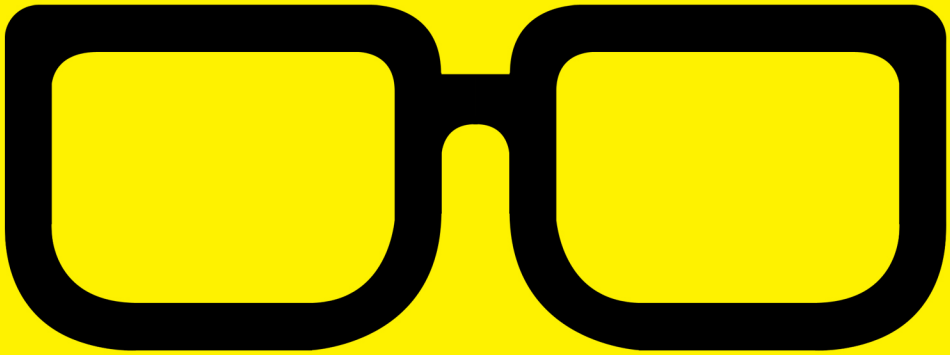
SPONSORED BY

Symantec™ | Website Security Solutions

GEEK GUIDE



Drupal 8 Migration Guide

LINUX™ JOURNAL

drupalize me

Table of Contents

Introduction to Migrations with Drupal 8 6

What is a migration? 6
Upgrade, update, or migrate? 7
Migrate from anywhere 8
Migrate understands Drupal 9
Execute, rollback, and debug with ease 11
Beware version numbers 12
Where should you start? 12
Additional resources 13

Preparing for a Migration 13

Why are you doing this? 14
Consider your content 14
Do you really need that feature? 16
Is it time for a face lift? 16
What is this all going to cost me? 17
Acceptance testing 20
Additional resources 20

Migrate System: Terms and Concepts 20

Migration 22
Drupal-to-Drupal Migration 22
Migration plugins 23
Extract, Transform, Load 23
Source plugins 24
Process plugins 25
Destination plugins 25
Upgrade path 25
Execute and rollback 25
Highwater marks 26
Additional resources 26

GEEK GUIDES:
Mission-critical information for the most technical people on the planet.

Drupal 8 Migration Guide

Courtesy of Drupalize.Me

With the recent release of Drupal 8.1.x (https://www.drupal.org/project/drupal/releases/8.1.0) and the inclusion of a minimal Drupal-to-Drupal migration UI in Core, migrating older sites to Drupal 8 is a topic that's on a lot of people's minds. Having done a number of migrations, and helped to improve the migration system along the way, we know firsthand that planning, preparing, and executing a successful migration can be a lot of work. We're hoping to

make it as easy as possible by sharing what we know and helping others get off on the right foot.

In this ebook, we'll take a look at the new Migrate API in Drupal 8 and cover some concepts and terminology. For information on where to find documentation and examples, make a migration plan, evaluate and prepare your source Drupal site, look at the most useful contributed modules, and finally execute a Drupal-to-Drupal migration via both the UI and Drush, see Drupalize.Me's guide at https://drupalize.me/series/drupal-8-migration-guide.

Topics planned for the Drupalize.Me migration guide include:

- Performing selective Drupal-to-Drupal migrations.
- Writing a migration plugin.
- Writing custom source, process, and destination plugins.
- Migrating from non-Drupal sources like CSV files, JSON, and custom database schemas.
- Automating processes in order to make testing and development of a migration easier.
- Handling images and other files.
- Making sure the Drupal 7 modules you maintain have a migration path.

During the process of creating this series, we've been actively involved in helping both core and contributed module development both through work that Will has done in the issue queue and by helping provide time for Mike Ryan (one of the primary architects of the system) to continue to make the tools better.

As with all of our Drupal 8 material, we're committed to keeping these tutorials up-to-date as work progresses. We'll keep an eye on the changes that are happening both in Core and with the main contributed modules and make adjustments as needed.

