+pThe average content management system (CMS) is like a toy truck—specific assumptions have been made about how it will be used, and these assumptions are difficult to override.

Drupal is designed to be the perfect content management solution for non-technical users who needs both simplicity and flexibility. Content management frameworks, on the other hand, are like the raw materials needed to make any toy—no assumptions have been made about how they’ll be used, rather a collection of wheels, windshields, axles, frames, etc., that a toy maker can easily connect together. With Drupal, a maker could create a toy truck, but she or he could just as easily create a toy airplane, submarine, or robot.

Commented

and the builder needs expert technical knowledge in order to make anything at all.

It accomplishes this through its modular approach to site building. Unlike other CMSs, Drupal isn’t a prefabricated toy truck, but rather a collection of wheels, windshields, axles, frames, etc., that a toy maker can easily connect together. With Drupal, a maker could create a toy truck, but she or he could just as easily create a toy airplane, submarine, or robot.

ou want the homepage to have a section featuring the five most recent ones.

In other ordinary cms u vl need to download plugins but in case of drupal u can very achieve this by using views.

Drupal, however, treats most content types as variations on the same concept: a node (more on this in a moment). Static pages, blog posts, and news items (some possible node types) are all stored in the same way, and the site's navigation structure is designed separately by editing menus, views (lists of content), and blocks (side content which often have links to different site sections).

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 menu system, as well as taxonomy (tagging of content) and views, creates an information architecture. Finally, the theme system, along with display modules like Panels, controls how all this looks to site visitors.

**Node**

Essentially, a node is a set of related bits of information. When you create a new blog post, you are not only defining its body text, but also its title, content, author link, creation date, taxonomy (tags), etc.

A node is the generic term for a piece of content on your web site. The content type of the node will define what fields are included with it. Depending on the type of node, different fields will be attached, and this is known as a content type. For example, a basic Pagecontent type has attached fields such as title and body fields. Other examples of content type are: Book pages for use in Books, Discussion topics in forums, Blog pages in blogs, and News articles.

**Drupal Flow**

1.To display anything u need to have some data to be shown this is at the base of the system and collection of nodes

2. The next layer up is where modules live. Modules are functional plugins 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.

3. t 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 (Regions) 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.

**Menus** are a core element of Drupal. Menus are a navigator of Drupal which provide links to all the pages created in Drupal.

4. Next are user permissions. This is where settings are configured to determine what different kinds of users are allowed to do and see. Permissions are defined for various roles, and in turn, users are assigned to these roles in order to grant them the defined permissions.

5. On the top layer is the site theme (the "skin"). This is made up predominantly of XHTML and CSS, with some PHP variables intermixed, so Drupal-generated content can go in the appropriate spots.

* **Custom Functionality:** Are you frustrated with web site builders or limitations of Wordpress? Drupal may be the answer. Drupal is highly customizable and even allows the integration of web applications and mashups using third party APIs.
* **Flexible Implementation:** Drupal allows your website to evolve in any direction. For example, you may start with a blog but then want the option of adding other features like a wiki, electronic commerce, forums, multiple content types, etc.
* **Complex Components:** Complex forms, workflows, multilingual sites, and multi-sites can be set up easily with Drupal. There are also over 16,000 Drupal modules available to provide excellent functionality for your website, from commerce to directories and image galleries.
* **Configurable Website:** Drupal can easily be configured to interact with other sites or technologies. If you want, you can even configure it to interact with another Drupal site!
* **Customize-able Content Types:** With Drupal, you have ability to create your own content types. For example, you can create a custom format for directory listings or[km,+maqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqmmmmmmmmmmmmmmqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqa<Q](https://www.drupal.org/project/cck)(CCK module prebuild in drupal7 etc). This is one of the most used features of Drupal.
* **List, Sort, & Search Information:**  
  You can quickly [organize and display lists of information](https://www.drupal.org/project/views).

**Node (Content)**

A *node* is the generic term for a piece of content on your web site. The *content type* of the node will define what fields are included with it. Depending on the type of node, different fields will be attached, and this is known as a *content type*. For example, a basic *Page*content type has attached fields such as title and body fields. Other examples of content type are: *Book pages* for use in Books, *Discussion topics* in forums, *Blog pages* in blogs, and *News articles*.

The word "node" is not meant in the mathematical sense as part of a network.

For more useful materials on content administration in Drupal, see [Resource Guide: Tools for Content Administration in Drupal](https://www.drupal.org/resource-guides/content-administration).

**Entity types**

An *entity type* is a useful abstraction to group together fields. Entity types are used to store and display data, which can be nodes (content), comments, taxonomy terms, user profiles, or something custom developed.

Read more about Entities in the [Entity API](http://drupal.org/node/1261744) documentation.

