

The **STEX3** Package Collection*

Michael Kohlhase, Dennis Müller
FAU Erlangen-Nürnberg
<http://kwarc.info/>

2024-11-20

Contents

1	Introduction & Setup	7
1.1	What is STEX ?	7
1.2	The STEX package	8
1.3	What is MMT ?	8
1.4	Math archives and the MathHub Directory	8
1.5	Setting Up the STEX IDE	9
I	Tutorial	11
2	The Basics	12
2.1	Text symbols	15
2.1.1	Using Modules & Search in the IDE	15
2.2	Symbol References	18
2.3	Modules and Simple Symbol Declarations	21
2.4	Documenting Symbols	24
2.5	Sectioning and Reusing Document Fragments	26
2.6	Building and Exporting HTML	28
3	Mathematical Concepts	31
3.1	Simple Symbol Declarations	31
3.1.1	Semantic Macros and Notations	31
3.1.2	Types and Variables	34
3.1.3	Flexary Macros and Argument Modes	39
3.1.4	Precedences	41
3.1.5	Implicit Arguments	41
3.1.6	Finishing Equality	43
3.1.7	Variable Sequences	43
3.2	Statements	44
3.2.1	Definitions	44
	Semantic Macros in Text Mode	46

*Version 4.0.0 (last revised 2024-11-20)

Definientia	47
Using <code>Symbols</code> Without <code>Semantic Macros</code> and Exporting Code in Modules	48
3.2.2 Assertions	49
3.2.3 Proofs	52
3.3 Mathematical Structures	52
3.3.1 Declaring and Using <code>Structures</code>	52
Instantiating <code>Structures</code>	54
3.3.2 Extending <code>Structures</code> and Axioms	55
Conservative Extensions	56
3.3.3 Nesting <code>Structures</code> and <code>\this</code>	57
3.4 Complex Inheritance and Theory Morphisms	59
3.4.1 Glueing <code>Structures</code> Together	61
3.4.2 Realizations	63
4 Extensions for Education	65
4.1 Slides and Course Notes	65
4.2 Problems and Exercises	65
4.2.1 Background	65
4.2.2 The Package, Options, and Configuration	66
4.2.3 (Open) Problems	67
4.2.4 Solutions and Answer Classes	68
4.2.5 Grading Support	69
4.2.6 Structured Problems	70
4.2.7 Single/Multiple Choice Blocks	71
4.2.8 Filling-In Concrete Solutions	74
4.2.9 Including Problems	75
4.2.10 Testing and Spacing	76
4.3 Homework Assignments and Exams	76
4.3.1 Introduction	76
4.3.2 Package Options	76
4.3.3 Assignments	77
4.3.4 Including Assignments	77
4.3.5 Typesetting Exams	77
II User Manual	79
5 Basics	80
5.1 Package and Class Options	80
5.2 <code>Math Archives</code> and the <code>MathHub</code> Directory	81
5.2.1 The Structure of <code>Math Archives</code>	82
5.2.2 <code>MANIFEST.MF</code> -Files	82
5.3 The <code>lib</code> -Directory	83
5.4 Basic Macros	84
6 Document Features	85
6.1 Document Fragments	85
6.2 Using and Referencing Document Fragments	86
6.3 Cross-Document References	86

7 Modules and Symbols	89
7.1 Modules	89
7.1.1 Signature Modules , Languages, and Multilinguality	90
7.2 Symbol Declarations	90
7.2.1 Returns	91
7.3 Referencing Symbols	91
7.4 Notations and Semantic Macros	93
7.4.1 Precedences and Bracketing	94
7.4.2 Notations for Argument Sequences	95
7.4.3 Semantic Macros	95
7.5 Simple Inheritance	96
7.6 Variables and Sequences	98
7.7 Structures	99
7.7.1 Semantic Macros for Structures	99
8 Statements	102
8.1 More on Definitions	103
8.2 More on Assertions	104
9 Customizing Typesetting	105
9.1 Highlighting Symbol References	105
9.2 Styling Environments and Macros	106
9.3 Custom CSS for Environments	107
10 Additional Packages	108
10.1 NotesSlides Manual	108
10.1.1 Introduction	108
10.1.2 Package Options	108
10.1.3 Notes and Slides	109
10.1.4 Customizing Header and Footer Lines	110
10.1.5 Frame Images	110
10.1.6 Ending Documents Prematurely	111
10.1.7 Global Document Variables	111
10.1.8 Excursions	112
10.2 Problem Manual	113
10.2.1 Introduction	113
10.2.2 Problems and Solutions	113
10.2.3 Markup for Added-Value Services	115
Multiple Choice Blocks	115
Filling-In Concrete Solutions	116
10.2.4 Including Problems	117
10.2.5 Testing and Spacing	118
10.3 HWExam Manual	118
10.3.1 Introduction	118
10.3.2 Package Options	118
10.3.3 Assignments	119
10.3.4 Including Assignments	119
10.3.5 Typesetting Exams	119
10.4 Tikzinput Manual	120

III Documentation	122
11 S<small>TeX</small> Developer Manual	123
11.1 Documents	124
11.2 Modules	124
11.3 Symbols	126
11.4 Notations	128
11.5 Structural Features	131
11.6 Imports and Morphisms	133
11.7 Expressions and Semantic Macros	134
11.8 Optional (Key-Value) Argument Handling	135
11.9 Stylistable Commands and Environments	136
11.10 Math Archives	137
11.11 SMS-Mode	138
11.11.1 Second Pass	138
11.11.2 First Pass	139
11.12 Strings, File Paths, URIs	139
11.12.1 File Paths	140
File Path Constants and Variables	141
11.12.2 URIs	141
URI Constants and Variables	142
11.13 Language Handling	142
11.14 Inserting Annotations	143
11.14.1 Backend macros	143
11.15 Persisting Content from Math Archives in sms-Files	144
11.16 Utility Methods	144
11.16.1 Group-like Behaviours	145
12 Additional Packages	147
12.1 NotesSlides Documentation	147
12.2 Problem Documentation	147
12.3 HWExam Documentation	147
12.4 Tikzinput Documentation	147
IV Implementation	148
13 The S<small>TeX</small> Implementation	149
13.1 Setting up	149
13.2 Utilities	150
13.2.1 Calling kpsewhich and Environment Variables	150
13.2.2 Logging	151
13.2.3 File Paths	151
13.2.4 Languages	152
13.2.5 Group-like Behaviours	154
13.2.6 HTML Annotations	156
13.2.7 Auxiliary Methods	157
13.2.8 Persistence	163
13.2.9 Files, Paths and URIs	164
13.2.10 File Hooks	171

13.3	Math Archives	173
13.4	Documents	177
13.4.1	Title	177
13.4.2	Sectioning	178
13.4.3	References	182
13.4.4	Inputs	190
13.5	SMS Mode	195
13.6	Modules	199
13.6.1	The smodule-environment	199
13.6.2	Structural Features	210
13.7	Inheritance	215
13.7.1	\importmodule/\usemodule	215
13.7.2	Theory Morphisms	221
13.8	Symbols	227
13.8.1	Declarations	227
13.8.2	Notations	236
	a/B-mode argument handling	248
13.8.3	Variables	249
13.8.4	Sequences	253
13.8.5	Expressions	257
	Invoking Semantic Macros	258
	Argument Handling and Annotating	265
	Term HTML Annotations	269
	Automated Bracketing	272
	Symname and Variants	273
	Highlighting	275
13.9	Mathematical Structures	276
13.10	Statements	290
13.11	Proofs	296
13.12	Metatheory	304
13.13	MMT Interfaces	306
14	Additional Packages	310
14.1	Implementation: The notesslides Package	310
14.1.1	Class and Package Options	310
14.1.2	Notes and Slides	314
14.1.3	Environment and Macro Patches	318
14.1.4	Styling Across Notes/Slides	319
14.1.5	Beamer Compatibility	319
14.1.6	TODO Excursions	321
14.2	Implementation: The problem Package	322
14.2.1	Package Options	322
14.2.2	Problems and Solutions	323
14.3	Implementation: The hwexam Package	340
14.3.1	Package Options	340
14.3.2	QR Codes	341
14.3.3	Assignments	346
14.3.4	Leftovers	349
14.4	Tikzinput Implementation	349



sTeX is – by now – relatively stable and ready to use for the general public. However, it is also actively being developed further and subject to ongoing research. Some of the features described in here might not fully work as expected, some are still experimental, there might occasionally be cryptic error messages, and in general bugs are expected.

We welcome all kinds of issues you might encounter at <https://github.com/slatex/sTeX>.

*If you have questions or problems with **sTeX**, you can talk to us directly at
<https://matrix.to/#/#stex:fau.de>.*

Chapter 1

Introduction & Setup

1.1 What is $\text{\textcolor{teal}{STEX}}$?

$\text{\textcolor{teal}{STEX}}$ is a system for generating human-oriented documents in either $\text{\textcolor{teal}{PDF}}$ or $\text{\textcolor{teal}{HTML}}$ format, augmented with computer-actionable semantic information (conceptually) based on the $\text{\textcolor{teal}{OMDoc}}$ format and ontology.

At its core is the $\text{\textcolor{teal}{STEX}}$ package for $\text{\textcolor{teal}{LATEX}}$, that allows for semantically marking up document fragments; in particular concepts, formulae and mathematical statements (such as definitions, theorems and proofs). Running $\text{\textcolor{teal}{pdflatex}}$ over $\text{\textcolor{teal}{STEX}}$ -annotated documents formats them into normal-looking $\text{\textcolor{teal}{PDF}}$.

But $\text{\textcolor{teal}{STEX}}$ also comes with a conversion pipeline into semantically annotated $\text{\textcolor{teal}{HTML}}$, which can host semantic added-value services that make the documents *active* (i.e. interactive and user-adaptive) – essentially turning $\text{\textcolor{teal}{LATEX}}$ into a document format for (mathematical) knowledge management (MKM).

The $\text{\textcolor{teal}{STEX}}$ system consists of the following components:

- The $\text{\textcolor{teal}{STEX}}$ package,
- the $\text{\textcolor{teal}{RusTEX}}$ system for converting $\text{\textcolor{teal}{LATEX}}$ documents to (semantically enriched) $\text{\textcolor{teal}{HTML}}$,
- the $\text{\textcolor{teal}{MMT}}$ system for advanced knowledge management service and semantically informed backend for hosting the $\text{\textcolor{teal}{HTML}}$ with integrated added-value services,
- a collection of $\text{\textcolor{teal}{math archives}}$ of $\text{\textcolor{teal}{STEX}}$ document fragments and $\text{\textcolor{teal}{symbols}}$ for reuse, available of $\text{\textcolor{teal}{mathhub.info}}$, and
- the $\text{\textcolor{teal}{STEX IDE}}$: A $\text{\textcolor{teal}{VS Code}}$ extension that, besides the usual $\text{\textcolor{teal}{IDE}}$ functionalities like syntax highlighting, integrates $\text{\textcolor{teal}{RusTEX}}$ and $\text{\textcolor{teal}{MMT}}$ and allows for accessing the public $\text{\textcolor{teal}{math archives}}$ on $\text{\textcolor{teal}{mathhub.info}}$ to make the entire toolchain easily accessible to authors.

If $\text{\textcolor{teal}{PDF}}$ is all you are interested in, the $\text{\textcolor{teal}{STEX}}$ package is all you *need*. Either way, however, we recommend using the $\text{\textcolor{teal}{STEX IDE}}$ – it very much helps with semantically annotating your $\text{\textcolor{teal}{LATEX}}$ documents.

1.2 The **STEX** package

The **STEX** package extends **LATEX** with:

- A mechanism to declare **symbols** – concepts, functions, relations, variables, etc., which can be used and referenced in text or via **semantic macros** in mathematical formulae,
- a **module system** based on logical identifiers – **modules** bundle declarations, definitions, theorems, document snippets, and **symbols** for reuse, and
- an analogous organizational structure for developing documents modularly from individual fragments and sections.

The **STEX** package has been designed to have minimal impact on other **packages** and **document classes**, or the actual document layout – formatting of semantic **environments**, **symbol** references and **semantic macros** can be fully customized.

1.3 What is **Mmt**?

MMT is a **software system** and **Scala** API for generic knowledge management services. It is based on a version of the **OMDOC** ontology and **document format (MMT/OMDoc)**.

Among the services **MMT** provides are compiling, building, converting and managing libraries, a built-in web-server for browsing content, various algorithms for generic computation, checking and translating expressions, and querying.

1.4 Math archives and the MathHub Directory

To make the most of **STEX**, it is strongly encouraged to follow a workflow of small document fragments and **modules** to maximize reuse.

One considerable weakness of **LATEX** is the way source files are referenced: they need to be either in a **texmf** directory, or else be referenced via file paths relative to the main **.tex**-file being compiled. This is highly inconvenient if we want to collaboratively develop many highly interrelated document fragments.

STEX therefore adds an organizational layer on top of **LATEX**'s: **math archives** stored in a fixed **MathHub** directory anywhere on your hard drive. Referencing source files and **modules** is then done relative to the containing **math archive**, and is thus *independent* of user's individual setups or the current **.tex**-file.

The drawback of this approach is that **STEX** needs to know the location of your **MathHub** directory. There are multiple ways to achieve that, but the simplest and recommended approach is to set an environment variable: Simply create a new directory **<path>/MathHub** somewhere on your hard drive and set the environment variable **MATHHUB** as the path to this new directory.

Alternatively, you can let the **STEX IDE** do the work for you (see ??).

For more on **math archives**, see ??.

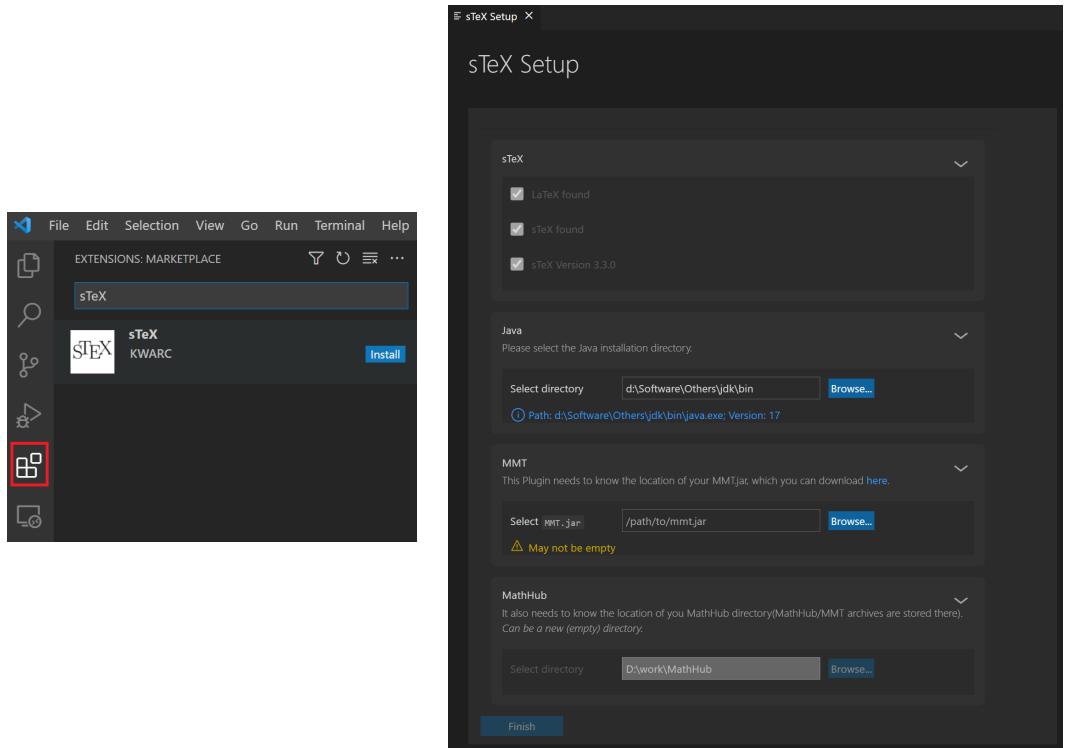


Figure 1: Installing the `sTeX` IDE

1.5 Setting Up the `sTeX` IDE

`sTeX` is based on `LATEX`, and adds additional layers of presentational and functional markup to it. As a consequence the source files of `sTeX` documents look quite different from the resulting `XHTML` and `PDF` documents. Thus the best way of interacting with the `sTeX` document collections is via an integrated development environment (IDE). In this tutorial we will use the `sTeX` plugin for the `VS Code`, which you should set up as a first step (this also sets up the necessary auxiliary software).

