WP engine™

Migrating From Drupal to WordPress

A guide for all skill levels

Introduction

Migrating your Drupal site to WordPress® might seem daunting, but there are several methods you can use to make the process more manageable—whether you choose to do it yourself or hire professional services.[1]

> If you plan to migrate content from Drupal to WordPress on your own, you'll need a strong technical background—ranging from intermediate to advanced skills—and should be prepared to invest a significant amount of time in the migration process and post-migration cleanup.

For those with less technical expertise, it may be more efficient to use a professional migration service. While these services come with a cost, they can save you considerable effort and ensure a seamless transition from Drupal to WordPress.

Before we dive into the specifics, let's explore some key reasons why you might consider migrating from Drupal to WordPress.

Table of Contents

[1] *WP Engine is a proud member and supporter of the community of WordPress® users. The WordPress® trademarks are the intellectual property of the WordPress Foundation, and the Woo® and WooCommerce® trademarks are the intellectual property of WooCommerce, Inc. Uses of the WordPress®, Woo®, and WooCommerce® trademarks in this website are for identification purposes only and do not imply an endorsement by WordPress Foundation or WooCommerce, Inc. WP Engine is not endorsed or owned by, or affiliated with, the WordPress Foundation or WooCommerce, Inc.*

Why migrate from Drupal to WordPress?

WordPress is the #1 CMS

WordPress is the world's leading content management system (CMS), powering over 40% of all websites globally and holding more than 60% of the CMS market share.

WordPress' scalability, flexibility, and extensive developer community make it a top choice for businesses of all sizes.

Major brands like UBER, Siemens, Hallmark, and National Geographic all use WordPress, and it continues to dominate because of its vast ecosystem of nearly 60,000 plugins and 10,000+ themes. Additionally, WordPress has a highly intuitive interface that simplifies content management for non-technical users, whether you're building a simple brochure site or a complex eCommerce platform.

WordPress supports high-traffic, complex sites

WordPress powers some of the world's most-visited websites, efficiently handling intense traffic and complex requirements. In fact, over 40% of the top 1 million websites, including 34% of the top 10,000, run on WordPress. Its scalability ensures reliable performance even during traffic spikes, making it an ideal choice for eCommerce, media, and enterprise-level sites. Businesses trust WordPress for its proven ability to support high-traffic demands and sophisticated site architectures.

Strong community support and a wide ecosystem of extensions

As the world's leading CMS, WordPress benefits from an active global community, which includes thousands of designers, developers, and innovative users.

Because WordPress is open source, millions of users worldwide continuously enhance its core codebase and documentation, driving ongoing improvements and empowering the WordPress user base with valuable knowledge.

Moreover, WordPress' vast ecosystem—including thousands of plugins, themes, and specialized services—provides unmatched flexibility and scalability, enabling users to tailor their digital experiences efficiently.

