zikula

CheatSheet 2.0

Locations

template search order:
1. /config/templates/YourModule
2. /themes/YourTheme/templates/modules/YourModule
3. /modules/YourModule/pntemplates

theme specific module customizations are stored here:
/themes/YT/templates/modules/YourModule/
for modified module templates
/themes/YT/style/YourModule/style.css
for modified/extra stylesheets
/themes/YT/templates/modules/YourModule/plugins/
for modified/extra plugins
/themes/YT/templates/modules/YourModule/images/
for modified/extra images
→ same works independent of the theme in the config folder

Calls

```
<!--[function]-->
<!--[function param1="val" param2=$var]-->
<!--[$variable]-->
<!--[$variable|modifier]-->
<!--[$variable|modifier:"value"]-->
<!--[$variable|phpFunction:"value"]-->
```

Assign

```
<!--[assign var="name" value="example" ]-->
```

Backticks

Simple Math:
```
<!--[assign var="name" value=`$var1+$var2`]-->
```
Combining Strings:
```
<!--[assign var="name"⁋
 value="`$var1` something `$var2`"]-->
```

Conditions

```
<!--[if $name == 'Fred']-->
   Welcome Sir.
<!--[elseif $name == 'Wilma']-->
   Welcome Ma'am
<!--[else]-->
   Welcome, whatever you are
<!--[/if]-->
```

Qualifiers:
$a == $b – equals
$a != $b – not equals
$a > $b – greater than
$a < $b – less than
$a >= $b – greater than or equal
$a <= $b – less than or equal
$a === 0 – check for identity
! $a – negation
$a mod $b – modulous
$a is div by 4 – divisible by
$a is even – an even number
$a is even by $b – grouping level even
$a is odd – an odd number
$a is odd by $b – an odd grouping

Loops

```
<!--[section name=id loop=$variable]-->
        <!--[$variable[id]]-->
<!--[sectionelse]-->
        $variable empty!
<!--[/section]-->
```
Or:
```
<!--[foreach item="currentItem" from=$items]-->
        <!--[$currrentItem]-->
<!--[foreachelse ]-->
        $item empty!
<!--[/foreach]-->
```

Math

```
<!--[math equation="x*y" x=$height y=$width]-->
```
or
```
<!--[math equation="x*y" x=$height y=$width⁋
assign="var"]-->
<!--[ $var ]-->
```
or with parenthesis
```
<!--[math equation="((x+y)/z)" x=2 y=10 z=2]-->
```

Modifiers

some widely used modifiers:
default – set a default value for a var
nl2paragraphs – newlines (\n) turned into paragraphs
regex_replace – regular expression search and replace
replace – simple search and replace
string_format – format strings like sprintf() in php
strip_tags – strips out all HTML markup
truncate – truncates a var to a set length
cat – value is concatenated to the given var
wordwrap – wraps a string to a column width
All php-functions can be used as modifiers implicitly.
Use @phpfunc with arrays
Some more specific modifiers:
activatelinks – turns URLs within a string into links
pnml – converts lanugage define into language string
pnmodcallhooks – call transform hooks
userprofilelink – create a link to a users profile
userprofilelink:"css" – adds class="css"
userprofilelink:'':"img/profile.gif" –
Using profile.gif instead of username, no class
yesno – returns Yes if var = 1 and No if var = 0
onlineoffline – returns Online if var = 1 and Offline if var = 0
activeinactive – returns Active if var = 1 and Inactive if var = 0

security modifiers:
pnvarprepfordisplay – removes HTML completely
pnvarprephtmldisplay – preserves some HTML tags
both prevent user input from destroying the layout