Setting up `sTeX` with the dedicated IDE is easy:

1. Download and install `VS Code` here: <https://code.visualstudio.com/download>
2. Start `VS Code` and navigate to the *Extensions*-tab on the left. Here you can search for Extensions in the `VS Code` marketplace. Look for the `sTeX` extension by `KWARC`, as in Figure 1 on the left.
3. Having done so, upon opening any folder in `VS Code` containing a `.tex`-file the setup window will pop up, as in Figure 1 on the right.

The IDE will attempt to determine your `Java` installation and your `MathHub` directory (if set via an environment variable). Alternatively, you can set the latter now.

4. Download the `MMT` `.jar`-file at the link provided in the setup and select it. The IDE should then be able to determine your `MMT` version.

And that's it. Click on *Finish* and your setup is finished. The extension will start and download **RuSTEX** and some fundamental **math archives** for you automatically (an internet connection is required when finishing the setup).

Part I

Tutorial

The dynamic [HTML](#) version of this part can be found at

*<https://stexmmt.mathhub.info/>:
[sTeX/fullhtml?archive=sTeX/Documentation&filepath=tutorial.en.xhtml](https://stexmmt.mathhub.info/sTeX/fullhtml?archive=sTeX/Documentation&filepath=tutorial.en.xhtml)*

[sTeX](#) is a system for generating human-oriented documents in either [PDF](#) or [HTML](#) format, augmented with computer-actionable semantic information (conceptually) based on the [OMDoc](#) format and ontology.

In this part, we will give a broad but shallow introduction to [sTeX](#), and what you can get out of it. Additionally, this serves as an introduction to the [sTeX IDE](#), and we consequently assume that you have that one set up, as described in ??.

Note that in [PDFs](#), the specific highlighting of semantically annotated text is fully customizable (see ??). In this document, we use [this highlighting](#) for [notation](#) components, [this highlighting](#) for [symbol](#) references, this highlighting for (local) [variables](#) and [this highlighting](#) for definienda; i.e. new concepts being introduced.

Chapter 2

The Basics

This document itself uses **sTeX** and serves as a direct example for the following. You can download its source files, the generated **PDF** files, and the generated **HTML** documents directly from within the **IDE**, by navigating to the **sTeX** tab in the menu on the left and finding **sTeX/Documentation** in the list of **math archives** and clicking the small “Install”-button next to it, see the screenshot on the left of Figure 2.

Once downloading is finished (this may take a while since dependencies are also downloaded), you can then browse the **.tex**-files in **sTeX/Documentation** directly from the **math archives** panel in the **sTeX** tab, as you can see in the right screenshot in Figure 2.

For example, you can now navigate to the file **tutorial/intro.en** to see the sources of this very chapter.

As a first example, consider the following document fragment from ??:

sTeX is a system for generating human-oriented documents in either **PDF** or **HTML** format, augmented with computer-actionable semantic information (conceptually) based on the **OMDoc** format and ontology.

If you were to look at the generated **HTML** from this fragment, you could hover over the highlighted words (**sTeX**, **PDF**, **HTML**, **OMDoc**) and get a little popup with their definitions (Figure 3). Neat, huh?

Here, in the **PDF**, hovering will only show you a unique identifier (**MMT-URI**) for the word, and link to a definition on the web. Still useful, but not quite as neat, of course.

A plain **LATeX**-version of the above document fragment, without any **sTeX** markup, could look like this:

Example 1

Input:

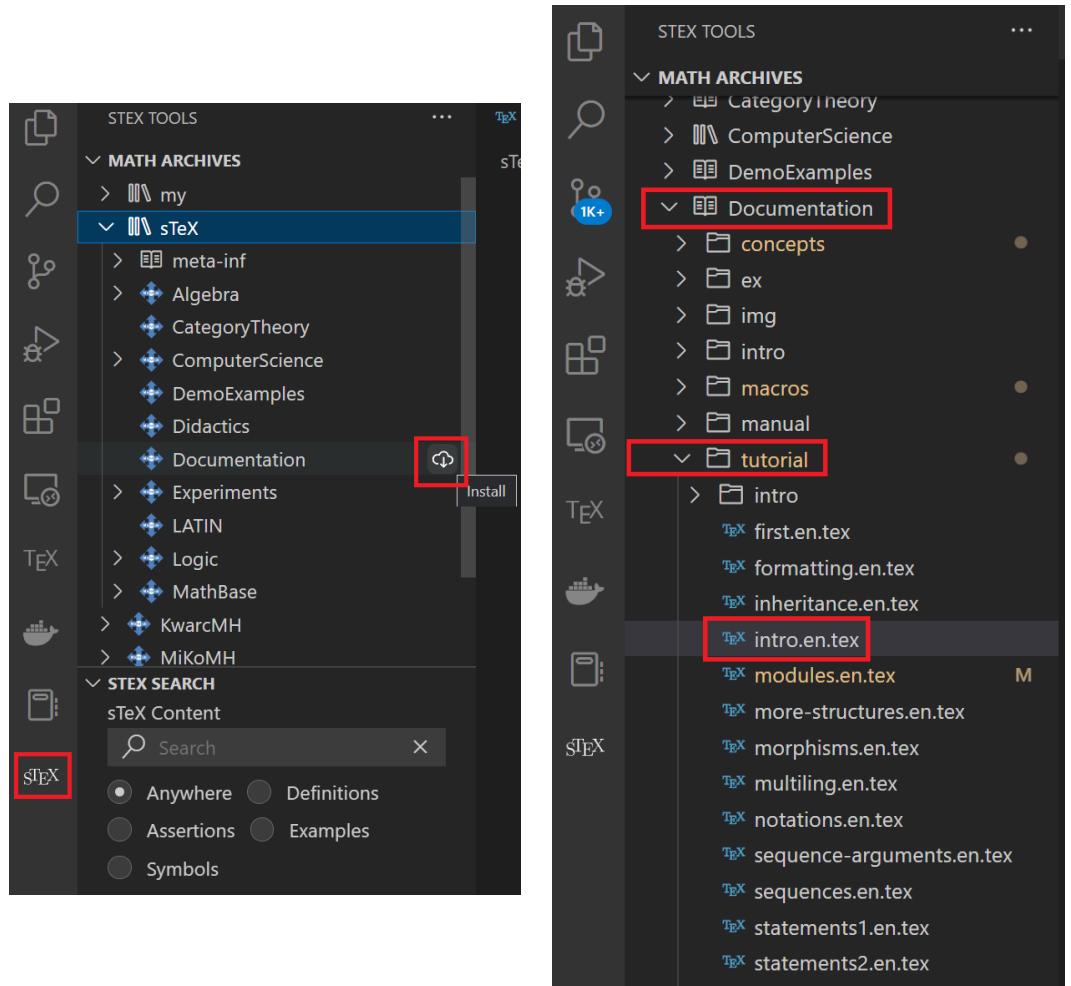


Figure 2: Installing Math Archives

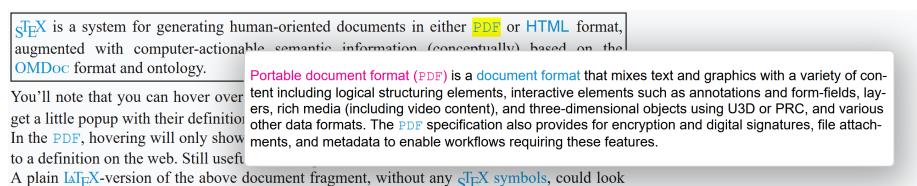


Figure 3: Definition on Hover

```

File [sTeX/Documentation]tutorial/intro/intro1plain.en.tex

1 \documentclass{article}
2 \usepackage{sTeX-logo}
3 \begin{document}
4
5   \sTeX{} is a system for generating human-oriented documents
6   in either \textsf{PDF} or \textsf{HTML} format, augmented
7   with computer-actionable semantic information (conceptually)
8   based on the \textsc{OMDoc} format and ontology.
9
10 \end{document}

```

Output:

sTeX is a system for generating human-oriented documents in either PDF or HTML format, augmented with computer-actionable semantic information (conceptually) based on the OMDoc format and ontology.

(Examples like the one above always show the file the source code is in, so if you have downloaded the sTeX/Documentation [math archive](#) you can toy around with it yourself)

If you save a file in the [IDE](#) (regardless of whether it has unsaved changes), a preview window will pop up, showing you the [HTML](#) generated from the .tex-file; see (Figure 4).

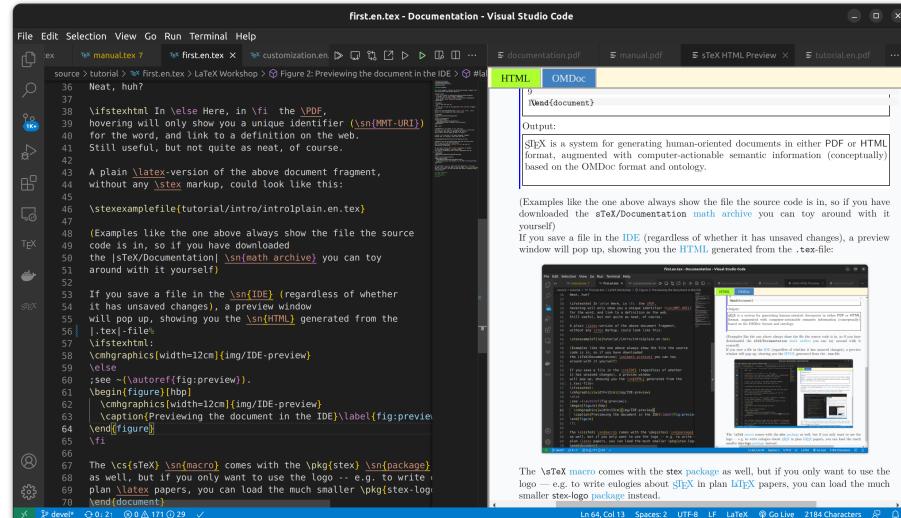


Figure 4: Previewing the document in the IDE

The \sTeX macro comes with the [stex package](#) as well, but if you only want to use the logo – e.g. to write eulogies about [sTeX](#) in plain [LaTeX](#) papers, you can load the much smaller [stex-logo package](#) instead.

2.1 Text symbols

The most central concept behind **s_Te_X** is that of a *symbol*:

A **symbol** is a *named* concept that can be defined, documented and referenced. Examples for **symbols** are mathematical constants, functions, theorems, statements, principles – anything that has a (somewhat) precise meaning and can be referenced by name can be a **symbol**.

Before we explain how we can declare new **symbols** and associate them with definitions, **notations** and all that, let's assume an ideal world in which others have done that job already for us – after all, **s_Te_X** is all about *reuse*, and naturally, there are **s_Te_X symbols** for all of the above already. Let's start with the one for **s_Te_X** itself:

2.1.1 Using Modules & Search in the IDE

In the **VS Code IDE**, navigate to the **s_Te_X-tab** on the left. In the search panel, select the “**Symbols**” radio button and search for “**s_Te_X**”. The second search result should be what we're looking for (Figure 5).

Search results are grouped into *local* and *remote* results. Local ones are the ones you already have in your local **MathHub** directory; remote ones you can download directly from within the **IDE**.

You can click the preview button to see the generated **HTML** for the document – the resulting window that pops up also has an **OMDoc** tab you can select, which (among other things) shows you the **semantic macros** provided by the respective **module**: In this case, it tells us that there is a **text symbol** named “**s_Te_X**” with **semantic macro** `\stex` in the **module** `mod/systems/tex?sTeX` that is in the `\sTeX/ComputerScience/Software` archive. It produces the presentation “**s_Te_X**” as we want (Figure 6).

A **text symbol** is a **symbol** `foo` with an associated **semantic macro** `\foo`. The **macro** `\foo` is allowed in text or math mode and produces a predefined piece of text output annotated with `foo`.

The variant `\fooname` produces the same output without annotation.

If we want to use the **s_Te_X symbol** in a document – which we have open in the **IDE** – we simply click on the **use** button, and the **IDE** will automatically insert the line `\usemodule[sTeX/ComputerScience/Software]{mod/systems/tex?sTeX}`, making all **symbols** in that **module** available to use – in particular, we can now use the `\stex` **semantic macro** instead of the plain, non-semantic `\sTeX macro` – that is, of course, after we include the **stex package** first.

s_Te_X The `\usemodule` macro takes as *optional* argument the name of a **math archive**, and as a regular argument the path to an **s_Te_X module** (see ??).

Analogously, we can also search for the **PDF**, **HTML** and **OMDoc** symbols, all of which are also **text symbols** and have the associated **semantic macros** `\PDF`, `\HTML` and `\omdoc`; the document should thus look like this:

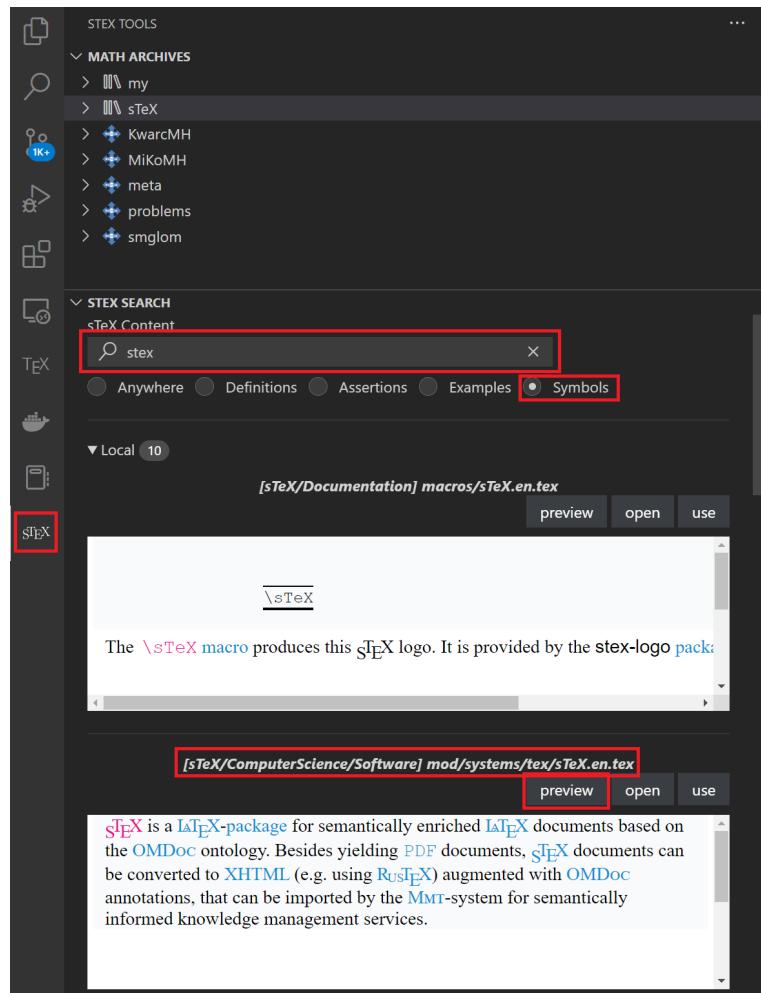


Figure 5: Search in the STEX IDE



Figure 6: OMDOC Preview

Example 2

Input:

```
File [sTeX/Documentation]tutorial/intro/introstex.en.tex
1 \documentclass{article}
2 \usepackage{sTeX}
3 \begin{document}
4   \usemodule[sTeX/ComputerScience/Software]{mod/systems/tex?sTeX}
5   \usemodule[sTeX/ComputerScience/Software]{mod/formats?PDF}
6   \usemodule[sTeX/ComputerScience/Software]{mod/formats?HTML}
7   \usemodule[sTeX/ComputerScience/Software]{mod/formats?OMDoc}
8
9   \stex is a system for generating human-oriented documents
10  in either \PDF or \HTML format, augmented
11  with computer-actionable semantic information (conceptually)
12  based on the \omdoc format and ontology.
13 \end{document}
```

Output:

STeX is a system for generating human-oriented documents in either **PDF** or **HTML** format, augmented with computer-actionable semantic information (conceptually) based on the **OMDOC** format and ontology.

