

Understanding Drupal

The beginning is the most important part of the work - Plato

This section provides background information that will help you understand the concepts and technologies that underpin Drupal.

Drupal concepts

Before you install Drupal and begin using it, it's a good idea to get an overall sense of what it is, and how it works. This section gives you the big picture, helps you to frame how you think about Drupal, and assists you in determining whether it's a good fit for your project. This small investment of time up front will help you immeasurably in the long run.

The Drupal overview

Although Drupal is often described as a "content management system" (CMS) it is also a "content management framework" (CMF). In other words, unlike a typical CMS, it is geared more towards configurability and customization. Picture a range of measurement where one end of the scale is labeled "specific" and the other end "abstract". On the "specific" end of the spectrum, you would have something whose form is very specialized because it's meant for a specific purpose—like, say, a hammer. On the other end of the spectrum, you would have something much more abstracted, that is available to be configured any way you like, for a variety of purposes—like some wood and a chunk of steel. You could make a hammer, or any number of other things with the wood and steel.

Of course, while chunks of wood and steel are more "configurable" than a hammer, they aren't terribly useful because few people have the specialized knowledge to work with such raw materials. Drupal's purpose is to sit in the sweet spot between the two ends of the scale, and create a sort of "builder's kit" made up of pre-designed components that can be used as-is or can be extensively reconfigured to suit your needs. Its design provides incredible flexibility while still allowing people who aren't programmers to make powerful websites. This principle of manageable abstraction is important to understand, because it is a central concept to all things Drupal. When you understand why a measured amount of abstraction is valuable, you'll begin to understand why this approach is such a strong argument for using Drupal.

The power of abstraction

Imagine you get yourself a shiny new toy truck and a matching toy boat, and as you are enjoying them, you find yourself thinking about how nice it would be if you had a toy that had certain "truck-like" qualities, and certain "boat-like" qualities. But, unfortunately, your truck and boat are firmly rooted at the "specific" end of the toy spectrum. They are what they are, you must enjoy them as-is.

Now, imagine that someone hands you a case full of neatly arranged little toy vehicle parts organized according to type. They explain that this "vehicle construction kit" will allow you to snap together whatever sort of toy vehicle you like, with the ability to choose different body types, varied cockpits, and multiple propulsion mechanisms—lots of jets, propellers, wheels, tank treads,

etc. Now, your truck/boat dream can become a reality. And, as you look at all the shiny new bits, you realize that a truck/boat is just the beginning.

Many popular content management systems are focused on a fixed, particular way of approaching the task of managing a website—they tend toward the “specific” end of our spectrum. While they may use various plug-ins to extend that functionality, the plug-ins are often authored in the same manner—they are very fixed in their task-oriented approach to getting things done. Drupal, on the other hand, with this idea of abstraction embedded in its DNA, is intentionally generalized in its approach to doing things. For instance, instead of creating a fixed “news engine,” Drupal provides systems and tools that allow you to quickly assemble your own custom news engine and tweak it to do exactly what you like. But because these systems and methods are generalized in their approach, they don’t lock you into just news-related things—you can use them to make all sorts of other “engines” and functional widgets. This means that once you learn some of the key tools within the Drupal universe, you will realize that you can endlessly combine them to do all sorts of clever things you (and the Drupal system/plugin creators) may never have imagined before.

A concrete example might go something like this: Say you want to have a news engine that lets you post news articles on your site, and you want to be able to put a little widget on the homepage that shows blurbs for the five most recent news stories. Next, you decide that you want to keep track of recent blog posts, and put a running list of blurbs of the five most recent of those on the homepage as well. With a CMS whose paradigm is rooted more on the “specific” end of the spectrum, you’d likely look for a module that handled news, and had a feature that let you put blurbs on the homepage (not an uncommon thing). Next, you’d locate a module that would track the latest blog posts and put a list of those on the homepage (again, not unlikely to find such a thing). Each module would concentrate on tracking, managing and displaying the information it handles.

But, what happens when you have that brilliant middle-of-the-night idea, and want to blend these two functions by showing a list of blog posts *about* the latest news items, ordered by most-active-contributor first? If you’re using a “toy truck” CMS, you may be out of luck, or need to hire a developer to write you something custom. But with a CMF that’s built around the idea of abstraction, you can whip out your kit full of parts and knock this together pretty quickly. The advantage of an architecture like Drupal’s is that the generalization and standardization in how it does things means that building all sorts of clever, customized site functions is just a matter of snapping parts together. And, to return to our toy metaphor, we aren’t talking about those shabby, knobbed blocks that let you create only rough approximations of the things you want—Drupal’s flexibility is layered, and it gives you highly granular control over nearly every aspect of what you’re building.

Of course, this flexibility comes at a certain cost. While a toy truck is instantly understandable and ready to use without much thought, a modular vehicle construction kit will by nature require you to read the instruction manual first. Keep in mind, this is not about building from scratch, it’s about learning how to connect modules and systems that are already available to create articulate, customized websites.

Drupal has been intentionally kept flexible enough that its power, and the means of accomplishing certain tasks are not immediately evident to those unfamiliar with it. However, after investing a little time, you will find that learning how to do things the Drupal way opens up a whole new world of possibility you never knew existed outside of custom programming. You will likely leave your toy truck and boat in the closet gathering dust.

How Drupal does it

As explained above, Drupal’s power comes from its more abstracted approach to handling web

content and functionality. People often think of a website as a collection of pages, with some functions (like a blog, or a news engine) thrown in to round it out. When they go to manage their site, they are thinking in terms of a tree-like hierarchy of pages that they will go in and edit.

Drupal, on the other hand, treats most content types as variations on the same concept: a *node* (more on these in a moment). Pages, blog posts and news items (some possible node types) are all stored in a common pool, and the sitemap (its information architecture) is an overlay that is designed separately by managing and editing navigation menus. It's a lot like the separation you find in standards-compliant page coding—XHTML provides the meaningful structure of the information, while CSS arranges it for presentation. In Drupal, nodes hold the structured information pertaining to a blog post (such as title, content, author, date) or a news item (title, content, go-live date, take-down date), while the menuing system creates the sitemap as a separate layer. Other elements (node layout themes, and modules like Views and Panels) provide the onscreen display of node contents.

The beautiful thing about keeping these layers separate, is that it's simple to provide completely remixed sitemaps for different user types just by serving them a different navigation menu based on their login information. Pages can be grouped differently, prioritized in a different order based on user needs, and various functions and content can be shown or hidden on a per-user-type basis. It just depends on the experience you want to create.

OK, so what exactly is a node? At its most basic, a node is a cluster of related bits of data. When you create a new blog post, you are actually creating a collection of things such as title, content, author link, creation date, etc. Some of these data points hold information that is displayed when the node is displayed. Others hold meta-data that describes important facts about the node (such as the category).

By keeping all nodes in one big "bucket" instead of segregating them out into separate systems, Drupal ensures that they are all built on the same foundation, and can be handled in the same way. This means that they can all be remixed with down-to-the-pixel control over how they are displayed, and their data points can be used and combined together to organize, search and relate nodes with other nodes. This flexibility makes for some powerful possibilities.

Again, because the idea of abstraction is built into the Drupal paradigm, instead of creating specialized solutions for each and every little thing someone might want to do, systems and methodologies have been built to handle more generalized tasks, and then threaded throughout the entire CMF. These more generalized ways of doing things mean that once you learn how to use a particular methodology, you can use it in all sorts of places, and begin to string things together in inventive new ways.

As an example, let's go back to that idea of remixing a site based on what type of user a visitor is. This concept is threaded throughout Drupal, so not only do you get to customize the sitemap based on user type, but you can show or hide blocks (chunks of data usually set in a sidebar, often the result of some kind of output, like our previous top five news blurbs example), change the site skin, and even shift how node information is displayed and whether a user can comment on a particular node. All of this and more can be switched around on-the-fly based on how a visitor identifies herself.

And a note about comments: this is another illustration of Drupal's generalized approach to things. Comments aren't just part of the blog system, since there really isn't a segregated "blog system." Comments are their own thing, and the ability to leave comments can be "snapped onto" blog posts, news items, book pages (a sort of wiki-like environment) and all sorts of other node types—even ones you custom-create.

Collaborative at the core

Creating an informational website that broadcasts from “one to many” is something that most CMSs do right out of the box. However, where Drupal really shines is when you want to expand into some form of the “many to many” communication model. Where some systems attempt to bolt on this functionality and require you to jump through hoops to get what you want, the idea of a collaborative community is a pervasive theme that has informed the Drupal architecture from the ground up.

With some systems, you can set up a blog, and you can install functionality to handle having a community of users, but what happens when you want to give individual blogs to each of your users, sifting and sorting their contents so that they can be displayed individually with their own skins, while also generating cross-blog topical digests, top five lists, and links out to elaborate, customized user profiles? What if you want to also integrate that in with forums, a wiki-like environment, and give each user their own gallery of taggable photos?

In Drupal, the community aspect is threaded through the system at the most fundamental level, so snapping these functions together and configuring them is not too different whether you want to do it as a mostly one-to-many site, or you decide to open it up and go many-to-many. Connecting the distributed, collaborative possibilities of community with all aspects of your site is something Drupal does very well.

Get started quickly, customize extensively

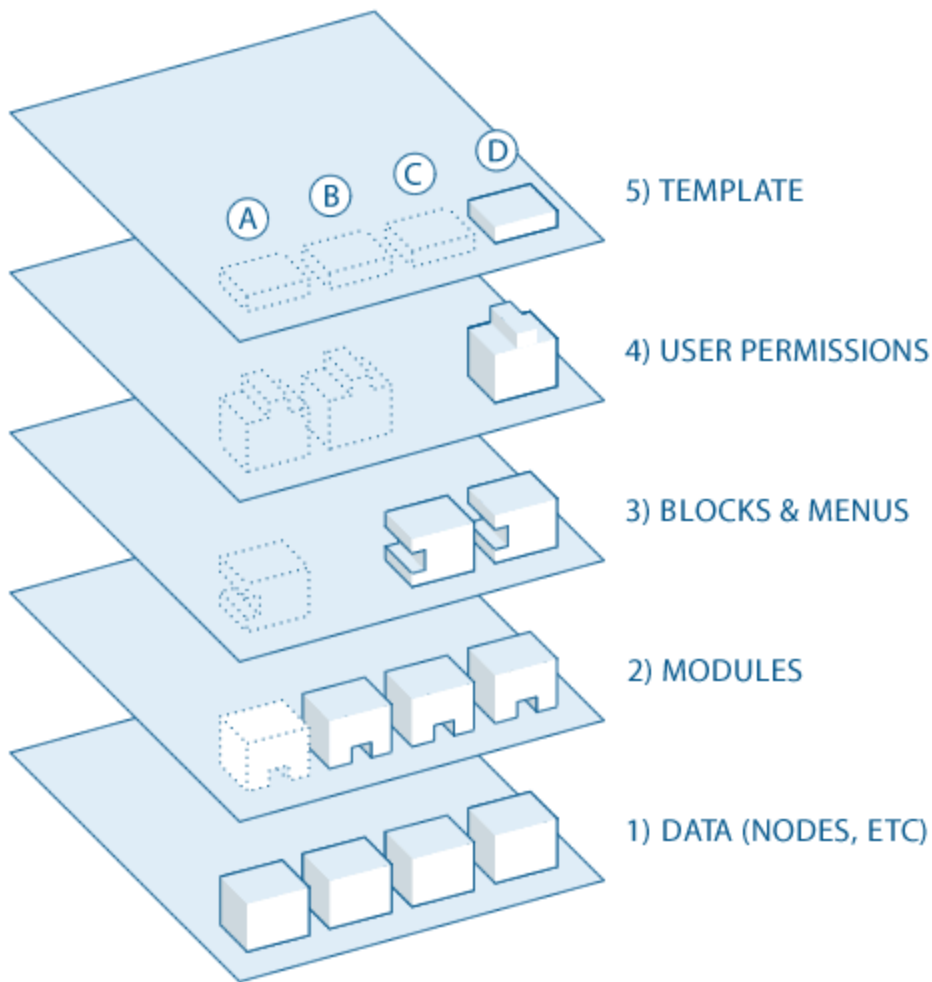
With all of this talk of Drupal’s power and flexibility, you might be thinking it would be overkill for a simple, informational site because of the time required to set it up. Interestingly, nothing could be farther from the truth. With a simple FTP upload and a few short web-based configuration questions, you can connect with your database and have a Drupal site up and running quite quickly.

