Purpose of categories

Categories (also known as "tags") are a way to organize and find related pages. Categories are implemented by default in Pm Wiki 2, and in most wikis they don't require any special code or markup, they're just a useful convention. The idea is that every page that falls into a particular subject area should have a link to a shared page containing links to other pages on that subject. These pages are created in the *Category* group, and thus these subject areas are called "categories".

Using categories

Getting categories to work requires a single step: adding links to each category. A category named Subject is created by adding a link to Category.Subject on any page. When you add the link to a page, the page can be described as being *in* the category "Subject".

There is a special markup for creating these links which makes categories work more smoothly: [[!Subject]] will create a link to Category.Subject. So [[!Subject]] is a kind of shortcut to the page Subject in the category group.

A Category.GroupFooter file is included in the Pm Wiki release that contains the line `(:pagelist link={*$FullName} list=normal:)` so that whenever a category page is displayed, it will show a list of links to pages that reference that page in the category group. Like any other page in `wikilib.d` you can modify this page and it will not get overwritten by another release.

It is worth noting that rather than using Category.Group Footer, the pagelist directive can be added to Category.Group Header? to similar effect; it just depends on whether you'd prefer to have the list of pages appear before or after any text that you add to the individual category pages (which can be edited just like normal pages).

Because we use the normal PageList `link=` markup, you can use it not only in the category group. If you want to show all pages belonging to the category Subject you can use on any wiki page `(:pagelist link=Category.Subject list=normal:)`.

Similarly, there's no requirement that a "category page" has to be in the *Category* group -- any page can define a "category" of pages that link to it.

An administrator can override the default category group name of "Category" by setting the $CategoryGroup variable in *config.php* to another group name. (Normally a change such as this should be done during initial setup on a new wiki; changing this on a wiki with existing content can cause problems with pagelists unless each page with a category is re-saved.)

A page author can also link to a category list without adding the linking page to the category by using [[{Category.Subject$PageUrl} | Subject]]. This will create a link that looks like [[!Subject]] without adding the linking page to the category listing.

Recap

So, by adding the link [[!Subject]] to a page, a link to that page will automatically appear on the page *Category.Subject*, as long as *Category.Group Footer* has been tweaked appropriately. Thus, you can create a page that automatically creates an alphabetized list of all movies discussed on your wiki by creating links to [[!Movies]] on each film's page; the resulting automatic list would be on the page *Category.Movies* .

authors (advanced)

Category nesting

Categories have the potential for even greater usefulness because `Category.*` pages can themselves be placed into categories! To follow an excellent example from John Rankin, let's suppose we have the following film pages in the categories listed to the right:

```
 Film.ShaunOfTheDead [[!Horror]] [[!Comedy]] [[!2003]] Film.InMyFathersDen
[[!Drama]] [[!2004]] Film.TheCorporation [[!Documentary]] [[!2003]]
```

Now then, we can create `Category.Horror`, `Category.Comedy`, `Category.Drama`, and `Category.Documentary`, and in each one of those pages we put `[[!Genre]]`. In `Category.2003` and `Category.2004`, we put `[[!Year]]`.

So, what happens when we display `Category.Genre` ? We see links to "Comedy", "Drama", "Documentary", and "Horror", because they're in the Genre category. When we click on one of those links, we see all of the films listed in one of those categories. Similarly, if we click on `Category.Year`, we see links to "2003" and "2004", each of which in turn displays the list of films for that year.

Finally, in `Category.Genre` and `Category.Year` we can put `[[!Category]]`, which makes them "top-level" categories reachable from the `Category.Category` page. Voila, we now have an instant "hierarchy":

```
 Category.Category Category.Genre Category.Comedy Film.ShaunOfTheDead
Category.Drama Film.InMyFathersDen Category.Documentary Film.TheCorporation
Category.Horror Film.ShaunOfTheDead Category.Year Category.2003
Film.ShaunOfTheDead Film.TheCorporation Category.2004 Film.InMyFathersDen
```

Note however that this isn't a "strict" hierarchy--i.e., any page or category can appear simultaneously in multiple categories. For example, `Category.Documentary` could be a member of both the Genre and top-level category listings.

Each category page can have content text before the generated list, e.g., to give a generic description of things in the category. (Or it can be empty, which works fine.) It can also contain associations to related categories ("see also" references). For example, in a tourism wiki, the "bed and breakfast" category might contain a see-also reference to the "self-catering" category.

administrators (intermediate)

The guts of the category markup

As mentioned, all of the necessary markup features for Categories are enabled by default in current releases of Pm Wiki 2.0, but here's how they work for those who are interested. The use of the Category group as the

repository for all categories is determined by setting the $CategoryGroup variable, and the special [[!Subject]] markup is activated by a call to the Markup() function:

```
SDV($CategoryGroup,'Category'); Markup('[[!','<links','/\[\[!([^\|\]] ?)\]\]/',
"<span class='category'>[[$CategoryGroup/$1]]</span>");
```

Coming up with good category schemes

The hard part about using categories is choosing a good vocabulary. Site content managers may wish to follow the Guidelines for the establishment and development of monolingual thesauri (ISO 2788-1986) and the Guidelines for the establishment and development of multilingual thesauri (ISO 5964-1985). Questions to think about include:

- whether a scheme already exists and can be reused
- number of levels in a multilevel scheme (not too shallow, not too deep -- e.g. 3)
- number of categories per page (not too many, not too few -- e.g. 3)
- consistent use of singular (`[[Mercury]] is a [[!planet]]`) or plural (`[[Mercury]] is in the [[!planets]] category`)
- disambiguation and use of phrases (`[[!musical instruments]]` and `[[!medical instruments]]`) or Cookbook:Subpage Markup (`[[!Instruments*Musical]]` and `[[!Instruments*Medical]]`)

Or you can just let people use whatever category terms they find meaningful. A vocabulary (or "folksonomy") will emerge over time.

Showing a list of categories

To show a list of categories we can use a pagelist for the pages in the category group.
For instance the following will list pages in the Category group, put it on page Category.Category? for convenience, or on any other page:

```
(:pagelist group=Category list=normal fmt=#title:)
```

But there is a problem: Just adding a category markup to a page will not create a corresponding category page, even though following the link will show the page with a list of pages linking to it!
To have category pages automatically created in group 'Category' add the following to config.php:

```
$AutoCreate['/^Category\./'] = array('ctime' => $Now, 'text' => $page['text']);
```

Change 'Category' to the name of your category group. You can also add more definitions for more category groups, useful if you use a recipe like Cookbook:Tagger which allows multiple category groups.

See also EditVariables#AutoCreate