Table of Contents

DBCachePlugin **1**
- Description 1
- Syntax 2
  - ATTACHMENTS 2
  - DBCALL 3
  - DBDUMP 4
  - DBQUERY 4
  - DBRECURSE 5
  - DBSTATS 7
  - TOPICTITLE 8
- Syntax of search queries 8
  - Examples 9
- Perl API 9
  - DBCachePlugin 9
    - getDB() 9
  - DBCachePlugin::WebDB 10
    - dbQuery() 10
    - getFormField() 10
- Plugin Settings 10
  - MemoryCache 11
- Plugin Installation Instructions 11
- Plugin Info 11

DBCachePlugin


Powered by
WikiRing Consultants

Description

This is a lightweight frontend to the DBCacheContrib. The provided macros DBQUERY and DBCALL can be used as a replacement for SEARCH and INCLUDE that use the database store instead of searching and extracting on the raw topic files.

DBQUERY is provided to ease the use of WikiApplications based on the DBCacheContrib combining its flexible query language with unrestricted formatting and an extend mechanism to extract topic properties.

DBCALL is named "call" and not "include" as its main purpose is to ease WikiApplications where TopicFunctions are treated as stored procedures. It fetches pre-compiled topics or sections of it. "Glue" chars (see) are applied in advance so that the actual rendering time is minimized. DBCALL does therefore not allow to call "external" pages as INCLUDE allows. DBCALL's "warn" parameter can only be set to "on" and "off" and does not allow alternative content in case of a warning. The "pattern" feature used to extract a fragment from an INCLUDEd text using regular expression has been dropped. That aside, DBCALLs can be called recursively as INCLUDEs can, parametrized using key-value arguments and obeys to %STARTINCLUDE%, %STOPINCLUDE%, %STARTSECTION{"..."}% and %ENDSECTION{"..."}%.

Using the DBCacheContrib topic sections are stored into the topic object of the database. The section between %STARTINCLUDE% ... %STOPINCLUDE% is called `_sectiondefault` whereas all sections between %STARTSECTION{"<name>"}% .... %ENDSECTION{"<name>"}% are called `_section<name>` and are stored accordingly.

In addition, DBCALL implements so called **TopicMethods**. A method call is of the form `%DBCALL{"SomeTopic->RenderMethod" ...}%` and will infer the actual `...RenderMethod` to be called on the base of the *TopicType* of `SomeTopic`. TopicTypes are a concept introduced by the . It basically refers to a convention to tag the type(s) of a topic in a formfield "TopicType". The lookup mechanism for TopicMethods proceeds as normal if the DBCALl is not of the form `"SomeTopic->RenderMethod"`. For example, if `SomeTopic` is of type `Video, MediaType, ClassifiedTopic` the lookup mechanism will try to find the methods `VideoRenderMethod`, `MediaTypeRenderMethod` and `ClassifiedTopicRenderMethod` in the given order and use the one found first. The web these types are defined in is inferred on the base of the DataForm definitions of each of these types. If none of these type-specific methods is found the method call will fallback to a normal non-specific function call. The TopicMethod itself is called with an implicit parameter `OBJECT` that refers to `SomeTopic`, that is the object this method has been called for.

ATTACHMENTS lists the attachments of a given topic giving you a high degree of control about what and how to display attachments.

DBRECURSE iterates over topics in a web following a certain topic-to-topic relation, like the parent-child relation among topics. You can define arbitrary relations using filters.

Syntax

ATTACHMENTS

List all attachments of a given topic. Without any parameters it displays a table of attachments similar to the one generated by the templates at the bottom of a topic. There are a couple of parameters that take regular expressions to limit the number of attachments to be listed.