Introduction to Migrations with Drupal 8

Whether you're updating from Drupal 6 or Drupal 7, or importing data from some other source, you need to know about the migrate system in Drupal 8. This section provides an overview and links to additional tutorials where you can learn more about how all the individual parts work.

By the end of this section, you should have a better understanding of what the migration system is capable of and know where to find more information about how to use it.

What is a migration? We use the word "migration" as a generic term for any process that seeks to take data from some source external to the current Drupal 8 site and use it to automatically create nodes, users, configuration, and any other component of your site. In short, automating what might otherwise be a tedious job of copying and pasting.

Drupal 8 brings a new migration system into Drupal core, with the goal of making it easier to import data from a variety of sources.

Drupal 8 brings a new migration system into Drupal core, with the goal of making it easier to import data from a variety of sources.

The system consists of three core modules: Migrate, Migrate Drupal, and Migrate Drupal UI. Learn more about the role that each fulfills in Drupalize.Me's Core Migration Modules tutorial (https://drupalize.me/tutorial/core-migration-modules?p=2578), as well as plugins contained by other core modules that make use of the migration framework to ensure the content or configuration they handle has a migration path.

Upgrade, update, or migrate? Previous versions of Drupal provided an upgrade mechanism that allowed for in-place version updates, which worked for both major version *upgrades*, and minor version *updates*. While convenient, this method also had some significant downsides. Especially tricky was moving between major versions of Drupal. Users often wanted to preserve their existing content, while making changes to take advantage of new systems—a process that begins to resemble a

migration much more than an in-place upgrade.

So in Drupal 8 there is no direct upgrade from Drupal 6 or 7 to Drupal 8. Instead *upgrading* to Drupal 8 will require you to migrate your site and files from a previous Drupal version to Drupal 8. The migrate system in core aims to make this process as easy as possible. As of right now, the Migrate Drupal and Migrate Drupal UI modules provide a way to connect your Drupal 8 site to your Drupal 6 or 7 source; extract both the content and configuration; transform it into the new format; and then save it into Drupal 8. For example, the Migrate Drupal module is smart enough to understand both Drupal 6 and Drupal 8 nodes, and it can extract a Drupal 6 node and all its field data and then save it as a Drupal 8 node. In fact, it's so smart that it'll even take care of migrating the content type definition for you.

You can read more about this change on Drupal.org: https://www.drupal.org/docs/8/upgrade/upgrading-from-drupal-6-or-7-to-drupal-8.

For many people, this is likely what you're looking for: a way to upgrade your older Drupal 6 or Drupal 7 site to the shiny new Drupal 8. The first part of this guide covers the process of preparing for and executing a Drupal-to-Drupal migration. If this sounds like what you're trying to do, check out the "Prepare for a Drupal-to-Drupal Migration" tutorial at https://drupalize.me/tutorial/prepare-drupal-drupal-migration?p=2578.

Migrate from anywhere The migration system makes it possible to pull content into Drupal from just about anywhere. The core API supports extraction from any SQL data source, including previous versions of Drupal.

Contributed modules (https://drupalize.me/tutorial/migration-related-contributed-modules?p=2578) extend this system to support other data types like CSV or JSON, as well as other platforms like WordPress.

Some of the existing data sources include:

- MySQL, MariaDB.
- Previous versions of Drupal.
- CSV.
- JSON, you could even use a REST endpoint.
- XML.
- Etc.

If there isn't already a way to extract the data from your current data store, you can write a custom source plugin. Source plugins are the mechanism that allows the Drupal migration framework to understand the ins and outs of extracting data from different data stores. Source plugins can also be smart about the data store—for example, a WordPress source plugin, written by someone who understands how WordPress works, could be smart enough to update the fields available for extraction dynamically based on the WordPress site in question.

