1

An Introduction to Plugins

WHAT'S IN THIS CHAPTER?

- Understanding a plugin
- Using available WordPress APIs
- Loading order of plugins
- Finding examples of popular plugins
- Determining the separation of plugin and theme functionality
- Managing and installing plugins
- Understanding types of WordPress plugins

WordPress is one of the most popular open source content management systems available today. One of the primary reasons WordPress is so popular is the ease with which you can customize WordPress through plugins. WordPress has an amazing framework in place giving plugin developers the tools needed to extend WordPress in any way imaginable.

Understanding how plugins work, and the tools available in WordPress, is critical knowledge when developing professional WordPress plugins.

WHAT IS A PLUGIN?

A plugin in WordPress is a PHP script that extends or alters the core functionality of WordPress. Quite simply plugins are files installed in WordPress to add a feature, or set of features, to WordPress. Plugins can range in complexity from a simple social networking plugin to an extremely elaborate e-commerce package. There is no limit to what a plugin can do in WordPress; because of this there is no shortage of plugins available for download.

How Plugins Interact with WordPress

WordPress features many different APIs for use in your plugin. Each API, or application programming interface, helps interact with WordPress in a different way. Following is a list of the main available APIs in WordPress and their function:

- **Plugin** — Provides a set of hooks that enable plugins access to specific parts of WordPress. WordPress contains two different types of hooks: Actions and Filters. The Action hook enables you to trigger custom plugin code at specific points during execution. For example, you can trigger a custom function to run after a user registers a user account in WordPress. The Filter hook to modifies text before adding or after retrieving from the database.
- **Widgets** — Create and manage widgets in your plugin. Widgets appear under the Appearance ➪ Widgets screen and are available to add to any registered sidebar in your theme. The API enables multiple instances of the same widget to be used throughout your sidebars.
- **Shortcode** — Adds shortcode support to your plugin. A shortcode is a simple hook that enables you to call a PHP function by adding something such as [shortcode] to a post or page.
- **HTTP** — Sends HTTP requests from your plugin. This API retrieves content from an external URL or for submitting content to a URL. Currently you have five different ways to send an HTTP request. This API standardizes that process and tests each method prior to executing. Based on your server configuration, the API will use the appropriate method and make the request.
- **Settings** — Inserts settings or a settings section for your plugin. The primary advantage to using the Settings API is security. All settings data is scrubbed, so you do not need to worry about cross site request forgery (CSRF) and cross site scripting (XSS) attacks when saving plugin settings.
- **Options** — Stores and retrieves options in your plugin. This API features the capability to create new options, update existing options, delete options, and retrieve any option already defined.
- **Dashboard Widgets** — Creates admin dashboard widgets. Widgets automatically appear on the Dashboard of WordPress and contain all standard customization features including minimize, drag/drop, and screen options for hiding.
- **Rewrite** — Creates custom rewrite rules in your plugin. This API enables you to add static end-points (`/custom-page/`), structure tags (`%postname%`), and add additional feed links (`/feed/json/`).
- **Transients** — Creates temporary options (cached data) in your plugins. This API is similar to the Options API, but all options are saved with an expiration time.
- **Database** — Accesses the WordPress database. This includes creating, updating, deleting, and retrieving database records for use in your plugins.

WordPress also features pluggable functions. These functions enable you to override specific core functions in a plugin. For example, the `wp_mail()` function is a pluggable function. You can easily define this function in your plugin and send email using SMTP rather than the default method. All pluggable functions are defined in the `/wp-includes/pluggable.php` Core WordPress file.

You can use some predefined functions during specific plugin tasks, such as when a plugin is activated or deactivated and even when a plugin is uninstalled. Chapter 2, "Plugin Foundation," covers these functions in detail.

When Are Plugins Loaded?

Plugins are loaded early in the process when a WordPress powered web page is called. Figure 1-1 shows a diagram of the standard loading process when loading a page in WordPress:

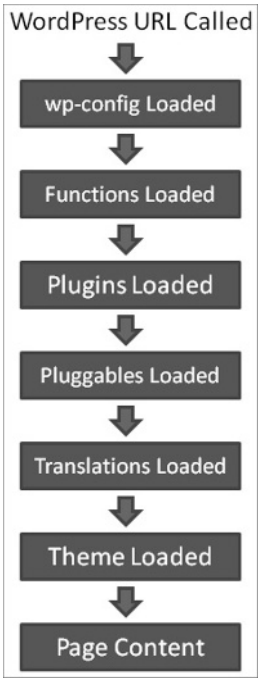

FIGURE 1-1

Figure 1-1 illustrates the standard process when loading a page in WordPress. The flow changes slightly when loading an admin page. The differences are minor and primarily concern what theme is loaded: admin theme versus your web site theme.