Syntax:

%ATTACHMENTS%, %ATTACHMENTS{"<topic>" ...}	
"<topic>"	topic whos attachments we want to query
attr="..."	match the attachments' attribute
autoattached="yes,no,undef"	limit attachments to those that are (not) autoattached
comment="..."	match the attachments' comment
maxdate="..."	specify latest attachment
maxsize="..."	specify biggest attachment
mindate="..."	specify earliest attachment
minsize="..."	specify smallest attachment
names="..."	match the name of attachments to be include in the list
user="..."	match the attachments' user who uploaded it
footer="..."	format string appended to the result
format="..."	how to format each attachment
header="..."	format string prefixing the resulting output
separator="..."	format string put between each formatted attachment
hidenull="on,off"	hide/display the empty result; only the header and the footer are included if set to "off" and there's no attachment at the given topic
sort="name,date,size,user,comment,comment:name"	sort results by specified field
reverse="on/off"	enable reverse sorting
limit="..."	limit the number of attachments to the given number; by default show all

The following variables can be used in the format parameter:

- $attr: the attributes
- $autoattached: the autoattach flag
- $comment: the comment field
- $date(<format>): the attachment date using <format> to format the date
- $date: the attachment date
- $delete: the action to delete the current attachment
- $deleteUrl: the url used in the delete action
- $icon: img tag representing the filetype
- $iconUrl: the url pointing to the icon used in the img tag
- $index: the position index of the attachment in the result
- $move: the action to move the current attachment

- $moveUrl: the url used in the move action
- $name: the name of the attachment
- $path: the path property of the attachment
- $props: the action to change the properties of the current attachment
- $propsUrl: the url used in the props action
- $size: the size in bytes
- $sizeK: the size in kilo bytes
- $sizeM: the size in mega bytes
- $topic: the name of the topic of whose attachments are listed
- $type: the file type of the current attachment
- $url: the attachments url
- $urlpath: the attachments urlpath
- $user: the user that uploaded the attachment
- $web: the web name of the current topic
- $webdav: the action to edit the current attachment using the Foswiki:Extensions/WebDavPlugin
- $webdavUrl: the url used in the webdav action
- $wikiuser: the user's home topic that recently changed the current attachment

The header and footer format strings may only contain

- $count: the number of attachments found

The format, header and footer may contain the standard escape sequences:

- $dollar: $ sign
- $n: newline
- $percnt: % sign

Example:

```
%ATTACHMENTS{format="| $name: | $type |"}%
```

generates:

wikiringlogo40x40.png:	$type

DBCALL

Fetch topic content or a section of it from the database store.

Syntax:

%DBCALL{"<topic" ... }%	
`"<topic>"`	topic that we'd like to call
`section="..."`	optional named section; without this parameter the "default" section is used
`<key>="<value>"`	topic arguments to be substituted within the included text; that is every occurrence of <key> is substituted with <value>
`warn="on,off"`	enable/suppress error warnings
`remote="on,off"`	switch fixing WikiWords in transcluded content; `on` means "don't touch, this is remote content" defaults to "off" because that's the default behaviour of `%INCLUDE{}%` (note, DBQUERY defaults to "on" aka do-not-touch-mode)

DBDUMP

Display the database record of a topic (useful for debugging).

Syntax:

%DBDUMP{"<topic>" ... }%	
`"<topic>"`	topic that we'd like to debug
`web="..."`	name of the web to look for the given topic; note that you can use the dot-notation in specifying the web.topic also

DBQUERY

DBQUERY can be used in either of two modes (a) as a pure search tool or (b) as a tool to extract properties of (a set of) known topics.

Syntax:

%DBQUERY{"<search>" ... }%	
`"<search>"`	search clause, see below
`topics="..."` or `topic="..."`	set of topics to consult (mode (b)); if "topics" are specified in addition to a "search" it will be restricted to the given topics
`web="..."`	web where to search for hits; defaults to the current one
`format="..."`	format string to display search hits; defaults to "$topic"; the special format string "none" disables the format string
`header="..."`	format string to prepended to the list of hits
`footer="..."`	format string to appended to the list of hits
`separator="..."`	format string used to separate hits; defaults to "$n"; the special separator "none" disables separation
`include="..."`	pattern each found topic name must match to be considered a hit
`exclude="..."`	pattern each found topic name must not match to be considered a hit
`sort="..."`	specifies the sorting of hits; defaults to "name"
`reverse="..."`	specify if hits should be sorted in reverse order; defaults to "off"
`limit="..."`	maximum number of topics to include in the hit set
`skip="..."`	number of topics to skip while constructing the hit set; defaults to "0"
`hidenull="..."`	flag to indicate that an empty hit set should not be displayed
`remote="on,off"`	switch fixing WikiWords in transcluded content; `on` means "don't touch, this is remote content" defaults to "on"

