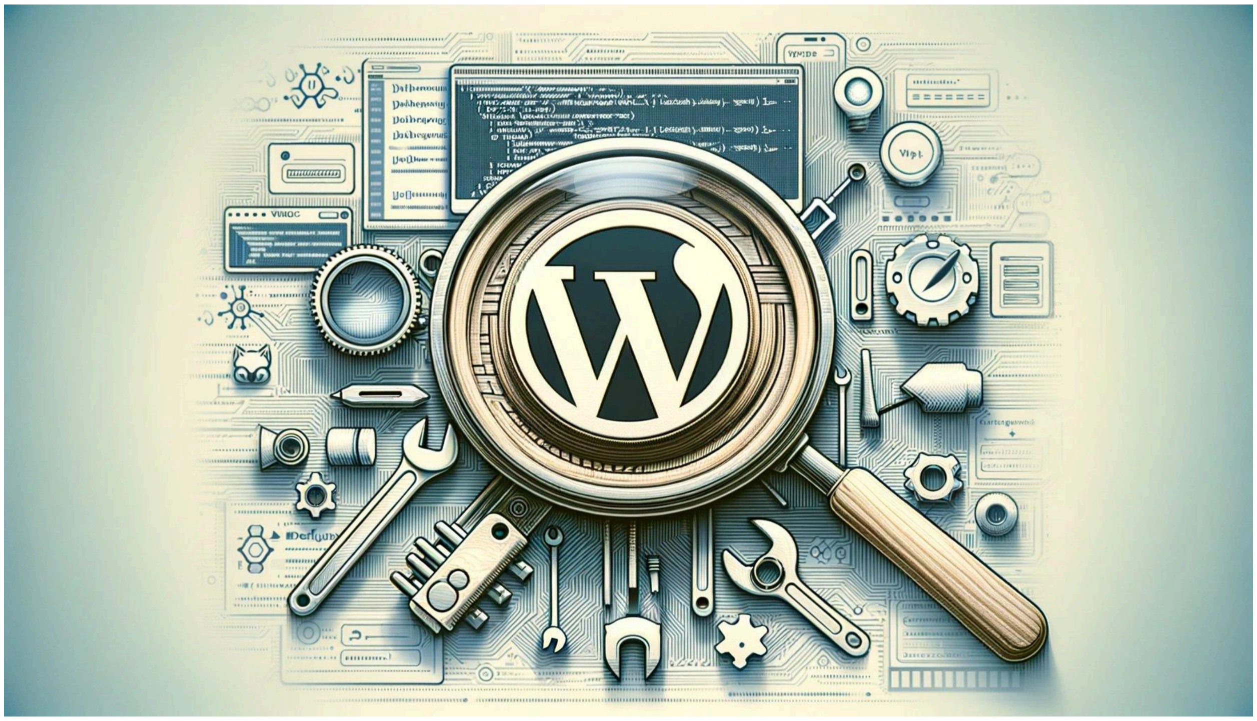

How to Enable Debugging Mode in WordPress

https://wpsandbox.net/1026

If you need some troubleshooting done or want to share your suggestions with us, contact us from **https://wpsandbox.net/contact** page.

Debugging is an essential process in any software to figure out what and when it went wrong.

WordPress has a way to enable debugging. It is normally turned off by default because it's not very professional to show errors on the site and this causes users to trust less that specific site thinking that if there are some basic errors what else is broken.

Why Logging is Very Important

Logging is very important because there are so many moving parts of a site and with constant WordPress, plugin and theme updates lots of things change and at once. You need to be able to see the issues and determine if you need to upgrade your php version or your database server if your site starts generating database specific error messages.

You or whoever is maintaining the site should regularly (at least weekly) check the server logs or WordPress to see what's going on. Sometimes plugins developers may introduce a glitch or two and that could fill up your logs with the same error message which is not good for performance reasons and also if your site or server runs out of disk space it won't be able to properly operate.

