Chamilo LMS - Feature #4635

Improve usage of autoload in global

20/04/2012 16:33 - Laurent Opprecht

Status:	Feature implemented	**Start date:**	20/04/2012
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2.0	**Spent time:**	0.00 hour
Complexity:	Normal	**SCRUM pts - complexity:**	?

Description

This may be my configuration but global.inc.php eats some time when loading. Not too much but still.
There a few class librairies that are loaded each time - Tracking for example. While this may required in most cases it won't be for rss feeds for example.
A solution could be to remove them from global.inc and moves them to main/inc/lib/autoload.class.php. So that they are loaded only when required.

By the way I went on with the following convention:

myname.class.php -> purely lib class
myname.lib.php -> mixed/function lib

Which should not be a surprised.

When moving files to autoload it would be good to rename to myname.class.php and avoid to mix in function declarations/calls

Related issues:

Related to Chamilo LMS - Feature #5523: Replace custom autoloader with Composer	**Feature implemented**	**21/09/2012**

Associated revisions

Revision 968f551a - 24/04/2012 11:51 - Julio Montoya

Fixing and cleaning system_status, fatal error due sortable class not found, now we use boostrap tab style and we use the new MIN_PHP_VERSION constant see #4635

Revision 04edfad3 - 24/04/2012 11:51 - Julio Montoya

Adding new MIN_PHP_VERSION constant see #4635

Revision 7cb2d544 - 08/05/2012 12:08 - Julio Montoya

Adding Twig options in order to improve performance see #4635

Revision 57c757dd - 08/05/2012 17:48 - Julio Montoya

Adding gradebook_block platform setting + adding c_id in the track_e_default table see #4635

Revision aa06716b - 25/05/2012 16:49 - Julio Montoya

Creating a new directory for Twig cache archive/twig see #4635

History

#1 - 20/04/2012 16:37 - Laurent Opprecht

I am seeing too includes of learnpath, mail, documents, etc. I don't believe those will be needed each time. They would be good candidate if we can move them - i.e. if they are classes.

#2 - 20/04/2012 16:48 - Laurent Opprecht

- File global.inc.png added

#3 - 20/04/2012 16:48 - Laurent Opprecht

- File callgrind.out.1334921687.4616 added

#4 - 20/04/2012 17:26 - Julio Montoya

I'm very happy that you analyze the chamilo code, I agree that the libraries need a sort of re organization.

We can also use the symfony2 classloader component

http://symfony.com/components
http://symfony.com/doc/current/components/class_loader.html
https://github.com/symfony/ClassLoader

#5 - 20/04/2012 17:37 - Yannick Warnier

Laurent Opprecht wrote:

> By the way I went on with the following convention:
>
> myname.class.php -> purely lib class
> myname.lib.php -> mixed/function lib

Please be extra-careful when renaming files (in case you did). It is particularly not welcome if this breaks something when we are already in alpha stage. I'm confident you won't do anything bad, but I had to say this :-)

#6 - 23/04/2012 16:39 - Laurent Opprecht

I actually did a few renamings - but was extra carefull indeed ;-) - I only changed those that where not having an impact. That is those with only classes in them. I left too the previous file name - at least for now - so that we don't hit a non-existing file.

The thing I saw so far:

- constants exists outside of the class. That is an issue because they may be called without going through the class. In this case the file is not autoloaded and we may hit an non existing const. Reason why I left those files the way they are. A better scheme is to including the const in the class and call them like that class_name::MY_CONST. That provides a namespace for the const and ensure they are loaded when needed.

- Some files includes if constructs. If class do not exists then declare it. This should go away with autoloading.

- Some files are function apis when a class would be better - cas_login, cas_logout. Using functions prevent autoloading and create other issues as well.

- Some files mix declarations: class and functions

- Lack of naming convention - which make difficult, well impossible in some circumstances, to locate the file containing the class

- the file names go on line of myclassname which is difficult to read, better schemes would be my_class_name, myClassName, MyClassName, whatever.

- some classes have duplicate names (I left them out of autoloading)

> Attendance => /main/inc/lib/attendance.lib.php
> Attendance => /main/coursecopy/classes/Attendance.class.php
> CourseDescription => /main/inc/lib/course_description.lib.php
> CourseDescription => /main/coursecopy/classes/CourseDescription.class.php
> Database => /main/inc/lib/database.mysqli.lib.php
> Database => /main/inc/lib/database.lib.php
> Model => /main/inc/lib/model.lib.php
> Model => /main/auth/shibboleth/lib/model.class.php
> Thematic => /main/inc/lib/thematic.lib.php
> Thematic => /main/coursecopy/classes/Thematic.class.php

I added too the path of known file in autoload - through a small script. So all known classes - outside of libraries that is - should autoload fine.