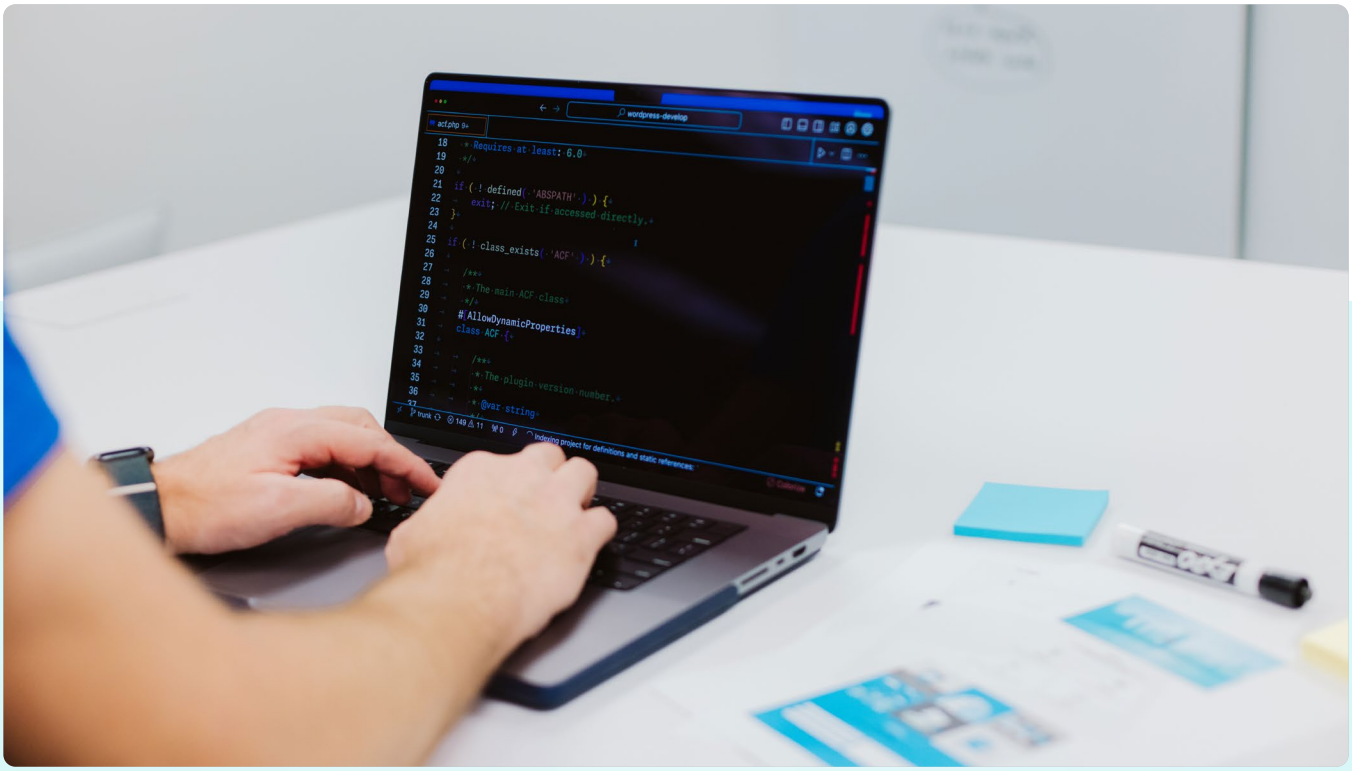
User-friendly interface

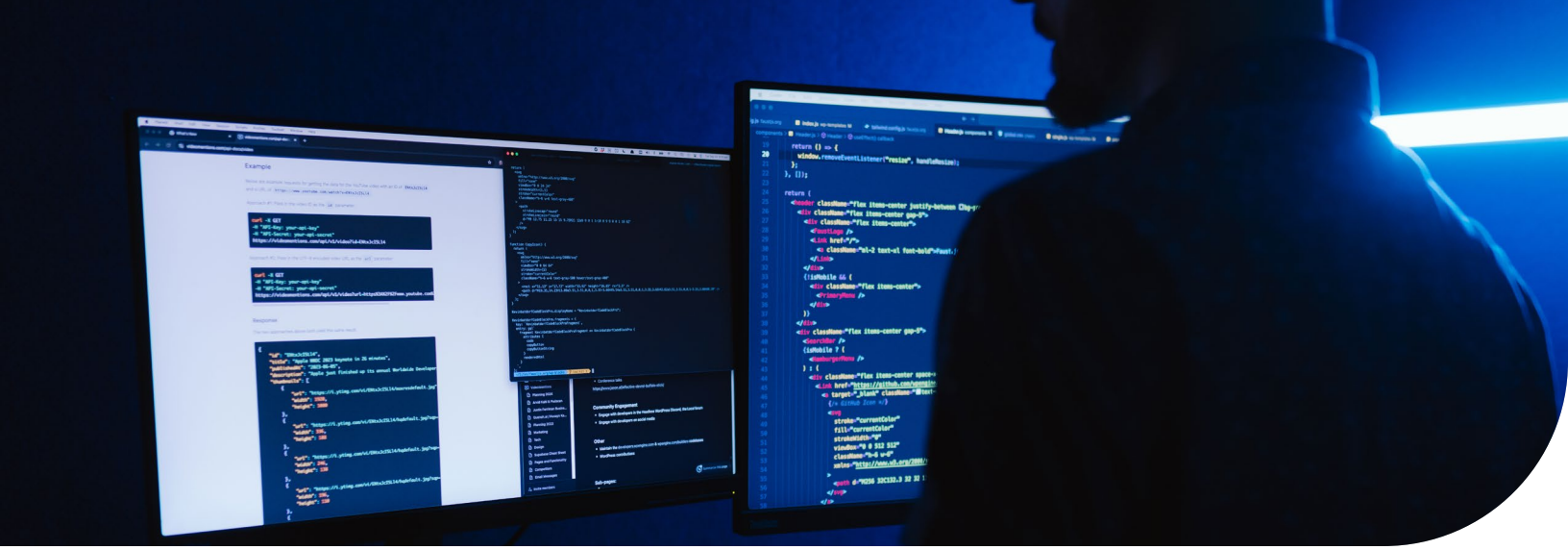
Unlike Drupal, which requires a deeper level of technical skill, WordPress is known for its user-friendly interface. Even if you have no coding experience, you can easily create and manage content using the platform's intuitive tools.

