conteq — continued equalities*

Joachim Breitner†

Released 2015/07/17

1 Introduction

There are various ways to layout an continued equality that spans multiple lines and has explanations along some steps. Often, the best layout is not clear beforehand, as it depends on the sizes of the various elements, or implementing it adds too much noise to the actual formulas. This package provides an environment for continued equalities (or inequalities) that allows you to easily define and switch layouts.

2 Usage

`conteq` Our running example is the following continued equality:

```
\begin{conteq}
e^{\pi\cdot i} \\
= -1 & Euler's formula \\
< 0 & this is an inequality \\
< \sqrt 3 \\
= \int e^{-x^2} dx & this is due to Gauss.
\end{conteq}
```

As you can see, the expressions of the continued equality are separated by `\\`, with equality signs (or other relations) at the beginning of all lines but the first. Some equalities also have explanations.

The result of the above code is

$$\begin{aligned} e^{\pi\cdot i} &= -1 && \{ \text{ Euler's formula } \} \\ &< 0 && \{ \text{ this is an inequality } \} \\ &< \sqrt{3} \\ &= \int e^{-x^2} dx && \{ \text{ this is due to Gauss. } \} \end{aligned}$$

*This file describes version v0.1.1, last revised 2015/07/17.

†E-mail: mail@joachim-breitner.de

The expressions are typeset in math mode, while the explanation is assumed to be regular text. The curly braces around the explanations come from the default `\ConteqExplStyle`.

There are other layouts available, which you select with an optional parameter to `\begin{conteq}`, e.g.

```
\begin{conteq}[explline]
[..]
\end{conteq}
```

The layouts defined by `conteq` are:

plain The default layout, shown above.

explline A layout that puts the explanations below the right-hand side of the equalty it is explaining. To be used when the explanations and righ-hand sides are long.

$$\begin{aligned} e^{\pi \cdot i} &= -1 \\ &\quad \{ \text{ Euler's formula } \} \\ &< 0 \\ &\quad \{ \text{ this is an inequality } \} \\ &< \sqrt{3} \\ &= \int e^{-x^2} dx \\ &\quad \{ \text{ this is due to Gauss. } \} \end{aligned}$$

headline Like plain, but the first expression is also vertically aligned with the right-hand sides.

$$\begin{aligned} & e^{\pi \cdot i} \\ &= -1 && \{ \text{ Euler's formula } \} \\ &< 0 && \{ \text{ this is an inequality } \} \\ &< \sqrt{3} \\ &= \int e^{-x^2} dx && \{ \text{ this is due to Gauss. } \} \end{aligned}$$

onecolumn A combination of **explline** and **headline** that puts everything in one column, for maximum space efficiency.

$$\begin{aligned} & e^{\pi \cdot i} \\ = & -1 \\ & \{ \text{ Euler's formula } \} \\ < & 0 \\ & \{ \text{ this is an inequality } \} \\ < & \sqrt{3} \\ = & \int e^{-x^2} dx \\ & \{ \text{ this is due to Gauss. } \} \end{aligned}$$

oneline A layout, more for demonstrational purposes, that ignores the explanations and puts everything in one line.

$$e^{\pi \cdot i} = -1 < 0 < \sqrt{3} = \int e^{-x^2} dx$$

`\ConteqSetDefaultLayout` You can change the default layout using `\ConteqSetDefaultLayout{`⟨*layout*⟩`}`

3 Defining layouts

`\ConteqDefineLayout` To define a new layout you use `\ConteqDefineLayout`, which takes seven arguments:

1. The name of the layout,
2. What to put before the continued equalities,
3. the left-hand side of the first equality,
4. the right-hand side of equalities without explanation,
5. the right-hand side of equalities with explanation,
6. the line separator and
7. what to put after the continued equalities.

For example the existing layouts are defined using

```
1
2 \ConteqDefineLayout
3     {plain}
4     {\begin{align*}}
5     { #1 }
6     { & #1 }
7     { & #1 && #2 }
```

```
    { \\ }
    {\end{align*}}
\ConteqSetDefaultLayout{plain}

\ConteqDefineLayout
    {explline}
    {\begin{align*}}
    { #1 }
    { & #1 }
    { & #1 \\ & \mathrel{\phantom{=}} #2 }
    { \\ }
    {\end{align*}}

\ConteqDefineLayout
    {headline}
    {\begin{align*}}
    { &\mathrel{\phantom{=}} #1 \\ }
    { & #1 }
    { & #1 && #2 }
    { \\ }
    {\end{align*}}

\ConteqDefineLayout
    {onecolumn}
    {\begin{align*}}
    { &\mathrel{\phantom{=}} #1 \\ }
    { & #1 }
    { & #1 \\ & \mathrel{\phantom{=}} #2 }
    { \\ }
    {\end{align*}}

\ConteqDefineLayout
    {oneline}
    {\[}
    { #1 }
    { #1 }
    { #1 }
    {}
    {\]}

