Template Toolkit Quick Reference Card

Revision 0.5 for Template Toolkit version 2.14
Andrew Ford

refcards.com™

The Template Toolkit (www.template-toolkit.org) is a sophisticated template system written by Andy Wardley.

Syntax

Directives

```
[% [GET] var %]
[% CALL var %]
[% [SET] var = value ... %]
[% DEFAULT var = value ... %]
[% META attr = value ... %]

[% INSERT filename %]
[% INCLUDE template [var = value ...] %]
[% PROCESS template [var = value ...] %]
[% WRAPPER template [var = value ...] %] text... [%
END %]
[% BLOCK [name] %] content...   [% END %]
[% FILTER filter %] text...   [% END %]
[% MACRO name[(varlist)] directive %]
[% USE plugin[(param, ...)] %]
[% PERL %] code...   [% END %]
[% RAWPERL %] code...   [% END %]

[% FOREACH var = list %] ...   [% END %]
[% WHILE cond %] ...   [% END %]
[% IF cond %] ...   [% ELSIF cond %] ...
    [% ELSE %] [% END %]
[% SWITCH var %] ...   [% CASE [{value|DEFAULT}] %]
    ...   [% END %]
[% TRY %] ...   [% CATCH [type] %] ...
    [% FINAL %] ...   [% END %]
[% THROW type info ... %]
[% NEXT %]
[% LAST %]
[% RETURN %]
[% STOP %]
```

Special variables

template	outermost template being processed
	methods: name, modtime
component	innermost template being processed
	methods: name, modtime
loop	loop iterator
	methods: count, first, last, max
error	exception object
content	captured output for WRAPPER
global	top level namespace

Virtual methods

Scalar variables

chunk(size)	negative size chunks from end
defined	is value defined?
hash	treat as single-element hash with key value
length	length of string representation
list	treat as single-item list
match(re)	true if value matches re
repeat(n)	repeated n times
replace(re, sub)	replace instances of re with sub
search(re)	returns list of matching subpatterns
size	returns 1, as if a single-item list
split(re)	split string on re

Hash variables

each	list of alternating keys/values
exists(key)	does key exist?
import(hash2)	import contents of hash2
import	import into current namespace hash
keys	list of keys
list	returns alternating key, value
nsort	keys sorted numerically
size	number of pairs
sort	keys sorted alphabetically
values	list of values

List variables

first	first item in list
grep(re)	items matching re
join(str)	items joined with str
last	last item in list
max	maximum index number (i.e. size - 1)
merge(list [, list...])	combine lists
nsort	items sorted numerically
pop	remove first item from list
push(item)	add item to end of list
reverse	items in reverse order
shift	remove last item from list
size	number of elements
slice(from, to)	subset of list
sort	items sorted lexically
splice(off, len [,list])	modifies list
unique	unique items (retains order)
unshift(item)	add item to start of list

Standard filters

collapse	collapses whitespace to a single space
eval(text)	evaluate as template text
evaltt(text)	evaluate as template text
evalperl(text)	evaluate text as Perl code
format(str)	format as per printf()
html	performs HTML escaping on '<', '>', '&'
html_break	convert empty lines to HTML linebreaks
html_entity	performs HTML escaping
html_line_break	convert newlines to ' '
html_para	convert blank lines to HTML paras
indent(pad)	indent by pad string or width
latex(outfmt)	process through LaTeX
lcfirst	lower case first character
lower	convert to lower case
null	output to the bit bucket
perl(text)	evaluate text as Perl code
redirect(file)	redirect output to file
remove(re)	removes occurrences of re
repeat(n)	repeat n times
replace(re, sub)	replace re with sub
stderr	redirect output to STDERR
stdout(binmode)	redirect output to STDERR in mode binmode
trim	removes leading and trailing whitespace
truncate(len)	truncate to length len
ucfirst	capitalize first character
upper	convert to upper case
uri	performs URI-escaping

Standard plugins

Refer to documentation for details of individual plugins.

Autoformat	autoformatting with Text::Autoformat
CGI	interface to CGI.pm
Datafile	data stored in plain text files
Date	generates formatted time and date strings
Directory	interface to directory contents
DBI	interface to DBI
Dumper	interface to Data::Dumper
File	provides general file abstraction
Format	provides printf-like formatting
GD::*	provide access to GD graphics library
HTML	generic HTML generation
Iterator	iterator creation
Pod	interface to Pod::POM (POD Object Model)
String	OO string manipulation interface
Table	table formatting
Url	URL construction
Wrap	simple paragraph wrapping
XML.DOM	interface to XML Document Object Model
XML.RSS	interface to XML::RSS
XML.Simple	interface to XML::Simple
XML.Style	simple stylesheet transforms of XML
XML.XPath	interface to XML::XPath

