We can not upload our videos on SoundCloud or provide our eBooks on Youtube. So, for not missing any high quality original content that we provide on various social networks, make sure you follow us on as many social networks as you're active in. You can find out Arashtad's profiles on different social media services.

Get Even Closer!

Did you know that only one universal Arashtad account makes you able to log into all Arashtad network at once? Creating an Arashtad account is free. Why not to try it? Also, we have regular updates on our newsletter and feed entries. Use all these benefitial free features to get more involved with the community and enjoy the many products, services, tools, tutorials, etc. that we provide frequently.

SIGN UP NEWSLETTER RSS FEED