TYPO3.Fluid - Feature #43346

Allow property mapping configuration via template

2012-11-27 13:45 - Karsten Dambekalns

Status:	Under Review		**Start date:**	2012-11-27
Priority:	Should have		**Due date:**	
Assignee:	Karsten Dambekalns		**% Done:**	0%
Category:	ViewHelpers		**Estimated time:**	0.00 hour
Target version:	2.1			
Has patch:	No			

Description

The use case of allowing the mapping of dynamically generated fields in forms is quite common. Instead of having to use the full-scale PHP API for that, a simple configuration directly in the form would be nice.

History

#1 - 2012-11-27 13:47 - 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/16797

#2 - 2012-11-28 11:13 - Gerrit Code Review

Patch set 2 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/16797

#3 - 2012-11-28 13:39 - Gerrit Code Review

Patch set 3 for branch **master** has been pushed to the review server.
It is available at http://review.typo3.org/16797

#4 - 2012-11-28 13:51 - Karsten Dambekalns

On the use of array for allowedProperties: I wanted to do that, but if all I need is foo,bar it feels really awkward to type 0:foo,1:bar just because Fluid is limited. If you insist, though, fine…

(Can we add support for "[foo,bar]" then? Because it seems the need to differentiate between an array and a "simple" placeholder is the major reason for the need to always specify key/value)

Then Sebastian wrote:

> What about an <f:form.for each="..." as="..."></f:form.for>, which adds the configuration for * correctly?

The following is the problem: You want to add fields dynamically, but you do not know how many will be there. Thus you need the wildcard (for the collection/array index). The suggested VH does that. What would your form.for do exactly? What is each and what is as?

Sidenote: This is not needed at all, if you let Fluid render just one field with a name of foo.* the needed parts will be in the trusted properties token. This VH is just a nice packaging for that hack… :)

#5 - 2012-11-28 13:51 - Karsten Dambekalns

- Status changed from Under Review to Needs Feedback

#6 - 2012-11-28 14:17 - Bastian Waidelich

Karsten Dambekalns wrote:

> On the use of array for allowedProperties: I wanted to do that, but if all I need is foo,bar it feels really awkward
> to type 0:foo,1:bar just because Fluid is limited. If you insist, though, fine…

I agree, it's a bit cumbersome. But we avoided comma separated strings where possible as they're inflexible.

> (Can we add support for "[foo,bar]" then? Because it seems the need to differentiate between an array and a "simple" placeholder

is the major reason for the need to always specify key/value)

The current syntax is in sync with the JSON object notation. It would be nice to be able to specify simple arrays in the JSON array notation, too. With https://review.typo3.org/#/c/16818/ the side effects would be manageable but it would be a profound change still.. let's see what Sebastian says

> The following is the problem: You want to add fields dynamically, but you do not know how many will be there.

With dynamic you mean via JavaScript for example?
Just for completeness: Currently there are two ways (apart from the *-hack you mentioned) to achieve this:

1. Add the fields via AJAX so the server side can adjust the configuration
2. adjust the configuration in some initialize*Action():

```
public function initializeFooAction() {
    $this->arguments['foo']->getPropertyMappingConfiguration()->allowAllProperties();
    $this->arguments['foo']->getPropertyMappingConfiguration()->forProperty('bar')->allowAllProperties();
}
```

#7 - 2012-11-28 14:38 - Sebastian Kurfuerst

OK, here we go for some more background:

Karsten Hachmeister: could you please paste a little longer snippet of the template? I assume it is something like:

```
<f:form>

  <f:for each="{quux}" as="el" key="k">
    <f:form.textfield property="quux.{k}.name" />
    <f:form.textfield property="quux.{k}.value" />
  </f:for>
  // add JS logic for adding additional elements here
</f:form>
```

I hope I am right in my assumption here :-) Currently you add the <f:form.configuration /> ViewHelper here to make the additional elements editable.

I would imagine that instead of the above code, the following should be written:

```
<f:form>

  <f:form.for each="{quux}" as="el">
    <f:form.textfield property="el.name" />
    <f:form.textfield property="el.value" />
  </f:form.for>
  // add JS logic for adding additional elements here
</f:form>
```

This way, the system would automatically know in which way your collection items are structured; and could add the "*" entries to the property mapping configuration on its own.

I still think we could add the <f:form.configuration /> VH nevertheless; although it is IMHO more of a low-level ViewHelper for rare edge-cases. I'm quite fine with the comma-separated list, as that is just pragmatic at this point. I think we can also support comma-separated strings AND arrays at this point.

Greets,
Sebastian

#8 - 2012-11-28 14:39 - Karsten Dambekalns

Bastian Waidelich wrote:

> With dynamic you mean via JavaScript for example?

Yes.

> Just for completeness: Currently there are two ways (apart from the *-hack you mentioned) to achieve this:

Right. And with the fix for shouldMap() that works just fine. But Sebastian said it would be nice to be able to do that in the form, and indeed it is a lot simpler than using the PHP API for this case.

#9 - 2012-11-28 14:49 - Karsten Dambekalns

Sebastian Kurfuerst wrote:

> [Karsten Hachmeister](): could you please paste a little longer snippet of the template? I assume it is something like:

Pretty much like that, indeed. https://gist.github.com/4723a319a6fda25ff890

> This way, the system would automatically know in which way your collection items are structured; and could add the "*" entries to the property mapping configuration on its own.

Ah, so that would be a clever "two-in-one" way to produce an edit form **and** add the wildcard automagically. Got it. Not really the solution for a create form, though. :)

> I think we can also support comma-separated strings AND arrays at this point.

Well, registerArgument() expects a type, and that is checked later. Boom. No?

#10 - 2012-11-28 15:00 - Sebastian Kurfuerst

Type "mixed" works with all arguments;

> Not really the solution for a create form, though. :)

That's the question. Couldn't we use the same <f:form.for> Helper there? Of course if would need to do some kind of "magic" to determine its fields, but that is certainly doable.

#11 - 2012-11-28 15:03 - Karsten Dambekalns

Sebastian Kurfuerst wrote:

> Type "mixed" works with all arguments;

Ok, didn't try that.

>> Not really the solution for a create form, though. :)

> That's the question. Couldn't we use the same <f:form.for> Helper there? Of course if would need to do some kind of "magic" to determine its fields, but that is certainly doable.

But over what would it iterate? And magic is not always the solution. :)

#12 - 2012-11-28 15:24 - Sebastian Kurfuerst

I'd just render the inner HTML without using it for finding out which fields are inside there. So just calling $this->renderChildren() and ignoring what it returns.

Greets, Sebastian

#13 - 2012-12-10 13:43 - Karsten Dambekalns

- Target version changed from 2.0 to 2.1

#14 - 2013-12-16 10:14 - Gerrit Code Review

- Status changed from Needs Feedback to Under Review

Patch set 4 for branch **master** of project **Packages/TYPO3.Fluid** has been pushed to the review server.
It is available at https://review.typo3.org/16797