tex2mn manual

Table of Contents

Reading the manual

Since `tex2mn` is a tool to transform `LaTeX` source files, some familiarity with `LaTeX` and its de-facto standard packages is assumed.

We will graphically differentiate between:

- `\standard` well-known macros which have standard behaviour, and
- **`\NonStandard`** macros which are unique to `tex2mn` and will be documented thoroughly.

NOTE In either case we will point out which `LaTeX` packages are used to define them (if any).

Code samples will *always* be presented as pairs of equivalent `LaTeX`/`AsciiDoc` listings to help you acquaint with both regardless of which side you're more familiar with:

```
What a \textbf{bold} statement.
```

```
What a *bold* statement.
```

That's it. You can confidently start reading, now!

If you happen to find any unexpected behaviour in the software or lack of clarity in the manual, please report it on our issue tracker to help us make this tool better for everyone.

Document

Document title

NOTE These features are verified by tests in test_title.rb.

You can define the document's title by using the `\title` macro *in the preamble*:

```
\documentclass{metanorma}
\title{My first document}
\begin{document}
  Document contents here.
\end{document}
```

```
= My first document

Document contents here.
```

Document attributes

NOTE These features are verified by tests in test_document_attributes.rb.

Document attributes are global metadata that can be interpreted by `metanorma` when processing the `AsciiDoc` output. Which attributes you should use depend upon the Metanorma flavour you're targeting. Attributes supported by most Metanorma flavours can be found in the generic attributes reference.

You can define a document attribute by using the `\set` macro *in the preamble*:

```
\documentclass{metanorma}
\set{my-attribute}{foobar}
\begin{document}
  Document contents here.
\end{document}
```

```
:my-attribute: foobar

Document contents here.
```

You can define a boolean attribute by using the `\set` with an empty argument:

```
\documentclass{metanorma}
\set{my-boolean-attribute}{}
\begin{document}
  Document contents here.
\end{document}
```

```
:my-boolean-attribute:

Document contents here.
```

You can use an attribute's value by using the `\get` macro:

```
\documentclass{metanorma}
\set{my-attribute}{some value}
\begin{document}
  The attribute value is \get{my-attribute}.
\end{document}
```

```
:my-attribute: some value

The attribute's value is {my-attribute}.
```

Document authors

NOTE These features are verified by tests in test_document_authors.rb.

Document authors are handled as document attributes.

You can `\set` a single author by specifying it in the preamble:

```
\documentclass{metanorma}
\set{author}{John Doe}
\begin{document}
  Document authored by \get{author}.
\end{document}
```

```
:author: John Doe

Document authored by {author}.
```

You can `\set` a multiple authors by enumerating them in the preamble:

```
\documentclass{metanorma}
\set{author_1}{Tom Sawyer}
\set{author_2}{Dick Tracy}
\set{author_3}{Harry Potter}
\begin{document}
  Document authored by \get{author_1}, \get{author_2}, and \get{author_3}.
\end{document}
```

```
:author_1: Tom Sawyer
:author_2: Dick Tracy
:author_3: Harry Potter

Document authored by {author_1}, {author_2}, and {author_3}.
```

`\mn`: TODO

Sections

NOTE These features are verified by tests in test_headings.rb.

You can perform sectioning with the usual hierarchy of macros: `\section`, `\subsection`, `\subsubsection`, `\paragraph`, and `\subparagraph`:

```
\section{My section}

\subsection{My subsection}

\subsubsection{My subsubsection}

\paragraph{My paragraph}

\subparagraph{My subparagraph}
```

```
== My section

=== My subsection

==== My subsubsection

===== My paragraph

====== My subparagraph
```

Fixed names

Metanorma relies on (case insensitive) canonical names to recognize a few standard sections:

- `Abstract`
- `Introduction`
- `Scope`
- `Normative References`

- `Terms and Definitions`
- `Symbols and Abbreviations`
- `Bibliography`

> **NOTE** `Terms and Definitions` and `Symbols and Abbreviations` expect a fixed structure, explained in the respective sections of this manual. Conversely, the heading for `Bibliography` is tipically generated automatically.

If you need to name these sections in a different way (e.g. in non-english documents), you can set the canonical name to be recognized by Metanorma as an attribute:

```
\section{Mõisted ja määratlused}
\mn{heading=terms and definitions}
```

```
[heading=terms and definitions]
== Mõisted ja määratlused
```

Foreword

> **NOTE** These features are verified by tests in test_foreword.rb.

