

7

Advanced Content

Most of our dealings with content so far, have been fairly basic in that they require us only to learn which settings to enable, and what text to enter. There is a fundamental difference between that and what is coming in this chapter, mainly because the content in this chapter requires us to think ahead, and plan what we want ahead of time, in order to prevent things going awry at a later date.

One of the most important aspects to managing content is the manner in which it is best organized for expedient retrieval – and for this, we need to discuss taxonomy. Taxonomy is what makes Drupal's classification system so powerful, and it is left for us to decide how best to implement. It might sound a little strange at first, but we will see later on in the chapter why this faculty of Drupal is one of the features that distinguishes it from everything else out there – it's really a good thing!

Being able to categorize information is one thing, but the ability to create entirely new content types and post complex pages will also come in handy at some stage. Accordingly, this chapter discusses the following subjects:

- Taxonomy
- Content Construction Kit (CCK)
- HTML, PHP, and Content Posting

The skills learned during the process of content classification, creation, and management will prove useful not only for this website but also in other aspects of life – whether it is creating and managing office reports for your boss, building a new website, or even writing a book. That is because, by and large, we are now going to learn *how content should be managed and created* rather than how to click buttons and links to enable or disable settings.

Taxonomy

At first glance, it might seem that taxonomy is yet another term indicating that your job is going to be more complex for some reason or other. After all, it's perfectly reasonable to set up a website to allow blog writers to blog, forum posters to post, administrators to administer, or any other type of content producer to produce content and leave it at that. With what we have covered so far, this is all quite possible, so why does Drupal insist on adding the burden of learning about new concepts and terms?

If your site is never going to gather a substantial amount of content (perhaps it is only meant as a more static, placeholder type of site), then spending time working with taxonomies and so on is probably not going to bring much advantage—go ahead and enable whatever content types you require and let users add whatever they want.

However, the aim is not generally to remain in obscurity when creating a website, so assuming that you do want to attract a community of users, then the method of categorizing content in Drupal makes it one of the most sophisticated content management systems around!

Take the time to master working with taxonomy in Drupal, because not only will this help you to work out how to manage content better, but it will also really set your site apart from others because of the flexible and intuitive manner in which the content is organized. These attributes allow you to manage a site of pretty much any size imaginable (just in case what you are working on is "*the next big thing*").

What and Why?

Taxonomy is described as the science of classification. In terms of how it applies to Drupal, it is the *method by which content is organized* using several distinct types of relationship between terms. Simple as that! This doesn't really encompass how useful it is, though, but before we move on to that, there is a bit of terminology that to pick up first:

- **Term:** A term used to describe content (also known as a *descriptor*)
- **Vocabulary:** A grouping of related terms
- **Thesaurus:** A categorization of content, which describes *is similar to* relationships
- **Taxonomy:** A categorization of content into a hierarchical structure
- **Tagging:** The process of associating a term (descriptor) with content
- **Synonym:** Can be thought of as *another word* for the current term. It may help to view the following diagram in order to properly grasp how these terms inter-relate:

This serves to illustrate the fact that there are two main types of vocabulary. Each type consists of a set of terms, but the relationship between them are different in that a *taxonomy* deals with a hierarchy of information, and a *thesaurus* deals with relationships between terms. The terms (shown as small boxes) and their relationships (shown as arrows) play a critical role in which type of vocabulary you use.

We have already seen an example of a taxonomy when the **Forum** module was discussed. In this case, there was a hierarchical relationship between forum containers and the forum topics they contained. But what would we need thesauri for? For one thing, if you were working on creating a scientific document and wanted to allow plenty of references between terms so that users could browse related pages, which didn't necessarily have child-parent relationships, then you would go for this type of structure.

What we have discussed so far is how to *control a taxonomy* from the administrator's point of view. It is also possible to pass that control on to everyone who uses the site to by creating a *free taxonomy*. One of the things that makes the Drupal taxonomy system so powerful, is that it allows content to be categorized on the fly (as and when it is created). This unburdens administrators because it is no longer necessary to moderate every bit of content coming into the site in order to put it into pre-determined categories.

We'll discuss both methods in some detail in the coming sections, but it's also worth noting quickly that it is also possible to tag a given node more than once. This means that content can belong to several vocabularies, at once. This is very useful for cross-referencing purposes because it highlights relationships between terms or vocabularies through the actual nodes.

Let's begin...

Implementing Controlled Taxonomies in Drupal

The best way to talk about how to implement some form of categorization is to see it in action. There are quite a few settings to work with and consider in order to get things up and running. Let's assume that the demo site has enlisted a large number of specialists who will maintain their own blogs on the website so that interested parties can keep tabs on what's news according to the people in the know.

Now, some people will be happy with visiting their blog of choice and reading over any new postings there. Some people, however, might want to be able to search for specific topics in order to see if there are correlations or disagreements between bloggers on certain subjects. As there is going to be a lot of content posted once the site has been up and running for a few months, we need some way to ensure that specific topics are easy to find regardless of who has been discussing them on their blogs.

Introduction to Vocabularies

Let's quickly discuss how vocabularies are dealt with in the administration tool in order to work out how to go about making sure this requirement is satisfied. If you click on the **Taxonomy** link under **Content management**, you will be presented with a page listing the current vocabularies. Assuming you have created a forum during the last few chapters, you should have something like this:

Taxonomy [List](#) [Add vocabulary](#)

The taxonomy module allows you to categorize your content using both tags and administrator defined terms. It is a flexible tool for classifying content with many advanced features. To begin, create a 'Vocabulary' to hold one set of terms or tags. You can create one free-tagging vocabulary for everything, or separate controlled vocabularies to define the various properties of your content, for example 'Countries' or 'Colors'.

Use the list below to configure and review the vocabularies defined on your site, or to list and manage the terms (tags) they contain. A vocabulary may (optionally) be tied to specific content types as shown in the *Type* column and, if so, will be displayed when creating or editing posts of that type. Multiple vocabularies tied to the same content type will be displayed in the order shown below. To change the order of a vocabulary, grab a drag-and-drop handle under the *Name* column and drag it to a new location in the list. (Grab a handle by clicking and holding the mouse while hovering over a handle icon.) Remember that your changes will not be saved until you click the *Save* button at the bottom of the page.

[\[more help...\]](#)

Name	Type	Operations
Forums	Forum topic	edit vocabulary list terms add terms

Before we look at editing terms and vocabularies, let's take a look at how to create a vocabulary for ourselves. Click on the **add vocabulary** tab to bring up the following page that we can use to create a vocabulary, manually

Taxonomy List Add vocabulary

Define how your vocabulary will be presented to administrators and users, and which content types to categorize with it. Tags allows users to create terms when submitting posts by typing a comma separated list. Otherwise terms are chosen from a select list and can only be created by users with the "administer taxonomy" permission.

[\[more help...\]](#)

Identification

Vocabulary name: *

The name for this vocabulary, e.g., "Tags".

Description:

Description of the vocabulary; can be used by modules.

Help text:

Instructions to present to the user when selecting terms, e.g., "Enter a comma separated list of words".

Content types

Content types:

Blog entry

Book page

By way of example, this vocabulary will deal with the topic of hunting, and there are a couple of notes to guide users when they intend to submit a blog entry. This only applies to blog entries because that is the only content (or node) type for which this vocabulary is enabled — you can select as many or as few as you like, depending on how many content types this vocabulary should apply to.

Looking further down the page, there are several other options that we will discuss in more detail, shortly. Clicking on **Submit** adds this vocabulary to the list, so that the main page now looks like this:

Created new vocabulary *Hunting*.

The taxonomy module allows you to categorize your content using both tags and administrator defined terms. It is a flexible tool for classifying content with many advanced features. To begin, create a 'Vocabulary' to hold one set of terms or tags. You can create one free-tagging vocabulary for everything, or separate controlled vocabularies to define the various properties of your content, for example 'Countries' or 'Colors'.

Use the list below to configure and review the vocabularies defined on your site, or to list and manage the terms (tags) they contain. A vocabulary may (optionally) be tied to specific content types as shown in the *Type* column and, if so, will be displayed when creating or editing posts of that type. Multiple vocabularies tied to the same content type will be displayed in the order shown below. To change the order of a vocabulary, grab a drag-and-drop handle under the *Name* column and drag it to a new location in the list. (Grab a handle by clicking and holding the mouse while hovering over a handle icon.) Remember that your changes will not be saved until you click the *Save* button at the bottom of the page.

[\[more help...\]](#)

Name	Type	Operations		
⊕ Forums	Forum topic	edit vocabulary	list terms	add terms
⊕ Hunting	Blog entry	edit vocabulary	list terms	add terms

So far so good, but this will not be of much use to us as it stands! We need to add some terms (descriptors) in order to allow tagging to commence.

Dealing with Terms

Click on **add terms** link for the **Hunting** vocabulary to bring up the following page:

Add term to *Hunting* [List](#) [Add term](#)

▽ Identification

Term name: *

Trapping
The name of this term.

Description:

A form of hunting that relies not on stealth but on mechanical devices to capture or kill animals...

A description of the term. To be displayed on taxonomy/term pages and RSS feeds.

▷ Advanced options

[Save](#)

The term **Trapping** has been added here, with a brief description of the term itself that guides contributors. We could, if we choose, associate the term **Poaching** with **Trapping** by making it a related term or synonym. Click on the **Advanced options** link to expose the additional features as shown here:

▽ Advanced options

Parents:

<root>
Poaching
Trapping

Parent terms.

Related terms:

<none>
Poaching
Trapping

Synonyms:

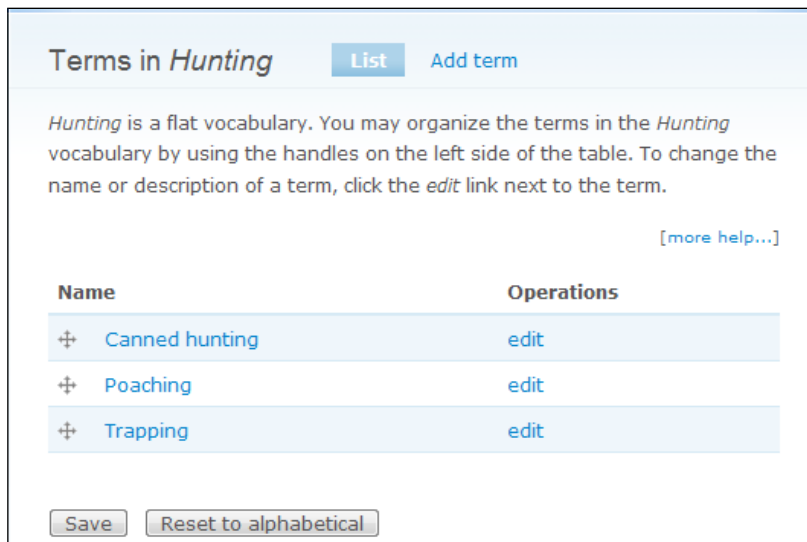
gin traps

Synonyms of this term, one synonym per line.

In this case, the term **Trapping** is specified as being related to **Poaching** and by way of example, **gin traps** is a synonym. Synonyms don't actually do anything useful at the moment, so don't pay too much mind to them yet, but there are modules that expose additional functionality based on related terms and synonyms such as the *Similar By Terms* module.

The **Parents** option at the start of the **Advanced options** warrants a closer inspection, but as it relates more closely to the structure of hierarchies, we'll look at it in the section on *Hierarchies* that's coming up.

For now, add a few more terms to this vocabulary so that the list looks something like this:



Terms in *Hunting* [List](#) [Add term](#)

Hunting is a flat vocabulary. You may organize the terms in the *Hunting* vocabulary by using the handles on the left side of the table. To change the name or description of a term, click the *edit* link next to the term.

[\[more help...\]](#)

Name	Operations
+ Canned hunting	edit
+ Poaching	edit
+ Trapping	edit

[Save](#) [Reset to alphabetical](#)

It's now time to make use of these terms by posting some blog content.