I am going to stop here for the moment as other changes would be too risky. We can make more changes after 1.9 is out.

As for Symph autoload component I agree it is a good idea but for now I don't believe we have a way to use it with so many differences in naming convention.

#7 - 23/04/2012 17:32 - Julio Montoya

Great contribution Laurent, It will require some testing, and checking that everything works fine. You're right that when we have to have first a better class naming pattern in order to use a Symphony component:

http://symfony.com/doc/current/components/class_loader.html

or something else

#8 - 23/04/2012 23:13 - Yannick Warnier

Just a few comments:

- Autoload is PHP 5.3 or superior, right ?(just so we know for the documentation on dependencies that have to be updated).

Laurent Opprecht wrote:

> - constants exists outside of the class. That is an issue because they may be called without going through the class. In this case the file is not autoloaded and we may hit an non existing const. Reason why I left those files the way they are. A better scheme is to including the const in the class and call them like that class_name::MY_CONST. That provides a namespace for the const and ensure they are loaded when needed.

OK. Would be better indeed (but more complicated for amateur developers - please always try to think about amateur developers, it is **very** important to us that Chamilo LMS's code does not become as complex as Chamilo LCMS Connect's code)

> - Some files includes if constructs. If class do not exists then declare it. This should go away with autoloading.

Should, indeed.

> - Some files are function apis when a class would be better - cas_login, cas_logout. Using functions prevent autoloading and create other issues as well.

I guess you meant "api functions" instead of "function apis"?

> - Some files mix declarations: class and functions

To be removed.

> - Lack of naming convention - which make difficult, well impossible in some circumstances, to locate the file containing the class

There are coding conventions to be respected, but the classes names have been admitted as an exception and can be named AsCamelCase or with_under_scores. In my opinion we should try and see what Symfony does with classes and use the same naming convention. However, for methods names, I would still recommend using underscores in lowercase as it is more easily understandable for non-English speakers.

> - the file names go on line of myclassname which is difficult to read, better schemes would be my_class_name, myClassName, MyClassName, whatever.

Agreed.

> - some classes have duplicate names (I left them out of autoloading)
>
> Attendance => /main/inc/lib/attendance.lib.php
> Attendance => /main/coursecopy/classes/Attendance.class.php

Should be renamed CopyAttendance, but as far as I know there are other classes of the same type, so rename them all at once (beware that this should not prevent the restauration of courses that have been archived in previous versions of Chamilo!)

> CourseDescription => /main/inc/lib/course_description.lib.php
> CourseDescription => /main/coursecopy/classes/CourseDescription.class.php

Same here: CopyCourseDescription for the second one

> Database => /main/inc/lib/database.mysqli.lib.php
> Database => /main/inc/lib/database.lib.php

OK, this is a non-problem as the database.mysqli.lib.php is experimental and is only provided as a drop-in replacement of the database.lib.php class (which is bad, I know, but it's just a test, it doesn't work so well for now).

> Model => /main/inc/lib/model.lib.php
> Model => /main/auth/shibboleth/lib/model.class.php

Second one to be renamed ShibbolethModel. I guess that's clear for everyone.

> Thematic => /main/inc/lib/thematic.lib.php
> Thematic => /main/coursecopy/classes/Thematic.class.php

Second one to be renamed to CopyThematic

> I added too the path of known file in autoload - through a small script. So all known classes - outside of libraries that is - should autoload fine.

Great. Performances will probably be greatly enhanced through that.

> I am going to stop here for the moment as other changes would be too risky. We can make more changes after 1.9 is out.

Agreed.

> As for Symph autoload component I agree it is a good idea but for now I don't believe we have a way to use it with so many differences in naming convention.

Agreed.

#9 - 24/04/2012 08:56 - Laurent Opprecht

To be more precise I am using spl_autoload_register for autoloading which is now the recommanded way of doing things - this allows to have several autoload functions at the same time. For example one that comes with pear another for chamilo, etc. This is for PHP 5 >= 5.1.2.

