

CNX \LaTeX : A \LaTeX -based Syntax for Connexions Modules*

Michael Kohlhasse
Jacobs University, Bremen
<http://kwarc.info/kohlhasse>

April 6, 2016

Abstract

We present CNX \LaTeX , a collection of \LaTeX macros that allow to write CONNEXIONS modules without leaving the \LaTeX workflow. Modules are authored in CNX \LaTeX using only a text editor, transformed to PDF and proofread as usual. In particular, the \LaTeX workflow is independent of having access to the CONNEXIONS system, which makes CNX \LaTeX attractive for the initial version of single-author modules.

For publication, CNX \LaTeX modules are transformed to CNXML via the \LaTeX XML translator and can be uploaded to the CONNEXIONS system.

*Version ? (last revised ?)

Contents

1	Introduction	3
2	The User Interface	3
2.1	Package Options	3
2.2	Document Structure	3
2.3	Mathematics	4
2.4	Statements	4
2.5	Connexions: Links and Cross-References	5
2.6	Metadata	6
2.7	Exercises	7
2.8	Graphics, etc.	7
3	Limitations	7
4	The Implementation	8
4.1	Package Options	8
4.2	Document Structure	8
4.3	Mathematics	9
4.4	Rich Text	10
4.5	Statements	10
4.6	Connexions	11
4.7	Metadata	12

1 Introduction

The Connexions project is a¹

The CNXML format — in particular the embedded content MATHML — is hard to write by hand, so we provide a set of environments that allow to embed the CNXML document model into L^AT_EX.

2 The User Interface

This document is not a manual for the Connexions XML encoding, or a practical guide how to write Connexions modules. We only document the L^AT_EX bindings for CNXML and will presuppose experience with the format or familiarity with². Note that formatting CNX^LA_TE_X documents with the L^AT_EX formatter does little to enforce the restrictions imposed by the CNXML document model. You will need to run the L^AT_EX_{ML} converter for that (it includes DTD validation) and any CNX-specific quality assurance tools after that.³

The CNX^LA_TE_X class makes heavy use of the KeyVal package, which is part of your L^AT_EX distribution. This allows to add optional information to L^AT_EX macros in the form of key-value pairs: A macro `\foo` that takes a KeyVal argument and a regular one, so a call might look like `\foo{bar}` (no KeyVal information given) or `\foo[key1=val1,...,keyn=valn]{bar}`, where `key1,...,keyn` are predefined keywords and values are L^AT_EX token sequences that do not contain comma characters (though they may contain blank characters). If a value needs to contain commas, then it must be enclosed in curly braces, as in `\foo[args={a,comma,separated,list}]`. Note that the order the key/value pairs appear in a KeyVal Argument is immaterial.

2.1 Package Options

`showmeta` The `cnx` package takes a single option: `showmeta`. If this is set, then the metadata keys are shown (see [Koh15a] for details and customization options).

2.2 Document Structure

The first set of CNX^LA_TE_X environments concern the top-level structure of the modules. The minimal Connexions document in L^AT_EX can be seen in Figure 1: we still need the L^AT_EX document environment, then the `cnxmodule` environment contains the module-specific information as a KeyVal argument with the two keys: `id` for the module identifier supplied by the CONNEXIONS system) and `name` for the title of the module.

`ccontent` The `content` environment delineates the module content from the metadata (see Section 2.6). It is needed to make the conversion to CNXML simpler.

`c*section` CNXML knows three levels of sectioning, so the CNX^LA_TE_X class supplies three

¹EDNOTE: continue; copy from somewhere...

²EDNOTE: cite the relevant stuff here

³EDNOTE: talk about Content MATHML and cmathml.sty somewhere

```

\documentclass{cnx}
\begin{document}
  \begin{cnxmodule}[name=Hello World,id=m4711]
    \begin{ccontent}
      \begin{cpara}[id=p01] Hello World\end{cpara}
    \end{ccontent}
  \end{cnxmodule}
\end{document}

```

Example 1: A Minimal CNX \LaTeX Document

as well: `csection`, `csubsection` and `csubsubsection`. In contrast to regular \LaTeX , these are environments to keep the tight connection between the formats. These environments take an optional KeyVal argument with key `id` for the identifier and a regular argument for the title of the section (to be transformed into the CNXML `name` element).