Posting Content with Categories Enabled

Using any account with the requisite permissions to add blog content, attempt to post to the site. You should now be able to view the newly inserted **Categories** section as shown here:

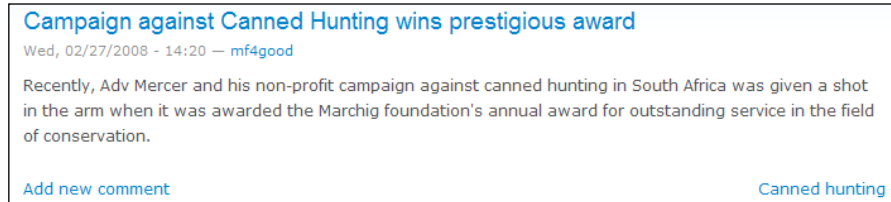


Hunting:

Canned hunting ▼

Associates your posts with the general topic of hunting

Now comes the clever bit! Once this blog node has been posted, users can view the blog as normal, except that it now has its term displayed along with the post (bottom right):



Where does the descriptor link take us? Click on the term, in this case **Canned hunting**, and you will be taken to a page listing all of the content that has been tagged with this term. This should really have you quite excited, because with very little work, users can now find focused content without having to look that hard — this is what content management is all about!

Hierarchies

What we have seen this far is really only the tip of the iceberg. You can build an entire hierarchy of terms in a vocabulary to give you a fairly complex taxonomy. Remember that if it is a hierarchy you are building, then the broadest terms should be towards the top of the pile, with the more focused terms near the bottom. At the moment, though, we don't really have a *hierarchy*, but rather, more of a *flat* structure.

What if we wanted a set of more specific terms that would allow bloggers to tag their content (which focuses on specific types of *Trapping*, for example)? The answer lies in restructuring the vocabulary by dragging and dropping its terms not only up and down the list but right to left — this is done when viewing the **list terms** page of the vocabulary.

For this example, I added a term entitled **Snaring** to the vocabulary, and then dragged it under and to the right of the term **Trapping** to indicate that it is lower in the hierarchy:

Terms in *Hunting* [List](#) [Add term](#)

Hunting is a flat vocabulary. You may organize the terms in the *Hunting* vocabulary by using the handles on the left side of the table. To change the name or description of a term, click the *edit* link next to the term. [\[more help...\]](#)

Name	Operations
+ Canned hunting	edit
+ Poaching	edit
+ Trapping *	edit
+ Snaring *	edit

* Changes made in this table will not be saved until the form is submitted.

[Save](#) [Reset to alphabetical](#)

Saving this change leaves us with the same page, only the description of the hierarchy has moved from flat to single:

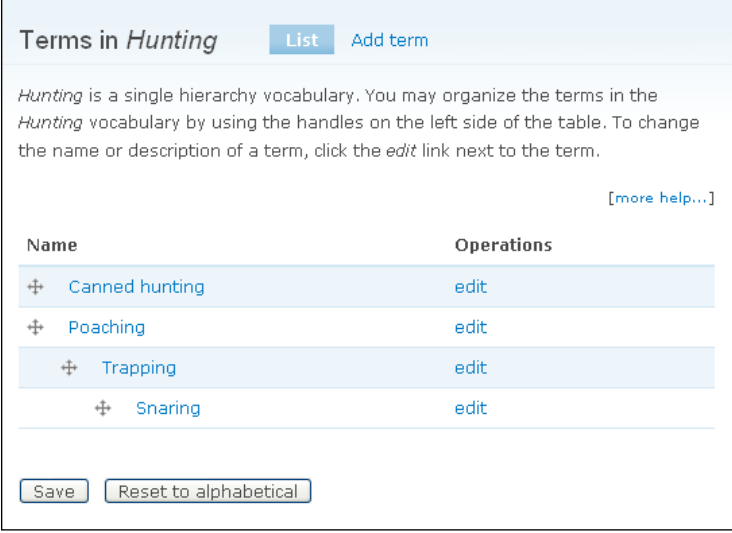
Terms in *Hunting* [List](#) [Add term](#)

Hunting is a single hierarchy vocabulary. You may organize the terms in the *Hunting* vocabulary by using the handles on the left side of the table. To change the name or description of a term, click the *edit* link next to the term. [\[more help...\]](#)

Name	Operations
+ Canned hunting	edit
+ Poaching	edit
+ Trapping	edit
+ Snaring	edit

[Save](#) [Reset to alphabetical](#)

That was fairly easy to do, and now we are free to create either flat hierarchies or single depth ones (i.e. one parent term with one child term – no grandchildren). If you wanted to create a deep hierarchy structure, then this is easily achieved by dragging either additional terms under **Snaring**, or moving **Trapping** under something else, like this:



The screenshot shows a web interface titled "Terms in *Hunting*". It includes a "List" button and an "Add term" button. Below the title is a paragraph explaining that *Hunting* is a single hierarchy vocabulary and that terms can be organized using handles on the left side of the table. A "[more help...]" link is also present.

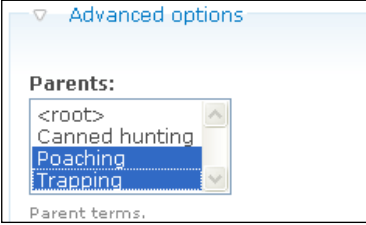
Name	Operations
+ Canned hunting	edit
+ Poaching	edit
+ Trapping	edit
+ Snaring	edit

At the bottom of the interface are two buttons: "Save" and "Reset to alphabetical".

This should not be confused with creating multiple hierarchies – notice that the hierarchy description in this screenshot still describes **Hunting** as a single hierarchy vocabulary.

But what happens if your topic is slightly more complex than a straightforward hierarchy? For example, it's quite possible that the terms **Pits** (referring to hunting pits) could be equally at home under both **Trapping** and **Poaching** (which in turn may also have multiple parents). In the event that one term has several parent terms, the phrase used to describe this structure is *multiple hierarchy*.

Recall that when dealing with terms previously, there was an **Advanced option** in the term edit page that allowed us to specify one or more parent terms. Selecting more than one parent, like so:



The screenshot shows a dialog box titled "Advanced options" with a dropdown arrow. Below the title is the label "Parents:" followed by a list box containing the following items: "<root>", "Canned hunting", "Poaching", and "Trapping". The "Poaching" and "Trapping" items are highlighted in blue. Below the list box is the text "Parent terms.".

Leads Drupal to warn us with the following page:

Set multiple term parents?

Adding multiple parents to a term will cause the *Hunting* vocabulary to look for multiple parents on every term. Because multiple parents are not supported when using the drag and drop outline interface, drag and drop will be disabled if you enable this option. If you choose to have multiple parents, you will only be able to set parents by using the term edit form.

You may re-enable the drag and drop interface at any time by reducing multiple parents to a single parent for the terms in this vocabulary.

Basically, it is necessary to warn users that the normal drag and drop facility for vocabularies are not implementable when terms have a complex hierarchy involving several parents – that said, drag and drop will still be enabled if it is at all possible and the structure will still be shown on the **List** page. If you want a multiple hierarchy, then the structural editing of the hierarchy must be done by hand in each term's **edit** form.

Go ahead and click **Set multiple parents** – you might want to add a few terms and set each of these to have multiple parents to make the structure a little more complex. With that done, note that the drag and drop features of the list page are disabled:

Terms in *Hunting*

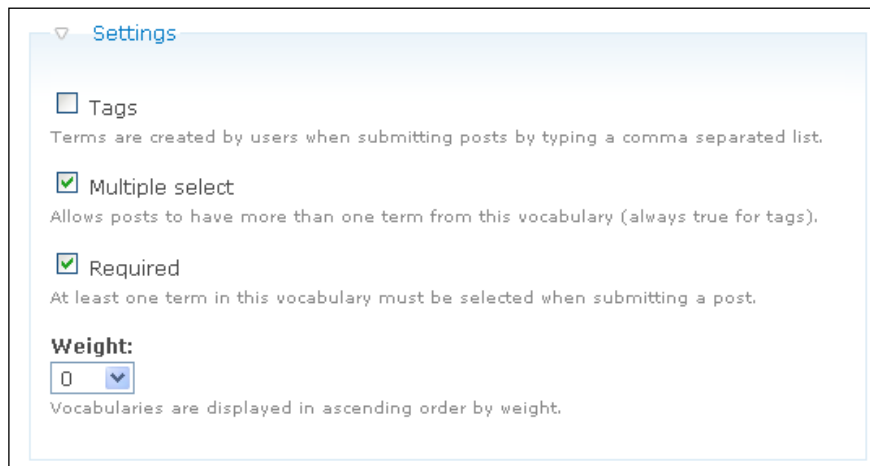
Updated term *gin traps*.

Hunting is a multiple hierarchy vocabulary. To change the name or description of a term, click the *edit* link next to the term. Drag and drop of multiple hierarchies is not supported, but you can re-enable drag and drop support by editing each term to include only a single parent.

[\[more help...\]](#)

Name	Operations
Canned hunting	edit
Pits	edit
Poaching	edit
gin traps	edit
Pits	edit
Pits	edit
Trapping	edit
gin traps	edit
Pits	edit

The hierarchy structure is useful when the topics of discussion fall fairly neatly into some sort of natural hierarchy – forums are the best example of this. However, it may well be that a given piece of content overlaps several terms and should really be tagged with more than one term. To achieve this, head back to the vocabulary editing page and select the **Multiple select** option in the **Settings** section:



Settings

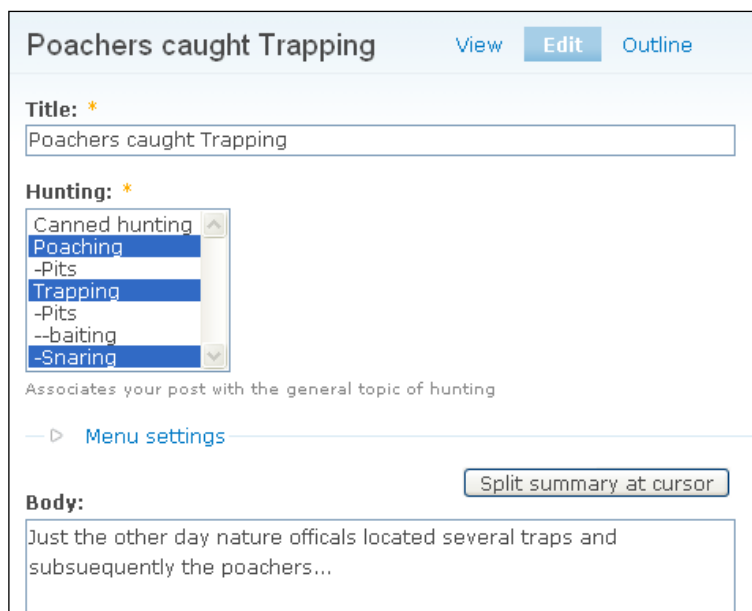
Tags
Terms are created by users when submitting posts by typing a comma separated list.

Multiple select
Allows posts to have more than one term from this vocabulary (always true for tags).

Required
At least one term in this vocabulary must be selected when submitting a post.

Weight:
0
Vocabularies are displayed in ascending order by weight.

Save this and then post some new content. Now, instead of being presented with a single term to associate with the post, it is possible to select as many as are relevant:



Poachers caught Trapping [View](#) [Edit](#) [Outline](#)

Title: *
Poachers caught Trapping

Hunting: *
Canned hunting
Poaching
-Pits
Trapping
-Pits
--baiting
-Snaring

Associates your post with the general topic of hunting

[Menu settings](#)

Body:
Just the other day nature officials located several traps and subsequently the poachers...

When this post is viewed on the site, it has several tags associated with it, and users can click on any of these tags to immediately locate more content that is of specific interest to them:

Poachers caught Trapping
Thu, 01/24/2008 - 16:15 — clean

Just the other day nature officials located several traps and subsequently the poachers...

[clean's blog](#) [Poaching](#) [Snaring](#) [Trapping](#)

Notice that the terms presented do not, in anyway, indicate their underlying structure to the reader – it simply tells them that these are all terms of this bit of content.

Content Structure

What if, in the demo site's case, we have the term **Trapping** available to tag content with (blog posts in this case), but someone is really talking about something other than hunting entirely, and there happens to be some sort of content overlap? An example scenario might be as follows:

- Several specialists are contracted to maintain blogs about the African continent.
- They tag their content using a new **Africa** vocabulary, which contains terms like **nature, gazelle, predators, lakes, rivers, mountains, hunting, weather, and tourism.**
- You wish to be able to allow material that is created from the Africa blogs to be cross-referenced by hunting-related topics in the **Hunting** specialists' blogs.

