TYPO3.Fluid - Feature #5158

Add "true" and "false" to Fluid syntax

2009-10-28 11:58 - Bastian Waidelich

Status:	Closed	**Start date:**	2009-10-28
Priority:	Should have	**Due date:**	
Assignee:	Sebastian Kurfuerst	**% Done:**	100%
Category:	Core	**Estimated time:**	0.00 hour
Target version:			
Has patch:	No		

Description

Currently

```
{f:uri.resource(path: 'somePath', absolute: true)}
```

throws an exception, because the true is "translated" into {true} internally.

```
{f:uri.resource(path: 'somePath', absolute: 'true')}
```

and

```
{f:uri.resource(path: 'somePath', absolute: 1)}
```

is possible, but it would be nicer to add "true" and "false" as language constructs.

Associated revisions

Revision beaa1450 - 2009-10-28 16:16 - Bastian Waidelich

[+FEATURE] Fluid (ViewHelpers): extended format.date viewhelper by an additional "date" argument. Now you can write {f:format.date(date: 'yesterday')} in your templates. This change is backwards compatible! Relates to #5150
[+FEATURE] Fluid (ViewHelpers): extended if viewhelper by arguments "then" and "else". Now you can write {f:if(condition: 1, then: 'yes', else: 'no'} in your templates. This change is backwards compatible! Relates to #5150
[~TASK] Fluid (Core): added check for reserved keywords to TemplateVariableContainer. This relates to #5158
[~TASK] Fluid (Tests): moved and renamed VariableContainerTest

Revision 3365 - 2009-10-28 16:16 - Bastian Waidelich

[+FEATURE] Fluid (ViewHelpers): extended format.date viewhelper by an additional "date" argument. Now you can write {f:format.date(date: 'yesterday')} in your templates. This change is backwards compatible! Relates to #5150
[+FEATURE] Fluid (ViewHelpers): extended if viewhelper by arguments "then" and "else". Now you can write {f:if(condition: 1, then: 'yes', else: 'no'} in your templates. This change is backwards compatible! Relates to #5150
[~TASK] Fluid (Core): added check for reserved keywords to TemplateVariableContainer. This relates to #5158
[~TASK] Fluid (Tests): moved and renamed VariableContainerTest

History

#1 - 2010-06-18 10:12 - Sebastian Kurfuerst

- Assignee deleted (Sebastian Kurfuerst)

#2 - 2010-06-18 14:51 - Sebastian Kurfuerst

- Status changed from New to Accepted

#3 - 2010-07-12 12:50 - Bastian Waidelich

- Status changed from Accepted to Closed

Thinking about this again - I guess, it's more consistent as it is right now:
'true' => String "true"
true => Variable $true
1 => Number 1

I'm closing this issue for now. Feel free to re-open if you do not agree or want to discuss this further.

#4 - 2010-07-12 14:22 - Sebastian Kurfuerst

- Status changed from Closed to Needs Feedback

I still think such a feature makes sense, the "true" string should be handled the same as the "true" constant / variable, and the same as 1.

#5 - 2010-07-12 14:52 - Bastian Waidelich

Sebastian Kurfuerst wrote:

> I still think such a feature makes sense, the "true" string should be handled the same
> as the "true" constant / variable, and the same as 1.

Ok, might be more intuitive that way.. But then we should make sure, that those "constant" variables are reserved and something like

```
$this->view->assign('true', 'foo');
$this->view->assign('FALSE', 'bar');
```

is not possible.

#6 - 2010-07-12 22:48 - Sebastian Kurfuerst

Yep, they cannot be re-declared, see
$reservedVariableNames inside
https://svn.typo3.org/TYPO3v4/CoreProjects/MVC/fluid/trunk/Classes/Core/ViewHelper/TemplateVariableContainer.php :-)

Greets,
Sebastian

#7 - 2010-10-11 13:15 - Karsten Dambekalns

- Status changed from Needs Feedback to Accepted

- Assignee set to Sebastian Kurfuerst

Seems the feedback state is resolved, I would be happy to have true be true and 'true' be a string. Noone wants a variable named true anyway...

#8 - 2010-10-11 13:30 - Bastian Waidelich

Karsten Dambekalns wrote:

> I would be happy to have true be true and 'true' be a string.
> Noone wants a variable named true anyway...

'true' is a string but is converted to TRUE when used in boolean arguments:

```
<f:if condition="some sting">
or
{f:if(condition: 'some string')}
```

will always be true for non-empty strings (even 'false').

The correct way to write it (when boolean constants are available) would be

```
<f:if condition="{true}">
or
{f:if(condition: true)}
```

right?

#9 - 2010-10-11 13:33 - Bastian Waidelich

Bastian Waidelich wrote:

> The correct way to write it (when boolean constants are available) would be
> [...]

thinking about it again..

```
<f:if condition="false" />
```

would be true, which is a bit confusing, innit?

#10 - 2010-12-17 16:38 - Karsten Dambekalns

Bastian Waidelich wrote:

> thinking about it again..
> [...]
> would be true, which is a bit confusing, innit?

Well, but correct.

#11 - 2011-07-20 01:09 - Stefan Neufeind

Bastian, I agree that the string (althought it contains "false") should evaluate to true. It's a string after all ...

#12 - 2013-06-05 10:40 - Bastian Waidelich

- Status changed from Accepted to Closed

- % Done changed from 0 to 100

- Has patch set to No

This has been resolved a long time ago (I can't find the respective issue at the moment).

BTW: The behavior is:

```
<!-- TRUE: -->
{f:if(condition: 'someString')}
{f:if(condition: 1)}
{f:if(condition: '1')}
{f:if(condition: someCollectionWithAtLeastOneItem)}

<!-- FALSE: -->
{f:if(condition: '-1')}
{f:if(condition: '0')}
{f:if(condition: 'FaLsE')}
{f:if(condition: '')}
{f:if(condition: null)}
{f:if(condition: someEmptyCollection)}
```

So

```
<f:if condition="false" />
```

is (and has been) **FALSE** in fact - see
https://git.typo3.org/FLOW3/Packages/TYPO3.Fluid.git/blob/HEAD:/Classes/TYPO3/Fluid/Core/Parser/SyntaxTree/BooleanNode.php#l324