Migrate understands Drupal When you import data into Drupal, you're dealing with entities, fields, and

The migrate system is smart about things like content type configuration. It will automatically import content into whatever fields you've defined for your application's unique information architecture.

configuration. The migrate framework understands how all of these Drupalisms work, making it possible to save an array of content as a new Drupal user account without having to understand the intricacies of Drupal's database schema, field system, or password hashing algorithms.

The migrate system is smart about things like content type configuration. It will automatically import content into whatever fields you've defined for your application's unique information architecture. And it even knows how to validate the content for each field type prior to saving new data.

Some of the things you can create with a migration include:

- Content (nodes, taxonomy, any generic entity) including any attached files and images.
- Content types.
- User accounts.
- Roles and permissions.

- Simple configuration like the site name.
- Complex configuration like image styles.
- Etc.

The majority of the work that you'll do when writing a migration path is creating migration plugins. Migration plugins are responsible for mapping the data extracted from a source to the Drupal definition of that data—for example, mapping the title and sub-title of an article in your previous CMS to the article node type's title and custom sub-title fields in Drupal, and perhaps opting to transform the data during the import using process plugins.

Execute, rollback, and debug with ease In addition to making it possible to write a migration, the system also facilitates executing those migrations. Using either the UI or Drush, you can do things like:

- Execute a complete migration plan.
- Run an individual migration, and its dependencies, without running the complete plan.
- Run a partial migration in order to facilitate testing and debugging.
- Roll back a migration that was previously run to allow for re-running it after making adjustments.

At the moment, Drupal core provides a relatively simple UI for handling Drupal-to-Drupal migrations (https://drupalize.me/tutorial/drupal-drupal-migration-ui?p=2578) and contributed modules doing the bulk of the work in this space. This space is currently the most volatile and subject to change. The tools for running a migration are all rapidly evolving and adding new features. The more people use them, the better they will get. In the future, we will likely see more of this functionality moved into Drupal core as it stabilizes.

Being able to execute migrations with Drush commands is extremely useful as it allows for better automation of processes that can be time-consuming and cumbersome to practice, test, and debug.

Beware version numbers Because of major changes to the Migration API during the Drupal 8.1.x development cycle (https://www.drupal.org/node/2625696), we chose to focus on versions of Drupal core >= 8.1.x throughout this migration guide. At the moment, we are not aware of any similarly large and disruptive changes between 8.1.x and 8.2.x, but since the core migrate modules (https://drupalize.me/tutorial/core-migration-modules?p=2578) are all experimental, they are subject to change. We'll do our best to keep the tutorials on the Drupalize.Me site up to date with the most accurate information available.

Where should you start? There is rarely a single way, or even necessarily a right way, of performing a migration—a fact that makes learning how to perform one especially challenging. Every site is different, and every site grows organically over time. All user-generated content is dirty

and needs massaging. Everyone performs migrations under different circumstances and for different reasons.

So, rather than attempt to define a gold standard for how a migration should be performed, with this guide, we've set out to help provide you with the information you need to make the best decision for your unique migration, and the know-how to be able to address and overcome the hurdles you encounter along the way.

Additional resources

- Migrate documentation from the Drupal.org handbook (Drupal.org): https://www.drupal.org/docs/8/upgrade/upgrading-from-drupal-6-or-7-to-drupal-8.
- Tutorials covering the Drupal 7 migrate module (Drupalize.Me): https://drupalize.me/videos/introduction-migrate-module-series?p=1271. The Drupal 8 migrate framework is an adaptation of the Drupal 7 contributed Migrate module: https://www.drupal.org/project/migrate.

Preparing for a Migration

Planning for a migration is essential. In our experience, we have never once seen someone sit down and execute a migration flawlessly on the first attempt. Migrations involve preparing and analyzing your source data, building a new website that data can be migrated into, and lots of testing, rolling back, and testing again, in order to get everything right. By the end of this guide you should be ready to start planning for your own migration and have a better understanding of what it is you're getting into.

This tutorial doesn't provide a specific set of steps to follow to accomplish the desired outcome. I'm going to pose a bunch of questions and give you some things to consider when planning your migration, but for the most part, we can't actually tell you the answer to the questions. That's your job.

