TYPO3.Flow - Feature #51570

Unpersisted changes in Safe Requests should throw an Exception

2013-08-30 12:03 - Marc Neuhaus

Status:	Resolved	**Start date:**	2013-08-30
Priority:	Should have	**Due date:**	
Assignee:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			
PHP Version:		**Complexity:**	
Has patch:	No		

Description

With the recent change to not trigger persistAll in safe request methods (GET/HEAD) some users stumble over not getting their entities saved because of this. Altough it's written to the log and mentioned in the docs i think we should also throw an exception telling the developer about this if the following conditions are met:

- the current request method is a safe request method (GET/HEAD)
- the current context is the development context
- the user added/updated/removed an entity using a Repository or PeristenceManager
- the user **did not** trigger persistAll using the PersistenceManager on it's own

Related issues:

Related to TYPO3.Flow - Feature #47951: Warn if persistence stack is not empt...	**New**	**2013-05-05**

Associated revisions

Revision e9b5de3f - 2014-03-06 21:34 - Marc Neuhaus

[!!!][FEATURE] Throw exception for unpersisted changes in Safe Requests

This changeset keeps track if the PersistenceManager has unpersisted changes
and notifies the Developer with a helpful exception if this happens in a
safe request (GET/HEAD). This exception is only thrown in development context

In case you implemented your own Persistence Manager, you must add the new
hasUnpersistedChanges() method, unless you extend the AbstractPersistenceManager.

Releases: master
Resolves: #51570
Change-Id: I52cd0b5d650327daa4507eb197f36da50a75daa7

Revision ad0c7178 - 2014-11-17 10:57 - Alexander Stehlik

[!!!][BUGFIX] Skip automatic persistence for updated entities

When trying to persist changes in a "safe request" (e.g. ``GET``) Flow
throws an exception::

```
Detected modified or new objects [...] to be persisted which is not
  allowed for "safe requests"
```

including details on how to work around this.

This currently only works if entities have been **added** or **removed**.

With this change also **updates** to entities are tracked correctly so
that automatic persistence is skipped for modified objects for safe
requests.

This is a breaking change when code relied on the incorrect behavior of
automatically persisting changes even for safe requests.
In this case make sure to trigger updates only via unsafe requests (e.g.
``POST`` or ``PUT``).
If that's not an option, the issue can be worked around with a manual
call to ``PersistenceManager::persistAll()``.

Change-Id: I9eb781c54c608c867a45206f64b6baf98c3d07f2
Releases: master, 2.2, 2.1
Fixes: FLOW-84
Related: #47252
Related: #51570

Revision bfffa3c5 - 2015-04-08 12:38 - Alexander Stehlik

[!!!][BUGFIX] Skip automatic persistence for updated entities

When trying to persist changes in a "safe request" (e.g. ``GET``) Flow
throws an exception::

```
Detected modified or new objects [...] to be persisted which is not
  allowed for "safe requests"
```

including details on how to work around this.

This currently only works if entities have been **added** or **removed**.

With this change also **updates** to entities are tracked correctly so
that automatic persistence is skipped for modified objects for safe
requests.

This is a breaking change when code relied on the incorrect behavior of
automatically persisting changes even for safe requests.
In this case make sure to trigger updates only via unsafe requests (e.g.
``POST`` or ``PUT``).
If that's not an option, the issue can be worked around with a manual
call to ``PersistenceManager::persistAll()``.

Change-Id: I9eb781c54c608c867a45206f64b6baf98c3d07f2
Releases: master, 2.3, 2.2
Fixes: FLOW-84
Related: #47252
Related: #51570

Revision a0b36f97 - 2015-04-26 19:01 - Alexander Stehlik

[!!!][BUGFIX] Skip automatic persistence for updated entities

When trying to persist changes in a "safe request" (e.g. ``GET``) Flow
throws an exception::

```
Detected modified or new objects [...] to be persisted which is not
  allowed for "safe requests"
```

including details on how to work around this.

This currently only works if entities have been **added** or **removed**.

With this change also **updates** to entities are tracked correctly so
that automatic persistence is skipped for modified objects for safe
requests.

This is a breaking change when code relied on the incorrect behavior of
automatically persisting changes even for safe requests.
In this case make sure to trigger updates only via unsafe requests (e.g.
``POST`` or ``PUT``).
If that's not an option, the issue can be worked around with a manual
call to ``PersistenceManager::persistAll()``.

