服务器环境

目录
WordPress 环境推荐
Web Server
PHP
WordPress 版本
WordPress 6.8
WordPress 6.7
WordPress 6.6
WordPress 6.5
WordPress 6.4
WordPress 6.3
关于 PHP
PHP 扩展
内置扩展（无需托管操作）
必需扩展
System Packages
数据库
官方推荐版本
End of life MySQL versions
End of life MariaDB versions
创新版本发布
MySQL Innovation Releases
MariaDB Innovation/Short-Term Releases Other MySQL servers
实验性功能： SQLite（功能插件）
如何查看当前版本？

尽管 WordPress 几乎能在任何环境中运行，但某些环境在功能和性能方面更为优化。以下是针对 WordPress 高效运行的最低服务器环境配置建议，同时考虑了使用第三方插件和主题的 WordPress

网站，这些插件和主题通常会引入额外的服务器级要求。

WordPress 环境推荐
快速推荐：

WordPress 6.8 服务器兼容性
WordPress 6.7 服务器兼容性
WordPress 6.6 服务器兼容性
WordPress 6.5 PHP 兼容性
WordPress 6.4 PHP 兼容性
WordPress 6.3 PHP 兼容性

所有已发布的兼容性文章可在以下位置查看：

版本兼容性

Web Server

Web
服务器是一种软件，它接收并处理来自网站访问者计算机的网络请求，然后将相应的网页数据返回给用户。存
在许多不同类型的 Web 服务器软件，它们运行在不同的操作系统上。通常，只要您的 Web 服务器支持并执行 PHP 文件，它就应该能够运行 WordPress。

两种最常见的、也是 WordPress 推荐的 Web 服务器软件是：

Apache HTTPD 2.4
nginx 1.26 & 1.27

已知与 WordPress 兼容良好的、由网络托管公司和开发者使用的其他软件包括：

Angie 1.7
LiteSpeed Web Server 6.3 / 6.2 / 6.1 / 6.0 / 5.4
OpenLiteSpeed 1.8 / 1.7

以上是撰写本文档时（针对 WordPress 6.8）的最新版本。请务必保持您的 Web 服务器为最新版本，以确保最佳性能！

PHP

PHP 是 WordPress
代码所基于的编程语言。该语言在服务器上运行，为了安全性和功能性，保持其更新至关重要。

WordPress 支持许多 PHP 版本，其中一些甚至已过时（ 参见 PHP 兼容性与 WordPress 版本
）。对于托管公司，建议如下：

WordPress 版本
以下是特定 WordPress 版本的详细信息、该版本的 PHP 兼容性以及发布时与 PHP 兼容性相关的开发工单。与 PHP 兼容性相关的工单可随时通过 搜索 WordPress Trac查找。

WordPress 6.8

PHP 8.1 (安全支持)
PHP 8.2 (活跃支持)
PHP 8.3 (活跃支持)
PHP 8.4 (候选支持)

重要提示： WordPress 6.8 完全兼容 PHP 7.2 (1)、 7.3 (1)、7.4 (1)、8.0(1)、8.1 和 8.2，并且 Beta 兼容 PHP 8.3 和 PHP 8.4。 自 2025 年 4 月发布的 WordPress 6.8 起，不再使用 “附带例外兼容 ”这一术语。

“Beta 兼容 ”或 “Beta 支持 ”是什么意思？

“Beta 兼容 ”或 “Beta 支持 ”意味着 WordPress 核心正在积极努力实现与该 PHP
版本的完全兼容，但可能仍有一些问题正在解决过程中。以下是概述 PHP 8.3 和 8.4 Beta
支持已知问题的工单。关于 PHP 版本何时从 Beta 转为完全支持 的更多信息，由核心团队记录。

PHP 8.3
当使用 “Beta 兼容 ”的 PHP 版本时，可能会在错误日志、 wp-admin
或页面上看到弃用通知。弃用通知不是错误，而是表明通知中引用的代码将在未来的 PHP
版本中更改。出现弃用通知时， PHP 代码将继续工作，没有任何问题。
#59231: 为 PHP 8.3 做准备。 。注意：已关闭 /已修复
#59232: 引入 #[Override] 属性来标记重载方法
此属性通过明确标记重载方法来帮助防止编码错误。它还有助于重构、调试以及捕获父类中潜
在的破坏性更改。 注意：已有补丁，但移至未来版本。
#59233: 改进 unserialize() 的错误处理 。maybe_unserialize() 函数仍可能遇到带有尾随字节的数据。 注意：移至未来版本。
PHP 8.4