What's more, no two migrations will ever be the same. There are simply too many possible variations, and a human factor, that make the likelihood of ever having a single solution for all possible cases unrealistic. However, that means you're going to learn a ton about Drupal along the way! You can always refer to the Drupalize.Me website where will continue to expand our tutorial library in order to do our best to help you have a successful migration.

A successful migration starts with a lot of careful planning and consideration of all the little details.

Why are you doing this? Taking on a data migration is a big deal. Even for relatively small sites, it can be a time-consuming and challenging project. The benefit is that you've now got a shiny new Drupal 8-powered website. But it is important to take the time to understand your motivations and the benefits that you hope to gain by moving your site to a new home. Even if you already know that the shiny new Drupal 8 features are super cool, you might have some convincing to do when it's time to sell the project to others.

Your existing site already works, and it's not like it's just going to suddenly stop working. So what benefits do you hope to gain by investing time and resources into migrating to a new platform?

Consider your content Prior to performing a migration,

Your existing site already works, and it's not like it's just going to suddenly stop working. So what benefits do you hope to gain by investing time and resources into migrating to a new platform?

you might want to consider doing a content audit. There's no sense spending a bunch of development cycles planning, crafting, testing, and debugging a migration path for content that you don't actually plan to make use of.

In our experience, the two most common things that come up when doing this are:

- Identifying content that is no longer relevant and doesn't need to be migrated.
- Refactoring existing information architecture and chunking large fields of text into more semantic data suitable for use as an API or in other non-HTML contexts.

For more information about content considerations when migrating from one Drupal site to another, see the "Consider your content and content types" section of the Preparing for a Drupal-to-Drupal Migration tutorial: https://drupalize.me/tutorial/prepare-drupal-drupal-migration?p=2578.

Do you really need that feature? Before migrating any data, consider the platform. Unlike an OS upgrade where you hit "upgrade", come back in a few hours, and everything just works, a move to Drupal 8 is likely to be as much a rebuild as it is a migration. Without creating the application in Drupal 8 first, you'll have nowhere to migrate your content to.

What functionality is required, what can be removed, and what new features are you looking to add? A lot of times when you set out to perform a migration, you're moving from one application to another and need to account for the functionality in the previous application that your clients, and stakeholders, have come to expect. But at the same time, if no one is using the "gift certificates" feature of your site, for example, is it really worth the effort to rebuild it and migrate all the data?

Ask yourself and your stakeholders:

- What features of my site are people really using?
- What features are not being used and can be left behind?
- What critical new features will I be able to add as a result of migrating to a new platform?

Is it time for a face lift? This is a great opportunity to give your website a face lift. That might mean simply tweaking a few design elements that you've been meaning to address for a while, finally implementing a responsive layout system, or a complete refresh of the

design. When you migrate from any system (including prior versions of Drupal) to Drupal 8 you're going to end up needing to create a new theme for your site. This is true even if you're building a site without a migration component. Themes are the one thing that is custom for almost every Drupal project.

Creating a new theme by adapting existing design components will almost always be more efficient, simply due to the fact that you've already got some of the CSS and JavaScript written, and you've already done the hard work to figure out where and when to implement certain styles. That said, if you're pondering a design refresh shortly, you might want to consider doing it now, since you'll already be spending considerable resources creating your custom theme.

Want to learn more about creating a custom theme for Drupal 8? Check out our amazing Drupal 8 theming guide at https://drupalize.me/series/drupal-8-theming-guide.

What is this all going to cost me? Whether you're moving from a previous version of Drupal, or another data source entirely, it's important to frame this migration as a rebuild and not simply an upgrade. Doing so paints a more accurate picture of the amount of time and money this is going to take.

A migration is much more than just moving data from one place to another. The source data needs to be cleaned and prepared, the new website where data will end up needs to be created, designs updated, the new site needs to have a theme developed, infrastructure will need updating, tests will need to be written, and more. When

assessing the cost of a migration, we like to treat it as two projects: 1) create the new site, and 2) automate the transfer of content from the old site into the new one.