Change-Id: I9eb781c54c608c867a45206f64b6baf98c3d07f2
Releases: master, 2.3, 2.2
Fixes: FLOW-84
Related: #47252
Related: #51570

History

#1 - 2013-08-30 12:04 - Gerrit Code Review

- Status changed from New to Under Review

Patch set 2 for branch **master** has been pushed to the review server.

It is available at https://review.typo3.org/23484

#2 - 2013-08-30 12:15 - Gerrit Code Review

Patch set 3 for branch **master** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#3 - 2013-08-30 14:10 - Gerrit Code Review

Patch set 4 for branch **master** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#4 - 2013-11-06 21:56 - Gerrit Code Review

Patch set 5 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#5 - 2013-11-07 10:53 - Gerrit Code Review

Patch set 6 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#6 - 2013-11-10 00:17 - Gerrit Code Review

Patch set 7 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#7 - 2013-11-10 00:34 - Gerrit Code Review

Patch set 8 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#8 - 2013-11-10 01:05 - Gerrit Code Review

Patch set 9 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#9 - 2014-02-21 01:16 - Gerrit Code Review

Patch set 10 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#10 - 2014-02-28 11:25 - Gerrit Code Review

Patch set 11 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#11 - 2014-03-06 15:12 - Gerrit Code Review

Patch set 12 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#12 - 2014-03-06 21:29 - Gerrit Code Review

Patch set 13 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#13 - 2014-03-06 21:42 - Gerrit Code Review

Patch set 14 for branch **master** of project **Packages/TYPO3.Flow** has been pushed to the review server.
It is available at https://review.typo3.org/23484

#14 - 2014-03-06 22:36 - Anonymous

- Status changed from Under Review to Resolved

- % Done changed from 0 to 100

Applied in changeset e9b5de3f8708ce136c2a08d07e173f953013ecce.

#15 - 2014-03-13 11:20 - Marco Falkenberg

Could you please NOT throw an exception and just log it?! Isn't it a little bit overdone accepting a breaking change just for a developer message?

What about if you wan't to persist the missing entities manually after the controller invocation? In my case I persist user messages entities, which I

want to show sometimes also for safe requests.

#16 - 2014-03-13 11:54 - Marc Neuhaus

Hey Marco :)

The Exception is only thrown if all these conditions are met:

- you are running inside the Development Context
- you added Entities to the PersistenceManger
- the request is a safe request which, by default does not persist anything that's added to the PersistenceManager
- you did **not** trigger persistenceManager->persistAll manually after adding the entities

We had several people quite confused because there changes weren't persisted in safe requests. The only case where this could be considered "breaking" is where you might have a bug because the changes weren't persisted because of the safe request.

We choose against a log entry because this would have destroyed the whole point of making novices aware of the safe request behavior.

If i understand you correctly you have a safe request method, were you change/add entities and persist manually? then this exception should not be thrown. do you use some kind of customer peristenceManager?

Cheers
Marc

#17 - 2014-03-28 16:42 - Marco Falkenberg

Hey Marc. Thanks for your answer. No I don't have a custom persitence manager. The only thing i did was to persist particular entities (Notifications) after every controller invocation. To realize this behaviour I hooked into the slot AfterControllerInvocation which fits very well.

```
$dispatcher->connect('TYPO3\Flow\Mvc\Dispatcher', 'afterControllerInvocation', function($request) use($bootstrap) {
$myRepo->persistEntities();
});
```

But Flow now also uses this slot to throw the discussed exception.

#18 - 2014-03-28 17:34 - Marc Neuhaus

ah, i see. yes that could fail then of course. usually the manual persisting is done right were you did the change in cases like that.

But Robert introduced another thing quite recently: "Whitelisted objects"

https://review.typo3.org/#/c/28792/

http://docs.typo3.org/flow/TYPO3FlowDocumentation/latest/TheDefinitiveGuide/PartIII/Persistence.html#whitelisted-objects

Maybe that could help you?