```

4 Changing the explanation style

`\ConteqExplStyle` The explanation is formatted using the macro `\ConteqExplStyle{`⟨*explanation*⟩`}`, which is by default defined as

```
\newcommand{\ConteqExplStyle}[1]{\{~#1~\}}
```

To change the style, simply redefine this macro using, for example:

```
\renewcommand{\ConteqExplStyle}{\textit{-- #1}}
```

5 Future work

This package is, at the time of writing, very new, so please le me know about problems you are having or features you are missing.

One feature that I am considering is an auto-selection of layouts, so when you specify `\begin{conteq}[plain,explline,onecolumn]...\end{conteq}`, it will analyze the table and select, from your list of layouts, the first one that is "ok", where "ok" would be some layout-specific heuristic taking the size of the expressions and explanations as well as the current `\linewidth` into account. If you think that this would be useful to you, please let me know.

You can follow the package's development at http://git.nomeata.de/?p=conteq.git or the mirror at https://github.com/nomeata/conteq.

6 Implementation

6.1 Package loading

```
\RequirePackage{amsmath}
\RequirePackage{environ}
```

6.2 Defining Messages

```
\msg_new:nnn
    { conteq }
    { empty }
    { Empty~conteq~environment~\msg_line_context: }

\msg_new:nnn
    { conteq }
    { ignoreddata }
    { Ignored~text~\msg_line_context: }

\msg_new:nnn
    { conteq }
    { undefined layout }
    { Undefined~layout~``#1''~\msg_line_context: }
```

6.3 Declaring local variables

```
\tl_new:N \l__conteq_default_layout_tl
\tl_new:N \l__conteq_layout_tl
\tl_new:N \l__conteq_body_tl
\tl_new:N \l__conteq_lines_seq
\tl_new:N \l__conteq_cells_seq
\tl_new:N \l__conteq_head_tl
\tl_new:N \l__conteq_lastline_tl
\tl_new:N \l__conteq_rhs_tl
\tl_new:N \l__conteq_expl_tl
\tl_new:N \l__conteq_result_tl
```

6.4 Layouts

\ConteqSetDefaultLayout

The code to set the default layout.

```
\cs_new_protected:Nn \__conteq_set_default_layout:n
  {
    \tl_set:Nn \l__conteq_default_layout_tl {#1}
  }
\cs_new_eq:NN \ConteqSetDefaultLayout \__conteq_set_default_layout:n
```

(*End definition for* `\ConteqSetDefaultLayout`*. This function is documented on page* **??**.)

\ConteqDefineLayout

The code to define new layouts.

```
\tl_const:Nn \c__conteq_prefix_tl { __conteq_layouts }

\cs_new_protected:Nn \__conteq_define_layout:nnnnnnn
  {
    \cs_new_protected:cn { \c__conteq_prefix_tl _ #1 _begin: } { #2 }
    \cs_new_protected:cn { \c__conteq_prefix_tl _ #1_lhs:n } { #3 }
    \cs_new_protected:cn { \c__conteq_prefix_tl _ #1_rhs_only:n } { #4 }
    \cs_new_protected:cn { \c__conteq_prefix_tl _ #1_rhs_expl:nn } { #5 }
    \cs_new_protected:cn { \c__conteq_prefix_tl _ #1_nl: } { #6 }
    \cs_new_protected:cn { \c__conteq_prefix_tl _ #1_end: } { #7 }
  }
\cs_new_eq:NN \ConteqDefineLayout \__conteq_define_layout:nnnnnnn
```

(*End definition for* `\ConteqDefineLayout`*. This function is documented on page* **??**.)

6.5 Expansion utility function

__conteq_args_once:Nn
__conteq_args_once:NV
__conteq_args_once:Nnn
__conteq_args_once:NVV

For ease of debugging(?) we construct a token list that contains exactly the tokens that a programmer would enter to create the layout manually. For that we need some fine-grained control over expansion.

```
\cs_new:Npn \__conteq_args_once:Nn #1#2
  {
   \exp_not:o { #1 {#2} }
  }
\cs_generate_variant:Nn \__conteq_args_once:Nn { NV }

\cs_new:Npn \__conteq_args_once:Nnn #1#2#3
  {
   \exp_not:o { #1 { #2 }{ #3 } }
  }
\cs_generate_variant:Nn \__conteq_args_once:Nnn { NVV }
```

(*End definition for* `\__conteq_args_once:Nn` *and others. These functions are documented on page* **??**.)