    - □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ PHP □□□□□□□□□□□□□ PHP □□□□□□□□□□□□□□□
    - #62061: □ PHP 8.4 □□□□ □□ □□□□□□ /□□□
- Other PHP Related Tickets
    - #51525: Add new functions apply_filters_single_type() and apply_filters_ref_array_single_type().. *Note: Moved to Future Release.*
    - #54183: Tests: decide on how to handle deprecations in PHPUnit. *Note: Moved to Future Release.*
    - #54537: Tests: Enable PHP version check once PHP 8.0 compatibility is achieved.. *Note: Moved to Future Release.*
    - #58874: Code Modernization: Consider using the null coalescing operator.. *Note: Moved to Future Release.*
    - #59234: Introduce a wp_json_decode() function, including validation when available. *Note: This ticket has been closed and won't be moving forward.*

WordPress 6.7

- PHP 8.1
- PHP 8.2
- PHP 8.3

*IMPORTANT: WordPress 6.7 is **compatible with exceptions** with PHP 8.0, PHP 8.1, PHP 8.2, and **beta compatible** with PHP 8.3, and PHP 8.4.*

What does 'compatible with exceptions' mean?

- PHP 8.0
    - **#48689**: Filesystem WP_Filesystem_FTPext and WP_Filesystem_SSH2 when connect fails. An investigation is underway as to why on some occasions the access to the files returns some type of error. *NOTE: Has a patch.*
    - **#49728**: Prepare for the internal functions throwing TypeError or ValueError exceptions on unexpected types/values. Internal functions will throw an exception if the function call arguments are of a type that is not expected. *NOTE: Has a patch.*
    - **#51019**: convert_smilies() fails on large tags. The function fails when dealing with large HTML tags, particularly when an image with a large data URL is included in the post content. *NOTE: Has a patch, but moved to WordPress 6.8.*
    - **#55121**: classic widgets with no settings won't show up in 5.9. Classic widgets with no settings do not appear correctly in WordPress 5.9 and above. This is due to changes in how widgets are handled in the block editor, causing compatibility problems with older widget setups. *NOTE: Has a patch, but moved to WordPress 6.3.*

#55257: map_deep() function incompatibility with incomplete objects in PHP 8.0+. The function becomes incompatible with incomplete objects when running on PHP 8.0 or higher. NOTE: Has a patch, but moved to Future Release.
#59649: Named parameters. WordPress does not support named parameters. PHP 8.0 supports optionally calling functions and class methods by specifying the parameter name, instead of calling them on the order of parameters that they are declared. PHP, and many other programming languages, support positional parameters: The caller passes the parameters in the same order the
function/method declares its parameters. NOTE: Moved to WordPress 6.8.

#60745: WP_Query::parse_query() does not handle invalid query arg values. The function does not properly handle invalid query argument values. This results in PHP fatal errors when unintended data types, like arrays, are passed where scalars are expected. NOTE: Has a patch, but moved to WordPress 6.8.
PHP 8.1
并非所有 “向不可为空参数传递 null”的问题都已被发现。在 PHP
中，你可以明确告知函数应接受何种类型的信息。如果你告诉函数期望某种特定类型的信息，
却完全没有提供任何信息（ null 类似于表示 “无”），那么 PHP
会感到困惑并报错。当有人意外地向一个未设计为处理 “无 ”的函数传递 “无 ”
时，就会出现此问题。

#53465: htmlentities() 需要显式设置 flags 参数的默认值。 根据 htmlentities()，PHP 8.1 中 flags 的默认值 “已从 ENT_COMPAT 更改为 ENT_QUOTES | ENT_SUBSTITUTE | ENT_HTML401 ” 。正在调查核心中此功能的所有用例。 注意：已有补丁。

#57579: 将大多数 strip_tags() 替换为 wp_strip_tags()。
提议通过最小化加载过程中的不必要文件系统调用来优化块模板的处理方式。
注意：已有补丁，但移至未来版本。