Now, our generated **HTML** looks a lot more interesting, with highlighting, pop-ups on hover and all that. Notably however, if we compile the file with `pdflatex`, it looks pretty much exactly as before.

That's because we haven't told **STeX** what to do with semantic annotations yet – and by default, it does not do anything fancy, except for wrapping them in an `\emph`. We can customize how we want **STeX** to highlight various semantic text fragments (see ??). A default highlighting schema is provided in the **stex-highlighting package** – including that will

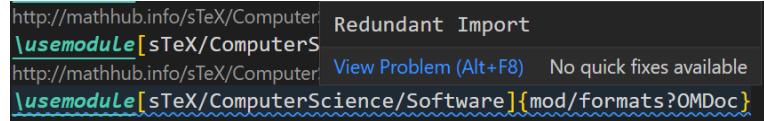


Figure 7: Redundant Imports

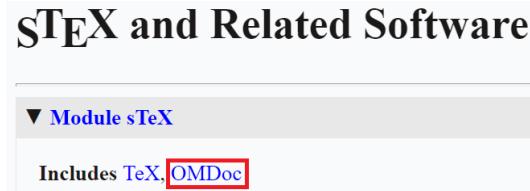


Figure 8: Includes in the OMDoc Preview

- highlight semantically annotated text in [this color](#),
- show the MMT-URI of the corresponding [symbol](#) in a tooltip on hovering over the text,
- make the text link to the place the [symbol](#) is being defined in the current document (if it is), or, alternatively,
- make it link to an external resource, if one is known. In our case, they link to [stexmm.info/:sTeX](#), where the [HTML](#) for all the [symbols](#) we use in this document are hosted.

Note that in the [IDE](#), the `\usemodule`-statement for [OMDoc](#) is underlined in blue (Figure 7) – [VS Code](#) is letting us know, that this `\usemodule` statement is *redundant*. That is because the [sTeX module](#) we imported earlier already imports the [OMDOC module](#); as such we have all [macros](#) therein available already. If we look at the [sTeX module](#) in the [VS Code](#) preview window again, we can see that (Figure 8).

We can consequently safely delete the `\usemodule` again.

2.2 Symbol References

Let's continue with the next paragraph of ??; for now unannotated:

Example 3

Input:

```

File [sTeX/Documentation]tutorial/intro/intro2plain.en.tex
1 \documentclass{article}
2 \usepackage{sTeX}
3 \begin{document}
4
5 At its core is the \sTeX{} package for \LaTeX{}, that allows for
6 semantically marking up document fragments; in particular
7 concepts, formulae and mathematical statements (such as
8 definitions, theorems and proofs). Running \texttt{pdflatex}
9 over \sTeX-annotated documents formats them into normal-looking
10 \textsf{PDF}.
11
12 \end{document}

```

Output:

At its core is the package for , that allows for semantically marking up document fragments; in particular concepts, formulae and mathematical statements (such as definitions, theorems and proofs). Running `pdflatex` over -annotated documents formats them into normal-looking `PDF`.

We already know how to annotate “” and “`PDF`”; and if we use the search field in the **IDE** again, we can also find a `text symbol` for “`\LaTeX`”. But if we look at the documentation, we will note that *more* is highlighted:

At its core is the package for `\LaTeX`, that allows for semantically marking up document fragments; in particular concepts, `formulae` and `mathematical` statements (such as definitions, theorems and proofs). Running `pdflatex` over -annotated documents formats them into normal-looking `PDF`.

The “`package`”-symbol can be found in the `\LaTeX` module too, and searching for the keywords “`formula`” and “`mathematics`” will yield the symbols “`well-formed formula`” and “`mathematics`”, but they are not `text symbols` and “`mathematics`” and “`package`” do not even have a `semantic macro` – and the one for “`well-formed formula`” would not work outside of math mode.

`Text symbols` are special in that way – they are intended for `symbols` that have a specific formatting associated (such as `\LaTeX`, `OMDOC`, or `HTML`, which we prefer to typeset as sans serif). For those settings, it makes sense to associate that formatting with a `semantic macro` that does the typesetting for us.

`Symbols without a text macro` can be referenced with the `\symname` macro: `\symname{package}` prints the *name* of the “`package`”-symbol and annotates it accordingly, without any special formatting – in particular it is compatible with being in `\emph`, `\textbf` and similar `macros`. That takes care of *one* of the missing annotations.

More generally, the `\symref` macro can be used to annotate arbitrary text with a `symbol`: `\symref{mathematics}{mathematical}` associates the text `mathematical` with the `symbol` “`mathematics`”; thus, we get “`mathematical`” and similarly “`formulae`”.

sTeX

In general, any `macro` that expects a `symbol` name can be given either

1. the *name* of the `symbol`,

2. the name of its [semantic macro](#),
3. or any suffix of its MMT-URI containing at least the [module](#) name.

The second option is often short – and therefore convenient to write; for example, to achieve “[formulae](#)”, we can also write `\symref{wff}{formulae}`, since `\wff` is the [semantic macro](#) for “[well-formed formula](#)”.

sTeX

The third option allows for distinguishing between multiple [symbols](#) with the same name – the [IDE](#) can help in the latter case, by underlining ambiguous [symbol](#) references in yellow, and offering the [Quick Fix](#) functionality to let you select and autocomplete the specific [symbol](#) you want to reference.

Since `\symname` and `\symref` are a lot to type for something that should ideally be used as often as possible, the [macros](#) `\sn` and `\sr` exist as well and behave exactly the same way. We also provide some convenience abbreviations for `\sn`; namely `\Sn` (capitalizes the first letter of the [symbol](#) name), `\sns` (adds an “`s`” at the end, for the most common pluralization of a name), and `\Sns` (both).

Using all of the above, our annotated fragment now looks like this:

Example 4

Input:

```
File [sTeX/Documentation]tutorial/intro/intro2stex.en.tex
5 \usemodule[sTeX/ComputerScience/Software]{mod/systems/tex?sTeX}
6 \usemodule[sTeX/Logic/General]{mod/syntax?Formula}
7 \usemodule[sTeX/MathBase/General]{mod?Mathematics}
8 \usemodule[sTeX/ComputerScience/Software]{mod/formats?PDF}
9
10 At its core is the \stex \sn{package} for \LaTeX, that allows for
11 semantically marking up document fragments; in particular
12 concepts, \sr{wff}{formulae} and \sr{mathematics}{mathematical}
13 statements (such as definitions, theorems and proofs). Running
14 \texttt{pdflatex} over \stex-annotated documents formats them
15 into normal-looking \PDF.
```

Output:

At its core is the [sTeX package](#) for [L^AT_EX](#), that allows for semantically marking up document fragments; in particular concepts, [formulae](#) and [mathematical](#) statements (such as definitions, theorems and proofs). Running `pdflatex` over [sTeX](#)-annotated documents formats them into normal-looking [PDF](#).

There’s only one problem: *the document does not compile*, with an error [Undefined control sequence](#). The reason being that *some macro* in the [module](#) `Formula` uses the `\text macro`. We can fix that by using the [amsfonts package](#) of course, but this points to a more general problem; namely that [modules](#) can make use of various [L^AT_EX](#) packages for typesetting [symbols](#).

Good practice suggests putting those packages into a *prelude* per [math archive](#), which we can then import from anywhere, using the `\libinput macro`. For more on that, see [??](#).

For now, suffice it to say that we can import all [packages](#) required for the [module](#) `Formula` from the [math archive](#) `sTeX/Logic/General` by adding the line

```
\libinput[sTeX/Logic/General]{preamble}
```

before the `\begin{document}`.

2.3 Modules and Simple Symbol Declarations

Consider again the first two paragraphs of ??:

sTeX is a system for generating human-oriented documents in either [PDF](#) or [HTML](#) format, augmented with computer-actionable semantic information (conceptually) based on the [OMDoc](#) format and ontology.

At its core is the [sTeX package](#) for [LaTeX](#), that allows for semantically marking up document fragments; in particular concepts, [formulae](#) and [mathematical](#) statements (such as definitions, theorems and proofs). Running `pdflatex` over [sTeX](#)-annotated documents formats them into normal-looking [PDF](#).

Firstly, note that the first paragraph would be perfectly suitable to serve as a pop-up definition on hover for the [sTeX symbol](#). Secondly, what if all the [symbols](#) used in the above *didn't* already exist?

In this section, we will describe how to make your own [symbols](#) and collect them as reusable fragments in [modules](#) and [math archives](#) from scratch.

We start by creating a new [math archive](#). In the [IDE](#), switch to the [sTeX](#)-tab on the left and click the “New sTeX Archive” button (Figure 9). You will then be asked for the

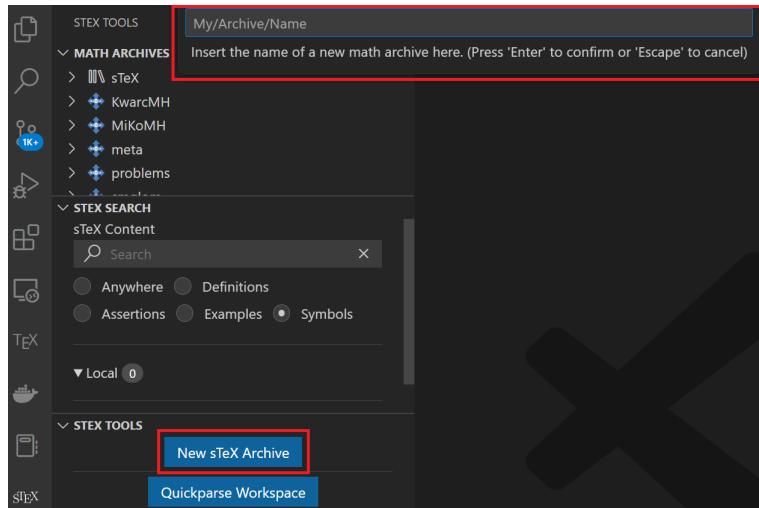


Figure 9: New Math Archive in the IDE

name of the [archive](#), a [namespace](#) for its content, and a [url-base](#), where the content is supposedly going to end up online. You can safely keep the defaults for the latter two. In the following, we assume that your archive is named `my/archive`.

The [IDE](#) will then create the following files and directories in your MathHub directory:

```

- my
  - archive
    - lib
      - preamble.tex
    - META-INF
      - MANIFEST.MF
    - source
      - helloworld.tex

```

...and open the file `helloworld.tex` with the content

```

1 \documentclass{sTeX}
2 \libinput{preamble}
3 \begin{document}
4 % A first sTeX document
5 \end{document}

```

You can now reference any newly created content in your new `archive` using for example `\usemodule[my/archive]{...}`.

Let's start with the “`\TEX`” symbol. Rename the file `helloworld.tex` to something more meaningful, for example `latex.en.tex` – the `.en` will be picked up on by `sTeX` to signify that the fragment will be in *english* (see ??).

What we want to achieve in this file is the following:

`\TEX` is a document typesetting software developed by Donald Knuth, with a focus on mathematical formulae. It is based on a powerful and extensible `macro` expansion engine.

`\LATEX` is a (nowadays) default collection of `\TEX` macros developed by Leslie Lamport. Among other things, `\LATEX` introduces `environments`, a distinction between preamble and document content, `packages` to bundle and distribute `macro` definitions, and `document classes`: special `packages` that govern the global layout of a document.

In particular, in the `HTML` the two paragraphs above should be shown when hovering over the `symbols` they define (as indicated by the magenta definiendum highlighting). So we need `symbols` and `semantic macros`, for: `\TEX`, `macro`, `\LATEX`, `environment`, `package` and `document class`.

`Symbol` declarations are only allowed within `modules`:

`\STeX` A `module` is a *named* block that bundles `symbol` declarations for subsequent reuse.
A `module` is introduced with the `smodule`-environment.

Let's name our `module` `LaTeX`. We then wrap the contents of our document in a `smodule` environment:

```

\begin{document}
\begin{smodule}{LaTeX}
...
\end{smodule}
\end{document}

```

Note, that the `IDE` immediately picks up on this and displays the full `MMT-URI` of our new `module` over the `\begin{smodule}{LaTeX}` (Figure 10) –

```
http://mathhub.info/my/archive?LaTeX
\begin{smodule}{LaTeX}
```

Figure 10: VS Code Code Lense

From this, we can glimpse that the `namespace` of the `module` is `http://mathhub.info/my/archive/latex`. This implies, that to use the `module` somewhere else, we will have to type `\usemodule[my/archive]{latex?LaTeX}` – the `latex`-part pointing to the `file` and `LaTeX` referring to the actual `module`.

If we rename the file to `LaTeX.en.tex`, we notice that the `namespace` changes to `http://mathhub.info/my/archive`, allowing us to now use it with `\usemodule[my/archive]{LaTeX}` directly. That's because the `module` name `LaTeX` and the file name `LaTeX` match now (see ??, Figure 11).

```
http://mathhub.info/my/archive?LaTeX
\begin{smodule}{LaTeX}
```

Figure 11: VS Code Code Lense



Note that “`LaTeX`” and “`latex`” only differ in capitalization – if your file system is case-insensitive (as e.g. MacOS’s was until quite recently), this distinction gets murky, but remains very important especially if you want to share your `math archive` with others!

It is therefore *highly recommended* to treat file names as case-sensitive either way.

Within the `module`, we can now declare new `symbols` using the `\symdecl`-macro. We start with those that are not `text symbols`:

```
\symdecl*[macro]
\symdecl*[environment]
\symdecl*[package]
\symdecl*[document class]
```

The `*` after the `\symdecl` indicates, that we do not want a `semantic macro` for the `symbol` – otherwise, it would generate one with the same name as the `symbol` itself and “pollute the `macro` space”, so to speak.

The `symbols` `TeX` and `LATEX`, however, have a definite way of being typeset associated with them, which can be produced using the standard `\TeX` and `\LaTeX` `macros`. So let's make them `text symbols`, using the `\textsymdecl` macro:

```
\textsymdecl{tex}{\TeX}
\textsymdecl{latex}{\LaTeX}
```

The first argument being the name of the generated `macro` (i.e. `\tex` and `\latex`) and the second one specifying the output to produce.

2.4 Documenting Symbols

We can now use the two new macros, `\symname/\sn`, `\symref/\sr` etc. to mark up the above two paragraphs. But the IDE also makes us aware of the symbols not yet being documented, via squiggly blue lines(Figure 12).

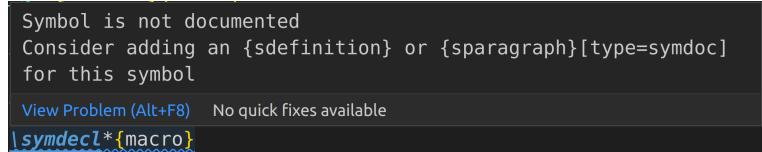


Figure 12: Undocumented Symbols

Among other things, this means that the system does not yet know what to show a reader when hovering over the symbol in the HTML. The IDE also recommends two ways to fix that: The `sdefinition` or `sparagraph` environments.

Ignoring the former for now, which is more useful for mathematical concepts, we can use the following to mark up the first paragraph:

```
\begin{sparagraph}[style=symdoc,for={tex,macro}]
  \tex is a document typesetting
  software developed by Donald Knuth, with a focus on
  mathematical formulae. It is based on a powerful
  and extensible \sn{macro} expansion engine.
\end{sparagraph}
```

In general, the `sparagraph` environment can be used to mark up arbitrary paragraphs semantically, but the `style=symdoc` option tells `STEX` to use this paragraph as a documentation for the symbols provided in the `for=` option. And indeed, doing so makes the squiggly blue lines in the IDE under `\textsymdecl{tex}{TeX}` and `\symdecl*{macro}` disappear.

We just used the semantic macro `\stex` and the `\sn` macro to mark up the fragment – but we can do better. Both concepts are being *introduced* in the above paragraph, and we can let `STEX` know that that is the case:

Within an `sparagraph` environment with `style=symdoc` (or an `sdefinition` environment), we can mark up *definienda*, meaning the terms *being defined*, explicitly. Analogously to `\symname` and `\symref`, we have the macros `\definame` and `\definiendum` for that purpose.