**Comment**

Comments are another type of content you can have on your site (if you have enabled the core Comment module). Each comment is typically a 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 known as *taxonomy*. This is provided by the core Taxonomy module. You can define your own *vocabularies* (groups of *taxonomy terms*) and add terms to each vocabulary. 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.

Read more about this concept in the [taxonomy module](http://drupal.org/documentation/modules/taxonomy) documentation.

**User**

A user is a type of entity which represents a real-world website user. By default, a user has a set of properties including their username, password, role, and e-mail address. However, they may also have other properties provided by other modules, and can be extended with new fields. For example, you could add a new "Link" field for a user's Twitter address.

**Module**

A module is software (code) that extends Drupal functionality. Modules fall into one of three categories:

* **Core** modules are those included with the main download of Drupal. These can be turned on or off without downloading additional components. Examples include *Blog, Book, Poll*, or *Taxonomy*.
* **Contributed** modules are downloaded from the Modules download section of drupal.org, and installed within your Drupal installation. Examples include *Panels, Views*or *Metatag*.
* **Custom** modules are modules you write yourself. This requires a thorough understanding of Drupal, PHP programming, and Drupal's API.

For a collection of useful materials about module development, see [Module Development with Drupal](https://www.drupal.org/module-development).

**Regions & Blocks**

Pages on your Drupal site are laid out in *Regions*. These can include the header, footer, sidebars,featured top,featured bottom, main content regions etc.. . 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 static chunks of HTML or text, menus (which are for site navigation)

, the output from modules (e.g. hot forum topics), or dynamic listings that you've created yourself (e.g. a list of upcoming events).

**Menus**

There are four standard menus in Drupal 7:

* The **Main** menu is built by site administrators and displayed automatically in the page header of many themes (and if not, you can enable their blocks to display them).
* **Management** is the administration menu, and is presented in the Admin toolbar.
* **Navigation** is a catch-all menu that usually contains links supplied by modules on your site.
* **User** menu contains links to the User account and the logout link.

You can also create your own custom menus, and display them by enabling their blocks.

You can customize menus in several ways, such as reordering menu items by setting their “weight” or simply dragging them 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.

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. For example, the admin menu item is not shown to visitors who are not logged in.

**Theme**

The *Theme layer* is separate from the data layer, the functionality extension layer (module) and Core. Theme controls the appearance (look and feel) of your site, or how your site is displayed, including the graphic look, layout, and colors. A theme consists of one or more PHP template 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.

For a collection of useful materials for themers, see [Theming and Front End Development with Drupal](https://www.drupal.org/theming).

**Views**

Although not all sites have Views, most sites include the Views module because of the excellent tools it provides. Views allows people to choose a list of nodes or other entities and present them as pages, blocks, RSS feeds, or other formats. The main use case for views is to create dynamically updating lists of content (for example, a listing of latest news), based on properties of that content (in the case of the news listing, that the content type is “News” and sorted by publication date).

**Drupal as MVC**

[**http://www.garfieldtech.cobvbm/blog/mvc-vs-pac**](http://www.garfieldtech.cobvbm/blog/mvc-vs-pac)

### MVC

The most commonly-known interactive system architecture is [Model-View-Controller](http://en.wikipedia.org/wiki/Model-view-controller), or MVC. Most good desktop applications use MVC or a variant of it, sometimes with the Controller partially merged into the View. In MVC, as the pretty picture at the other end of that links shows, the Model holds data, the View is the part the user sees, and the Controller is an intermediary for business logic. Seems reasonable, right? Now take a closer look.

### PAC

A less publicized but still widely used architecture is [Presentation-Abstraction-Control](http://en.wikipedia.org/wiki/Presentation-abstraction-control), or PAC. The two main differences between MVC and PAC are that in PAC the Presentation component is "dumb" while all the intelligence resides in the Controller and PAC is layered. Again, see the pretty picture.

You'll notice that the Presentation and Abstraction components never speak to each other. The Controller takes input, not the display component. The Controller has all the business logic and routing information. The Presentation component is essentially just a filter that takes raw data that the Controller pushes through it and renders it to HTML (or WML, or XML, or text, or an icon in a graphical monitoring system, or whatever). It's just a templating system.

Drupal

 Drupal is very much a PAC architecture. In fact, it's a rather good PAC architecture. The menu system acts as the Controller. It accepts input via a single source (HTTP GET and POST), routes requests to the appropriate helper functions, pulls data out of the Abstraction (nodes and, in Drupal 5, forms), and then pushes it through a filter to get a Presentation of it (the theme system). .