Ah yes "api functions" indeed :-)

#10 - 24/04/2012 11:54 - Julio Montoya

Talking about PHP versions the twig library needs PHP version 5.2.4
http://twig.sensiolabs.org/doc/intro.html

I created a new constant in order to change/setup this variable easily.
Constant MIN_PHP_VERSION added in main_api.lib.php

http://code.google.com/p/chamilo/source/detail?r=44878d3be35bbb6df858b8732006f1623ad5859c&repo=classic

#11 - 25/04/2012 08:10 - Yannick Warnier

- Target version set to 1.9.2

Setting to 1.9.1 for the second round. Good job (although we fixed many links to sortabletable.class.php which became sortable_table.class.php, not sure who did this, but he should have done it with more attention :-)).

#12 - 27/04/2012 11:28 - Laurent Opprecht

May well have been be. Sorry for that. Now one issue may comes from cAsing as the autoloader is case sensitive - as arrays are. So sortabletable will not autoload when SortableTabe will.

I noticed that classes are sometimes called with all lower cases. In this case the preffered solution is to replace the class name with the proper casing so that autoload works. Note that adding require_once will work as well. Just a bit simpler to relay on autoad in my point of view.

#13 - 07/05/2012 21:55 - Yannick Warnier

- File XHProf-Hierarchical-Profiler-Report.html added

I have just tested xhprof on Chamilo (following http://techportal.ibuildings.com/2009/12/01/profiling-with-xhprof/ and modifying a few stuff from there because it didn't work as is - like the inclusion of the xhprof lib in header.php has to be modified and the libs directories should be added to the Chamilo sources somewhere).

This shows that Chamilo has a total memory usage (for a user_portal.php script with two courses) of between 43M and 17M; 8M to 13M of which are used only by Twig.

In the details of what global.inc.php loads (8MB in total, well spread), there is only one odd library: document.lib.php, which uses by itself 1.3M (more than the double of any other).

I think that by just removing the quota calculation at the beginning and using it in an autoload way, we could at least save 1M in most of Chamilo pages.
Surprisingly, document.lib.php is loaded through something in messages.lib.php, apparently.

xhprof looks like a very good tool to profile, mostly for the table reports it generates, which include memory usage and allow you to make diffs between two different loads of the same page (one before and one after a change that could reduce memory usage considerably).

This being said, I'm horrified by the memory usage of Twig. I hope this is justified by the features it brings (I think it just doubled memory consumption of Chamilo - but maybe Smarty did so before, too). Technically it is likely that it could be beneficial overall because it caches things, but still... I'm a bit shocked right now (the xhprof output is pretty much exclusively showing Twig libs).

Attaching HTML results of a second load...

#14 - 08/05/2012 12:10 - Julio Montoya

Talking about Twig:

I'm adding some options in template.lib.php in order to use "Twig compilation cache" see:

http://code.google.com/p/chamilo/source/detail?r=f2f51c52ca751db7f16c7469fa4b45ad87c35d2e&repo=classic

That might be the reason why all Twig libraries are loaded every time ...

> Notice that the second argument of the loader is a compilation cache directory, where Twig caches the compiled templates to avoid the parsing phase for sub-sequent requests. It is very different from the cache you might want to add for the evaluated templates. For such a need, you can use any available PHP cache library.

http://twig.sensiolabs.org/doc/api.html#environment-options

Compilation Cache:
All template loaders can cache the compiled templates on the filesystem for future reuse. It speeds up Twig a lot as templates are only compiled once; and the performance boost is even larger if you use a PHP accelerator such as APC. See the cache and auto_reload options of Twig_Environment above for more information.

#15 - 08/05/2012 17:21 - Yannick Warnier

Julio Montoya wrote:

> All template loaders can cache the compiled templates on the filesystem for future reuse.

OK, please make sure template cache is stored in the archives/ directory, for example archives/cache/twig/ (in the future, the tendency will be to upload **all** temporary files into the archives directory, this will make it much more likely for us to be able to make a Debian package in the future).

#16 - 25/05/2012 16:50 - Julio Montoya

- Status changed from New to Needs more info

- % Done changed from 0 to 80

archive/twig folder added

#17 - 02/08/2012 00:52 - Yannick Warnier

- Target version changed from 1.9.2 to 2.0

#18 - 17/01/2013 13:21 - Julio Montoya

see #5523 we're using composer to generate the autoload class no need to create a custom autoload class

#19 - 02/04/2013 15:22 - Julio Montoya

- Status changed from Needs more info to Feature implemented

- % Done changed from 80 to 100

Files

global.inc.png	304 KB	20/04/2012	Laurent Opprecht
callgrind.out.1334921687.4616	337 KB	20/04/2012	Laurent Opprecht
XHProf-Hierarchical-Profiler-Report.html	72.7 KB	07/05/2012	Yannick Warnier