`cpara`, `cnote` The lowest levels of the document structure are given by paragraphs and notes. The `cpara` and `cnote` environment take a KeyVal argument with the `id` key for identification, the latter also allows a `type` key for the note type (an unspecified string⁴).

2.3 Mathematics

Mathematical formulae are integrated into text via the \LaTeX math mode, i.e. wrapped in `$` characters or between `\(` and `\)` for inline mathematics and wrapped in `$$` or between `\[` and `\]` for display-style math. Note that CNXML expects Content MATHML as the representation format for mathematical formulae, while run-of-the-mill \LaTeX only specifies the presentation (i.e. the two-dimensional layout of formulae). The \LaTeX XML converter can usually figure out some of the content MATHML from regular \LaTeX , in other cases, the author has to specify it e.g. using the infrastructure supplied by the `cmathml` package.

`cequation` For numbered equations, CNXML supplies the `equation` element, for which CNX \LaTeX provides the `cequation` environment. This environment takes a KeyVal argument with the `id` key for the (required) identifier.

2.4 Statements

CNXML provides special elements that make various types of claims; we collectively call them statements.

`cexample` The `cexample` environment and `definition` elements take a KeyVal argument with key `id` for identification.

`crule`, `statement`, `proof` In CNXML, the `rule` element is used to represent a general assertion about the state of the world. The CNX \LaTeX `rule`⁵ environment is its CNX \LaTeX coun-

⁴EDNOTE: what are good values?

⁵EDNOTE: we have called this “`crule`”, since “`rule`” is already used by \TeX .

terpart. It takes a KeyVal attribute with the keys `id` for identification, `type` to specify the type of the assertion (e.g. “Theorem”, “Lemma” or “Conjecture”), and `name`, if the assertion has a title. The body of the `crule` environment contains the statement of assertion in the `statement` environment and (optionally) a proof in the `proof` environment. Both take a KeyVal argument with an `id` key for identification.

```
\begin{crule}[id=prop1,type=Proposition]
  \begin{statement}[id=prop1s]
    Sample statement
  \end{statement}
  \begin{proof}[id=prop1p]
    Your favourite proof
  \end{proof}
\end{crule}
```

Example 2: A Basic `crule` Example

`definition`, `cmeaning`

A definition defines a new technical term or concept for later use. The `definition` environment takes a KeyVal argument with the keys `id` for identification and `term` for the concept (definiendum) defined in this form. The definition text is given in the `cmeaning` environment¹, which takes a KeyVal argument with key `id` for identification. After the `cmeaning` environment, a `definition` can contain arbitrarily many `cexamples`.

```
\begin{definition}{term=term-to-be-defined, id=termi-def]
  \begin{cmeaning}[id=termi-meaning]
    {\term{Term-to-be-defined}} is defined as: Sample meaning
  \end{cmeaning}
\end{definition}
```

Example 3: A Basic `definition` and `cmeaning` Example

2.5 Connexions: Links and Cross-References

As the name `CONNEXIONS` already suggests, links and cross-references are very important for `CONNEXIONS` modules. `CNXML` provides three kinds of them. Module links, hyperlinks, and concept references.

`cnxn`

Module links are specified by the `\cnxn` macro, which takes a keyval argument with the keys `document`, `target`, and `strength`. The `document` key allows to specify the module identifier of the desired module in the repository, if it is empty, then the current module is intended. The `target` key allows to specify the document fragment. Its value is the respective identifier (given by its `id` attribute in

¹we have called this `cmeaning`, since `meaning` is already taken by `TeX`

CNXML or the `id` key of the corresponding environment in CNX \LaTeX). Finally, the `strength` key allows to specify the relevance of the link.

- `link` The regular argument of the `\cnxn` macro is used to supply the link text. Hyperlinks can be specified by the `\link` macro in CNX \LaTeX . It takes a KeyVal argument with the key `src` to specify the URL of the link. The regular argument of the `\link` macro is used to supply the link text.
- `term` The `\term` marco can be used to specify the⁶

2.6 Metadata

Metadata is mostly managed by the system in CONNEXIONS, so we often do not need to care about it. On the other hand, it influences the system, so if we have work on the module extensively before converting it to CNXML, it may be worth-wile specify some of the data in advance.

```
\begin{metadata}[version=2.19,
                  created=2000/07/21,revised=2004/08/17 22:07:27.213 GMT-5]
\begin{authorlist}
  \cnxauthor[id=miko,firstname=Michael,surname=Kohlhase,
             email=m.kohlhase@iu-bremen.de]
\end{authorlist}
\begin{keywordlist}\keyword{Hello}\end{keywordlist}
\begin{cnxabstract}
  A Minimal CNX $\text{\LaTeX}$  Document
\end{cnxabstract}
\end{metadata}
```

