Core - Bug # 52125

Epic # 55070 (Accepted): Workpackages
Epic # 55065 (New): WP: Overall System Performance (Backend and Frontend)
Bug # 52949 (Resolved): Speed decrease since 4.5

Status:	Resolved	**Priority:**	Should have
Author:	Xavier Perseguers	**Category:**	Performance
Created:	2013-09-19	**Assigned To:**	
Updated:	2014-12-12	**Due date:**	
TYPO3 Version:	6.1		
PHP Version:	5.4		
Complexity:			
Is Regression:	No		
Sprint Focus:			
Subject:	Saving records takes ages to complete		

Description

After Xhprof'ing a website to understand why it took about 17 seconds to save a record in Backend, I found that cache files (on disk) were read 130k times during the process!

Digging more in it, I found that the problem was that a few Core caches are configured by default with FileBackend which is extremely inefficient in its flushCacheByTag implementation because it basically needs to open and check each file in a row to check if it should be unlinked.

The big problem here comes on one hand from the "high" number of Cache files I have in my website:

- 507 files in Cache/Data/t3lib_10n
- 8 in cache_core
- 1 in cache_phpcode
- 35 in fluid_template
- 1 in static_info_tables

(total = 552 files)

on the fact that on my production website I don't have SSD for storing data (which is typically the case) and on the other hand on the caching framework which basically is invoked on every cache Backend when a record is saved with a call like clearCacheByTag('page_<uid>') recursively for each page in the rootline. If you are a few levels deep, you end up with N * 552 files read for nothing because tags are not used anyway on those files.

I switched to APC for t3lib_l10n and the time dropped from 17 sec. down to 4 sec.

Suggestion

- Try to change the default configuration from FileBackend to SimpleFileBackend for as much as many default configuration
- Discuss if FileBackend should be changed, to possibly remove the "tag handling" and simply purge files or at least issue a big warning in documentation that this cache is extremely inefficient and the more cache files we have, the slower TYPO3 will be, no magic here.
- Something else?

Related issues:

related to Core - Bug # 34886: t3lib_cache_backend_FileBackend don't support ...	**Resolved**	**2012-03-15**
related to Core - Bug # 51116: Massive speed problem from TYPO3 version 6.x a...	**Resolved**	**2013-08-16**
related to Core - Bug # 52235: Timeout when copying pages recursively due to ...	**Resolved**	**2013-09-24**

related to Core - Task # 52295: Use SimpleFileBackend for t3lib_l10n cache	**Resolved**	**2013-09-26**

Associated revisions

Revision 3d0008fe - 2013-09-26 21:23 - Christian Kuhn

[TASK] Use SimpleFileBackend for t3lib_l10n cache

The language cache by default uses the FileBackend to store its data.
Language cache entries need to be deleted only if new extensions are
loaded and if new language overlays are fetched. They do not need
tagging and can have an unlimited lifetime.
Switching to SimpleFileBackend removes the tagging and sets unlimited
lifetime by to reduce read and write load on this cache.

Change-Id: I5c4778f4c38ae369b6873574e961fa65208d77a1
Resolves: #52295
Related: #52125
Releases: 6.2, 6.1, 6.0
Reviewed-on: https://review.typo3.org/24064
Reviewed-by: Wouter Wolters
Reviewed-by: Xavier Perseguers
Tested-by: Xavier Perseguers
Reviewed-by: Thomas Maroschik
Tested-by: Thomas Maroschik
Reviewed-by: Christian Kuhn
Tested-by: Christian Kuhn

Revision 52ff4008 - 2013-09-26 21:25 - Christian Kuhn

[TASK] Use SimpleFileBackend for t3lib_l10n cache

The language cache by default uses the FileBackend to store its data.
Language cache entries need to be deleted only if new extensions are
loaded and if new language overlays are fetched. They do not need
tagging and can have an unlimited lifetime.
Switching to SimpleFileBackend removes the tagging and sets unlimited
lifetime by to reduce read and write load on this cache.