A successful migration will require at least a subset of the following roles among you and/or your team:

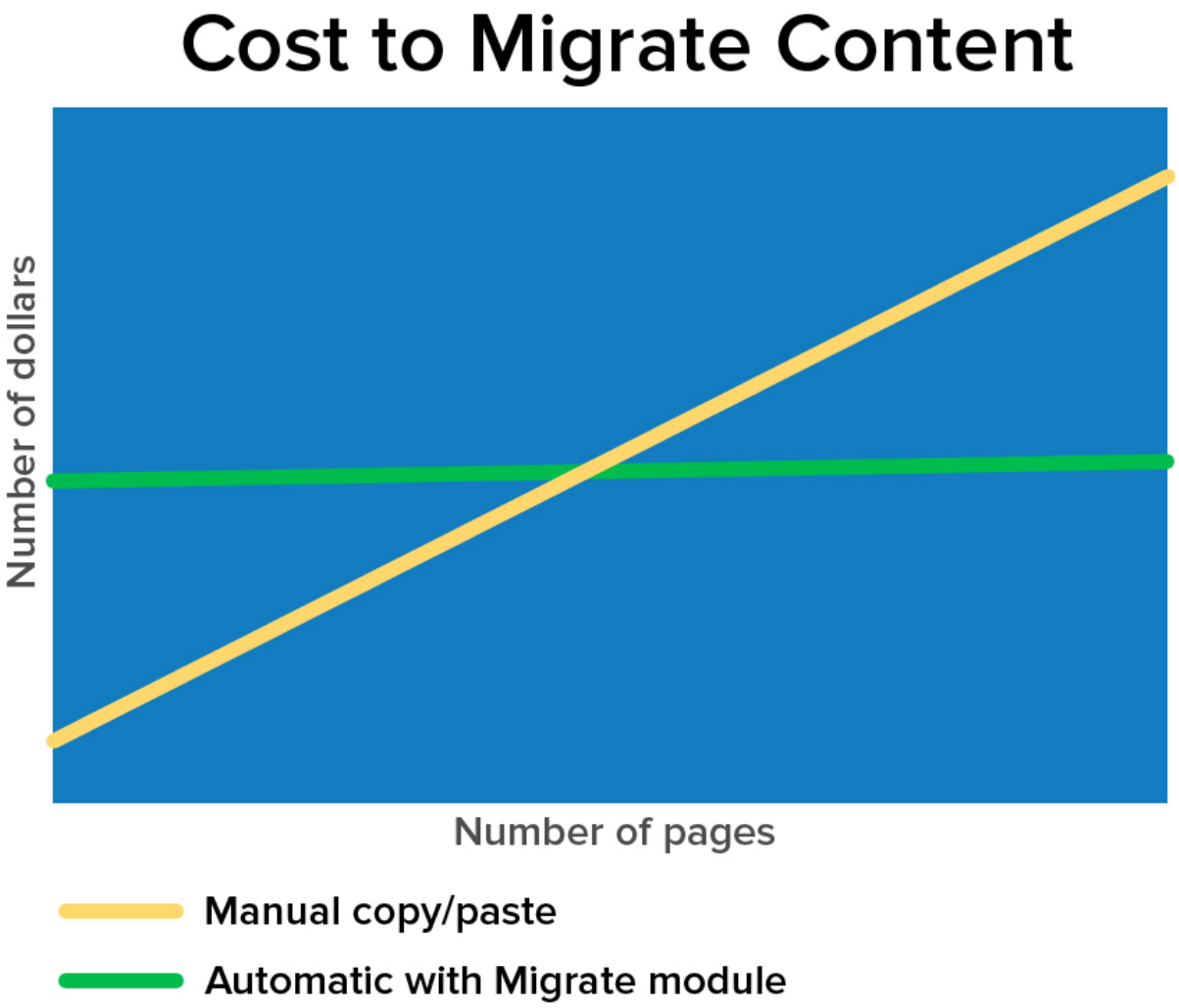
- Someone who can do a content audit. A domain specialist who understands the idiosyncrasies of both your site's functionality and your content.
- A strong understanding of Drupal 8 site-building and best practices.
- Drupal 8 back-end developer.
- Drupal 8 front-end developer.
- Systems Administrator/DevOps.
- Designer.
- Copywriter.
- QA.