pndate_format

pndate_format:"dateSpecifiers"
formats a date and time into the given strftime() format:
%a – abbr. weekday name
%A – full weekday name
%b – abbr. month name
%B – full month name
%d – day of month (range 01 to 31)
%D – same as %m/%d/%y
%e – day of month (range 1 to 31)
%H – hour (range 00 to 23)
%I – hour (range 01 to 12)
%j – day of year (range 001 to 366)
%k – Hour (range 0 to 23)
%l – hour (range 1 to 12)
%m – month (range 01 to 12)
%M – minute as a decimal number
%n – newline character
%p – either "am" or "pm"
%r – time in am. and pm notation
%R – time in 24 hour notation
%S – second as a decimal number
%t – tab character
%T – equal to %H:%M:%S
%u – weekday as a decimal number
%V – ISO 8601:1988 week number
%x – preferred date for current locale
%X – preferred time for current locale
%y – year (range 00 to 99)
%Y – year including century
%Z – time zone or name or abbr.
%% – a literal "%" character
pndate_format:"":"dateString"
uses Zikula language defines for standard dates.
English example:
datebrief – %b %d, %Y
datelong – %A, %B %d, %Y
datestring – %A, %B %d @ %H:%M:%S
datetimebrief – %b %d, %Y - %I:%M %p
datetimelong – %A, %B %d, %Y - %I:%M %p
dateinput – %Y-%m-%d
datetimeinput – %Y-%m-%d %H:%M
timebrief – %I:%M %p

Modifer Examples

<!--[$title|default:'no title']--> – set default value incase variable is empty
<!--[$title|truncate:40:'&hellip;']--> – truncate var to 40 char, ending with ...
<!--["%2.f"|sprintf:$var]--> – use PHP function on var. (1st PHP param is the one in front of the |)
<!--[$accountlinks|@sort]--> – use PHP function „sort“ on all array elements

Functions

<!--[pnvarcleanfrominput name="param"⁋
assign="var"]--> – fetch the value of an URL parameter

<!--[pnusergetvar name="user_icq" uid=1]--> – fetch user data (no uid = present user)

<!--[pnusergetidfromname name=$name⁋
assign="uid"]--> – return the user ID for a given username

<!--[pnimg src="image.png"]--> – returns an img-tag
src – file name of the image
modname – module name (default – the current module)
width, height – If none is set, they are obtained from the image
alt – If not set, an empty string is being assigned
altml – If true, alt string is assumed to be a ML constant
title – If not set, an empty string is being assigned
titleml – If true, title string is assumed to be a ML constant
optional – If set, the plugin will return no error if image isn't found
default – If set, a default image is used should the requested image not be found (Note – full path required)
set – If modname is 'core' then the set parameter is set to define the directory in /images/
nostoponerror – If set and error ocurs, do not trigger_error, but return false and fill pnimg_error instead
assign – If set, the results are assigned to a variable
All remaining parameters are passed to the module function

<!--[pnml name="_EXAMPLESTRING"]--> – read a language constant
Available parameters:
name – Name of the language constant to return
html – Treat the language define as HTML
noprocess – If set, no processing is applied to the constant value
assign – If set, the results are assigned to a variable
All remaining parameters are passed to the module function
<!--[pnmodurl modname="Name" type="user"⁋
func="display"]--> – create a URL for a specific module function
Available parameters:
modname – modulename (required)
type – currently one of 'user' or 'admin' (default is 'user')
func – module function (default is 'main')
fragment – The fragment to target within the URL
ssl – set to constant null, true, false
append – a string to be appended to the URL
assign – If set, the results are assigned to a variable
All remaining parameters are passed to the module function

Module Functions

<!--[pnmodfunc modname="Name" type="user"⁋
func="main"]-->
execute a module function (pnuser.php, pnadmin.php ...)
Available parameters:
modname – modulename (required)
type – currently one of 'user' or 'admin' (default is 'user')
func – module function (default is 'main')
assign – If set, the results are assigned to a variable
All remaining parameters are passed to the module function

<!--[pnmodapifunc modname="Name" type="user"⁋
func="main"]-->
execute a module API function (pnuserapi.php, pnadminapi.php ...)
Available parameters:
modname – modulename (required)
type – currently one of 'user' or 'admin' (default is 'user')
func – module function (default is 'main')
assign – If set, the results are assigned to a variable
All remaining parameters are passed to the module function

zikula CheatSheet 2.0

Further Module Plugins

`pnmodavailable` – is module X installed and active?
`pnmodgetinfo` – get e.g. displayname
`pnmodgetname` – get name of the current module
`pnmodgetvar` – get a module var
`pnmodishooked` – is module A hooked into module B?