Pick one of the included themes, and just start adding content. Do you want to have visitors log in? Switch authentication on or off. Want to switch on some of the included tools? Turn on forums; attach the ability to comment on things; set up collaborative wiki-like books; switch on user polls; give site content structured, hierarchical categorization or employ free-form tagging with taxonomy.

Do you want your own skin applied to the site? Drupal uses a system of PHP tokens that you can drop into the appropriate spots in your design to replace your placeholder *Lorem Ipsum* text with system-generated content. Drupal’s generated markup is clean, standards-compliant XHTML. No old school tables. No cruft. No kidding.

The Drupal flow

Integral to understanding Drupal is having the right concept of how things flow within the system. Drupal is cleanly separated into different layers that keep things organized and flexible. There are five main layers in the Drupal system:



1. At the core of the system is the big bucket of nodes—the data pool. Before anything can be displayed on the site, it must be input as data.
2. The next layer out from the center is where modules live. Modules are functional plug-ins that are either part of the Drupal core (they ship with Drupal) or they are contributed items that have been created by members of the Drupal community. Modules provide various functionality to expand your site's capabilities to include things like the creation of custom data points (fields) for your nodes; event calendars; e-commerce; programmatic sorting and display of content (custom output keyed off of any number of configurable parameters that interrelate your content) and more. There are hundreds of different options within the fast growing [repository of contributed Drupal modules](#). They represent the work of everyone from individuals to large corporations who use and rely on Drupal and are working to extend its power and usefulness.
3. At the next layer, we find blocks and menus. Blocks often provide the output from a module or can be created to display whatever you want, and then can be placed in various spots in your template (theme) layout. Blocks can be configured to output in various ways, as well as only showing on certain defined pages, or only for certain defined users.
4. Next are user permissions. This is where settings are configured to determine which things different user types have access to. Permissions are assigned to various roles, and in turn, users are associated with those various roles in order to grant them the associated permissions.
5. On the surface layer is the site template. This is made up predominately of XHTML and CSS,

with some PHP tokens sprinkled throughout to insert content from the system into the correct spots. Also included with each template is a set of functions that can be used to override standard functions in the modules in order to provide complete control over how the modules generate their markup at output time. Templates can also be assigned on-the-fly based on user permissions.

This directional flow from the core to the surface dictates how a lot of the things in Drupal work. Is some new functionality you want not showing up? Perhaps you uploaded the module into the system but have not activated it yet, and this is making everything downstream non-functional (as in "A" in the diagram above).

Maybe the module is installed and activated, but you still don't see what you want on your site. Did you forget to place the block, as in "B"? Or are your user permission settings conflicting with what you want and your users are not set to see the output as in "C"?

Additionally—as mentioned earlier—getting the kind of granular control you want over the details of the XHTML module outputs requires understanding this flow. Are you using a module that does exactly what you want, only you wish the markup was just a little bit different? Maybe you'd like it tagged differently, or you'd like to assign a CSS class to something? You accomplish this by copying the output function from the module and pushing it downstream to the functions document in your template (theme). Modify the code there, and when the system goes to output, it will see your downstream customized function and output through that instead.

Get up close and personal

Now that you've got a map for the territory that is Drupal, take some time to install a copy on a handy server and read the [Installation & Configuration guide](#) for step-by-step instructions on how to get going.

Welcome to the community of Drupal users, and happy site-building!

Is Drupal the right tool for the job?

Drupal is a very powerful and flexible framework to build virtually any kind of website. There are some situations where Drupal is a better choice than other solutions and here are some considerations that may help you decide.

Drupal is an excellent choice for any of the following situations:

- You need a site that is flexible enough to evolve in any direction. For example, you might start with a blog but want the option of adding other features like a wiki, electronic commerce, forums etc.
- You need a site that can easily be configured to interact with other sites or with other technologies.
- You need a site that can easily handle complex forms and workflows.
- You need the ability to create your own content types. For example, you need to [add a custom field to a page](#).
- You need the ability to quickly [organize and display lists of information](#).
- One or more of the many [contributed Drupal modules](#) addresses your needs.
- You need to quickly develop custom functionality.

There are several cases where Drupal may not be the best choice:

- If your only requirement is to write a personal blog, you may also want to evaluate one of the more specialized blogging platforms like [WordPress](#) or [one of the hosted blogging solutions](#). Although Drupal does provide an excellent blogging platform out-of-the-box, you will probably find that blog-specific software typically has a simpler administration interface.
- Similarly, if your only requirement is to create a wiki, you should probably consider using dedicated wiki software like [MediaWiki](#) or a [hosted wiki solution](#). You can certainly configure Drupal so that anyone can edit content (and even enable advanced features of systems like MediaWiki with the help of several contributed modules like [wikitools](#) and [Diff](#)), but it may be simpler for you to use a more specialized solution.
- If your only requirement is to host discussion forums, you will want to consider a system such as [SimpleMachines](#) or [phpBB](#) with a mature set of Forum features, or [Vanilla](#) which has many plug-ins, although Drupal's forum module [with forum enhancement modules](#) may be better suited to extension if you'll need custom features.
- With every release, Drupal is becoming easier to use; but like most powerful tools, it will always have a learning curve. If you or your organization are not prepared to spend some time [learning how Drupal works](#) (or if you are not able to [hire Drupal expertise](#)), it may not be your best option.

General concepts

This page discusses some general concepts that will be useful as you begin to explore Drupal. The remaining topics in this section go into more detail on a few of these concepts.

Module

A module is software (code) that extends Drupal features and/or functionality. Core modules are those included with the main download of Drupal, and you can turn on their functionality without installing additional software. Contributed modules are downloaded from the [Modules download section of drupal.org](#), and installed within your Drupal installation. You can also create your own modules; this requires a thorough understanding of Drupal, PHP programming, and Drupal's module API.

User, Permission, Role

Every visitor to your site, whether they have an account and log in or visit the site anonymously, is considered a *user* to Drupal. Each user has a numeric user ID, and non-anonymous users also have a user name and an email address. Other information can also be associated with users by modules; for instance, if you use the core Profile module, you can define user profile fields to be associated with each user.

Anonymous users have a user ID of zero (0). The user with user ID one (1), which is the user account you create when you install Drupal, is special: that user has permission to do absolutely everything on the site.

Other users on your site can be assigned permissions via *roles*. To do this, you first need to create a role, which you might call "Content editor" or "Member". Next, you will assign *permissions* to that role, to tell Drupal what that role can and can't do on the site. Finally, you will grant certain users on your site your new role, which will mean that when those users are logged in, Drupal will let them do the actions you gave that role permission to do.

You can also assign permissions for the special built-in roles of "anonymous user" (a user who is not logged in) and "authenticated user" (a user who is logged in, with no special role assignments).

Drupal permissions are quite flexible -- you are allowed to assign permission for any task to any role, depending on the needs of your site.

Node

A *node* in Drupal is the generic term for a piece of content on your web site. (Note that the choice of the word "node" is not meant in the mathematical sense as part of a network.) Some examples of nodes:

- Pages in books
- Discussion topics in forums
- Entries in blogs
- News article stories

Each node on your site has a [Content Type](#). It also has a Node ID, a Title, a creation date, an author (a user on the site), a Body (which may be ignored/omitted for some content types), and some other properties. By using modules such as the contributed [Content Construction Kit \(CCK\)](#) module, the core [Taxonomy](#) module, and the contributed [Location](#) module, you can add fields and other properties to your nodes.

Comment

Comments are another type of content you can have on your site (if you have enabled the core Comment module). Each comment is a typically small piece of content that a user submits, attached to a particular node. For example, each piece of discussion attached to a particular forum topic node is a comment.

Taxonomy

Drupal has a system for classifying content, which is known as *taxonomy* and implemented in the core Taxonomy module. You can define your own *vocabularies* (groups of taxonomy terms), and add terms to each vocabulary. Vocabularies can be flat or hierarchical, can allow single or multiple selection, and can also be "free tagging" (meaning that when creating or editing content, you can add new terms on the fly). Each vocabulary can then be attached to one or more content types, and in this way, nodes on your site can be grouped into categories, tagged, or classified in any way you choose.

Database

Drupal stores information in a database; each type of information has its own *database table*. For instance, the basic information about the nodes of your site are stored in the Node table, and if you use the CCK module to add fields to your nodes, the field information is stored in separate tables. Comments and Users also have their own database tables, and roles, permissions, and other settings are also stored in database tables.

Path

When you visit a URL within your Drupal site, the part of the URL after your base site address is known as the *path*. When you visit a path in your Drupal site, Drupal figures out what information should be sent to your browser, via one or more *database queries*. Generally, Drupal allows each module you have enabled on your site to define paths that the module will be responsible for, and

when you choose to visit a particular path, Drupal asks the module what should be displayed on the page.

For instance, this site (drupal.org) is (of course) built with Drupal. The page you are now viewing is `http://drupal.org/node/19828`, whose path is "node/19828". The module that is responsible for this path is the core Node module, so when you visit this page, Drupal lets the Node module determine what to display.

To determine the path to a particular page on your site, for purposes of creating a link, go to the page you want to link to and look at the URL in the address bar. By default the URL, after the base address of your site, will begin with '?q='. When 'Clean URLs' are enabled you will see a directory structure in the URL. The "path" for use in a menu item is the part of the URL after the site's base address and without the "?q=".

Theme

The *theme* controls how your site is displayed, including the graphic look, layout, and colors. A theme consists of one or more PHP files that define the HTML output of your site's pages, along with one or more CSS files that define the layout, fonts, colors, and other styles.

Region, Block, Menu

Pages on your Drupal site are laid out in *regions*, which can include the header, footer, sidebars, and main content section; your theme may define additional regions. *Blocks* are discrete chunks of information that are displayed in the regions of your site's pages. Blocks can take the form of *menus* (which are concerned with site navigation), the output from modules (e.g., hot forum topics), or dynamic and static chunks of information that you've created yourself (e.g., a list of upcoming events).

There are three standard menus in Drupal: Primary Links, Secondary Links, and Navigation. Primary and Secondary links are built by site administrators, and displayed automatically in the page header of many themes (if not, you can enable their blocks to display them). Navigation is the catch-all menu that contains your administration menus, as well as links supplied by modules on your site. You can also create your own custom menus, and display them by enabling their blocks.

You can customise menus in several ways, such as reordering menu items by setting their "weight" or simply dragging into place, renaming menu items, and changing the link title (the tooltip that appears when you mouse over a menu item). You can move a menu item into a different menu by editing the Parent property of the menu item.