Note that the `\tex` macro induced by the `text` symbol above already marks up the “`TeX`” it produces, so wrapping it in another `\definiendum` would be redundant. However, every `text` symbol also generates a *second* macro with the suffix `name` that generates a non-marked-up version of the same presentation. In other words, we get the macro `\texname` for free, that produces “`TeX`” (of course, we could just as well use the `\TeX` macro, but that one you probably know already).

Furthermore, every `\definiendum` or `\definame` automatically adds the symbol being referenced to the internal `for=`-list of the `sparagraph` environment, obviating the need to list it explicitly.

As such, we can produce a better markup like this:

```
\begin{sparagraph}[style=symdoc]
  \definiendum{tex}{\texname} is a document typesetting
```

```

software developed by Donald Knuth, with a focus on
mathematical formulae. It is based on a powerful
and extensible \definename{macro} expansion engine.
\end{sparagraph}

```

Exercise

In your archive `my/archive`, create additional files that produce the following outputs:

Mathematics.en.tex

To do **mathematics** is to be, at once, touched by fire and bound by reason. This is no contradiction. Logic forms a narrow channel through which intuition flows with vastly augmented force.

– Jordan Ellenberg

PDF.en.tex

Portable Document Format (PDF) is a document format that mixes text and graphics with a variety of content.

HTML.en.tex

The **HyperText Markup Language (HTML)** is a representation format for web-pages.

OMDoc.en.tex

OMDoc is a document format for representing **mathematical** documents with their flexiformal semantics.

such that the following file compiles and shows the above snippets on hover:

sTeX.en.tex

```

1 \documentclass{sTeX}
2 \libinput{preamble}
3 \begin{document}
4 \begin{smodule}{sTeX}
5   \usemodule{OMDoc}
6   \usemodule{PDF}
7   \usemodule{HTML}
8   \textsymdecl{sTeX}{\sTeX}
9   \begin{sparagraph}[style=symdoc]
10     \definiendum{sTeX}{\stexname} is a system for generating
11     documents in either \PDF or \HTML format, augmented with
12     computer-actionable semantic information (conceptually)
13     based on the \OMDoc format and ontology.
14   \end{sparagraph}
15 \end{smodule}
16 \end{document}

```

sTeX is a system for generating documents in either **PDF** or **HTML** format, augmented with computer-actionable semantic information (conceptually) based on the **OMDOC** format and ontology.

The preamble of every file should only be

```
\documentclass{stex}
\libinput{preamble}
```

and the macros `\OMDoc`, `\PDF`, `\HTML` should produce `\textsc{\OMDoc}`, `\textsf{\PDF}` and `\textsf{\HTML}`, respectively (but with semantic annotations of course).

Lösung: Can be found in [sTeX/Documentation]source/tutorial/solution

2.5 Sectioning and Reusing Document Fragments

We know now how to import and reuse the `symbols` of some `module` (using `\usemodule`). What about the actual document `content`?

Assume we want to write a new article that includes all of the fragments in `my/archive` we made so far, in a file `all.en.tex` in the same `math archive`:

```
1 \documentclass{article}
2 \usepackage{stex}
3 \libinput{preamble}
4 \begin{document}
5   \author{Me}
6   \title{The \texttt{my/archive} Archive}
7   \maketitle
8   \tableofcontents
9 ...
10 \end{document}
```

In there, we want sections as follows:

```
- 1 Preliminaries
  (Mathematics)
  - 1.1 Document Formats
    (PDF)
    (HTML)
    (OMDoc)
- 2 \TeX and Friends
  (LaTeX)
  (sTeX)
```

We could of course do the following:

```
\section{Preliminaries}
\input{Mathematics.en}
\subsection{Document Formats}
\input{PDF.en}
\input{HTML.en}
\input{OMDoc.en}
\section{\TeX and Friends}
\input{LaTeX.en}
\input{sTeX.en}
```

...but this approach has two drawbacks:

Firstly, we need to manually keep track of the section levels, by explicitly writing `\section`, `\subsection` etc. This is fine as long as we are just interested in this particular

article. But what if we want to *reuse* the article's content in another document at some point? The section levels might be entirely different then – e.g. we might want the “Preliminaries” section to be a subsection instead.

Secondly, the `\input` macro considers the file name/path provided to be either *absolute* or relative to the *current tex file being compiled* – which means that the `\input{Mathematics.en}` only works for files in the same directory as `Mathematics.en.tex`.

In short: using `\section`, `\chapter` etc. explicitly, and `\input` to reuse fragments, breaks reusability.

Instead of using `\section` and `\subsection`, **sT_EX** therefore provides the **sfragment environment**.

`\begin{sfragment}{Foo}... \end{sfragment}` inserts a sectioning header depending on the current section level and availability. These are: `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph` and `\ subparagraph`. This allows us to do the following instead:

```
\begin{sfragment}{Preliminaries}
  \input{Mathematics.en}
  \begin{sfragment}{Document Formats}
    \input{PDF.en}
    \input{HTML.en}
    \input{OMDoc.en}
  \end{sfragment}
\end{sfragment}
\begin{sfragment}{\TeX and Friends}
  \input{LaTeX.en}
  \input{sTeX.en}
\end{sfragment}
```

The only problem remaining now is that if we do this, **sT_EX** will insert a `\part` for the first **sfragment**. If we want the “top-level” sectioning level to be `\section` instead, we can insert a `\setsectionlevel{section}` in the preamble.

As a more reuse-friendly replacement of `\input`, **sT_EX** provides the `\inputref` macro. Using that has two advantages: Firstly, its argument is relative to some (optionally provided, or the current) **math archive** and is thus independent of the specific location of the file relative to the currently being compiled .tex-file. Secondly, when converting to **HTML**, it will *not* “copy” the referenced file's content in its entirety (as `\input` would), but instead dynamically insert the already existent (if so) **HTML** of the referenced file, avoiding content duplication and having to process the file all over again.

In general `\inputref[some/archive]{file/path}` inputs the file `file/path.tex` in the **archive** `some/archive`. As the `\input`-ed files in the example above are in the same **archive** anyway, we can simply substitute the `\inputs` by `\inputrefs` and call it a day.

Finally, we can make two more minor changes:

1. The *title* of our document is only supposed to be there, if we compile the document directly – if we were to `\inputref` our file into a “driver file” `all.en.tex`, the title and the table of contents should be omitted.

We can achieve this using the `\ifinptref` conditional: by wrapping the header in an `\ifinptref \else... \fi`, it will only be processed if the file is *not* being loaded using `\inputref`. `\ifinptref` is a “classic” `TeX` conditional and is treated as such in both `PDF` and `HTML` compilation. A smarter `macro` to use is `\IfInputref`, which takes two arguments for the *true* and *false* cases, respectively. Additionally, when compiling to `HTML`, *both* arguments to `\IfInputref` will be processed, and the backend will decide which of the two to present when serving a document.

2. The table of contents should also be omitted in `HTML` mode. To achieve that, we can use the `\ifstexhtml` conditional, which is *true* if the document is being compiled to `HTML`, and *false* if compiled to `PDF`.



Note, that since *both* arguments of `\IfInputref` are processed, they should *not* open `TeX` groups or `environments`!

In summary, we can modify our document to do the following:

```
\IfInputref{}{
  \author{Me}
  \title{The \texttt{my/archive} Archive}
  \maketitle
  \ifstexhtml \else \tableofcontents \fi
}
```

The final `all.en.tex` can be found in `[sTeX/Documentation]tutorial/solution/all.en.tex`.

2.6 Building and Exporting `HTML`

So far we know how to write `STeX` documents, (we assume) how to build `PDF` files from them (via `pdflatex` of course), and on saving documents the `IDE` will preview the generated `HTML`. But if we do that with our new `all.en.tex`, we get presented with Figure 13. Where did all of our fragments go?



Figure 13: Missing Fragments in the `HTML` Preview

Well, they don’t exist yet as `HTML`. The `HTML` Preview window in the `IDE` is really just that: A *preview*. But when using `\inputref`, it has to find the `HTML` of the `\inputref`ed fragment *somewhere*. Meaning: we have to compile all of the fragments



Figure 14: The Build PDF/XHTML/OMDoc Button

we used to [HTML](#) first. Individually, we can compile the currently open file in [VS Code](#) using the button in Figure 14.

This will do the following:

1. Run `pdflatex` over the file three times.
2. Store the resulting `.pdf` in `[archive]/export/pdf/<filepath>.pdf`.
3. Convert the file to [HTML](#) and store it in `[archive]/xhtml/<filepath>.xhtml`.
4. Extract all the semantics and store them as [OMDoc](#) in `[archive]/content/..., [archive]/narration/... and [archive]/relational/....`
5. Construct a search index in `[archive]/export/lucene/....`

Doing all of this for every individual file *in hindsight* would of course be a huge hassle. We can therefore just compile the full [archive](#), folders in an [archive](#), or whole *groups of archives* via right-clicking an element in the [Math Archives](#) viewer in the [STeX](#) tab (Figure 15).

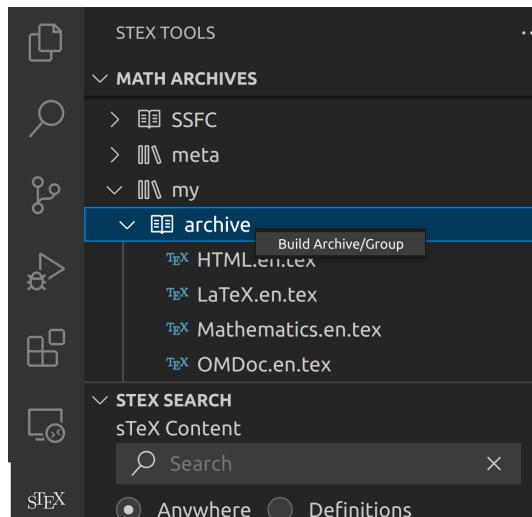


Figure 15: Building Archives in the [IDE](#)

Once that's done, saving `all.en.tex` again yields the correct [HTML](#) in the preview window.

At this point, it should be noted that you can't actually just open the [HTML](#) files exported to [archive]/xhtml in your browser and get all of the expected functionality – that shouldn't be too surprising. Features like the fancy pop-up windows require a semantically informed backend infrastructure, in the form of the [MMT](#) system. However, [MMT](#) *can* dump a standalone version for you. Let's do that now:

With our `all.en.tex` file open and everything built as above, click the `Export Standalone HTML`-button in the [IDE](#) (see Figure 16).



Figure 16: Exporting [HTML](#) in the [IDE](#)

In the dialog box that opens now, select an **empty** directory and [MMT](#) will dump a standalone version of our `all.en.tex` document there. You will still not be able to open it in the browser directly, because most browser forbid javascript modules on the `file://` protocol, but opening the file via `http` will yield the desired result, and you can now upload the directory's content to wherever you might want to use it.

If you want to test this, a quick and easy way to do so is to use [VS Code](#): You can install the `Live Server` extension, open the directory and click the `Go Live` button on the lower right of the window, which will start a small web-server in the selected directory and open its `index.html` in the browser for you.

Chapter 3

Mathematical Concepts

So far, we have seen how to declare and reference `symbols` generate `semantic macros` for `text symbols`, collect them in `modules` and document them properly.

But where `sTeX` really shines is when it comes to mathematics and related subject areas: `semantic macros` are significantly more useful when used for generating symbolic `notations` in math mode, and by associating `symbols` with (flexi-)formal semantics, `sTeX` can even *check* that your content is (to some degree) formally correct, or at least well-formed.

Also `sTeX` provides specialized functionality for mathematical `statements`: the text fragments marked as Definition, Theorem, Proof that are iconic to mathematical documents.

The example snippets in this chapter can be found in the `math archive sTeX/MathTutorial`. If you downloaded the `sTeX/Documentation archive` in the `sTeX IDE`, you already have that `archive`. If not, you can download it from within the `IDE`, as described in ??.

3.1 Simple Symbol Declarations

We will start with `symbols` and `semantic macros` for mathematical concepts and objects and their contribution to mathematical formulae.

3.1.1 Semantic Macros and Notations

Let us start with a very fundamental concept; namely `equality`. As you should by now know, declaring a new `symbol` requires a `module`, so let's open a new one and use `\symdecl`:

```
\begin{smodule}{Equality}
\symdecl{equal}
\end{smodule}
```

As mentioned in ??, the starred variant `\symdecl*` does not create a `semantic macro`, so presumably, the variant without a * *does*. And indeed, we now have a macro `\equal`, which however will produce errors if we try to use it. That's because we haven't told `sTeX` what to do with it yet.

A **semantic macro** is a **LATEX**-macro that allows for referencing a **symbol** itself, or – in the case of e.g. a function – the *application* of a **symbol** to (one or multiple) *arguments*; primarily by invoking a **symbol**'s **notation** in *math mode*.

The command `\symdecl{macroname}` declares a new **symbol** with name **macroname** and a **semantic macro** `\macroname`. In the case where we want the name and the **semantic macro** to be distinct, the command `\symdecl{macroname}[name=some name]` declares the name of the **symbol** to be **some name** instead.

The starred variant `\symdecl*{name}` declares the concept with the given name, but does not generate a **semantic macro**.

So let's provide equality with a **notation**. As a first step, we should let **STEX** know that “**equal**” takes two arguments. We might also want to shorten the **semantic macro** to e.g. `\eq`, without changing the name. Hence:

```
\symdecl{eq}[name=equal,args=2]
```

Next, we add an infix notation with the **notation** macro:

```
\notation{eq}{#1 = #2}
```

That seems like a lot to write, so for the very common case where we want to declare a **symbol** with a **semantic macro** and a **notation** all at once, the `\symdef` macro does all three by combining the optional and mandatory argument of `\symdecl` and `\notation`:

```
\symdef{eq}[name=equal,args=2]{#1 = #2}
```

and indeed, we can now use the `\eq` **macro** in math mode to invoke our new **notation**: $\backslash eq\{a\}\{b\}$ now yields $a = b$ – notably without any highlighting (and hover interaction in the **HTML**) though. Since our **semantic macro** takes *arguments*, which should be differently highlighted, we need to let our **notation** know which parts of the **notation** are highlightable components.

We can do so with the `\comp` and `\maincomp` macros:

The `\comp`-macro marks components to be highlighted in a **notation** for a **symbol** taking (one or more) arguments.

This is necessary because it is (nearly) impossible for **LATEX** to figure out, which parts of a **notation** to highlight and which not on its own – in particular, the highlighting should stop for the *arguments* of a **semantic macro**.

Additionally, the `\maincomp` macro can be used to mark (at most) one **notation** component to represent the *primary* component of the **notation**.

Notations that do not take arguments, as well as **operator notations**, are automatically wrapped in `\maincomp`.

In our case, this applies only to the “ $=$ ”, symbol, so:

```
\symdef{eq}[name=equal,args=2]{#1 \mathrel{\maincomp{=}} #2}
```



You may be wondering about the role of the `\mathrel` macro in the example above: **TeX** determines spacing/kerning in math mode by assigning a *class* to every character. Both individual characters and whole subexpressions can be assigned one of these classes using dedicated macros. These are:



class	TeX macro	examples
ordinary (default class)	<code>\mathord</code>	$\alpha i \diamond$
large operator	<code>\mathop</code>	$\sum \prod \int$
opening	<code>\mathopen</code>	([{
closing	<code>\mathclose</code>)] }
binary relation	<code>\mathrel</code>	$\leq > =$
binary operator	<code>\mathbin</code>	$+ \cdot \circ$
punctuation	<code>\mathpunct</code>	, ;

TeX “forgets” the class of an expression if it is wrapped in a `\comp` macro. It is therefore a good idea to wrap any occurrence of a `\comp` in the corresponding TeX macro for the desired class (e.g. `\mathrel{\comp{\leq}}`).

Having done so, we can now type `$\leq{a}{b}$` to get $a = b$. Thanks to using `\maincomp`, we now also have an **operator notation**, which we can invoke using `$\eq!`, yielding $=$.

What if we want to add more **notations**? Say we want to be able to invoke `equality` to get the variant notation $a \equiv b$ (without changing the intended meaning). If we want to be able to choose one of several **notations**, we should give the **notation** an *identifier*.