Example 4: Typical CNX \LaTeX Metadata

- `metadata` The `metadata` environment takes a KeyVal argument with the keys `version`, `created`, and `revised` with the obvious meanings. The latter keys take ISO 8601 norm representations for dates and times. Concretely, the format is `CCYY-MM-DDThh:mm:ss` where “CC” represents the century, “YY” the year, “MM” the month, and “DD” the day, preceded by an optional leading “-” sign to indicate a negative number. If the sign is omitted, “+” is assumed. The letter “T” is the date/time separator and “hh”, “mm”, “ss” represent hour, minutes, and seconds respectively.
- `authorlist, maintainerlist` The lists of authors and maintainers can be specified in the `authorlist` and `maintainerlist` environments, which take no arguments.
- `cnxauthor, maintainer` The entries on this lists are specified by the `\cnxauthor` and `\maintainer` macros. Which take a KeyVal argument specifying the individual. The `id` key is the identifier for the person, the `honorific`, `firstname`, `other`, `surname`, and `lineage` keys are used to specify the various name parts, and the `email` key is used to specify the e-mail address of the person.
- `keywordlist, keyword` The keywords are specified with a list of `keyword` macros, which take the

⁶EdNOTE: continue, pending Chuck’s investigation.

respective keyword in their only argument, inside a **keyword** environment. Neither take any KeyVal arguments.

cnxabstract The abstract of a CONNEXIONS module is considered to be part of the meta-data. It is specified using the **cnxabstract** environment. It does not take any arguments.

2.7 Exercises

cexercise, **cproblem**, **csolution** An exercise or problem in CONNEXIONS is specified by the **cexercise** environment, which takes an optional keyval argument with the keys **id** and **name**. It must contain a **cproblem** environment for the problem statement and a (possibly) empty set of **csolution** environments. Both of these take an optional keyval argument with the key **id**.

2.8 Graphics, etc.

EdN:7

cfigure For graphics we will use the **cfigure**⁷ macro, which provides a non-floating environment for including graphics into CNXML files. **cfigure** takes three arguments first an optional CNXML keys, then the keys of the **graphicx** package in a regular argument (leave that empty if you don't have any) and finally a path. So

```
\cfigure[id=foo,type=image/jpeg,caption=The first F00]{width=7cm,height=2cm}{../images/f
```

EdN:8

Would include a graphic from the file at the path `../images/foo`, equip this image with a caption, and tell L^AT_EX that⁸ the original of the images has the MIME type `image/jpeg`.

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 sT_EX GitHub repository [sTeX].

1. none reported yet

⁷EdNOTE: probably better call it `cgraphics`

⁸EdNOTE: err, exactly what does it tell latexml?

4 The 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). First we have the general options

```
1 <*cls>
2 \DeclareOption{showmeta}{\PassOptionsToPackage{\CurrentOption}{metakeys}}
3 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{omdoc}}
```

Finally, we need to declare the end of the option declaration section to \LaTeX .

```
4 \ProcessOptions
```

We first make sure that the `sref` [Koh15b] and `graphicx` packages are loaded.

```
5 \RequirePackage{sref}
6 \RequirePackage{graphicx}
```

The next step is to declare (a few) class options that handle the paper size; this is useful for printing.

```
7 \DeclareOption{letterpaper}
8   {\setlength\paperheight {11in}%
9    \setlength\paperwidth  {8.5in}}
10 \DeclareOption{a4paper}
11   {\setlength\paperheight {297mm}%
12    \setlength\paperwidth  {210mm}}
13 \ExecuteOptions{letterpaper}
14 \ProcessOptions
```

Finally, we input all the usual size settings. There is no sense to use something else, and we initialize the page numbering counter and tell it to output the numbers in arabic numerals (otherwise label and reference do not work).

```
15 \input{size10.clo}
16 \pagenumbering{roman}
```

4.2 Document Structure

Now, we start with the document structure markup. The `cnxmodule` environment does not add anything to the \LaTeX output, it's attributes only show up in the XML. There we have a slight complication: we have to put an `id` attribute on the `document` element in CNXML, but we cannot redefine the `document` environment in \LaTeX . Therefore we specify the information in the `cnxmodule` environment. This means however that we have to put in on the `document` element when we are already past this. The solution here is that when we parse the `cnxmodule` environment, we store the value and put it on the `document` element when we leave the `document` environment (thanks for Ioan Sucan for the code).