Configuration Options

START_TAG	start of directive token	([%)
END_TAG	end of directive token	(%])
TAG_STYLE	set pre-defined START_TAG/END_TAG style	
PRE_CHOMP	remove whitespace before directives	(0)
POST_CHOMP	remove whitespace after directives	(0)
TRIM	remove leading and trailing whitespace	(0)
INTERPOLATE	interpolate embedded variables	(0)
ANYCASE	allow lower case directive keywords	(0)

Template files and blocks

INCLUDE_PATH	search path for templates	
DELIMITER	delimiter for separating paths	(:)
ABSOLUTE	allow absolute file names	(0)
RELATIVE	allow relative filenames	(0)
DEFAULT	default template	
BLOCKS	hash array pre-defining template blocks	
AUTO_RESET	reset BLOCK definitions each time	(1)
RECURSION	permit recursion in templates	(0)

Template variables

PRE_DEFINE	hash array of variables and values to pre-define
VARIABLES	synonym for PRE_DEFINE

Runtime processing options

EVAL_PERL	process PERL/RAWPERL blocks	(0)
PRE_PROCESS	template(s) to process before main template	
POST_PROCESS	template(s) to process after main template	
PROCESS	template(s) to process instead of main template	
ERROR	name of error template or reference to hash array mapping error types to templates	
OUTPUT	default output location or handler	
OUTPUT_PATH	directory into which output files can be written	
DEBUG	raise 'undef' error on access to undefined variables	

Caching and Compiling Options

CACHE_SIZE	max compiled templates to cache (undef, i.e. cache all)
COMPILE_EXT	extension for compiled template files (undef)
COMPILE_DIR	directory for compiled template files (undef)

Plugins and Filters

PLUGINS	reference to a hash array mapping plugin names to Perl packages.
PLUGIN_BASE	base class(es) under which plugins may be found
LOAD_PERL	load Perl modules if plugin not found (0)
FILTERS	hash array mapping filter names to filter sub-routines or factories.

Compatibility, Customisation and Extension

V1DOLLAR	backwards compatibility flag
LOAD_TEMPLATES	list of template providers
LOAD_PLUGINS	list of plugin providers
LOAD_FILTERS	list of filter providers
TOLERANT	set providers to tolerate errors as declinations (0)
SERVICE	custom service obj (Template::Service)
CONTEXT	custom context obj (Template::Context)
STASH	custom stash object (Template::Stash)
PARSER	custom parser object (Template::Parser)
GRAMMAR	custom grammar obj (Template::Grammar)

Command line tools

tpage

tpage processes supplied templates and sends output to STDOUT; variables can be defined with:
`--define var=value ...`

ttree

ttree processes directory hierarchies of templates; it takes the following options:

-a	(--all)	process all files ignoring mod-times
-r	(--recurse)	recurse into sub-directories
-p	(--preserve)	preserve file ownership and permissions
-n	(--nothing)	do nothing, just print summary (enables -v)
-v	(--verbose)	verbose mode
-h	(--help)	display help
-dbg	(--debug)	debug mode
-s dir	(--src=dir)	source directory
-d dir	(--dest=DIR)	destination directory
-c dir	(--cfg=DIR)	location of configuration files
-l dir	(--lib=DIR)	library directory (INCLUDE_PATH) (multiple)
-f file	(--file=FILE)	read named configuration file (multiple)

File search specifications (all may appear multiple times):

--ignore=regex	ignore files matching regex
--copy=regex	copy files matching regex
--accept=regex	process only files matching regex

Additional options to set Template Toolkit configuration items:

--define var=value	define template variable
--interpolate	interpolate variables in text
--anycase	accept keywords in any case
--pre_chomp	chomp leading whitespace
--post_chomp	chomp trailing whitespace
--trim	trim blank lines around blocks
--eval_perl	evaluate PERL code blocks
--load_perl	load regular Perl modules via USE directive
--pre_process=TEMPLATE	add TEMPLATE as header for each file
--post_process=TEMPLATE	add TEMPLATE as footer for each file
--process=TEMPLATE	wrap TEMPLATE around each file
--default=TEMPLATE	use TEMPLATE as default
--error=TEMPLATE	use TEMPLATE to handle errors
--start_tag=STRING	STRING defines start of directive tag
--end_tag=STRING	STRING defines end of directive tag
--tag_style=STYLE	use pre-defined tag style STYLE
--plugin_base=PACKAGE	base PACKAGE for plugins
--compile_ext=STRING	extension for compiled templates
--compile_dir=DIR	directory for compiled templates
--perl5lib=DIR	additional Perl library directory

Perl API

```
use Template;
$tt = Template->new(\%config);
$tt->process($template, \%vars[, $output]);
$tt->service;
$tt->context;
$tt->error;
```