

omtext: Semantic Markup for Mathematical Text Fragments in L^AT_EX*

Michael Kohlhase
FAU Erlangen-Nürnberg
<http://kwarc.info/kohlhase>

October 20, 2020

Abstract

The **omtext** package is part of the sT_EX collection, a version of T_EX/L^AT_EX that allows to markup T_EX/L^AT_EX documents semantically without leaving the document format, essentially turning T_EX/L^AT_EX into a document format for mathematical knowledge management (MKM).

This package supplies an infrastructure for writing OMDoc text fragments in L^AT_EX.

*Version v1.2 (last revised 2019/03/20)

Contents

| | | |
|----------|--|----------|
| 1 | Introduction | 3 |
| 2 | The User Interface | 3 |
| 2.1 | Package Options | 3 |
| 2.2 | Mathematical Text | 3 |
| 2.3 | Phrase-Level Markup | 3 |
| 2.4 | Declarations (under development) | 4 |
| 2.5 | Block-Level Markup | 5 |
| 2.6 | Index Markup | 5 |
| 2.7 | Miscellaneous | 6 |
| 3 | Limitations | 6 |
| 4 | Implementation | 7 |
| 4.1 | Package Options | 7 |
| 4.2 | Mathematical Text | 7 |
| 4.3 | Phrase-level Markup | 8 |
| 4.4 | Declarations (under development) | 9 |
| 4.5 | Block-Level Markup | 9 |
| 4.6 | Index Markup | 10 |
| 4.7 | Miscellaneous | 11 |
| 4.8 | Deprecated Functionality | 12 |

1 Introduction

The `omtext` package supplies macros and environment that allow to mark up mathematical texts in \LaTeX , a version of \TeX / \LaTeX that allows to markup \TeX / \LaTeX documents semantically without leaving the document format, essentially turning \TeX / \LaTeX into a document format for mathematical knowledge management (MKM). The package supports direct translation to the OMDoc format [Koh06]

2 The User Interface

2.1 Package Options

`showmeta` The `omtext` package takes a single option: `showmeta`. If this is set, then the metadata keys are shown (see [Koh20b] for details and customization options).
`mh` The `mh` option turns on MathHub support; see [Koh20a].

2.2 Mathematical Text

`omtext` The `omtext` environment is used for any text fragment that has a contribution to a text that needs to be marked up. It can have a title, which can be specified via the `title=` `title` key. Often it is also helpful to annotate the `type` key. The standard relations from rhetorical structure theory `abstract`, `introduction`, `conclusion`, `thesis`, `comment`, `antithesis`, `elaboration`, `motivation`, `evidence`, `transition`, `note`, `annotate` are recommended as values. Note that some of them are unary relations like `introduction`, which calls for a target. In this case, a target using the `for` key should be specified. The `transition` relation is special in that it is binary (a “transition between two statements”), so additionally, a source should be specified using the `from` key.
`display=` Note that the values of the `type` keys are often displayed as cues in the text. This can be turned off by setting the `display` key to the value `flow`. Sometimes we want to specify that a text is a continuation of another, this can be done by giving the identifier of this in the `continues=` `continues` key.
`functions=` Finally, there is a set of keys that pertain to the mathematical formulae in the text. The `functions` key allows to specify a list of identifiers that are to be interpreted as functions in the generate content markup. The `theory=` `theory` specifies a module (see [KGA20]) that is to be pre-loaded in this one¹ Finally, `verbalizes=` `verbalizes` specifies a (more) formal statement (see [Koh20c]) that this text verbalizes or paraphrases.²

2.3 Phrase-Level Markup

`\phrase` The `phrase` macro allows to mark up phrases with semantic information. It takes an optional `KeyVal` argument with the keys `verbalizes` and `type` as above and `style`, `class`, `index` that are disregarded in the \LaTeX , but copied into the gen-
`verbalizes=`
`type=`
`style`
`class`
`index`

¹EdNOTE: this is not implemented yet.

²EdNOTE: MK:specify the form of the reference.

erated content markup.