Let’s again modify our earlier **notation** by adding the identifier `eq` to the optional arguments of `\symdef`, like so:

```
\symdef{eq}[name=equal,args=2,eq]{#1 \mathrel{\maincomp{=}} #2}
```

We can now invoke the specific **notation** provided here by writing `$\eq[eq]{a}{b}$` to the same effect. But we can also add more **notations** using the `\notation` macro:

```
\notation{eq}[equiv]{#1 \mathrel{\maincomp{\equiv}} #2}
```

which we can now invoke with `$\eq[equiv]{a}{b}$`, yielding $a \equiv b$.

By default, the *first* **notation** provided for a given **symbol** is considered the *default notation*, which is invoked if the **semantic macro** is used without an optional argument – hence, `$\eq{a}{b}$` still yields $a = b$.

If we use the starred variant of the `\notation` macro, the **notation** is set as the new default. Hence, had we done

```
\notation*[eq][equiv]{#1 \mathrel{\maincomp{\equiv}} #2}
```

then `$\eq{a}{b}$` would now yield $a \equiv b$.

Any already existing notation can be set as default using the `\setnotation` macro; e.g. instead of using `\notation*`, we could also do

```
\notation{eq}[equiv]{#1 \mathrel{\maincomp{\equiv}} #2}
\setnotation{eq}{equiv}
```

Exercise

Implement the **symbol** “equal” as above in a new **module** “Equality” and add a documentation such that hovering over the **symbol** in the **HTML** yields the following snippet:

Two objects a, b are considered **equal** (written $a = b$ or $a \equiv b$), if there is no property that distinguishes them.

Lösung: Can be found in [sTeX/MathTutorial]/mod/Equality1.en.tex

3.1.2 Types and Variables

You might have noticed – after you save the file – that the expressions `\eq{a}{b}` and `\eq[equiv]{a}{b}` are underlined in yellow in the **IDE** and have a warning attached to them (Figure 17). If we click on the **Invalid Unit** link in the error message, we get

```
$\text{\eq{a}{b}}$ or $\text{\eq[equiv]{a}{b}}$,  

\eq{a}{b}  

invalid unit:  

http://mathhub.info/sTeX/MathTutorial/mod/Equality1/Equality?en?term 1?definition: Judgment |-- (implicit bind  

[a:/I/1, b:(/I/2 a)] (apply (apply equal a) b)) ::  

/omitted_type (Invalid Unit)  

View Problem (Alt+F8) No quick fixes available
```

Figure 17: Type Checking Warning

a somewhat cryptic stacktrace-like window (Figure 18). The reason being, that **MMT**

<http://mathhub.info/sTeX/MathTutorial/mod/Equality1/Equality?en?term 1?definition>

- - Judgment $\{ \} \dashv \{a: /I/1, b: /I/2 (a) \}_{I,a=b} :: /omitted_type$
 - trying typing rules
 - trying to simplify /omitted_type
 - no rule applicable
 - trying inference/typing rules
 - inferring type
 - inferring type of $\{a: /I/1, b: /I/2 (a) \}_{I,a=b}$
 - applying inference rule rule LambdaLikeRule\$LambdaTypingRule for implicit bind
 - Judgment $\{ \} \dashv /I/1 INHABITABLE$

Figure 18: Type Checking Proof Tree

actually tries to formally verify *everything we write using semantic macros!* It does so, by attempting to infer the *type* of an expression – success implies that the expression is in fact well-typed.

If the former paragraph is difficult to comprehend for you, don't worry – you'll likely pick up on things as we go along. For now, suffice it to say that we can assign “*types*” to *symbols*, and the **MMT** system is smart enough to use those to check that what we're writing actually “makes sense”; for example, $a + b$ makes perfect sense if $+$ is addition and a and b are numbers, or elements of a vector space, but not if a and b are, say, triangles.

 Every **symbol** or **variable** can be assigned a **type**, signifying what “kind of object” the **symbol** represents, or what (primary) set it is contained in.

In order to *formally verify* a mathematical statement, we have to rely on a set of *rules* that determine what is or isn’t a valid statement. There are many systems of such rules with very different flavours, called **(logical) foundations**.

The most commonly used **foundation** in (informal) mathematics is *set theory*, in particular *ZFC*; a set of axioms in (usually) *first-order logic*. However, in *computer proof assistants* and similar systems, *type theories* like *higher-order logic* or the *calculus of (inductive) constructions* are more popular, because they lend themselves better to computer implementations.



In as far as possible, we prefer to remain “foundationally agnostic”, or **foundation independent**: Every **foundation** has advantages and disadvantages, and which one is appropriate often depends on the particular setting one is working in. Nevertheless, certain “meta-principles” have proven themselves to be extremely effective in representing and checking mathematical content in software, and while we do not fix a particular **foundation** or specific checking rules, we will make use of those principles in general. These include e.g. the *Curry-Howard Correspondance*, or *Judgments-as-Types paradigm*, and *Higher-Order Abstract Syntax*.

 Full formal verification of document content is an extremely lofty goal, and hardly realistic if you’re not willing to write your content in pretty specific ways, and informed by a decent amount of background knowledge in formal logic. Moreover, formally verifying content in **STEX** is an ongoing research project, so we will not go into the specifics in detail here.

While full formal verification is out of reach for now, annotating adequate **types** can strike a useful balance between the effort required and the benefit of automated meaning checking afforded by them. In this sense **STEX** is pragmatically similar to programming languages where adding types can raise the quality and correctness assurance in programs.



Keep in mind that getting **Invalid Unit** warnings does not impact at all what your document is going to look like – feel free to ignore them entirely.

Types are particularly useful for **variables**:

A **variable** represents a *generic* or *unspecified* object.

 **Variables** can be declared using the `\vardef`-macro, whose syntax is analogous to `\symdef`.

Note that **variables** are local to the current **T_EX**-group (e.g. environment).

Let’s leave our equality-**module** aside for now and turn our attention to something simpler: **natural numbers**. Consider the following module:

Example 5

Input:

```
\begin{smodule}{Nat}
  \symdef{Nat}[name=natural numbers]{\mathbb N}
  \begin{sparagraph}[style=symdoc]
    The \defname{Nat} $\defnotation{\Nat}$ are the numbers
    $0,1,2,\dots$ 
  \end{sparagraph}
  \symdef{plus}[name=addition,args=2]{#1 \mathbin{\maincomp{+}} #2}
  \begin{sparagraph}[style=symdoc]
    \Defname{addition} $\defnotation{\plus{a}{b}}$-
    refers to the process of adding two \sn{Nat}.
  \end{sparagraph}
\end{smodule}
```

Output:

```
The natural numbers N are the numbers 0,1,2,....
Addition a+b refers to the process of adding two natural numbers.
```

(like `\defname` and `\definiendum`, the `\defnotation` macro is only allowed in documenting environments like `sparagraph[style=symdoc]` or `sdefinition`, and highlights the `notation` components marked with `\comp` or `\maincomp` the same way as `\defname` and `\definiendum` do.)

Note, that as the `\Nat` semantic macro does not take any arguments, we do not need to wrap the `notation` in a `\comp` or `\maincomp`.

Note also, that the `\plus{a}{b}` is again underlined in the IDE with an `Invalid Unit` warning.

The above fragment uses two `variables` *a* and *b*. In fact, `MMT` will consider them `variables` even though they are not marked up as such – but since they are not marked up, we are missing out on useful functionality.

Let's change that by adding two `variable` definitions¹:

Example 6

Input:

```
\begin{sparagraph}[style=symdoc]
  \vardef{va}[name=a]{a}\vardef{vb}[name=b]{b}
  \Defname{addition} $\defnotation{\plus{\va}{\vb}}$-
  refers to the process of adding two \sn{Nat}.
\end{sparagraph}
```

Output:

```
Addition a+b refers to the process of adding two natural numbers.
```

Okay, so now *a* and *b* are gray, but besides that, we haven't achieved much yet.

¹Technically, this is called a *variable reservation*, for those in the know.

Let's change that by giving the variables the type \mathbb{N} :

Example 7

Input:

```
\begin{sparagraph}[style=symdoc]
\vardef{va}{name=a,type=\Nat}{a}\vardef{vb}{name=b,type=\Nat}{b}
\Defname{addition} $\defnotation{\plus{\va}{\vb}}$ refers to the process of adding two \sn{\Nat}.
\end{sparagraph}
```

Output:

Addition $a+b$ refers to the process of adding two natural numbers.

Now if we hover over the a and b (in the [HTML](#)), it will show us that their type is \mathbb{N} !

We can of course also assign [types](#) to [symbols](#). In the [IDE](#), find the [symbol “function space”](#) with [semantic macro](#) `\funspace` (in `[sTeX/MathBase/Functions]{mod?Function}`). The [OMDoc](#) preview window shows you how to use this [symbol](#) (Figure 19). This tells

▼ Symbol <code>function space (\funspace{a_1, ..., a_n}{b})</code>		
Type	$(A : \text{SET}, B : \text{SET}) \rightarrow \text{SET}$	
Notations	id	notation
	arrowtimes	$a_1 \times \dots \times a_n \rightarrow b$
	arrowcurry	$a_1 \rightarrow \dots \rightarrow a_n \rightarrow b$
	Arrowtimes	$a_1 \times \dots \times a_n \Rightarrow b$
	Arrowcurry	$a_1 \Rightarrow \dots \Rightarrow a_n \Rightarrow b$

Figure 19: Syntax Preview

us that if we write `\funspace{a_1, ..., a_n}{b}` (depending on which notation we use), we will get $a_1 \times \dots \times a_n \rightarrow b$.

We want `addition` to have type $\mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$, hence we do:

```
\syndef{plus}[name=addition,args=2,
type=\funspace{\Nat,\Nat}{\Nat}
]{#1 \mathbin{\maincomp{+}} #2}
```



So far (and when using the `use` button in the [IDE](#)), we have been using the `\usemodule` macro to import content. `\usemodule` is allowed anywhere and imports the referenced `module` content local to the current [TeX](#) group.

Now that we use imported `symbols` in `types` (and since we are *in* a `module`), we



need to make sure that the imported `modules` are also (transitively) *exported*, since our new `symbols` now *depend* on the imported `module`. For that we use the `\importmodule` macro within the `module`; i.e. the file should now look something like this:

```
\begin{smodule}{Nat}
\importmodule[sTeX/MathBase/Functions]{mod?Function}
...
```

Note that the `HTML` is aware of this now (after you save): *Clicking* on any occurrence of `addition` now yields Figure 20.

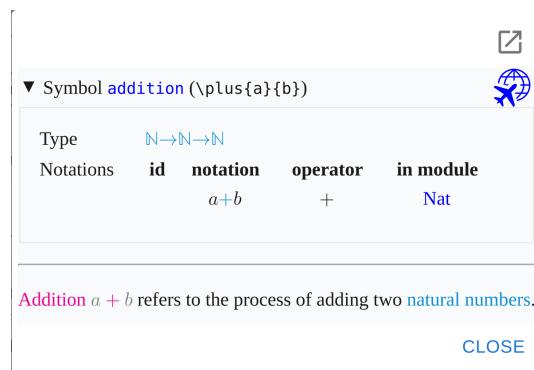


Figure 20: On-Click Popup in the `HTML`

However, the squiggly yellow `Invalid Unit` warnings are still there – that's because everything we did with `types` so far still depends on our `natural numbers symbol`, which does not have a `type` yet.

By virtue of using `[sTeX/MathBase/Functions]{mod?Function}`, we also imported `[sTeX/MathBase/Sets]{mod?Set}`, which gives us the “`collection`” `symbol`. Let's use this as a `type` for the `natural numbers`:

```
\symdef{Nat}[name=natural numbers,type=\collection]{\mathbb{N}}
```

Now if we save the file, all the squiggly lines are gone. Moreover, if you look at the `OMDoc` tab in the preview window, you can find Figure 21. The **Document Elements** block collects all semantically annotated expressions in a `module` or document; including `variables` and the `$\plus{\va}{\vb}$`. Here, it tells us that it has checked the expression $a + b$ (in the context of $a : N$ and $b : N$), and inferred that it has `type N`.

Here's what just happened:

1. The `MMT` system realized, that `$\plus{\va}{\vb}$` is the symbol “`addition`” applied to the two arguments a and b .
2. It knows, that “`addition`” has `type N × N → N`².

²Do not worry that the IDE actually reports the type `{a : N, b : N} ⊢ N`, this is an artefact of the underlying type system with dependent types used by `sTeX`; it just means $N \times N \rightarrow N$ in this special case, but would also allow a and b to appear in the range type in more complex situations; see ?? for details.

▼ Document Elements

- Variable `a` (`\va`) of type `N`
- Variable `b` (`\vb`) of type `N`
- ▼ $\{a: N, b: N\}_I^{a+b}$

Inferred Type: $\{a: N, b: N\}_I^N$

Figure 21: Inferred Type

3. It knows, that this means that if the two arguments `a` and `b` both have type `N`, then the full expression has type `N`.

Here's something you can now try: If we *remove* the types from the variables `a` and `b` again, the warnings are *still* gone. We lose the type information on hover, but MMT still doesn't complain, because it now realizes that since `a` and `b` have no explicit types given, it should infer them. And by the same chain of reasoning as above, it can infer that since they are being used as arguments for addition, they need to have type `N`.

3.1.3 Flexary Macros and Argument Modes

Here is one thing you might wonder: Writing `$\plus{a}{b}$` is one thing, but what if we want to produce $a + b + c + d + e$? Do we really need to write `$\plus{a}{\plus{b}{\plus{c}{\plus{d}{e}}}}$`?

Of course not. We can declare the symbol such that the semantic macro `\plus` expects a (comma-separated) sequence of arguments instead of two “normal” arguments.

The optional `args`-argument of `\symdecl` expects a string of characters indicating the semantic macro's argument modes. There are four such modes:

- STEX**
- i a simple argument,
 - a a – (left or right) associative – sequence argument, represented as a single TeX-argument `{a,b,...}`,
 - b A binding argument that expects a variable that is bound by the symbol in its application, and
 - B A binding sequence argument of arbitrarily many bound variables by the symbol `({x,y,z,...})`.

If `args` is given as a number `n` instead, the semantic macro takes `n` arguments of mode i.

Example 8

- For `\plus{a,b,c}` yielding $a + b + c$, we do `\symdecl{plus}[args=a]`,
- for `\inset{a,b,c}{A}` yielding $a, b, c \in A$, we do `\symdecl{inset}[args=ai]`,
- in `\add{i}{1}{n}{f(i)}` yielding $\sum_{i=1}^n f(i)$, the variable `i` is bound in the expression, we hence do `\symdecl{add}[args=biii]`,

- in `\foral{x,y,z}{P(x,y,z)}` yielding $\forall x, y, z. P(x, y, z)$, the variables x, y, z are all **bound** by the \forall , we hence do `\symdecl{foral}[args=Bii]`.

So when we wrote `\symdecl{plus}[args=2]`, this was actually shorthand for `\symdecl{plus}[args=ii]`.

Let's revise our previous declaration and the syntax of the `\plus` macro:

```
\symdef{plus}[name=addition,args=a,
  type=\funspace{\Nat,\Nat}{\Nat}
] {#1 \mathbin{\maincomp{+}} #2}
\begin{sparagraph}[style=symdoc]
  \vardef{va}[name=a]{a}\vardef{vb}[name=b]{b}
  \Definename{addition} $ \defnotation{\plus{\va,\vb}}$%
    refers to the process of adding two \sn{\Nat}.
\end{sparagraph}
```

Now we get new errors, that are easy to explain: Our **notation** `{#1 \mathbin{\maincomp{+}} #2}` refers to *two* arguments, but our **semantic macro** only takes *one* (albeit a **sequence argument**). We now need to let **sTeX** know what to do with the **sequence argument** in our **notation**. Using the `\argsep` macro, we can tell **sTeX** to insert the *separator* “`+`” between the individual elements of the **argument sequence** `#1`:

```
\symdef{plus}[name=addition,args=a,
  type=\funspace{\Nat,\Nat}{\Nat}
] {\argsep{#1}{\mathbin{\maincomp{+}}}}
```

Now we can finally write `$\plus{a,b,c,d,e}$` and get $a + b + c + d + e$ – hooray! ...expect that our squiggly yellow **Invalid Unit** warnings are back. That's because the **type** of **addition** still corresponds to a binary operation, rather than a unary function on sequences.