Another special section is the foreword, which has the canonical name `Foreword` and should be used at the very beginning:

```
\section{Foreword}
First paragraph of foreword.

Second paragraph of foreword.
```

```
[[Foreword]]
.Foreword
First paragraph of foreword.

Second paragraph of foreword.
```

Just like the other standard sections, you can also pass the canonical name as an attribute:

```
\section{Prefazione}
\mn{heading=foreword}
Primo paragrafo della prefazione.

Secondo paragrafo della prefazione.
```

```
[[Foreword]]
.Prefazione
Primo paragrafo della prefazione.

Secondo paragrafo della prefazione.
```

Blank headings

To define a subclause which is numbered but bears no header text, you can use a blank name:

```
\subsection{}

This subclause bears no title text.
```

```
=== {blank}

This subclause bears no title text.
```

Deep levels

Metanorma allows headings up to seven levels. `LaTeX` has only 5 macros, so we account for the two deepest levels by attaching a **`level`** attribute to `\subparagraph`:

```
\subparagraph{Heading at level 6}
\mn{level=6}

\subparagraph{Heading at level 7}
\mn{level=7}
```

```
[level=6]
====== Heading at level 6

[level=7]
====== Heading at level 7
```

CAUTION This feature has bad ergonomics and might change before the next major release.

Appendices

NOTE These features are verified by tests in test_appendix.rb.

You can typeset appendices (i.e. annexes) by using the `\appendix` macro as it's customary in LaTeX. After you've called it all `\sections` will be treated as appendices.

```
\section{Ordinary section}

Lorem ipsum.

\appendix

\section{My first appendix}

Lorem ipsum.

\section{Another appendix}

Lorem ipsum.
```

```
== The last ordinary section

Lorem ipsum.

[appendix]
== My first appendix

Lorem ipsum.

[appendix]
== Another appendix

Lorem ipsum.
```

Just like other sections, you can pass attributes to appendices. E.g. to change the `obligation`:

```
\appendix
\section{My appendix}
\mn[obligation=informative]

Lorem ipsum.
```

```
[appendix,obligation=informative]
== My appendix

Lorem ipsum.
```

Inline headings

You can typeset inline headings by passing `\%inline-header` as an attribute:

```
\subsection{Inline header,}
\mn{\%inline-header}

which begins this line.
```

```
[%inline-header]
=== Inline header,

which begins this line.
```

CAUTION This feature has bad ergonomics and might change before the next major release.

Language and script

You can set the `language` and the `script` of a section using attributes:

```
\section{French section}
\mn{language=fr}

\section{Math section}
\mn{script=Zmth}
```

```
[language=fr]
== French section

[script=Zmth]
== Math section
```

Obligations

You can set the `obligation` of a section (which can be either `informative` or `normative`) using an attribute:

```
\section{Informative section}
\mn{obligation=informative}
```

```
[obligation=informative]
== Informative section
```

Note that most sections have a fixed default. Annexes and clauses default to `normative` but you can set them to `informative`.

Symbols and abbreviations

`Symbols and Abbreviations` sections are expected to be simple description lists.

Metanorma takes care of sorting the symbol entries in the order prescribed by ISO/IEC DIR 2, but it does not support sorting LaTeX math entries. Also note that the PDF rendered directly by LaTeX will not be sorted.

Terms and definitions

`Terms and Definitions` subsections must be composed by these elements, in order:

1. a heading of the appropriate level containing the term
2. an optional `\label` to cross-reference the term from the others

3. these optional macros, *whose parameter can contain markup*:

   `\alt`

   to specify alternative/admitted terms

   `\deprecated`

   to specify deprecated terms

   `\domain`

   to specify the term domain

4. the term definition as a normal paragraph
5. optional examples using the **`example`** environment
6. optional notes using the **`note`** environment
7. an optional citation using the **`source`** environment composed by a bibliographic reference followed by optional modification details

Here is a full example:

```
\section{Terms and Definitions}

\subsection{paddy}
\label{paddy}
\alt{paddy rice}
\alt{rough \textbf{rice}}
\deprecated{cargo rice}
\domain{rice}

rice retaining its husk after threshing

\begin{example}
  Foreign seeds, husks, bran, sand, dust.
\end{example}

\begin{note}
  The starch of waxy rice consists almost entirely
of amylopectin. The kernels have a tendency to
stick together after cooking.
\end{note}

\begin{source}
  \mncite[section 3.2]{ISO7301}, The term "cargo
rice" is shown as deprecated, and Note 1 to entry
is not included here
\end{source}
```

```
== Terms and Definitions

[[paddy]]
=== paddy
alt:[paddy rice]
alt:[rough **rice**]
deprecated:[cargo rice]
domain:[rice]

rice retaining its husk after threshing

[example]
Foreign seeds, husks, bran, sand, dust.

NOTE: The starch of waxy rice consists almost
entirely of amylopectin. The kernels have a
tendency to stick together after cooking.

[.source]
<<ISO7301,section 3.2>>, The term "cargo rice" is
shown as deprecated,
and Note 1 to entry is not included here
```