WordPress is built for an evolving tech landscape

WordPress' continuous updates and commitment to backward compatibility ensure it stays aligned with modern development needs. By integrating technologies like the REST API and GraphQL, WordPress facilitates seamless customization and integration with contemporary frameworks, making it a flexible and scalable choice for businesses.

Now that we've discussed some of the key reasons to use WordPress, here are a few terminology differences between it and Drupal.





Drupal vs. WordPress: Terminology differences

Migrating from Drupal to WordPress requires understanding some terminology differences, especially in newer versions of Drupal. Here's a comparison:

Drupal	WordPress
Entities	Posts + Custom Post Types
Fields	Custom Fields (Meta Fields)
Blocks	Blocks (Gutenberg)
Modules	Plugins
Taxonomy Terms	Categories + Tags
Views	Custom Queries + Page Builders
Configuration YAML	No Direct Equivalent (Handled via Plugins/Settings)

Entities: In Drupal, entities are the fundamental data types used to structure content and configuration, such as nodes, taxonomy terms, and users. In WordPress, this concept translates to posts and custom post types, which organize content into different formats like blog posts, pages, or custom content types for specialized data.

Fields: Drupal fields add metadata to content types, such as titles, images, or custom data. In WordPress, custom fields (or meta fields) serve a similar purpose, enabling the addition of metadata to posts, pages, and users. Plugins like Advanced Custom Fields (ACF®) make managing custom fields in WordPress more flexible.

Blocks: In Drupal, blocks are used to place content elements in various regions of a site, like sidebars or headers. In WordPress, the Gutenberg block editor (introduced in version 5.0) provides a drag-and-drop interface for arranging blocks of content, such as text, images, and widgets, anywhere on the page.

Modules: Drupal modules are extensions that add or modify functionality on a Drupal site, similar to WordPress plugins. WordPress plugins allow users to add features and enhance their site's capabilities without needing to write custom code.

Taxonomy terms: In Drupal, taxonomy terms organize content into vocabularies. In WordPress, this translates to categories and tags, which provide a simpler way to organize and classify content.

Views: Drupal views are a powerful tool for creating complex queries and displaying content in various formats, such as lists, tables, or grids, without writing code. In WordPress, custom queries achieve similar results, often paired with page builders like Elementor or Beaver Builder for visual content display.

Configuration YAML: Drupal 8+ uses YAML files to store configuration settings, making it easy to export, import, and synchronize configurations between different environments. WordPress doesn't have a direct equivalent for this; instead, configuration settings are managed through the WordPress admin interface or plugins, and any site configuration is often exported manually.

This comparison of terms should help users migrating from Drupal to WordPress better understand the terminology differences.

Checklist before you migrate

Before you migrate your site, there are certain things you need to take care of to ensure it goes smoothly.

Use this checklist as a handy guide:

- **Create a backup of your site**
- **Take inventory of your content**
- **Establish a plan**
- **Prepare for SEO preservation**
- **Allot time for the migration**

Here are a few key things to keep in mind before starting your migration from Drupal to WordPress.

