Default page list templates

PmWiki's default templates for page lists are in Site.PageListTemplates, which is replaced during upgrades. These default templ can be supplemented or overridden with custom templates stored in other locations.

If the page name is not specified as part of the template name, PmWiki's default configuration looks for templates in the following locations in the following order

1. the current page
2. Site.LocalTemplates,
3. Site.PageListTemplates

Administrators can change those locations by using the `$FPLTemplatePageFmt` variable.

If the template is on the current page, the current page must be saved for changes involving the template to show up *(preview al will not work)*.

Custom page list templates

Custom templates are used in the same way as default templates: by referencing the desired format with the `fmt=#anchor` opti There are several ways to indicate which template to use:

- `fmt=#custom` uses the #custom section from the current page, Site.LocalTemplates, or Site.PageListTemplates, (sections denoted by `[[#custom]]` anchors.
- `fmt=MyTemplatePage#custom` uses a custom format from page `MyTemplatePage` from its `#custom` section.
- `fmt=MyTemplatePage` uses a custom format from the entire page `MyTemplatePage`.
- `fmt=custom` uses custom format which is defined in a cookbook script as *custom*.

See Cookbook:PagelistTemplateSamples for examples of custom pagelist formats.

Creating page list templates

A pagelist template contains standard pmwiki markup. When creating pagelist output, pmwiki iterates over each page returned fr the pagelist and will include the pagelist template markup once for every page in the list.

Special references

During the page list iteration pmwiki sets 3 special page references: **=**, **<** and **>**. These special page references are updated on e pagelist iteration and can be used with the page variables syntax, such as *{=$variable}*, to define a pagelist template which will fo the pagelist output. The meaning of the special references are:

= current page	so *{=$Title}*	displays the title of the current page in the iteration
< previous page	so *{<$Group}*	displays the group of the previous page in the iteration
> next page	so *{>$Name}*	displays the name of the next page in the iteration

The > and < references are most useful to help structure pagelist output before and after the actual pagelist. Some common tests used to structure pagelist output are:

`(:template first:)`	~~`(:if equal {<$Group}:)`~~	Iteration is at the beginning of list
`(:template last:)`	~~`(:if equal {>$Group}:)`~~	Iteration is at the end of list

`(:template first {=$Group}:)`	~~`(:if ! equal {=$Group} {<$Group}:)`~~	Iteration is at the first item in a group
`(:template last {=$Group}:)`	~~`(:if ! equal {=$Group} {>$Group}:)`~~	Iteration is at the last item in a group
`(:template defaults:)`		Default options to be used in the pagelist comma
`(:template each:)`		Signifies the repeated part

Note: the markup in column 2 is deprecated.

See also page variable special references.

Page list template special markup

Pagelist templates may have special sections

- `(:template first ...:)` and `(:template ! first ...:)`
- `(:template last ...:)` and `(:template ! last ...:)`

to specify output for the first or last page in the list or a group (use `!first` and `!last` for output *except* for the first/last pag

There's also a

- `(:template defaults ...:)` to allow a template to specify default options,
- `(:template each ...:)` to signify the repeated part, and
- `(:template none:)` whose content will appear if no page was found (from version 2.2.5).

These allow Pagelist templates to be easily separated into "sections" that are included or not included in the output based on a v of conditions. These are intended to be improved versions of the (:if ...:) conditions that have traditionally been used to control pa output (however, the (:if:) conditions still work as before).