In order to achieve this, it is necessary to create a new vocabulary called **Africa**. Attach this vocabulary to the blog content type, and then create several terms, ensuring that one of them is entitled **Hunting** as follows:

Terms in *Africa* [List](#) [Add term](#)

Africa is a single hierarchy vocabulary. You may organize the terms in the *Africa* vocabulary by using the handles on the left side of the table. To change the name or description of a term, click the *edit* link next to the term. [\[more help...\]](#)

Name	Operations
+ Commerce	edit
+ Hunting	edit
+ Tourism	edit

Now, when users attempt to post content, they are presented with not one but two options to classify their content, and assuming you have correctly ordered the vocabularies on the **Taxonomy** page, you can apply a kind of hierarchy to the tags. For example, a blog post on poaching by one of the **Africa** bloggers might look like this:

The screenshot shows a 'Create Blog entry' form. The title field contains 'Canned hunting in South Africa'. Below the title, there is a 'Vocabularies' section. Under 'Africa', the 'Hunting' tag is selected. Under 'Hunting', the 'Canned hunting' tag is selected from a dropdown menu. The dropdown menu also shows other options: 'Poaching', '-Pits', 'Trapping', '-Pits', '--baiting', and '-Snaring'. A note at the bottom of the form states: 'Associates your post with the general topic of hunting'.

Once this is posted to the site, it is then possible to view both categories on the content page instead of just one. In other words, the node has been tagged with several terms in what is known as **faceted tagging**.

Faceted tagging uses a *bottom up* system of classification, where facets or properties of the content are described by the terms. In this way, a very intuitive method of classifying content can be created without users needing to understand the top-down path of a content hierarchy in order to find the content they are after. Ironically, in this case, the specific method of tagging used here helps to elucidate the hierarchy of terms too (i.e. **Canned hunting** is a child of **Hunting**):

The screenshot shows a blog post titled 'Canned hunting in South Africa' by 'mf4good' on 'Tue, 11/06/2007 - 11:35'. The post content is: 'This article talks about hunting, specifically canned hunting, that takes place mainly in South Africa but also throughout sub-Saharan Africa...'. At the bottom, there are links for 'mf4good's blog', 'Add new comment', 'Hunting', and 'Canned hunting'.

Taking a look at this posting on the site confirms that users can now go directly to both the **Hunting** and **Canned hunting** category pages by clicking on the links provided in the posting.

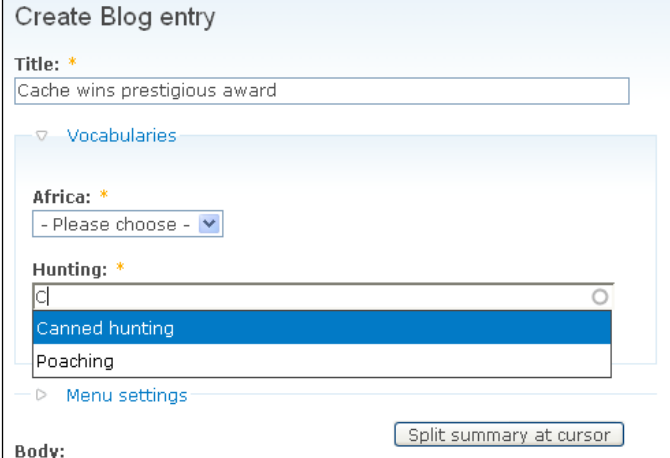
What happens if one of the **Hunting** bloggers simply wants to make an entry and tag it with the **Canned** term from the **Hunting** vocabulary, without having to first specify that this content also belongs to the **Africa** vocabulary? The answer lies once again in editing the vocabulary page, which contains a **Required** checkbox right at the bottom. If this option is enabled, then posters must select at least one tag from the vocabulary, but if we leave it unselected, then posters can choose whether to include a term from that vocabulary or not.

Talking of new options, there is one more that we should take a look at quickly – tagging. Since tagging has a number of considerations to consider before implementation, we treat it in its own section.

Implementing Thesuari in Drupal (Tags)

Tagging is an interesting option because it allows posters to choose their own terms for their content. While posters effectively have free reign when it comes to tagging their posts, Drupal understands that a hundred different people might come up with a hundred different terms to describe the same post, and this can be very detrimental to the usability of the site.

In order to combat this effect, Drupal provides helpful clues to keep the tagging of posts as uniform as possible, without placing restrictions on what can and is used for tagging. Enabling **Tags** for the **Hunting** vocabulary, for example, means posters are given the following category options when creating a blog entry:



The screenshot shows the 'Create Blog entry' form. The 'Title' field contains 'Cache wins prestigious award'. Below it is a 'Vocabularies' section with a dropdown menu for 'Africa' (set to '- Please choose -') and a dropdown menu for 'Hunting' (set to 'Canned hunting'). A 'Menu settings' link is visible below the 'Hunting' dropdown. At the bottom right of the form is a 'Split summary at cursor' button. The 'Body' field is partially visible at the bottom.

Notice that there is a red asterisk superscript above the **Hunting** category. This is because despite the fact that we are using free tagging, the **Required** option on the **edit vocabulary** page is still enabled – so something *has* to be entered here. Secondly, there is a drop-down list of all the tags available (starting with whatever letter you type). This means that giving people free reign to type in their own tags is not as random as it may at first seem, because they can still be guided as to what terms are already available using this drop-down list. In this way, Drupal can encourage a more coherent body of terms.

"But **Poaching** doesn't begin with a C", some of you may be remarking. That is quite correct, but **Poaching** contains a c so it is displayed here nevertheless – it's a good way to provide a range of available tags that narrows down quickly as the user types.

Tagging has some pros in that it is far more flexible. People can tag their content exactly as they please – making the tagging system fit the content far more snugly. The problem is, however, that the vocabulary may well become unwieldy, because similar content could be tagged with entirely different terms, making it hard for users to find what they are looking for.



Allowing free tagging of content is a very powerful method for categorizing content. Be wary though, it can lead to a lot of redundant tags, which in turn lead to content that is hard to find.

You should make note of the fact that it is not possible to create a hierarchy of terms using the free tagging system, because every new tag is on the same level as all the other tags. So what you end up with is really a *thesaurus*, instead of a taxonomy (hence this section's heading).

Remember that it is still possible to moderate a thesaurus because any and all terms that posters create will still be added to the list of terms in the vocabulary, and they can be viewed, edited or deleted as you like.

It is interesting to note that a middle ground between controlled taxonomies and free tagging is achievable using the already mentioned **Multiple select** option and disabling **Free tagging**. This allows users to tag their posts with as many terms as are made available by the creator of the vocabulary – giving you control over the terms used, and posters the freedom to choose which ones they make use of.

With that we come to the end of the discussion on taxonomy. As mentioned when we first began working on this section, it may take a little while to get the hang of things, because the way in which the categorization works in Drupal is not always immediately intuitive. However, once you have mastered it, you will find that your content is readily accessible and well organized with very little effort.

Content Construction Kit (CCK)

It is likely that at some stage, you will want to upgrade at least some content from plain text to something that looks a little *out of the ordinary*. Drupal provides the CCK module as a way to build custom content types that can be tailored to suit your needs. In effect, it gives you control over which *fields* are presented to a user whenever they post content using custom content types.



The term *field* refers to a given piece of content within a node. Conversely, a node is a collection of fields.

In addition to the basic field types provided by the CCK module, you should also keep an eye out for contribs that extend CCK functionality to provide a huge range of useful field enhancements. Everything from Brazilian id numbers to validated email fields, voting widgets and Amazon ASINS have been made available in the past.

There are also a number of other modules that make use of CCK in a variety of ways. Most important among these is *Views*. Views provide administrators with the means to modify how Drupal displays lists of content, and CCK exposes its fields to the **Views** module making them perfect partners when it comes to creating custom content and then displaying that content in a highly configurable manner.

At the time of writing, *Views* is not available for Drupal 6 (although the module is being actively developed and should hopefully be ready by the time you read this) so it is left as an exercise to download and install it, and create at least one new view utilizing fields created in the following sections.

Installing CCK

CCK is available, so go ahead and download the latest version and extract the file to your **modules** folder. CCK adds its own section to the **Modules** page under **Site building**:

Enabled	Name	Version	Description
<input checked="" type="checkbox"/>	Content	HEAD	Allows administrators to define new content types. Required by: Content Copy (disabled), Fieldgroup (disabled), Node Reference (disabled), Number (disabled), Option Widgets (disabled), Text (disabled), User Reference (disabled)
<input type="checkbox"/>	Content Copy	HEAD	Enables ability to import/export field definitions. Depends on: Content (enabled)
<input type="checkbox"/>	Fieldgroup	HEAD	Create field groups for CCK fields. Depends on: Content (enabled)
<input type="checkbox"/>	Node Reference	HEAD	Defines a field type for referencing one node from another. Depends on: Content (enabled), Text (disabled), Option Widgets (disabled)
<input type="checkbox"/>	Number	HEAD	Defines numeric field types. Depends on: Content (enabled)
<input type="checkbox"/>	Option Widgets	HEAD	Defines selection, check box and radio button widgets for text and numeric fields. Depends on: Content (enabled) Required by: Node Reference (disabled), User Reference (disabled)
<input type="checkbox"/>	Text	HEAD	Defines simple text field types. Depends on: Content (enabled) Required by: Node Reference (disabled), User Reference (disabled)
<input type="checkbox"/>	User Reference	HEAD	Defines a field type for referencing a user from a node. Depends on: Content (enabled), Text (disabled), Option Widgets (disabled)

There are a number of interdependent sections for this module, but all of them rely on the first option, **Content**, so go ahead and enable this first. We are going to look over all the features provided by CCK, by default, in this section. So go ahead and enable those modules that rely only on **Content**. With that done, enable the remaining options so that you end up with everything working, like this:

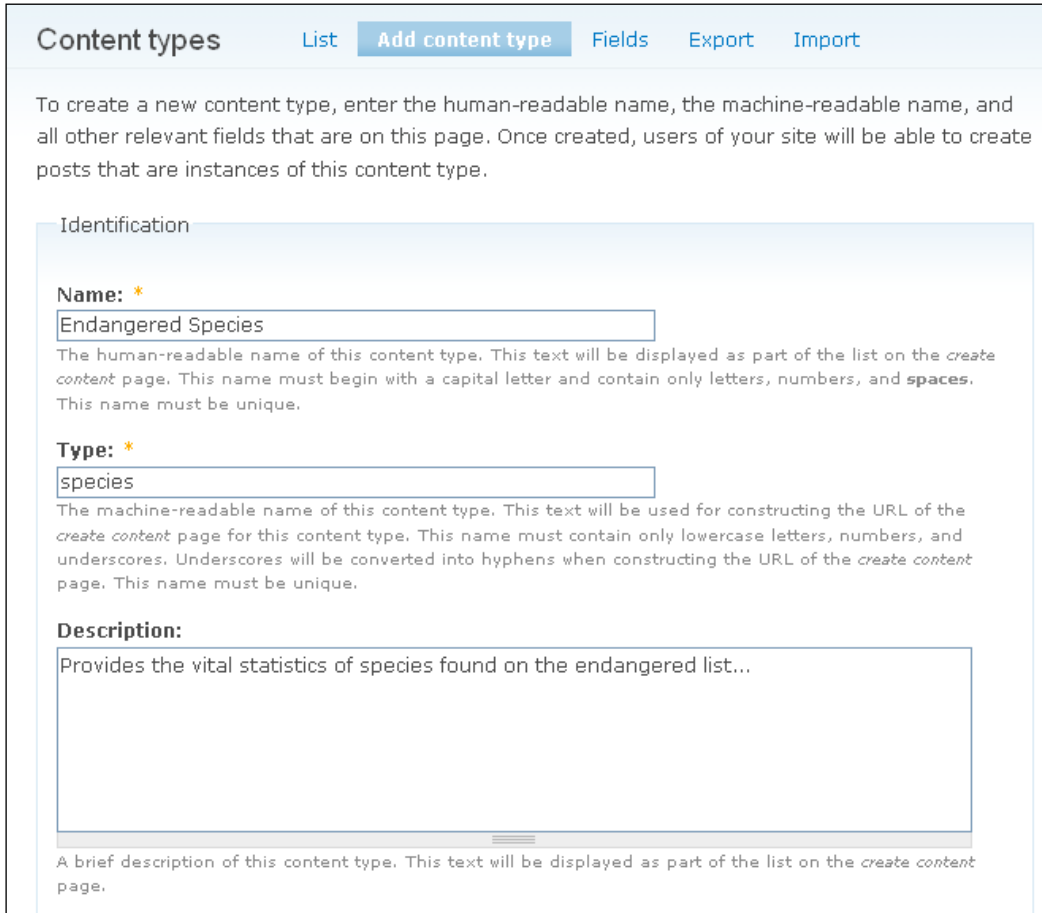
Enabled	Name	Version	Description
<input checked="" type="checkbox"/>	Content	HEAD	Allows administrators to define new content types. Required by: Content Copy (enabled), Fieldgroup (enabled), Node Reference (enabled), Number (enabled), Option Widgets (enabled), Text (enabled), User Reference (enabled)
<input checked="" type="checkbox"/>	Content Copy	HEAD	Enables ability to import/export field definitions. Depends on: Content (enabled)
<input checked="" type="checkbox"/>	Fieldgroup	HEAD	Create field groups for CCK fields. Depends on: Content (enabled)
<input checked="" type="checkbox"/>	Node Reference	HEAD	Defines a field type for referencing one node from another. Depends on: Content (enabled), Text (enabled), Option Widgets (enabled)
<input checked="" type="checkbox"/>	Number	HEAD	Defines numeric field types. Depends on: Content (enabled)
<input checked="" type="checkbox"/>	Option Widgets	HEAD	Defines selection, check box and radio button widgets for text and numeric fields. Depends on: Content (enabled) Required by: Node Reference (enabled), User Reference (enabled)
<input checked="" type="checkbox"/>	Text	HEAD	Defines simple text field types. Depends on: Content (enabled) Required by: Node Reference (enabled), User Reference (enabled)
<input checked="" type="checkbox"/>	User Reference	HEAD	Defines a field type for referencing a user from a node. Depends on: Content (enabled), Text (enabled), Option Widgets (enabled)