Create a backup of your site. Before you begin the migration, it's crucial to create a complete backup of your Drupal site to prevent any potential data loss. If you already have a WordPress site, don't forget to back that up as well to protect your existing content.

Take inventory of your content. Review your site's content and create a simple spreadsheet to keep track of what you want to move and what you can leave behind. If needed, there are various services that can assist you in creating a thorough content inventory.

Establish a plan. Next, take the time to define your migration goals. Decide if you want to keep your current design or make updates, and consider whether your URL structure will stay the same or change. Your plan should clearly outline everything that needs to be migrated, such as your domain name, content, design elements, site architecture, and any hosting requirements. Having a well-thought-out plan will make the migration more organized and less stressful.

Prepare for SEO *preservation.* Preserving your SEO rankings is one of the most critical aspects of a *successful migration*, as any missteps can negatively impact your search engine visibility.

To safeguard your SEO, use tools like Xenu Link Sleuth to crawl your site and export URLs, which will give you a complete picture of your current site architecture. If you're not confident in your technical skills, consider contacting a professional migration service to learn about their strategies for maintaining SEO during the transition.

Allot time for the migration. The amount of time required for your migration will depend on the size and complexity of your site. To minimize disruption, try to schedule the migration during a low-traffic period for your business. This will give you the flexibility to handle the process thoroughly and efficiently, reducing the risk of any major issues.

How to migrate from Drupal to WordPress

There's no single correct approach for migrating your site from Drupal to WordPress. However, here are some suggested methods to consider, depending on your level of technical expertise.

Your technical expertise: Beginner

Solution #1: Migration services

If your coding skills are limited, consider hiring a service to handle the migration instead of struggling with the technical complexities discussed further down. Many online services specialize in migrations from Drupal to WordPress and can offer you peace of mind that the move will be seamless.

When choosing a migration service, consider factors like budget, site complexity, and whether the service offers SEO preservation and custom content type support.

Here are a few services worth checking out:

- **Another Cup of Coffee** offers a Drupal to WordPress migration tool and additional migration services to help move your site from Drupal to WordPress. They migrate pages, stories, taxonomies, users, and comments, with options for additional migration features. anothercoffee.net
- **gConverter** provides access to experienced developers who can migrate your Drupal site to WordPress. Services include Drupal to WordPress conversion of content, modules, and templates. www.gconverters.com

- **WordHerd** is a well-known migration provider offering comprehensive Drupal to WordPress migration services, including content, themes, and SEO preservation. Migration timelines vary based on project scope, ensuring a tailored approach to meet your site's needs. www.wordherd.io
- **WordPrax** will migrate your Drupal site to WordPress with services including responsive templates, SEO optimization, widget-ready frameworks, and versatility in themes. www.wordprax.com
- **WordSuccor** can help you migrate your Drupal site and provide responsive templates, SEO optimization, and more. You can inquire about a custom migration plan. www.wordsuccor.com

Solution #2: Teach yourself

If hiring a professional migration service isn't an option, you can still migrate from Drupal to WordPress with the right resources. While it may seem challenging, there are plenty of accessible learning options to guide you.

You can start with video tutorials and courses, which are ideal for visual learners. Platforms like Udemy offer comprehensive guides such as How to Migrate from Drupal to WordPress, with step-by-step instructions and practical examples.

For free resources, explore WordPress-focused YouTube channels that cover migration, custom post types, and troubleshooting, all in easy-to-follow videos.

For deeper learning, consult online documentation. The WordPress Codex provides essential guidance on setting up and managing your site, while the Drupal.org Migration Documentation offers insights into using the Migrate API, useful for content extraction. Engaging with the WordPress Support Forums is also helpful—you can ask questions and learn from developers experienced in similar migrations.

Lastly, hands-on practice is vital. Tools like Local allow you to safely experiment with the migration process in a local environment. Alternatively, you can use XAMPP or WAMP to set up a local server and test database imports, content transfers, and configurations without impacting your live site.

