The textualicomma package

https://gitlab.com/SFr682k/textualicomma

Sebastian Friedl
sfr682k@t-online.de

2018/08/23 (v1.1)

Abstract

Use the textual comma character as decimal separator in math mode.

textualicomma is based on the icomma package and intended as a solution for cases where the comma character used in horizontal (i.e. text) mode discerns from the one used in math mode.
Note: Escaping to text mode every time a comma is used in math mode might slow down the compilation of huge documents. Please check, whether icomma already satisfies your needs.

Contents

Dependencies and other requirements

textualicomma requires LaTeX 2$_\varepsilon$ and depends on the amstext package.
Also, the three NFSS commands `\rmfamily`, `\sffamily` and `\ttfamily` have to be defined (which holds true for every common document class).

Installation

Extract the *package* file first:

1. Run LaTeX over the file `textualicomma.ins`
2. Move the resulting `.sty` file to `TEXMF/tex/latex/textualicomma/`

Then, you can compile the *documentation* yourself by executing

```
lualatex textualicomma-doc.dtx
makeindex -s gind.ist textualicomma-doc.idx
makeindex -s gglo.ist -o textualicomma-doc.gls textualicomma-doc.glo
lualatex textualicomma-doc.dtx
lualatex textualicomma-doc.dtx
```

or just use the precompiled documentation shipped with the source files.
In both cases, copy the files `textualicomma-doc.pdf` and `README.md` to `TEXMF/doc/latex/textualicomma/`

License

This work consists of the following files:

- `textualicomma.sty`,
- `textualicomma.ins`,
- `textualicomma-doc.dtx` and
- the derived file `textualicomma.sty`

Part I
The documentation

1 Introduction

Using the comma as decimal separator in ordinary LaTeX has always been problematic: since the comma is treated as punctuation,TeX adds some space after it. The solution was defining an 'intelligent' comma recognizing whether it's used as decimal separator or punctuation character.

The common package implementing this solution is icomma by Walter Schmidt [1]. However, some fonts, e.g. Source Serif Pro, don't provide any math support. Although there are fonts looking very similar to Source Serif and providing math support (like Utopia and its 'clones' Fourier and Erewhon), more or less obvious differences are visible (compare ',' to ',' at high magnifications). The only solution solving this 'problem' is escaping to text mode.

2 The textualicomma package

2.1 textualicomma's behaviour

textualicomma makes some changes and additions to icomma's code in order to provide automated escaping to text mode. The behaviour remains the same:

- The comma is treated as decimal separator if it is directly followed by a number. Thus, a number is to be entered as, for instance,
  `3,142`
- whereas mathematical expressions like (x, y) are to be written with a space after the comma:
  `(x,␣y)`

Also, the period is redefined so that it adapts to the used font. This feature might be useful with LaTeX beamer's sans-serif math.

2.2 Package options

By default, the font used for typesetting the comma is the current roman text font. This behavior can be modified by using package options:

`sffamily` Passing the `sffamily` option to textualicomma results in the comma character being taken from the current sans serif text font

`ttfamily` Passing the `ttfamily` option to textualicomma results in the comma character being taken from the current mono-spaced text font

`\textualicommafont` Both options change the used font family *for the whole document*. Changes *inside* the document can be made using the `\textualicommafont` command:

`\textualicommafont{\rmfamily}`	→	roman comma in math mode
`\textualicommafont{\sffamily}`	→	sans-serif comma in math mode
`\textualicommafont{\ttfamily}`	→	monospaced comma in math mode

3 Bugs and problems

Generally, since the comma is made 'active', problems are not unlikely.

For example, when using the 'intelligent comma' together with the dcolumn package, a comma to be *printed* as the decimal separator in a column of type `D` is to be specified as `{\mathord\mathcomma}`, rather than `{,}`, since the latter leads to an error. For example:

```
\begin{tabular}{... D{,}{\mathord\mathcomma}{2} ...}
```

Note that specifying the comma as the related input character works as usual.