You can also add custom menu items to a menu, from the Add menu item tab of the Menu administration screen. To create a menu item, you will need to provide the path to the content (see above).

In all cases a menu item will only be shown to a visitor if they have the rights to view the page it links to; e.g., the admin menu item is not shown to visitors who are not logged in.

See also

- [Drupal Terminology](#)
- The rest of the pages in this section of the Handbook

Content types

A single web site could contain many types of content, such as informational pages, news items, polls, blog posts, real estate listings, etc. In Drupal, each item of content is called a *node*, and each node belongs to a single *content type*, which defines various default settings for nodes of that type, such as whether the node is published automatically and whether comments are permitted. (Note that in previous versions of Drupal, content types were known as *node types*.)

When you first install Drupal with the default installation profile, you will have two *content types* defined: "Page" and "Story". When you enable other core and contributed modules (by visiting Administer >> Site Building >> Modules), you will find that you have other content types available; you can also create your own content types (see below). Here is a list of the content types associated with core Drupal modules:

Blog Entry

A *Blog* (short for weblog) is an online journal or diary, and the core Blog module allows registered users on your site to create their own blogs. Each entry in a user blog has content type *Blog Entry*.

Book Page

Book pages are designed to be part of a collaborative book, enabled by the core Book module. An example of a collaborative book is the [Drupal developer documentation](#). In older versions of Drupal, only nodes of content type Book Page could be added to a book, but now nodes of any content type can be part of a book.

Comment

Comments actually aren't nodes, so Comment is technically not a "content type". Enabling the Comment module allows site visitors to add comments (typically short notes and replies to other comments) to nodes on the site.

Forum

A *Forum* node defines a topic for a forum discussion; people can reply to the topic by using comments. Forum nodes are organized into subject areas via a Taxonomy (list of categories).

Page

The *Page* content type is enabled in Drupal in the default installation profile. Typically Pages are used for static content that can (but are not required to) be linked into the main navigation bar.

Poll

A *poll* is where a multiple choice question is asked, and users can answer and see other people's answers to questions.

Story

The *Story* content type is enabled in Drupal in the default installation profile. Stories are generally used for information whose relevance decreases as time passes (such as news items), so that newer Stories will typically be placed higher on the page than older Stories.

In addition to these basic types, custom content types can also be created by going to *Administer > Content > Content types > Add content type*. You might do this as a way to organize your content -- for instance, you might have "Article" and "News flash" as two simple content types on your site, rather than just using "Story" for both.

If you want to add fields to your custom content types, install the [Content Construction Kit \(CCK\)](#) contributed module. Custom fields are used to store additional information beyond the Drupal defaults (title, body, authoring information, time created/updated, and publishing status); for instance, on a real estate site, a real estate listing content type might have fields for the type of property, land area, etc. Additional information on the CCK module is available from the [Content Construction Kit Handbook](#).

Finally, some contributed modules define their own content types. Check the [Contributed Modules page of the Drupal handbook](#) for more information on locating an appropriate module for your needs.

In Drupal, viewing a page and editing a page are almost the same

Occasionally, people who have previously used other editors (FrontPage, Dreamweaver) or CMSs (Joomla etc) ask how they access the admin area or "back-end" of Drupal. Sometimes they ask how they can preview their changes and see the "front-end".

- In Drupal, there is no such distinction, it just provides a unified interface.
- If you are looking at your website, you are previewing it. Log out and you get the full 'anonymous' experience.
- Drupal usually looks pretty much the same to anonymous browsers as it does to editors, only with different features and menu items available.
- Authenticated users with appropriate permissions will see the 'edit' tabs above their pages. That's often the only difference between editor and user experience.

We find that people who have never used a CMS before are often much less confused about this approach than [people who have previously used systems](#) where the 'input' screens look totally different from the 'output' screens. It's an un-learning thing.

However, if you want, you can still set an 'Administration theme' (in the settings `Administer > Site configuration > Administration theme`) so that your admin pages *look* different from the front end. ... It still actually behaves the same, but it may help if your public presentation theme is highly graphic or stylized.

Understanding Drupal paths

In Drupal terms, a *path* is the unique, last part of the URL for a specific function or piece of content. For instance, for a page whose full URL is `http://example.com/?q=node/7`, the path is `node/7`. If your site is using [clean URLs](#), the full URL in this example would be `http://example.com/node/7`; the path would still be `node/7`. Because [URL aliases](#) can completely replace what visitors see as the URL, the paths discussed here (which are still how Drupal decides what content to show) are sometimes called *internal paths*.

Drupal paths are important because many configuration screens in the Drupal admin area use them. For instance, when you are adding a new item to a menu, you tell Drupal what page the menu item should point to by entering the path to the page.

Here are some examples of paths you might find in a Drupal site:

- `node/7`
- `taxonomy/term/6`
- `admin/content/comment`
- `user/login`
- `user/3`

How to find Drupal paths

There are several ways to find the path to a particular page on your Drupal site. The first step is to find the URL of the page of interest:

- If you know how to navigate to the page, you can go there and find the URL in your browser's URL bar.
- You can also hover your mouse over a link to the page (such as in the Views administration screen or the content management screen at Administer >> Content management >> Content), and most browsers will show you the URL in the status section at the bottom of the browser window.
- For Taxonomy term pages, you can find the URL in the Taxonomy administration page (at Administer >> Categories in Drupal 4.6 and 4.7, Administration >> Content Management >> Categories in Drupal 5, Administer >> Content Management >> Taxonomy in Drupal 6, or Administer >> Structure >> Taxonomy in Drupal 7). If you are viewing a list of terms for a particular vocabulary, each term should be a link to its taxonomy page. Hover over the link or follow the link to find its URL.

The URL you find could have several forms:

- `http://example.com/?q=[something]` -- In this case, the *[something]* after `?q=` is the path. For example, if the URL is `http://example.com/?q=node/7`, the path is *node/7*.
- `http://example.com/[something]` **OR** `http://example.com/[your Drupal subdirectory]/[something]` -- In this case, the *[something]* after the base path of your Drupal site is the path. For example, your URL could be `http://example.com/node/7` or `http://example.com/mysubdir/node/7`; the path in either case is *node/7*.

Differentiating between kinds of "users"

This article is meant to disambiguate the use of the term "user" as it applies to the installation of a Drupal web site.

There are several steps involved in installing Drupal. In several of the steps, "user" information is needed for a particular kind of "account". In order to help you keep the concept of "user" straight, this article outlines the different kinds of "users." A "user" is *not* associated with an individual person in three of the four user types mentioned here. To help sort that out for each kind of user, the question is asked, "Who is..." in order to map the "user" to the actual people.

This is *not* an installation guide. Rather, it is a background piece which provides a bigger picture that might make installation and planning easier.

1. **Computer User.** The person installing Drupal must have access to the computer where the installation lives. If you are installing Drupal locally on your own computer, then the "computer user" is you. If you are installing Drupal at a remote server such as a web hosting company, then the computer user account could be called one of the following (not a complete list):
 - Web hosting account login
 - FTP login
 - Cpanel login
 - Secure shell (SSH) login

Who are the people who are associated with the computer user? In some cases, like for your own computer or for some FTP logins, these "users" are individual people. In other cases, such as a web hosting account login or Cpanel, one user (or "account") name/password is used to log in to control the account and may be shared by several people.

2. **Database User:** A Drupal installation requires a database to run. (MySQL and PostgreSQL are the best supported database programs compatible with Drupal.) Databases typically have access control mechanisms and require "users" with the right permissions to change the database (add/remove/edit data, create tables, etc.). A Drupal installation needs to have full control over the database, so when you set up a new Drupal site, you (or your web hosting company) create a database user with full privileges and then give the user name and password to the Drupal installation so that Drupal will have full control over the database. The database user information is stored in the settings.php file, which is either in sites/default or a different subdirectory of sites in your Drupal installation.

Who is the database user? The database "user" is *not* a person. It is an account created with the database software in order to give Drupal control of the database.

3. **User/1:** "User/1", also known as the "maintenance" account or "super-user account" is the Drupal account you are prompted to create immediately after you have successfully installed a new Drupal site. This account is unique to your site (it doesn't have anything to do with Drupal.org or any other web site). This account is different from all other users in a Drupal installation because it has no permissions limitations ("permissions" were called "access control" in Drupal 5 and before). "User/1", in Drupal 6 and before, is also the *only* account that can launch the update.php script that you need to run after a software upgrade.

Who is user/1? User/1 shouldn't be associated with an individual person, but rather with the person or persons who have the responsibility for keeping software up-to-date on your site.

It is a best practice to *avoid* creating site content with user/1. That is so because it is awkward when responsibility for site maintenance done as User/1 needs to change to a new person if the original User/1 wrote content that still needs to be associated with him/her. The content written by the original author would have to then be assigned to a new user account. It's better to simply create that second account immediately after installing the site.

4. **User/2 and all other users:** User/2 and all other registered users on your Drupal site should each be associated with an individual person. Registered users can be assigned to roles, which are given fine-grained permissions to allow different users different access to administer the site and add content.

Technology stack

Drupal is part of a technology stack that contains a number of important pieces:

Server - A server is a computer which provides information or services to other computers on a network.

Operating system - The software that runs the server. Unix, Linux, BSD, OS X and Windows are some examples.

Database - A structured collection of records. Drupal uses a database to store most content and configuration settings for your site, some content such as media files are generally stored in the server's file system.

Web server - The software component responsible for serving web pages. Examples are Apache and Microsoft IIS.

PHP - PHP is a programming language that allows web developers to create dynamic content

that interacts with databases.

Drupal - A framework for building dynamic web sites offering a broad range of features and services including user administration, publishing workflow, discussion capabilities, news aggregation, metadata functionalities using controlled vocabularies and XML publishing for content sharing purposes. A Drupal installation is generally comprised of a mix of core and contributed modules.

Tip

Drupal 7 will be developed for PHP 5.2. While this does not apply to Drupal 5 or 6 it would reduce later complication to build new sites using PHP 5.2 if possible.

Is Drupal secure?

Drupal has a very good track record in terms of security, and has an organized process for investigating, verifying, and publishing possible security problems.

Drupal's [security team](#) is constantly working with the community to address security issues as they arise. More information about this process can be found in that section of the handbook.

Anyone using Drupal should subscribe to the security mailing list (by editing [your account profile](#)) in order to automatically keep up to date with the latest [security advisories](#) of all types (see below).

Frequently asked questions:

Is open source software secure?

The short answer is that open source software is as secure or more secure (in general) than commercial software. A good summary of the relevant issues can be found in this article from IBM: [The security implications of open source software](#). The increased security of using open source was cited as one reason the [White House switched to Drupal](#).

Why does Drupal have more (or fewer) security advisories than another project?

The absolute number of security advisories (especially when including contributed projects) is a totally meaningless number and should never be used for comparison. Drupal has over 4000 contributed projects which are scrutinized by their users for any *potential* problem, and a security advisory may be issued for a relatively minor issue (such as partial disclosure of private content to anonymous users).

A security advisory also indicates the discovery of a potential problem, and also that the problem is resolved already. It's extremely rare that such holes are exploited in the wild prior to the security fix being announced in the security advisory. Thus, your most important protection is keeping Drupal up to date whenever a security advisory is issued for Drupal core or contributed code you are using.

On live sites, what vulnerabilities have been found or exploited?

Professional security audits of Drupal sites have generally found that the vast majority of security holes (90% or more) are present in the custom theme or modules written by that site's developers. That code did not get the same public scrutiny that all code on drupal.org receives.