Your technical expertise: Intermediate to strong

If you have a solid understanding of code, then manually migrating your Drupal site to WordPress is a relatively costless way to move your site.

Solution: Manual migration using queries

If you're comfortable with SQL and working with databases, you can attempt a manual migration. This method allows for greater customization but requires a solid understanding of Drupal and WordPress data structures.

Step-by-step manual migration:

1. **Backup:** Make a backup of both your Drupal and WordPress databases to avoid data loss.
2. **Prepare for the migration:**
    a. Ensure your Drupal taxonomies are correctly labeled.
    b. Set up a new WordPress installation in a different database from your Drupal site.

3. **Clear WordPress Tables.** Clear out previous content from the WordPress database by running the following command using a database management tool such as phpMyAdmin or Adminer.

   Make sure you review and customize these SQL queries based on your specific Drupal setup. Running queries without adjustments could result in incomplete or incorrect data migration.

```
TRUNCATE TABLE wordpress.wp_comments;
TRUNCATE TABLE wordpress.wp_links;
TRUNCATE TABLE wordpress.wp_postmeta;
TRUNCATE TABLE wordpress.wp_posts;
TRUNCATE TABLE wordpress.wp_term_relationships;
TRUNCATE TABLE wordpress.wp_term_taxonomy;
TRUNCATE TABLE wordpress.wp_terms;
```

   Note: Ensure this step is performed only on a new or non-critical WordPress installation, as it will permanently delete existing data. If there's a chance of needing the current data, make sure to create a backup.

4. **Migrate users.** Apply the following code to convert over multiple users. Before proceeding with this step, consider adding a safeguard or confirmation step to ensure you don't accidentally delete important user data. Also, it's important to verify that user ID 1 is indeed the admin.

```
DELETE FROM wordpress.wp_users WHERE ID > 1;
DELETE FROM wordpress.wp_usermeta WHERE user_id > 1;
```

5. **Migrate taxonomy terms.** You can use the following code to migrate over taxonomy terms. To ensure duplicate names don't get lost, it's important to ensure taxonomies are clean of duplicate names before running.

```
REPLACE INTO wordpress.wp_terms
(term_id, `name`, slug, term_group)
SELECT DISTINCT
d.tid, d.name, REPLACE(LOWER(d.name), ' ', '_'), 0
FROM drupal.taxonomy_term_field_data d;
INSERT INTO wordpress.wp_term_taxonomy
(term_id, taxonomy, description, parent)
SELECT DISTINCT
d.tid, 'post_tag', d.description, 0
FROM drupal.taxonomy_term_field_data d;
```

6. **Migrate posts and pages.** To move over posts and pages, apply the following query:

```
INSERT INTO wordpress.wp_posts
(id, post_author, post_date, post_content, post_title, post_excerpt,
post_name, post_modified, post_type, post_status)
SELECT DISTINCT
n.nid, n.uid FROM_UNIXTIME(n.created)  b.body_value, n.title,
b.body_summary, n.uuid,
FROM_UNIXTIME(n.changed) ‘page’, IF(n.status = 1, 'publish', 'private')
FROM drupal.node n
INNER JOIN drupal.node__body b ON n.nid = b.entity_id
WHERE n.type IN ('page', 'article');
```

If your Drupal installation includes multiple post types, make sure to list each one explicitly in the `WHERE` clause like this: `WHERE n.type IN('post', 'page', 'blog', 'custom_type1', 'custom_type2')`.

Omitting any content types will result in incomplete migration, leaving some posts behind. Additionally, review how the IF conditions handle post statuses and URL generation. Edge cases, such as URLs with special characters or custom content types, may require extra handling.

It’s important to test thoroughly with all your content types and post statuses to ensure proper data conversion. If you encounter issues with slug formatting or URL inconsistencies, consider implementing a dedicated function for URL sanitization.

