Templates

Aptean Ltd

Contents

1 Help:Templates

If you have standard texts you want to include on several pages, the MediaWiki template feature comes into play.

1.1 Creation

Templates are standard wiki pages whose content is designed to be transcluded (embedded) inside other pages. Templates follow a convention that the name is prefixed with "`Template:`", assigning it to that namespace; besides this, you can create them like any other wiki page.

The simplest use of templates is as follows. If you create a page called "Template:Welcome" with contents:

```
Hello! Welcome to the wiki.
```

you'll have created your first template! If you then insert the code:

```
{{Welcome}}
```

in any other page, when that page is viewed the text "Hello! Welcome to the wiki." will appear instead of `{{Welcome}}`. The template content is "transcluded" into the other page, i.e. it is integrated in the page.

You can then insert `{{Welcome}}` at any point of any page where you wish to welcome someone. Suppose it is used in 100 pages. If you then change the template contents to:

```
Hi there! Welcome to this wonderful wiki.
```

and revisit any of the 100 pages where the template was used, you'll see the new text instead of the original one. In this way, you have changed the content of 100 pages without editing them, because the template is transcluded into these pages.

This is the basic mechanism. There are several additional features of transclusion that enrich this mechanism and make templates very useful.

1.2 Usage

Templates can be used in other pages in these ways:

- `{{Name}}`, described above, 'transcludes' (i.e. includes a copy of) the content of the template (stored in the page [[Template:Name]]) whenever the page containing the template transclusion is fetched and displayed; i.e. if the template is later changed, the displayed transcluding page will automatically change too
- `{{subst:Name}}` replaces that string with the contents of the template, in the source of the transcluding page, when you save that page; the copy of the template contents can then be edited normally (and separately from the original in the template page). *Note*: don't use this if you are looking to continually propagate changes from the source template to the page(s) that references it.
- `{{safesubst:Name}}` was introduced in rev:61710 to allow for substitution that doesn't break transclusion, see w:en:Help:Substitution#The safesubst: modifier.
- `{{msgnw:Name}}` includes the template in a form that displays it as raw wiki syntax (the way `<nowiki>` does) when the page containing it is fetched.

In fact, an ordinary wiki page can also be used as a template, simply by specifying the namespace it resides in, so:

- `{{Template:Pagename}}` includes `[[Template:Pagename]]`
- `{{Foo:Pagename}}` includes `[[Foo:Pagename]]`
- `{{:Pagename}}` includes `[[Pagename]]`
    - ♦ `{{subst::Pagename}}` replaces itself with the contents of `[[Pagename]]`

If no such namespace exists, the full title is assumed to be a template:

- `{{Foo:Bar}}` includes `[[Template:Foo:Bar]]`

1.3 Parameters

To enrich the mechanism of transclusion, MediaWiki allows parameters to be passed to a template when it is transcluded. Parameters allow the template to produce different contents or have different behaviors.

Suppose you wish to insert a little thank you note in the talk page of other users, such as:

A little thank you...

for all your effort.
hugs, Me

The thank you note will have a reason (in this case, "all your effort") and a signature ("Me"). Your objective is that any user is able to thank any other user, for any reason whatsoever.

So that the note will look similar everywhere it is used, you can define a template called Template:Thankyou, for example. Although the note should look similar whenever a user thanks another user, its specific contents (i.e. the reason and the signature) will be different. For that reason, you should pass them as parameters. If we ignore the remaining elements to format the box and place the image, the core content of the template will be this:

```
'''A little thank you...'''
for {{{1}}}.
hugs, {{{2}}}
```

Notice the use of `{{{1}}}` and `{{{2}}}`. This is the way to identify, within templates, the parameters that will be passed in when the template is used. Note that, within the template, each parameter is surrounded by **three** braces: `{{{ }}}`. This is different from normal template name usage.

When using the template on a page, you fill in the parameter values, separated by a pipe char (`|`). MediaWiki allows parameters to be passed to the template in three ways.

1.3.1 Anonymous parameters

To pass in anonymous parameters, list the values of those parameters sequentially:

```
{{Thankyou|all your effort|Me}}
```

In this case, template `{{Thankyou}}` receives parameters `{{{1}}}=all your effort` and `{{{2}}}=Me` and produces:

A little thank you...

for all your effort.
hugs, Me

Inverting the order of the parameters:

```
{{Thankyou|Me|all your effort}}
```

causes template `{{Thankyou}}` to receive parameters `{{{1}}}=Me` and `{{{2}}}=all your effort` and inverts the result:

A little thank you...

for Me.
hugs, all your effort

So, the order in which anonymous parameters are passed in is crucial to its behaviour.