Notice that some of the options are disabled to prevent us from inadvertently disabling an option that is required by something else. If, for example, you wish to disable **Text**, then disable **Node Reference** and **User Reference** first.

Working with CCK

With all the options enabled, we can now go ahead and create a new content type. Actually, it is possible to create new content types without the use of CCK, it's just that the new content types will look pretty much like the standard content types already available, because there are no really interesting fields to add.

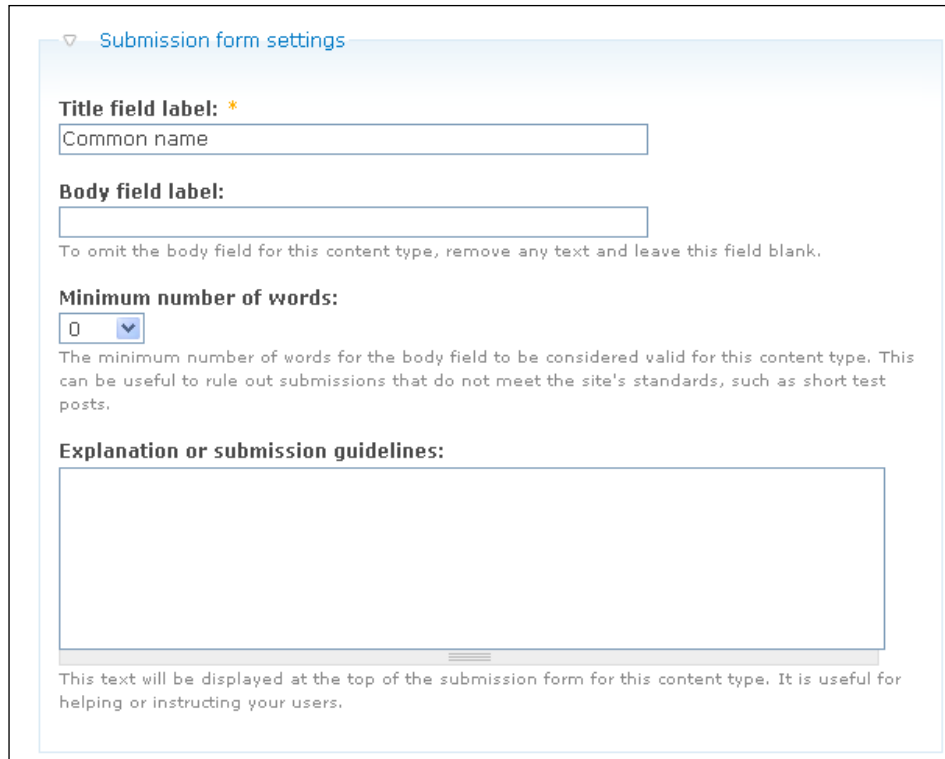
Head over to **Content types** under **Content management** and select the **Add content type** field to bring up the following page:



The screenshot shows the 'Add content type' form in Drupal. At the top, there are navigation tabs: 'List', 'Add content type' (which is active), 'Fields', 'Export', and 'Import'. Below the tabs, there is a text block explaining that to create a new content type, one must enter the human-readable name, the machine-readable name, and all other relevant fields. The form is divided into sections:

- Identification:**
 - Name: *** (required): A text input field containing 'Endangered Species'. Below it, a text block explains that this is the human-readable name, must begin with a capital letter, contain only letters, numbers, and spaces, and must be unique.
 - Type: *** (required): A text input field containing 'species'. Below it, a text block explains that this is the machine-readable name, must contain only lowercase letters, numbers, and underscores, and must be unique.
- Description:** A large text area containing the text 'Provides the vital statistics of species found on the endangered list...'. Below it, a text block explains that this is a brief description of the content type, displayed as part of the list on the 'create content' page.

The identification section is pretty straightforward. You can go ahead and fill in whatever new content settings are appropriate. Of special interest, is the **Submission form settings** below this that allows you to decide whether the default **Title** and **Body** fields should be changed or even retained (in the case of the Body field):



The screenshot shows a settings panel titled "Submission form settings" with a dropdown arrow. It contains four sections:

- Title field label:** * A text input field containing "Common name".
- Body field label:** A text input field that is currently empty.
- Minimum number of words:** A dropdown menu showing "0". Below it is a note: "The minimum number of words for the body field to be considered valid for this content type. This can be useful to rule out submissions that do not meet the site's standards, such as short test posts."
- Explanation or submission guidelines:** A large text area that is currently empty.

At the bottom of the panel, there is a note: "This text will be displayed at the top of the submission form for this content type. It is useful for helping or instructing your users."

In the case of the **Endangered Species** content type, it doesn't really make sense to have a species **Title**, rather a **Common name** makes more sense in this instance. Leaving the **Body field label** blank will cause this field to be omitted completely in the event that it is not suitable for the type of content you have in mind.

Notice too that there are several additional tabs to the right of **Add content type** tab that provide additional functionality. These options are discussed a little later on in this section. So for now, go ahead and fill out the **Name**, **Type**, and **Description** fields and click **Save content type** to add this to the default list:

Name	Type	Description	Operations
Blog entry	blog	A <i>blog entry</i> is a single post to an online journal, or <i>blog</i> .	edit
Book page	book	A <i>book page</i> is a page of content, organized into a collection of related entries collectively known as a <i>book</i> . A <i>book page</i> automatically displays links to adjacent pages, providing a simple navigation system for organizing and reviewing structured content.	edit delete
Endangered Species	species	Provides the vital statistics of species found on the endangered list...	edit delete

We are now ready to begin customizing this new type utilizing whatever options are available—depending on what is or is not enabled. It is possible to customize *any type* that is available on Drupal, including the default ones like **Blog entry** or **Poll**, but to begin with it is best to leave these alone.

To begin working on the new content type, click on **edit** in the **Endangered Species** row. We can now look at the various aspects of working with content types, beginning with...

Adding Fields

Select the **Add field** tab to bring up the following page:

Endangered Species Edit Manage fields Display fields **Add field** Add group

Create new field

Field name: *
field_ numbers_in_wild
The machine-readable name of the field. This name cannot be changed later! The name will be prefixed with 'field_' and can include lowercase unaccented letters, numbers, and underscores. You'll be able to choose a human-readable label for the field on next page

Field type: *
Choose the type of value to store and an input method from the list below.

Node Reference
Store the id of a related node as an integer value.

- Select List
- Autocomplete Text Field

Integer
Store a number in the database as an integer.

- Text Field
- Select list
- Check boxes/radio buttons
- Single on/off checkbox

This dialog allows you to specify the new field's machine readable name and then select what type of input it is going to be.

Presently, only the **Create new field** section is displayed on this page because we have yet to add new fields. Once there is at least one custom field available, this page will have an additional section allowing existing fields to be added directly (you can come back here once there are a few saved fields):

Blog entry Edit Manage fields Display fields **Add field** Add group

Add existing field

Captive Breeding Pairs (field_breeding_pairs_captivity) ▼

Add field

Create new field

Regardless, the **Create new field** list presently comprises of the following options:

Node Reference – Allows the poster to reference another node using its ID value

Integer, Decimal, Float – Allows posters to store numbers in various formats

Text – Allows posters to enter content

User Reference – Allows posters to reference other users

Remember that this list is subject to change, depending on whether you disable various components of the default package, for example, **Node Reference** or **User Reference**, or include additional modules that add field types such as **Date** or **Fivestar**.

Each value type comes with a set of options for how that data should be entered. Looking at the **Integer** type, we can see that users can be prompted for an integer with a **Text Field**, **Select list**, **Check boxes**, and **radio buttons** – in this case, the **Select list** is going to be used.

Be careful about how information is stored – it is important to be efficient. For example, don't store information as text when there is only a certain number of options available, instead, store them as a number and provide the right input type to display the various options appropriately.

To demonstrate this point, consider that at the moment, the **numbers_in_wild** field is set as an integer with the **Select list** input type. We are not going to provide a select list of every possible integer, but we are going to represent a range of numbers with an integer. For example, the value 1 will correspond to the range 1-10, 2 will correspond to 11-100, and so on.

With the new field created, the configuration page for this field (under **Manage fields**) now displays the current settings for each field. To begin with, the options in the Widget settings sections are not of much interest as we have not specified what data this field will hold. To do this, scroll down the page to the **Data settings** section. From here, you can decide on how the data will be presented to the user and whether or not the field itself will be compulsory or not:

Global settings

These settings apply to the `field_numbers_in_wild` field in every content type in which it appears.

Required

Number of values:
8

Select a specific number of values for this field, or 'Unlimited' to provide an 'Add more' button so the users can add as many values as they like.
Warning! Changing this setting after data has been created could result in the loss of data!

Minimum:

Maximum:

Prefix:

Define a string that should be prefixed to the value, like \$ or €. Leave blank for none. Separate singular and plural values with a pipe (pound|pounds).

Suffix:

Define a string that should be suffixed to the value, like m², m/s², kb/s. Leave blank for none. Separate singular and plural values with a pipe (pound|pounds).

Allowed values

Allowed values list:

1	1-10
2	11-100
3	101-500
4	501-1000
5	1001-5000
6	5001-10000
7	100001-50000
8	50000+

Along with the **Allowed values list** used to input key-value pairs, there are a few other settings that may be of use depending on what data the field should capture. **Minimum** and **Maximum** values along with **Suffix** and **Prefix** values allow for some minor input validation as well as some useful display properties like currency denominations or physical units of measurement.

The **Data settings** section is type-specific. If, for example, you wanted to save a field in text format, you will find that the following options are presented instead of the **Prefix** and **Suffix** options:

Text processing:

Plain text

Filtered text (user selects input format)

Maximum length:

The maximum length of the field in characters. Leave blank for an unlimited size.

In our case, there is a specified list of eight values that denote the range of animal numbers in the wild. Clicking on **Save field settings** should now display the field type in a drag and drop representation of the content type, as it will appear when users attempt to use it:

Endangered Species [Edit](#) [Manage fields](#) [Display fields](#) [Add field](#) [Add group](#)

Saved field *field_numbers_in_wild*.