To treat a subsection of `Terms and Definitions` as a normal subsection instead of a term, e.g. an introductory section, you apply the **`.nonterm`** attribute to it as follows:

```
\section{Terms and definitions}

\subsection{Introduction}
\mn{.nonterm}

The following terms have non-normative effect, and
should be ignored by the ametrical.
```

```
== Terms and definitions

[.nonterm]
=== Introduction
The following terms have non-normative effect, and
should be ignored by the ametrical.
```

Text formatting

NOTE These features are verified by tests in test_text_formatting.rb.

You can set bold, italic, monospace and small caps text using respectively `\textbf`, `\textit`, `\texttt` and `\textsc`.

```
\textbf{Bold} text.

\textit{Italic} text.

\texttt{Monospace} text.

\textsc{Small caps} text.
```

```
**Bold** text.

__Italic__ text.

``Monospace`` text.

[smallcap]#Small caps# text.
```

The *switch* versions of these macros are available too: `\bfseries`, `\itshape`, `\ttfamily` and `\scshape`.

```
{\bfseries Bold} text.

{\itshape Italic} text.

{\ttfamily Monospace} text.

{\scshape Small caps} text.
```

```
**Bold** text.

__Italic__ text.

``Monospace`` text.

[smallcap]#Small caps# text.
```

You can set strikethrough, superscript and subscript text using respectively **`\textst`**, `\textsuperscript` and `\textsubscript`.

```
Strikethrough \textst{text}.

Superscript \textsuperscript{text}.

Subscript \textsubscript{text}.
```

```
Strikethrough [strike]#text#.

Superscript ^text^.

Subscript ~text~.
```

NOTE Strikethrough text is implemented using the ulem package.

Paragraph alignment

NOTE These features are verified by tests in test_paragraph_alignment.rb.

You can set paragraph alignment explicitly using the `flushleft`, `center` and `flushright` environments.

```
\begin{flushleft}
  This paragraph is left aligned.
\end{flushleft}
```

```
[align=left]
This paragraph is left aligned.
```

```
\begin{center}
  This paragraph is centered.
\end{center}
```

```
[align=center]
This paragraph is centered.
```

```
\begin{flushright}
  This paragraph is right aligned.
\end{flushright}
```

```
[align=right]
This paragraph is right aligned.
```

The default alignment is *justified*, so no environment is provided for that.

Cross-references (aka internal references)

NOTE These features are verified by tests in test_cross_references.rb.

To reference some part of the document you must first of all label it using `\label`. Many parts of the document accept labels (headings, tables, list items, etc.) and you will find an example in the related section when that's the case.

For the sake of the next examples, let's assume a section labeled `sec:foo` exists.

You can reference `sec:foo` by using the `\ref` macro:

```
Go look at \ref{sec:foo}.
```

```
Go look at <<sec:foo>>.
```

To reference `sec:foo` with a custom text you can use the `\hyperref` macro:

```
Go look at \hyperref[sec:foo]{this}.
```

```
Go look at <<sec:foo, this>>.
```

Links (aka external references)

NOTE These features are verified by tests in test_links.rb.

References can also point to URLs identifying resources external to the document.

You can link to a website by using the `\url` macro:

```
Go look at \url{https://www.metanorma.com/}.
```

```
Go look at link:++https://www.metanorma.com/++[].
```

To link to a website with a custom text you can use the `\href` macro:

```
Go look at \href{https://www.metanorma.com/
}{Metanorma home}.
```

```
Go look at
link:++https://www.metanorma.com/++[Metanorma
home].
```

Bibliography and citations

Bibliography

NOTE These features are verified by tests in test_bibliography.rb.

You can cite a bibliographic source using the `\cite` macro as usual.

CAUTION Multiple citations, e.g. `\cite{KEY1,KEY2}` are not supported.

The bibliographic database can be specified in two different ways.