`cnxmodule`

```
17 \addmetakey{cnxmodule}{name}
18 \srefaddidkey{cnxmodule}{id}
```



```

19 \newenvironment{cnxmodule}[1] [] {\metasetkeys{cnxmodule}{#1}}{}

ccontent The ccontent environment is only used for transformation. Its optional id at-
tribute is not taken up in the LATEX bindings.
20 \newenvironment{ccontent}{}{}

c*section The sectioning environments employ the obvious nested set of counters.
21 \newcounter{section}
22 \srefaddidkey{sectioning}{id}
23 \newenvironment{csection}[2] []%
24 {\stepcounter{section}\strut\ [.5ex]\noindent%
25 {\Large\bfseries\arabic{section}.~{#2}}\ [.5ex]
26 \metasetkeys{sectioning}{#1}}
27 {}
28 \newcounter{subsection}[section]
29 \newenvironment{csubsection}[2] []
30 {\refstepcounter{subsection}\strut\ [1ex]\noindent%
31 {\large\bfseries\arabic{section}.\arabic{subsection}.~#2\ [1ex]}}%
32 \metasetkeys{sectioning}{#1}}%
33 {}
34 \newcounter{subsubsection}[subsection]
35 \newenvironment{csubsubsection}[2] []
36 {\refstepcounter{subsubsection}\strut\ [.5ex]\noindent
37 {\bfseries\arabic{section}.\arabic{subsection}.\arabic{subsubsecction}~#2\ [.5ex]}}%
38 \metasetkeys{sectioning}{#1}}{}

cpara For the <cnx:para> element we have to do some work, since we want them to be
numbered. This handling is adapted from Bruce Miller's LATEX.ltxml numbered.
39 \srefaddidkey{para}{id}
40 \newenvironment{cpara}[1] [] {\metasetkeys{para}{#1}}{\par}

cnote
41 \srefaddidkey{note}
42 \addmetakey{note}{type}
43 \newenvironment{cnote}[1] []%
44 {\metasetkeys{note}{#1}\par\noindent\strut\hfill\begin{minipage}{10cm}{\bfseries\note@type:~}%
45 {\end{minipage}\hfill\strut\par}

4.3 Mathematics

cequation
46 \srefaddidkey{equation}{id}
47 \newenvironment{cequation}[1] []%
48 {\metasetkeys{equation}{#1}\begin{displaymath}}
49 {\end{displaymath}}

```

4.4 Rich Text

In this section, we redefine some of L^AT_EX commands that have their counterparts in CNXML.

quote

```
50 \srefaddidkey{cquote}
51 \addmetakey{cquote}{type}
52 \addmetakey{cquote}{src}
53 \newenvironment{cquote}[1] [] {%
54 \metasetkeys{cquote}{#1}\begin{center}\begin{minipage}{.8\textwidth}}{\end{minipage}\end{center}}
```

displaymath We redefine the abbreviate display math envionment to use the CNXML equation tags, everything else stays the same.⁹

```
55 \newcommand\litem[2] [] {\item[#1]\label{#2}}
```

4.5 Statements

cexample

```
56 \srefaddidkey{example}
57 \addmetakey{example}{name}
58 \newenvironment{cexample}[1] [] {\metasetkeys{example}{#1}
59 {\ifx\example@name\@empty\else\noindent\bfseries{\example@name}\fi}}
60 {}
```

cexercise The cexercise, cproblem and csolution environments are very simple to set up for L^AT_EX. For the L^AT_EXML side, we simplify matters considerably for the moment by restricting the possibilities we have on the CNXML side: We assume that the content is just one <cnx:para> element for the <cnx:problem> and <cnx:solution> elements.¹⁰

```
61 \newcounter{cexercise}
62 \srefaddidkey{cexercise}
63 \addmetakey{cexercise}{name}
64 \newenvironment{cexercise}[1] [] {\metasetkeys{cexercise}{#1}
65 {\ifx\cexercise@name\@empty\else\stepcounter{cexercise}\noindent\bfseries{\cexercise@name~\arab
66 {}
67 \srefaddidkey{cproblem}
68 \newenvironment{cproblem}[1] [] {\metasetkeys{cproblem}{#1}}{\}}
69 \srefaddidkey{csolution}
70 \newenvironment{csolution}[1] [] {\metasetkeys{csolution}{#1}}{\par\noindent\bfseries{Solution}}{\}}
```