Label	Name	Type	Operations
+ Common name			
+ field_numbers_in_wild	field_numbers_in_wild	number_integer	configure remove
+ File attachments			
+ Menu settings*			
+ Body*			

* Changes made in this table will not be saved until the form is submitted.

It's easy enough to move things around to place them exactly as required on the page – although you must remember to click **Save** to implement the changes.

To get back to the field's configuration page, click on **configure**. Now we can implement some of the **Widget settings** that were skipped the first time round:

Endangered Species settings

These settings apply only to the `field_numbers_in_wild` field as it appears in the *Endangered Species* content type.

Widget: *

Text Field

Select list

Check boxes/radio buttons

Single on/off checkbox

Label: *

Estimated Numbers in the Wild

▼ Default value

field_numbers_in_wild: *

- 1-10
- 11-100
- 101-500
- 501-1000
- 1001-5000
- 5001-10000
- 100001-50000
- 50000+

— ▶ [Php code](#)

Display in group:

<none> ▼

Select a group, in which the field will be displayed on the editing form.

Help text:

Please provide the best estimates for the numbers(not including those in captivity or breeding programs)of this species...

In this case, an informative, human readable name has been provided along with a default value for the select list. Initially, no default value options could be presented because there were none specified in the **Data settings** section. At present, we have no groups so the **Display in group** option is left as is. It is possible to use PHP to determine what the default value should be, clicking on the **Php code** link and entering it into the space provided – unless you have an exceedingly pressing reason to use this, stay clear.

Go ahead and save the changes by clicking on **Save field settings**, and head on over to the **Create content** section and create a new **Endangered Species** post:

Create Endangered Species

Common name: *
Lion

Numbers in Wild:


- 1-10
- 11-100
- 101-1000
- 1001-10000
- 10001-100000
- 50000+

Menu settings

Split summary at cursor

Body:
There are quite a few lions in the wild...

We now have, in addition to the default fields provided, a perfectly good method of specifying the number of animals in the wild by selecting the appropriate values from the select list. Furthermore, the data stored in the database is only an integer and not the entire range; so, this is a pretty efficient way of doing things.

 Be aware that representing information with an integer, while efficient, means that if you do decide to change your data, then all legacy content will reflect these changes (because the data stored is only a representation and not the actually text information being displayed).

What if, in addition to the estimated numbers of the species, a field for entering the scientific name for the animal in genus species format is required. It may also be nice to have the number of breeding pairs in captivity, the total number of captive animals, the number of wild breeding pairs, and any other statistical information people might want about the species.

The species name is not really a statistic, so ideally, it would be presented separate from the stats about the animal. CCK provides a way on intuitively grouping information like this through the use of groups...

Adding Groups

To create a group, click on the **Add Group** tab and fill out the form according to how it should be presented—options that are required should always be open, but it may be better to present less important options in collapsed form, so that users can quickly scroll over them if they are not going to fill them out:

The screenshot shows the 'Add group' form for 'Endangered Species'. The form has a title bar with 'Endangered Species' and tabs for 'Edit', 'Manage fields', 'Display fields', 'Add field', and 'Add group'. The 'Add group' tab is active. The form contains the following fields:

- Label:** * Statistics
- Form settings:** These settings apply to the group in the node editing form.
 - Style:**
 - always open
 - collapsible
 - collapsed
 - Help text:** Enter information about the various important statistical features of the species population... (text area)

Instructions to present to the user on the editing form.

This screenshot shows the **Statistics** group that will hold the vital statistics of a given endangered species. Once you have filled out the form, click **Add** to include it in the content type. The group will now be present on the drag and drop list of the **Manage fields** tab, and is presented with **configure** and **remove** options as shown here:

The screenshot shows the 'Manage fields' tab for 'Endangered Species'. The tab is active, and the 'Add group' tab is also visible. The table below shows the list of fields and groups:

Label	Name	Type	Operations
+ Common name			
+ Estimated Numbers in the Wild	field_numbers_in_wild	number_integer	configure remove
+ File attachments			
+ Menu settings			
+ Body			
+ Statistics	group_statistics		configure remove

Save

With the group added to the content type, we can now go ahead and create the other fields that will constitute the input data for this group. This includes **Estimated Numbers in the Wild**, so click on its **configure** link and modify the **Display in group** section of the **Widget settings** like so:

Display in group:

Statistics

Select a group, in which the field will be displayed on the editing form.

Click **Save field settings**, and the field is now contained within the **Statistics** group:

Label	Name	Type	Operations
+ Common name			
+ File attachments			
+ Menu settings			
+ Body			
+ Statistics	group_statistics		configure remove
+ Estimated Numbers in the Wild	field_numbers_in_wild	number_integer	configure remove

After creating a few more fields, including the **Species name**, and adding them to the **Statistics** group, we now have something like this:

Endangered Species [Edit](#) [Manage fields](#) [Display fields](#) [Add field](#) [Add group](#)

Label	Name	Type	Operations
+ Common name			
+ Species	field_species_name	text	configure remove
+ Statistics	group_statistics		configure remove
+ Estimated Numbers in the Wild	field_numbers_in_wild	number_integer	configure remove
+ Estimated Numbers in the Captivity	field_numbers_in_captivity	number_integer	configure remove
+ Captive Breeding Pairs	field_breeding_pairs_captivity	number_integer	configure remove
+ File attachments			
+ Menu settings			
+ Body			

Take note that links to each field are displayed above the drag and drop dialog for quick access to each one's configuration page. You can also re-order fields within a group, in the same way fields and groups are re-ordered within the content type. Once everything is in the right place, go back to the **Create content** and look at the new post page:

Species: *

Acinonyx jubatus

Provide the species name in genus species format...

Statistics

Enter information about the various important statistical features of the species population...

Estimated Numbers in the Wild: *

- 1-10
- 11-100
- 101-500
- 501-1000
- 1001-5000
- 5001-10000
- 10001-50000
- 50000+

Please provide the best estimates for the numbers (not including those in captivity or breeding programs) of this species...

Estimated Numbers in the Captivity: *

- 1-10
- 11-100
- 101-500
- 501-1000
- 1001-5000
- 5001-10000
- 10001-50000
- 50000+

Captive Breeding Pairs: *

- 1-10
- 11-100
- 101-500
- 501-1000
- 1001-5000
- 5001-10000
- 10001-50000
- 50000+

The **Species** (anyone know what **Acinonyx** means?) field is presented above the **Statistics** group with each field in the correct order as specified in the **Manage fields** page. So the **Manage fields** page allows us to control how the fields are presented on the content creation page, but this still leaves us with the task of presenting the captured data correctly in its various forms – be it in search results, teasers or full pages.

Displaying Fields

In order to determine how input data should be displayed on screen, click on the **Display fields** tab next to **Manage fields**, when editing a content type. This brings up two tabs that cater for **General** and **Advanced** settings, respectively:

The screenshot shows the 'Display fields' configuration interface for the 'Endangered Species' content type. At the top, there are navigation tabs: 'Edit', 'Manage fields', 'Display fields' (which is active), 'Add field', and 'Add group'. Below these are sub-tabs for 'General' and 'Advanced'. A message states: 'Configure how this content type's fields and field labels should be displayed when it's viewed in teaser and full-page mode.'

Field	Label	Teaser	Full node
Species	Above	Plain text	Plain text
Statistics	Above	fieldset - collapsed	fieldset - collapsed
Estimated Numbers in the Wild	Above	unformatted	unformatted
Estimated Numbers in the Captivity	Above	unformatted	unformatted
Captive Breeding Pairs	Above	unformatted	unformatted
Similar Animals	Above	Title (link)	Title (link)

A 'Save' button is located at the bottom left of the configuration area.

The **General** tab allows the **Teaser** and **Full node** fields and their labels to be displayed according to the settings made here. In the same way, the **Advanced** tab deals with the display of this content type in RSS items, the search index, and search results. Each field and group has a **Label** option to determine where the label is displayed relative to the data it contains. For fields, the three options are **Above**, **Inline** or **Hidden**. Groups only have two options, **Above** and **Hidden** – **Inline** would not make much sense for a group.

Setting the **Species Label** to **Above**, as shown in the previous screenshot, displays its field content like this:

Cheetah

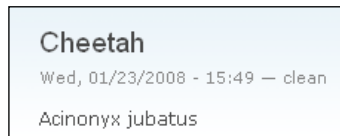
Wed, 01/23/2008 - 15:49 — clean

Species:
Acinonyx jubatus

Setting it to **Inline** gives:



And finally, setting it to **Hidden** gives:

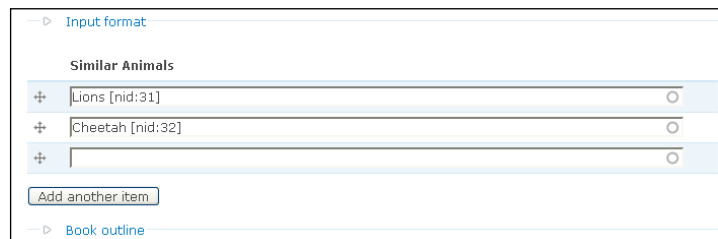


As an exercise, try out the different options available for each of the fields (and group) in the **Full node** section. For example, changing the option to **Hidden** here, produces this result:



It's important to realize that different fields have different options, depending on what underlying type they are. An example of this is the **Node reference** type that can display the **Title(Link)**, **Title(no link)**, **Full node**, **Teaser**, and **Hidden**. As mentioned earlier, the **Node reference** type allows posters to reference another node and in this case, I added a **Similar Animals** field to the **Endangered Species** content type.

Selecting the **Title(Link)** option in the **Full node** column for this type and ensuring that a few nodes have been referenced in a new post, like so (incidentally, this **Node reference** field was created with the **Autocomplete** option that allows posters to begin typing in the name of a post and select which of the available options are presented in the drop down list—the actual node number is automatically added when a choice is made):



Displays the following when the content is viewed:

Leopard
Wed, 01/23/2008 - 16:03 — clean

Similar Animals:
[Lions](#)
[Cheetah](#)

As requested, the user is presented with the titles of two other **Endangered Species** posts in the form of links that will take the user to their content. This is a really nice feature for providing connected and relevant content. Just as easily, we could have specified **Full node** for the **Similar Animals Node reference** field in the **Full node** column:

Captive Breeding Pairs	Above ▾	unformatted ▾	unformatted ▾
Similar Animals	Above ▾	Title (link) ▾	Full node ▾

in which case, the entire nodes would be displayed instead of the title links in any content that referenced them:

Leopard
Wed, 01/23/2008 - 16:03 — clean

Similar Animals:
[Lions](#)
Wed, 01/23/2008 - 15:23 — clean
Lions are the king of the jungle... and for some reason are always called Simba

Similar Animals:
[Cheetah](#)
Wed, 01/23/2008 - 15:49 — clean
Cheetahs never prosper...

How you choose to present content is really up to your own preferences and what is most expedient and useful for users in the context of how they utilize your site. It is left as an exercise to go through the settings available under the **Advanced** tab and to make use of all the default field types in the process. Remember that you need to run the cron script before any new content will show up in search results (just in case you want to play around with the Search Result column and view the results).

Export & Import

Earlier in this section, we saw that there were additional tabs added to the Content type page under Content management. The first one, **Fields**, provides a list of all the fields added to the various content types and doesn't warrant much explanation here. The other two, **Export** and **Import** provide a powerful facility to effectively *copy and paste* content types whole or piece by piece.

For argument's sake, let's assume that the **Endangered Species** content type is close but not quite the same as an **Extinct Species** content type that is going to be added. Instead of recreating the new content type from scratch, we can duplicate the **Endangered Species** type and modify it during the **Export** process, before importing it as the new type.

Click on the **Export** tab and select the **Endangered Species** content type, and click **Submit**. This brings up a list of field and group definitions that should be included in the export:

Groups:

Statistics (group_statistics)
Select the group definitions to export from *Endangered Species*.