#57580: 避免 add_submenu_page() 中 null 参数导致的错误。 当 null
作为参数传递时，由 wp_normalize_path() 函数引起的已弃用错误。此问题主要由于插件在
add_submenu_page() 函数中错误地传递了 null。注意：已有补丁，但移至未来版本。
#61179: 关于在 widgets.php 中传递 null 的已弃用消息 解决 widgets.php
中的一个已弃用消息问题。 注意：已有补丁，但移至 WordPress 6.7。
PHP 8.2

#55603: utf8_{encode|decode} deprecation with pending decision on requiring a PHP extension. NOTE: Has a patch, but moved to WordPress 6.8.
#56034: Unknown dynamic properties' deprecation. NOTE: Moved to Future
Release.

#57304: Add SensitiveParameter attribute to DB connection and login variables.
This enhancement aims to protect sensitive data in case of errors, making it less likely to be exposed in logs or bug reports. NOTE: Moved to WordPress 6.7. #60875: Handler proposal for known dynamic properties that are initialized and set

late only when getting its value. Handling dynamic properties that are initialized only when accessed. Since PHP 8.2 deprecates dynamic (non-declared) properties, the proposal aims to pre-declare these properties while retaining the current design

where they are lazily initialized. *NOTE: Moved to WordPress 6.8.*

- **#61154**: Fix the 'attributes' dynamic property in WP_Block. Fixing the 'attributes' dynamic property in the `WP_Block` class. *NOTE: Has a patch, but moved to WordPress 6.8.*
- **#61890**: Handle WP_Term dynamic properties for PHP 8.2. Handling of dynamic properties in the `WP_Term` class to ensure compatibility. *NOTE: Has a patch, but moved to WordPress 6.8.*

What does "beta" mean?

- PHP 8.3
  - Deprecation notices. A deprecation notice is not an error, but is an indicator that the code being cited in the notice will be changed in future php versions. With a deprecation notice, the PHP code will continue to work and nothing is broken.
  - **#59231**: Prepare for PHP 8.3.. *NOTE: Has a patch, but moved to WordPress 6.7.*
  - **#59232**: Introduce #[Override] attribute to mark overloaded methods This attribute helps prevent coding errors by making it clear when a method is overloaded. It also assists with refactoring, debugging, and catching potential breaking changes in the parent class. *NOTE: Has a patch, but moved to Future Release.*
  - **#59233**: Improve error handling for unserialize(). `maybe_unserialize()` function could still be confronted by data with trailing bytes. *NOTE: Moved to Future Release.*
  - **#59654**: PHP 8.x: various compatibility fixes for WordPress 6.7. This ticket acts as a central hub for smaller patches that fix specific PHP 8.x failures. It continues the work from previous releases, ensuring that WordPress maintains compatibility with newer PHP versions like PHP 8.0, 8.1, 8.2, and upcoming versions like PHP 8.3. *NOTE: Moved to WordPress 6.7.*
- PHP 8.4
  - □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ PHP □□□□□□□□□□□□□□□ PHP □□□□□□□□□□□□□□□□□
  - **#62061**: □ PHP 8.4 □□□□ □ □□□□□□□□

□□□□□□

- PHP
  - **#51525**: □□□□ apply_filters_single_type() □ apply_filters_ref_array_single_type() □
  - **#54183**: □□□□□□□□□ PHPUnit □□□□□□
  - **#54537**: □□□□□□□ PHP 8.0 □□□□□□□ PHP □□□□□
  - **#58874**: □□□□□□□□□□□□□□□□□□
  - **#59234**: □□ `wp_json_decode()` □□□□□□□□□□□□

