1. Edit/Create Custom Module

This section allows the creation of a new custom module.

Image missing - old was: {img src="img/wiki_up/admin_modules_07.png" align="center"}

You need to indicate the name (used in the select box to assign the module), the title and the data that will be displayed, the data must be HTML (any since you are admin). The right part of this panel can be used to help you include some objects that can be used in user modules, you can include in an user module:

Use 'Create New User Module' to create custom modules

Image missing - old was: {img src=http://doc.tiki.org/img/wiki_up/admin_modules_07.png align="center"}

Once a user module is created, it can be selected and assigned from the Module Name list, just like the standard modules available to the site.
When assigning a module you can set optional parameters to it.

```
{if $language eq 'fr'} {menu id=44} {elseif $language eq 'es'} {menu id=45} {else} {menu id=43} {/if}
```

The advantage of such code is to allow a default menu (id=43) when a translated version does not exist.

Inside the module, you can also display a menu depending on the user's group using the same syntax with the *$group* variable (which seems empty for Anonymous)

```
category[0]=9&category[1]=78&category[2]=79
```

or

```
nocategory[0]=9&nocategory[1]=78&nocategory[2]=79
```

2. Including content from feature XYZ in Custom Modules

2.1. A poll

(it will display the form to vote for the indicated poll)

+

```
{poll id=n}
```

2.2. Any poll from the active polls

```
{poll}
```

2.3. A random image from any gallery

```
{gallery id=-1 showgalleryname=1}
```

2.4. A random image from a single gallery

```
{gallery id=n showgalleryname=0}
```

2.5. The current value of a dynamic content block

```
{content id=n}
```

2.6. A random value from a dynamic content block

```
{rcontent id=n}
```

2.7. An RSS feed

```
{rss id=n}
```

2.8. A banner

```
{banner zone=foo}
```

2.9. A menu

"normal menu" (Until Tiki 12.x)

just a normal menu

```
{menu id=n}
```

normal menu with auto close non-active sections

```
{menu id=n menu_cookie=n}
```

A normal menu with an automatic opening of the sections the current url is in and the sub-section of this url. The other sections will be closed automatically. There is no consideration of the cookies

normal menu with a custom class id

```
{menu id=n css_id=custom_name}
```

CSS/Suckerfish style

vertical

```
{menu id=n css=y bootstrap=n type=vert}
```

horizontal

```
{menu id=n css=y bootstrap=n type=horiz}
```

Bootstrap style (From Tiki 14.x +)

vertical

```
{menu id=n bootstrap=y type=vert}
```

horizontal

```
{menu id=n bootstrap=y type=horiz}
```

2.10. A structure "menu from table of contents of structure of pages"

old way:

```
{wikistructure id=name}
```

Image missing - old was: {img src="img/wiki_up/admin_modules_08b.png" align="center"}

Note that there is a last checkbox to allow parsing the content of the menu, so that, wiki markup can be recognized and converted to html when showing the module in the browser (HomePage will be a clickable link then, in the previous example).

new way:

```
{menu structureId=name type=vert bootstrap=n css=y}
```

2.11. The current date

The following syntax can be used to display the current server date on a module:

```
{showdate mode="d/m/Y h:i"}
```

Where mode is any valid format for the PHP "date" function (see the date function on the PHP manual).

2.12. Dynamic Content

You can include dynamic blocks in a user module using a simple syntax:

```
Today quote: {content id=3}
```

In the example we are including the current version of the dynamic block 3 . You can create/edit and program dynamic blocks using the Dynamic Content System (DCS). Read about the DCS in the DCS section to learn more.

2.13. Banners

If you want you can include a banner in a user module, using the following syntax:

```
{banner zone=sidebar}
```

Where zone must be the name of an existing zone in the banners system and there should be at least one banner assigned to the zone (or nothing will be displayed). You can learn more about banners in the banners section.

2.14. Polls

You can display the form to vote a poll in a user module using the syntax

```
{poll id=n}
```

If you want Tiki to display the form to vote some (random) poll from all the active polls you can use:

```
{poll}
```

2.15. A random image

You can use the following function to display one random image from an image gallery:

```
{gallery id=n}
```

2.16. RSS feeds

You can display the content of an RSS feed from another site in a user module after you created the feed

(see using external RSS feeds later on this section), the syntax is:

```
{rss id=n max=m}
```

It will display the rss feed with id=n and it will display up to m elements from the feed.

3. Develop your own Custom Module

A module is made up of one or two files:

- A Smarty template file in the **templates/modules** directory - extension .tpl
- A php file in the **modules** directory - extension .php. The php fle is not mandatory

The best way to begin, is to copy the one or 2 files from another module, to change the name of the files, the name of the php functions.
You can find an easy example in modules/mod-directory_stats.php , templates/modules/mod-directory_stats.tpl.

4. Little Tutorial in 3 Steps from a forum post

(Torsten once more answered comprehensive instead of updating the docs and linking, so he copy-pasted it here ;-))

Question was

> I want to create a custom footer very very much like the one at the bottom of this page, formatted the same way etc. I don't seem to be able to see a footer module.
>
> Can anyone shed any light on this?
>
> Thanks

Answer

Torsten Fabricius wrote:

Hello HadleysHope,

you could create a so called "custom module" and apply this to the footer module zone.