Fields:

Estimated Numbers in the Wild (field_numbers_in_wild)

Species (field_species_name)

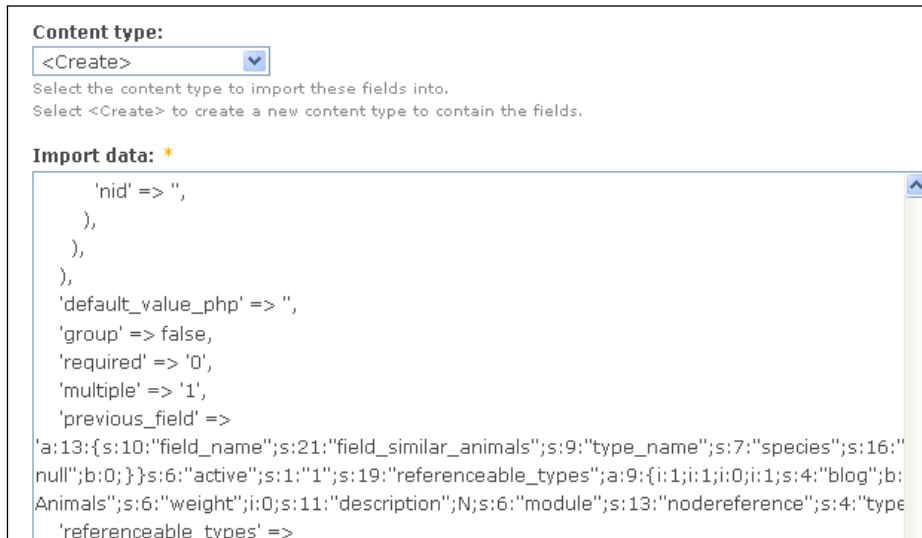
Captive Breeding Pairs (field_breeding_pairs_captivity)

Estimated Numbers in the Captivity (field_numbers_in_captivity)

Similar Animals (field_similar_animals)
Select the field definitions to export from *Endangered Species*.

For somewhat macabre reasons, we no longer need to include any of the fields previously held in the **Statistics** group. Although, the **Statistics** group itself can be used to house other stats about the extinct species. Accordingly, only the **Species** and **Similar Animals** fields need to be exported to the new content type.

Clicking on **Submit**, brings up the **Export data** text area, containing the information that will be used by the **Import** feature to build the new content type. Copy this code and then click on the **Import** tab and paste it into the space provided:



Content type:
<Create>

Select the content type to import these fields into.
Select <Create> to create a new content type to contain the fields.

Import data: *

```
'nid' => ",
),
),
),
'default_value_php' => ",
'group' => false,
'required' => '0',
'multiple' => '1',
'previous_field' =>
'a:13:{s:10:"field_name";s:21:"field_similar_animals";s:9:"type_name";s:7:"species";s:16:"
null";b:0;}s:6:"active";s:1:"1";s:19:"referenceable_types";a:9:{i:1;i:1;i:0;i:1;s:4:"blog";b:
Animals";s:6:"weight";i:0;s:11:"description";N;s:6:"module";s:13:"nodereference";s:4:"type
'referenceable_types' =>
```

In this instance, the **Content type** is left as the default option, **<Create>**, because we do not want to import these group and field definitions into any of the current content types. If you did want to add a field or two into one of the existing content types, you would select that **Content type** from the drop down list and add the new field definition (obtained from your own site or someone else's site – wherever) into the **Import data** section.

The export definition contains the content type value. In this case, we can't directly import the new definition because the **species** content type already exists, so Drupal would report a failure to import the type because is already there. To get around this, edit the first few values of the **Import data** as follows:

```
$content[type] = array (
  'name' => 'Extinct Species',
  'type' => 'extinct',
  'description' => '',
  'title_label' => 'Common name',
  'body_label' => '',
```

Now we are effectively importing a completely new type of content (called **extinct**) that happens to be based on the existing **species** content type. Clicking **Submit** will add the new type to your list and you can now view it as expected:

Content types			
Name	Type	Description	Operations
Blog entry	blog	A <i>blog entry</i> is a single post to an online journal, or <i>blog</i> .	edit
Endangered Species	species	Description of currently endangered species...	edit delete
Extinct Species	extinct	Description of recently extinct species...	edit delete

You might find that there are field definitions in other content types (yours or someone else's) that could be imported instead of being recreated when modifying content types. In this case, go through the process again, selecting the relevant content type to export from, and then, choose the field to export, copy the export code across to the **Import** section, and add the field to the relevant content type.

Adding Contributed Fields

One of the great things about CCK is that anyone can build and add custom field types that can be incorporated into your own content types. Let's take a quick look at an example of this. Head on over to the Drupal site and download both the *Voting API* and *Fivestar* modules – the *Fivestar* module provides a custom field type that we can use to add rating facilities to content, but it requires the *Voting API* module to be enabled first.

Go ahead and extract and install both modules and then, first enable the **Voting API** module then the **Fivestar** module. With that done, head on over to the **Content types** page under **Content management** and click **edit** for the content type that you would like users to be able to rate – for argument's sake, I will go with **Endangered species**. Click on the **Add field** tab, enter a field name (something like **field_rating**), and scroll down the page till you get to the *FiveStar* Rating option:

Fivestar Rating

Radio Buttons

Stars

Choose an option and click **Create field** to add it to the content type and then, ensure that you set the configuration options to something appropriate for you – each option is fairly well explained. Bear in mind that the **Node ID** field is the most important because it determines exactly, the node that receives the value of the rating. In this case, it's time to dust off your PHP skills and use it to dynamically select the appropriate node ID – admittedly, this isn't ideal but it does serve to illustrate the process for adding a contributed field, regardless of what it is.

As an addendum, Fivestar actually provides a nice rating widget that can be enabled for a content type – on its **Edit** page, in the list of settings categories, towards the bottom of the page:

The screenshot shows the 'Comment settings' page for a content type. Under the 'Fivestar ratings' section, the 'Enable Fivestar rating' checkbox is checked. The 'Number of stars' is set to 5. Below this is the 'Star Labels' section, which contains the 'Direct rating widget' settings. These settings include: 'Star display style' set to 'Display average vote value', 'Text display style' set to 'Both user and average vote', 'Show widget title' (unchecked), 'Allow users to undo their votes' (unchecked), 'Teaser display' set to '<Hidden>', and 'Full node display' set to 'Below the node body'. A 'Direct rating widget preview' is shown on the right, displaying 4 stars, 'Your rating: 4', and 'Average: 2.5 (20 votes)'.

While not shown in the previous screenshot, there is also a Fivestar comment rating widget that can be disabled, optional or required. Like the **Direct rating widget**, there is a preview available on the right-hand-side of the page. With your choices made, click **Save** settings and then go post a piece of content for the type that now has the Fivestar rating enabled.

For example, a post of the **Endangered species** type, with Fivestar enabled, looks like this:



Whenever a user makes a vote, the widget registers it for that specific node. How these votes are tallied, displayed, and so on can all be configured from the **Fivestar** and **Voting API** pages under **Site configuration**. Don't be surprised if you click the widget only to find the vote is not registered immediately.

It is possible to set the widget to update straight away, but more than likely you can leave this to update with the cron run – the rating widget can end up looking like this:



In this case, Fivestar actually provides dual functionality in that it is possible to provide a voting widget for users to rate a given piece of content as well as a contributed field type that is utilized by posters of content – this could be useful if, for example, a danger level rating should be supplied with each **Endangered species** post.

Of course, each contributed field will come with its own entourage of settings and configuration parameters, so be careful to ensure that you are entirely clear on how the contrib functions before making it available to the site's content posters.

HTML, PHP, and Content Posting

In the event that you can't find a suitable module to do a task for you, or simply want to create something yourself quickly, it's important to look at how to harness the power of HTML and PHP to get the job done.

If it's layout you are talking about, then HTML is the order of the day. Alternatively, if you want to create some dynamic content that can change depending on the state of your site, or respond to user interaction, then PHP is the way forward. More than likely, you will end up using a combination of both.

Unfortunately, we can't possibly hope to give you a comprehensive introduction into either technology in the space we have here (although we will look over HTML quickly in a moment). However, there are many online resources available to learn about HTML and PHP for free, and we will list a bunch of them throughout this section.

For now, we will look at how to achieve some fairly useful tasks by way of demonstrating how to create an *about us* page that will contain links to other useful sites, pictures of the imaginary site team as well as some dynamic content.

Input Formats and Filters

It is necessary to stipulate the type of content we will be posting, in any given post. This is done through the use of the **Input format** setting that is displayed when posting content to the site – assuming the user in question has sufficient permissions to post different types of content.

In order to control what is and is not allowed, head on over to the **Input formats** link under **Site configuration**. This will bring up a list of the currently defined input formats, like this:

Input formats [List](#) [Add input format](#)

Input formats define a way of processing user-supplied text in Drupal. Each input format uses filters to manipulate text, and most input formats apply several different filters to text, in a specific order. Each filter is designed to accomplish a specific purpose, and generally either removes elements from or adds elements to text before it is displayed. Users can choose between the available input formats when submitting content.

Use the list below to configure which input formats are available to which roles, as well as choose a default input format (used for imported content, for example). The default format is always available to users. All input formats are available to users in a role with the "administer filters" permission.

Default	Name	Roles	Operations
<input checked="" type="radio"/>	Filtered HTML	All roles may use default format	configure
<input type="radio"/>	Full HTML	No roles may use this format	configure delete

[Set default format](#)

At the moment, you might be wondering why we need to go to all this trouble to decide whether people can add certain HTML tags to their content. The answer to this is that because both HTML and PHP are so powerful, it is not hard to subvert even fairly simple abilities for malicious purposes.

For example, you might decide to allow users the ability to link to their homepages from their blogs. Using the ability to add a hyperlink to their postings, a malicious user could create a Trojan, virus or some other harmful content, and link to it from an innocuous and friendly looking piece of HTML like this:

```
<p>Hi Friends! My <a href="link_to_trojan.exe">homepage</a> is a great  
place to meet and learn about my interests and hobbies. </p>
```

This snippet writes out a short paragraph with a link, supposedly to the author's homepage, but in reality, the hyperlink reference attribute points to a trojan, `link_to_trojan.exe`. That's just HTML; PHP can do a lot more damage—to the extent that if you don't have proper security or disaster-recovery policies in place, then it is possible that your site can be rendered useless or destroyed entirely.

Security is the main reason why, as you may have noticed from the previous screenshot, anything other than **Filtered HTML** is unavailable for use by anyone except the administrator. By default, PHP is not even present, let alone disabled.

When thinking about what permissions to allow, it is important to re-iterate the tenet:



Never allow users more permissions than they require to complete their intended tasks!

As they stand, you might not find the input formats to your liking, and so Drupal provides some functionality to modify them. Click on the **configure** link adjacent to the **Filtered HTML** option, and this will bring up the following page:

Filtered HTML [Edit](#) [Configure](#) [Rearrange](#)

Every *filter* performs one particular change on the user input, for example stripping out malicious HTML or making URLs clickable. Choose which filters you want to apply to text in this input format. If you notice some filters are causing conflicts in the output, you can [rearrange them](#).

Name: *

Specify a unique name for this filter format.

Roles

All roles for the default format must be enabled and cannot be changed.

- anonymous user
- authenticated user
- blog administrator user
- forum moderator user

Filters

Choose the filters that will be used in this filter format.

- HTML corrector**
Corrects faulty and chopped off HTML in postings.
- HTML filter**
Allows you to restrict whether users can post HTML and which tags to filter out. It will also remove harmful content such as JavaScript events, JavaScript URLs and CSS styles from those tags that are not removed.
- Line break converter**
Converts line breaks into HTML (i.e.
 and <p> tags).

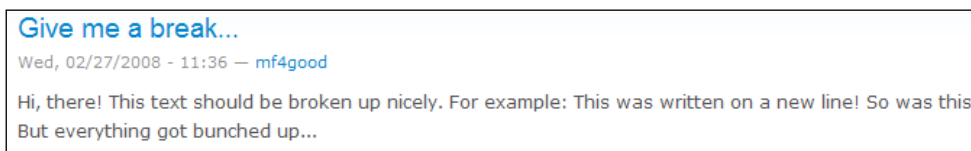
The **view** tab provides the option to alter the **Name** property of the input format; the **Roles** section in this case cannot be changed, but as you will see when we come around to creating our own input format, roles can be assigned however you wish to allow certain users to make use of an input format, or not.

The final section provides a checklist of the types of **Filters** to apply when using this input format. In this previous screenshot, all have been selected, and this causes the input format to apply the:

- **HTML corrector** – corrects any broken HTML within postings to prevent undesirable results in the rest of your page.
- **HTML filter** – determines whether or not to strip or remove unwanted HTML.