In addition, problems at the server level (such as using insecure protocols like FTP) are more likely to be the means of a successful attack than a vulnerability in Drupal - especially Drupal core.

For information on how to manage security in Drupal, see the [Securing Your Site](#) section of the Drupal Administration Guide.

Drupal version numbering

This page explains Drupal's version numbering scheme for both Drupal core and contributed modules. It is designed to help you understand what the various version numbers mean and how you can use this information when building a Drupal site.

Which version of Drupal core should I run?

In brief, you should always run one of the recommended official releases. These can be found at the [Drupal Project](#) page.

At any given time, there are two major *release series* of Drupal which are supported. Currently, these are Drupal 5 and Drupal 6. Updated versions of each of these are issued on a regular basis. For example, within the Drupal 6 series, several versions have been and will continue to be released: 6.0, 6.1, 6.2, etc. These versions are sometimes collectively referred to as "6.x"; however, they are not all equal. The newest version in each series fixes problems that were discovered since the previous version, and these fixes sometimes include critical [security updates](#). Thus, regardless of whether you are using Drupal 5 or 6, you should always run the most current version in the series.

In deciding between Drupal 5 and Drupal 6, several factors should be taken into consideration. Drupal 6 is newer, will be supported longer, and contains more features and enhancements than Drupal 5; therefore, if all else is equal, use Drupal 6. However, the releases of contributed [modules](#), [themes](#), and [translations](#) compatible with each major Drupal release series sometimes lag behind Drupal itself, so if the functionality you need is not yet available for Drupal 6, this may affect your decision. (The [contributed module status list](#) is a useful reference, although you should always verify the information there with the module's official project page.)

Drupal 7 is currently under development and is not yet fit for production use. If previous patterns continue, a stable version (Drupal 7.0) will likely be released sometime in the first half of 2010. Once that happens, Drupal 5 will no longer be supported or receive security updates, and you will need to update your Drupal 5 sites to Drupal 6 at that time.

Which version of contributed modules, themes, and translations should I run?

You usually should run one of the "official releases" from the module, theme, or translation's project page, because these are the versions that are currently supported by the maintainer. Note that when examining a particular release available on these pages, the first part of the version number shows you which major release series of Drupal core a module, theme, or translation is compatible with. For example, a module with a version number of 5.x-1.2 will only work with

Drupal 5, whereas a module with a version number of 6.x-1.2 will only work with Drupal 6.

When is the next release?

Per the Open Source tradition, *when it's ready*. When sufficient testing of fixes of bug reports is complete. More attention is given to a rapid release of security issues. The more people involved in fixing and testing reported issues, the faster a release is likely to come out.

How do I know which version of Drupal I'm running?

If it's available, go to **Administer >> Reports >> Status report**. This will list your version number if you have Drupal 6.0 or later. In Drupal 5.x and earlier, the path to go to is **Administer >> Logs >> Status report**.

Failing that, look for a file called **CHANGELOG.txt** in the root of your Drupal directory and open it up to find the version you are running.

If CHANGELOG.txt is missing, you can also check in **system.module** for a line at the top like:

```
define('VERSION', '5.5');
```

If this is present, it will tell you which version you are running. If not, you have a version earlier than 4.7.2.

How do I know which version of a module or theme I'm running?

You can see the versions of your installed modules and themes on the modules and themes administration screens. Go to **Administer >> Site building >> Modules** or **Administer >> Site building >> Themes**, and you will see a column listing the version number.

What about upgrading and backwards compatibility?

For more details read [this overview of the Drupal's philosophy on backwards compatibility](#).

More details

The pages below have some more details for advanced users.

What do version numbers mean?

Drupal release versions

In Drupal 4.7.x and previous, the first two numbers indicated the Drupal version number, while the last indicated a "point" or bug-fix release with a specific "patch level". For example, "4.7.3" means "the third bug-fix release (patch level 3) of the 4.7 version of Drupal."

This was confusing for people who expected 4.7.x to be a minor update of 4.6.x, when in fact this

was not the case. APIs were incompatible across Drupal versions and require contributed modules to be upgraded.

Starting with Drupal 5.x, the "5" indicates the major version of Drupal, and the .x is the bug-fix release or patch level. That means that 5.0, 5.1, and 5.2 all have the same underlying structure and modules for 5.x are all compatible with each other. However, modules written for Drupal 6.x will not work with 5.x and vice-versa.

Development snapshots

A release with "-dev" at the end of the version indicates a development snapshot from the end of a [CVS branch](#) (as opposed to an official release from a [CVS tag](#)). These snapshots, by their nature, include changing code. It is therefore hard to know exactly what revisions of each file they contain, even if it is a snapshot of a stable branch where only bug fixes and security enhancements are being made (such as the [6.x-dev snapshot release](#) of Drupal core). This makes them more difficult to debug, and they should be avoided for production sites. Development snapshots also use "x" for the patch level to further indicate the variable nature of the code they contain. Both Drupal core and the contributions can have development snapshots.

What do version numbers mean on contributed modules and themes?

As of 2006-11-11 and later

The version number indicates the version of Drupal core the contribution is compatible with, whether it is a stable or development release, and what specific "patch level" of the code it represents. These numbers have the form:

`CoreCompatibility-Major.PatchLevel[-Extra]`

- *CoreCompatibility* is required, and will be something like "4.7.x", or "5.x", to indicate what version of Drupal core this contribution is compatible with. Since the API for Drupal core does not change across stable releases, a module that is compatible with Drupal core 4.7.1 would also work with 4.7.3. This is why the version strings for contributions contain the character 'x' to show that they are compatible with any core release ("4.7.x") not just a specific version of core ("4.7.1").
- *Major* is the major revision number, which indicates what set of functionality a given release has, if it is stable (bug-fixes only) or development (new features), and so on. The default branch for a given release of core (for example, the "DRUPAL-4-7" branch) will be major version 1. By convention, this major revision should always remain stable, and only security patches and bug fixes should be added to it. Optionally, project maintainers could create additional branches, which would have higher major versions (2, 3, etc.). These additional major revisions will be for whatever the maintainer of the project sees fit. It is up to the project maintainer to document on their project node and in their releases how they're using their own branches and what users of their modules should expect from any available major revision numbers. By convention, if the maintainer is adding new features to a module, they would use major revision 2 or higher.
- *PatchLevel* is for a specific release of a given major revision. The first release would be patch level 0. As bugs are fixed, each new releases would have a higher patch level number. If the patch level is the letter "x", it indicates a development snapshot (see *[-Extra]* immediately below for more information).
- *[-Extra]* is optional, and is for specifying things like "-dev", to indicate a development

snapshot from the end of a CVS branch (as opposed to an official release from a CVS tag). These snapshots, by their nature, include changing code. It is therefore hard to know exactly what revisions of each file they contain, and this makes them more difficult to debug. Development snapshots also use "x" for the patch level to further indicate the variable nature of the code they contain.

Some example version strings and what they mean should help clarify:

- 4.7.x-1.0 The initial (patch-level 0) stable release (major revision 1) of a module compatible with any version of Drupal core 4.7.*.
- 4.7.x-2.1 An updated (patch-level 1) new-feature release (major revision 2) of a module compatible with any version of Drupal core 4.7.*.
- 5.x-1.0-dev A nightly development snapshot ("-dev") of a stable release (major revision 1) of a module compatible with any version of Drupal core 5.*.

Contributed module versions prior to 2006-11-11

If you happen to have an older "release" of a contributed module, the version numbers are different. Prior to 2006-11-11, the version numbers for contributions were of the form "X.Y.0".

- X.Y indicates the version of Drupal that a given module is compatible with. A module with a version of 4.6 is compatible with Drupal core 4.6.1 and 4.6.2.
- There was no patch level indicator for modules, so the version was always "X.Y.0". Modules were packaged automatically (readied for download in a tar.gz format) from the CVS repository whenever their maintainer updated them. The only way to know what specific code you had and if a newer "release" was available was to check the release date on the projects page to see if it had been updated.

What are betas, release candidates, and HEAD?

What are betas and release candidates?

Before every official "x.0" stable release of a new major version of Drupal core, there are usually a handful of *beta releases* and *release candidates* that are made available. These releases are **not yet stable enough for production use**, but are essential milestones on the way towards the official release. They allow a much wider pool of users to test the latest code and provide feedback before the official stable release. **These releases should only be downloaded and used by developers** very familiar with Drupal **or those wishing to help find bugs** in the software.

Beta releases

These are the first to come out, and are therefore the least stable. During the period of beta releases, usability features are still considered, the translatable strings (help texts, words in the interface, etc) might be altered, and if absolutely necessary, the API or database schema could change (to fix a critical bug). Of course, other kinds of bug fixes are always applied.

Release candidates

Release candidates are usually only created once no more critical bugs have been reported in a given beta release. These are considered nearly stable code, something the Drupal development community is considering as a *candidate* to be released as the official .0 version. No more usability changes are made, and the translatable strings are usually unchanged at this point.

Some maintainers might choose to provide beta releases or release candidates of their contributed modules and themes (though this is not required). You should read the release notes carefully in these cases, since the details might vary across projects, though the basics explained here should hold (a "beta" is less stable than an "RC", etc).

Once a *feature freeze* is announced, no new features will be added to that version of Drupal. That version of Drupal's feature set is locked and any new features or change of behavior will need to go into the next release version.

What is "HEAD"?

In the forums you will see references to Drupal HEAD or just HEAD. You will also see references to Drupal CVS which is often the same thing. The name comes from the fact that Drupal uses the [CVS](#) version control system to track changes to the code, and in CVS, "HEAD" refers to the main development area (also known as "the trunk").

In *Drupal core*, HEAD is the name given to the version of Drupal core being worked on by developers right now. Of course, now that core is only using two digits for the version number (starting with the 5.0 release), there's no longer any ambiguity about what the next version of core will be called, so the use of "HEAD" to identify a release is no longer necessary. For example, now that the official 6.0 release of Drupal core is out, everyone knows that the next version of core under development will eventually become the 7.x release series, so the nightly snapshot releases are more properly called "7.x-dev", not "HEAD".

HEAD versions of Drupal core are not meant for use on production sites and should only be downloaded and used by developers who are very familiar with Drupal, or by testers who are helping to find bugs. HEAD is a moving target, and is prone to serious bugs and security holes.

When a *contributed module or theme* is tagged with HEAD, that means that it is the latest version on the main development branch. But "latest" can be any age -- perhaps over a year old! When a file was made for Drupal 4.3.x, it is HEAD of its own branch, but that does not mean you can run it with Drupal 4.7.x, or 5.5, or that it is necessarily compatible with any other module or core code that's tagged with HEAD.

In general, **production sites should never use HEAD versions of any module or theme; they should always use official releases with real version numbers.** Under the new release system, even the HEAD releases of contributed projects are required to include version information and to indicate which versions of Drupal core they are compatible with. A contribution without any official releases indicates that the contribution maintainer is not being very responsible, and you should use these contributions at your own risk.

Terminology

This glossary describes the vocabulary used by Drupal and the Drupal community.