AVAILABLE PLUGINS

When researching available plugins you need to know where to find WordPress plugins. You can download plugins anywhere on the Internet, but this isn't always a good idea.

As with any software, downloading plugins from an untrusted source could lead to malware injected and compromised plugin files. It's best to download plugins only from trusted web sites and official sources such as the official Plugin Directory.

Official Plugin Directory

The first place to start when researching available WordPress plugins is the official Plugin Directory at WordPress.org. The Plugin Directory is located at `http://wordpress.org/extend/plugins/`. With more than 10,000 plugins available and well over 100 million plugin downloads, it's easy to see the vital role plugins play in every WordPress web site. All plugins available in the Plugin Directory are 100% GPL and free to use for personal or commercial use.

Popular Plugin Examples

Take a look at the five most downloaded WordPress plugins available to get a sense of their diversity:

- **All in One SEO Pack** — Adds advanced search engine optimization functionality to WordPress. Features include custom meta data for all content, canonical URLs, custom post type support, and more!
  - `http://wordpress.org/extend/plugins/all-in-one-seo-pack/`
- **Google XML Sitemaps** — Generates an XML sitemap of all content for submission to the popular search engines such as Google, Bing, and Ask.com.
  - `http://wordpress.org/extend/plugins/google-sitemap-generator/`
- **Akismet** — A popular comment spam filter for WordPress. Checks all comments against the Akismet web service to verify whether the comment is spam.
  - `http://wordpress.org/extend/plugins/akismet/`
- **NextGEN Gallery** — Adds advanced image gallery support to WordPress. You can easily create and manage image galleries and slideshows. Galleries can be embedded in posts or pages.
  - `http://wordpress.org/extend/plugins/nextgen-gallery/`
- **Contact Form 7** — Adds a contact form to any post or page in WordPress. Supports multiple contact forms, Akismet spam filtering, and CAPTCHA.
  - `http://wordpress.org/extend/plugins/contact-form-7/`

As you can see, the preceding plugins can handle any task. The features added by these plugins are universal and features that most web sites on the Internet should have.

Popular Plugin Tags

Now you will look at some popular tags for plugins. Plugin tags are just like blog post tags, simple keywords that describe a plugin in the Plugin Directory. This makes it easy to search for existing plugins by tag. Following are popular examples:

- **Twitter** — Everyone loves Twitter for micro-blogging and sharing links. You can find an abundance of Twitter-related plugins for WordPress.
  - `http://wordpress.org/extend/plugins/tags/twitter`
- **Google** — With so many different services and APIs, Google is a popular plugin tag. Everything from Google ads to Google maps have been integrated into a WordPress plugin.
  - `http://wordpress.org/extend/plugins/tags/google`
- **Widget** — Most plugins that include a widget also use the widget tag. This is great for viewing the many different types of widgets available for WordPress.
  - `http://wordpress.org/extend/plugins/tags/widget`

Viewing popular plugin tags is a great way to get inspiration when developing new plugins for WordPress.

ADVANTAGES OF PLUGINS

WordPress offers many advantages to using plugins. You need to understand the advantages to building plugins to truly understand why you should build plugins. This can also help when determining the need for a specific plugin in WordPress.

Not Modifying Core

One of the main advantages to plugins is the ability to modify the behavior of WordPress without modifying any core files. Core files refer to any file that is a part of the default WordPress installation.

Hacking core files can make it difficult to update WordPress when a new version is released. If you made any modifications to a core file, that modification would be overwritten when the update occurs. Keeping WordPress up to date with the latest version is essential in keeping your web site secure.

Modifying core files can also lead to an unstable web site. Different areas of WordPress rely on other areas to function as expected. If you modify a core file and it no longer works as expected, it can cause instability and quite possibly break a completely unrelated feature in WordPress.

Why Reinvent the Wheel

Another advantage to building plugins is the structure that already exists for your plugin. Many of the common features have already been developed and are ready for use in your plugin. For example, you can take advantage of the built-in user roles in WordPress. Using the user roles you can easily restrict your code to execute only if a user is an administrator. Look at an example:

```
<?php
if ( current_user_can( 'manage_options' ) ) {
  //any code entered here will only be executed IF
  //user is an administrator
}
?>
```

As you can see it's easy to verify a user has proper permissions prior to executing any code in your plugin. You learn about user accounts and roles in Chapter 8, "Users."

As another example, look at sending an email in WordPress. Sure you could create a new function in your plugin to send email, but why? WordPress has a handy function called `wp_mail()` for sending email. Look at an example:

```
<?php
$email_to = 'you@example.com';
$email_subject = 'Plugin email example';
$email_message = 'How do you like my new plugin?';

wp_mail( $email_to, $email_subject, $email_message );
?>
```

As you can see sending an email in WordPress couldn't be easier. Unless your plugin needs some customized emailing functionality, you don't need to re-create this function from scratch. Using this function also ensures the widest adoption for sending emails from WordPress because you use the built-in function.