If the bibliography is small or you want to typeset each entry manually you can use the `thebibliography` environment to wrap a list of bibliographic items described by `\bibitem`.

```
\documentclass{metanorma}

\begin{document}

  This document cites \cite{ISO7301}.
  It also cites \cite{einstein}.

  \begin{thebibliography}{2}
    \bibitem[ISO 7301]{ISO7301}
    Rice -- Specification

    \bibitem{einstein}
    Albert Einstein.
    \textit{Zur Elektrodynamik bewegter K{\"o}rper
}. (German)
    [\textit{On the electrodynamics of moving
bodies}].
    Annalen der Physik, 322(10):891▯921, 1905.
  \end{thebibliography}
\end{document}
```

```
This document cites <<ISO7301>>.
It also cites <<einstein>>.

[bibliography]
== References

* [[[ISO7301,ISO 7301]]]
Rice ▯ Specification

* [[[einstein,2]]]
Albert Einstein.
__Zur Elektrodynamik bewegter Körper__. (German)
[__On the electrodynamics of moving bodies__].
Annalen der Physik, 322(10):891▯921, 1905.
```

If the bibliography is big or you have an existing BibTeX database, you can use that too by passing the filename to the `\bibliography` macro.

```
\documentclass{metanorma}

\begin{document}
  Everyone should read \cite{einstein}.
  Many interesting books are found in \cite
{knuthwebsite}.

  \bibliography{my_database.bib}
\end{document}
```

Contents of `my_database.bib`

```
@article{einstein,
    author =  "Albert Einstein",
    title =   "{Zur Elektrodynamik bewegter
K{\"o}rper}. ({German})
        [{On} lectrodynamics of moving bodies]",
    journal = "Annalen der Physik",
    volume =  "322",
    number =  "10",
    pages =   "891--921",
    year =    "1905",
    DOI =
"http://dx.doi.org/10.1002/andp.19053221004"
}

@misc{knuthwebsite,
    author = "Donald Knuth",
    title  = "Knuth: Computers and Typesetting",
    url    = "http://www-cs-
faculty.stanford.edu/\~{}uno/abcde.html"
}
```

```
Everyone should read <<einstein>>.
Many interesting books are found in
<<knuthwebsite>>.

[bibliography]
== References

* [[[einstein,1]]]
{blank}A. Einstein (1905)
Zur Elektrodynamik bewegter Körper. (German) [On
the electrodynamics of moving bodies].
Annalen der Physik322 (10),  pp. 891▯921.
External Links:
http://dx.doi.org/10.1002/andp.19053221004[Document
]

* [[[knuthwebsite,3]]]
{blank}D. Knuth
Knuth: computers and typesetting.
External Links: http://www-cs-
faculty.stanford.edu/\~uno/abcde.html[Link]
```

CAUTION While you can use bibliographic styles on the `LaTeX` side and Relaton databases on the `Adoc` side, these features are still not supported on the opposing side.

Advanced citations

NOTE These features are verified by tests in test_mncite.rb.

To leverage the full power of Metanorma you can use **`\mncite`** to typeset bibliographic references.

`\mncite` accepts three arguments as `\mncite[<1>]{<2>}[<3>]`:

1. the first argument (optional) is the citation which will be rendered literally;
2. the second argument (mandatory) is the bibliographic key;
3. the third argument (optional) is a list of localities.

```
\mncite{ISO712}

\mncite[the foregoing reference]{ISO712}

\mncite{ISO712}[section=5,page=8-10]

\mncite[5:8-10]{ISO712}[section=5,page=8-10]
```

```
<<ISO712>>

<<ISO712,the foregoing reference>>

<<ISO712,section=5,page=8-10>>

<<ISO712,section=5,page=8-10,5:8-10>>
```

CAUTION Multiple citations, e.g. `\cite{KEY1,KEY2}` are not supported.

CAUTION Localities are not rendered by the LaTeX class yet, only by Metanorma compilation.

Block quotations

NOTE These features are verified by tests in test_block_quotations.rb.

You can typeset block quotations using the `quote` environment:

```
\begin{quote}
  This is a block quotation.
\end{quote}
```

```
  [quote]
  ____
  This is a block quotation.
  ____
```

The attribution and the citation title can be passed through to AsciiDoc by using the **`\mn`** macro:

```
\begin{quote}
  \mn{ISO,"ISO7301, section 1"}
  This is a block quotation.
\end{quote}
```

```
 [quote,ISO,"ISO7301, section 1"]
 ____
 This is a block quotation.
 ____
```

CAUTION This feature is not complete; attribution and citation title are currently not rendered by `LaTeX`.

