TYPO3.Fluid - Feature # 9037

Status:	Rejected	**Priority:**	Won't have this time
Author:	Thomas Mammitzsch	**Category:**	ViewHelpers
Created:	2010-07-26	**Assigned To:**	
Updated:	2010-10-20	**Due date:**	
Has patch:			
Subject:	new comparator for CheckboxViewhelper		

Description

i have a suggestion for a new "in_array" comparator in ViewHelperNode. Reason is that i'm creating forms in the frontend, where form definition happens in the backend. Everything works fine, even if i get validation errors, the form shows up prefilled.

The only exception is the CheckboxViewHelper. Because this is an array of values (at least if you have multiple checkboxes with the same name and different values) you can not compare it against a single value currently.
It would be nice if one can check if a value is in that array, to check the checkbox in case of validation errors, a return from a last-check-before-submit page (or generally when the form of the newAction has to be displayed again). I added one case to the evaluateComparator method in ViewHelperNode to do that and it works:

```
case 'in_array':
   return in_array($leftSide, $rightSide);
```

of course i also added 'in_array' also to the comparators static array in ViewHelperNode.
Could we add this permanently, what do you mean?

History

#1 - 2010-07-26 18:17 - Bastian Waidelich

- Category set to ViewHelpers

- Status changed from New to Needs Feedback

- Branch set to v4 + v5

Hi Thomas,

could you provide an example of what you want to achieve?
normally you shouldn't have to care about selecting/unselecting the checkbox as that's handled by the ViewHelper:

```
// PHP
$user = new stdClass();
$user->interests = array('Snowboarding', 'TYPO3');
$this->view->assign('user', $user);

// Fluid
<f:form object="{user}" objectName="user">
  Interests:<br />
  Snowboarding: <f:form.checkbox property="interests" value="Snowboarding" /><br />
  TYPO3: <f:form.checkbox property="interests" value="TYPO3" /><br />
  Skiing: <f:form.checkbox property="interests" value="Skiing" /><br />
</f:form>
```

the first two checkboxes should be pre-selected.

There is an issue currently, if you try to bind a checkbox to a property that does not exist (for example because the form object is NULL). I'm currently working on this (#8854)

#2 - 2010-07-27 18:36 - Thomas Mammitzsch

hi Bastian,

its a bit tricky to explain. I have fieldsets and formfields which can be arranged by the be-user (similar to powermail). The be-user also maps a each form field to a fe-user property. Now my extension renders the form in the frontend. If the website user sends the form, there is a new frontend user object created. If there is a validation error, the newAction (aka the form) is redisplayd and has to be prefilled with the user content to let the user correct the form. The problem is, that the form is generated to display fieldset and field objects, not a fe_user object. So i map the fe_user properties back to the field properties, which works - except for checkboxes.
Here is some code:

```
//PHP
    public function newAction(Tx_Easyferegister_Domain_Model_frontendUser $newfrontendUser = null) {
        $fieldsets = $this->fieldsetRepository->findAll();

        if($newfrontendUser == null){
            $newfrontendUser = Tx_Easyferegister_Utility_SessionUtility::getSession(); //do we have a user in session? (coming from confirm
action)
        }

        //fill the fields with values from newfrontendUser so user can edit it in case of validation error, not confirmed, etc.
        if($newfrontendUser != null) {
            foreach($fieldsets as $fieldset){
                foreach($fieldset->getFields() as $field){
                    $field->setFormfieldValue($newfrontendUser);
                }
            }
        }

        $this->view->assign('fieldsets', $fieldsets);
        $this->view->assign('newfrontendUser', $newfrontendUser);
    }

//Fluid new.html

            <f:form method="post" controller="frontendUser" action="confirm" name="newfrontendUser" object="{newfrontendUser}"
arguments="{fieldset : fieldset}">
                <f:for each="{fieldsets}" as="fieldset">
                    <h2>{fieldset.title}</h2>
                    <f:if condition="{fieldset.fields}">
                        <f:then>
                            <f:for each="{fieldset.fields}" as="field">
                                <f:render partial="formFields" arguments="{field: field}" />
                            </f:for>
                        </f:then>
                        <f:else>
                            <p>There are no fields within this fieldset definded.</p>
```

```
                    </f:else>
                  </f:if>
                </f:for>
                </f:form>

    //Fluid partial formFields.html

    {namespace efr=Tx_Easyferegister_ViewHelpers}
    <efr:switch value="{field.formtype}">
      <efr:case value="select">
          <label for="{field.name}">{field.title}{field.description}<f:if condition="{field.flexform.mandatory}"> *</f:if></label>
          <f:form.select name="newfrontendUser[{field.fefield}]" options="{field.flexform.options}" multiple="{field.flexform.multiple}"
    value="{field.formfieldValue}"/>
      </efr:case>
      <efr:case value="check">
        <p class="tx_easyferegister_fieldwrap_html_checkbox_title">{field.title}{field.description}<f:if condition="{field.flexform.mandatory}">
    *</f:if></p>

          <f:for each="{field.flexform.checkboxValues}" as="value">
            <label for="{field.name}">{value}</label>
            <f:form.checkbox name="newfrontendUser[{field.fefield}][]" value="{value}" checked="{value} in_array {field.formfieldValue}" />
          </f:for>
      </efr:case>
```

my proposal is in the third last line...

#3 - 2010-08-06 20:50 - Bastian Waidelich

- Status changed from Needs Feedback to Rejected

- Priority changed from Should have to Won't have this time

Hi Thomas,

I get your point now, but I think that we should try to avoid to add too many keywords to Fluid.
Instead you could write a ViewHelper that you can use like this:

```
    checked="{value -> x:inArray(array: field.formfieldValue)}"
```

Having said that, using too many ViewHelpers could be a sign for an architecture that is too complex.
You might think about reflecting your formfields in a stricter model.
Then you can bind the form field to properties of this model and use partials to render reoccurring field sets.

Feel free to reopen the issue if you don't agree or want to discuss this further.