crule

```
71 \srefaddidkey{rule}
72 \addmetakey{rule}{name}
```

⁹EDNOTE: check LaTeX.ltxml frequently and try to keep in sync, it would be good, if the code in LaTeXXML.ltxml could be modularized, so that the cnx/ltx namespace differences could be relegated to config options

¹⁰EDNOTE: relax this when we have automated the generation of cnx:para elements

```

73 \addmetakey{rule}{type}
74 \newenvironment{crule}[1] [] {\metasetkeys{rule}{#1}%
75 {\noindent\bfseries{\rule@type:}\ifx\rule@name\@empty\else~(\rule@name)\fi}}%
76 {}

```

statement

```

77 \srefaddidkey{statement}
78 \newenvironment{statement}[1] [] {\metasetkeys{statement}{#1}}{}

```

proof

```

79 \srefaddidkey{proof}
80 \newenvironment{proof}[1] [] {\metasetkeys{proof}{#1}}{}

```

definition

```

81 \srefaddidkey{definition}
82 \addmetakey{definition}{term}
83 \addmetakey{definition}{seealso}
84 \newenvironment{definition}[1] [] {\metasetkeys{definition}{#1}{\noindent\bfseries{Definition:}}}

```

cmeaning

```

85 \srefaddidkey{meaning}
86 \newenvironment{cmeaning}[1] [] {\metasetkeys{meaning}{#1}}{}

```

4.6 Conexxions

cnxn

```

87 \addmetakey{cnxn}{document}
88 \addmetakey{cnxn}{target}
89 \addmetakey{cnxn}{strength}
90 \newcommand\cnxn[2] [] % keys, link text
91 {\metasetkeys{cnxn}{#1}{\underline{#2}}\footnote{{\ttfamily\@ifx\cnxn@document\@empty\cnxn@docu
92 \newcommand\@makefnmark[1]{\parindent 1em\noindent\hb@xt@1.8em{\hss\@makefnmark}#1}

```

link

```

93 \addmetakey{link}{src}
94 \newcommand\link[2] [] {\metasetkeys{link}{#1}\underline{#2}}

```

cfigure The **cfigure** only gives us one of the possible instances of the `<figure>` element^{11,12}. In \LaTeX , we just pipe the size information through to `includegraphics`, in \LaTeXML , we construct the CNXML structure¹³

```

95 \srefaddidkey{cfigure}
96 \addmetakey{cfigure}{type}
97 \addmetakey{cfigure}{caption}
98 \newcounter{figure}
99 \newcommand\cfigure[3] [] {% cnx_keys, graphicx_keys, path

```

¹¹EDNOTE: extend that

¹²EDNOTE: do more about required and optional keys in arguments.

¹³EDNOTE: what do we do with the graphicx information about size,... CSS?

```

100 \begin{center}%
101 \includegraphics[#2]{#3}%
102 \metasetkeys{cfigure}{#1}\sref@target%
103 \ifx\cfigure@caption\@empty\else
104 \par\noindent Figure\refstepcounter{figure} {\arabic{figure}}: \cfigure@caption%
105 \protected@edef\@currentlabel{\arabic{figure}}%
106 \sref@label{id{Figure \thefigure}}\fi
107 \end{center}}

```

ccite

```

108 \addmetakey{ccite}{src}
109 \newcommand\ccite[2] [] {\metasetkeys{ccite}{#1}\emph{#2}}

```

term

```

110 \newcommand\term[1]{{\bfseries\underline{#1}}}

```

4.7 Metadata

metadata

```

111 \addmetakey{metadata}{version}
112 \addmetakey{metadata}{created}
113 \addmetakey{metadata}{revised}
114 \newsavebox{\metadatabox}
115 \newenvironment{metadata}[1] []%
116 {\noindent\hfill\begin{lrbox}{\metadatabox}
117 \begin{minipage}{.8\textwidth}%
118 {\Large\bfseries CNX Module: \cnx@name\hfill\strut}\[2ex]}%
119 {\end{minipage}\end{lrbox}\fbox{\usebox\metadatabox}\hfill}
120 % \newenvironment{metadata}[1] []%
121 % {\noindent\strut\hfill\begin{lrbox}{\metadatabox}\begin{minipage}{10cm}%
122 % {\strut\hfill\Large\bfseries CNX Module: \cnx@name\hfill\strut}\[2ex]}%
123 % {\end{minipage}\end{lrbox}\fbox{\usebox\metadatabox}\hfill\strut}\[3ex]}