Following variables are expanded in format strings:

- $percnt: % sign
- $dollar: $ sign
- $n: newline
- $nop: "empty string"
- $count: the number of hits
- $index: the current row index in the search
- $web: the web where the hits where found
- $formfield(<formfield-name>): the value of the given formfield
- $formatTime(<formfield-accessor> [,<format>]): format the datefield pointed to by the <formfield-accessor> using the given <format>; the format can be in any form supported by `Foswiki::Func::formatTime()` (see Macros documentation).
- $expand(<formfield-accessor>): return the formfield pointed to by the <formfield-accessor>

- $rss(...): encode entities so that the text can be included in an rss feed
- $encode(...): encode entities so that the text can be included in HTML input form elements
- $flatten(...): remove any special markup producing a flattened text representation
- $trunc(...,length): truncate a string to the given length
- $d2n(...): converts a date string to epoch seconds
- $uc(...): converts a string to upper case
- $lc(...): converts a string to lower case

Formfield accessors:

In its most basic form `$expand(Ref)` is identical to `$formfield(Ref)`. In addition `$expand()` can be used to refer to topic properties of related topics that can be reached from the current one using the '@' symbol. For example, if a topic A uses a form that contains a field named "Ref" and specifies a reference to another topic B (by using its name) you can access the "Headline" in B from A using `$expand(@Ref.Headline)`.

A formfield accessor can be a composed one using 'or' and 'and'. Example: `$expand(Name or Author)` will expand to the value of the formfield "Name" if it exists and is non-empty and to the value of the formfield "Author" otherwise. More valid examples:

- `$expand(@Ref.Headline)`: headline of the referred topic
- `$expand(Nr and '.')`: append a dot to the formfield value "Nr" if it exists
- `$expand(Name or Author)`: expand to "Name" or "Author"

A formfield accessor can be computed using TopicMarkup. If the string starts with '%', it is expanded and its result is taken as the real formfield accessor. This is useful when the name of the formfield must be computed by different means.

DBRECURSE

Traverse topic-to-topic relations, like the intrinsic parent-child relation

Syntax:

%DBRECURSE{"<topic>" ...}%	
`<topic>`	starting point, can be in web.topic format, defaults to the current topic
`web="..."`	web where to recurse, defaults to the web given in the `topic` parameter or the current web
`format="..."`	format string for each found topic defaults to `'  $indent* [[$web.$topic][$topic]]'`
`single="..."`	format string used when only one topic is found, defaults to value of `format` parameter
`separator="..."`	separator string to be put between formatted topics, defaults to `'\n'`
`header="..."`	format string to be prepended to the output, empty by default
`subheader="..."`	format string to be prepended to each subordinated search, defaults to `header` parameter
`singleheader="..."`	format string to be used when only one topic is found
`footer="..."`	format string to be appended to the output
`subfooter="..."`	format string to be appended to each subordinated search, defaults to `footer` parameter
`singlefooter="..."`	format string to be used when only one topic is found
`hidenull="..."`	flag to indicate that an empty hit set should not be displayed

%DBRECURSE{"<topic>" ...}%	
`filter="..."`	search expression that is used in each recursion step; the current topic name is inserted into the filter expression by replacing the `'$ref'` string; the default filter is `'parent='$name'`; this will search for topics of which the current topic is a parent
`sort="..."`	determines the sorting order of topics in each iteration step
`reverse="on,off"`	reverse sorting order
`limit="..."`	maximum iterations, default is `0` (unlimited)
`skip="..."`	skips the first `n` hits when formatting the output, defaults to `0` (no skip)
`depth="..."`	only recurse to a given depth, defaults to `0` (unlimited)
`include="..."`	regular expression topics must match to be included in the search
`exclude="..."`	regular expression that excludes matching topics from the search

The `...format`, `...header` and `...footer` format strings may contain the following variables:

- $web: the current web
- $topic: the current topic in the search
- $index: the topic index number in one recursion step, that is an enumeration in each recursion step
- $number: the "paragraph" number in the recursion, e.g. `1.2.1.5` depicts the path alternative in the recursion tree
- $count: number of found topics; this is the total number of found topics in the `header` and `footer` and the number of topics *found so far* during recursion
- $indent: 3 spaces per depth
- $indent(<string>): uses `<string>` for indenting the result in each depth
- $formfield(...): see #DBQUERY
- $expand(...): see #DBQUERY
- $formatTime(...): see #DBQUERY

All format strings may contain the following variables:

- $dollar: $ sign
- $n: newline
- $percnt: % sign

Example:

```
%DBRECURSE{"%SYSTEMWEB%.FrequentlyAskedQuestions"}%
```

lists all FrequentlyAskedQuestions topics (with properly set topic parent):

- AnApplicationWithWikiForm
- DeleteOrRenameATopic
- DeleteOrRenameAnAttachment
- DownloadSources
- EditDoesNotIncreaseTheRevision
- FaqTemplate
- GnuGeneralPublicLicense
- SearchDoesNotWork
- SimultaneousEdits
- WhatIsWikiWiki

DBSTATS

Compute and format a statistics on the database.

Syntax:

%DBSTATS{"<search>" ...}%	
"<search>"	query string that defines a search, see DBCacheContrib
web="..."	the web where to search in (default: current web)
field(s)="..."	name of one or more formfields to be extracted (default: "text")
split="..."	regex split up a field value into a list before matching each item against pattern
pattern="..."	regex pattern to extract keys from the formfield(s); note, that before the pattern is applied the field value is split up using the split parameter; you must provide at least one grouping ("(...)") in the pattern string that encloses the key to be extracted (default: "^(.*)$"); you may have up to five groupings in one pattern that are accessible each in the format string
header="..."	header format string to be prepended to the output
format="..."	format string used for each key in the output
sep(arator)="..."	separator to put between formatted keys
footer="..."	footer format string to be appended to the output
sort="alpha,created"	order of keys in the output
reverse="on,off"	sort keys descending or ascending
limit="..."	maximum number of keys to be formatted in the output
hidenull="..."	flag to indicate that an empty hit set should not be displayed

The DBSTATS searches for all topics in given web and reads the given form fields. It then uses the split and pattern parameters to extract all keys in the data counting their occurrences. The result is rendered using the format parameter for each token found, separating them by sep, prepend the header and append the footer. Results can be sorted alphabetically or by createdate.

Format strings (header, format, footer) can use the following variables to insert the computed statistics:

- $key, $key1: the found key, this must match the first group in the pattern argument
- $key2, $key3, $key4,$key5: second, third, etc grouping in the pattern argument
- $count: the number of occurrences of the key
- $index: index of the key within the sorted result set
- $min: minimum count found in the search
- $max: maximum count found in the search
- $sum: sum of all counts
- $mean: mean occurrence of a key in the result set
- $keys: total number of keys found
- $web: the web we search in
- $topics: the list of all topics where the keys where found

For example if you want to extract all month/years from a database's Date field that has the format dd mon yyyy and then format a link to an archive using the month and the year as separate url parameters then use something like this:

```
%DBSTATS{"<db query>"
   field="Date"
   pattern="((\w+) (\d\d\d\d))"
   header="---++ Archive"
   format="   * <a href="...?month=$key2&year=$key3">$key</a>"
```

```
}%
```

Note, that the above pattern will crop away the day. Groups can be nested counting them from left to right, outside to inside: the first group matches the compete pattern, the second the month the third the year.

TOPICTITLE

derive the title of a topic from a couple of properties:

Syntax:

%TOPICTITLE{"<topic>"}%	
`<topic>`	topic we want the page title off, defaults to current topic

- from a TOPICTITLE preference variable, or
- from a "TopicTitle" formfield attribute, or
- defaults to the topic name

Syntax of search queries

A search query is a boolean expression on fields of a topic. (Tip: use %DBDUMP to explore the available fields).

Fields are given by name, and values by strings or numbers. Strings should always be surrounded by 'single-quotes'. Strings which are regular expressions (RHS of =, != =~ operators) use 'perl' regular expression syntax (google for `perlre` for help). Numbers can be signed integers or decimals. Single quotes in values may be escaped using backslash (\).