Click one of the letters below to be taken to the list of all terms beginning with that letter. If you have some "english expression" and or [acronyms](#), you are not clear with, you can also see [here](#)

Alphabet

[A](#)|[B](#)|[C](#)|[D](#)|[E](#)|[F](#)|[G](#)|[H](#)|[I](#)|[J](#)|[K](#)|[L](#)|[M](#)|[N](#)|[O](#)|[P](#)|[Q](#)|[R](#)|[S](#)|[T](#)|[U](#)|[V](#)|[W](#)|[X](#)|[Y](#)|[Z](#)

[Back to Top](#)

A

anonymous <#>

An anonymous user is a visitor to a Drupal website who is not currently logged in. Drupal considers any such visitor as being the anonymous user, with the user ID 0, and belonging to the anonymous user role.

argument <#>

A section of the URL for a page on a Drupal website. For example, in the URL for this page (<http://drupal.org/node/937>), the first argument is "node", and the second is "937". Some modules, most notably Views, allow the use of "wildcard" arguments that allow a particular page to vary depending on context.

[Back to Top](#)

B

block <#>

Blocks are a method for positioning data within a page. They often contain lists of nodes or other navigational content and are frequently placed in the left or right regions of a page. Assignment to a region is specified through the admin settings. Blocks themselves are not nodes. You can specify that a block only appears on certain pages or in certain contexts. The appearance of a block is controlled in a theme by the (code: *block(\$subject, \$content, \$region = "main")*) method.

[Configuring and Managing Blocks](#)

book <#>

A book is a set of *book pages* tied together in a hierarchical sequence, perhaps with chapters, sections, subsections, and so on. You can use books for manuals, site resource guides, Frequently Asked Questions (FAQs), or whatever you'd like. To use books, enable the core [Book module](#).

[Back to Top](#)

C

CCK (Content Construction Kit) <#>

A contributed module which permits site developers to define custom fields and content types. A variety of extension modules to CCK exist permitting specialized field definitions such as images, dates, and computed values.

child <#>

Used to describe objects that can have hierarchical relationships, such as menu items, book pages, taxonomy terms and so on. A "child" menu item, for example, is nested under another menu item, which is referred to as the "parent" menu item. See also: [parent](#)

code freeze <#>

Refers to a date at which no new features can go in the next version of Drupal, unless specific dispensations have been made by the core committers (even then, only when the impact on other systems is minimal). At code freeze, the focus in Drupal core shifts to bug fixing and usability improvements. It is the time when contributed module developers can begin working on updating their code to work with the next version of Drupal.

content <#>

Often erroneously used in Drupal documentation as a synonym for [node](#), content refers generically to the text, images, and other information on a web site. Some content on a typical Drupal site is not actually nodes, such as comments that are attached to nodes, and file attachments.

content type <#>

Every [node](#) belongs to a single 'node type' or 'content type', which defines various default settings for nodes of that type, such as whether the node is published automatically and whether comments are permitted. Modules can define their own content types; the core Drupal Book and Poll modules are two examples of modules that define content types. You can find out more about content types on the [content types handbook page](#).

core or Drupal core <#>

Refers to the Drupal files and modules included with the Drupal project download.

core committers <#>

A team of Drupal developers that review proposed changes to the *Drupal core* and maintain code. They are the only ones who have write access to the core CVS repository.

core contributor <#>

Core contributors are software developers who contribute code patches or documentation for the *Drupal core*. Contributions are peer reviewed and then evaluated by the core committers.

cron <#>

Cron (short for chronograph) is a command scheduler that executes commands or scripts (groups of commands) automatically at specified time/date intervals. Drupal uses a 'cron job' to perform periodic tasks that help Drupal to run smoothly and efficiently.

CVS (Concurrent Versions System) <#>

CVS is a version control system used by Drupal code contributors to coordinate their individual code changes. CVS records everyone's changes to a given project in a directory tree called a CVS repository .

[Back to Top](#)

F

filter <#>

Filters are used to strip out HTML, PHP, JavaScript, and other undesirable elements from content before pages are displayed. Other filters add formatting and features such as smilies. It is possible to create custom filters that allow or forbid only those tags you wish. [Text filters and input formats](#)

[Back to Top](#)

I

input format <#>

An input format is a collection of filters, arranged in a certain order, that defines the processing that happens to user-entered text before it is shown in the browser. Usually different user roles are given permission to use different input formats depending on how much they are trusted. For those roles, the input format may often be available as an option that shows up underneath the body of a node edit form. [Text filters and input formats](#)

[Back to Top](#)

L

log <#>

A log is a list of recorded events. A log may contain usage data, performance data, errors, warnings and operational information. Watchdog is Drupal's primary event log.

[Back to Top](#)

M

In Drupal, the term menu refers both to the clickable navigational elements on a page, and to Drupal's internal system for handling requests. When a request is sent to Drupal, the menu system uses the provided URL to determine what functions to call.

module <#>

A module is software (code) that extends Drupal features and/or functionality. *Core modules* are those included with the [main download of Drupal](#). *Contributed (or "contrib") modules* are available for separate download from the [modules section of downloads](#). Note: Be sure that the [version](#) of the contributed module you wish to use matches your version of Drupal -- when you are viewing the list of modules, you can filter for compatibility to a particular version of Drupal by clicking on links in the sidebar.

[Back to Top](#)

N

node <#>

A node is a piece of content in Drupal, typically corresponding to a single page on the site, that has a Title an optional Body, and perhaps additional fields. Every node also belongs to a particular [content type](#), and can additionally be classified using the [taxonomy](#) system.

Examples of nodes are polls, stories, book pages, images, etc.

node type <#>

See [content type](#).

[Back to Top](#)

P

parent <#>

Used to describe objects that can have hierarchical relationships, such as menu items, book pages, taxonomy terms and so on. A "parent" menu item, for example, contains other menu items, which are referred to as "children" menu items. See also: [child](#)

path <#>

In Drupal terms, *path* is the unique, last part of the URL for a specific function or piece of content. For instance, for a page whose full URL is `http://example.com/?q=node/7`, the path is `node/7`. Drupal can use "clean URLs" if the [Path module](#) is enabled, which would change the full URL in this example to `http://example.com/node/7`; the path would still be `node/7`.

permission <#>

1) Drupal - Permissions control access to content creation, modification and site administration at the application level. Administrators assign permissions to roles, then assign roles to users. The first user ID (uid) of a Drupal site (uid=1) automatically receives all permissions, no matter what role that user belongs to. Any anonymous user has uid=0 (see also [anonymous](#)).

2) UNIX/Linux/Windows - Permissions are security settings restricting or allowing users to access information or perform certain functions at the operating system level. In the case of

files on UNIX or Linux systems, there are three types of permissions: read, write, and execute.

project <#>

An open-source collaborative effort; there are several [types of projects](#) in the Drupal community, including contributed modules, contributed themes, handbook documentation on the drupal.org web site, and the core Drupal software itself.

[Back to Top](#)

R

role <#>

Roles are sets of permissions that can be applied to individual users. Users can belong to more than one role. Two roles, authenticated users (those users that sign up for an account) and anonymous users (those either without an account or not logged in), are supplied by default with Drupal installations. Users with sufficient permission can create additional roles, and the permissions granted to the two default roles can also be configured.

RSS <#>

RSS (Really Simple Syndication) is a family of Web feed formats used to publish frequently updated content such as blog entries, news headlines or podcasts. An RSS document (which is called a *feed* or *web feed* or *channel*) contains either a summary of content (*teaser*) from an associated web site or the full text.

[Back to Top](#)

S

story <#>

A type of [node](#) that is defined in a default Drupal installation, typically used for content that has a date associated with it, such as a news item.

style <#>

A CSS file (or files) replacing the default CSS of a theme or engine. Appears in the theme selection list with the same precedence as themes and templates.

[Back to Top](#)

T

tarball <#>

In computing, *tar* ("short for tape archive") is both a file format and the name of the program used to handle such files. The jargon term *tarball* is used to describe an archive that has been created with the *tar* command. A *.tar* file is commonly referred to as a tarball. Tarballs are used within the developer community to collate collections of files into one larger file, for distribution or archiving, while preserving file system information such as user and group permissions, dates, and directory structures.

taxonomy <#>

Taxonomy is literally "the science of classification". The Drupal taxonomy system enables authorized users to categorize content using both tags and administrator-defined terms. It is a flexible tool for classifying content with many advanced features. Further information can be found within the [taxonomy system](#) documentation.

- term <#> - a category or tag or keyword ie what gets assigned to nodes. Terms can be

structured as [children](#) and [parents](#) to create hierarchies.

- vocabulary <#> - a collection of terms that share some sort of relationship with each other. The vocabulary isn't part of any hierarchy as such.
- Taxonomy - the name of the whole system and the name of the module that implements it. In some versions of Drupal, that is also the name of the menu item in the admin menu.
- category <#> - A synonym for taxonomy, and the name of the menu item in some versions of Drupal.

teaser <#>

The first few words or sentences of a piece of content, usually with a link to the complete node.

template <#>

A file that is mostly HTML with some special PHP code to substitute in values provided by an engine.

theme <#>

A theme is a file or collection of files (PHP, INFO, CSS, JPG, GIF, PNG), which together determine the look and feel of a site. Drupal modules define [themeable functions](#) which can be overridden by the theme file. There are additional themes available in the [themes section of downloads](#).

theme engine <#>

A theme engine is a set of scripts that interprets code and makes theming a site easier. This takes the dynamically generated content and outputs it to HTML. Drupal has three theme engines in addition to being able to write a theme that bypasses the theme engine. The default theme engine is phpTemplate. See the [theme engines section of downloads](#) for contributed engines.

[Back to Top](#)

U

URL (uniform resource locator) <#>

A URL is the address that defines the route to locate an object on an Internet server.

Generally, the syntax for a URL contains the scheme, host-name, port, path and filename, for example; `http://www.drupal.org/node/937`

[Back to Top](#)

V

Views <#>

A contributed module which allows site developers a simple graphical interface for modifying the presentation of content. Views permits selection of specific fields to display, filtration against various node attributes, choice of basic layout options (ie. list, full nodes, teasers, etc.), and other more advanced features. Many Drupal sites use Views extensively.

[Back to Top](#)

W

weight <#>

Weight is a term used by Drupal to define the priority or order in which an function is processed or a block / node is displayed. From Drupal 6, the weight field is adjusted

dynamically using a drag-and-drop interface. Note: A lower weight value (-10) will float to the top of lists, while heavier (+10) weights will appear lower in lists.

WYSIWYG

WYSIWYG is an acronym for What You See Is What You Get, used in computing to describe a method in which content is edited and formatted by interacting with an interface that closely resembles the final product.

Other resources

For alternative explanations of Drupal terms, these external sites may be useful:

- ["Getting started with Drupal" page by the IBM Internet Technology Group](#)

Acronyms and expressions used on Drupal.org

Common English expressions and [acronyms](#) used by members of Drupal.org in posts, issues and comments. This page gives basic help and reference to Drupal.org visitors, newbies, beginners and [NNEs](#). It is a list of [terms](#) and their [definitions](#) that can help you understand contents on Drupal.org better. We aim for your contribution. [Feel free to propose missing terms](#) in this list, that you may find while reading contents on Drupal.org. If you do not find them here you can take a look at:

- [Terminology: vocabulary](#) used by Drupal and the Drupal community.
- [Drupal Jargon](#): Glossary for beginners.
- [Using English](#) a collection of ([ESL](#)) tools & resources for study English language.
- [Acronym Finder](#) : dictionary of acronyms, abbreviations, and initial isms.
- [Jargon file](#) from [www.catb.org](#) "Hackers slang"
- [Slang Site](#): dictionary of slang, web-speak, made up words, and colloquialisms.

