TYPO3.Flow - Bug #35136

Problem with validating simple types

2012-03-22 12:19 - Rens Admiraal

Status:	Resolved	**Start date:**	2012-03-22
Priority:	Should have	**Due date:**	
Assignee:	Bastian Waidelich	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
PHP Version:		**Complexity:**	
Has patch:	No		

Description

When an argument of a controller action is annotated with @param integer $var, the validation never fails if a string is passed to the argument.

This is because the TypeConverter returns (integer) $value, which will always match on the Validator. This could for example be solved with something like this in the TypeCoverter:

```
        if (!is_numeric($source)) {
            throw new \TYPO3\FLOW3\Exception('Can\'t convert source value to integer', 1332411849)
;
        }
```

Problem with this approach is the output: an unfriendly error.

Maybe it would be possible to do a quick validation on simple types before the actual TypeConverter starts doing his job?

Associated revisions

Revision a9156475 - 2012-05-29 16:04 - Bastian Waidelich

[!!!][BUGFIX] Float and Integer converters do not correctly handle errors and empty values

When an argument of a controller action is annotated with
@param integer $var, the validation never fails if a string
is passed to the argument.
This is because the IntegerConverter implicitly casts the
string to an integer. The same problem exists with floats.

This change fixes this by checking the value and returning
an error object if it is not numeric. Besides this tweaks
the FloatConverter to accept float and integer values as
input and it adds a bunch of unit & functional tests.

Furthermore, empty strings are now correctly converted as NULL values.
This is a breaking change if you relied upon the old behavior that
empty values are converted to the number 0.

Change-Id: I178f616d0dd4acea90938384fb21600dd0f7c252
Fixes: #35136
Releases: 1.1, 1.2

Revision 5badcdb0 - 2012-05-30 09:59 - Sebastian Kurfuerst

[Bugfix] Property mapper should distinguish between error and NULL

currently, when a nested type converter returns NULL, this is silently
caught by the property mapper. Thus, it is not possible to reset f.e.
an integer value to NULL again.

This change fixes that behavior, and adds a functional test for it.

Related: #35136
Releases: 1.1, 1.2
Change-Id: Ibc51f4066d09d084e02a67696cb3d6bff98a6451

Revision 996f20ce - 2012-05-30 10:37 - Bastian Waidelich

[!!!][BUGFIX] Float and Integer converters do not correctly handle errors and empty values

When an argument of a controller action is annotated with
@param integer $var, the validation never fails if a string
is passed to the argument.
This is because the IntegerConverter implicitly casts the
string to an integer. The same problem exists with floats.

This change fixes this by checking the value and returning
an error object if it is not numeric. Besides this tweaks
the FloatConverter to accept float and integer values as
input and it adds a bunch of unit & functional tests.

Furthermore, empty strings are now correctly converted as NULL values.
This is a breaking change if you relied upon the old behavior that
empty values are converted to the number 0.

Change-Id: I178f616d0dd4acea90938384fb21600dd0f7c252
Fixes: #35136
Releases: 1.1, 1.2

Revision ca4ef91c - 2012-06-20 10:41 - Sebastian Kurfuerst

[BUGFIX] Property mapper should distinguish between error and NULL

Currently, when a nested type converter returns NULL, this is silently
caught by the property mapper. Thus, it is not possible to reset e.g.
an integer value to NULL again.

This change fixes that behavior, and adds a functional test for it.

Change-Id: Ibc51f4066d09d084e02a67696cb3d6bff98a6451
Related: #35136
Releases: 1.1, 1.2

History

#1 - 2012-03-25 11:33 - Christian Müller

I see what you mean but I think the Converter should just do it's job and don't validate anything, so throwing an exception is out of question for me.
Also this problem about first mapping then validating or the other way around would probably need to be decided on a case by case basis (objects clearly first need mapping) and then custom validators and mappers are really complex to configure as you would need to define the mapping/validation order too.
I think the current approach is fine for most cases, I would say the validation rule on this argument needs to be tightened to example a number range then it could still throw some validation error.

#2 - 2012-03-25 12:45 - Rens Admiraal

I'm totally fine with the conversion, that's not something to discuss I think. But validation always has to be done in a meaningful way. If we are converting types based on the @param tag AFTER the TypeConverter converts the value, then the validation is useless and will cause confusion and errors.

So either we should fix the validation, or we should clearly state the situations in which validation on simple types will not work (or maybe even totally skip validation on the @param annotation and require a @FLOW3\Validation rule...)

#3 - 2012-03-26 17:14 - Sebastian Kurfuerst

you are not allowed to throw an exception inside the type converter in case of a user error.

Instead, you should return an \TYPO3\FLOW3\Error\Error object which is then shown to the user.

However, you at least need to make sure that the empty value is always correctly passed through, else no empty values are allowed.

This change should only go in with a proper functional test.

Greets, Sebastian

#4 - 2012-03-28 12:28 - Bastian Waidelich

- Project changed from TYPO3 Flow Base Distribution to TYPO3.Flow

- Status changed from New to Accepted

#5 - 2012-03-28 13:01 - Bastian Waidelich

- Has patch set to No

Rens Admiraal wrote:

Hi Rens,

> This is because the TypeConverter returns (integer) $value [...]

That's not what I observe. A string of "foo" is simply passed on to the action instead of being casted to integer (which would result in 0).

#6 - 2012-03-28 13:05 - Bastian Waidelich

Bastian Waidelich wrote:

> That's not what I observe.

Not true, forget my comment. Caching issue **doh** ;)

#7 - 2012-03-28 19:59 - Gerrit Code Review

- Status changed from Accepted to Under Review

Patch set 1 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/10071

#8 - 2012-04-19 11:04 - Gerrit Code Review

Patch set 2 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/10071

#9 - 2012-04-19 11:25 - Gerrit Code Review

Patch set 3 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/10071

#10 - 2012-05-29 14:28 - Gerrit Code Review

Patch set 4 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/10071

#11 - 2012-05-29 16:04 - Gerrit Code Review

Patch set 5 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/10071

#12 - 2012-05-30 10:37 - Bastian Waidelich

- Status changed from Under Review to Resolved

- % Done changed from 0 to 100

Applied in changeset a915647549a52621d1b23b7787f0bd2e03a91261.

#13 - 2012-05-30 10:37 - Gerrit Code Review

- Status changed from Resolved to Under Review

Patch set 1 for branch **FLOW3-1.1** has been pushed to the review server.
It is available at http://review.typo3.org/11719

#14 - 2012-05-30 12:38 - Bastian Waidelich

- Status changed from Under Review to Resolved

Applied in changeset 996f20ce3a61fd4c9f4645d64df44235246b79bf.