1.3.2 Numbered parameters

To pass in parameters by number, identify each parameter when passing it:

```
{{Thankyou|2=Me|1=your friendship}}
```

This time, template `{{Thankyou}}` receives parameters `{{{1}}}=your friendship` and `{{{2}}}=Me`, though they have been supplied in inverse order, and produces:

A little thank you...

for your friendship.
hugs, Me

1.3.3 Named parameters

The third way of passing parameters is by name, instead of numbers. In this case, the template contents would be changed to:

```
'''A little thank you...'''
for {{{reason}}}.
hugs, {{{signature}}}
```

Within the template, we use `{{{reason}}}` and `{{{signature}}}` to identify each parameter, instead of a number. To pass these parameters by name, identify each parameter when passing it:

```
{{Thankyou|signature=Me|reason=being who you are}}
```

In this case, template `{{Thankyou}}` receives parameters `{{{reason}}}=being who you are` and `{{{signature}}}=Me` and produces:

A little thank you...

for being who you are.
hugs, Me

The advantage of using named parameters in your template, besides also being flexible in the order parameters can be passed, is that it makes the template code much easier to understand if there are many parameters.

1.3.4 Default values

If you transclude a template that expects parameters, but do not provide them, in this way:

```
{{Thankyou}}
```

in the numbered parameters example above you would get the following:

A little thank you...

for {{{1}}}.
hugs, {{{2}}}

Since no parameters were passed in, the template presents the parameters themselves, instead of their respective values. In these cases, it may be useful to define *default* values for the parameters, i.e. values that will be used if no value is passed in. For example, if the template contents are changed to:

```
'''A little thank you...'''
for {{{reason|everything}}}.
hugs, {{{signature|Me}}}
```

then `{{{reason|everything}}}` defines that if no parameter `{{{reason}}}` is provided, then the value `everything` will be used. Similarly, `{{{signature|Me}}}`, defaults parameter `{{{signature}}}` to value `Me`. Now, transcluding the template again without passing any parameter, results in the following:

A little thank you...

for everything.
hugs, Me

1.3.5 Control template inclusion

You can control template inclusion by the use of `<noinclude>` and `<includeonly>` tags.

Anything between `<noinclude>` and `</noinclude>` will be processed and displayed only when the template's page is being viewed directly.

Possible applications are:

- Categorising templates
- Interlanguage links to similar templates in other languages
- Explanatory text about how to use the template

The converse is `<includeonly>`. Text between `<includeonly>` and `</includeonly>` will be processed and displayed only when the page is being included. The obvious application is to add all pages containing a given template to a category, without putting the template itself into that category.

Note: when you change the categories applied by a template, the categorization of the pages that use that template may not be updated until some time later: this is handled by the job queue.

1.3.6 Organizing templates

For templates to be effective, users need to find them, and find out how to use them.

To find them, users can:

1. Click **Special Pages > All Pages**
2. In the **Namespace** list, choose **Template** and click **Go**.

To give usage information, include an example like this one on the template page:

```
<noinclude>
== Usage ==
Welcome users:
{{Thankyou|reason=your reason|signature=your signature}}
</noinclude>
```

Then, an editor can simply copy and paste the example to use the template.

1.4 Copying from one wiki to another

Templates often require CSS or other templates, so users frequently have trouble copying templates from one wiki to another. The steps below should work for most templates:

1.4.1 If you have import rights on the new wiki

1. Go to Special:Export on the original wiki. Enter the name of the template in the big text box, check "Include templates" and click Export. This will download a .xml file.
2. Go to Special:Import on the new wiki and upload the .xml file.
3. Look for CSS classes (like class="foobar") in the template text. If those classes appear in "MediaWiki:Common.css" or "MediaWiki:Monobook.css" on the original wiki, copy them to "MediaWiki:Common.css" on the new wiki.
4. If the template uses ParserFunctions, you have to install the ParserFunctions extension.

1.4.2 If you don't have import rights on the new wiki

1. Copy the template text to the new wiki; link to the original page in the edit summary for attribution.
2. Edit the template on the new wiki, and look through the list of templates at the bottom. The ones in red will also need to be copied from the original wiki to the new wiki. You may have to repeat this process multiple times before all dependency templates have been recognized and copied.
3. Look for CSS classes (like class="foobar") in the template text. If those classes appear in "MediaWiki:Common.css" or "MediaWiki:Monobook.css" on the original wiki, copy them to "MediaWiki:Common.css" on the new wiki.
4. If the template uses ParserFunctions, you must install the ParserFunctions extension.

1.5 See also

- Help:External searches - a template special use case example
- Help:Magic words - fancy stuff you may find in some templates
- Help:Parser functions in templates
- MetaWiki:Help:Template - contains a much more thorough manual on how exactly templates function
- MetaWiki:Help:Embed page - embedding pages from namespaces other than `Template:`.