`\nlex` We use the `\nlex{⟨phrase⟩}` for marking up phrases that serve as natural
`\nlcex` language examples and `\nlcex{⟨phrase⟩}` for counter-examples (utterances that
 are not acceptable for some reason). In natural language examples, we sometimes
 use “co-reference markers” to specify the resolution of anaphora and the like. We
`\coreft` use the `\coreft{⟨phrase⟩}{⟨mark⟩}` to mark up the “target” of a co-reference and
`\corefs` analogously `\corefs` for coreference source – e.g. for an anaphoric reference. The
 usage is the following:

```
\nlex{If \coreft{a farmer}1 owns \coreft{a donkey}2,
      \corefs{he}2 beats \corefs{it}2.}
```

is formatted to

If a farmer¹ owns a donkey², he₂ beats it₂.

`\sinlinequote` The `\sinlinequote` macro allows to mark up quotes inline and attribute them.
 The quote itself is given as the argument, possibly preceded by the a specification
 of the source in a an optional argument. For instance, we would quote Hamlet
 with

```
\sinlinequote[Hamlet, \cite{Shak:1603:Hamlet}]{To be or not to be}
```

which would appear as “*To be or not to be*” Hamlet, (Shakespeare 1603) in the
 text. The style in which inline quotations appear in the text can be adapted

`\@sinlinequote` by specializing the macros `\@sinlinequote` — for quotations without source and
`\@@sinlinequote` `\@@sinlinequote` — for quotations with source.

2.4 Declarations (under development)

Declarations are special phrases that carry a lot of meaning in mathematics: they
 introduce and further specify the identifiers available in formulae. They are marked

`\vdec` up via the `\vdec` macro. Inside a declaration we can use the `\vids` macro to mark
`\vids` up the variable names and `\vrest` to mark up the restrictions. In the simplest
`\vrest` case we have a single variable as in “...for all u ”, which we mark up as.

```
... for all \vdec{\vids[u]{ $u$ }}.
```

A more complex example has multiple identifiers embedded in a restriction, as in
 “Let $x, y, z \in \mathbb{R}$, such that $x + 2y = z$, then ...”, which we mark up as

```
Let \vdec[x,y,z]{\vcond$\minset{x,y,z}\Reals$},
    such that \vrest{ $x+2y=z$ }, then \ldots''
```

345 6

³EDNOTE: explain and make better examples

⁴EDNOTE: talk with Frederic about this see what other examples there are.

⁵EDNOTE: how do we identify the variables in complex restriction patterns. maybe with LMXref,
 which we should reinstate for this.

⁶EDNOTE: document strucdec and impdec

2.5 Block-Level Markup

`sblockquote` The `sblockquote` environment is the big brother of the `\sinlinequote` macro. It also takes an optional argument to specify the source. Here the four internal macros `\begin@sblockquote` to `\end@@sblockquote` are used for styling and can be adapted by package integrators. Here a quote of Hamlet would marked up as

```
\begin{sblockquote}[Hamlet, \cite{Shak:1603:Hamlet}]\obeylines
  To be, or not to be: that is the question:
  Whether 'tis nobler in the mind to suffer
\end{sblockquote}
```

and would render as

To be, or not to be: that is the question:
Whether 'tis nobler in the mind to suffer
Hamlet, (Shakespeare 1603)

`\lec` The `\lec` macro takes one argument and sets it as a comment at the end of the line, making sure that if the content is too long it is pushed into a new line. We use it internally for placing the of source of the `sblockquote` environment above. The actual appearance of the line end comment is determined by the `@@@lec` macro, which can be customized in the document class.

2.6 Index Markup

The `omtext` package provides some extensions for the well-known indexing macros of L^AT_EX. The main reason for introducing these macros is that index markup in OMDoc wraps the indexed terms rather than just marking the spot for cross-referencing. Furthermore the index commands only indexes words unless the `noindex` option is set in the `\usepackage`. The `omtext` package and class make the usual `\index` macro undefined⁷.

`\indi` The `\indi` macro renders a word and marks it for the index. Sometimes, we want to index a slightly different form of the word, e.g. for non-standard plurals: while `\indi{word}s` works fine, we cannot use this for the word “datum”, which has the plural “data”. For this we have the macro `\aindi`, which takes another argument for the displayed text, allowing us to use `\aindi{data}{datum}`, which prints “data” but puts “datum” into the index.

