TYPO3.Fluid - Feature #46257

Add escape sequence support for Fluid

2013-03-14 09:01 - Rafael Kähm

Status:	Under Review	**Start date:**	2013-03-14
Priority:	Should have	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Core	**Estimated time:**	0.00 hour
Target version:			
Has patch:	No		

Description
In some cases you need to print some characters to rendered view, which Fluid uses. f.e.: {{object}} should output {value} but Fluid renders this to {{object}}   i have patch for this, which make it possible following possible: \{{object}\} renders to {value}   \\\{{object}\\\} renders to \{value\}   please make gerrit from following gist: https://gist.github.com/RafaelKa/5159642   there are 54,55,56 , 119-123 and 455,456 lines interesting

History

#1 - 2013-03-14 09:45 - Rafael Kähm

yes it is a copy of CDATA but most convenient:

CDATA:
<![CDATA[{]]>{object}<![CDATA[}]]> renders to {value}

vs

escape sequenze:
\{{object}\} renders to {value}

#2 - 2013-03-14 10:27 - Bastian Waidelich

- Category set to Core

- Status changed from New to Needs Feedback

To be honest, I'm not so sure that we should adjust the parser for this. Wouldn't

```
&#123;{object}&#125;
```

work in your case?

#3 - 2013-03-14 11:06 - Rafael Kähm

Bastian SGML/HTML/XML entity will not work
output is

{value}

and i think this is good so as it is -> maybe you need to print SGML/HTML/XML to

Escape sequence is usefull for StandaloneView.
Here is a demo
/** * An example command to render views with escape sequenze * * This renders a view with Fluid to String and print this to your shell *
* @return void */publicfunctionescapeSequenzeDemoCommand(){$templateSource=' cdata <![CDATA[{]]>{object.property1}<![CDATA[}]]>
escapeSequence \{{object.property1}\} <![CDATA[\{]]>{object.property1}<![CDATA[\}]]
<![CDATA[{]]>{object.property1}<![CDATA[}]]] XML entity {{object.property1}} ';$view=new\TYPO3\Fluid\View\StandaloneView();
$view->setTemplateSource($templateSource);$object=array('property1'=>'value');$view->assign('object',$object);echo$view->render().PHP_EOL;}

output:

```
cdata{value}escapeSequence{value}<![CDATA[{]]>value<![CDATA[}]]<![CDATA[{]]>value<![CDATA[}]]XMLentity&#123;value&#125;
```

#4 - 2013-03-19 14:30 - Adrian Föder

TBH I just discovered another use case for my very own, I needed to

```
<f:form.hidden name="message[recipient]" value="{recipientIdentifier}" />
```

while the value should literally be {recipientIdentifier} (it's for JavaScript substitution lateron).
Any ideas how to achieve this currently?

#5 - 2013-03-24 11:39 - Rafael Kähm

Adrian Föder wrote:

> TBH I just discovered another use case for my very own, I needed to
>
> [...]
>
> while the value should literally be {recipientIdentifier} (it's for JavaScript substitution lateron).
> Any ideas how to achieve this currently?

---

currently you can use CDATA

<f:form.hiddenname="message[recipient]"value="<![CDATA[{recipientIdentifier}]]>"/>

---

i tried your example with my solution of escape sequenze as follows:

<f:form.hiddenname="message[recipient]"value="\{recipientIdentifier\} />

and Fluid prints:

<inputtype="hidden"name="message[recipient]"value="{recipientIdentifier}"/>

#6 - 2013-03-24 16:53 - Rafael Kähm

with on https://gist.github.com/RafaelKa/5159642 code kickstarter breaks

...useTYPO3\Flow\AnnotationsasFlow;...

to

...useTYPO3FlowAnnotationsasFlow;...

in
https://gist.github.com/RafaelKa/5232405

i added a list of to escaped characters [ ] { } \

are there some more characters to be escaped by Fluid?

#7 - 2013-03-25 11:49 - Tarcisio Fedrizzi

Rafael Kähm wrote:

> with on https://gist.github.com/RafaelKa/5159642 code kickstarter breaks
>
> ...>useTYPO3\Flow\AnnotationsasFlow;>...
>
> to
>
> ...>useTYPO3FlowAnnotationsasFlow;>...
>
> in
> https://gist.github.com/RafaelKa/5232405
>
> i added a list of to escaped characters [ ] { } \
>
> are there some more characters to be escaped by Fluid?

This works as well (I know that is not so clean but):

In the controller:

```
$this->view->assign('oBrc', '{');
$this->view->assign('cBrc', '}');
```

In the view:

```
{oBrc}MyValue{cBrc}
```

you can write the literal closing brace and you obtain the same value. Like this:

```
{oBrc}MyValue}
```

The use case that Adrian points out can be worked out in a simpler way like this:
controller:

```
$this->view->assign('recipientIdentifier', '{recipientIdentifier}');
```

view:

```
<f:form.hidden name="message[recipient]" value="{recipientIdentifier}" />
```

Of course if you have to print out many literal values (or to produce for example a JSON template) the ability to escape braces would be the best way to achieve it.

#8 - 2013-09-20 01:09 - Artus Kolanowski

Tarcisio Fedrizzi wrote:

> This works as well (I know that is not so clean but):
>
> In the controller:
> [...]
>
> In the view:
> [...]
>
> you can write the literal closing brace and you obtain the same value. Like this:
>
> [...]
>
> The use case that Adrian points out can be worked out in a simpler way like this:
> controller:
> [...]
>
> view:
> [...]
>
> Of course if you have to print out many literal values (or to produce for example a JSON template) the ability to escape braces would be the best way to achieve it.

I think this is currently the most usable workaround. But instead of using a controller you can also use f:alias in a template.

```
<f:alias map="{ocb: '{', ccb: '}'}">
    <f:uri.action arguments="{start: '{ocb}start{ccb}'}" [...] />
</f:alias>
```

Anyway such a feature would make some templates much more cleaner.

#9 - 2013-10-18 15:51 - Gerrit Code Review

- Status changed from Needs Feedback to Under Review

Patch set 1 for branch **master** has been pushed to the review server.
It is available at https://review.typo3.org/24898

#10 - 2014-08-12 11:26 - Hannes Lau

To my eyes, it would be better to escape single { characters, so one can still use Fluid variables inside a pair of escaped curly braces.

As an example, I am trying to use Fluid to produce some CSS code like this:

```
<n:headerData>
    <style type="text/css">
        .article-{newsItem.uid} \{
            background-image: url('{f:uri.image(src: "{mediaElement.uid}")}');
        \}
    </style>
</n:headerData>
```

This would not work with the \{escaped} approach.

#11 - 2015-01-26 09:00 - Stefan Neufeind

@Hannes: An escaped opening brace \{ can immediately be followed by another opening brace (for the variable). So imho you could do: \{{variable}}
It could be discussed whether escaping the closing brace as well would make it easier to read. But it's not mandatory since with no active opening brace at that point the closing brace is just regarded as text.