WordPress 6.6

PHP 8.1

PHP 8.2

PHP 8.3

重要提示： WordPress 6.6 与 PHP 8.1 和 PHP 8.2 存在例外兼容 ，与 PHP 8.3 为 Beta 兼容 。

“存在例外兼容 ”是什么意思？

PHP 8.1
并非所有 “向不可为 null 的参数传递 null”的问题都已被发现。 在 PHP
中，你可以明确告诉函数它应该接受什么类型的信息。如果你告诉函数期望某种类型的信息，
而你却什么都没给它（ null 就像是说 “什么都没有 ”），那么 PHP
会感到困惑并报错。当有人不小心给一个函数传递了 “什么都没有 ”，而该函数并未设计为处理 “
什么都没有 ”时，就会发生此问题。
_ htmlentities() 需要显式设置 flags 参数的默认值 _。根据 htmlentities()，PHP 8.1 中
flags 的默认值 “已从 ENT_COMPAT 更改为 ENT_QUOTES | ENT_SUBSTITUTE |
ENT_HTML401”。正在调查 WordPress
核心中此功能的所有使用案例。注意：已有补丁，但移至 WordPress 6.7。
将大多数 strip_tags() 替换为 wp_strip_tags()。在极少数情况下， strip_tags()
函数会接收到 null 值，这会生成一个关于字符串已弃用的警告。注意：已有补丁。
更新 is_serialized 函数以接受 Enums。 Enums 与旧版 PHP
不向后兼容。注意：已有补丁，但移至 WordPress 6.7。
PHP 8.2

utf8_{encode|decode} 弃用 ，关于是否需要 PHP 扩展的决定待定。注意：已有补丁，但移至
WordPress 6.7。
“未知动态属性 ”弃用。注意：移至 WordPress 6.7。

“Beta”是什么意思？

PHP 8.3
弃用通知 。弃用通知不是错误，而是表明通知中引用的代码将在未来的 PHP
版本中更改。出现弃用通知时， PHP 代码将继续工作，没有任何问题。
改进 unserialize() 的错误处理 。 maybe_unserialize()
函数仍可能遇到带有尾随字节的数据。注意：移至 WordPress 6.7。

WordPress 6.5

PHP 8.1

PHP 8.2

PHP 8.3

重要提示： WordPress 6.5 与 PHP 8.0、PHP 8.1 和 PHP 8.2 存在例外兼容 ，与 PHP 8.3 为 Beta 兼容 。

“存在例外兼容 ”是什么意思？

PHP 8.0
命名参数 。WordPress 不支持命名参数。
连接失败时的文件系统 WP_Filesystem_FTPext 和 WP_Filesystem_SSH2。
PHP 8.1
并非所有 “向不可为 null 的参数传递 null”的问题都已被发现。
htmlentities() 等函数需要显式设置 flags 参数的默认值 。
将大多数 strip_tags() 替换为 wp_strip_tags()。
PHP 8.2

utf8_{encode|decode} 弃用 ，关于是否需要 PHP 扩展的决定待定。
未知动态属性 弃用。

“Beta”是什么意思？

PHP 8.3
弃用通知：弃用通知不是错误，而是表明通知中引用的代码将在未来的 PHP
版本中更改。出现弃用通知时， PHP 代码将继续工作，没有任何功能损坏。

WordPress 6.4

PHP 8.1
PHP 8.2

重要提示： WordPress 6.4 与 PHP 8.0、PHP 8.1 和 PHP 8.2 存在例外兼容 ，与 PHP 8.3 为 Beta 兼容 。

“存在例外兼容 ”是什么意思？

PHP 8.0
命名参数 。WordPress 不支持命名参数。
连接失败时的文件系统 WP_Filesystem_FTPext 和 WP_Filesystem_SSH2。
PHP 8.1
并非所有 “向不可为 null 的参数传递 null”的问题都已被发现。
htmlentities() 等函数需要显式设置 flags 参数的默认值 。
将大多数 strip_tags() 替换为 wp_strip_tags()。
PHP 8.2