- **Line break converter** – Turns standard typed line breaks (i.e. whenever a poster clicks **Enter**) into standard HTML.
- **URL filter** – allows recognized links and email addresses to be clickable without having to write the HTML tags, manually.

The line break converter is particularly useful for users because it means that they do not have to explicitly enter `
` or `<p>` HTML tags in order to display new lines or paragraph breaks – this can get tedious by the time you are writing your 400th blog entry. If this is disabled, then unless the user has the ability to add the relevant HTML tags, the content may end up looking like this:



Click on the **Configure** tab, at the top of the page, in order to begin working with the **HTML filter**. You should be presented with something like this:

Configure *Filtered HTML*
Edit **Configure** Rearrange

If you cannot find the settings for a certain filter, make sure you have enabled it on the [view tab](#) first.

URL filter

Maximum link text length:

URLs longer than this number of characters will be truncated to prevent long strings that break formatting. The link itself will be retained; just the text portion of the link will be truncated.

HTML filter

Filter HTML tags:

Strip disallowed tags
 Escape all tags

How to deal with HTML tags in user-contributed content. If set to "Strip disallowed tags", dangerous tags are removed (see below). If set to "Escape tags", all HTML is escaped and presented as it was typed.

Allowed HTML tags:

If "Strip disallowed tags" is selected, optionally specify tags which should not be stripped. JavaScript event attributes are always stripped.

Display HTML help
If enabled, Drupal will display some basic HTML help in the long filter tips.

Spam link deterrent
If enabled, Drupal will add rel="nofollow" to all links, as a measure to reduce the effectiveness of spam links. Note: this will also prevent valid links from being followed by search engines, therefore it is likely most effective when enabled for anonymous users.

The **URL filter** option is really there to help protect the formatting and layout of your site. It is possible to have quite long URLs these days, and because URLs do not contain spaces, there is nowhere to naturally split them up. As a result, a browser might do some strange things to cater for the long string and whatever it is; this will make your site look odd.

Decide how many characters the longest string should be and enter that number in the space provided. Remember that some content may appear in the sidebars so you can't let it get too long if they are supposed to be a fixed width.

The **HTML filter** section lets you specify whether to **Strip disallowed tags**, or escape them (**Escape all tags** causes any tags that are present in the post to be displayed *as written*). Remember that if all the tags are stripped from the content, you should enable the **Line break converter** so that users can at least paragraph their content properly. Which tags are to be stripped, is decided in the **Allowed HTML tags** section, where a list of all the tags that *are* to be allowed can be entered – anything else gets removed!

Selecting **Display HTML help** forces Drupal to provide HTML help for users posting content – try enabling and disabling this option and browsing to this relative URL in each case to see the difference: `filter/tips`. There is quite a bit of helpful information on HTML in the long filter tips; so take a moment to read over those.



The filter tips can be reached whenever a user expands the **Input format** section of the content post and clicks on **More information about formatting** options at the bottom of that section.

Finally, the **Spam link deterrent** is a useful tool if the site is being used to bombard members with links to unsanctioned (and often unsavory) products. Spammers will use anonymous accounts to post junk (assuming anonymous users are allowed to post content) and enabling this for anonymous posts is an effective way of breaking them.

This is not the end of the story because we also need to be able to create input formats in the event we require something that the default options can't cater for. In this case, let's assume that we want to add some picture files to the *about us* page that will be created in due course. Now, there are several ways in which this can be done, but there are three main criteria that need to be satisfied before we can consider creating the page. We need to be able to:

1. Upload image files and attach them to the post.
2. Insert and display the image files within the body of the post.
3. Use PHP in order to dynamically generate some of the content (this option is really only necessary to demonstrate how to embed PHP in a posting for future reference).

We have already seen how to perform task one when we discussed *Upload* in Chapter XX on *Adding Functionality*. So assuming that you are able to attach files to posts, this leaves us to deal with the second and third criterion. There are several methods for displaying image files within posts. The one we will discuss here, does not require us to download and install any contribution modules such as **Img_assist**. Instead, we will use HTML directly to achieve this, specifically, we use the `` tag.

Take a look at the previous screenshot that shows the **configure** page of the **Filtered HTML** input format. Notice that the `` tag is not available for use. Let's create our own input format to cater for this, instead of modifying this default format.

Before we do, first enable the **PHP Filter** module under **Modules** in **Site building** so that it can easily be used when the time comes. With that change saved, you will find that there is now an extra option to the **Filters** section of each input format configuration page:



Filters

Choose the filters that will be used in this filter format.

- HTML corrector**
Corrects faulty and chopped off HTML in postings.
- HTML filter**
Allows you to restrict whether users can post HTML and which tags to filter out. It will also remove harmful content such as JavaScript events, JavaScript URLs and CSS styles from those tags that are not removed.
- Line break converter**
Converts line breaks into HTML (i.e. `
` and `<p>` tags).
- PHP evaluator**
Executes a piece of PHP code. The usage of this filter should be restricted to administrators only!
- URL filter**
Turns web and e-mail addresses into clickable links.

It's not a good idea to enable the **PHP evaluator** for either of the default options, but adding it to one of our own input formats will be ok to play with. Head on back to the main **input formats** page under **Site configuration** (notice that there is an additional input format available, called **PHP code**) and click on **Add input format**. This will bring up the same configuration type page we looked at earlier. It is easy to implement whatever new settings you want, based on how the input format is to be used.

For our example, we need the ability to post images and make use of PHP scripts so make the new input format as follows:

PHP code [Edit](#) [Configure](#) [Rearrange](#)

Every *filter* performs one particular change on the user input, for example stripping out malicious HTML or making URLs clickable. Choose which filters you want to apply to text in this input format. If you notice some filters are causing conflicts in the output, you can [rearrange them](#).

Name: *

Specify a unique name for this filter format.

Roles

Choose which roles may use this filter format. Note that roles with the "administer filters" permission can always use all the filter formats.

- anonymous user
- authenticated user
- blog administrator user
- forum moderator user

Filters

Choose the filters that will be used in this filter format.

- HTML corrector**
Corrects faulty and chopped off HTML in postings.
- HTML filter**
Allows you to restrict whether users can post HTML and which tags to filter out. It will also remove harmful content such as JavaScript events, JavaScript URLs and CSS styles from those tags that are not removed.
- Line break converter**
Converts line breaks into HTML (i.e.
 and <p> tags).
- PHP evaluator**
Executes a piece of PHP code. The usage of this filter should be restricted to administrators only!
- URL filter**
Turns web and e-mail addresses into clickable links.

As we will need to make use of some PHP code a bit later on, we have enabled the **PHP evaluator** option, as well as prevented the use of this format for anyone but ourselves – normally, you would create a format for a group of users who require the modified posting abilities, but in this case, we are simply demonstrating how to create a new input format; so this is fine for now.



PHP should not be enabled for anyone other than yourself or a highly trusted administrator who needs it to complete his or her work.

Click **Save configuration** to add this new format to the list, and then click on the **Configure** tab to work on the **HTML filter**. The only change required between this input format and the default **Filtered HTML**, in terms of HTML, is the addition of the `` and `<div>` tags separated by a space in the **Allowed HTML tags** list, as follows:

Allowed HTML tags:

As things stand at the moment, you may run into problems with adding PHP code to any content postings. This is because some filters affect the function of others, and to be on the safe side, click on the **Rearrange** tab and set the **PHP evaluator** to execute first:

Name	Weight
+ PHP evaluator	-10
+ HTML filter	-9
+ Line break converter	-8
+ URL filter	-7
+ HTML corrector	-6

Since the **PHP evaluator**'s weight is the lowest, it is treated first, with all the others following suit. It's a safe bet that if you are getting unexpected results when using a certain type of filter, you need to come to this page and change the settings. We'll see a bit more about this, in a moment.

Now, the PHP evaluator gets *dibs* on the content and can properly process any PHP. For the purposes of adding images and PHP to posts (as the primary user), this is all that is needed for now. Once satisfied with the settings, save the changes before using this to create the *about us* page.

Before building the new page, it is probably most useful to have a short discourse on HTML because it is a requirement if you are to attempt more complex postings.

HTML

For a browser to render the neatly laid out and colorful pages it needs instructions on what goes where and what color to give everything. This is the domain of **HyperText Markup Language (HTML)**, and Drupal is no exception in its use of HTML here.

Let's have a quick crash course on the various aspects of HTML before we go any further:

- **Simplicity:** From tables and frames to lists and images, as well as specifying fonts and styles, HTML is a convenient and readily understandable convention for web-page creation and layout.
- **Platform independence:** HTML is platform independent (although not all browsers are exactly the same), which makes sense if you think about it; the last thing you would want, as the builder of a website, is to have to cater for every different type of machine that could make use of HTML.
- **Tags:** HTML comes in the form of opening and closing tags that tell your browser how to display the information enclosed within them. For example, the title of a page would be enclosed within the title tags like this: `<title>My Title Page</title>`. Notice that a forward slash is used to distinguish a closing tag from an opening tag.
- **Attributes:** Tags can have attributes that modify, identify or define certain aspects of a tag's behavior. For example, the `style` attribute in the following HTML snippet defines the color of the paragraph, `<p style="color: blue;">I have a blue font</p>`, when it is rendered in a browser.
- **Sections:** An HTML page is enclosed within `<html></html>` tags and is divided into `<head></head>` and `<body></body>` sections. The body tags enclose the bulk of the page and contain the information seen on the actual web page. In our case, we need not worry about this because all content is automatically posted between the `<body>` tags.

This gives us a fair overview of what HTML is and does, but for practical purposes, it is important to see what can be achieved right here and now, using the HTML that is available to us. Actually, all HTML tags are available for you as the administrator to use, but recall that you should only use this account during development; once the site has gone live, you should post content using an input format that is designed for the task.

The following table discusses each of the default allowed tags along with the `` and `<div>` tags that have just been added. Bear in mind that it is not really practical to show each and every attribute for each tag here, so if you would like to learn more about each tag individually, then please take a look at <http://www.htmlhelp.com>, which is an excellent resource for all things HTML and more:

Tag	Important Attributes	Description
	src: gives the path to the image file. alt: holds a description of the image.	The tag, unlike other tags, does not require a closing tag. It is used to display images within HTML pages, and through the use of optional attributes can accurately control the appearance and layout of images.
<div>	style: used to specify a number of stylistic issues such as background. Remember to make use of CSS and class and id attributes for anything but simple or once off style issues.	The <div> tag is the basic building block of a page's layout. It is used to define divisions or section within a page and can be controlled through the use of attributes.

Tag	Important Attributes	Description
<a>	href: specifies the destination URL of the link. name: allows bookmarks to be created within web pages. target: defines where to open the link – most often this is a new page, <code>_blank</code> , or the same page, <code>_self</code> .	The anchor element facilitates the creation of hyperlinks or bookmarks, which can be navigated by users.
		The emphasis tag converts standard text to italics.
		The strong tag renders text in bold.
<cite>	title: can be used to specify the source or author of the citation in question.	The citation tag allows text to be referenced as coming from another source or author. It is often rendered in italics.
<code>		The code tag changes the style of the enclosed text to mimic computer code's style.
	type: defines the type of bullet point to be used: <code>disc</code> , <code>square</code> , or <code>circle</code> .	The unordered list creates a list of bullet points—it requires the use of the tag to stipulate items in the list.

Tag	Important Attributes	Description
<code></code>		The order list creates a numbered list of bullets – it requires the use of the <code></code> tag to stipulate items in the list.
<code></code>		The list item tag creates a new item within either an ordered or unordered list; because of this it is contained within <code></code> or <code></code> tags.
<code><dl></code>		The definition list tag creates a structured list of items that are defined by the <code><dt></code> and <code><dd></code> tags.
<code><dt></code>		The definition term tag creates a term within a definition list. It is contained within <code><dl></dl></code> tags.
<code><dd></code>		The definition description tag creates a description of its parent term – it is contained within <code><dl></dl></code> tags.

This table really only lists a fraction of all the tags that are available to you to use. Most tags also have a wide variety of required or optional attributes that you can play around with in order to achieve the desired effect.

With that out of the way, we are ready to begin creating a slightly more advanced posting than all the previous ones.