The following operators are available:

Operator	Result	Meaning
=	Boolean	LHS exactly matches the regular expression on the RHS. The expression must match the whole string.
!=	Boolean	Inverse of =
=~	Boolean	LHS contains RHS i.e. the RHS is found somewhere in the field value.
<	Boolean	Numeric <
>	Boolean	Numeric >
>=	Boolean	Numeric >=
<=	Boolean	Numeric <=
lc	String	Unary lower case
uc	String	Unary UPPER CASE
IS_DATE	Boolean	Compare two dates e.g. '1 Apr 2003' IS_DATE '1 Apr 2004'
EARLIER_THAN	Boolean	Date is earlier than the given date
EARLIER_THAN_OR_ON	Boolean	Date is earlier than, or on, the given date
LATER_THAN	Boolean	LHS is later than the given date
LATER_THAN_OR_ON	Boolean	LHS is later than the given date
WITHIN_DAYS	Boolean	Date (which must be in the future) is within n *working* days of todays date
!	Boolean	Unary NOT

Operator	Result	Meaning
AND	Boolean	AND
OR	Boolean	OR
()	any	Bracketed subexpression

Dates for the date operators (IS_DATE, EARLIER_THAN etc) must be dates in the format expected by Time::ParseDate (like the). WITHIN_DAYS works out the number of *working* days assuming a 5 day week (i.e. excluding Saturday and Sunday). Apologies in advance if your weekend is offset ± a day! Integers will automatically be converted to dates, by assuming they represent a number of seconds since midnight GMT on 1st January 1970. You can also use the d2n operator to convert a date string to such an integer.

Examples

- Find all topics that have been last modified before 1st January 2008

```
%DBQUERY{"info.date EARLIER_THAN '1st January 2008'"}%
```

- Find all topics last modified by Peter

```
%DBQUERY{"info.author =~ 'Peter'"}%
```

- Find all FAQs

```
%DBQUERY{"topic =~ 'FAQ'"}%
```

- Find all topics that have the word "random" in it (case-insensitive) excluding the current topic

```
%DBQUERY{"lc(text) =~ 'random' and topic != 'DBCachePlugin'"}%
```

Perl API

The DBCachePlugin supports overloading the contained default database cache by inheriting from DBCachePlugin::WebDB being itself a DBCacheContrib. The only purpose of the DBCachePlugin::WebDB is to extract the TopicFunctions contained in a topic as described above. You can easily extend this functionality by deriving a WebDB specific to your own WikiApplication. For example, the defines a

DBCachePlugin

getDB()

Returns a database object used for further queries. This function must be called ahead of any actual database access. If your WikiApplication is using a derived WebDB then use the WEBDB variable in your WebPreferences to point to its implementation (example: =Set WEBDB = Foswiki::Plugins::BlogPlugin::WebDB=). If WEBDB is not defined the default implementation Foswiki::Plugins::DBCachePlugin::WebDB is used. Note, that this way only one application-specific database cache can be loaded *per web*. This is rather a design decision; otherwise each DBQUERY and DBCALL tag would have to declare which database it operated on. So currently the rule of thumb is: one web one WikiApplication making use of this plugin.

returns a database object. See the documentation for its interfaces.

DBCachePlugin::WebDB

This implements the database object that is used to access the topic records in a web. Note, that for each web there is exactly one WebDB database object. A database object is fetched (and initialized if needed) by the DBCachePlugin::getDB() function.

dbQuery()

The DBQUERY functionality can be access from within perl using the plugins `dbQuery()` method.

(@$topicNames, %$topicObjs, $errorMsg) = $db->dbQuery([$search, @$topics, $order, $reverse, $include, $exclude])	
`webDB`	database object
`search`	search clause
`topics`	restrict search to this list of topics
`order`	define a sorting on the hit set; this can be any formfield accessor or one of the shortcuts "created" (for createdate) or "modified" (for info.date) provided for compatibility with default %SEARCH
`reverse=on,off`	revert the sorting order
`include`	pattern that topic names must match to be considered a hit
`exclude`	pattern that topic names must *not* match to be considered a hit

returns a list pointer of all found topics names, a hash pointer to an array of all found topic objects (keys are the name of the topics) and a possibly defined error message.

getFormField()

Access the formfield value of an arbitrary topic.