6.6 Main code

`\__conteq_print_line:nn` This macro splits and prints one line of the table. The second argument is either `\\` or, for the last line of the table, empty.

```
\cs_new_protected:Npn \__conteq_print_line:Nnn #1#2#3
  {
    \seq_set_split:Nnn \l__conteq_cells_seq { & } { #2 }
    \seq_pop_left:NN \l__conteq_cells_seq \l__conteq_rhs_tl
    \tl_clear:N \l__conteq_expl_tl
    \seq_if_empty:NF \l__conteq_cells_seq
      {
\seq_pop_left:NN \l__conteq_cells_seq \l__conteq_expl_tl
      }
    \tl_if_blank:VTF \l__conteq_expl_tl
      {
\tl_put_right:Nx #1
  {
    \__conteq_args_once:NV \__conteq_rhs_only:n \l__conteq_rhs_tl
  }
      }
      {
\tl_set:Nx \l_tmpa_tl { \exp_not:N \text { \exp_not:N \ConteqExplStyle { \exp_not:V \l__conteq
\tl_put_right:Nx #1
  {
    \__conteq_args_once:NVV \__conteq_rhs_expl:nn \l__conteq_rhs_tl \l_tmpa_tl
         }
      }
      \seq_if_empty:NF \l__conteq_cells_seq
{
  \msg_warning:nn{conteq}{ignoreddata}
}
    \tl_put_right:Nn #1 {#3}
  }
\cs_generate_variant:Nn \__conteq_print_line:Nnn { Nno, NVn }
```

(*End definition for* `\__conteq_print_line:nn`*. This function is documented on page* **??**.)

`conteq` The main environment of the package.

```
\NewEnviron{ conteq }[1][ \l__conteq_default_layout_tl ]{
  \tl_set:NV \l__conteq_body_tl \BODY

  \tl_if_blank:oT \l__conteq_body_tl
    {
      \msg_warning:nn{conteq}{empty}
    }
```

Figure out the layout to use...

```
  \tl_set:Nn \l__conteq_layout_tl { #1 }
```

and set the various functions accordingly, if the layout exists.

```
\cs_if_exist:cTF { \c__conteq_prefix_tl _ \l__conteq_layout_tl _ begin: }
  {
    \clist_map_inline:nn{ begin:, lhs:n,rhs_only:n, rhs_expl:nn, nl:, end: }
      {
        \cs_set_eq:cc { __conteq_ ##1 }{ \c__conteq_prefix_tl _ \l__conteq_layout_tl _ ##1 }
      }
  }{
    \msg_critical:nnx{conteq}{undefined layout}{\l__conteq_layout_tl}
  }
```

Split the body into individual lines.

```
\seq_set_split:NnV \l__conteq_lines_seq { \\ } \l__conteq_body_tl

\tl_clear:N \l__conteq_result_tl
```

If there is only one line, simply print it.

```

\int_case:nnF { \seq_count:N \l__conteq_lines_seq }
  {
    {1}
      {
        \tl_put_right:Nx \l__conteq_result_tl { \exp_not:o \__conteq_begin: }
        \tl_put_right:NV \l__conteq_result_tl \l__conteq_body_tl
        \tl_put_right:Nx \l__conteq_result_tl { \exp_not:o \__conteq_end: }
      }
  }
  {
```

Otherwise extract the head and the last line, and print each line using `\__conteq_print_line:nn`

```
    \seq_pop_left:NN \l__conteq_lines_seq \l__conteq_head_tl
    \seq_pop_right:NN \l__conteq_lines_seq \l__conteq_lastline_tl

    \tl_put_right:Nx \l__conteq_result_tl { \exp_not:o \__conteq_begin: }
    \tl_put_right:Nx \l__conteq_result_tl
      {
        \__conteq_args_once:NV \__conteq_lhs:n \l__conteq_head_tl
      }
    \seq_map_inline:Nn \l__conteq_lines_seq
      {
        \__conteq_print_line:Nno \l__conteq_result_tl { ##1 } { \__conteq_nl: }
      }
    \__conteq_print_line:NVn \l__conteq_result_tl \l__conteq_lastline_tl {}
    \tl_put_right:Nx \l__conteq_result_tl { \exp_not:o \__conteq_end: }
  }
  % Use this for debugging
  %\tl_show:N \l__conteq_result_tl
  \tl_use:N \l__conteq_result_tl
}

\endinput
```

Change History

0.1

General: Converted to DTX file 1

0.1.1

General: Stop using deprecated expl3 macros 1