```

authorlist

```

124 \newenvironment{authorlist}{\bfseries{Authors}:~}\[1ex]}

```

maintainerlist

```

125 \newenvironment{maintainerlist}{\bfseries{Maintainers}:~}\[1ex]}

```

cnxauthor

```

126 \srefaddidkey{auth}
127 \addmetakey{auth}{honorific}
128 \addmetakey{auth}{firstname}
129 \addmetakey{auth}{other}
130 \addmetakey{auth}{surname}
131 \addmetakey{auth}{lineage}
132 \addmetakey{auth}{email}
133 \newcommand\cnxauthor[1] [] {\metasetkeys{auth}{#1}\auth@first~\auth@sur,}

```

```

maintainer
134 \newcommand\maintainer[1] [] {\metasetkeys{auth}{#1}\auth@first~\auth@sur,}

keywordlist
135 \newenvironment{keywordlist}{\bfseries{Keywords}:~}{\[/[1ex]}

keyword
136 \newcommand\keyword[1]{#1,}

cnxabstract
137 \newenvironment{cnxabstract}%
138 {\par\noindent\strut\hfill\begin{minipage}{10cm}{\bfseries{Abstract}:~}}%
139 {\end{minipage}\hfill}
140 \</cls>

```

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.

authorlist, maintain- erlist=authorlist, maintainerlist (environment), 6	statement, proof (en- vironment), 4	environments:;cpara, cnote=cpara, cnote, 4 environments:;crule, statement, proof=crule, statement, proof, 4 environments:;definition, cmean- ing=definition, cmeaning, 5 environments:;keywordlist, key- word=keywordlist, keyword, 6 environments:;metadata=metadata, 6 environments:;cequation=cequation, keywordlist, key- word=keywordlist, keyword (en- vironment), 6 link= \subitem **\link+, \usage{6}
c*section=c*section (environment), 3	definition, cmean- ing=definition, cmeaning (en- vironment), 5	
ccontent=ccontent (environment), 3		
cequation=cequation (environment), 4	environments:;authorlist, maintain- erlist=authorlist, maintainerlist, 6	
cexample=cexample (environment), 4		
cexercise, cprob- lem, csolu- tion=cexercise, cproblem, csolution (en- vironment), 7	environments:;c*section=c*section, 3 environments:;ccontent=ccontent, 3 environments:;cequation=cequation, 4 environments:;cexample=cexample, 4 environments:;cnxauthor,maintainer=, \subitem **\cnxauthor,maintainer+, \usage{6}	
cfigure= \subitem **\cfigure+, 4\usage{7}		
cnxabstract=cnxabstract (environment), 7		
cnxauthor,maintainer= \subitem **\cnxauthor,maintainer+, \usage{6}		
cnxmodule=cnxmodule (environment), 3	cproblem, csolu- tion=cexercise, 7	
cnxn= \subitem **\cnxn+, \usage{6}		
cpara, cnote=cpara, cnote (en- vironment), 4	csolution, 7 environments:;cnxabstract=cnxabstract, 7 environments:;cnxmodule=cnxmodule, 3	metadata=metadata (environment), 6 showmeta= \subitem **\showmeta+, \usage{3}
crule, statement, proof=crule, 3		term= \subitem **\term+, \usage{6}

References

- [Koh15a] Michael Kohlhase. *metakeys.sty: A generic framework for extensible Metadata in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/metakeys/metakeys.pdf>.
- [Koh15b] Michael Kohlhase. *sref.sty: Semantic Crossreferencing in L^AT_EX*. Tech. rep. Comprehensive T_EX Archive Network (CTAN), 2015. URL: <http://www.ctan.org/tex-archive/macros/latex/contrib/stex/sref/sref.pdf>.
- [sTeX] *KWARC/sTeX*. URL: <https://svn.kwarc.info/repos/stex> (visited on 05/15/2015).