Lists

All usual typologies of lists available in `LaTeX` can be used; as usual, they work by listing items usinf the `\item` macro inside a specific environment.

Ordered lists

NOTE These features are verified by tests in test_ordered_lists.rb.

You can typeset ordered lists using the `enumerate` environment:

```
\begin{enumerate}
  \item Hey,
  \item ho,
  \item let's go!
\end{enumerate}
```

```
. Hey,
. ho,
. let□s go!
```

Unordered lists

NOTE These features are verified by tests in test_unordered_lists.rb.

You can typeset unordered lists using the `itemize` environment:

```
\begin{itemize}
  \item Foo
  \item Bar
  \item Baz
\end{itemize}
```

```
* Foo
* Bar
* Baz
```

Description lists

NOTE These features are verified by tests in test_description_lists.rb.

You can typeset description lists using the `description` environment:

```
\begin{description}
  \item[Lee] bass guitar and lead vocals
  \item[Lifeson] guitars and backing vocals
  \item[Peart] drums and percussion
\end{description}
```

```
Lee:: bass guitar and lead vocals
Lifeson:: guitars and backing vocals
Peart:: drums and percussion
```

Nested lists

NOTE These features are verified by tests in test_stressed_lists.rb.

You can nest ordered, unordered and description lists freely.

CAUTION While `AsciiDoc` has no depth limit, `LaTeX` is limited to a depth of 4.

Math

NOTE These features are verified by tests in test_math_inline.rb and test_math_display.rb.

You can typeset inline math delimiting it with `S`/`$` pairs:

```
Euler's identity is $e^{i\pi}+1=0$.
```

```
Euler's identity is stem:[e^{i\pi}+1=0].
```

You can typeset display math delimiting it with `\[`/`\]` pairs:

```
Here is the Binomial Theorem:
\[(1+x)^n = \sum_{k=0}^n {n \choose k}x^k\]
```

```
Here is the Binomial Theorem:

[stem]
++++
(1+x)^n = \sum_{k=0}^n {n \choose k}x^k
++++
```

Tables

NOTE These features are verified by tests in test_tables.rb.

You can typeset simple tables using the `tabular` environment:

```
\begin{tabular}{ll}
  A & B \\
  C & D \\
\end{tabular}
```

```
[cols=2*]
|===
| A
| B

| C
| D
|===
```

You can add labels and captions by wrapping a `tabular` with the `table` environment and using the `\label` and `\caption` macros:

```
\begin{table}
  \label{tab:example}
  \caption{This is the caption}
  \begin{tabular}{ll}
    A & B \\
    C & D \\
  \end{tabular}
\end{table}
```

```
[[tab:example]]
.This is the caption
[cols=2*]
|===
| A
| B

| C
| D
|===
```

Figures

NOTE These features are verified by tests in test_figures.rb.

You can typeset figures by using the `figure` environment.

Use `\includegraphics` to include an image.

Use `\label` and `\caption` to add labels and captions to figures.

```
\begin{figure}
  \label{fig:example}
  \caption{This is the caption}
  \includegraphics{example.jpg}
\end{figure}
```

```
[[fig:example]]
.This is the caption
image::example.jpg[]
```

Subfigures

NOTE These features are verified by tests in test_subfigures.rb.

To typeset a figure containing subfigures you can nest the `subfigure` environment inside a `figure`.

The macros `\caption` and `\label` work as you would expect inside subfigures too.

```
\begin{figure}\centering
  \begin{subfigure}[b]{0.4\textwidth}
    \includegraphics[width=\textwidth]{left.jpg}
    \caption{A subfigure on the left}
    \label{fig:left}
  \end{subfigure}
  \qquad
  \begin{subfigure}[b]{0.4\textwidth}
    \includegraphics[width=\textwidth]{right.jpg}
    \caption{A subfigure on the right}
    \label{fig:right}
  \end{subfigure}
  \caption{A figure with two subfigures}
  \label{fig:whole}
\end{figure}
```

```
[[fig:whole]]
.A figure with two subfigures
====
[[fig:left]]
.A subfigure on the left
image::left.jpg[]

[[fig:right]]
.A subfigure on the right
image::right.jpg[]

====
```

NOTE Subfigures are implemented using the subcaption package.

Keys

NOTE These features are verified by tests in test_figures_key.rb.