$value = $db->getFormField($topic, $formfield)	
`topic`	a topic name
`formfield`	a formfield name

returns the value of the named formfield

Plugin Settings

See also DBCacheContrib for additional configuration settings.

Each web can specify an alternative implementation inheriting from DBCachePlugin::WebDB. To specify it you have to set the `WEBDB` variable in the WebPreferences. The default is `Foswiki::Plugins::DBCachePlugin::WebDB`

MemoryCache

If `$Foswiki::cfg{DBCache}{MemoryCache}` is set to FALSE (defaults to TRUE) the dbcache once loaded into memory will stay there among multiple requests. This option only has an effect using speedy/perperl or fastcgi setups.

Plugin Installation Instructions

You do not need to install anything in the browser to use this extension. The following instructions are for the administrator who installs the extension on the server.

Open configure, and open the "Extensions" section. Use "Find More Extensions" to get a list of available extensions. Select "Install".

If you have any problems, or if the extension isn't available in `configure`, then you can still install manually from the command-line. See http://foswiki.org/Support/ManuallyInstallingExtensions for more help.

Plugin Info

Author:	Michael Daum
Copyright ©:	2005-2010, Michael Daum http://michaeldaumconsulting.com
License:	GPL (GNU General Public License)
Release:	3.50
Version:	8342 (2010-07-28)
Change History:	
28 Jul 2010:	added Config.spec to ease configuration, i.e. memory caching; added documentation for `$sum`; added support for Foswiki:Extensions/MetaCommentPlugin
12 Feb 2010:	ATTACHMENTS can now list old revisions of an attachment
17 Nov 2009:	fixed incremental cache updates; exclude autoinc pattern in TOPICTITLE; properly combine header, body and footer in DBQUERY to support CALC; added `split` and `casesensitive` to DBSTATS; added support for standard escapes in DBQUERY's `sort`; added `$d2n()` , `$uc()`, `$lc()`
02 Jul 2009:	optimed cache maintenance cycles by making use of new `loadTopic()` api in DBCacheContrib
28 Jun 2009:	final detwikification; added compatibility hack for current Foswiki/trunk
18 Jun 2009:	Crawford Currie: reverted to using DBCacheContrib
09 Jan 2009:	internalized compatible version of DBCacheContrib; added limit to ATTACHMENTS and a few more ways to sort them
07 Jan 2009:	new global variable `dbQueryCurrentWeb` being set during a DBQUERY so that external predicates that depend on the web information can make use of it; renderWikiWord also displays topic titles for explicit links where the link text equals the topic name; internalized urlDecode and parseTime for compatibility reasons; fixed info.author for newer foswiki engines
11 Dec 2008:	added `MemoryCache` flag to switch off memory persistence
11 Dec 2008:	working around issue in the Foswiki parse where an undefined `%VAR%` in parametrized includes is expanded to VAR instead of leaving it to `%VAR%`
24 Sep 2008:	expanding common variables in footers and headers
03 Jul 2008:	