utf8_{encode|decode} 弃用 ，关于是否需要 PHP 扩展的决定待定。
未知动态属性 弃用。

“Beta”是什么意思？

PHP 8.3

弃用通知：弃用通知不是错误，而是表明通知中引用的代码将在未来的 PHP
版本中更改。出现弃用通知时， PHP 代码将继续工作，没有任何问题。

WordPress 6.3

PHP 8.1

PHP 8.2

重要提示： WordPress 6.3 与 PHP 8.0 和 PHP 8.1 存在例外兼容 ，与 PHP 8.2 为 Beta 兼容 。

“存在例外兼容 ”是什么意思？

PHP 8.0
命名参数。 WordPress 不支持命名参数。

连接失败时的文件系统 WP_Filesystem_FTPext 和 WP_Filesystem_SSH2。
PHP 8.1

htmlentities() 等函数需要显式设置 flags 参数的默认值 。

将大多数 strip_tags() 替换为 wp_strip_tags()。

对未知设置的 unregister_setting()。

“Beta”是什么意思？

PHP 8.2
弃用通知：弃用通知不是错误，而是表示通知中引用的代码将在未来的 PHP
版本中更改。出现弃用通知时， PHP 代码将继续工作，没有任何问题。

关于 PHP

自 2022 年 11 月 26 日起， PHP 社区仅以 仅安全修复 模式维护 PHP 8.1。将 PHP 保持在最新的稳定版本对于 WordPress 的速度和安全性至关重要。

PHP 社区不再维护 PHP 8.1 之前的版本，尽管它们可能会从操作系统发行版获得安全更新。

已终止支持的 PHP 版本：

PHP 8.3：2027 年 12 月 31 日
PHP 8.2：2026 年 12 月 31 日
PHP 8.1：2025 年 12 月 31 日
PHP 8.0：2023 年 11 月 26 日 最后版本： 8.0.30
PHP 7.4：2022 年 11 月 28 日 最后版本： 7.4.33
PHP 7.3：2021 年 12 月 6 日 最后版本： 7.3.33
PHP 7.2：2020 年 11 月 30 日 最后版本： 7.2.34
PHP 7.1：2019 年 12 月 1 日 最后版本： 7.1.33

PHP 7.0：2019 年 1 月 10 日 最后版本： 7.0.33
PHP 5.6：2018 年 12 月 31 日 最后版本： 5.6.40
PHP 5.5：2016 年 7 月 21 日 最后版本： 5.5.38
PHP 5.4：2015 年 9 月 3 日 最后版本： 5.4.45
PHP 5.3：2014 年 8 月 14 日 最后版本： 5.3.29
PHP 5.2：2011 年 1 月 6 日 最后版本： 5.2.17
PHP 5.1：2006 年 8 月 24 日 最后版本： 5.1.6
PHP 5.0：2005 年 9 月 5 日 最后版本： 5.0.5
PHP 4.4：2008 年 8 月 7 日 最后版本： 4.4.9
PHP 4.3：2005 年 3 月 31 日 最后版本： 4.3.11
PHP 4.2：2002 年 9 月 6 日 最后版本： 4.2.3
PHP 4.1：2002 年 3 月 12 日 最后版本： 4.1.2
PHP 4.0：2001 年 6 月 23 日 最后版本： 4.0.6

PHP 扩展

当可用时， WordPress 核心会利用各种 PHP 扩展。如果缺少首选扩展， WordPress
要么需要做更多工作来完成该模块辅助的任务，要么在最坏情况下，会移除相关功能。

说明：

一些扩展已内置在 PHP 中（或默认启用），在典型的托管环境中无需特殊操作。
其他扩展是可选的，可能需要根据 PHP 的打包方式来安装 /启用。

以下列表假设 PHP 版本 >= 7.2（撰写本文时 WordPress 支持的最低 PHP 版本）。

内置扩展（无需托管操作）

以下扩展是 PHP 自身的组成部分，在支持的 PHP 版本（ >= 7.2）中始终启用。无需进行托管配置：

pcre - 在 PHP 和 WordPress 中广泛使用的正则表达式引擎。

必需扩展

以下列出的 PHP 扩展是 WordPress 网站运行所 必需 的。

json - 用于与其他服务器通信以及处理 JSON 格式的数据。（在 PHP >= 8.0 中始终启用；在 PHP 7.x 中默认启用，但在自定义构建中可能被禁用。）

mysqli - 连接到 MySQL/MariaDB 以进行数据库交互。许多 PHP 构建使用 mysqlnd
作为底层驱动程序，但 mysqlnd 并不能独立替代 mysqli。