Change-Id: I5c4778f4c38ae369b6873574e961fa65208d77a1
Resolves: #52295
Related: #52125
Releases: 6.2, 6.1, 6.0
Reviewed-on: https://review.typo3.org/24064
Reviewed-by: Wouter Wolters
Reviewed-by: Xavier Perseguers
Tested-by: Xavier Perseguers
Reviewed-by: Thomas Maroschik
Tested-by: Thomas Maroschik
Reviewed-by: Christian Kuhn
Tested-by: Christian Kuhn
(cherry picked from commit 3d0008feb8afa84b1f39ceeb017de2a2d4aca3e4)
Reviewed-on: https://review.typo3.org/24079

Revision d00db271 - 2013-09-26 21:35 - Christian Kuhn

[TASK] Use SimpleFileBackend for t3lib_l10n cache

The language cache by default uses the FileBackend to store its data.
Language cache entries need to be deleted only if new extensions are
loaded and if new language overlays are fetched. They do not need
tagging and can have an unlimited lifetime.
Switching to SimpleFileBackend removes the tagging and sets unlimited
lifetime by to reduce read and write load on this cache.

Change-Id: I5c4778f4c38ae369b6873574e961fa65208d77a1
Resolves: #52295
Related: #52125
Releases: 6.2, 6.1, 6.0
Reviewed-on: https://review.typo3.org/24080
Reviewed-by: Christian Kuhn
Tested-by: Christian Kuhn

History

#1 - 2013-09-19 09:06 - Xavier Perseguers

- Subject changed from Record saving takes ages to complete to Saving records takes ages to complete

#2 - 2013-09-19 10:08 - Tymoteusz Motylewski

In Magento there was very similar issue with the default implementation of the file cache backend (Zend_Cache_Backend_File).
Colin Mollenhour wrote a custom implementation of the file backend which makes tags cleaning thousands times faster.
I think it would be good idea to inspire new TYPO3 file backend implementation on it.

https://github.com/colinmollenhour/Cm_Cache_Backend_File

quote from the readme:
"This cache backend works by indexing tags in files so that tag operations do not require a full scan of every cache file. The ids are written to the tag files in append-only mode and only when files exceed 4k and only randomly are the tag files compacted to prevent endless growth in edge cases.

The metadata and the cache record are stored in the same file rather than separate files resulting in fewer inodes and fewer file stat/read/write/lock/unlink operations. Also, the original hashed directory structure had very poor distribution due to the adler32 hashing algorithm and prefixes. The multi-level nested directories have been dropped in favor of single-level nesting made from multiple characters."

#3 - 2013-09-24 21:18 - Jan-Erik Revsbech

I have the exact same problem, and have debugged down to the same issue. Switching to APC helped for us as well, but I think this should be fixed. I agree that the FileBackend should not be used for anything by default, as it has serious scaling problems.

Another thing is, why does the DataHandler flush all caches? Should i not only clear the Page (and possibly the pagesection) cache? I would suggest changing

```
$GLOBALS['typo3CacheManager']->flushCachesByTag('pageId_' . $pageId);
```

to

```
$GLOBALS['typo3CacheManager']->get('page_cache')->flushByTag('pageId_' . $pageId);
```

Would any other cache have identifiers with the pageId_ prefix?

Another problem is that clearCache is called every time insertDB or updateDb is called in the DataHansler. So Copying a page with 4 content elements, will result in (at least) 5 calls to $GLOBALS['typo3CacheManager']->get('page_cache')->flushByTag('pageId_' . $pageId) making the matter even worse. I will create another ticket for this as it is not really related.

#4 - 2013-09-25 00:31 - Christian Kuhn

If so many calls go to t3lib_l10n, we may refactor to create cached php code and require_once it (similar to cache_core), APC would automatically step in then.

I also wonder why cache_l10n is a file backend at all.

Caching in geneal has some issues, eg. default cache lifetimes are semi-clever and need an eye.

#5 - 2013-09-25 08:25 - Xavier Perseguers

- Target version set to next-patchlevel

#6 - 2013-09-26 16:09 - Christian Kuhn

- Parent task set to #52304

#7 - 2014-01-09 16:41 - Markus Klein

- Parent task changed from #52304 to #52949

#8 - 2014-06-12 19:49 - Stephan Großberndt

Is this resolved by the "[TASK] Use SimpleFileBackend for t3lib_l10n cache"-revisions? Or should this stay open because of the Zend_Cache_Backend_File idea? Close this issue and create a Feature for it?

#9 - 2014-12-12 23:57 - Christian Kuhn

- Status changed from New to Resolved

I'll set this issue to "resolved" for now - next to the SimpleFileBackend change, there where additional changes that lowered the load from l10n caches.

If write load in this area is still an issue, it should be handled with new and dedicated tickets.