The second set of macros adds an infrastructure for multi-word compounds. Take for instance the compound “OMDoc document”, which we usually want to add into the index under “OMDoc” and “document”. `\indii{OMDoc}{document}` is a variant of `\indi` that will do just this. Again, we have a version that prints a variant: This is useful for situations like this the one in Figure 1:

`\indiii` Analogously, there are variants for tree/four-word compounds: `\indiii`,
`\aindiii` `\aindiii`, `\indiv`, and `\indiv`. For instance for “wonderful OMDoc document”.
`\atwin{wonderful}{OMdoc}{document}` will make the necessary index entries un-

⁷EdNOTE: implement this and issue the respective error message

We call group `\aindii{Abelian}{Abelian}{group}`, iff `\ldots`

will result in the following

We call group Abelian, iff ...

and put “Abelian Group” into the index.

Example 1: Index markup

der “wonderful” and “document”.

`\Indi*` Finally, there are variants `\Indi`, `\Indii`, `\Indiii`, and `\Indiv` that print the capitalized version of the word complex, and `\indis`, `\indiis`, `\indiiis`, and `\indivs` that add a plurals, and ultimately `\Indis`, `\Indiis`, `\Indiiis`, and `\Indivs` that print the capitalized version of the plural.

All index macros take an optional first keyword argument: If the `loadmodules` key is given, we import the module we are in, otherwise all the currently imported modules. We do not have to require the module files, since the index is at the end of the document. If the `at` key is given, then we use that for sorting in the index.

2.7 Miscellaneous

We supply some text-level shortcuts for mathematical formulations, for instance $\hat{=}$ for “this corresponds to” or \leadsto for “therefore”. They are semantic in the sense that they are used as special words – not part of formulae, even though they look like mathematical symbols. The following table gives the full set.

`\hateq`
`\hatequiv`
`\ergo`
`\ogre`

| macro | pres. | stands for |
|------------------------|--------------|-------------------------------|
| <code>\hateq</code> | $\hat{=}$ | this corresponds to |
| <code>\hatequiv</code> | $\hat{=}$ | this statement corresponds to |
| <code>\ergo</code> | \leadsto | therefore |
| <code>\ogre</code> | \Leftarrow | because of |

3 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the `sTeX` GitHub repository [sTeX].

1. none reported yet

4 Implementation

4.1 Package Options

We declare some switches which will modify the behavior according to the package options. Generally, an option `xxx` will just set the appropriate switches to true (otherwise they stay false).⁸

```
1 \langle*package\rangle
2 \newif\if@omtext@mh@\@omtext@mh@false
3 \DeclareOption{mh}{\@omtext@mh@true}
4 \PassOptionsToPackage{\CurrentOption}{modules}
5 \newif\ifindex\indextrue
6 \DeclareOption{noindex}{\indexfalse}
7 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{modules}}
8 \ProcessOptions
9 \ifindex\makeindex\fi

10 \RequirePackage{stex-base}
11 \if@omtext@mh\RequirePackage{omtext-mh}\fi
12 \RequirePackage{xspace}
13 \RequirePackage{modules}
14 \RequirePackage{comment}
15 \RequirePackage{latexsym}
16 \RequirePackage{graphicx}
```

The next package relies on the L^AT_EX3 kernel, which L^AT_EXML only partially supports. As it is purely presentational, we only load it for PDF_{La}TeX

```
17 \if@latexml\else\RequirePackage{mdframed}\fi
```

4.2 Mathematical Text

We define the actions that are undertaken, when the keys are encountered. The first set just records metadata; this is very simple via the `\addmetakey` infrastructure [Koh20b]. Note that we allow math in the `title` field, so we do not declare it to be `Semiverbatim` (indeed not at all, which allows it by default).

```
18 \srefaddidkey{omtext}
19 \addmetakey[] {omtext} {functions}
20 \addmetakey* {omtext} {display}
21 \addmetakey {omtext} {for}
22 \addmetakey {omtext} {from}
23 \addmetakey {omtext} {type}
24 \addmetakey* {omtext} {title}
25 \addmetakey* {omtext} {start}
26 \addmetakey {omtext} {theory}
27 \addmetakey {omtext} {continues}
28 \addmetakey {omtext} {verbalizes}
29 \addmetakey {omtext} {subject}
```