To typeset the key for a figure you can use a **`key`** environment (which behaves like a `description`) inside a `\paragraph` titled `Key`.

```
\begin{figure}
  \includegraphics{example.jpg}
  \label{fig:example}
  \caption{This is the caption}
  \paragraph*{Key}
  \begin{key}
    \item[A] First letter
    \item[B] Second letter
    \item[C] Third letter
  \end{key}
\end{figure}
```

```
[[fig:example]]
.This is the caption
image::example.jpg[]

*Key*

A::
First letter
B::
Second letter
C::
Third letter
```

NOTE Figures keys are implemented using the enumitem package.

Footnotes

NOTE These features are verified by tests in test_footnotes.rb.

You can typeset footnotes by using `\footnote`:

```
Footnotes are useful\footnote{Unless abused.}.
```

```
Footnotes are useful footnote:[Unless abused.].
```

Requirements, Recommendations, and Permissions

NOTE These features are verified by tests in test_blocks.rb.

You can typeset requirements, recommendations, and permissions by using the homonymous environments `requirement`, `recommendation` and `permission`:

```
\begin{requirement}
  This is a requirement block.
\end{requirement}
```

```
[.requirement]
====
This is a requirement block.
====
```

```
\begin{recommendation}
  This is a recommendation block.
\end{recommendation}
```

```
[.recommendation]
====
This is a recommendation block.
====
```

```
\begin{permission}
  This is a permission block.
\end{permission}
```

```
[.permission]
====
This is a permission block.
====
```

Attributes

CAUTION TODO

Nesting

Requirements, recommendations and permissions can be nested:

```
\begin{requirement}
  Here is a nested requirement.

  \begin{requirement}
    First nested requirement.
  \end{requirement}

  Here is another nested requirement.

  \begin{requirement}
    Second nested requirement.
  \end{requirement}
\end{requirement}
```

```
[.requirement]
====
Here is a nested requirement.

[.requirement]
=====
First nested requirement.
=====

Here is another nested requirement.

[.requirement]
======
Second nested requirement.
======
====
```

Internal structure

The internal structure of requirements, recommendations and permissions can be further marked up with environments which will help making the document machine-readable. Such environments are: **`specification`**, **`measurement-target`**, **`verification`**, and **`import`**.

```
\begin{requirement}
  This is descriptive text.

  \begin{specification}
    This is a formal statement, which may be
considered the object of the requirement.
  \end{specification}

  \begin{measurement-target}
    This is a quantitative measure for the
requirement.
  \end{measurement-target}

  \begin{verification}
    This is a verification step for the requirement
  \end{verification}

  \begin{import}
    This is a code stub.
  \end{import}
\end{requirement}
```

```
[.requirement]
====
This is descriptive text.

[.specification]
--
This is a formal statement, which may be considered
the object of the requirement.
--

[.measurement-target]
--
This is a quantitative measure for the requirement.
--

[.verification]
--
This is a verification step for the requirement
--

[.import]
--
This is a code stub.
--
====
```

Reviewer notes

NOTE These features are verified by tests in test_reviewer_notes.rb.

You can typeset reviewer notes by using the **`reviewernotes`** environment.

By using **`\mn`** at the beginning of the environment you can set the following attributes:

`reviewer` (*mandatory*)

Name of the reviewer

`from` (*mandatory*)

Starting target anchor

`to` (*optional*)

Ending target anchor

`date` (*optional*)

Date attribute (optionally including time)

Here is a full example:

```
\begin{reviewernote}
  \mn{reviewer="John
Doe",date=20180125T0121,from=A,to=B}
  Lorem ipsum.
\end{reviewernote}
```

```
[reviewer="John
Doe",date=20180125T0121,from=A,to=B]
****
Lorem ipsum.
****
```

CAUTION The `reviewer` and `date` attributes are currently not rendered by LaTeX but only passed to Adoc.

Todos

NOTE These features are verified by tests in test_todos.rb.

You can typeset todo notes by using the **`todo`** environment.

By using **`\mn`** at the beginning of the environment you can set the following attributes:

`reviewer` (*optional*)

Name of the reviewer

`from` (*optional*)

Starting target anchor

`to` (*optional*)

Ending target anchor

`date` (*optional*)

Date attribute (optionally including time)

Here is a full example:

```latex
\begin{todo}
  \mn{reviewer="John
Doe",date=20180125T0121,from=A,to=B}
  Lorem ipsum.
\end{todo}
```

```
[TODO,reviewer="John
Doe",date=20180125T0121,from=A,to=B]
====
Lorem ipsum.
====
```

CAUTION

The `reviewer` and `date` attributes are currently not rendered by LaTeX but only passed to Adoc.