以下列出的 PHP 扩展是 强烈推荐 的，以便让 WordPress
以最佳状态运行，并最大限度地提高与许多流行插件和主题的兼容性。

- curl (PHP >= 7.3 requires libcurl >= 7.15.5; PHP >= 8.0 requires libcurl >= 7.29.0 ; PHP >= 8.4 requires libcurl >= 7.61.0) - Performs remote request operations.
- dom (requires libxml) - Used to validate Text Widget content and to automatically configure IIS7+.
- exif - Works with metadata stored in images.
- fileinfo - Used to detect MIME types of file uploads.
- hash - Used for hashing, including passwords and update packages. (Always enabled in PHP >= 7.4; on PHP 7.2-7.3 it is commonly enabled but not guaranteed.)
- igbinary - Increases performance as a drop in replacement for the standard PHP serializer.
- imagick (requires ImageMagick >= 6.2.4) - Provides better image quality for media uploads. See WP_Image_Editor for details. Smarter image resizing (for smaller images), and PDF thumbnail support when Ghost Script is available.
- intl (PHP >= 7.4.0 requires ICU >= 50.1) - Enable to perform locale-aware operations including but not limited to formatting, transliteration, encoding conversion, calendar operations, conformant collation, locating text boundaries and working with locale identifiers, timezones and graphemes.
- mbstring - Used to properly handle UTF8 text.
- openssl (PHP 7.1-8.0 requires OpenSSL >= 1.0.1 / < 3.0; PHP >= 8.1 requires OpenSSL >= 1.0.2 / < 4.0; PHP >= 8.4 requires OpenSSL >= 1.1.1 / < 4.0) - SSL-based connections to other hosts.
- xml (requires libxml) - Used for XML parsing, such as from a third-party site.
- zip (requires libzip >= 0.11; recommended libzip >= 1.6) - Used for decompressing Plugins, Themes, and WordPress update packages.

The PHP extensions listed below are *recommended* to allow some WordPress cache (if necessary). APCu, Memcached, and Redis are alternatives of which only one needs to be used.

- apcu – In-memory key-value store for PHP (former APC stripped of opcode caching).
- memcached (requires libmemcached >= 1.0.0) - memcached is a high-performance, distributed memory object caching system, generic in nature, but intended for use in speeding up dynamic web applications by alleviating database load.
- opcache - PHP can be configured to preload scripts into the opcache when the engine starts.
- redis - PHP extension for interfacing with Redis.

The PHP extensions listed below are *optional* to improve some WordPress functionality.

- timezonedb - Timezone Database to be used with PHP's date and time functions

For the sake of completeness, below is a list of the remaining PHP modules WordPress *may* use in certain situations or if other modules are unavailable. These are fallbacks or optional and not necessarily needed in an optimal environment, but installing them won't hurt.

- bc - For arbitrary precision mathematics, which supports numbers of any size and precision up to 2147483647 decimal digits.
- filter - Used for securely filtering user input.
- image (requires libgd >= 2.1.0; requires zlib >= 1.2.0.4; optional freetype2) - If Imagick isn't installed, the GD Graphics Library is used as a functionally limited fallback for image manipulation.
- iconv (requires libiconv/POSIX) - Used to convert between character sets.
- shmop - Shmop is an easy to use set of functions that allows PHP to read, write, create and delete Unix shared memory segments.
- simplexml (requires libxml) - Used for XML parsing.
- sodium (bundled with PHP as of 7.2.0; requires libsodium >= 1.0.8) - Validates signatures and provides securely random bytes.
- xmlreader (requires libxml) - Used for XML parsing.
- zlib (requires zlib >= 1.2.0.4) - Gzip compression and decompression.

These extensions are used for file changes, such as updates and plugin/theme installation, when files aren't writeable on the server.

- ssh2 (requires OpenSSL and libssh >= 1.2; recommended libssh >= 1.2.9) - Provide access to resources (shell, remote exec, tunneling, file transfer) on a remote machine using a secure cryptographic transport.
- ftp - Implement client access to files servers speaking the File Transfer Protocol (FTP).
- sockets - Implements a low-level interface to the socket communication functions based on the popular BSD sockets.

The priority of the transports are Direct file IO, SSH2, FTP PHP Extension, FTP implemented with Sockets, and FTP implemented through PHP alone.

System Packages

- curl (recommended >= 8.4)
- Ghost Script (recommended Ghost Script >= 10.0)- Enables Imagick/ImageMagick to generate PDF thumbnails for the media library. See Enhanced PDF Support in WordPress 4.7 for details.
- ImageMagick (recommended ImageMagick >= 7.1) - Required by Imagick extension.

