Engineer

WordPress

an open-source content management system for websites

What it is

WordPress is an open-source content management system: PHP software that keeps a site's posts, pages, media records, users and settings in a MySQL or MariaDB database and assembles pages from them on request. Uploaded media files normally live outside the database. Its name is often shortened to WP, the same prefix its own files and folders carry (wp-config.php, wp-content, wp-admin). It is released under the GPLv2 (or later), so the software costs nothing to download, run or change.

Three layers make up a WordPress site. Core is the shared code that everyone downloads and updates. A theme controls presentation and plugins add behaviour, which is where the project's own theme handbook draws the line. Both reach into core through hooks, named points in the WordPress code where other code is allowed to run, and hooks come as actions and filters. An action runs at a moment, such as a post being saved, and returns nothing. A filter receives a value, changes it and hands it back. This one shortens automatic post excerpts from the default 55 words to 30:

```
add_filter( 'excerpt_length', function ( $length ) {
    return 30;
} );
```

Themes have changed shape over the years. Classic themes use the PHP template system that arrived with WordPress 1.5 in 2005, and they are still in wide use. Block themes are the current approach: their templates are HTML files built from blocks, editable in the Site Editor, with colours, fonts and spacing held in a theme.json file. Hybrid themes are classic themes that have taken on some of the block features.

Content also leaves the site as JSON through the REST API, which the block editor itself is built on. A request to the site's /wp-json/ address returns a map of every route it offers, public content is readable without signing in, and anything private needs authentication. Any program can fetch the three latest posts like this:

```
curl -s 'https://example.com/wp-json/wp/v2/posts?per_page=3'
```

To run it, the project recommends PHP 8.3 or greater, MySQL 8.0 or MariaDB 10.11 or greater, and HTTPS, preferably on Apache or Nginx. Joomla and Drupal are content management systems of the same kind, and the two other routes to weigh it against are a custom application on a framework such as Django and a static site generator.

Use the concept in a decision

Put "What is WordPress?" to work in a real decision. We can help relate the concept to the system you are building or buying. A site that is quick, findable and editable by your own staff, without a bill for every change of wording.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Developed 100 web applications using HTML/HTML5 — 100 web applications built

Developed 100 web applications using HTML/HTML5 Get in touch →

Why it matters

The real decision behind WordPress is who changes the site after launch, and who keeps it safe while they do. WordPress hands non-technical staff a complete editor: they can publish a page, swap an image, add a user or rework a menu without opening a ticket. On a site that changes every week, the cheapest edit is the one that needs no developer.

The licence is free, but the site is not. The GPL covers the software, and wordpress.org takes the view that plugins and themes are derivative works that inherit the same licence. The spending moves elsewhere: hosting that runs PHP and a database, the hours to choose, set up and test a theme, and the ongoing work of keeping core, plugins, themes and the PHP version current. That last cost runs for the life of the site, and a quote should name who carries it.

The upkeep follows from WordPress being a live application. Uncached page generation runs PHP against a database; a full-page cache can serve a saved page. Plugin code runs with the same access as core, and the login screen sits on the open internet unless someone puts a second lock in front of it. Background core updates arrived in version 3.7. New installations from 5.6 onward enable both minor and major core updates by default, subject to configuration and update eligibility; older installations retain their previous settings unless changed. PHP keeps a separate update schedule.

A custom build fits data and workflows that look nothing like pages and posts, and its editing workflows need to be built and maintained. A static build removes the PHP and database application from the request path, while its server and build tools still need upkeep. Its editors usually work through files and a build step. WordPress fits the wide middle: a fairly standard site that its owners will run themselves, where the editing screens are worth the upkeep they bring. Lock-in stays modest, because the code is GPL, the content sits in an ordinary SQL database, and the REST API serves posts, pages and taxonomies as JSON.

What it looks like in practice

In this company's work WordPress is one answer among several, picked on purpose. The write-up of a hundred web applications covers 2003–2014 and states the rule behind the choice. When a client needed something custom, the build was HTML/HTML5, CSS/SCSS and Django. When a client needed a more standard site that they could also look after themselves, WordPress or Joomla got there sooner and made more sense.

What that write-up values is the judgement, not the tool. Part of the work it describes was telling the two cases apart: talking a client out of a bespoke build they did not need, and into one when the need was real.

Put as questions a buyer can answer before asking for any quote, the judgement looks like this:

- Will people without a developer edit the site every week, or will it change a few times a year?
- Does the content fit pages, posts, categories and media, or does it need its own data model and rules?
- Who will keep core, plugins and PHP current for as long as the site is live?

- Will the site lean on a handful of well-kept plugins, or on dozens of them stitched together to imitate an application?

Weekly edits, content that fits, a named maintainer and a short plugin list point to WordPress or a CMS like it. Unusual data, or nobody to keep it current, points to a custom build or a static one.

Where it goes wrong

A routine update breaks the site

Someone updates a plugin, and a page starts showing an error or the dashboard refuses to load. The site had gathered plugins for years, some of them abandoned by their authors, each one third-party PHP running with full access, and two of them clashed. Keep the list short, and delete every plugin the site has stopped using, as the hardening handbook asks. Take a backup before every update, try major ones on a copy first, and prefer plugins whose authors still ship releases.

Custom changes vanish after an update or a redesign

A tweak that worked for months disappears the day the theme updates, or a feature the business relies on goes missing when the site gets a new look. In the first case someone edited the files of a theme they did not write, and the update replaced them; the theme handbook warns that a function added straight into a theme's functions.php disappears with its next update. In the second, site-critical behaviour lived in the theme, so it left with the theme. Put changes to someone else's theme in a child theme, and put features in a small plugin of your own that hooks into core, so the theme carries presentation and nothing else.

An admin password turns into code on the server

Whoever steals one administrator's password can go from editing posts to running their own PHP, because the dashboard ships with an editor for theme and plugin files. One line in wp-config.php switches that editor off, which has the same effect as removing the edit_themes, edit_plugins and edit_files capabilities from every user:

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

It closes one route among several, and the handbook says plainly that it will not stop an attacker who can upload malicious files. Pair it with strong passwords, two-step authentication for every administrator, and as few administrator accounts as the work allows.

The site falls behind the current release

The dashboard shows a new-version banner for months while the site runs a major release, or a PHP version, well behind the current one. A site configured for minor core updates only can keep patching itself while its major version and PHP fall behind, until a plugin it depends on demands something newer. New installations from 5.6 also enable major core updates by default, but configuration can change that. Put major updates on a calendar, check the PHP version against the project's requirements page at the same time, and test both on a copy before the live site.

Further reading

- WordPress Documentation — the project's manual for running a site, from publishing and media to installing and maintaining it.
- Requirements — the PHP, MySQL and MariaDB versions the project recommends today.
- Upgrading WordPress — background, one-click and manual updates, with the backup to take first.
- Hardening WordPress — file permissions, database privileges, the switch for the file editor and plugin hygiene.
- Plugin Handbook: Hooks — how actions and filters let code change WordPress without touching core.
- REST API Handbook — routes, endpoints and authentication for the API the block editor is built on.



Glossary

Choose for your actual constraints

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is WordPress?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/wordpress/