User Functions

User related:
`pnusergetidfromname` – get Username by UID
`pnusergetlang` – which lang does user use?
`pnusergettheme` – which theme does user use?
`pnusergetvar` – get profile details by UID
`pnuserloggedin` – is visitor logged in?

Blocks

Block functions enclose a template block and operate on the contents of this block:

```
<!--[securityutil_checkpermission_block⁋
component="News::" instance="::"⁋
level="ACCESS_COMMENT"]-->
            do some stuff now we have permission
<!--[/securityutil_checkpermission_block]-->
```

Permissions don't have to exist in a module. You can just make up some and use them in a template and the Permissions module.

```
<!--[literal]-->
            content will not be interpreted by the
templating engine
<!--[/literal]-->
```

This block is often used if you need conditional comment to define special style sheets for Internet Explorer.

```
<!--[php]-->
            echo("I am a bad programmer");
<!--[/php]-->
```

The php block can be used to include normal php code into a template. Anyway: That is really bad bad bad style. Put PHP into plugins, dumbass!

Includes

`<!--[include file="file.htm"]-->`
Includes template.htm from the same directory into the current template.

`<!--[include file="file.htm" var1="NEW"]-->`
Includes the file and passes a variable to it. Use normal call to display the variable:
`<!--[$var1]-->`

`<!--[include file="file.htm" var1="$array]-->`
Even arrays can be passed. Use normal call to display the array:

```
<!--[foreach from=$links item=l]-->
. do stuff  ...
<!--[/foreach]-->
```

`<!--[include file='/usr/local/include/templates/header.tpl']-->`
absolute filepath
`<!--[include⁋ file='file:/usr/local/include/templates/header⁋ .tpl']-->`
absolute filepath (same thing)
`<!--[include⁋ file='file:C:/www/pub/templates/header.tpl']-->`
windows absolute filepath (MUST use "file:" prefix)
`<!--[include file='db:header.tpl']-->`
include from template resource named "db"
`<!--[include file="$module.tpl"]-->`
include a $variable template - eg $module = 'contacts'
`<!--[include file='$module.tpl']-->`
won't work as its single quotes ie no variable substitution
`<!--[include file="$path/$module.$view.tpl"]-->`
include a multi $variable template - eg amber/links.view.tpl

pndebug

The most important function in Zikula for designers:
`<!--[pndebug]-->` – Popup with all available variables within a template

Inserts

`<!--[insert name="getstatusmsg"]-->`
obtains the last status message posted for this session.
Available parameters:
`style, class` – msg put in a div tag with these attributes
`tag` – specifies a span or a div tag
`assign` – assign to var

`<!--[insert name=setpagevar var="title" value="mytitle"]-->`
sets a page-specific variable.
available vars:
`title` – set html title (sitename and slogan are added)
`description` – set meta description
`keywords` – set meta keywords
`stylesheet` – include a stylesheet in the header
`javascript` – include a javascript in the header
`body` – add attributes to the opening body tag
`rawtext` – add rawtext in the header
`footer` – add rawtext above the closing body tag

Examples:
`<!--[insert name=setpagevar var="stylesheet" value="/themes/MyTheme/style/extrastyle.css"]-->`
inserts an extrastyle.css into the HTML header. Valid HTML or xHTML according to the settings in theme.php

`<!--[insert name=setpagevar var="javascript" value="/javascript/ajax/prototype.js"]-->`
inserts prototype.js into the HTML header. Zikula eliminates redundant calls of the same script.