- OpenSSL (recommended >= 3.0)
- WebP
- AVIF

□□□

WordPress □□□□ MySQL □□□□□□□□□□

□□□□□

□□□ MySQL □ MariaDB □□□□□□□□□□□□□

□□	□□	□□□□□□□□
MySQL	8.4 LTS	2032□ 4□
MySQL	8.0 LTS	2026□ 4□
MariaDB	11.8 LTS	2030□ 6□
MariaDB	11.4 LTS	2029□ 5□ 29□
MariaDB	10.11 LTS	2028□ 2□ 16□
MariaDB	10.6 LTS	2026□ 7□ 6□

End of life MySQL versions

The following versions of MySQL have reached their end of life (EOL) and are no longer supported by the MySQL community. **It is highly recommended to upgrade to a supported version for security and performance reasons.**

Version	Initial GA Release	Final Release	EOL Date
9.2*	January 21, 2025	9.2.0 (January 21, 2025)	April 15, 2025
9.1*	October 15, 2024	9.1.0 (October 15, 2024)	January 21, 2025
9.0*	July 1, 2024	9.0.1 (July 23, 2024)	October 15, 2024
8.3*	January 16, 2024	8.3.0 (January 16, 2024)	July 1, 2024
8.2*	October 25, 2023	8.2.0 (October 25, 2024)	January 16, 2024
8.1*	July 18, 2023	8.1.0 (July 18, 2023)	October 25, 2023
5.7	October 21, 2015	5.7.44 (October 25, 2023)	October 21, 2023

Version	Initial GA Release	Final Release	EOL Date
5.6	February 5, 2013	5.6.51 (January 20, 2021)	February 5, 2021
5.5	December 3, 2010	5.5.62 (October 22, 2018)	December 3, 2018
5.1	November 14, 2008	5.1.73 (December 3, 2013)	December 31, 2013
5.0	October 19, 2005	5.0.96 (March 21, 2012)	March 21, 2012

End of life MariaDB versions

The following versions of MariaDB have reached their end of life (EOL) and are no longer supported by the MariaDB community. **It is highly recommended to upgrade to a supported version for security and performance reasons.**

Version	Initial GA Release	Final Release	EOL Date
11.5	August 15, 2024	11.5.2 (August 15, 2024)	November 21, 2024
11.3	February 19, 2024	11.3.2 (February 19, 2024)	May 16, 2024
11.2	November 21, 2023	11.2.6 (November 4, 2024)	November 21, 2024
11.1	August 21, 2023	11.1.6 (August 14, 2024)	August 21, 2024
11.0	June 7, 2023	11.0.6 (May 17, 2024)	June 6, 2024
10.10	November 17, 2022	10.10.7 (November 14, 2023)	November 17, 2023
10.9	August 22, 2022	10.9.8 (August 14, 2023)	August 22, 2023
10.8	May 21, 2022	10.8.8 (May 10, 2023)	May 20, 2023
10.7	February 14, 2022	10.7.8 (February 6, 2023)	February 9, 2023
10.5	June 24, 2020	10.8.8 (May 10, 2023)	June 24, 2025
10.4	June 16, 2019	10.4.34 (May 17, 2024)	June 18, 2024
10.3	May 25, 2018	10.3.39 (May 10, 2023)	May 25, 2023
10.2	May 23, 2017	10.2.44 (May 21, 2022)	May 23, 2022
10.1	October 17, 2015	10.1.48 (November 4, 2020)	October 17, 2020
10.0	March 31, 2014	10.0.38 (January 31, 2019)	March 31, 2019
5.5	April 11, 2012	5.5.68 (May 12, 2020)	April 11, 2020
5.3	February 29, 2012	5.3.12 (January 30, 2013)	March 1, 2017
5.2	November 10, 2010	5.2.14 (January 30, 2013)	November 10, 2015
5.1	February 1, 2010	5.1.67 (January 30, 2013)	February 1, 2015

