TYPO3 Core - Bug #66482

Extbase persistence layer fails to create empty objects in PHP 5.6.

2015-04-20 00:15 - Klaus Bitto

Status:	Closed	**Start date:**	2015-04-19
Priority:	Must have	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Extbase	**Estimated time:**	0.00 hour
Target version:	6.2.16		
TYPO3 Version:	6.2	**Complexity:**	easy
PHP Version:	5.6	**Is Regression:**	No
Tags:		**Sprint Focus:**	Stabilization Sprint

Description

The extbase persistence layer (actually the object container) creates empty objects, by-passing the constructor, by unserializing a faked serialized empty object.
This no longer works in PHP 5.6. The unserialize() call fails fatally ("Erroneous data format").

http://php.net/manual/en/function.unserialize.php:

```
Changelog
---------
Version    Description
5.6.0      *Manipulating the serialised data by replacing C: with O: to force object instantiation
 without calling the constructor will now fail.*
```

Instead of unserialize(), public object ReflectionClass::newInstanceWithoutConstructor (void) must be used:
http://php.net/manual/de/reflectionclass.newinstancewithoutconstructor.php (PHP >= 5.4)

TYPO3 6.2 was declared to be compatible with PHP <= 5.5, until 2 days ago, when I reported https://forge.typo3.org/issues/66468. Now, TYPO3 6.2 is officially declared as compatible to PHP 5.3.7-5.6.x (http://typo3.org/download/). Therefore I regard this a high priority issue, as extbase is factually broken on PHP 5.6.

The error occurs in this line:
https://github.com/TYPO3/TYPO3.CMS/blob/TYPO3_6-2/typo3/sysext/extbase/Classes/Object/Container/Container.php#L121

PHP 5.3 will still need the unserialize() call. PHP 5.4-5.6 can use ReflectionClass::newInstanceWithoutConstructor().

See also how Doctrine2 fixed the same issue:
https://github.com/marmotz/doctrine2/commit/93c276d059b40b0783ba9a24549a8b135e257693

Related issues:

Related to TYPO3 Core - Bug #66473: Cannot create object implementing Seriali...	**Closed**	**2015-04-17**
Related to TYPO3 Core - Bug #70873: Fix inconsistency of the calling order of...	**Rejected**	**2015-10-20**
Related to TYPO3 Core - Feature #70874: ClassInfo::getIsInitializeable() is n...	**Closed**	**2015-10-20**

History

#1 - 2015-04-20 00:19 - Klaus Bitto

This is "half a duplicate" of https://forge.typo3.org/issues/66473.

Note that Bug #66473 targets TYPO3 7, but mentions "This also needs to fixed for TYPO3 6.2 though which supports PHP 5.3, in which case checking for the Serializable interface and using C instead of O could work".

Should a separate Bug targeting TYPO3 6.2 be kept?
Could "doctrine/instantiator" be introduced even in a 6.2.x release?

A Doctrine2-style fix like https://github.com/marmotz/doctrine2/commit/93c276d059b40b0783ba9a24549a8b135e257693 should certainly be possible in the next 6.2 patchlevel.

#2 - 2015-06-22 11:42 - Gerrit Code Review

- Status changed from New to Under Review

Patch set 1 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at http://review.typo3.org/40522

#3 - 2015-07-24 17:03 - Christian Kuhn

- Description updated

- Target version changed from next-patchlevel to 6.2.15

- Sprint Focus set to Stabilization Sprint

#4 - 2015-09-08 13:44 - Alexander Opitz

- Target version changed from 6.2.15 to 6.2.16

#5 - 2015-10-18 19:36 - Gerrit Code Review

Patch set 2 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at http://review.typo3.org/40522

#6 - 2015-10-18 19:39 - Gerrit Code Review

Patch set 3 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at http://review.typo3.org/40522

#7 - 2015-10-20 13:12 - Klaus Bitto

> http://review.typo3.org/40522

In getInstanceInternal(), first injectDependencies() is called, then initializeObject().

Doing it the other way around in getEmptyObject() would mean introducing a new bug. :) (initializeObject() might use injected objects.)

Also, there might be code depending on the fact that initializeObject is NOT called when reconstituting an object from DB. Therefore, adding initializeObject() here is a breaking change for some, while adding newInstanceWithoutConstructor() (ideally with the Doctrine workaround for final classes etc.) is only a bugfix without negative side effects.

Regarding if($classInfo->getIsInitializeable() && is_callable(array($object, 'initializeObject')): Shouldn't getIsInitializeable() have to be enough? Or in other words: If an is_callable() check is needed, isn't getIsInitializable() broken? I do see that one works on the class and the other works on the object, but the current solution still seems wrong by design.

#8 - 2015-10-20 14:59 - Gerrit Code Review

Patch set 4 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at https://review.typo3.org/40522

#9 - 2015-10-20 15:21 - Gerrit Code Review

Patch set 5 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at https://review.typo3.org/40522

#10 - 2015-10-20 15:31 - Gerrit Code Review

Patch set 6 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at https://review.typo3.org/40522

#11 - 2015-10-20 19:15 - Gerrit Code Review

Patch set 7 for branch **TYPO3_6-2** of project **Packages/TYPO3.CMS** has been pushed to the review server.
It is available at https://review.typo3.org/40522

#12 - 2015-10-20 22:47 - Helmut Hummel

- Status changed from Under Review to Needs Feedback

Isn't this a duplicate of #66473 ? We should close this one or the other, I'd opt to close this one, as the other has more information gathered.

#13 - 2015-10-20 22:51 - Helmut Hummel

Klaus B wrote:

> See also how Doctrine2 fixed the same issue: https://github.com/marmotz/doctrine2/commit/93c276d059b40b0783ba9a24549a8b135e257693

This fix isn't part of doctrine any more. Now the doctrine/instantiator code is used:

https://github.com/doctrine/doctrine2/blob/master/lib/Doctrine/ORM/Mapping/ClassMetadataInfo.php#L912

There is imho no way around that!

#14 - 2015-10-21 14:44 - Klaus Bitto

Helmut Hummel wrote:

> Now the doctrine/instantiator code is used:
>
> https://github.com/doctrine/doctrine2/blob/master/lib/Doctrine/ORM/Mapping/ClassMetadataInfo.php#L912
>
> There is imho no way around that!

I guess you are right.

For me, you can close this issue in favour of #66473.
I only find it very important that this makes it into the 6.2 LTS, since you cannot keep people using PHP < 5.6 as long as they run the LTS.

#15 - 2015-10-27 16:05 - Oliver Hader

- Status changed from Needs Feedback to Closed