`<!--[insert name=setpagevar var="description" value=$text|truncate:'250']-->`
inserts the first 250 characters of the variable text into the meta-description

`<!--[insert name=setpagevar var="body" value='onload="onLoad()"']-->`
add the onload attribute to the body tag

Plugins

Naming plugins:
`function.NAME.php` – for functions
`modifier.NAME.php` – for modifiers
`block.NAME.php` – for blocks

Naming the function:
`function smarty_function_NAME($params, &$smarty)`
for functions
`function smarty_modifier_NAME($params, &$smarty)`
for modifiers
`function smarty_block_NAME($params, &$smarty)`
for blocks

`NAME` must be the same in file-name and function-name (case-sensitive!)

Should always contain parameters:
`assign` – to be able to assign return value to var

Romp Plugin

```
<?php
function smarty_TYPE_NAME($params, &$smarty) {

    //your code here

    if (isset($params['assign'])) {
          $smarty->assign($params['assign'],
$value);
    } else {
          return $value;
    }
}
```

Theme Functions

`<!--[title]-->`
usually the title of a page, consisting of content title, sitename and site slogan.
available parameters:
`separator` – title elements will be seperated using this string (optional: default '`::`')
`noslogan` – slogan will not be appended if `true`
`nositename` – sitename will not be appended if `true`

pager (numeric)

`<!--[pager rowcount="400" limit="50"]-->`
adds a numeric pager to a template
available parameters:
`rowcount` – total number of items to page in between (if an array is assigned, it's count will be used)
`limit` – number of items on a page (if <0 unlimited)
`posvar` – name of the variable that contains the position data, eg "offset"
`display` – either '`page`' or '`startnum`'. `page` = 1, 2, 3, 4, ..) `startnum` = (1, 11, 21, 31, 41, ..)
`anchorText` – optional text for hyperlink anchor (e.g. 'comments' for the anchor #comments) (default: '')
`maxpages` – optional maximum number of displayed pages, others will be hidden / suppressed (default: 0 = show all pages)
`class` – optional class to apply to the pager container (default : pn-pager)
`processDetailLinks` – should the single page links be processed? (default: false if using pagerimage.html, otherwise true)
`template` – optional name of a template file
`includeStylesheet` – use predefined stylesheet file? Default is yes.

already available templates:
`pagercss2.html`
`pagercss.html`
`pager.html`
`pagerimage.html`
`pagerintervals.html`
`pageritems.html`
`pagerjs.html`
Of course you can add you custom templates

pager (alphabetic)

`<!--[pagerabc posvar="letter" ]-->`
display an alphabethical pager
Available parameters
`names` – values to select from (array or csv) e.g. `"A;B;C;D"` and so on
`separator` – string to put between the letters, eg `"|"` makes | A | B | C | D |
`posvar` – name of the variable that contains the position data, eg `"letter"`
`forwardvars` – comma- semicolon- or space-delimited list of POST and GET variables to forward in the pager links. If unset, all vars are forwarded.
`additionalvars` – comma- semicolon- or space-delimited list of additional variable and value pairs to forward in the links. eg "foo=2,bar=4"
`class_num` – class for the pager links (<a> tags)
`class_numon` – class for the active page

`printempty` – print empty sel ('-')

CSS Classes

Common Classes
`.pn-hide` – equals display:none
`.pn-clearfix` – use to avoid evil `clear:both`!

Class for Tables
`.pn-datatable` – makes all tables look alike
use
`<tr class="<!--[cycle values="odd,even"]-->">` to give table rows changing background colors
use
`<th class="pn-sortable>` to make colums sortable via javascript (prototype!)

Classes for Forms
`.pn-form` – use to make forms look like Zikula form
`.pn-formrow` – use to style a div containing a label and control pair

Licence