We *could* change the **type** of course, but we shouldn't *want* to or *have* to: platonically, **addition** is *still* a *binary function*; we just introduced the **a-mode** argument for *our convenience as authors*.

Instead, we can tell **MMT** how to “resolve” the **sequence argument** into a nested application of **addition**. In the very common case we have here, where the **symbol** represents an *associative binary operator*, we can just add the argument **assoc=bin** to the `\symdecl` (or `\symdef`) **macro**:

```
\symdef{plus}[name=addition,args=a,assoc=bin,
  type=\funspace{\Nat,\Nat}{\Nat}
] {\argsep{#1}{\mathbin{\maincomp{+}}}}
```

and the warnings are gone again. Formally/internally, **MMT** will now turn the term `addition(sequence(a,b,c))` into `addition(a,addition(b,c))`.

Exercise

Analogously to the above, implement a **symbol** “**multiplication**” with **semantic macro** `\mult`, that takes a single **sequence argument** and has a default **notation** such that `\mult{a,b,c}` produces $a \cdot b \cdot c$.

Lösung: Can be found in `[sTeX/MathTutorial]mod/Nat.en.tex`

3.1.4 Precedences

If you have done the previous exercise, you now have *semantic macros* `\plus` and `\mult` at your disposal. We can of course nest them to produce e.g. $a + b \cdot c$ (with `\plus{a, \mult{b, c}}`). If we do `\mult{a, \plus{b, c}}` however, we get $a \cdot b + c$. Annoying – we now have to insert parentheses: `\mult{a, (\plus{b, c})}`... or do we?

We do *not*. Instead, we can assign *precedences* to *notations* to have *STEX* insert parentheses automatically.

notation (and hence *\symdef*) take an optional argument `prec=<opprec>;<argprec1>x...x<argprec n>` consisting of an **operator precedence** `<opprec>` and for each argument `k` an **argument precedence** `<argprec k>`.

All *precedences* are integers, e.g. 10 or -500. It is good practice to use *precedences* that leave enough room to smuggle values inbetween, so that we can fine-tune them later as more symbols may intervene.

The precise numbers used for *precedences* are arbitrary – what matters is which *precedence* is higher than which other *precedence* when used together.

By default, all *precedences* are 0, unless the *symbol* takes no arguments, in which case the *operator precedence* is `\neginfprec` (negative infinity).

If we only provide a single number in `prec=`, this is taken as both the *operator precedence* and all *argument precedences*.

The *lower* a *precedence*, the *stronger* a *notation* binds its arguments. In our particular case, we want *multiplication* to bind stronger than *addition*, so we can (arbitrarily) assign them *precedences* e.g. 10 and 20:

```
\symdef{plus}[name=addition,args=a,assoc=bin,prec=20,
  type=\funspace{\Nat,\Nat}{\Nat}
] {\argsep{\#1}{\mathbin{\maincomp{+}}}}
\symdef{mult}[name=multiplication,args=a,assoc=bin,prec=10,
  type=\funspace{\Nat,\Nat}{\Nat}
] {\argsep{\#1}{\mathbin{\maincomp{\cdot}}}}
```

And now if we type `\mult{a, \plus{b, c}}`, *STEX* will automatically insert parentheses and yield $a \cdot (b + c)$ – and conversely, if we do `\plus{a, \mult{b, c}}`, *STEX* will *not* insert parentheses and yield $a + b \cdot c$.

3.1.5 Implicit Arguments

Let us turn our attention back to *equality*. Here's an almost philosophical question: *What is the type of “equality”?* Asking (the right kind of) mathematicians this question can cause fist fights to break out. As such, we will not give a definitive answer, *but* here is an informative approach that has proven to be quite effective in computational settings:

Equality is a *polymorphic binary relation* on an *implicit collection A*. And a *relation* is a function into a *type of propositions*.

We will see the advantage of this approach over time. For now, consider that given objects a and b , the expression “ $a = b$ ” is either true or false³, and “*equal*” takes two argu-

³Assuming classical logic – if you prefer to remain intuitionistic/constructive, note that *STEX*, being *foundation independent*, does not enforce the law of excluded middle!

ments, so if we have a `type` of “truth values”, it makes sense to model “`equal`” as a function taking two arguments and returning that `type`. So we do `type=\funspace{...}`?

Here’s the idea with respect to *implicit arguments*. Let’s first declare a new `variable` of `type “collection”`:

```
\vardef{\vA}[name=a,type=\collection]{A}
```

We now assign the `type` $A \times A \rightarrow \text{Prop}$ to `equal`:

```
\symdef{eq}[name=equal,args=2,eq,
  type=\funspace{\vA,\vA}{\prop}
]#1 \mathrel{\maincomp{=}} #2
```

(The symbol “`proposition`” with `semantic macro` `\prop` comes with `STEX` directly; we say that it is part of the `STEX`.)

Now our `type` has a free variable A . For `MMT`, this now means that `equal` actually takes *one more argument*, but one whose value is uniquely determined from the other arguments. Indeed, if you consider `equal` to take three arguments (the first one being some A of `type collection`), then the *next* two arguments *enforce* that the first argument has to be the `type` of the other two.

In other words: A is now an implicit argument that `MMT` is tasked with inferring whenever we use `equal`, and that we never explicitly provide in `STEX`.

Indeed, if we use our `module Nat` from before, and apply `\eq` to a variable of type \mathbb{N} , `MMT` does not complain:

```
\usemodule{mod?Nat}
\vardef{\vn}[name=n,type=\Nat]{n}
\$ \eq{\vn}{m} \$
```

And if we inspect the `OMDOC` tab in the `HTML` preview, we can see exactly what `MMT` did (Figure 22). We can see

The screenshot shows the MMT interface with the 'OMDOC' tab selected. The main area displays the following information:

- Module Equality**
- Includes Function**
- Symbol equal (`\eq{a}{b}`)**
 - Type: $\{A: \text{SET}\}_I A \rightarrow A \rightarrow \text{Prop}$
 - Notations: id notation operator in module

eq	$a = b$	=	Equality
equiv	$a \equiv b$	≡	Equality
- Document Elements**
- Includes Nat**
- Variable n (`\vn`) of type \mathbb{N}**
- Inferred Type: $\{n: \mathbb{N}, m: \mathbb{N}\}_I \text{Prop}$**

Figure 22: Implicit Arguments

1. (by the $\{\cdot\}_I \dots$) that `MMT` considers A an implicit argument in the `type` of `equal`,

2. that the *inferred* type of $n = m$ is `Prop`,
3. that `MMT` inferred the implicit argument of `equal` in $n = m$ to be \mathbb{N} (by the \dots),
and
4. that it was enough to give \backslashvn the explicit `type` \mathbb{N} – `MMT` also inferred that hence m also has to have `type` $\mathbb{N}!$

3.1.6 Finishing Equality

You might wonder if – as with `addition` – we can make “`equal`” take a `sequence` argument as well. Naturally, we can:

```
1 \symdef{eq}{name=equal,args=a,eq,
2   type=\funspace{\vA,\vA}{\prop}
3   }{\argsep{\#1}{\mathrel{\maincomp}}}
4 \notation{eq}[equiv]{\argsep{\#1}{\mathrel{\maincomp\equiv}}}
```

and as before, we now get `Invalid Unit` warnings. Unlike before, however, we can not just fix this with adding `assoc=bin`. As mentioned, `bin` instructs `MMT` to “fold” the `symbol` over the arguments, so when doing `\eq{a,b,c}`, `MMT` would turn this into `equal(a,equal(b,c))`, i.e. the claim that “ a ” is equal to “ $b = c$ ” – but that’s not what $a = b = c$ means. What we mean by $a = b = c$ is really “ $a = b$ and $b = c$ ”.

For that, we can use `assoc=conj` – however, that requires that some `symbol` that can be used for *conjunction* (i.e. “and”) is in the current scope.

If we search for `conjunction` in the `IDE`, we should find the `module`
[`sTeX/Logic/General`] {`mod/syntax?Conjunction`}.

Using that, we can now write the following:

```
\usemodule{mod?Nat}
\usemodule[sTeX/Logic/General]{mod/syntax?Conjunction}
\vardef{vn}{name=n,type=\Nat}{n}
\$ \eq{\vn,m,p} $
```

Upon saving, `MMT` does not complain; and if we inspect the `OMDoc` tab in the `HTML` window again, we now notice that `MMT` correctly resolved this as in Figure 23.

3.1.7 Variable Sequences

There is a special kind of `variable` in `STEX` for when we want to use *sequences* of `variables`.

We can use the `\varseq` macro to declare a new sequence `variable`; in the simplest case that looks something like the following:

```
\varseq{seqn}{name=n,type=\Nat}{1,\ellipses,k}{\maincomp{n}_{\#1}}
```

We have just declared a new variable sequence of `type` \mathbb{N} , that ranges over indices $1, \dots, k$, with `notation` n_i for some specific index i .

If we now do `\seqn{i}`, we get n_i , and if we do `\seqn!`, we get n_1, \dots, n_k .

We can also do multi-dimensional sequences, e.g.

```
\varseq{seqm}{name=m,type=\Nat, args=2}
{\{1\}\{1\},\ellipses,\{\ell\}\{k\}}
{\maincomp{m}_{\#1}^{\#2}}
```

The screenshot shows a software interface for defining mathematical structures. At the top, there's a section titled "Module Equality". Below it, under "Includes Function", is a symbol "equal" (\eq{a_1, ..., a_n}) with a small globe icon next to it. Under "Document Elements", there's a section for "Nat, Conjunction". It shows a conjunction of equalities: $\{n: \mathbb{N}, m: \mathbb{N}, p: \mathbb{N}\} \frac{n=m \wedge m=p}{\mathbb{N}}$. Below this, in a box, is the "Inferred Type": $\{n: \mathbb{N}, m: \mathbb{N}, p: \mathbb{N}\} \text{Prop}$.

Figure 23: Conjunction of Equalities

Now $\text{\seqm}{i}{j}$ produces m_i^j , and $\text{\seqm}!$ produces m_1^1, \dots, m_ℓ^k .

Of course, we can manually change the way $\text{\seqn}!$ is typeset by providing an explicit **operator notation** using `op=`; e.g. if we do

```
\varseq{\seqn}[name=n,type=\Nat,op={(n_i)_{i=1}^k}]
  {1,\dots,k}\{\maincomp{n}_{\#1}\}
```

then $\text{\seqn}!$ produces $(n_i)_{i=1}^k$.

So far so nice, but sequence variables get especially useful in combination with **sequence arguments**: Consider for example the `\plus` semantic macro for **addition**. This expects one **sequence argument**, or alternatively, a **sequence variable**: `\plus{\seqn}` now produces $n_1 + \dots + n_k$, and `\eq{\seqm}` now produces $m_1^1 = \dots = m_\ell^k$.

TODO⁴

3.2 Statements

Now that we have **equality**, **natural numbers**, **addition** and **multiplication** at our disposal, let's implement some *statements*. Both **addition** and **multiplication** are, for example, *associative* and *commutative*.

We could state these properties directly for the two operations, but we can also first define *associativity* and *commutativity* in general, and then assert them specifically for **addition** and **multiplication**.

3.2.1 Definitions

Let's define what it means to be *associative*. This means, of course, declaring a new **symbol**. Note that we don't need a **semantic macro** for **associativity**, since there is no **notation** to attach to it. We will also for now ignore its **type**. Note however, that **associativity** is still a property of (binary) operations, so it still makes sense to have the **symbol** take an *argument*; namely the operation it applies to.

⁴TODO: seqmap

We will also finally provide an actual (more or less) formal *definition* for the `symbol`, so where we used the `sparagraph` environment with `style=symdoc` before, we will now use the `sdefinition` environment, which also gives us `\defname`, `\definiendum`, `\defnotation` and all that.

A first variant of a corresponding `module` could look like this:

Example 9

Input:

```
File [sTeX/MathTutorial]props/Associative1.en.tex
4 \begin{smodule}{Associative}
5   \importmodule{mod?Equality}
6
7   \symdecl*[associative][args=1]
8   \begin{sdefinition}[for=associative]
9     \vardef{vA}[name=A,type=\collection]{A}
10    \vardef{vop}[name=op,type=\funspace{\vA,\vA}\vA,args=a,assoc=bin]
11    {\argsep{\#1}{\mathbin{\maincomp{\circ}}}}
12    %
13    A binary operation \$\fun{\vop!}{\vA,\vA}\vA\$ is called
14    \defname{associative}, if
15    \$\eq{
16      \vop{(\vop{a,b}),c},
17      \vop{a,(\vop{b,c})}
18    }$ for all \$\inset{a,b,c}\vA$.
19  \end{sdefinition}
20 \end{smodule}
```

Output:

Definition 3.2.1. A binary operation $\circ : A \times A \rightarrow A$ is called **associative**, if $(a \circ b) \circ c = a \circ (b \circ c)$ for all $a, b, c \in A$.

Note, that the **semantic macros** `\fun` and `\inset` come from `[sTeX/MathBase/Functions]mod?Function` and `[sTeX/MathBase/Sets]mod?Set`, respectively. Also note, that the **variable** declaration for `\vop` makes use of all the fun features we already discussed for `addition`.

Note that the above is more than good enough, if you merely want to produce nice-looking, “wikified” **HTML** and **PDF** documents. The rest of this subsection will cover how to add more flexiformal semantics to the above.



If this seems laborious and/or difficult, keep in mind that this is to some degree experimental still, and you are not forced to go overboard with semantic annotations!

But if you aim to create a “library of symbols” for mathematical concepts, then all of the possibilities that we discuss here will add value for the community. Generally, the higher the ratio of readers to authors the more any investment in semantization will pay off.

Semantic Macros in Text Mode

The first thing we can do to further improve this document is marking up the “for all” in the definition – after all, there naturally is a `symbol` for the `universal quantifier`, which can be found in `[sTeX/Logic/General]mod/syntax?UniversalQuantifier` and has the `semantic macro` `\foral` (as to not conflict with the `TeX` primitive `macro` `\forall`).

The naive approach would be to replace the “for all” by e.g. `\sr{foral}{for all}`. That would (correctly) associate and highlight the text fragment with the `symbol` “universal quantifier”, *but* we are not just referencing the `symbol` here – we are actually using it, by *applying* it to the `variables` a, b, c and the expression $(a \circ b) \circ c = a \circ (b \circ c)$.

In *math mode*, we can just use the `semantic macro` `\foral` – that will take two arguments (of `modes` `B` and `i`) and produce the corresponding `notation`, so that

```
$\foral{\inset{a,b,c}{\vA}}{  
    \eq{\vop{(\vop{a,b}),c}, \vop{a,(\vop{b,c})}} }  
}$
```

will produce $\forall a, b, c \in A. (a \circ b) \circ c = a \circ (b \circ c)$.

In *text mode*, however, we don’t have a specific `notation` – instead, the specific “`notation`” is whatever sentence we want to mark up semantically. In text mode, `semantic macros` therefore behave differently:

1. They take *precisely* one argument, regardless of how many arguments the `macro` would take in math mode or (equivalently) the `args` property of the `symbol`.
2. *Within* that argument, we can use `\comp` to highlight arbitrary text fragments, and
3. we can use the `\arg` macro to mark up the *actual* arguments that the `symbol` is supposed to be applied to.

`\arg` takes as optional argument the index of the argument that is being marked up; if not they are used consecutively. The starred variant `\arg*` produces no output.

So we could now do

```
\foral{\comp{For all}}{$\arg{\inset{a,b,c}{\vA}}$, we have  
$ \arg{  
    \eq{\vop{(\vop{a,b}),c}, \vop{a,(\vop{b,c})}} }  
}$}
```

which produces “For all $a, b, c \in A$, we have $(a \circ b) \circ c = a \circ (b \circ c)$ ”.

In our case though, we want to “switch the arguments around” – first comes the equation, then the `variables` to be bound. Hence:

```
\foral{  
    $ \arg[2]{  
        \eq{\vop{(\vop{a,b}),c}, \vop{a,(\vop{b,c})}} }  
    }$  
    \comp{for all}  
    $ \arg[1]{\inset{a,b,c}{\vA}} $  
}
```

which produces “ $(a \circ b) \circ c = a \circ (b \circ c)$ for all $a, b, c \in A$ ”.