Drupal bootstrapping

Drupal architecture

Hierarchy mvc

Currently the closest Drupal comes to MVC is the [Panels module](http://drupal.org/project/panels). Panels allows the site admin to set up a custom layout with custom regions and then pull blocks, Views, or nodes into it in those regions. Currently that's the only real pull-based logic that Drupal supports for display given the active discouragement of database access (even via accessors like node\_load()) in theme\_functions and template files. Even that, however, is still limited to specific panel pages. There's no supported way to randomly pull a block into a node page, for instance. I know Earl "merlin" Miles is deeply engrossed in Panels 2.0, The Next Generation, but I don't believe it currently involves turning Panels inside out. He is, of course, welcome to correct me if he does. ;-)

Menu depth 9

Db count unlimited

Php 5.2

Staging db connect

$databases = array(

'default' =>

array(

'default' =>

array(

'database' => 'db\_expert\_search\_new',

'username' => 'root',

'password' => '',

'host' => 'localhost',

'port' => '',

'driver' => 'mysql',

'prefix' => '',

),

),

'db1' => array(

'default' =>

array(

'database' => 'db\_expert\_search\_new',

'username' => 'root',

'password' => '',

'host' => 'localhost',

'port' => '',

'driver' => 'mysql',

'prefix' => '',

),

),

);

Robots no sites- because alt title metatag ..prevents crawling

Htaccess : change url of old sie

Append www to datamatics

[www.dmatics.com/Index.php](http://www.dmatics.com/Index.php)

Taxonomy

User

Custom menu

# **Users, permissions, and roles**

Last updated May 30, 2016. Created on March 10, 2013.Edited by*[davidjguru](https://www.drupal.org/u/davidjguru" \o "View user profile.)*,*[avinashm](https://www.drupal.org/u/avinashm" \o "View user profile.)*,*[gdaw](https://www.drupal.org/u/gdaw" \o "View user profile.)*,*[chriscerk](https://www.drupal.org/u/chriscerk" \o "View user profile.)*.[*Log in to edit this page*](https://www.drupal.org/user?destination=node/1938536).

Every visitor to your site, whether they have an account and log in or visit the site anonymously, is considered a user to Drupal. Every user also has a numeric user ID special to the type of user.

## **Types of Users**

### Master Administrator

This user has the ID one (1). User of ID one (1) is the primary admin user account created during Drupal installation. This user is very special because it has permission to do absolutely everything on the site.

### Logged In

These users are assigned a user ID when they register for the website. A user name and email address is associated with any user that isn't anonymous (therefore must be logged in).

### Anonymous

Anonymous users who visit the website but do not login all share a user ID of zero (0).

#### Associating more information with Users

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.

## **Permissions**

Other users on your site can be assigned permissions via roles. To do this, you first need to create a role by navigating to people --> permissions --> role. A common role is "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.

<http://yourenergysavings.gov.au/information/energy-efficient-appliances>

<http://yourenergysavings.gov.au/actions/buy-energy-efficient-appliances>

<http://yourenergysavings.gov.au/information/energy-rating-labels>

<http://eartheasy.com/live_energyeffic_appl.htm>

<http://cleantechnica.com/2013/11/03/7-ways-reduce-electricity-bill/>

drupal features -- terms my exp

reuasibilty --

FEATURE

modules

security - not easy to break, informs us

it makes seasy to create n manage

translates anything using the site with built in translate ui

an open source no licensicng cost

socu=ial media contents publish

What is drupal

cms, open php, manage organise pulish,non technical simplicity flexibilty, raw material blocks, menus and .. integrate it together and create any page as per ur require,ment

CORE DRUPAL compo

modules menu taxonomy,poll, theme

NODE

is a entity

entity

entity type 4 commnts nodes users taxo

feature

patch

glossary

**CDN**

This module provides easy Content Delivery Network integration for Drupal sites. It changes file URLs, so that files (CSS, JS, images, fonts, videos …) are downloaded from a CDN instead of your web server.

assets can be cached in a proxy which is geographically closer to the end user, which usually leads to lower latency and increased download speed;

Each page response is shared between your origin server and the CDN, meaning that your origin server can serve more concurrent requests;

**Apache solr**

Solr (pronounced "solar") is an open source enterprise search platform, written in Java, from the Apache Lucene project.

Its major features include full-text search, hit highlighting, faceted search, real-time indexing, dynamic clustering, database integration, NoSQL features[1] and rich document (e.g., Word, PDF) handling.

Providing distributed search and index replication, Solr is designed for scalability and fault tolerance.[2]

Solr is the second-most popular enterprise search engine after Elasticsearch.

However, when the number of documents to search is potentially large, or the quantity of search queries to perform is substantial, the problem of full-text search is often divided into two tasks: indexing and searching. The indexing stage will scan the text of all the documents and build a list of search terms (often called an index, but more correctly named a concordance). In the search stage, when performing a specific query, only the index is referenced, rather than the text of the original documents.[2]

The indexer will make an entry in the index for each term or word found in a document, and possibly note its relative position within the document. Usually the indexer will ignore stop words (such as "the" and "and") that are both common and insufficiently meaningful to be useful in searching.