7. **Combine post types in WordPress.** You can run the following script to combine post types in WordPress but double-check that the term_taxonomy_id values align correctly with the terms imported earlier. If the counts seem off, consider adding a check or using a debugging query to verify relationships.

```
UPDATE wordpress.wp_posts
SET post_type = 'post'
WHERE post_type IN ('blog');
```

8. **Migrate comments.** Apply this query to migrate comments:

```
INSERT INTO wordpress.wp_comments
(comment_post_ID, comment_date, comment_content, comment_parent,
comment_author, comment_author_email, comment_author_url, comment_approved)
SELECT DISTINCT
nid, FROM_UNIXTIME(timestamp), comment, thread, name, mail, homepage,
((status + 1) % 2) FROM drupal.comment_field_data;

UPDATE wordpress.wp_posts
SET `comment_count` = (
SELECT COUNT(`comment_post_id`)
FROM wordpress.wp_comments
WHERE wordpress.wp_posts.id = wordpress.wp_ comments.comment_post_id
);
```

9. **Update image URLs.** If you want to keep your Drupal images and files in the same location, you can skip this step, but if you're FTP-ing your files to the uploads folder in your WordPress wp-content folder, use the following code to fix the image URLs.

```
UPDATE wordpress.wp_posts
SET post_content = REPLACE(post_content, '/sites/default/files/', '/wp-content/
uploads/');
```

Note: This only works if the images have been correctly transferred to the wp-content/uploads folder. If paths differ, consider adding more comprehensive replacements. If your site has many image path variations, you might consider using a more robust search-and-replace tool.

10. Fix taxonomy relationships. To fix taxonomy relationships (assuming they were set up correctly in your original Drupal site), use the following code:

```
UPDATE IGNORE wordpress.wp_term_relationships, wordpress.wp_term_taxonomy
SET wordpress.wp_term_relationships.term_taxonomy_id = wordpress.wp_term_
taxonomy.term_taxonomy_id
WHERE wordpress.wp_term_relationships.term_taxonomy_id =
wordpress.wp_term_taxonomy.term_id;
```

Note: Using IGNORE may skip errors, but it can also hide potential issues. Only use IGNORE if you understand the consequences. Run a test query without IGNORE to identify problems first.

Using the Drupal Migrate API for data export

The Migrate API is a powerful tool for users running Drupal 8+. It's designed to simplify and structure the migration process.

Unlike manual methods, the Migrate API provides a more systematic approach to exporting your content, reducing the risk of data loss and making it easier to map content accurately to WordPress.

The Migrate API allows you to create detailed configurations for how content should be exported, using YAML files to define the source data and specify how it should be transformed and imported.

This is especially useful if your site has complex content types or if you need to perform a highly customized migration. The API is flexible, reliable, and particularly suited for developers or advanced users who want fine-grained control over their data migration.

Follow the steps below to get started with the Drupal Migrate API:

1. **Enable necessary modules:**
   First, you need to activate the required modules to prepare your Drupal site for migration. This will set the groundwork for using the Migrate API. Run the following Drush command:

   ```
   drush en migrate migrate_drupal
   ```

   This command ensures that the core Migrate and Migrate Drupal modules are enabled, giving you access to all the migration features.

2. 2. Create YAML migration configurations
Next, define how your content should be migrated by creating a YAML file with your migration settings. Here's an example configuration for migrating articles:

```
id: migrate_articles
  label: 'Migrate Articles from Drupal'
  source:
    plugin: d7_node
    node_type: article
  process:
    title: title
    body: body/value
    created: created
    changed: changed
  destination:
    plugin: entity:node
    default_bundle: post
  migration_dependencies: null
```

This YAML file specifies that "article" nodes will be migrated, mapping fields like the title and body content. You can customize it further to suit your site's content structure.