⁸EDNOTE: need an implementation for L^AT_EXML

`\st@flow` We define this macro, so that we can test whether the `display` key has the value `flow`

```
30 \def\st@flow{flow}
```

We define a switch that allows us to see whether we are inside an `omtext` environment or a statement. It will be used to give better error messages for inline statements.

```
31 \newif\if@in@omtext\@in@omtextfalse
```

`omtext` The `omtext` environment can have a title, which is used in a similar way. We redefine the `\lec` macro so the trailing `\par` does not get into the way.

```
32 \def\omtext@pre@skip{\smallskip}
33 \def\omtext@post@skip{}
34 \newenvironment{omtext}[1][\@in@omtexttrue%
35 \bgroup\metasetkeys{omtext}{#1}\sref@label@id{this paragraph}%
36 \def\lec##1{\@lec{##1}}}%
37 \omtext@pre@skip\par\noindent%
38 \ifx\omtext@title\@empty%
39 \ifx\omtext@start\@empty\else%
40 \ifx\omtext@display\st@flow\omtext@start\else\stDMemph{\omtext@start}\fi\enspace%
41 \fi% end omtext@start empty
42 \else\stDMemph{\omtext@title}:\enspace%
43 \ifx\omtext@start\@empty\else\omtext@start\enspace\fi%
44 \fi% end omtext@title empty
45 \ignorespacesandpars}
46 {\egroup\omtext@post@skip\@in@omtextfalse\ignorespacesandpars}
```

4.3 Phrase-level Markup

`\phrase` For the moment, we do disregard the most of the keys

```
47 \srefaddidkey{phrase}
48 \addmetakey{phrase}{style}
49 \addmetakey{phrase}{class}
50 \addmetakey{phrase}{index}
51 \addmetakey{phrase}{verbalizes}
52 \addmetakey{phrase}{type}
53 \addmetakey{phrase}{only}
54 \newcommand\phrase[2][\metasetkeys{phrase}{#1}%
55 \ifx\phrase@only\@empty\only<\phrase@only>{#2}\else #2\fi}
```

`\coref*`

```
56 \providecommand\textsubscript[1]{\ensuremath{_{#1}}}
57 \newcommand\corefs[2]{#1\textsubscript{#2}}
58 \newcommand\coreft[2]{#1\textsuperscript{#2}}
```

`\n*lex`

```
59 \newcommand\nlex[1]{\green{\sl{#1}}}
60 \newcommand\nlcex[1]{*\green{\sl{#1}}}
```


sinlinequote

```
61 \def\@sinlinequote#1{{\sl{#1}}''}
62 \def\@@sinlinequote#1#2{\@sinlinequote{#2}~#1}
63 \newcommand\sinlinequote[2]{}
64 {\def\@opt{#1}\ifx\@opt\@empty\@sinlinequote{#2}\else\@@sinlinequote\@opt{#2}\fi}
```

4.4 Declarations (under development)

The declaration macros are still under development (i.e. the macros) are still under development and may change at any time. Currently they are completely empty.

```
65 \newcommand\vdec[2][]{#2}
66 \newcommand\vrest[2][]{#2}
67 \newcommand\vcond[2][]{#2}
```

EdN:9

\strucdec

```
9
68 \newcommand\strucdec[2][]{#2}
```

EdN:10

\impdec

```
10
69 \newcommand\impdec[2][]{#2}
```

4.5 Block-Level Markup

sblockquote

```
70 \def\begin@sblockquote{\begin{quote}\sl}
71 \def\end@sblockquote{\end{quote}}
72 \def\begin@@sblockquote#1{\begin@sblockquote}
73 \def\end@@sblockquote#1{\def\@lec##1{\textrm{##1}}\@lec{#1}\end@sblockquote}
74 \newenvironment{sblockquote}[1]{}
75 {\def\@opt{#1}\ifx\@opt\@empty\begin@sblockquote\else\begin@@sblockquote\@opt\fi}
76 {\ifx\@opt\@empty\end@sblockquote\else\end@@sblockquote\@opt\fi}
```

sboxquote

```
77 \newenvironment{sboxquote}[1]{}
78 {\def\@@src{#1}\begin{mdframed}[leftmargin=.5cm,rightmargin=.5cm]}
79 {\@lec{\textrm\@@src}\end{mdframed}}
```

The line end comment macro makes sure that it will not be forced on the next line unless necessary.