Other things to remember:

- Your infrastructure needs will likely change with Drupal 8.
- Staff may need to be retrained how to use the site.

- Any new Drupal site, even if you're "just upgrading", is first and foremost a new website and should be handled as such.

One thing to consider might be whether or not it's worth it to develop a custom migration path. If you've got a limited amount of data, the cost to develop a migration path may be higher than the cost to pay someone to copy/paste content between the old site and the new Drupal 8 one.



For Drupal-to-Drupal migrations, we believe that this will be less of an issue, as a large part of the work in developing migrations and field mappings is handled automatically. But, it's still worth considering.

Acceptance testing What does a successful migration look like, and how are you going to assure yourself and your team that things are complete and ready to be shared with the public? If you've got more then a few hundred pages or so on your site, it's unrealistic to expect someone to go through every single page and confirm that it migrated correctly. But you should test a large enough subset of them that you're confident things are working.

We recommend creating a list of key features of your site—maybe pulling a list of your top pages from Google Analytics—and generating a baseline list of functionality that you'll want to test. This can also serve as a benchmark during the development process. How many of these requirements have you met currently?

Additional resources

- Upgrading to Drupal 8: A Planning Guide (Forum One): https://forumone.com/ideas/upgrading-to-drupal-8-a-planning-guide.
- Upgrading to a Drupal 8 Site: Your Next Migration (Phase2 Technology): https://www.phase2technology.com/blog/upgrading-to-a-drupal-8-site.

Migrate System: Terms and Concepts

To follow along with the rest of the migration tutorials

on the Drupalize.Me site, you'll want to make sure you understand the following concepts and terms as they relate to Drupal migrations.

- What is a migration?
- Migration templates.
- The extract, transform, load process.
- Destinations and sources.
- Additional Drupalisms.

By the end of this tutorial, you should be able to identify the various components that a migration is composed of, and explain at a basic level what each is responsible for:

- Migration.
- Drupal-to-Drupal Migration.
- Migration plugins.
- Extract, Transform, Load.
- Source plugins.
- Process plugins.

- Destination plugins.
- Upgrade path.
- Execute and rollback.
- Highwater marks.

Migration Defining the term "Migration" is complex, and its exact meaning depends on the context in which it's used.

We use to it describe the entire process of importing content to Drupal—for example, "I'm going to perform a Drupal 7 to Drupal 8 migration."

We also use the term migration to represent the definition of how a particular subset of data flows from a source into Drupal—for example, "the Flag module provides a migration path for its own data."

In general, a migration imports data from some alternate source into your Drupal 8 site.

Drupal-to-Drupal Migration Unlike previous versions of Drupal, there is no direct upgrade path for Drupal 6 or 7 to Drupal 8. Moving content from previous Drupal versions to Drupal 8 should utilize the Migrate system.

Using the Migrate Drupal module allows you to migrate all content (with supported upgrade paths) to Drupal 8.

If you want to learn more about this change and why it was made you can:

- Read a blog post we wrote about it: https://drupalize.me/blog/201412/drupal-8-upgrade-path.

- Mike Ryan also wrote a blog post (http://mikeryan.name/blog/mikeryan/update-on-migration-in-drupal-8) that covers some of the reasoning, and this really long issue provides a lot of additional background: https://www.drupal.org/node/2313651.

Migration plugins Each individual module is responsible for its own content and configuration, and it needs to describe to the migration system both the data and how to access it. A migration template provides a set of instructions the migration system can use to map the path that a set of data needs to take to get from the source to the destination.