Using the available built-in features of WordPress can greatly reduce the time to develop a plugin. Another advantage to not reinventing the wheel is that this approach more often than not will allow for your plugins to work across a greater number of servers and setups, thereby maximizing compatibility. Don't reinvent the wheel with features that already exist in WordPress.

Separating Plugins and Themes

A plugin can take control of the rendering process; therefore, the plugin can become a "theme." Similarly a theme can have plugin functionality included. Because of this the difference between the two can sometimes become blurred, so why not just include your plugin code directly in a theme? This is a common question and one that can have a few different answers.

Should themes include plugin functionality? The short answer is no. The primary reason for this is because plugins are meant to add features and functionality to WordPress, regardless of the theme used. This creates a nice separation between your web site design and the functionality of your web site. The reason this separation is needed is so your theme is not directly tied to the functionality required. WordPress is built so that you can easily change your design, or theme, at any point with just a couple clicks. If all plugin functionality existed in your theme, and you switched themes, you will have lost all that functionality you required.

There is also a strong argument that certain features should be included in a theme. A common feature most themes include is breadcrumb navigation. This feature could certainly exist in a plugin, but being a navigation-centric feature it makes sense to include this in the theme. Search engine optimization features are also a common feature found in themes today.

Easy Updates

WordPress makes it easy to update a plugin to the latest version. Every plugin installed from the WordPress.org Plugin Directory alerts you when a new version of the plugin has been released. Updating the plugin is as simple as clicking the update notification listed just below the plugin details on the Plugin screen.

Plugins not installed from the Plugin Directory can also be updated using the auto-update functionality of WordPress. The plugin author must define where WordPress can download the latest version, and it will take care of the rest. If the plugin author doesn't define this location, you must manually update the plugin.

Keeping plugins updated is an important part in keeping your web site free from security vulnerabilities and bugs.

Easier to Share and Reuse

Plugins are easy to share with others. It's much easier to share a plugin than tell someone to modify specific lines of code in your theme or WordPress. Using plugins also makes it easy to use the same functionality across multiple sites. If you find a group of plugins that you like, you can easily install them on every WordPress web site you create.

Plugin Sandbox

When you activate a broken plugin in WordPress, it won't break your site. If the plugin triggers a fatal error, WordPress automatically deactivates the plugin before it has a chance to. This fail-safe feature makes it less risky when activating and testing out new plugins. Even if the plugin does cause a white screen of death (error message), you can easily rename the plugin folder, and WordPress deactivates the plugin. This makes it impossible for a rogue plugin to lock you out of your own site because of an error.

On the other hand, if you were to hack the WordPress core, you can most certainly cause fatal errors that will crash your web site. This can also include making unrecoverable damage to WordPress.

Plugin Community

A huge community is centered around plugin development, sharing knowledge and code, and creating wonderful plugins. Getting involved in the community is a great way to take your plugin development skills to the next level. Chapter 18, "The Developer Toolbox," covers many of these resources.

INSTALLING AND MANAGING PLUGINS

All plugin management in WordPress happens under the Plugins screen in the WordPress Dashboard, as shown in Figure 1-2.

The menu shown in Figure 1-2 is available only to administrators in WordPress, so nonadministrators cannot see this menu. If you use the Multisite feature of WordPress, the Plugins menu is hidden by default. You need to enable the menu under Network Admin ➪ Settings.

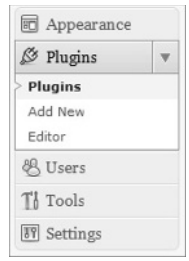

FIGURE 1-2

Installing a Plugin

WordPress features three different methods for installing a new plugin. Your server setup dictates which method is the best to use.

The first method uses the built-in auto installer. This method enables you to search the Plugin Directory on WordPress.org directly from the admin dashboard of your WordPress web site. After you find a plugin to install, simply click the Install link, and the plugin automatically downloads and installs.

The second method uses the zip uploader. Zipped plugin files can be uploaded, extracted, and installed by WordPress. To use this method click the Upload link at the top of the Install Plugins

page. Click the Browser button and select the plugin zip file you want to install. After you select the plugin, click the Install Now button, as shown in Figure 1-3.

FIGURE 1-3

The third and final method to install a plugin in WordPress uses File Transfer Protocol (FTP). Using FTP is simply connecting to your web server using an FTP client and manually uploading the plugin to your WordPress installation. To use this method upload the uncompressed plugin folder or file to the `wp-content/plugins` directory on your web server.

Managing Plugins

After you install a plugin in WordPress, you can manage it, along with all other plugins, under the Plugins ➪ Plugins screen. Here you can find a list of all plugins, active or not, available in your WordPress installation. You can easily activate, deactivate, edit, update, and delete plugins from this screen.