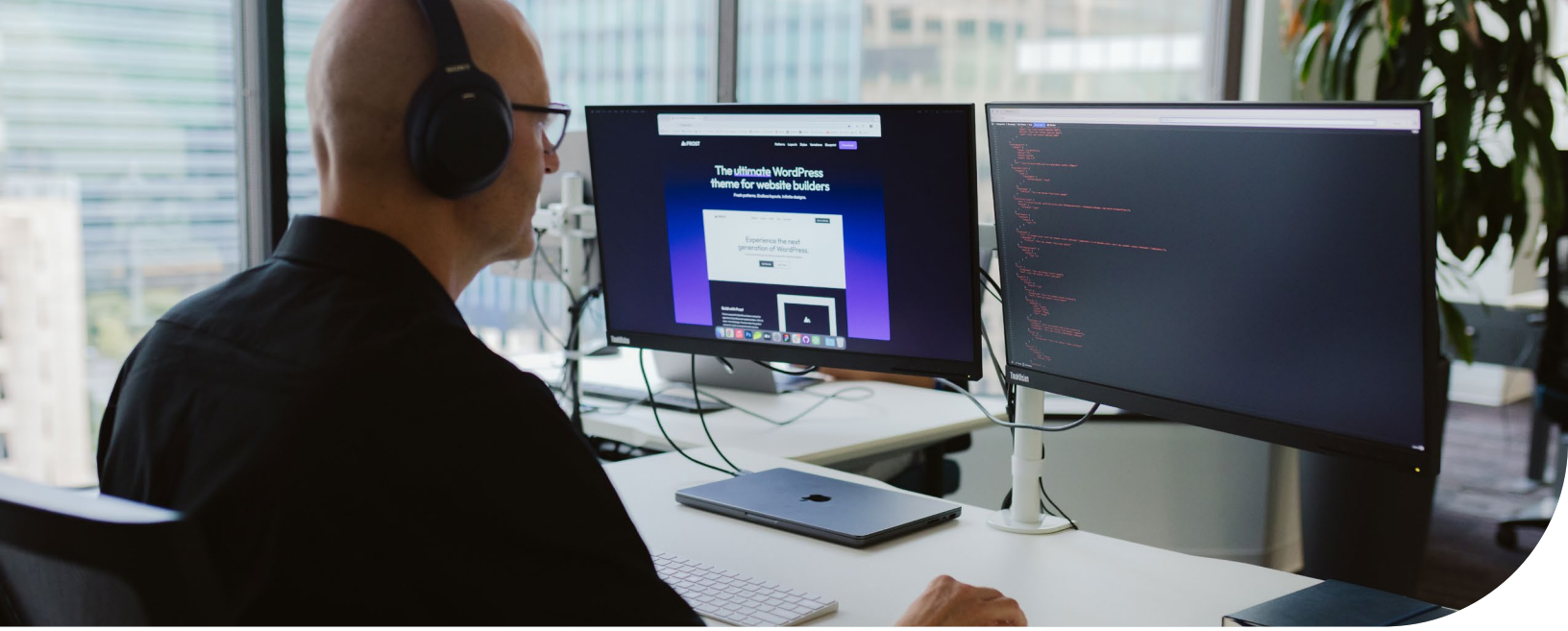
Note: If your site includes custom content types, be prepared to adjust the YAML configuration files accordingly. You may need to map additional fields to ensure a successful migration.

3. Run migration using Drush
Finally, execute the migration using Drush to start transferring your data to WordPress. Use this command:

```
drush migrate-import
--group=my_migration_group
```

This command initiates the migration process, using your defined YAML configuration to import content efficiently and accurately.

By following these steps, you'll be able to manage your content migration in a structured and reliable way. The Migrate API offers the flexibility needed for complex migrations, making it an invaluable tool for developers and advanced users.



Simplified examples for common scenarios

Migrating from Drupal to WordPress can be complicated, but breaking down the process into simpler steps makes it far more manageable.

Here are some common scenarios you might encounter and a straightforward approach for handling each one:

Migrating custom content types

If your Drupal site has custom content types like “Events” or “Products,” you’ll need to replicate these in WordPress.

Start by creating your custom post types using a plugin like Custom Post Type UI. This tool allows you to define each post type’s settings and behavior, making it easy to tailor them to your needs.

Once your custom post types are set up, you’ll need to add fields to capture all the data from your Drupal site. This is where Advanced Custom Fields (ACF) comes in. ACF lets you add custom fields such as dates, images, or locations, making your content types as versatile as they were in Drupal.