Please note: For step three below, I assume you have a Tiki 14 or 15 or trunk. For Tiki 12 it would be slightly different, as Tiki 12 not yet knows about the Bootstrap magic.

I assume, that you are not familiar with custom modules, so a few pointers:

1) What is a module

The Tiki integrated "legacy modules" (not an official name!), so the modules that are integrated from installation (in the Tiki code base) are literally simply containers with predefined content based on templates.

Templates contain HTML and smarty code and optionally can contain WikiSyntax aswell.

Custom modules are empty containers where you can write arbitrary HTML, smarty and WikiSyntax and then apply them to the module zones in the exact same way as you would do with the legacy modues. Only one difference: with custom modules you do not have predefined fields for module specific parameters, but only one textbox, where you have to write the parameters and values manually.

2) Steps for a simple custom module in the footer like on this page/site

First you create a custom module in /tiki_adminmodules.php in the Tab "Custom Modules" and name it this way:

Name: test_footermenu
Title: test_footermenu

I always name them in lower case to distinguish them in the list from the legacy modules - but you are free to name and label (title) them differently and with upper case aswell.

The content of a module must not be empty, so we add some temporary code and to make it visible when applied:

```
<div style="background: red; height: 300px; width: 100%; font-weight: bold; color: black;">Hello, I am your new custom module. <br /> Test and enjoy!</div>
```

Save

Second you go to the Tab "Assigned Modues" and click on the button "Add Module". You will find the newly created module in the drop down list which is offered to you to choose one.
Choose your new module and click on "Module Options".

Now you need to provide the necessary parameters in the parameters textbox.

Let's assume you want to not show the box and backgound and headng etc. of the module but only the plain content of the module, so we need nobox=y and we want to display the module only on the HomePage so we need page=HomePage

The syntax for the parameters and values would be following:

```
nobox=y&page=HomePage
```

Make sure to choose the module zone "bottom" (for the site footer) in the dropdown - default is "top". If you forget this, your module would be displayed in the header.

Assign

And voilà, when you open the HomePage in another Browser Tab, you will see a red 300px high rectangle in the footer with the text:

Hello, I am your new custom module.
Test and enjoy!

Three now:

Go back to the Tab "Custom Modules" and choose to edit your new custom module (you can keep it assigned, as the content will update. For testing on a production site you should create a wiki page "Test"

and limit visibility to it (see above) with the parameter page=Test instead of page=HomePage.

You want 4 menus (created before, assuming they have the Id Numbers 44, 45, 46 and 73) besides each other and they shall flip below each other on a narrow screen. We do that with Bootstrap classes in the divs and we **avoid** inline styling!
The menus themselves you include with a little smarty magic.
Here we use non-bootstrap "classic" css menus.

Bootstrap class col-md-3:
col = column
md = middle viewport brakepoint = tablet landscape
3 = width = portion of 12th (1 = 1/12; 2 = 2/12; 3 = 3/12; ... 12 = 12/12)

```
<div class="row"> <!-- always wrap columns in a row - each row has "12 width" ! --> <div class="col-md-3"> <!-- 4 columns of "3 width" = 1 row of "12 width" --> {menu id=44 type=vert bootstrap=n css=y} <!-- this is the smarty magic --> </div> <!-- close the column --> <div class="col-md-3"> <!-- open the next column --> {menu id=45 type=vert bootstrap=n css=y} <!-- next menu --> </div> <!-- close the column --> <div class="col-md-3"> <!-- open the next column --> {menu id=46 type=vert bootstrap=n css=y} <!-- next menu --> </div> <!-- close the column --> <div class="col-md-3"> <!-- open the next column --> {menu id=73 type=vert bootstrap=n css=y} <!-- next menu --> </div> <!-- close the column --> </div> <!-- close the row -->
```

Dunno ! Save and check the footer on the page Test.

In case the menus would not be in the same line (stepped display) you can try the following (usually necessary with the similar setup with {DIV()}...{DIV} plugins on wiki pages, I do not off-hand know if ths is necessary with plain HTML in custom modules aswell (no time to test myself):

```
<div class="row"> <!-- always wrap columns in a row - each row has "12 width" ! --> <div class="col-md-3"> <!-- 4 columns of "3 width" = 1 row of "12 width" --> {menu id=44 type=vert bootstrap=n css=y} <!-- this is the smarty magic --> </div><div class="col-md-3"> <!-- close and open in the same line --> {menu id=45 type=vert bootstrap=n css=y} <!-- next menu --> </div><div class="col-md-3"> <!-- close and open in the same line --> {menu id=46 type=vert bootstrap=n css=y} <!-- next menu --> </div><div class="col-md-3"> <!-- close and open in the same line --> {menu id=73 type=vert bootstrap=n css=y} <!-- next menu --> </div> <!-- close the column --> </div> <!-- close the row -->
```

Alias

Creating User Modules