TYPO3.Flow - Bug # 46210

Status:	Needs Feedback	**Priority:**	Should have
Author:	Benno Weinzierl	**Category:**	Session
Created:	2013-03-12	**Assigned To:**	
Updated:	2013-03-27	**Due date:**	

PHP Version:	
Has patch:	No
Complexity:	
Affected Flow version:	Git master
Subject:	securityContext->getParty() in the initializeObject() method of a session-Scope object throws exception on second request

Description

When i put the following code in a session-scope object and am logged in it works on the first request. But after reloading the same page this code throws an exception.

```
        /**
      * @Flow\Inject
      * @var \TYPO3\Flow\Security\Context
      */
     protected $securityContext;

     public function initializeObject() {
        $loginUser = $this->securityContext->getParty();
     }
```

The exceoption thrown:

```
#1358513802: The security Context cannot be initialized yet. Please check if it can be initialized with
$securityContext->canBeInitialized() before trying to do so.
```

Related issues:

related to TYPO3.Flow - Bug # 46424: Infinite recursive call in DependencyProxy	**Resolved**	**2013-03-19**

History

#1 - 2013-03-27 08:15 - Sebastian Kurfuerst

- Status changed from New to Needs Feedback

Hey Benno,

in order to properly debug your problem, I need some more input, more namely about what kind of session-scoped object it is, and where it is needed from:
Is it a dependency of a controller, of a model class, of a Route Part Handler, ...?

Greets,
Sebastian

#2 - 2013-03-27 08:37 - Sebastian Kurfuerst

The Security Context is only initialized inside the MVC Dispatcher, as it needs the Web Request.

Thus, you should try to initialize the $loginUser lazily, when you try to access it for the first time. I think this would already solve the issue.

Greets, Sebastian

#3 - 2013-03-27 10:44 - Adrian Föder

Maybe I even ran into a similar issue, where in my case a Widget gets woken up, leading into trouble with the DependencyProxy.
In \TYPO3\Flow\Security\Aspect\PersistenceQueryRewritingAspect::Flow_Proxy_injectProperties, the securityContext is created the following way:

```
$this->securityContext = \TYPO3\Flow\Core\Bootstrap::$staticObjectManager->getLazyDependencyByHash('48836470c14129ade5f39e28c4816673', $securityContext_reference);
if ($this->securityContext === NULL) {
    $this->securityContext = \TYPO3\Flow\Core\Bootstrap::$staticObjectManager->createLazyDependency('48836470c14129ade5f39e28c4816673',  $securityContext_reference, 'TYPO3\Flow\Security\Context', function() { return \TYPO3\Flow\Core\Bootstrap::$staticObjectManager->get('TYPO3\Flow\Security\Context'); });
}
```

but it looks like the closure's return again returns a Dependency Proxy.