\lec

The actual appearance of the line end comment is determined by the \@@lec macro, which can be customized in the document class. The basic one here is provided so that it is not missing.

```
80 \providecommand{\@lec}[1]{(##1)}
81 \def\@lec#1{\strut\hfil\strut\null\nobreak\hfill\@lec{#1}}
82 \def\lec#1{\@lec{#1}\par}
```

⁹EdNOTE: document above

¹⁰EdNOTE: document above

4.6 Index Markup

`\omdoc@index*` These are the main internal indexing commands – dividing them into four macros is awful, but I did not get list processing running. It makes sure that the modules necessary for interpreting the math in the index entries are loaded. If the `loadmodules` key is given, we import the module we are in otherwise all the currently imported modules. We do not have to require the module files, since the index is at the end of the document. If the `at` key is given, then we use that for sorting in the index.

```

83 \addmetakey{omdoc@index}{at}
84 \addmetakey[false]{omdoc@index}{loadmodules}[true]
85 \newcommand\omdoc@indexi[2][\ifindex%
86 \metasetkeys{omdoc@index}{#1}%
87 \@bsphack\begingroup\@sanitize%
88 \protected@write\@indexfile{}\string\indexentry%
89 {\ifx\omdoc@index@at\@empty\else\omdoc@index@at @\fi%
90 \ifx\omdoc@index@loadmodules\@true%
91 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#2}%
92 \else #2\fi% loadmodules
93 }\the page}}%
94 \endgroup\@esphack\fi}%ifindex
95 \newcommand\omdoc@indexii[3][\ifindex%
96 \metasetkeys{omdoc@index}{#1}%
97 \@bsphack\begingroup\@sanitize%
98 \protected@write\@indexfile{}\string\indexentry%
99 {\ifx\omdoc@index@at\@empty\else\omdoc@index@at @\fi%
100 \ifx\omdoc@index@loadmodules\@true%
101 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#2}!%
102 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#3}!%
103 \else #2!#3\fi% loadmodules
104 }\the page}}%
105 \endgroup\@esphack\fi}%ifindex
106 \newcommand\omdoc@indexiii[4][\ifindex%
107 \metasetkeys{omdoc@index}{#1}%
108 \@bsphack\begingroup\@sanitize%
109 \protected@write\@indexfile{}\string\indexentry%
110 {\ifx\omdoc@index@at\@empty\else\omdoc@index@at @\fi%
111 \ifx\omdoc@index@loadmodules\@true%
112 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#2}!%
113 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#3}!%
114 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#4}%
115 \else #2!#3!#4\fi% loadmodules
116 }\the page}}%
117 \endgroup\@esphack\fi}%ifindex
118 \newcommand\omdoc@indexiv[5][\ifindex%
119 \metasetkeys{omdoc@index}{#1}%
120 \@bsphack\begingroup\@sanitize%
121 \protected@write\@indexfile{}\string\indexentry%
122 {\ifx\omdoc@index@at\@empty\else\omdoc@index@at @\fi%

```

```

123 \ifx\omdoc@index@loadmodules \@true%
124 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#2}!%
125 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#3}!%
126 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#4}%
127 \string\withusedmodules{\@ifundefined{mod@id}\used@modules\mod@id}{#5}%
128 \else #2!#3!#4!#5\fi% loadmodules
129 }\the page}}%
130 \endgroup\@esphack\fi}%ifindex

```

Now, we make two interface macros that make use of this:

`*indi*`

```

131 \newcommand\aindi[3] []{{#2}\omdoc@indexi[#1]{#3}}
132 \newcommand\indi[2] []{{#2}\omdoc@indexi[#1]{#2}}
133 \newcommand\indis[2] []{{#2}\omdoc@indexi[#1]{#2s}}
134 \newcommand\Indi[2] []{{\captitalize{#2}}\omdoc@indexi[#1]{#2}}
135 \newcommand\Indis[2] []{{\capitalize{#2}}\omdoc@indexi[#1]{#2s}}
136
137 \newcommand\@indii[3] []{\omdoc@indexii[#1]{#2}{#3}\omdoc@indexii[#1]{#3}{#2}}
138 \newcommand\aindii[4] []{{#2}\@indii[#1]{#3}{#4}}
139 \newcommand\indii[3] []{{#2 #3}\@indii[#1]{#2}{#3}}
140 \newcommand\indiis[3] []{{#2 #3s}\@indii[#1]{#2}{#3}}
141 \newcommand\Indii[3] []{{\captitalize{#2 #3}}\@indii[#1]{#2}{#3}}
142 \newcommand\Indiis[3] []{{\capitalize{#2 #3}}\@indii[#1]{#2}{#3}}
143
144 \newcommand\@indiii[4] []{\omdoc@indexiii[#1]{#2}{#3}{#4}\omdoc@indexii[#1]{#3}{#2 (#4)}}
145 \newcommand\aindiii[5] []{{#2}\@indiii[#1]{#3}{#4}{#5}}
146 \newcommand\indiii[4] []{{#2 #3 #4}\@indiii[#1]{#2}{#3}{#4}}
147 \newcommand\indiis[4] []{{#2 #3 #4s}\@indiii[#1]{#2}{#3}{#4}}
148 \newcommand\Indiii[4] []{{\captitalize{#2 #3 #4}}\@indiii[#1]{#2}{#3}{#4}}
149 \newcommand\Indiis[4] []{{\capitalize{#2 #3 #4s}}\@indiii[#1]{#2}{#3}{#4}}
150
151 \newcommand\@indiv[5] []{\omdoc@indexiv[#1]{#2}{#3}{#4}{#5}}
152 \newcommand\aindiv[6] []{{#2}\@indiv[#1]{#3}{#4}{#5}{#6}}
153 \newcommand\indiv[5] []{{#2 #3 #4 #5}\@indiv[#1]{#2}{#3}{#4}{#5}}
154 \newcommand\indivs[5] []{{#2 #3 #4 #5s}\@indiv[#1]{#2}{#3}{#4}{#5}}
155 \newcommand\Indiv[5] []{{\capitalize{#2 #3 #4 #5s}}\@indiv[#1]{#2}{#3}{#4}{#5}}
156 \newcommand\Indivs[5] []{{\capitalize{#2 #3 #4 #5s}}\@indiv[#1]{#2}{#3}{#4}{#5}}

```

4.7 Miscellaneous

Some shortcuts that use math symbols but are not mathematical at all; in particular, they should not be translated by L^AT_EX_ML.

```

157 \newcommand\hateq{\ensuremath{\widehat{=}}\xspace}
158 \newcommand\hatequiv{\ensuremath{\widehat{\equiv}}\xspace}
159 \@ifundefined{ergo}%
160 {\newcommand\ergo{\ensuremath{\leadsto}\xspace}}%
161 {\renewcommand\ergo{\ensuremath{\leadsto}\xspace}}%
162 \newcommand{\reflect@squig}[2]{\reflectbox{$\m@th#1\rightsquigarrow$}}%

```

```

163 \newcommand\ogre{\ensuremath{\mathrel{\mathpalette\reflect@squig\relax}}\xspace}%
164 \newcommand\notergo{\ensuremath{\not\leadsto}}
165 \newcommand\notogre{\ensuremath{\not\mathrel{\mathpalette\reflect@squig\relax}}\xspace}%

```

4.8 Deprecated Functionality

In this section we centralize old interfaces that are only partially supported any more.