References

[1] Walter Schmidt. The icomma package for LaTeX 2ε. Package and Documentation available on CTAN (`https://www.ctan.org/pkg/icomma`)

Part II
The package code

Initialization

```
\ProvidesPackage{textualicomma}%
    [2018/08/23 v1.1 Use textual comma characters as decimal separators]
\NeedsTeXFormat{LaTeX2e}
```

Load dependencies and check requirements

Load the amstext package to enable escaping to horizontal mode:

```
\RequirePackage{amstext}
```

Check whether `\rmfamily`, `\sffamily` and `\ttfamily` are available:

```
\AtBeginDocument{%
    \ifx\rmfamily\@undefined%
        \PackageError{textualicomma}%
            {\noexpand\rmfamily is undefined. textualicomma doesn't work}%
            {textualicomma requires the \noexpand\rmfamily command to be
             defined; otherwise, this package won't work properly}%
        \PackageWarning{textualicomma}{Emergency stop}%
        \stop%
    \fi%
    \ifx\sffamily\@undefined%
        \PackageError{textualicomma}%
            {\noexpand\sffamily is undefined. textualicomma doesn't work}%
            {textualicomma requires the \noexpand\sffamily command to be
             defined; otherwise, this package won't work properly}%
        \PackageWarning{textualicomma}{Emergency stop}%
        \stop%
    \fi%
    \ifx\ttfamily\@undefined%
        \PackageError{textualicomma}%
            {\noexpand\ttfamily is undefined. textualicomma doesn't work}%
            {textualicomma requires the \noexpand\ttfamily command to be
             defined; otherwise, this package won't work properly}%
        \PackageWarning{textualicomma}{Emergency stop}%
        \stop%
    \fi%
}
```

Package options

Provide a `sffamily` option for using the sans serif font:

```
\newif\if@txticmma@sffamily
\@txticmma@sffamilyfalse
\DeclareOption{sffamily}{\@txticmma@sffamilytrue}
```

Provide a `ttfamily` option for using the typewriter font:

```
\newif\if@txticmma@ttfamily
\@txticmma@ttfamilyfalse
\DeclareOption{ttfamily}{\@txticmma@ttfamilytrue}
```

Process all options passed to textualicomma:

```
\ProcessOptions
```

Configuration

Use the roman text font family by default:

```
\def\txticmma@ffamily{\rmfamily}
```

If one of the `sffamily` or `ttfamily` options has been used, changes are applied:

```
\if@txticmma@sffamily\def\txticmma@ffamily{\sffamily}\fi
\if@txticmma@ttfamily\def\txticmma@ffamily{\ttfamily}\fi
```

`\textualicommafont` Provide a command for switching the font family inside the document:

```
\def\textualicommafont#1{\def\txticmma@ffamily{#1}}
```

Defining the “textual intelligent comma”

At the very beginning, provide access to the textual comma character. Also, define a math character referring to the comma contained in the math font, to be used in enumerations:

```
\AtBeginDocument{%
  \def\m@thtextcomma{\text{\txticmma@ffamily,}}
  \mathchardef\mathcomma\mathcode`\,%
  \mathcode`\,="8000 %
}
```

Make the comma character active and map it to the ‘smart comma’:

```
{\catcode`,=\active
  \gdef,{\futurelet\@let@token\sm@rtcomma}
}
```

Last but not least, define the ‘smart comma’ …

```
\def\sm@rtcomma{%
  \ifx\@let@token\@sptoken\m@thtextcomma\protect\,\else
  \ifx\@let@token\space \else
    \m@thtextcomma\fi\fi}
```

… and allow changing the period’s font:

```
\DeclareMathSymbol{.}{\mathalpha}{operators}{`.}
```

Famous last words

That’s enough code. `\endinput`.

```
\endinput
```

Change History

Index