Definientia

Now we have a fully semantically annotated expression in the definition for “`associative`”. Can we let `MMT` know, that this expression really is *the* definition of the `symbol`?

Yes, we can. All we need to do is wrap the sentence in a `\definiens` macro (plural: `definientia`; like the word “`definiendum`” refers to “the term being defined”, “`definiens`” refers to “the thing the term is being defined *as*”).

The `\definiens` macro is only allowed within the `sdefinition` environment, and requires that the `environment` lists the `symbol` that gets the definientia attached explicitly in its `for=` argument. It is possible to attach definientia to multiple `symbols` within an `sdefinition environment`, in which case the symbol needs to be provided as an optional argument, e.g. we could do `\definiens[associative]{...}`. Since “`associative`” is the only `symbol` being defined in our definition, we can omit that argument.

Alternatively, as with `types` we can attach definientia to a `\symdecl` directly using the optional argument `def=....`

At this point, you might justifiably wonder, why we even still need to declare `associative` with `\symdecl*` before we define it. And indeed, we don’t – the `sdefinition environment` takes the same optional arguments as the `\symdecl` macro, and if we explicitly provide a `name=` (or a `macro=`), it will generate a `symbol` for us. We can hence get rid of the `\symdecl*` and instead do:

```
1 \begin{sdefinition}[name=associative,args=1]
2 ...
3 \end{sdefinition}
```

One more problem remains: We stated that `associative` is to take one argument – but we haven’t told `STEX` what it is yet. In our case, the argument is represented by the `variable` `\vop`. In fact, chances are that arguments to symbols in `types` or definientia are almost always represented by some `variable`.

We can use one of two ways to a `variable` as being an argument:

1. If the `variable` (e.g. `\vop` with name `op`) was already declared prior to the `sdefinition environment`, we can use the `\varbind` macro in the `environment`; e.g. by adding `\varbind{op}`.
2. We can move (or copy) the `\vardef` for the `variable` into the `environment` and add `bind` to its optional arguments.

In total, our fully annotated definition now looks like this:

Example 10

Input:

```

File [sTeX/MathTutorial]props/Associative.en.tex

8 \begin{sdefinition}[name=associative,args=1]
9   \vardef{vA}[name=A,type=\collection]{A}
10  \vardef{vop}[name=op,type=\funspace{\vA,\vA}\vA,
11    args=a,assoc=bin,bind % <- argument for the symbol
12  ]{\argsep[#1]{\mathbin{\maincomp{\circ}}}}
13  \vardef{va}[name=a,type=\vA]{a}
14  \vardef{vb}[name=b,type=\vA]{b}
15  \vardef{vc}[name=c,type=\vA]{c}
16  %
17  A binary operation $\mathit{fun}(\mathit{vop})\{\vA,\vA\}\vA$ is called
18  \definename{associative}, if
19  \definiens{$\forall a,b,c:A.\, \overline{(a\circ b)\circ c = a\circ(b\circ c)}_A$}
20  \vop{(\vop{\va,\vb}),\vc},
21  \vop{(\va,(\vop{\vb,\vc}))}
22 }$ \comp{for all} $\arg[1]{\inset{\va,\vb,\vc}\vA}$}.
23 \end{sdefinition}
24 %

```

Output:

Definition 3.2.2. A binary operation $\circ : A \times A \rightarrow A$ is called **associative**, if $(a \circ b) \circ c = a \circ (b \circ c)$ for all $a, b, c \in A$.

And indeed, if we look at the **OMDoc** tab of the **HTML** preview, we can see that not only does **MMT** attach the definiens to the **symbol**, it has also inferred the **type** of “**associative**” from the definiens (Figure 24).

Figure 24: Type Inferred from Definiens

Using Symbols Without Semantic Macros and Exporting Code in Modules

So now we don’t have a **semantic macro** for “**associative**”, but it *does* take an argument. How can we ever actually *use* the **symbol** now?

The answer is: with the **\symuse** macro. Like **\symref** and friends, **\symuse** takes a **symbol** name or the name of its **semantic macro** as argument, but behaves otherwise like using a **semantic macro** directly. So for, say, **addition**, **\symuse{addition}** and **\symuse{plus}** behave exactly like **\plus**.

In our case, this means we can do **\symuse{associative}**. “**associative**” does not have a **notation**, but in practice, we say something like “**+ is associative**” rather than using some specific mathematical **notation** for the same thing.

Combining this with what we just learned, we can now say that `addition` is associative by doing:

```
\symuse{associative}{$\arg{\plus!}$} \comp{is associative}
```

In fact, we would do the exact same thing every time we want to say that *some* operator is associative, so it makes sense to introduce a `macro` for this. In fact, such a `macro` is easy to define using standard `LATEX` methods. This is where `\STEXexport` becomes very handy:

In a `module`, we can put arbitrary `LATEX` code in an `\STEXexport`, and this code will be executed every time the `module` is imported via `\usemodule` or `\importmodule`. This is especially useful for `macro` definitions, and this way `modules` can almost act like `LATEX` packages!

So we can define a new `macro` `\isassociative` that applies “`associative`” to an arbitrary operation and produces the semantically marked-up text “#1 is `associative`”, and wrap that `macro` definition in an `\STEXexport`, and whenever we use the `Associative module`, we also get the `\isassociative` `macro`:

```
\STEXexport{
  \def\isassociative#1{
    \symuse{associative}{$\arg{\#1}$} ~is ~\comp{associative}
  }
}
```

And now, we can do e.g. `\isassociative{\plus}` to produce “`+ is associative`”.

For technical reasons, `\STEXexport` processes its content in the `expl3` category code scheme – what this means is that all spaces are ignored entirely, and the characters `_` and `:` are valid characters in `macro` names.



In practice, this means you will have to use the `-` character for spaces, and if you want to use a subscript `_`, you should use the `macro` `\c_math_subscript_token` instead.

Exercise

Analogously to all the above, implement a `module` for `commutativity`; i.e the property of a binary operation that $a \circ b = b \circ a$ for all a, b . Make the `module` export a macro `\iscommutative` analogously to `\isassociative`.

Lösung: Can be found in [sTeX/MathTutorial]props/Commutative.en.tex

TODO⁵

3.2.2 Assertions

Having defined `associativity` and `commutativity`, we can now assert that both properties hold for `addition` and `multiplication`.

For `assertions` (i.e. theorems, lemmata, axioms, claims,...), `sTeX` provides the `sassertion environment`.

In the simplest case, that can look like the following:

⁵TODO: intent?

```
\begin{sassertion}
  \isassociative{\Sn{plus}}
\end{sassertion}
```

which yields

Addition is associative

Do we want this to be typeset as a **Theorem**? For that we just add a `[style=theorem]` to the **sassertion environment**, provided we have a customization for that – (see ??). We can also load the **stexthm package**, which uses the **amsthm package** to provide common typesettings for the types: **theorem**, **observation**, **corollary**, **lemma**, **axiom** and **remark**.

So far, this is not too useful – after all, we could have just as well used e.g. the **amsthm package** and gone straight for the non-**STEX** variant

```
\begin{theorem}
  \isassociative{\Sn{plus}}
\end{theorem}
```

But as with **sdefinition**, we can immediately add a corresponding **symbol** in the **sassertion environment**, and have it be documented directly by the **environment**:

```
\begin{sassertion}[style=theorem, name=addition is associative]
  \isassociative{\Sn{plus}}
\end{sassertion}
```

And now, if we do `\sn{addition is associative}`, we get **addition is associative** with a corresponding hover pop-up (in the **HTML**).

Of course, the usefulness of this grows with more elaborate assertions. For very short assertions such as the above, we might not even want to typeset them in such a space hungry manner.

For that purpose, we provide the **\inlineass macro** (and analogously: **\inlinedef** for **sdefinition**), which takes the same optional arguments as the **environment**. So we could also do:

```
\inlineass [name=addition is associative]{\isassociative{\Sn{plus}}}
```

So far, **MMT** is blissfully unaware of the semantic contents of our assertions. We can easily remedy that by wrapping the expression representing the assertion in a **\conclusion macro**, analogously to the **definiens macro** in **sdefinitions**:

```
\inlineass [name=addition is associative]{
  \conclusion{\isassociative{\Sn{plus}}}
}
```

We can now see the statement in the **OMDOC** tab of the **HTML** preview (Figure 25).

$$\triangleright \text{Assertion addition is associative} \vdash \text{apply } \left(\text{apply } \left(\underbrace{\text{associativeN}}_{\mathbb{N}} \right) + \right)$$

Figure 25: Assertion Statement in **OMDOC**

Not exactly pretty – the **OMDOC** tab uses **notations** to render content, and we did not provide any for **associative**.

Notice the \vdash symbol after the name of the assertion? As an aside for those who are curious:

The **judgments as types** paradigm represents the validity of **proposition** via a designated *type of proofs*: For any **proposition** P , we introduce a collection $\vdash P$ of *proofs* of P .
sTeX
To say that the **proposition** *holds* is then equivalent to positing that *some* element $p : \vdash P$ exists – in which case *proofs* become typed objects in their own right.

Let's consider a more interesting statement now. How about the **induction axiom**?

```
\begin{assertion}[style=axiom,name=induction axiom]
  Let  $\varphi(n)$  a property on  $\mathbb{N}$ . If
  \begin{enumerate}
    \item  $\varphi(0)$  and
    \item if  $\varphi(m)$  holds for some  $m$ , then
       $\varphi(\text{plus}\{m, 1\})$  also holds,
  \end{enumerate}
  then  $\varphi(n)$  holds for all  $n \in \mathbb{N}$ .
\end{assertion}
```

Axiom 3.2.3. Let $\varphi(n)$ a property on **natural numbers**. If

1. $\varphi(0)$ and
 2. if $\varphi(m)$ holds for some m , then $\varphi(m + 1)$ also holds,
- then $\varphi(n)$ holds for all $n \in \mathbb{N}$.

Exercise

Annotate the above by:

1. **Variables** with appropriate **notations** for φ , m and n , and
2. marking up the second premise (“if $\varphi(m)$ holds for some...”) in text mode as the formula $\forall m. \varphi(m) \Rightarrow \varphi(m + 1)$ using the **semantic macros** **\foral** (which we saw earlier already) and **\imply** (**implication**) from **[sTeX/Logic/General]mod/syntax?Implication**. The text fragments that should be highlighted are “if” and “then”.
3. marking up the conclusion (“ $\varphi(n)$ holds for all $n \in \mathbb{N}$ ”) in text mode as the formula $\forall n. \varphi(n)$. The text fragment that should be highlighted is “**for all**”.

Lösung: Can be found in **[sTeX/MathTutorial]mod/NatTheorems.en.tex**

So how can we teach **MMT** the semantics of this statement? Here's what we can do:

1. As with the simpler assertions (and hence the name), the *conclusion* of the assertion can be marked up with **\conclusion**.

2. As with `sdefinition`, we can mark `variables` as *bound* (using either `bind` in the `\vardef` or `\varbind`). If a `symbol` that can act as a `universal quantifier` is in scope, `variables` marked as bound are abstracted away using that `symbol`.
3. Similarly to `\conclusion`, `premises` can be marked up as such using the `\premise` macro. If a `symbol` is in scope that can act as an `implication`, that will be used to connect the premise(s) to the conclusion.

Hence, if we mark the variable φ as bound and use `\premise` and `\conclusion` (see [sTeX/MathTutorial]mod/NatTheorems.en.tex), we can inspect the OMDoc tab in the `HTML` preview again and see that MMT has now constructed the proposition (Figure 26).

```
> Assertion induction axiom ⊢ ∀φ:N→Prop.φ(0)⇒( ∀m:N.φ(m)⇒φ(m+1) )⇒( ∀n:N.φ(n) )
```

Figure 26: The Induction Axiom in OMDoc

3.2.3 Proofs



`sTeX` provides the `sproof` environment for marking up *proofs*. The markup mechanism for `sproof` is still highly experimental and likely subject to change in the near future. As such, we omit a closer explanation of its usage until the syntax and functionality have sufficiently stabilized.

3.3 Mathematical Structures

A common concept in mathematics is that of a `mathematical structure` – a *tuple* of interdependent components. For example: A `monoid` is a `structure` $\langle M, \circ, e \rangle$ such that certain axioms hold; where M is a set, \circ is a binary operation, and $e \in M$.

From a representational perspective, this is particularly interesting: M , \circ and e in the above are not `symbols` in the same way that the previous `symbols` we considered were – they don't represent definite objects. Instead, they are *components* of some other object, namely a monoid; where a *particular* monoid could either be a fixed object (such as $\langle \mathbb{Z}, +, 0 \rangle$) or an *indefinite* monoid; i.e. a `variable`. We call the components of a `mathematical structure` `fields`.

In this section, we will discuss how to declare and use `mathematical structures` in `sTeX`, build them up modularly, and connect them among each other to avoid duplication.

We will do so by considering *lattices* both algebraically and order-theoretically, and identify the two perspectives.

3.3.1 Declaring and Using Structures

The simplest kinds of `structures` are *magmas* and *(directed) graphs*, so we might as well start there:

Definition 3.3.1. A **magma** is a **structure** $\langle U, \circ \rangle$, where U is a **collection** and \circ a binary operation $U \times U \rightarrow U$.

The obvious start is to create a new **module** `Magma`. Within this **module**, we import the **Functions module** so we can later assign a **type** to the operation. We can then use the **mathstructure environment**, that creates a new symbol “**magma**”:

```
\begin{smodule}{Magma}
  \importmodule[sTeX/MathBase/Functions]{mod?Function}
  \begin{mathstructure}{magma}
    ...
  \end{mathstructure}
\end{smodule}
```

mathstructure behaves very similarly as **smodule** – within the **environment**, we can declare new **symbols**, **notations** and all that.

So within the **mathstructure**, we can add **symbols** for the two fields U and \circ :

```
\symdef{univ}[name=universe,type=\collection]{U}
\symdef{op}[name=operation,args=a,assoc=bin,
  type=\funspace{\univ,\univ}\univ
]{\argsep{\#1}{\mathbin{\maincomp{\circ}}}}
```

Once we close the **environment** (with `\end{mathstructure}`), the **symbols** are “gone”. However, we now have a new **symbol** “**magma**” with **semantic macro** `\magma`. Its usage is somewhat more complicated than “normal” **semantic macros**, but one thing we *can* do with it now is `$\magma!`, which will produce $\langle U, \circ \rangle$.

Notably however, the `\magma` **macro** is already available *within* the **mathstructure environment** as well.

This allows us to provide an **sdefinition** using the **semantic macros** declared in the **structure**:

Example 11

Input:

```
File [sTeX/MathTutorial]algebra/Magma.en.tex
7 \begin{mathstructure}{magma}
8   \symdef{univ}[name=universe,type=\collection]{U}
9   \symdef{op}[name=operation,args=a,assoc=bin,
10     type=\funspace{\univ,\univ}\univ
11   {\argsep{\#1}{\mathbin{\maincomp{\circ}}}}]
12 
13 \begin{sdefinition}[for={magma,univ,op}]
14   A \definename{magma} is a \sr{mathstruct}{structure} $\magma$,
15   where $\univ$ is a \sn{collection} and $\op$ is
16   a binary operation $\funspace{\univ,\univ}\univ$.
17 \end{sdefinition}
18 \end{mathstructure}
```

Output:

Definition 3.3.2. A **magma** is a **structure** $\langle U, \circ \rangle$, where U is a **collection** and \circ a binary operation $U \times U \rightarrow U$.

Instantiating Structures

More importantly however, we can now declare a `variable magma`, using the optional `return=` argument. For example, we can now do

```
\vardef{vM}[name=M,return=\magma]{M}
```

and we get the semantic macro `\vM` with which we can do the following:

Syntax	Result
<code>\$\vM\$</code>	M
<code>\$\vM{}\$</code>	$\langle U_M, \circ_M \rangle$
<code>\$\vM{univ}\$</code>	U_M
<code>\$\vM{op}!\$</code>	\circ_M
<code>\$\vM{op}{a,b,c}\$</code>	$a \circ_M b \circ_M c$

In other words: Given a `symbol` or `variable` with `semantic macro` `\foo` and `return=\struct`, then `\foo{<fn>}` behaves like the `semantic macro` `\fn` *within* the `mathstructure environment` for `struct` – but instantiated for the specific instance `foo`.

