TYPO3.Flow - Feature #47404

Add getters and setters methods for introduced properties

2013-04-20 17:27 - Rafael Kähm

Status:	New	**Start date:**	2013-04-20
Priority:	Could have	**Due date:**	
Assignee:		**% Done:**	0%
Category:	AOP	**Estimated time:**	0.00 hour
Target version:			
PHP Version:		**Complexity:**	medium
Has patch:	No		

Description

Currently you must introduce methods over interface introduction and then wrap introduced methods with around advice.

Like this:

```
<?php
namespace Acme\Test\Aspect;

use TYPO3\Flow\Annotations as Flow;

use TYPO3\Flow\Reflection\ObjectAccess;

/**
 * Inject attribute to every as "MyOwnTypeOfEntity" annotated entity
 *
 * @Flow\Scope("singleton")
 * @Flow\Aspect
 * @Flow\Introduce("Acme\Test\Aspect\PropertyIntroductionAspect->isMyOwnTypeOfEntity", interfaceNa
me="Acme\Test\Aspect\PropertyIntroductionInterface")
 */
class PropertyIntroductionAspect {

    /**
     * @Flow\Inject
     * @var \TYPO3\Flow\Reflection\ReflectionService
     */
    protected $reflectionService;

    /**
     * @Flow\Pointcut("TYPO3\Flow\Persistence\Aspect\PersistenceMagicAspect->isEntityOrValueObject
 && classAnnotatedWith(Acme\Test\Annotations\MyOwnTypeOfEntity)")
     */
    public function isMyOwnTypeOfEntity() {}

    /**
     * @var string
     * @Flow\Introduce("Acme\Test\Aspect\PropertyIntroductionAspect->isMyOwnTypeOfEntity")
     */
    protected $introducedProp;

    /**
     * Around advice, implements the new method "newMethod" of the
     * "NewInterface" interface
     *
     * @param  \TYPO3\Flow\AOP\JoinPointInterface $joinPoint The current join point
     * @return void
     * @Flow\Around("Acme\Test\Aspect\PropertyIntroductionAspect->isMyOwnTypeOfEntity && method(.*
->setIntroducedProp())")
     */
```

```
    public function setIntroducedPropImplementation(\TYPO3\Flow\AOP\JoinPointInterface $joinPoint)
 {
         $introducedProp = $joinPoint->getMethodArgument('introducedProp');
        $proxy = $joinPoint->getProxy();

        ObjectAccess::setProperty($proxy, 'introducedProp', $introducedProp, TRUE);

        $someResult = $joinPoint->getAdviceChain()->proceed($joinPoint);
    }

}
```

Acme\Test\Aspect\PropertyIntroductionInterface

```
<?php
namespace Acme\Test\Aspect;

/**
 * Inject community-attribute to every class
 *
 */
interface PropertyIntroductionInterface {

    /**
     * @param string $introducedProp some string
     * @return void
     */
    public function setIntroducedProp($introducedProp);

}
```

then you can see following code in (..)/Data/Temporary/Development/Cache/Code/Flow_Object_Classes/Acme_Test_Domain_Model_Model.php

```
...
    /**
     * Autogenerated Proxy Method
     */
     public function setIntroducedProp($introducedProp) {


// FIXME this can be removed again once Doctrine is fixed (see fixMethodsAndAdvicesArrayForDoctrin
eProxiesCode())
            $this->Flow_Aop_Proxy_fixMethodsAndAdvicesArrayForDoctrineProxies();
        if (isset($this->Flow_Aop_Proxy_methodIsInAdviceMode['setIntroducedProp'])) {
        $result = NULL;

        } else {
            $this->Flow_Aop_Proxy_methodIsInAdviceMode['setIntroducedProp'] = TRUE;
            try {

                    $methodArguments = array();

                $methodArguments['introducedProp'] = $introducedProp;

                    $adviceChains = $this->Flow_Aop_Proxy_getAdviceChains('setIntroducedProp');
                    $adviceChain = $adviceChains['TYPO3\Flow\Aop\Advice\AroundAdvice'];
                    $adviceChain->rewind();
                    $joinPoint = new \TYPO3\Flow\Aop\JoinPoint($this,
'Acme\Test\Domain\Model\Model', 'setIntroducedProp', $methodArguments, $adviceChain);
                    $result = $adviceChain->proceed($joinPoint);

            } catch (\Exception $e) {
                unset($this->Flow_Aop_Proxy_methodIsInAdviceMode['setIntroducedProp']);
                throw $e;
```

```
            }
            unset($this->Flow_Aop_Proxy_methodIsInAdviceMode['setIntroducedProp']);
        }
        return $result;
    }
...
```

as you can see there are to many steps and to many operations to add a simple getter and setter methods.

my suggestion:
add options to *TYPO3\Flow\Annotations\Introduce* :
addGetterIntroduction or **addSetterIntroduction** for manually method introductions
skipGetterIntroduction or **skipSetterIntroduction** for automatically method introductions

History

#1 - 2013-05-21 13:28 - Robert Lemke

- Target version deleted (2.1)