Must-Use Plugins

- When to Use Must-Use Plugins
- When Not to Use Must-Use Plugins
- Features
- Caveats
- Maintenance and Security
- History and Naming
- Source Code

Must-use plugins, also known as mu-plugins, are plugins installed in a special directory inside the content folder. They are automatically enabled on all sites in the installation.

Must-use plugins do not show in the default list of plugins on the **Plugins** screen in wp-admin, although they do appear in a separate **Must-Use** section. They cannot be disabled from wp-admin. To disable a must-use plugin, remove the plugin file from the must-use plugins directory, which is **wp-content/mu-plugins** by default.

Web hosts commonly use mu-plugins to add support for host-specific features, especially features where removing the plugin could break the site.

To change the default directory manually, define `WPMU_PLUGIN_DIR` and `WPMU_PLUGIN_URL` in wp-config.php.

When to Use Must-Use Plugins

Must-use plugins are useful for code that should always run and should not be disabled accidentally from wp-admin.

Common use cases include:

- Host-specific integrations, such as caching, monitoring, or platform compatibility code.
- Site-specific bootstrap code that must load before normal plugins.

- Security, performance, or maintenance functionality that should remain active for the whole site.
- Network-wide functionality on Multisite installations.

When Not to Use Must-Use Plugins

Must-use plugins are not a replacement for normal plugins in every situation. Avoid using the mu-plugins directory for plugins that:

- Need activation or deactivation from wp-admin.
- Depend on activation hooks, deactivation hooks, or uninstall hooks.
- Need normal plugin update notifications in wp-admin.
- Are intended to be distributed and managed like regular WordPress plugins.
- Should be optional for site owners or administrators.

Features

- Always on, with no need to enable them in wp-admin.
- Cannot be disabled by users by accident.
- Can be enabled by uploading a PHP file to the mu-plugins directory.
- Loaded by PHP in alphabetical order before normal plugins. This means hooks added in an mu-plugin can apply to normal plugins, even if those normal plugins run hooked functions in the global namespace.

Caveats

Despite its suitability for many special cases, the mu-plugins system is not always ideal and has several downsides that make it inappropriate in certain circumstances. Below are several important caveats to keep in mind:

- Plugins in the must-use directory will not appear in update notifications or show their update status on the plugins screen, so you are responsible for learning about and performing updates yourself.
- Activation hooks are not executed for plugins added to the must-use plugins folder. These hooks are used by many plugins to run setup code when the plugin is activated, or cleanup code when the plugin is deleted. Plugins that depend on these hooks may not function correctly in the mu-plugins folder. Test plugins carefully in the mu-plugins directory before deploying them to a live site.

- WordPress only looks for PHP files directly inside the mu-plugins directory. Unlike normal plugins, files in subdirectories are not included automatically. You may want to create a proxy PHP loader file inside the mu-plugins directory:

```
wp-content/
  mu-plugins/
    load.php
    my-plugin/
      my-plugin.php
```

```
<?php // mu-plugins/load.php
require WPMU_PLUGIN_DIR . '/my-plugin/my-plugin.php';
```

Maintenance and Security

Because must-use plugins always load, keep them small, reviewed, and documented. A broken mu-plugin can affect the whole site, and a compromised mu-plugin can be harder to notice than a normal plugin because it cannot be disabled from the main plugins list.

Keep a record of why each mu-plugin exists, who maintains it, and how it should be updated. If a host or agency adds mu-plugins to a site, site owners should know that those files exist and that they may affect site behavior.

History and Naming

The *mu-plugins* directory was originally implemented by WPMU, or WordPress MultiUser, to offer site admins an easy way to activate plugins by default on all blogs in the network. This was needed because, at the time, the multi-user-specific code did not offer ways to manage this from the site admin section.

WordPress MultiUser was later merged into WordPress core, and the feature is now known as Multisite. Today, Multisite includes features to manage plugins from inside the admin.

The code handling `/mu-plugins/` was merged into the main WordPress code on March 7, 2009 with this changeset, before the WPMU codebase was initially merged. This made autoloaded plugins available to all WordPress sites, whether or not Multisite was enabled.

In this process, the name “mu plugins” became a misnomer because it no longer applied exclusively to Multisite installations. Despite this, the name was kept and reinterpreted to mean “must-use plugins”. These are plugins that must always be used, so they are autoloaded regardless

of the settings in the **Plugins** screen of wp-admin.

Thus, “must-use” is effectively a backronym.

Source Code

- `get_mu_plugins()` checks the mu-plugins directory and retrieves all mu-plugin files with plugin data.
- `wp_get_mu_plugins()` retrieves an array of must-use plugin files.

□□□ #2
□ □□□ □□□ 2026-10-31 18:05:16 UTC
□ □□□ □□□ 2026-10-31 18:05:18 UTC