	fixed calculation of a topic title; disabled `renderWikiWordHandler` for legacy wiki engines, so not replacing the WikiName with its TopicTitle
28 Apr 2008:	implemented TopicMethods in DBCALL
05 Feb 2008:	addded `rss()` feature to format strings
4 Jan 2008:	Item5319: fix empty separator
11 Dec 2007:	renamed PAGETITLE to TOPICTITLE, matching formfield TopicTitle
07 Dec 2007:	added PAGETITLE, defaulting to BASEWEB instead of INCLUDINGWEB in all of the tags
14 Nov 2007:	protect against data store pollution, at least don't die
12 Sep 2007:	made significant speedups to DBQuery fixed sorting to be O(N log(N)) removed topic access check unless absolutly necessary - SvenDowideit@home.org.au
19 Jun 2007:	added compatibility hack to display users as WikiNames; added expand(%TML%) feature
08 May 2007:	fixed loading db cache file since recent changes in DBCacheContrib
03 Apr 2007:	fixed $date(format) in ATTACHMENTS; fixed 'segfault' in DBRECURSE while formatting sub-results
19 Feb 2007:	multiple enhancements to the ATTACHMENTS tag (e.g. icons, actions)
31 Jan 2007:	fixed caching "web" property of topics; fixed DBDUMP of topics w/o attachments; added numerical sorting of formfields; added support for the Foswiki::Cache; added new tag DBRECURSE to itterate along topic relations
24 Jan 2007:	added ATTACHMENTS tag
09 Nov 2006:	fixed bug where a topic modification was not detected correctly resulting in an outdated cache when using perl accelerators; added pseudo-variable `$topics` to DBSTATS to list all topics in a class; added DESTROY methods for caches fixing memory leakage; fixed expansion order of pseudo-variable; speed improved DBQUERY by first checking the match and access rights afterwards (thanks to CDot)
13 Oct 2006:	don't use the separator for header and footer format strings; fix links in DBQUERY the same way we do it in DBCALL; added `remote` parameter to switch on/off fixing links in transcluded content; expansion of `$nop` and `$n` in that order
28 Sep 2006:	DBQUERY and DBSTATS did not obey access rights
19 Sep 2006:	added $key1...$key5 to the pattern matcher and formatter
18 Sep 2006:	properly fixed parsing $trunc,$flatten and $encode in format strings; multiple fixes for calls and formatting accross webs
31 Aug 2006:	added NO_PREFS_IN_TOPIC; fixed possibly insecure eval
22 Aug 2006:	fixed parsing $trunc() and $flatten() in format strings
15 June 2006:	generate an inline error instead of an oops on a bad search string to DBQUERY
05 Mai 2006:	don't segfault on uncompiled WebDB implementations
30 Mar 2006:	added WebDB property createdate formerly only known by the BlogPlugin; dont choke if DBDUMPing a non-existing topic;
27 Mar 2006:	added $trunc()
20 Mar 2006:	using Foswiki::Attrs to parse section names
12 Mar 2006:	added hidenull to DBSTATS
10 Mar 2006:	use ENV instead of the CGI object to detect the current cgi action
02 Mar 2006:	fixed bad init error showing up on speedy/mod_perl
01 Mar 2006:	added DBSTATS tag; modularized the plugin for delayed compilation
15 Feb 2006:	don't use the query object in scripted mode
14 Feb 2006:	forcing to reload the database after a save
10 Feb 2006:	more performance on mod_perl/speedy_cgi: don't reload the topic cache if it hasn't changed
03 Jan 2006:	fixed INCLUDING(WEB,TOPIC) in DBCALL; don't apply glue in advance anymore when storing topic sections, it's serving a better purpose when done during a DBCALL
26 Jan 2006:	fixed internal links in DBCALL; support for STARTSECTION, old SECTION being deprecated
24 Nov 2005:	Initial version

Dependencies:	Name	Version	Description
	Time::ParseDate	>=2003.0211	Required. Available from CPAN.
	Storable	>=2.07	Recommended for sites with few topics. Available from CPAN
	Foswiki::Contrib::DBCacheContrib	>=4324	Required. Database module.
	BerkeleyDB	>=0	Recommended for sites with many topics. Available from CPAN
Home:	Foswiki:Extensions/DBCachePlugin		
Support:	Foswiki:Support/DBCachePlugin		

Edit | Attach | Print version | History: %REVISIONS% | Backlinks | Raw View | More topic actions

Topic revision: r3 - 17 Feb 2010 - 14:18:24 - ProjectContributor

- . System

- Log In

- **Toolbox**
- Users
- Groups
- Index
- Search
- Changes
- Notifications
- RSS Feed
- Statistics
- Preferences

- **User Reference**
- BeginnersStartHere
- TextFormattingRules
- Macros
- FormattedSearch
- QuerySearch
- DocumentGraphics
- SkinBrowser
- InstalledPlugins

- **Admin Maintenance**
- Reference Manual
- AdminToolsCategory
- InterWikis
- ManagingWebs
- SiteTools
- DefaultPreferences
- WebPreferences

- **Categories**
- Admin Documentation
- Admin Tools
- Developer Doc
- User Documentation
- User Tools

- **Webs**
- Public
- System

Copyright © by the contributing authors. All material on this site is the property of the contributing authors.
Ideas, requests, problems regarding Wiki? Send feedback