创新版本发布

除了长期支持版本外，MariaDB 和 MySQL 都提供 "创新版本"。这些版本的支持周期要短得多（通常最多 2-6 个月），主要面向需要体验最新功能的用户。

从质量和稳定性角度来看，创新版本通常已具备生产就绪状态，可以安全使用。但不建议在生产环境中使用这些版本，原因如下：

- 创新版本中包含的任何新功能在未来都可能发生变化。
- 支持周期（包括安全修复）仅限于数月而非数年。

为了发现即将在长期支持版本中引入的新功能可能存在的问题，WordPress 核心会针对当前支持的创新版本进行测试。但该项目并不推荐普通用户使用这些版本。

MySQL Innovation Releases

Version	Initial GA Release	Final Release	EOL Date
9.3*	April 15, 2025		July 2025
9.2	January 21, 2025	9.2.0 (January 21, 2025)	April 15, 2025
9.1	October 15, 2024	9.1.0 (October 15, 2024)	January 21, 2025
9.0	July 1, 2024	9.0.1 (July 23, 2024)	October 15, 2024
8.3	January 16, 2024	8.3.0 (January 16, 2024)	July 1, 2024
8.2	October 25, 2023	8.2.0 (October 25, 2024)	January 16, 2024
8.1	July 18, 2023	8.1.0 (July 18, 2023)	October 25, 2023

`*` Indicates the current innovation release.

MariaDB Innovation/Short-Term Releases

The MariaDB project has decided to retire the concept of innovation releases, replacing them with the concept of "rolling GA releases". This will start with the 12.0 release.

Previously MariaDB referred to these releases as "short-term releases".

Version	Release Type	Initial GA Release	Final Release	EOL Date
11.5	Innovation	August 15, 2024	11.5.2 (August 15, 2024)	November 21, 2024
11.3	Innovation	February 19, 2024	11.3.2 (February 19, 2024)	May 16, 2024
11.2	Short-term	November 21, 2023	11.2.6 (November 4, 2024)	November 21, 2024

Version	Release Type	Initial GA Release	Final Release	EOL Date
11.1	Short-term	August 21, 2023	11.1.6 (August 14, 2024)	August 21, 2024
11.0	Short-term	June 7, 2023	11.0.6 (May 17, 2024)	June 6, 2024
10.10	Short-term	November 17, 2022	10.10.7 (November 14, 2023)	November 17, 2023
10.9	Short-term	August 22, 2022	10.9.8 (August 14, 2023)	August 22, 2023
10.8	Short-term	May 21, 2022	10.8.8 (May 10, 2023)	May 20, 2023
10.7	Short-term	February 14, 2022	10.7.8 (February 6, 2023)	February 9, 2023

Other MySQL servers

While WordPress does not regularly test against these MySQL servers, they are known to perform well.

- Percona MySQL Server
- Amazon Aurora
- Amazon RDS for MariaDB
- Amazon RDS for MySQL
- Azure Database for MySQL
- Google Cloud MySQL
- DigitalOcean MySQL
- IBM Cloud Databases for MySQL
- MySQL HeatWave

Although WordPress may run on older versions, it is recommended to use these or newer ones for security and performance reasons.

实验性功能: SQLite (功能插件)

注意: WordPress 核心在生产环境中正式支持 **MySQL/MariaDB**。SQLite 是通过社区 SQLite 数据库集成 功能插件实现的 可选且实验性 功能。请将其视为主机的测试 /评估路径，而非全面替代方案。

征求主机反馈

如果您对此功能进行群体试用，请分享：流量概况、缓存情况、写入速率、数据库文件大小、错误日志以及任

何插件 /主题不兼容情况。如果未来提议合并到核心，这些反馈将非常有用。

如何查看当前版本？
若您使用的是 WordPress 5.2+ 版本，管理后台的 站点健康 功能（位于菜单栏 工具
项下）已内置相关信息查看工具。

若您使用较旧版本，可通过安装名为 Health Check & Troubleshooting的社区插件来激活 站点健康 功能（更多

插件使用帮助 ）。

[info]若您有意完善本手册，请访问 Github 手册仓库 ，或在官方 WordPress Slack的 #hosting 频道 留言。 [/info]

版本号 #2
由 壳壳虫 创建于 2026-03-27 01:08:36 UTC
由 壳壳虫 更新于 2026-03-27 01:08:36 UTC