Alphabet:

[| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z](#)

[Back to Top](#)

A

accessibility

An approach to design that aims to ensure the widest access to a website. While this often refers to accessibility for people with blindness, it includes people with limited bandwidth, older computers or browsers, and people with a range of disabilities including poor eyesight, blindness, deafness, motor impairments, seizure disorders and cognitive impairments. It also includes people using a wide range of user agents such as graphical browser, text-only browsers, screen readers and hand held devices.

acronym(s)

An abbreviation composed of the initial letters or syllables of a compound term and which is pronounced as a single word.(i.e. [D.O.](#))

AFAICT

reads: as far as I can tell

AFAIK

stands for "*as far as I know*"

AKA

Acronym for "**a**lso **k**nown **a**s".

amended

Referring to a version that has been modified from a previous form

API

application programming interface. A set of routines that an application program uses to request and carry out lower-level services performed by a computer's operating system.

ASAP

stands for "*as soon as possible*". Also "ASAP".

ATM

read at the moment. (i.e. "I'm a little distracted atm")

authoring

A term for the process of writing a document. "Authoring" seems to have come into use in order to emphasize that document production ...

[Back to Top](#)

B**bare bones**

the essential elements of something, described without going into detail; A basic thing, devoid of even rudimentary ornamentation.

BLOB

a blob is a collection of binary data stored as a single entity in a database management system.

blog

Shorthand for web log, a regularly updated, journal-style web page that is generally presented in reverse chronological order and allows readers to post comments and discuss topics or posts. Also used as a verb; to weblog, meaning to write in a weblog. See also vblog, splog.

BoD

board of directors. Those individuals selected to sit on an authoritative standing committee or governing body, taking responsibility for the management of an organization. ([Drupal Association - Board of Directors](#)).

BoF

Birds of a feather. In the world of computing, BoF (Birds Of a Feather) can refer to: An informal discussion group. Unlike Special Interest Groups or Working groups, BoFs are informal and often formed in an ad-hoc manner. (i.e. [Groups.Drupal.org](#))

bogus

fraudulent; having a misleading appearance, the term is used in relation to the content of a comment or page feedback etc.

brainstorm

insight: the clear (and often sudden) understanding of a complex situation, try to solve a problem by thinking intensely about it.

breadcrumb(s)

is a term borrowed from "Hansel and Gretel", who left crumbs of bread along their path so they could find their way back out of the forest. In current computer parlance, it refers to the section, usually near the top of the page, that shows the path you followed to locate the current page. For example, it might show `Home > Macadamia Nuts > Current Events > News Articles`, meaning that you started at the home page, clicked on "Macadamia Nuts" in the menu, then selected "Current Events" in the sub-menu, and finally selected, "News Articles."

browser(s)

is the "program" that you use to display content from the Internet. In reality, it is usually a set of programs, not a single one; it is also a set of tables (e.g. settings) that are used to control its display. Examples are **Internet Explorer** (i.e. IE), **Netscape**, **Firefox** (i.e. FF) and **Safari**. This operates on the **client**, or user, side of the presentation.

BTW

1. abbreviation for: **by the way**.
2. Incidentally; not central or crucial to the topic at hand.

bubble up

move upwards in bubbles, as from the effect of heating; also used metaphorically; "Gases bubbled up from the earth"; "Marx's ideas have bubbled up ..."

[Back to Top](#)

C

callback

Callbacks simply register a path so that the correct function is fired when the URL is accessed. They are not shown in the menu. **Note 1**: Description taken from API.Drupal.org
Note 2: for better and deeper understanding "callback" & "hook" see : [Drupal's page serving mechanism](#)

CGI

Common Gateway Interface. A specification for communication between an HTTP server and gateway programs on the server.

client

An application or system that accesses a (remote) service on another computer system known as a server by way of a network. The term was first applied to devices that were not capable of running their own stand-alone programs, but could interact with remote computers via a network.

CMS

Content management system. In the context of a Web site a CMS is a collection of tools designed to allow the creation, modification organization and removal of information.

cookbook

The term cookbook is sometimes used metaphorically to refer to any book containing a straightforward set of already tried and tested "recipes" or instructions for a specific field or activity other than cooking, that others can use unchanged. e.g. [The Drupal Cookbook \(for beginners\)](#)

CVS

Concurrent versioning system. A system that tracks all changes made to Drupal and its modules, allowing multiple developers to work on a single project, regardless of the geographic distribution of those developers. Projects in the CVS repository are not considered stable, and caution should be used when installing them on a live site.

[Back to Top](#)

D

DA

abbreviation/acronym for [Drupal Association](#), also (**D.A.**).

database

A computer database is a structured collection of records or data that is stored in a computer system. A database relies upon software to organize the storage of data. In other words, the software models the database structure in what are known as database models (or data models).

definition

A concise explanation of the meaning of a word or phrase or symbol.

deprecated

In computer software standards and documentation, the term *deprecated* is applied to software features that are superseded and should be avoided. That means also: this standard (*Documentation or Code*) has been replaced by one or more other standards.

Dev Team, Development Team

[Code Maintainers](#)

dismal,(is)

Disappointingly inadequate; gloomy and bleak; depressing; blue. (*i.e. "Currently, the status of drupal_add_js is dismal."*)

DIY

DoIt Yourself

documentation sprint

During documentation sprints people meet in person at a specific time to communicate and work on documentation tasks. Read more [about software development Sprints on wikipedia](#).

D.O., DO, d.org

[Drupal.org](#). The website of Drupal (*where you are right now*). You may find this abbreviation also in small letters. i.e. (d.o.)

Doc Team, Documentation Team

[Handbook maintainers](#) and [book contributors](#) are one of Drupal's most valuable assets, and are one of the sole forces behind improvements to the platform and the community itself.

Without this kind people, we might don't have this documentation. **Note:** [Join the Doc Team](#)

ditto

Same as above or before. Synonymous to ditto mark, a mark used to indicate the word above it should be repeated. Occasionally used in discussions (including forums and IRC) to indicate "me too".

Us3er37: I'm getting errors when I log in and immediately visit 'Home'
C0pyC@t: ditto. Any idea why?

Drupal

From the Dutch word "Dorpje", which in English means "*little village*". For more info about how Drupal did start see [Drupal History](#). Drupal.org was first registered on: 26-Apr-2001

Drupal Object Reference

see [Drupal Object Reference Page](#). **Note:** page is tagged as related to *Drupal 4.6.x*, will be updated sooner or later.

Drupal Project

1. A content management system: The forms you fill out, the buttons you click, and the content you work with. The stuff you interact with every day while you manage your site.
2. A content management framework: The low-level APIs that let you extend and modify Drupal to make it do everything from ratings to image galleries to your dishes.
3. A community: The thousands of documentation writers, developers, testers, support providers, designers, and evangelists from all over the world, working together to make Drupal a better platform every single day.

More info about ["Drupal Project usage overview"](#): A summary page of the usage information for the projects on this site.

Note: this definition is being extracted from an Article on [Lullabot.com](#) authoring by [Angie Byron](#)

Druplicon

Word formed from *Drupal* and *Icon*, the Drupal logo. read the [History of the Druplicon](#).

dunno

Stands for *I don't know*

[Back to Top](#)

E

e.g.

exempli gratia; for example. A good rule of thumb when choosing e.g. versus i.e. is to think of e.g. as "example given" and i.e. as "in essence"

embed

is a command or process by which an item from one program is placed into an item in another program. It sometimes refers to a font being embedded into an art file. Most often used to describe Flash or Video 'objects' embedded in a web page.

etc.

and so forth: continuing in the same way. **Note:** from Latin ["Et cetera"](#)

ESL

English as a Second Language. ESL is a program model that delivers specialized instruction to students who are learning English as a new language.

[Back to Top](#)

F

FAPI

An application programming interface that enables programmers to make programs that run on both OS/2 and DOS operating systems.

FAQ

stand for "Frequently Asked Questions". **Note:** the most relevant FAQs about the Drupal Project can be found at [Drupal Project FAQs](#)

flag

1. In computer programming, flag refers to one or more bits that are used to store a binary value or code that has an assigned meaning. 2. To mark an information item for selection for further processing. 3. A character that signals the occurrence of some condition, such as the end of a word. **Note:** on Drupal.org we use mostly (def. 2) for flagging Issues.

flaming

In chatting [jargon](#) *flaming* is the sending of messages that include bad language or repeat messaging especially of undesirable or obscene text. Flaming (also known as 'flame wars') occurs in unmoderated chat rooms. **Note:** this is also applied in posts in Forums [Topics](#).

foobar

The term *foobar* is a common placeholder name, also referred to as metasyntactic variable, used in computer programming or computer-related.

fork

1. In software engineering, a project *fork* happens when developers take a copy of source code from one software package and start independent development on it, creating a distinct piece of software. ((fork-like) i.e. bifurcate: resembling a fork; divided or separated into two branches; "the biramous appendages of an arthropod"; "long branched hairs on its ...)

FWIW

it reads: for what it's worth

FYI

(pronounced EFF-WAI-AI) is an abbreviation for "For your information," and is often used in forwarding e-mail or printed material to. In practice it indicates additional details that the recipient may not have requested or find relevant, but could be interesting or archived for reference. This prefix often implies that a response is not expected.

[Back to Top](#)

G

GA

General assembly. means the assembly that is the governing body of an organization and the purposes, functions and composition of which are provided for. Also (G.A.). Here is the link to the [Drupal General Assembly](#) body.

G.D.O., GDO

Groups.Drupal.org serves the Drupal community by providing a place for groups to organize, plan and work on projects. Real world local user groups in particular are encouraged to setup their online presence at GDO. You may find this abbreviation also in small letter. i.e. (gdo)

geek

A "geek" is a slang term, noting individuals as "a peculiar or otherwise odd person, especially one who is perceived to be overly intellectual."

GHOP(task)

stands for [Google Highly Open Participation](#);
we should use DHOP on Drupal.org ;-)

glossary

An alphabetical list of abstruse, obsolete, unusual, technical, or other terms concerned with a subject field, together with definitions. **Note:** more info see [What is Terminology](#)

Google's "define:"

Search tag to find definitions of things from various different sources. To use it, all you need to do is highlight a word in your browser, copy, paste it in the search form of your browser an add before the word: *define:* then push the "enter" button on your keyboard. You will wonder how many definitions will appear for helping you to define an english word! Enjoy!. Here is the link to the [Google Help Features](#)

GUI

Graphical user interface. A program interface that takes advantage of the computer's graphics capabilities to make the program easier to use.

[Back to Top](#)

H

HCI (Human Computer Interaction)

a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.

HEAD

The current development version of Drupal. It is not meant for use by regular users and should be downloaded and used only by developers or those wishing to help find bugs in the software. (Thanks to [Steve Dondley](#) for this [newbie-centered explanation](#).)

hook

A hook is a PHP function that is named `foo_bar()`, where "foo" is the name of the module (whose filename is thus `foo.module`) and "bar" is the name of the hook. Each hook has a defined set of parameters and a specified result type. **Note:** Definition token from [API.Drupal.org](#)

HTML

(Hyper Text Markup Language) is the set of symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup symbol is referred to as an element or tag.

htmlawed

1. make HTML markup in text more secure and standard-compliant.
2. processed text can be used in HTML, XHTML or XML documents.
3. restrict HTML elements, attributes, protocols, etc.
4. balance tags, check element nesting, transform deprecated attributes and tags, convert relative to absolute URLs, etc.

Note: For more information, see [PHP code to purify and filter HTML](#)

[Back to Top](#)

I

I18n (Internationalization)

[A numerical abbreviation](#) because the full word is long and tedious to write. 18 refers to the number of letters skipped. Internationalization refers to enabling translations and other-language support (including alternate character sets and right-to-left rendering) in computer systems.

