TYPO3.Form - Feature # 36722

Status:	Resolved	**Priority:**	Should have
Author:	Marc Neuhaus	**Category:**	
Created:	2012-04-30	**Assigned To:**	Bastian Waidelich
Updated:	2012-05-04	**Due date:**	
Subject:	Support identifiers with special characters		

Description

Currently it isn't possible to add internal Arguments like "__identity" because it isn't a valid name identifier.

$page1->createElement("__identity", "Foo.ContentManagement:Hidden")

In my opinion the most logical way would be a new Method $page1->addInternalArgument("__identity", $id); to add elements like that.

Related issues:

related to TYPO3.FormBuilder - Bug # 36823: Undefined class constant 'PATTERN...	**Resolved**	**2012-05-03**
related to TYPO3.Flow - Bug # 36988: PropertyMapper chokes on Objects that ar...	**Resolved**	**2012-12-25**

Associated revisions

Revision b90ec333 - 2012-05-02 18:34 - Bastian Waidelich

[FEATURE] Support identifiers with special characters

Currently it's not possible to add FormElements with non alpha
numeric characters in the identifier.
This is required in order to specify property paths (foo.bar.baz)
and to add elements that should map to internal arguments (__foo).

This change removes the strict regex check from the FormElement
constructor and instead makes sure that
FormElement::getUniqueIdentifier() returns an identifier without
special characters so it can be used as id attributes in forms for
example.

Change-Id: I99f8a9bf3ef13147e1baef655a6900a0761d085c
Resolves: #36722

History

#1 - 2012-05-02 17:52 - Bastian Waidelich

- Tracker changed from Bug to Feature

- Subject changed from Internal Arguments can't be added to Support identifiers with special characters

I just noticed a related issue: Dots are not allowed as identifiers either, but they're required for more advanced property mapping:

```
1// ...
2$page1 = $formDefinition->createPage('page1');
3
4$lastname = $page1->createElement('user.lastName', 'TYPO3.Form:SingleLineText');
5$lastname->setLabel('Last name');
```

```
6
7$firstname = $page1->createElement('user.firstName', 'TYPO3.Form:SingleLineText');
8$firstname->setLabel('Last name');
9
10$formDefinition->getProcessingRule('user')->setDataType('Some\Domain\User');
11$formDefinition->getProcessingRule('user')->getPropertyMappingConfiguration()->setTypeConverterOption('TYPO3\FLOW3\Property\T
ypeConverter\PersistentObjectConverter', \TYPO3\FLOW3\Property\TypeConverter\PersistentObjectConverter::
CONFIGURATION_CREATION_ALLOWED, TRUE);
12// ...
```

#2 - 2012-05-02 17:54 - Marc Neuhaus

Agreed, i've noticed that and will probably take that on when i get to the inline editing of child records. :)

#3 - 2012-05-02 18:02 - Bastian Waidelich

- Status changed from New to Accepted

- Assigned To set to Bastian Waidelich

#4 - 2012-05-02 18:34 - Gerrit Code Review

Patch set 1 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/10936

#5 - 2012-05-02 19:06 - Bastian Waidelich

- Status changed from Accepted to Under Review

#6 - 2012-05-02 23:38 - Marc Neuhaus

I just noticed, that the patch above only solves part of the problem with advanced property mappings.

Example:

The TextfieldViewHelper gets the property "item.title" set and tries to render that field by using ObjectAccess::getPropertyPath from the FormRuntime.
This invokes the getElementValue of the Runtime through the offsetExists function which fails, because it doesn't return the variable item.

How to Reproduce:
Create a form with a property like "item.test" and set a default value. The default value won't reach the textfield.

#7 - 2012-05-03 12:58 - Bastian Waidelich

- Status changed from Under Review to Resolved

Marc Neuhaus wrote:

> *I just noticed, that the patch above only solves part of the problem with advanced property mappings.*

Good catch! The change has been merged in the meantime, but I'll take care of a follow-up. Thanks, Marc.

#8 - 2012-05-04 09:43 - Marc Neuhaus

(Since the Gerrit Patch is about to merge i'll put this here as well so it doesn't get forgotten :))

Now i noticed another thing. The PropertyMapping Rules fail in cases where the ViewHelper handles propertyMapping as well like the DatePickerViewHelper. The PropertyMapper chokes on the already converted DateTime and throws an Exception. I'm thinking if this should maybe be considered in the PropertyMapper itself to check if the source Data is already of the target Class and skip it in that case.