The Plugin screen also features bulk actions for activating, deactivating, updating, and deleting plugins. Check all the plugins you want to manage and then select the appropriate bulk action from the drop-down menu. This process makes managing multiple plugins a breeze!

Editing Plugins

WordPress features a built-in plugin editor under the Plugins ➪ Editor screen. The plugin editor enables you to view and edit the source code of any plugin installed in WordPress. Keep in mind you can only edit the source code if the plugin file is writeable by the web server, otherwise you can only view the code.

To use the editor, select the plugin from the drop-down menu on the top-left portion of the Edit Plugins page. The editor lists all files associated with the selected plugin. There is also a documentation lookup feature making it easy to research a specific function's purpose in the plugin you are reviewing.

A word of caution when using the built-in plugin editor: A browser doesn't have an Undo button. There is also no code revision history, so one bad code edit can crash your entire site with no way to revert the changes back. It's best to use the code editor for reference only and never use it to edit your plugin files.

Plugin Directories

A lesser known fact is WordPress actually features two plugin directories. The primary plugin directory is located under `wp-content/plugins` in a standard WordPress installation. The second, lesser known, plugin directory is located under `wp-content/mu-plugins`. The `mu-plugins`

directory, which stands for Must-Use, is not auto-created by WordPress, so it must be manually created to be used.

The primary difference between the two is the `mu-plugins` directory is for plugins that are always executed. This means any plugin included in this directory will automatically be loaded in WordPress and across all sites in the network if you run Multi-site.

The `mu-plugins` *directory will not read plugins in a subfolder, so all plugins must be individual files or must include additional files that exist in a subdirectory. Any plugin files in a subfolder will be ignored unless included in the primary plugin file.*

Types of Plugins

WordPress features a few different types and statuses for plugins, as shown in Figure 1-4. You need to understand the difference when administering and creating plugins for WordPress.

FIGURE 1-4

- **Active** — Plugin is active and running in WordPress.
- **Inactive** — Plugin is installed but not active. No code from the plugin is executed.
- **Must-Use** — All plugins installed in the `wp-content/mu-plugins` directory. All Must-Use, or MU, plugins are loaded automatically. The only way to deactivate an MU plugin is to remove it completely from the directory.
- **Drop-ins** — Core functionality of WordPress can be replaced by Drop-in plugins. These plugins are a specifically named PHP files located in the `wp-content` directory. If WordPress detects one of these files, it will be auto-loaded and listed under the Drop-in filter on the Plugin screen. Currently ten Drop-in plugins are available:
  - `advanced-cache.php` — Advanced caching plugin
  - `db.php` — Custom database class
  - `db-error.php` — Custom database error message
  - `install.php` — Custom installation script
  - `maintenance.php` — Custom maintenance message
  - `object-cache.php` — External object cache
  - `sunrise.php` — Advanced domain mapping
  - `blog-deleted.php` — Custom blog deleted message
  - `blog-inactive.php` — Custom blog inactive message
  - `blog-suspended.php` — Custom blog suspended message

The last four drop-in plugins are specific to the WordPress Multisite feature. A standard WordPress installation will have no use for these plugins.

When developing a new plugin, determine what type of plugin you want to create before you start the development process. Most plugins will be standard WordPress plugins, but occasionally you might need to create a Must-Use or Drop-in specific plugin.

Testing Plugin Functionality

On occasion you may want to test some plugin functionality without actually creating a plugin to do so. Many developers will place code directly in the `wp-config.php` file to do so. This is a bad technique and should not be used because when the config file is parsed and loaded, WordPress is not wholly instantiated yet.

Instead of hacking `wp-config.php`, make a `test.php` file with the following code snippet and place it in your WordPress root directory:


```
<?php
// Load the WordPress Environment
// define( 'WP_DEBUG', true ); /* uncomment for debug mode */
require('./wp-load.php');
// require_once ('./wp-admin/admin.php'); /* uncomment for is_admin() */
?>
<pre>
<?php

/* test stuff here */
var_dump( is_admin() );

?>
</pre>
```

Code snippet test.php

This is a quick way to load all of the required WordPress functions to test plugin functionality without actually creating a plugin. As you can see `wp-load.php` is included at the beginning of the file. You can also include `wp-admin/admin.php` if you want to test admin side functionality. Once you have included the required WordPress core files, you want test any code that would otherwise exist reside in your plugin. Don't forget to remove your `test.php` file when you are done testing.

SUMMARY

In this chapter you learned what about plugins and how they can interact with WordPress using the available APIs. The major advantages to using plugins and why plugin functionality shouldn't always be included in a theme was discussed. Installing and managing plugins in the WordPress admin dashboard was covered.

Now that you understand how plugins work in WordPress, it's time to create the plugin foundation!