IA (Information Architecture)

The way in which an information space (such as a website) is organised. It refers to the structure, content, labeling and categorization of information, including the design of navigation and search systems. The aim of information architecture is to help users find information and complete their tasks.

i.e.

id est; that is to say; in other words. A good rule of thumb when choosing i.e. versus e.g. is to think of e.g. as "example given" and i.e. as "in essence"

IDE

Integrated development environment. An advanced code editor, often a large GUI that presents source files within a project, Build and test tools, syntax highlighters and checkers, code formatters, context-sensitive code assist. Many more plugins are often available, such as CVS integration, debug trace tools, and inline documentation. A popular IDE used by Drupal developers is the open-source (Java) [Eclipse](#) (all platforms).

IIRC

(If I Remember/Recall Correctly) indicates that the writer knows they may not be 100% correct, so you should probably double-check, and don't get confused if the information given was wrong. See [IRC slang](#).

IMO

abbrev. for **In my Opinion**. also **IMHO In My Humble/Honest Opinion**.

index

1. [lit] An alphabetical directory at the end of a book that references names and subjects discussed in the book and the pages where such mentions can be found. Importantly, an index doesn't contain any information, definition, or explanations at all - only pointers to multiple places where the word, term, acronym is used.
2. [database] A database table index is an internal list of pointers to records kept to speed up searches and lookups. In general, more indexes in a database take up more space but can speed up performance. In a few cases it can actually slow it down. A common task in database optimization is to inspect the table indexes to ensure they are

appropriate.

3. [programming] Generally a list or array of pointers or labels used for lookups.
4. [programming] The *counter* variable used when looping through an array.

```
for($i=0; $i<10; $i++) { print "The index is $i"; }
```

IOW

abbrev. for: In Other Words

IRC

Internet Relay Chat. A network protocol that allows people like our developers to chat in real time over the internet. Drupal discussions are often going on in [The Drupal IRC Channels](#)

ISAPI

Short for Internet Server API, ISAPI is an API for Microsoft's IIS (Internet Information Server) Web server. Enables programmers to develop Web-based applications that run much faster than conventional CGI programs because they're more tightly integrated with the Web server

ISP

(Internet Service Provider) An organization offering and providing Internet access to the public using computer servers connected directly to the Internet. Generally this means the company you pay bills to for your home/modem/router connection. They are not often the same thing as your **Hosting Provider** who provides the hardware and network connection for your site.

issue

In computing, the term issue is a unit of work to accomplish an improvement in a data system. An issue could be a bug, a requested feature, task, missing documentation, and so forth. The word "issue" is popularly misused in lieu of "problem." This usage is probably related.

[Back to Top](#)

J

jargon

slang: a characteristic language of a particular group (as among thieves); "they don't speak our lingo"

[Back to Top](#)

K

Kudos

An expression of congratulations or approval, a bit like "bravo", but treated like a noun in English. "Kudos to the team for a job well done!"

[Back to Top](#)

L

LAMPP

Linux **A**pache **M**ySQL **P**HP **P**erl, an interoperable group of open-source computer programs. see also: [WAMP](#), [XAMPP](#).

layman or layperson

1. Someone who is not a clergyman or a professional person. (hist.)The term layman originated from the use of the term laity, but over the centuries, changed definition to mean a person who is a non-expert in a given field of knowledge.(i.e. "If someone who actually knows enough about these could give a quick layman's sketch of what's going on.") Layperson is the preferred, gender-inclusive term.

leech

In computing and specifically on the Internet, being a leech or leecher refers to the practice of benefiting, usually deliberately, from others' information or effort but not offering anything in return, or only token offerings in an attempt to avoid being called a leech. It also refers to inappropriately using another sites bandwidth - by 'hotlinking' or directly embedding images hosted on one site into another unauthorized one. This results in the host site paying for bandwidth while the 'leech' site benefits from the pageviews.

log message

When troubleshooting a complex web application (like a content management), there are times that sending a message with a log message will help. When using the development environment of an application or as on Drupal.org issue's or posting's , the "*log message*" will show up in the development which will make following the [thread](#) much easier.

[Back to Top](#)

M

Mac

1. A Macintosh Apple computer.

mock-up

In common usage, a mock-up is a scale model of a structure or device, usually used for teaching, demonstration, testing a design, etc.

moratorium

1. A freeze on a particular type of development or activity. 2. A delay, postponement or waiting period.

mojo

:**1.** is a term commonly encountered in the African-American folk belief called hoodoo. A mojo is a type of magic charm, often of red flannel cloth and tied with a drawstring, containing botanical, zoological, and/or mineral curios, petition papers, and the like. **2.** On Drupal.org *mojo* is meant "more authoritative" than the post or issue authoring.

MySql

Database Management System which is available for Linux Windows and Mac. (Mostly referred as database Server because is the mostly and broadly used database software system on the internet).

[Back to Top](#)

N

na, n/a

1. abbreviation for "**not applicable**" (for inputs, eg survey forms), "**not available**" (for

outputs, eg database reports). Occasionally seen on Drupal sites where theme functions or expected data for those functions is 'unavailable'. This indicates a small misconfiguration or synchronization problem. Can occasionally be fixed by executing 'update.php'

nod (on)

Lower and raise the head, as to indicate assent or agreement or confirmation; "The teacher nodded when the student gave the right answer". (i.e. express or signify by nodding; "He nodded his approval"). Note: verb: nodding.

NNE(s)

Non-native English speakers, in the meaning of "English is not the mother language-tongue".

Note: *Acronym was first used in the discussion Thread about the existence of a [D.O. Documentation Task force](#) and commented in [Article - for a new Drupal Documentation Project](#)*

neophyte

New member of a community, someone who has not yet been initiated.

newbie

Individual lacking the knowledge of application functions. Someone who has little or no knowledge of computer programs.

newcomer

One who has recently come to a community; a recent arrival; A new participant in some activity; a neophyte.

no-brainer

1. anything that requires little thought. 2. An easy or obvious conclusion, decision, solution, task, etc.; something requiring little or no thought

noob

Newbie, is an online slang term for a newcomer, or someone with little knowledge of a given activity or topic. Sometimes considered disparaging.

novice

Beginner; one who is not very familiar or experienced in a particular subject.

[Back to Top](#)

O

OPML

(Outline Processor Markup Language) This is an XML format that allows exchange of outline-structured information.

[Back to Top](#)

P

patch(es)

is a small piece of software designed to update or fix problems with a computer program or its supporting data. This includes fixing bugs, replacing graphics and improving the usability or performance.

path

is generally site-specific and refers to the means by which a resource is located. This could be a full [URL](#), or a relative location (such as "files/xyz/image.jpg" - where "files/xyz" would be the path to the file "image.jpg").

peer (to)

(IT) When two programs are sending data to each other over a network, they are peers; client or user trying to get authenticated.

PHP

PHP (or *PHP: Hypertext Preprocessor*) is one of the programming languages that Drupal is written in.

PIFR

[Project Issue File Review](#), a project that runs on Drupal.org's [Quality Assurance](#) server.

PITA

reads: Pain in the ass, pain in the arse.

Pitfall(s)

An unapparent source of trouble or danger; a hidden hazard.

production site

The live version of a site. When a site is in production, it's available to the public.

[Back to Top](#)

Q**query**

question: an instance of questioning; (i.e. "there was a question about my training"; "we made inquiries of all those who were present")

queue

A job management technique. Jobs waiting in a line (queue) are usually processed on a first in, first out basis or by priority, if specified.

On Drupal.org exist a issue queue for each Site maintenance departments:

- the [Documentation Queue](#) dedicated to [Documentation Administration](#)
- the [Webmaster Queue](#) dedicated to [Site Administration](#)
- the [Infrastructure Queue](#) dedicated to [Hardware & Software Administration](#)

Note: To post or create an Issue on drupal.org You must [get a drupal.org account](#)

[Back to Top](#)

R**RCS**

Revision Control System. A system of managing multiple revisions of files. RCS is useful for text that is revised frequently.

RTBC

Reviewed and Tested By the Community. [More about Drupal Issue status levels.](#)

RTL (Right-to-left)

Refers to languages such as Arabic and Hebrew that [run from right to left across a written page](#), and need to to the same on a web browser. RTL support can be problematic in theme design, but is possible. [Drupal theme guide for RTL](#).

RTFM

Read the freaking manual (sometimes a less polite term is used in place of "Freaking")

[Back to Top](#)

S**server**

Generally a computer that provides services. These services may be things like running the database ([MySQL](#)) or managing the gathering and dissemination of information.

site-maintainer

This is a composed expression that is build from two words:

1. **Site**: A file section of a computer on which files reside; for example, a Web site, a Gopher site, an FTP site.
2. **maintainer**: Someone who ports a product to a distribution. A person who does maintenance work.

For more info see ([Drupal.org Membership types Table](#))

slang

An informal non-standard vocabulary composed typically of coinages, arbitrarily changed words, and extravagant, forced, or facetious figures of speech

SQL

(Structured **Q**uery **L**anguage) is a database access language that originated on mainframes and minicomputers, and which is now popular on PCs. MySQL is a popular *implementation* of SQL favored by Drupal developers.

status of issues

Each Drupal.org issue has a status assigned so that we can tell at a glance what progress has been made with each issue.

- *active*: ([Dev.](#)) No patch is attached to the issue. ([Doc.](#)) (to be defined)
- *active (needs more info)*: There is insufficient information in the issue to proceed.
- *patch (code needs review)*: abbr. **CNR** (Dev.) A patch has been created and needs review and testing. (Doc.) (to be defined)
- *patch (code needs work)*: abbr. **CNW** (Dev.) The patch needs additional work before it should be reviewed. (Doc.) (to be defined)
- *patch (reviewed & tested by the community)*: **RTBC** (Dev.) The patch has received a thorough review and test by one or more experienced developers. (Doc.) (to be defined).
- *patch (to be ported)*: (Dev.) The patch has been committed to a branch of the project. (Doc.) (to be defined)
- *fixed*: (Dev.) The issue has been resolved (usually by committing a patch). (Doc.) (to be defined).
- *duplicate*: A similar issue has already been created.
- *postponed*: The issue seems like a good idea, but other (often related) issues need to be dealt with first.

- *won't fix*: It has been decided that this issue will not be fixed.
- *by design*: (Dev.) The raised issue has been deemed not to be an issue. (Doc.) (to be defined).
- *closed*: The issue is no longer current.

For more informations see [Status levels of Issues](#) and [Issue Submit Form - Description Table](#), discussion.

Stub

1. [programming] A dummy procedure definition that has a name and parameters but whose body is not yet implemented.
2. [programming] A trivial procedure whose purpose is to simply provide a route to other functions that actually to the work - created for naming consistency or API abstraction.
3. [content management] A page on a site that's been created, but doesn't have any real content on it yet. A placeholder used for prototype sitemapping. See [Wikipedia:Stub](#)

[Back to Top](#)

T

tag
 [programming] A command inserted in a document (mostly in [HTML](#) documents) that specifies how the document, or a portion of the document, should be formatted. [content management] "Tag" or "tagging" is now widespread as a means of loosely classifying content. Flickr etc use "tags" to add categories to images or posts. In Drupal, applying "tags" to content is one of the simplest uses of "taxonomy", and the word "**Term**" is used to describe what some other systems call "tags".

terminology

A set of terms representing the system of concepts of a particular subject field. **Note**: for more informations see [The importance of Terminology](#)

theme

is a means of manipulating and describing how you want your content displayed to your visitors. This includes elements such as your header, icons, block layout, etc. It also includes programming and style sheets.