Finally, to import your content seamlessly, consider using WP All Import. This plugin makes it simple to map your data from a CSV or XML file into the corresponding fields in WordPress. With a bit of planning, you can ensure that your content is transferred accurately and efficiently.

Migrating taxonomies

Taxonomies are essential for organizing content, and Drupal’s taxonomy system is particularly robust. In WordPress, you’ll need to recreate these taxonomies to maintain your content structure.

You can start by using the Custom Post Type UI plugin to set up categories, tags, or any custom taxonomies you have in Drupal. After setting up your taxonomies, import your terms and link them to the appropriate content.

This step may require some manual effort, but it's crucial for keeping your site's organization intact. Once everything is set up, double-check that the relationships between content and taxonomies are preserved to avoid any broken connections.

Handling media files

Media files, such as images and documents, are often scattered throughout your Drupal site. To migrate these files, start by exporting all media from the /sites/default/files/ directory on your Drupal server.

Once you have your media files, upload them to the wp-content/uploads/ directory in your WordPress installation.

Next, you'll need to update any URLs in your content that point to the old media locations. The Better Search Replace plugin is perfect for this task. It allows you to search for the old file paths and replace them with the new paths, ensuring that your images and media load correctly on your WordPress site.

Problem-solving tips

Even with careful planning, issues can arise during migration. Here are some tips to tackle common problems:

Issues with categories and tags
If you notice that your categories and tags aren't showing up as expected, it may be because your taxonomies weren't labeled correctly in Drupal before migration. Double-check your taxonomy setup and make adjustments as needed to ensure everything is properly mapped.

Duplicate errors
Sometimes, the migration process may be interrupted, leading to duplicate errors. In these cases, identify where the import stopped and resume from that point, ensuring that you don't re-import data unnecessarily.

Redirect problems
URL redirection is a common issue after migration. If users are being redirected to the homepage instead of the correct pages, you can fix this by adding a snippet to your wp-config.php file:

```
  if (defined('WP_HOME') &&
isset($_SERVER['REQUEST_URI'])) {
      $_SERVER['REQUEST_URI'] =
str_replace('/index.php', '',
$_SERVER['REQUEST_URI']);
  }
```

This code snippet ensures that permalinks work correctly, directing visitors to the appropriate content.

By following these examples and tips, you can navigate your Drupal to WordPress migration with greater ease and confidence, ensuring your site remains functional and well-organized.

Remember, taking your time to plan and test each step will pay off in the long run, resulting in a smoother transition to your new WordPress home.

Final thoughts

Migrating from Drupal to WordPress is a significant project that requires thoughtful planning and execution. However, the advantages—such as a more intuitive interface, extensive plugin options, and robust community support—make it a rewarding endeavor.

A successful migration can elevate your digital presence, whether you want greater flexibility or a user-friendly content management experience.

Approach your migration with a clear plan: start by inventorying your content, outline your goals, and prioritize SEO preservation to maintain search engine rankings. If the technical challenges feel overwhelming, consider professional migration services for a seamless transition.

Be prepared for common problems, such as broken taxonomies or URL redirect errors, and use the troubleshooting strategies outlined in this guide. Flexibility and preparation are key. Leverage WordPress's community resources, experiment with different solutions, and adjust as needed.

After the migration, thoroughly test your site to ensure everything works as expected. Monitor your site's performance, set up security measures, and optimize for SEO to make the most of your new WordPress environment.

For further optimizations, ironclad security, and award-winning support, WP Engine's managed hosting for WordPress offers valuable solutions for sites of all sizes.

Remember, this isn't just a technical task—it's an investment in your website's future. With careful planning, your migration will create a more dynamic and efficient online experience!

WP Engine empowers companies and agencies of all sizes to build, power, manage, and optimize their WordPress websites and applications with confidence.

Serving 1.5 million customers across 150+ countries, the global technology company provides premium, enterprise-grade solutions, tools, and services, including specialized platforms for WordPress, industry-tailored eCommerce and agency solution suites, and developer-centric tools like Local, Advanced Custom Fields, and more. WP Engine's innovative technology and industry-leading expertise are why 8% of the web visits a WP Engine-powered site daily. Learn more at wpengine.com.