#19 - 2014-03-31 11:52 - Bastian Waidelich

Marco Falkenberg wrote:

Hi Marco ;)

Just a little comment regarding:

> ```
> $dispatcher->connect('TYPO3\Flow\Mvc\Dispatcher', 'afterControllerInvocation', function($request) use($bootstrap) {
> $myRepo->persistEntities();
> });
> ```

I guess you're aware of this, but for anyone else who might just copy that:
With this you circumvent our CSRF-protection. That means: You could change the server state (update, add, delete data) just by clicking a link (in an email for example) if you have the required privileges (e.g. you are logged in to the flow app).
That's because the CSRF token is only checked for "unsafe" requests.

While this might be perfectly fine in your case it's worth mentioning that there are ways around this:

For example you could check the CSRF-token explicitly for the affected actions (and make sure any link contains the token..).
Or you could encapsulate the functionality (in this case: notifications) in some service that persists changes immediately.
Or create those entities asynchronously via AJAX.

It depends on the use case but I usually prefer the last solution because it follows the HTTP spec and it allows for caching(!) - for example if you want

to increase a view-counter on each view of a certain article

#20 - 2014-04-10 09:49 - Marco Falkenberg

Marc Neuhaus wrote:

> ah, i see. yes that could fail then of course. usually the manual persisting is done right were you did the change
> in cases like that.
>
> But Robert introduced another thing quite recently: "Whitelisted objects"
>
> https://review.typo3.org/#/c/28792/
>
> http://docs.typo3.org/flow/TYPO3FlowDocumentation/latest/TheDefinitiveGuide/PartIII/Persistence.html#whitelisted-objects
>
> Maybe that could help you?

That's a clean (white) solution and fits perfect to me needs. Thanks Robert!

#21 - 2014-04-10 10:11 - Marco Falkenberg

Bastian Waidelich wrote:

> Marco Falkenberg wrote:
>
> Hi Marco ;)
>
> Just a little comment regarding:
>
> $dispatcher->connect('TYPO3\Flow\Mvc\Dispatcher', 'afterControllerInvocation', function($request) use($bootstrap) {
> $myRepo->persistEntities();
> });
>
> I guess you're aware of this, but for anyone else who might just copy that:
> With this you circumvent our CSRF-protection. That means: You could change the server state (update, add, delete data) just by clicking a link (in an email for example) if you have the required privileges (e.g. you are logged in to the flow app).
> That's because the CSRF token is only checked for "unsafe" requests.
>
> While this might be perfectly fine in your case it's worth mentioning that there are ways around this:
>
> For example you could check the CSRF-token explicitly for the affected actions (and make sure any link contains the token..).
> Or you could encapsulate the functionality (in this case: notifications) in some service that persists changes immediately.
> Or create those entities asynchronously via AJAX.
>
> It depends on the use case but I usually prefer the last solution because it follows the HTTP spec and it allows for caching(!) - for example if you want to increase a view-counter on each view of a certain article

Hi Bastian,

many thanks for this hints and enforcing me to think about more often to follow the HTTP specs :)

Of course I thought about not persisting just everything. While ago I found something like this:

```
class NotificationRepository {
  ...

  public function persistEntities() {
    foreach ($this->entityManager->getUnitOfWork()->getIdentityMap() as $className => $entities) {
      if ($className === $this->entityClassName) {
        foreach ($entities as $entityToPersist) {
          $this->entityManager->flush($entityToPersist);
        }
        return;
      }
    }
  }
}
```

Please correct if I'm wrong. If I call NotificationRepository::persistEntities only entities of this certain type will be persisted? That's what I tried in code above. But you're right: at least for this entities the CSRF-protection is circumvented.

FYI: I tried the second solution (persisting immediately) with the mentioned special repository function. But then some changes for other objects get lost. After some debugging I could see why. Look at the last lines of \Doctrine\ORM\UnitOfWork::commit.

In the end I used the new introduced whitelist :)

Thanks anyway!

#22 - 2014-04-10 10:18 - Bastian Waidelich

Marco Falkenberg wrote:

> many thanks for this hints and enforcing me to think about more often to follow the HTTP specs :)

You're welcome ;)

> If I call NotificationRepository::persistEntities only entities of this certain type will be persisted?

Correct, probably including the dependencies (for example product.category).

> FYI: I tried the second solution (persisting immediately) with the mentioned special repository function. But then some changes for other objects get lost.
> After some debugging I could see why. Look at the last lines of \Doctrine\ORM\UnitOfWork::commit.

Uh.. ok..

> In the end I used the new introduced whitelist :)

Good choice ;)