`*def*`

```

166 \newcommand\indextoo[2] [] {\indi[#1]{#2}}%
167 \PackageWarning{omtext}{\protect\indextoo\space is deprecated, use \protect\indi\space instead}%
168 \newcommand\indexalt[2] [] {\aindi[#1]{#2}}%
169 \PackageWarning{omtext}{\protect\indextoo\space is deprecated, use \protect\aindi\space instead}%
170 \newcommand\twintoo[3] [] {\indii[#1]{#2}{#3}}%
171 \PackageWarning{omtext}{\protect\twintoo\space is deprecated, use \protect\indii\space instead}%
172 \newcommand\twinalt[3] [] {\aindii[#1]{#2}{#3}}%
173 \PackageWarning{omtext}{\protect\twinalt\space is deprecated, use \protect\aindii\space instead}%
174 \newcommand\atwintoo[4] [] {\indiii[#1]{#2}{#3}{#4}}%
175 \PackageWarning{omtext}{\protect\atwintoo\space is deprecated, use \protect\indiii\space instead}%
176 \newcommand\atwinalt[4] [] {\aindii[#1]{#2}{#3}{#4}}%
177 \PackageWarning{omtext}{\protect\atwinalt\space is deprecated, use \protect\aindiii\space instead}%
178 \end{package}

```

`\my*graphics`

```

179 \newcommand\mygraphics[2] [] {\includegraphics[#1]{#2}}%
180 \PackageWarning{omtext}{\protect\mygraphics\space is deprecated, use \protect\includegraphics}%
181 \newcommand\mycgraphics[2] [] {\begin{center}\mygraphics[#1]{#2}\end{center}}%
182 \PackageWarning{omtext}{\protect\mycgraphics\space is deprecated, use \protect\includegraphics}%
183 \newcommand\mybgraphics[2] [] {\fbox{\mygraphics[#1]{#2}}}%
184 \PackageWarning{omtext}{\protect\mybgraphics\space is deprecated, use \protect\includegraphics}%
185 \newcommand\mycbgraphics[2] [] {\begin{center}\fbox{\mygraphics[#1]{#2}}\end{center}}%
186 \PackageWarning{omtext}{\protect\mycbgraphics\space is deprecated, use \protect\includegraphics}%

```

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

| | | | |
|---------|----------|----------|---|
| Abelian | | group | |
| group, | <u>6</u> | Abelian, | 6 |

Change History

| | | |
|------|--|---|
| v0.4 | General: added index markup 1 | similarly 1 |
| | augmenting the index macros with optional values 1 | v0.9 General: separated out from <code>omdoc.dtx</code> 1 |
| v0.6 | General: removing <code>ttin</code> <code>emin</code> and friends, they were almost never used. 1 | v1.0 General: fixing typos 1 |
| | renmanig <code>myindex</code> to <code>omdoc</code> <code>index</code> , <code>twin</code> to <code>twin</code> , and <code>atwin</code> to <code>atwin</code> to make them package-local 1 | v1.1 General: changing <code>\textleadsto</code> to <code>\ergo</code> and defining inverse <code>\ogre</code> 1 |
| v0.7 | General: changing blockquote to <code>sblockquote</code> and <code>inlinequote</code> | moving MathHub support out to separate package 1 |
| | | v1.2 General: deprecating <code>my*graphics</code> 1 |

References

- [KGA20] Michael Kohlhase, Deyan Ginev, and Rares Ambrus. *modules.sty: Semantic Macros and Module Scoping in sTeX*. Tech. rep. 2020. URL: <https://github.com/sLaTeX/sTeX/raw/master/sty/modules/modules.pdf>.
- [Koh06] Michael Kohlhase. *OMDoc – An open markup format for mathematical documents [Version 1.2]*. LNAI 4180. Springer Verlag, Aug. 2006. URL: <http://omdoc.org/pubs/omdoc1.2.pdf>.
- [Koh20a] Michael Kohlhase. *MathHub Support for sTeX*. Tech. rep. 2020. URL: <https://github.com/sLaTeX/sTeX/raw/master/sty/mathhub/mathhub.pdf>.
- [Koh20b] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Tech. rep. 2020. URL: <https://github.com/sLaTeX/sTeX/raw/master/sty/metakeys/metakeys.pdf>.
- [Koh20c] Michael Kohlhase. *statements.sty: Structural Markup for Mathematical Statements*. Tech. rep. 2020. URL: <https://github.com/sLaTeX/sTeX/raw/master/sty/statements/statements.pdf>.
- [sTeX] *sTeX: A semantic Extension of TeX/LaTeX*. URL: <https://github.com/sLaTeX/sTeX> (visited on 05/11/2020).