First, Each, Last, None

The simplest versions of the directives are:

```
(:template first:)      # markup to display only for first page in list
(:template ! first:)    # markup to display for every page in list but the first
(:template each:)       # markup to display for each page in list
(:template last:)       # markup to display only on last page in list
(:template ! last:)     # markup to display for every page in list but the last
(:template none:)       # markup to display only if no pages were found
```

So, a pagelist template can specify:

```
(:template first:)
Pages in the list:
(:template each:)
* [[{=$FullName}]] [-{{=$FullName}$:Summary}-]
(:template last:)
Displayed {$$PageCount} pages.
```

In addition, the "first" and "last" options can have control break arguments that identify markup to be displayed on the first or last within a particular control section. For example, to specify markup to be displayed upon reaching the first or last page of a group, can use

```
(:template first {=$Group}:)
(:template last {=$Group}:)
```


Thus, instead of writing control breaks using directives, as in

```
(:if ! equal {<$Group} {=$Group}:)
Group: {=$Group}
(:ifend:)
* [[{=$FullName}]]
```

one can now write

```
(:template first {=$Group}:)
Group: {=$Group}
(:template each:)
* [[{=$FullName}]]
```

Page text variables and page variables can also be used, for example

```
(:template default $:Maintainer=- order=$Maintainer,name:)
(:template first {=$:Maintainer}:)
```

Default options

In addition, a template may specify default options to be used in the pagelist command. For example, a pagelist template to displ list of pages by their titles (and sorted by title) might use:

```
[[#bytitle]]
(:template defaults order=title:)
* [[{=$FullName}|+]]
[[#bytitleend]]
```

Then an author could write (:pagelist fmt=#bytitle:) and the pages would automatically be sorted by title without having to specify additional "order=title" option to the (:pagelist:) directive.

To specify multiple parameters to an option enclose the parameters in double quotes, eg to sort by a page text variable and then page name

```
 (:template defaults order="$:Database,name" :)
```

Examples

(:template defaults ... :)

default options for pagelists using this template

(:template each:)

markup for each page in the pagelist

(:template first:)

markup output only for the first page in the pagelist

(:template last:)

markup output only for the last page in the pagelist

(:template first {=$Group}:)

markup output only for a page where the value of {=$Group} has just changed

(:template last {=$Group}:)

markup output only for a page where the value of {=$Group} will change with the next page

So, we have:

```
[[#template]]
(:template defaults order=name:)
(:template first:)
Pages ordered by group
(:template first {=$Group}:)

Pages in group [[{=$Group}/]]
(:template each:)
* [[{=$FullName}]]
(:template last {=$Group}:)
   {=$Group} contains {$$GroupPageCount} pages.
(:template last:)
   {$$PageCount} pages total.
[[#templateend]]
```

Page list template additional page variables

Additional Page Variables that are only available during pagelist are:

```
{$$PageCount}       The current page count of this iteration
{$$GroupCount}      The current group count of this iteration
{$$GroupPageCount}  The current page count within the current group of this iteration
{$$option}          The argument option values from (:pagelist:)
```

Use of {$$option}: For example {$$trail} returns the page name entered in the trail= option of the pagelist directive. You can make up custom "options" with no other purpose than being displayed in the pagelist.

Redirect

To enable searches that return only one page to automatically redirect to that page add the following to a pagelist template where the "jump to a page" functionality is desired:

```
(:template last:)
(:if equal {$$PageCount} 1:)(:redirect {=$FullName}:)(:ifend:)
```

Closure of markup

Any open tables, divs, or other structures inside of (:pagelist:) are, by default, automatically closed at the end of the pagelist command. In other words, (:pagelist:) acts like its own complete page, as opposed to generating markup that is then inserted into the enclosing page.

For example a table generated by the (:cell:) directive in the first (:pagelist:) command is automatically closed at the end of the pagelist. The (:cell:) in the second (:pagelist:) command then starts a new table.

Note that the (:table:) directive doesn't actually start a new table, it's the (:cell:) or (:cellnr:) directive that does it. All that the (:table:) directive does is set attributes for any tables that follow.

Usage

It is advisable to not modify the page Site.PageListTemplates directly so that you will still benefit from upgrades. Instead, modify your Site.LocalTemplates page (which is not part of the PmWiki distribution). Cookbook:PagelistTemplateSamples has many examples of custom pagelist formats.

Other recipes

In addition, the Cookbook has other recipes for special `fmt=` options, including `fmt=dictindex` (alphabetical index) and `fmt=forum` (forum postings).

Copyright: simon - 14/10/2010