TOC

Table of contents

topic(s)

1. What the text is about. The topic is not the same as the main idea. 2. (or discussion topic) – A series of threads that have been posted that all concern a more specific topic such as quality assuring appraiser training.

thread(s)

In online discussions, a series of messages that have been posted as replies to each other. A single forum or conference typically contains many threads covering different subjects. By reading each message in a thread, one after the other, you can see how the discussion has evolved.

tutorial

A manual designed to teach novices how to implement and use a procedure or system of,

usually in a step-by-step instructions. (i.e.) [The Drupal Cookbook](#)

[Back to Top](#)

U

usability

Usability is a property of websites (and other systems and products) that relates to ease of use. Usability is commonly defined as having three core components: effectiveness (how well a task can be completed), efficiency (how easy or quick it is to complete the task), and satisfaction (the user's perception or opinion of the system).

URL(s)

Uniform **R**esource **L**ocater) is the "address" of a resource (such as a page of content) on the web. It is the way the web browser locates your content or site. You will see the URL listed in the address bar on your browser.

URI

Uniform **R**esource **I**dentifier is a formatted string that serves as an identifier for a resource, typically on the Internet. Also (uri). URLs are a subset of URIs. Generally, the term URIs is used when talking about ports, services and protocols beyond the common http: document-request types common on web pages. The exact distinction between the two is confusing, and largely unimportant.

UX (User Experience)

An umbrella term referring to the overall experience and satisfaction a user has when using a website. It is important to note that this *includes only what the user perceives and not all that is presented*.

[Back to Top](#)

V

VCS

Version **C**ontrol **S**ystem. See also [cvs](#) , [RCS](#)

Vhost

vhost, or virtual host, 1. Virtual hosting is a method that servers such as web servers use to host more than one domain name on the same computer, sometimes on the same IP. 2. on IRC is a method to mask a user's real IP address to protect privacy and prevent Denial-of-service attacks. **Note**: for more info see [How-To: Virtual Hosting with Drupal](#)

vocabulary

Usually alphabetized and explained collection of words e.g. of a particular field

[Back to Top](#)

W

WAMP

1. WAMPs are packages of independently-created programs installed on computers that use a Microsoft Windows operating system.

2. **Windows, Apache, MySQL, and PHP** -- a popular combination of software for use as a web server.

weblog

see [Weblog](#)

web syndication

The practice of making content available to other sites through web feeds. Generally, a title and summary are displayed on the syndicating site with a link to the primary content page. The common method for this is RSS

[Back to Top](#)

X

XAMPP

Apache HTTP Server, MySQL database, PHP and Perl programming languages.

[Back to Top](#)

Y

YAD

1. the pointer used by the reader, often shaped like a "hand." 2. Source: A yad (Hebrew: יָד), literally, "hand," is a Jewish ritual pointer, used to point to the text during the Torah reading from the parchment Torah.

YAG

speak: Yet Another Group

YMMV

(Your Mileage May Vary) 1. A ritual warning often found in Unix freeware distributions. Translates roughly as "Hey, this works for me, but who knows what'll happen on your system?". 2. More generally, a qualifier attached to advice. "I find that sending flowers works well, but your mileage may vary." **Note:** in a comment used on Drupal.org was refer to *Browser service structure development*.

[Back to Top](#)

Z

zebra striping

Alternating the background colors of rows of data. This is often used in tabular data where rows of data alternate background colors between white and a shade of gray.

[Back to Top](#)

Thanks to:

Contributors: [Shannon Lucas](#), [Mauror](#), [Mike Stewart](#), [Dman](#), [JohnNoc](#), [other initiators](#).

Notes:

1. Most of the terms listed and published here were defined with the use of the Google "define" tool. For visitors and [D.O.](#) members who are non-native English speakers, not confident with the English language, or have

"... a bit of trouble with the fine points of English nuance."

(to cite one of the Documentation Team members [@Dman](#)), please feel free to [create an issue](#). The unclear term, expression or [acronym](#) will then be added to this list.

2. For those interested in helping with the building of this "terminology" reference for Drupal.org, please take a quick look how it came about and read the [post](#) on the original author's motivations.

Third party resources

There are a variety of third party sites that offer excellent Drupal-related information and functionality. For more general reading, you may also want to read the aggregated postings from hundreds of Drupal-related sites via [Drupal Planet](#).

General

- [Drupal Sites](#): a directory listing of Drupal-powered sites with tags and ratings

Modules

- [Drupal Modules](#): a site featuring reviews and ratings of Drupal [contributed modules](#)

Themes

- [The Theme Garden](#): a site allowing you to demo many of the themes listed for download in the [contributed themes](#) section.

Podcasts and video

- [Lullabot Podcast](#): a weekly audio podcast, 30 to 90 minutes long, about a variety of different Drupal topics, from geeky to super geeky

Drupal community

- [Groups.Drupal.org](#): home to user groups, module- and topic- specific working groups, as well as Drupal job postings

Note: these are a few selected resources that are maintained by the community. This is not meant to be a link farm or other resource to raise the page rank of the linked to sites, but rather a pointer to a handful of sites that have already proven themselves generally useful. Typically, personal or company blogs / sites are not included unless they have extensive "how to" or supplementary information. If you believe your site fits this criteria, feel free to [submit an issue](#) to the [webmasters](#) queue requesting you be added with details about your site. A site maintainer will review it and approve it if appropriate.

Drupal distributions

Information on quick pre-configured Drupal installations, usually including a selection of modules.

At drupal.org:

- [Installation profiles](#) - Customized "distributions" that enable and configure a set of modules that work together for a specific kind of site.
- [Distribution profiles](#) - Drupal group.

Other resources:

- [Acquia Drupal](#) - A collection of essential Drupal software, packaged together to build dynamic community websites faster. Free Drupal with optional paid support. There is also a stack installer for test sites on PC, including Apache, MySQL and PHP.
- [BitNami Drupal Stack](#) - Free, pre-configured, all-in-one native installers, virtual appliance and cloud images of Drupal plus all dependencies such as Apache, MySQL and PHP. Runs on Linux, Windows and OS X.
- [Brainstormblogger](#) - distribution of Drupal 6 intended for easy creation of standalone blogs on Drupal. The goal was - get configured system for blogger right after install.
- [DrupalEd](#) - A distribution designed for education sites that offer courseware, online lessons, etc.
- [Open Atrium](#) - An intranet/extranet that allows teams to have their own conversations, weblogs, wiki pages, calendars, to-do lists, a shoutbox, and a dashboard to manage it all. Open source, from Development Seed.
- [Pressflow](#) - A derivative of Drupal core providing enhanced performance, scalability and data integrity for high traffic sites. Open source, from Four Kitchens.
- [Tattler](#) - A topic monitoring tool for journalists, researchers, advocates and communications professionals that finds and aggregates content on specified topics.

Lists that might also facilitate an initial module selection:

- [Drupal.org: Module browsing by usage](#). Browse and filter modules, ordered by [reported usage](#).
- [Drupal.org: Project usage overview](#) - Modules can be ordered by [reported usage](#) -ascending or descending order- clicking on the dates.
- [Drupal Modules: Most Downloaded](#) - Download data from a Drupal site.

Language-specific communities

Listed here are the language-specific Drupal communities that offer excellent Drupal related resources and information. These sites have already proven themselves generally useful to their respective language/countries.

If you wish to be included in this list, please add your request to the [webmasters issue queue](#).

- [Drupal Belarus](#) (Беларуская)
- [Drupal Belgium](#) (Nederlands)
- [Drupal Brazil](#) (Português)
- [Drupal Catalan](#) (Català)
- [Drupal China](#) (中文)
- [Drupal Croatia](#) (Hrvatska)

- [Drupal Czech](#) (Český)
- [Drupal Denmark](#) (Dansk)
- [Drupal Esperanto](#) (Esperanto)
- [Drupal France](#) (Français)
- [Drupal Finland](#) (Suomi)
- [Drupal Germany](#) (Deutsch)
- [Drupal Greece](#) (Ελλάς)
- [Drupal Hungary](#) (Magyarország)
- [Drupal Hispano](#) (Español)
- [Drupal India](#) ()
- [Drupal Israel](#) (עברית)
- [Drupal Italy](#) (Italiano)
- [Drupal Japan](#) (日本語)
- [Drupal Korea](#) (한국어)
- [Drupal Norway](#) (Norsk)
- [Drupal Persian/Iranian](#) ()
- [Drupal Poland](#) (Polska)
- [Drupal Russia](#) (Русский)
- [Drupal South Africa](#) (Afrikaans)
- [Drupal Taiwan](#) (中文)
- [Drupal Thailand](#)

These third-party sites offer different functionalities that cater specifically to their language/country. Most of these communities have language-specific support forums, handbooks, directories of country-specific Drupal sites, companies and freelancers, and offer Drupal workshops and seminars. More often than not, these sites are also the home of the translation projects of their respective languages.

Didn't find your language-specific community site here? [Drupal Vietnam](#) and [Drupal Sweden](#), for example, are to be found at [Groups.Drupal.org](#). Turkish users can check the [turkish user group](#) for more information on their community.

Note that the third-party sites listed above often have their corresponding groups at g.d.o. as well. See the [geographical listing](#) of user groups at g.d.o.

Tips on using the documentation

Tips on using the documentation handbooks:

- [Searching](#)
- [Browsing](#)
- [Asking](#) where documentation is located
- [Saving](#) for off-line use

Searching the Documentation handbooks: enter your search term in the Drupal search box and click Search. On the results page, click Book page in the Filter by type right-hand menu to limit your search results to those in the documentation only.

Browsing the documentation handbooks: click on <http://drupal.org/handbooks> to see a wide variety of guides, tutorials and case studies.

Asking the location of documentation, in case you can't find what you are looking for, can be done as follows:

- Posting a topic in one of the [support forums](#).
- Asking on the [IRC](#) (chat) channel `#drupal-support`.

Saving copies of any the documentation handbooks for off-line use can be accomplished as follows:

1. From the [handbooks tab](#), go to the top level page of one of the handbooks.
2. Click "Printer-friendly version" link at the bottom of the page. This link is at the bottom of the main content area of *book* pages, above any comments that may have been added. [Note: You MUST be logged in to see this option]
3. If you know the node number of the book page, then it has a path in the form "node/14279", and you can substitute "book/export/html/14279" to get the printer-friendly version.
4. The current page and its sub-pages will be output in a clean format with no headers, footers, or side blocks.
5. Comments to book pages are not included in the printer-friendly copies.
6. Save the content for off-line use, either by printing, by saving as a .pdf, or saving as HTML.
7. This works for sub-sections of a book as well, but there is no way to output all books at once.

Create an offline copy of Drupal documentation

Off-line copies are files of the handbooks that can be downloaded and read offline, like any other file. To create offline copies, you must be logged-in.

1. From the [handbooks tab](#), go to the top level page of one of the handbooks.
2. Click "Printer-friendly version" link at the bottom of the page. This link is at the bottom of the main content area of *book* pages, above any comments that may have been added. You must be logged in to see the link.
3. If you know the node number of the book page, then it has a path in the form "node/14279", and you can substitute "book/export/html/14279" to get the printer-friendly version.
4. The current page and its sub-pages will be output in a clean format with no headers, footers, or side blocks.
5. Comments to book pages are not included in the printer-friendly copies.
6. Save the content for off-line use, either by printing, by saving as a .pdf, or saving as html.
7. This works for sub-sections of a book as well, but downloading all handbooks at once is currently not supported.