To enable debugging in WordPress you need to carefully edit the wp-config.php file. Make sure you create a copy of the file before editing it and keep the same extension (for security reasons) e.g. wp-config-bak1.php or put the current date as one word wp-config-bak-YYYYMMDD.php.

We highly recommend before adding those debugging php constants to inspect the wp-config.php and remove any existing debugging constants.

Enable Debugging in WordPress

then place those lines at the top of the wp-config.php file so they are clearly visible right after the opening php tag <?php

```
define( 'WP_DEBUG', true );
define( 'WP_DEBUG_LOG', true );
```

Hopefully, you don't have to troubleshoot a live site but if you do then set the **WP_DEBUG_DISPLAY** php const to false, so any debug errors/warnings/notices are logged only in the debug file. Again users should not be seeing any warnings or errors.

```
Define( 'WP_DEBUG_DISPLAY', false );
```

if you're running your site on a Staging WordPress like WPSandbox then it's ok to turn that on.

You can also check this page about debugging on WordPress.org called Debugging in WordPress

Where are WordPress Debug Logs Stored

When debugging is enabled WordPress stores the log files in wp-content/debug.log unless instructed otherwise.

If you want to store the files elsewhere for security reasons you can change the location like shown in this Orbisius WordPress Debugging article.

```
if ( empty( $_SERVER['DOCUMENT_ROOT'] ) ) {
	define( 'WP_DEBUG_LOG', '.ht_debug.log' );
} else {
	define( 'WP_DEBUG_LOG', dirname($_SERVER['DOCUMENT_ROOT']) .
'/.ht_debug.log' );
}
```

This will instruct WordPress to use your custom log file instead.

Additional and More Advance Debugging

When you enable WordPress debugging and also have turned the debug display on it WordPress will naturally show errors/notices for every request. This may break Ajax requests. For this reason we need to check if it’s an ajax request and only enable if isn’t. Keep in mind that we’re referring to error/notice displaying .. the errors will be still logged into the debug file.

We won’t see them on the page.

```
defined('WP_DEBUG_DISPLAY') || define( 'WP_DEBUG_DISPLAY',
!empty($_SERVER['HTTP_X_REQUESTED_WITH']) &&
strtolower($_SERVER['HTTP_X_REQUESTED_WITH']) == 'xmlhttprequest' ? false : true );
```

WordPress at some point started catching some majour errors such as non-existent php function so to turn off that behaviour add this constant to the wp-config.php

```
define( 'WP_DISABLE_FATAL_ERROR_HANDLER', true );
```

How to Enable WordPress Debugging only a CLI or for Specific IP addresses

You can have the following check which will check the client’s IP address which is stored in **$_SERVER[‘REMOTE_ADDR’]** field. Normally, it’s empty when running WordPress the command line to do some CLI tasks.

In the second check we check for localhost IP addresses and a specific IP address.

Make sure you change 11.22.33.44 with your IP address. Keep the quotes or the site will crash!

```
// Enable WordPress debugging for a CLI or for specific IP addresses
if (empty($_SERVER['REMOTE_ADDR']) || in_array($_SERVER['REMOTE_ADDR'], [
'127.0.0.1', '::1', '11.22.33.44', ] ) ) {
// put all debugging constants here
}
```

Debugging on Development Machine

The suggestions above were about debugging/troubleshooting via logging any error messages. If you're striving to become a WordPress professional it's highly recommended that you have installed xdebug php extension and the related xdebug browser addons so you can more effectively troubleshoot your or somebody else's code.

xDebug is an awesome extension and you can put breakpoints and stop the page's execution and inspect all the variables the script has at the moment.

IDE. Your code editor is super important too. It can catch glitches very early on. For example if a variable is used before it has received a value. We highly recommend PHPStorm with GitHub Copilot Plugin installed. You will even gain some 20-30% productivity out of this.

Do you have other WordPress techniques on how to troubleshoot WordPress?