Migration templates are stored in the `module_name/migrations` (for backward compatibility they may also appear in `module_name/migration_templates`) sub-directory of a module. They are plugins, described in YAML files, and are responsible for determining how to source, process, and import data in to Drupal 8.

Look at `Drupal/core/modules/file/migration_templates/d6_file.yml` for an example. This plugin defines which source, process, and destination plugins to use, as well as configuration for each.

Extract, Transform, Load Migrating data to Drupal 8 follows what is known as an ETL process (https://en.wikipedia.org/wiki/Extract,_transform,_load). This stands for Extract, Transform, and Load, and it's a pattern used in many aspects of computing. The basic flow of the process is as follows:

[Extract] --> [Transform] --> [Load]

Source plugins extract data from a source and return it as a set of rows representing an individual item to import andsome additional information about the properties that make up that row.

The *Extract* phase is where data is extracted from a data source: this might be a previous Drupal version or some other form of data store, a database, XML, CSV, or YAML, to list just a few.

The *Transform* phase of a migration is the stage in which source data is manipulated, prior to being imported to its final destination.

The *Load* phase is the stage of the process where data is inserted into Drupal. This phase is handled by *destination plugins*.

In terms of a Drupal migration, the process is controlled by three types of plugins:

[Source] --> [Process] --> [Destination]

Source plugins The extract phase in the Drupal migrate system is handled by *source plugins*. Source plugins extract data from a source and return it as a set of rows representing an individual item to import and some additional information about the properties that make up that row.

Learn more about Source Plugins in this tutorial: https://drupalize.me/tutorial/source-plugins?p=2578.

Process plugins In a Drupal migration, the transform phase is handled by *process plugins*. Each row of data provided by a source plugin will be passed through one or more process plugins that operate on the source data to transform it into the desired format. This result is then passed to a destination plugin during the load phase.

Learn more about Process Plugins in this tutorial: https://drupalize.me/tutorial/process-plugins?p=2578.

Destination plugins In the context of a Drupal migration, the load phase is handled by *destination plugins*, which are responsible for saving new data as Drupal content or configuration.

Learn more about Destination Plugins in this tutorial: https://drupalize.me/tutorials/migrate/understanding_destination_plugins.md.

Note: in Drupal parlance, the term *load* is generally used to represent any action that retrieves data from the database and loads it into memory—for example, loading a node. In this particular case, *load* refers to the process of taking data from memory and loading, or saving, it into the database.

Upgrade path If modules have support for migrating data to Drupal 8, they are said to have an "Available Upgrade Path".

Modules that do not support migrating data from previous Drupal versions are said to have a "Missing Upgrade Path".

Execute and rollback Migrations are performed (executed) via the Drupal UI or using Drush.

Once a migration has run, it can also be "rolled back" to remove anything imported during the migration and provide you with a clean slate.

Highwater marks The Drupal 8 migration system supports highwater marks, which are used to support continuous upgrades. These highwater marks are used as a sanity check for imported content. If a highwater mark exists for the import, the content will not be re-imported unless changed.

With Drupal 7 Nodes, for example, the data used as the highwater mark is the `node_last_changed` field.

Additional resources

- Migration handbook (Drupal.org): https://www.drupal.org/docs/8/upgrade/upgrading-from-drupal-6-or-7-to-drupal-8.
- The API documentation contains more information about plugins and how they relate to the ETL process (api.drupal.org): https://api.drupal.org/api/drupal/core%21modules%21migrate%21migrate.api.php/group/migration/8.2.x.

This Drupal 8 Migration GeekGuide has been put together using the amazing training lessons of Drupalize.Me. They're the leading provider of Drupal training services and empower anyone to build and maintain Drupal-based websites. Their premium online tutorial library, including many more tutorials and topics not included here, is trusted by thousands of individuals and organizations around the world and can be accessed in its entirety at https://drupalize.me. Thank you Drupalize.Me!