Creating a Feature-Rich Page

One of the cool things about creating a new page like this is that once it is done, it can be reused pretty much anywhere else, substituting in only those values or content that need to change. Obviously, you want the site to look fairly uniform, and this supports the principle of code reuse – at least in terms of HTML.

It is quite likely that at some stage, you will want to create more than just one standalone page. If this is the case, simply cut and paste whatever page is created here and make whatever modification you need, before posting. Doing things in this way will lend all your pages a similar look and feel above and beyond the attributes already given to them by the current theme.

The *about us* page is going to have the following features:

- Well-structured content
- List of objectives
- Inline pictures of the team
- Information about the project
- List of important links
- Some dynamic, PHP-based content
- Advertising

In order to meet the requirements stated, we are going to need to make use of the following tags:

- `<div>`
- ``
- ``
- ``
- `<a>`

along with a few others that we will use to demonstrate the various types of available font styles. In order to create a slightly more complex page like this, consider working with a proper code/HTML editor (a search on Google will turn up many results) that can indent code automatically as well as color code the various tags and content, to make life easier.

OK; we are pretty much ready to begin. I am going to list the entire page's code piece by piece instead of all at once because there are quite a few important things that are worth discussing as we go. However, nothing here is too complicated once you have the hang of HTML and PHP. Before we begin, it is better that we look at the resulting page to get a good idea of what we are working towards. The following screenshot shows the bulk of the page:

1. To provide an online meeting place for like-minded people
2. To discuss and monitor global conservation and wildlife activities
3. To influence policy and effect change in hard-hit areas
4. To support front-line activists like SeaShepherd
5. To raise funds for animal relief efforts and care

Meet the Team

In no specific order, the following people constitute the bulk of the full-time staff here at CWC (You can email them by clicking on the names shown below):

- [Tolis Welch](#)
- [David Mercer](#)
- [Bronagh Casey](#)
- [Nic Malan](#)
- [Brian Reid](#)
- [Rochelle Reid](#)



At CWC we strive to do the *right thing*! Please take the time to look over the site and register in order to start interacting with the community - our natural world needs all the help it can get. If you are interested in getting involved in any one of the number of critically important organizations around the world, then please feel free to browse any of the links given below.

Our Friends

- [Sea Shepherd](#)
- [World Wildlife Fund](#)
- [The Royal SPCA](#)

Our Sponsors!



I hope you'll agree that this page is fairly pleasing to the eye—no comments on the photo please! For very little work, it is quite easy to achieve a look and layout such as this. What isn't apparent from this page, is that the list of names given here, along with their email links, was provided by a short PHP script that was embedded into the HTML page.

Let's get on with the code – to start with, we have the following:

```
<div style="text-align: center;">
  <strong>The CWC</strong>
</div>
<div>
  The <em>Contechst Wildlife Community</em> was started by a group
  of individuals in <cite title="South Africa">Cape Town</cite>. Through
  hard work, dedication and plenty of play time, they have built a truly
  international community that strives to effect change with regards to
  all things related to our biosphere.
</div>
<div>
  <p>
    We have the following goals:
  </p>
</div>
```

This first section is used to declare div regions that are ultimately responsible for laying out all the content. Notice that I have used the **text-align** attribute to make the heading move to the center of the page.

If you look past this snippet of code in the previous listing, you will notice the use of the `<cite>` tag, with a `title` attribute defined. This is here to show you a novel use for providing references. If a user hovers their cursor over the text contained within the `<cite>` tag (in this instance, Cape Town), the text defined in the `title` attribute (in this case, South Africa) will be displayed on screen. In this way, you can clarify or explain important terms without cluttering up your pages.

Continuing along, we get the following ordered list of goals:

```
<ol>
  <li>To provide an online meeting place for like-minded people</li>
  <li>To discuss and monitor global conservation and wildlife
  activities</li>
  <li>To influence policy and effect change in hard-hit areas</li>
  <li>To support front-line activists like SeaShepherd</li>
  <li>To raise funds for animal relief efforts and care</li>
</ol>
```

As you can see, each list item contains exactly one line of content (or one goal, in this case), and all are contained within the `` and `` tags. The next section is where we meet some PHP code as well as insert our image of the team:

```
<div style="text-align: center;">
  <strong>Meet the Team</strong>
```

```
</div>
<div>
    In no specific order, the following people constitute the bulk
of the full-time staff here at CWC (You can email them by clicking on
the names shown below):
</div>
<div style="float: right;">
    
</div>
<div>
    <ul>
<?php
    $team = array('Tolis Welch', 'David Mercer', 'Bronagh Casey', 'Nic
Malan', 'Brian Reid', 'Rochelle Reid');
    foreach($team as $item){
        $name = explode(" ", $item);
        echo '<li><a href="mailto:' . $name[0] . '@cwc.org">' . $item .
'</a></li>';
    }
?>
    </ul>
</div>
```

To summarize, in this section we:

1. Added an inline image with `img` and aligned it to the right of the page.
2. Created an unordered list with the `` tag.
3. Opened up a PHP script by using the `<?php` tag.
4. Created an array of team member names.
5. Used a PHP `foreach` loop, to iterate through each name in the array.
6. Obtained the first name of each member by using the built-in PHP `explode` function.
7. Echoed the results, replete with HTML tags to the screen.

The actual email links were created using the `<a>` tag and the special `mailto` option within the `href` attribute.



If you have the **URL filter** enabled then Drupal would automatically make any email addresses clickable without the need for the `<a>` tag.

As you can see, there are three attributes used here to get the image properly displayed. The first, `src`, gives the path to the image file to be displayed; the second gives a description of the photo so that if, for some reason, the picture is not displayed, then the text **The Team** will be shown instead. Finally, we picked a size for the width of the photo in order to fit it to the page properly. Take note:



The email addresses were built from the first name of the team member so the first two addresses are `Tolis@cbc.org` and `David@cbc.org`. This is slightly contrived as you might not have such an ordered system to your email addresses, but it serves to demonstrate how PHP can be embedded into pages quite easily.

The following section of HTML prints out a list of links to a few other organizations that may be of interest to users:

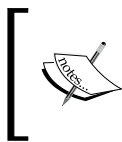
```
<div>
<p>At CWC we strive to do the <em>right thing</em>! Please take the
time to look over the site and register in order to start interacting
with the community - our natural world needs all the help it can
get.</p>
<p>If you are interested in getting involved in any one of the number
of critically important organizations around the world, then please
feel free to browse any of the links given below.</p>
</div>
<div align="center">
<strong>Our Friends</strong>
</div>
<div>
<ul>
<li>
<a href="http://www.seashepherd.org">Sea Shepherd</a>
</li>
<li>
<a href="http://www.worldwildlife.org">World Wildlife Fund</a>
</li>
<li>
<a href="http://www.rspca.org.uk">The Royal SPCA</a>
</li>
</ul>
</div>
```

This part is fairly straightforward, so we move on to the last item on the page – the advertisement:

```
<div style="text-align: center;">
  <strong>Our Sponsors!</strong>
</div>
<div style="text-align: center;">
  <a href="http://www.packtpub.com">
    
  </a>
</div>
</table>
```

This makes use of both a hyperlink and an image file. In effect, we have turned the image, the *Packt* logo, into a hyperlink by enclosing it within `<a>` and `` tags. This means that people can not only view the sponsor's logo, but if they wish, they can also visit the sponsor's site directly by clicking on the image.

With that done, you not only have a nice, shiny new *about us* page, but also a rough template from which to make other pages with a similar look and feel. There is a lot more that goes into giving pages their look and feel, but this involves the use of themes, which we have not yet discussed.



It is generally a bad idea to include absolute file paths in content because if you move your site to another host, or deploy it after development, these links can break – though it's quick and easy for the purposes of this demonstration. Consider using the *Inline* module or something to help in this regard.

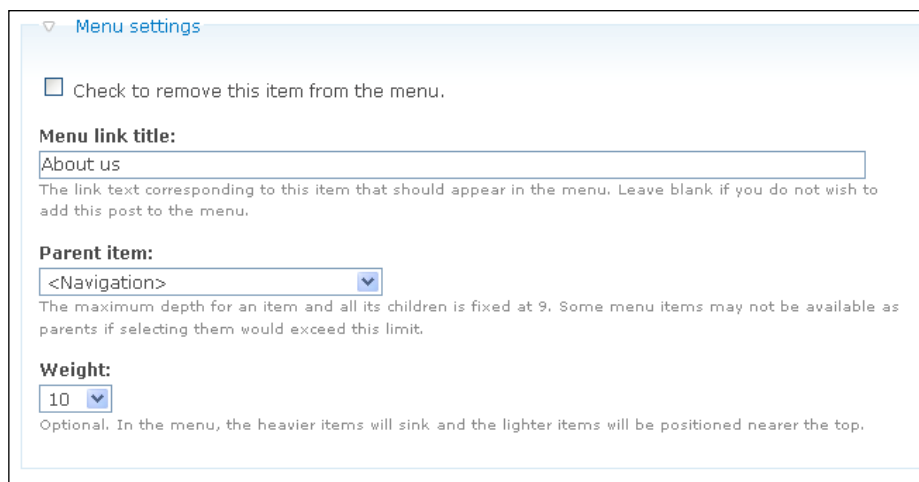
With the page completed, we are not quite finished yet, because it still needs to be added to the site. In order to do this, we need to look at how to actually work with the content we are adding.

Posting a Feature-Rich Page

This brief section will outline the process for getting pages up and on the site. The following list shows the steps required:

1. Create a new post, or edit one that is to be modified. In our case, we have an *About us* page already, so as the administrator we can simply click on the **edit** tab when viewing the **About us** page.
2. Enter or modify the title of the page accordingly.

3. Select the correct input format. In this case, we have a specially created format called **Special pages**.
4. Copy and paste the HTML into the **Body** text area.
5. Ensure the **Authoring information** and **Publishing options** are all correct.
6. For something like the *About us* page, it is probably best to disable comments as you really want this to be a standalone page and not subject to any debate from the rest of the community.
7. Next, ensure that the **Menu settings** are appropriate for the page being added. In this case, we have the following settings in place:



The screenshot shows a 'Menu settings' form with the following fields and options:

- Check to remove this item from the menu.
- Menu link title:**
Text input field containing 'About us'.
The link text corresponding to this item that should appear in the menu. Leave blank if you do not wish to add this post to the menu.
- Parent item:**
Dropdown menu showing '<Navigation>'.
The maximum depth for an item and all its children is fixed at 9. Some menu items may not be available as parents if selecting them would exceed this limit.
- Weight:**
Dropdown menu showing '10'.
Optional. In the menu, the heavier items will sink and the lighter items will be positioned nearer the top.

That's it! Once you are ready to view the page, click on **Preview**, and if everything looks in order, submit it. Remember that it can easily be edited again if anything is wrong.

Obviously, if you are not familiar with either HTML or PHP, you will need to practice a bit with these, but the following links should provide a start:

- <http://www.php.net>
- <http://www.phpbuilder.com>
- <http://www.htmlgoodies.com>
- <http://www.w3schools.com>

Hopefully, you now feel quite confident about incorporating more complex pages into your site.

Summary

With this chapter out of the way, you should have a good understanding of the tasks that lie ahead in creating a fully functional, content-focused website. If you are not already familiar with HTML and PHP, then I recommend you spend some time learning a bit about HTML before continuing on with the next chapter. That said, you have seen how to create input formats to allow different types of HTML or PHP content into posts, as well as looking at how to create a fairly nice HTML-based dynamic web page.

While this is certainly important in terms of creating an aesthetically pleasing site, the real nuts and bolts of your content management lesson came with the discussion on taxonomy. Drupal's taxonomy system sets it apart from other CMS technologies and provides the flexibility and power to implement pretty much any type of structure that we can imagine for our content. With powerful features like tagging available at the click of a button, you are sitting at the controls of one of the best systems around.

CCK is another powerful feature of Drupal that can provide a vast array of new and interesting content types to cater for virtually any type of content. As time goes by, more and more CCK related contributions will become available and you are urged to keep checking the available modules. Unfortunately, at the time of writing, the Views contrib was not available to work in conjunction with CCK but please take the time to check it out when it does arrive.

With much of the hard work out of the way, we can turn our attention to the most creative and, in my opinion, fun part of creating a Drupal site. The following chapter will discuss themes and how to create a unique and appealing look for the new site.