

Taxonomy

Drupal's powerful classification system

A talk for FOSDEM 2009 by *Bart Feenstra*

Introduction

Good afternoon everybody. Welcome to FOSDEM. My name is Bart Feenstra and I study communication studies at the University of Twente in The Netherlands. Drupal is my main hobby. I am the maintainer of several modules, including External link filter, External Search and Vocabulary Index and I write patches for Drupal core. What I like about Drupal is the logic and flexibility that can be found everywhere in Drupal.

This session we will talk about Taxonomy. It is my intention to tell you about Taxonomy in a clear and simple fashion, so you will leave with a good understanding of how it works and what you can do with it.

How many of you are using Drupal? And how many have experience with Taxonomy? For those of you who don't know what it is: Drupal is a content management framework and Taxonomy is its classification system.

The difference between categorisation and Taxonomy

Show a filing cabinet, captioned 'Categorisation'.

Show some Post-It notes, captioned 'Taxonomy'.

Most content management systems offer a categorisation system where each piece of content may be put in only **one** category. Taxonomy works in a more flexible way. In Drupal, every piece of content is a node. Whether they are blog posts, articles, maps, events or book reviews, they're all nodes. Using Taxonomy, you can create *vocabularies*. These are comparable to main categories. Site administrators may select which vocabularies should be available to which node types.

Show a diagram with a vocabulary at the top.

A vocabulary called *Tags*, for instance, may only be available to blog posts and articles. Vocabularies are mere containers, meaning they cannot contain nodes directly.

Extend the diagram in the previous slide with terms, hanging under the vocabulary.

This is where *terms* come into play. Terms may be considered subcategories, which are placed in vocabularies.

Extend the diagram with one or more nodes under some, but not all, terms.

If you want to classify a node, you may assign several terms to it, depending on what vocabularies are available to that node type and how many terms from a certain vocabulary may be assigned to each node.

Show a new diagram with a node at the top and several terms and their vocabularies under it.

Like I just said, a node may be assigned multiple terms, which may come from multiple vocabularies. Where categorisation would allow nodes to be present in only **one** category, comparable with one of the filing cabinet's drawers, classification allows nodes to be tagged with as much terms from as much vocabularies as you'd like, which you can imagine as a bunch of Post-Its stuck to a node.

Book versus Taxonomy

Drupal comes with two modules that allow you to add structure to your content. Taxonomy is one of those two. The other one is Book. Book offers a custom node type and structures those using a node to node relationship model, while Taxonomy offers site-wide classification. With Book you can create hierarchical parent-child relationships. A good example usage of taxonomy and Book are the handbooks at drupal.org.

Demonstrate the usage of both modules at <http://drupal.org/handbooks>.

Core Taxonomy features

Taxonomy itself offers some nice features, ranging from interface elements to handy functions.

Administration

Terms may be configured either by site administrators or by users. For site administrators, there is a fancy drag and drop interface that allows you to order terms by simply pointing and clicking.

Demonstrate the drag and drop interface.

Term selectors

Show a list of core's three term selectors.

Drupal core features three types of term selectors, each of them matching certain needs. The first selector is a simple drop-down list. These are used if users may select one term only from a certain vocabulary. Hierarchy is being maintained by indenting nested terms with hyphens.

Demonstrate the single select list.

The single select selector has a bigger brother, which, you might have guessed, is the multiple select selector. This is exactly the same element as used for single selections, but with the *multiple* attribute set to *multiple*. It replaces the single select selector if *Multiple select* has been enabled for the vocabulary in question.

Demonstrate the multiple select list.

As you can see hierarchy is being maintained here as well.

Show screenshots of famous sites using free tagging.

The last, but certainly not the least selector is used for **free tagging**. Here users may enter the terms they want to apply themselves. If a user types in something that matches an existing term, the word is automatically being completed. If not, a new term with that name will be created upon node submission.

Demonstrate free tagging by actually creating a Page.

Functions

So far core's fancy Taxonomy features for end users. It's time to dig into some code. Naturally, there are functions to create, load, edit and delete terms and vocabularies, but some functions are extremely handy to know about, because they can save you loads of time.

Show taxonomy_get_tree().

taxonomy_get_tree()

The famous, and according to some also notorious, `taxonomy_get_tree()` function returns an array of term objects that are children of the provided parent. This parent may be the vocabulary itself, but also a term within that vocabulary. It also calculates a term's depth and sets the parents for each term.

Show taxonomy_get_path().

taxonomy_get_path()

Taxonomy provides `hook_term_path()` modules can use to set custom paths for certain terms. If you are writing a module that creates links to term pages, for instance, use `taxonomy_term_path()` to retrieve the correct path to those terms, rather than hardcoding `/taxonomy/term/[tid]`.

Show taxonomy_node_x().

taxonomy_node_X()

By default, taxonomy is only being applied to nodes. Taxonomy uses `hook_nodeapi()` to detect node actions and to update the taxonomy data related to a new, edited or deleted node. You can easily use those functions yourself. All you need is a `$node` object with `NID` and `VID` (revision ID) properties and an array of terms, and sometimes even less than that. For more information, take a look at `taxonomy_nodeapi()` where all node-related functions are used.

Contributions

Show a list of the three contributed modules that will be discussed.

Vocabulary Index

One of the drawbacks of Taxonomy is that the only way to browse it, is on a per-term basis. You click a term, see a list of nodes that have this term applied to them and that's it. What about actually browsing your vocabularies? Vocabulary Index allows you to do this in a number of very familiar ways; If you are using hierarchically structured vocabularies, you might want to display their terms in a tree or browse them like you would browse the folders on your harddisk. If hierarchy is not important, but the term names are, you can easily browse them using an alphabetical pager.

Demonstrate Vocabulary Index.

Tagadelic

Another way of browsing through your vocabularies that's extremely simple and popular at the same time is using tag clouds. The essence of tag clouds is that they display the tags used on a website, terms in Drupalspeak, all huddled together. Font size and color are used to distinct between more and less popular terms.

Demonstrate Tagadelic.

Relevant Content

When viewing nodes, Relevant Content displays a block that informs the user about other content that is related to the node he's looking at, by term comparison. A drawback of this module is that related nodes aren't looked up randomly. No matter how much nodes are related to the one you're looking at, you will always see the same list of relevant content.

Demonstrate Relevant Content.

Category

Category is a contributed module that replaces Taxonomy and allows you to categorise your nodes in a combination of how Taxonomy and Book would do it. Category has **nothing** to do with Taxonomy and if people talk about categories, they are often referring to vocabularies or terms. If you want to try Category, **DO NOT DO THIS ON A LIVE SITE**, since the module may cause side effects and is extremely hard to uninstall, if that is even possible at all, according to its maintainer.