By default, `STEX` attaches the `symbol`'s (or `variable`'s) `operator notation` as a subscript suffix to the notation component marked with `\maincomp` – e.g., since the “`\circ`” in the `notation` for `op` is marked with `\maincomp`, doing `$\vM{op}{a,b}$` ultimately outputs a `\circ_{\vM{op}} a b`. Hence, we get $a \circ_M b$.

We can change the way the `\maincomp` notation component is modified, by using the optional argument `copm=` in the `semantic macro` for the `mathematical structure`. For example, to not change it at all, we can do:

```
\vardef{vM}[name=M,return={\magma[comp=##1]}]{M}
```

...or to suffix it with a `,` we can do

```
\vardef{vMp}[name=Mp,return={\magma[comp=##1']}]{M'}
```

This allows us to do things like:

```
Let $\vM! := \vM{}$ and $\vMp := \vMp{}$ \sns{magma}. Then...
```

yielding

Let $M := \langle U, \circ \rangle$ and $M' := \langle U', \circ' \rangle$ magmas. Then...

We can also *assign* fields to (arbitrary) expressions, by doing `name=<tex>` in square brackets. For example we can do the following:

```
\vardef{vA}[type=\collection]{A}
\vardef{vM}[name=M,return={\magma[comp=##1][univ=\vA]}]{M}
\vardef{vMp}[name=Mp,return={\magma[comp={##1'}][univ=\vA]}]{M'}
```



```
Let $\vM! := \vM{}$ and $\vMp := \vMp{}$ \sns{magma} on $\vA$....
```

Let $M := \langle A, \circ \rangle$ and $M' := \langle A, \circ' \rangle$ magmas.

Of course, we can also use `return=` with `variable` sequences – for example:

```
\varseq{vMs}[name=M,return={\magma[comp={##1}_{##1}],op=(M_i)_{1^n}}
{1,\dots,n}{\maincomp{M}_{##1}}
Let $\vMs! := \vMs{i}_{1^n}$ a sequence of \sns{magma}...
```

Let $(M_i)_1^n := \langle U_i, \circ_i \rangle_1^n$ a sequence of *magma*s...

Note that in the above, it seems that using #1 in the `return` argument is allowed. Indeed, it is - the `return` statement takes the same arguments as the `semantic` macro itself does and is appropriately instantiated. Since the first (and only) argument to the sequence `\vMs` is the index, when doing `\vMs{i}...` the #1 in the `return`-statement will be replaced by *i*.

Also, note that if we want to produce M_i – i.e. the *magma* at index *i* in the sequence, we can do `\vMs{i}!`.



Think of the ! as a “stop sign” - if the expression up to the ! has an associated presentation, the ! tells `STEX` to “stop eating arguments” and present whatever it has until now.

3.3.2 Extending Structures and Axioms

It is extremely common to “build up” `structures` in a hierarchical manner by adding new fields or axioms: A *semigroup* is an associative magma. A *band* is an idempotent semigroup. A *monoid* is a semigroup with a unit. A *partial order* is an antisymmetric preorder.

We alluded to the fact earlier, that the `mathstructure` environment behaves like an `smodule` – that is literally true: Every `mathstructure` *foo* in a `module` *FooMod* is in fact also a `module` `?FooMod/foo-module`. We can therefore easily extend `structures` using `\importmodule{...?FooMod/foo-module}` – but extending `structures` is so common, and using `\importmodule` tiring, that there is a shortcut: the `extstructure` environment. It takes as second argument a comma-separated list of `structure` names. That allows us to easily define `semigroups`:

Example 12

Input:

```
File [sTeX/MathTutorial]algebra/Semigroup.en.tex
8 \begin{extstructure}{semigroup}{magma}
9  \begin{sdefinition}
10   A \definename{semigroup} is a \sn{magma} $\semigroup$,
11   where \inlineass[name=associative axiom]{
12     \conclusion{\isassociative{$\op$}}.
13   }
14  \end{sdefinition}
15 \end{extstructure}
```

Output:

Definition 3.3.3. A **semigroup** is a *magma* $\langle U, \circ \rangle$, where \circ is *associative*.

Note our usage of `\inlineass` to generate a new `symbol` for the *associative axiom*. If we look at the `OMDoc` tab in the `HTML` preview window, we can see the output in Figure 27.

So `MMT` has decided that our statement is an *axiom*.

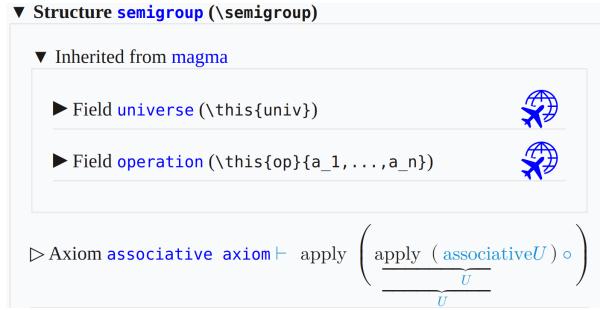


Figure 27: Axioms in OMDoc

Conservative Extensions

For **structures**, there is a *critical* distinction between *defined* and *undefined symbols*; and analogously between *theorems* and *axioms*.

Remember that **structures** are more like *templates* that are *instantiated* by particular objects. An *undefined* field in a **structure**, in that sense, is like an *obligation*: If something is supposed to be a **semigroup**, it *has to* have a **universe**, an **operation** and the **operation** needs to satisfy the **associative axiom**.

Defined fields on the other hand have a *definiens* on the basis of the remaining fields – they don't need to be explicitly provided for something to instantiate the **structure**; if all the *undefined* fields are provided, the *defined* ones we get “*for free*”.

The same holds for *theorems*: If a statement is *provable* from the axioms, then we don't need to explicitly prove it to hold for some particular instance – we have a proof already, provided the axioms hold.

The relation between axioms and theorems is not just analogous to that between undefined and defined **symbols**: It is the very same. Remember the **judgments as types** paradigm?

STEX For a **proposition** P , an assertion in **STEX** induces a **symbol** of type $\vdash P$. Without a proof, this **symbol** is *undefined* – and hence an *axiom*. A *proof* for P is a specific term of type $\vdash P$ – i.e. a potential *definiens*. To prove an assertion turns it into a *theorem*, which is to say that the **symbol** can be *defined*.

One consequence of this is: Extending a **structure** only by *defined* fields does not actually (conceptually) introduce a *new structure* – every instance of the old one *should* also be an instance of the new one. The new fields are basically just “syntactic sugar”.

There is a name for extending a **structure** only by defined fields (or theorems): A *conservative extension*.

STEX provides the **extstructure*** environment for that purpose. Unlike **extstructure**, it does *not* take a name (technically, **STEX** generates one internally). Instead, conceptually **extstructure*** modifies the extended **structure** directly, rather than generating a new **structure**. The caveat however is, that every **symbol** introduced in an **extstructure*** **must** be defined.

Consider the following conservative extension:

Example 13

Input:

```
File [sTeX/MathTutorial]algebra/MagmaSquare.en.tex
7 \begin{extstructure}{magma}
8   \begin{sdefinition}[macro=sq,args=1]
9     \notation{sq}[op=\cdot^2]{\#1^\text{comp} 2}
10    \vardef{va}[name=a,type=\univ,bind]{a}
11    Let $ \inset{va}{\univ} $. We define
12    $\defnotation{\sq{va}} := \definiens{\op{va,va}}$.
13  \end{sdefinition}
14 \end{extstructure}
```

Output:

```
Definition 3.3.4. Let  $a \in U$ . We define  $a^2 := a \circ a$ .
```

Via `\definiens`, the new symbol `sq` is now *defined* (note the `macro=` argument, that generates a `semantic macro` as well). Whenever we import the containing `module`, we now have an additional field `sq` in (any extension of) `magma` – e.g., as `semigroup` extends `magma`, the following is now valid:

```
\usemodule[sTeX/MathTutorial]{algebra?MagmaSquare}
\vardef{vsg}[name=S,return=\semigroup]{S}
$\vsg{sq}{a}$
```

...producing a^2 .

3.3.3 Nesting Structures and `\this`

A perhaps not too surprising, but a notable aspect of `structures` is that fields themselves can be instances. This is important for example for implementing *vector spaces*, but can also be used to bundle things that are not normally thought of as `structures`, such as objects with certain defining properties.

Take as an example, the notion of a (`magma`) homomorphism:

```
Definition 3.3.5. Let  $M_1 = \langle U_1, \circ_1 \rangle$  and  $M_2 = \langle U_2, \circ_2 \rangle$  magmas. A magma homomorphism is a function  $F : U_1 \rightarrow U_2$  such that  $F(a \circ_1 b) = F(a) \circ_2 F(b)$  for all  $a, b \in U_1$ .
```

So a `homomorphism` is a `function` with certain properties. And `structures` can be used to “bundle” the `function` itself with both the `magmas` on whose universes the `function` operates, as well as the *axiom* that *makes* it a `homomorphism`. After all, considered as a mere `function`, $F : U_1 \rightarrow U_2$ contains no information about the operation with respect to which it is homomorphic.

The first thing to note is that we can provide `mathstructure` with an optional argument for a `name` distinct from the name of its `semantic macro`. We then add two fields that `return` `magmas`. So far, so unexciting:

```
\begin{mathstructure}{magmahom}[magma homomorphism]
\symdef{dom}[name=domain,return={\magma[comp={##1}_1]}]{M_1}
\symdef{cod}[name=codomain,return={\magma[comp={##1}_2]}]{M_2}
```

For the `function` itself, we know how to give it a meaningful `type`, already:

```
\symdef{f}[type=\funspace{\dom{univ}}{\cod{univ}},args=1]{??}
```

...but what should its `notation` be? Ideally we would want it to just be the `notation` of whatever particular instance it is – in informal mathematics, we rarely distinguish notationally between a `homomorphism` and its underlying `function` (to the point where it's not clear, whether *informally* the distinction is even meaningful). Similarly, we rarely distinguish e.g. between a `magma` (or semigroup, monoid, group, ring, vector space,...) and its underlying universe.

This is where `\this` comes into play (pun intended). Within an `mathstructure` or `exstructure`, or in the context of a particular instance of one, `\this` represents “the” instance.

We can set it in the context of `mathstructure` as a further optional argument; e.g.

```
\begin{mathstructure}{magmahom}[magma homomorphism,this=F]
```

and then use `\this` in the `notation` for the `function`. We can further provide the `homomorphism condition` as an axiom using `\inlineass`:

Example 14

Input:

```
File [sTeX/MathTutorial]algebra/Homomorphism.en.tex
9 \begin{mathstructure}{magmahom}[magma homomorphism,this=F]
10   \symdef{dom}[name=domain,return={\magma[comp={##1}_1]}]{M_1}
11   \symdef{cod}[name=codomain,return={\magma[comp={##1}_2]}]{M_2}
12   \symdef{f}[op=\this,args=1,
13     type=\funspace{\dom{univ}}{\cod{univ}}
14   ]{\this \dobrackets{#1}}
15 
16 \begin{sdefinition}[for={magmahom,dom,cod,f}]
17   \vardef{va}[name=a,type=\dom{univ}]{a}
18   \vardef{vb}[name=b,type=\dom{univ}]{b}
19   Let $\dom!=\dom{}$ and $\cod!=\cod{}$ \sns{magma}.
20   A \definename{magmahom} is a function
21   $f:\dom{}\rightarrow\cod{}$ such that
22   \inlineass[name=homomorphism condition]{\conclusion{\forall{a \in \dom{}}
23     $arg[2]\leq{f(\dom{} op va,vb)} \cod{op}(f(va),f(vb))
24     } \forall{b \in \cod{}} $arg[1]\inset{va,vb}{\dom{univ}}}.
25   })
26 \end{sdefinition}
27 \end{mathstructure}
```

Output:

Definition 3.3.6. Let $M_1 = \langle U_1, \circ_1 \rangle$ and $M_2 = \langle U_2, \circ_2 \rangle$ magmas. A **magma homomorphism** is a function $F : U_1 \rightarrow U_2$ such that $F(a \circ_1 b) = F(a) \circ_2 F(b)$ for all $a, b \in U_1$.

Now if we instantiate our `magma homomorphism`:

```
\vardef{vh}[name=H,return={\magmahom[this=H]}]{H}
```

Here is a list of what we can do now:

Syntax	Result
$\$\\vh!$$	H
$\$\\vh{}$$	$\langle M_1, M_2, H \rangle$
$\$\\vh{f}!$$	H
$\$\\vh{f}{a}$$	$H(a)$
$\$\\vh{dom}!$$	M_1
$\$\\vh{cod}{}$$	$\langle U_2, o_2 \rangle$
$\$\\vh{cod}{univ}$$	U_2
$\$\\vh{dom}{op}!$$	o_1
$\$\\vh{cod}{op}{a,b,c}$$	$a \circ_2 b \circ_2 c$

Note how – as one would expect – we can treat $\backslash vh\{dom\}$ and $\backslash vh\{cod\}$ like any other instance of `magma`.

Note that some of the outputs in the above table are probably not quite what we want. Determining the precise typesetting of an expression involving *nested paths* of fields is difficult, to say the least (e.g., what exactly should `\this` refer to in a deeply nested sequence of fields?).

Using instances within `structures` is still very useful; at the very least when defining `structures`. When subsequently *using structures*, however, accessing fields of fields (of fields (of ...)) of an instance should be avoided.

Luckily, there is rarely a need for doing so – in practice, those fields we might want to access in such a way, we usually also want to provide specific `notations` and talk about independently of the “containing” instance, such that introducing a new `variable` (or `symbol`), and assigning the corresponding field to that `variable`, makes considerably more sense. And subsequently using the `variable` is easier than concatenating `{...}`, too.

3.4 Complex Inheritance and Theory Morphisms

We are starting to approach seriously experimental territory.

While the theory behind all the following is relatively well understood, and their implementation in `MMT` is mature, the same can not be said out the implementation in `sTeX`.

There are still kinks to be ironed out, but feel free to experiment.

We now have all the tools available to progress towards something more interesting. Here is a list of documents with respective `modules` and `symbols` we will build on in the following:

[`sTeX/MathTutorial`] `props/Idempotent.en.tex`

Definition 3.4.1. Let $e \in A$ and $\circ : A \times A \rightarrow A$. e is called **idempotent** with respect to \circ , if $e \circ e = e$.

Definition 3.4.2. The operation $\circ : A \times A \rightarrow A$ is called **idempotent**, if every element $a \in A$ is **idempotent** with respect to \circ .

[sTeX/MathTutorial]props/Distributive.en.tex

Definition 3.4.3. Let $\odot : B \times A \rightarrow A$ and $\oplus : A \times A \rightarrow A$. We say \odot **distributes over** \oplus , if $b \odot (a_1 \oplus a_2) = (b \odot a_1) \oplus (b \odot a_2)$ for all $a_1, a_2 \in A$ and $b \in B$.

[sTeX/MathTutorial]props/Absorption.en.tex

Definition 3.4.4. Let $\odot : A \times B \rightarrow A$ and $\oplus : A \times B \rightarrow B$. We say \odot **absorbs** \oplus , if $a_1 \odot (a_1 \oplus b) = a_1$ for all $a_1 \in A$ and $b \in B$.

[sTeX/MathTutorial]algebra/Band.en.tex

Definition 3.4.5. A **band** is an **idempotent semigroup**.

[sTeX/MathTutorial]algebra/Semilattice.en.tex

Definition 3.4.6. A **semilattice** is a **commutative band**.

[sTeX/MathTutorial]props/Reflexive.en.tex

Definition 3.4.7. A binary relation R on A is called **reflexive**, if $R(a, a)$ for all $a \in A$.

[sTeX/MathTutorial]props/Symmetric.en.tex

Definition 3.4.8. A binary relation R on A is called **symmetric**, if $R(a, b)$ implies $R(b, a)$ for all $a, b \in A$.

[sTeX/MathTutorial]props/Transitive.en.tex

Definition 3.4.9. A binary relation R on A is called **transitive**, if $R(a, b)$ and $R(b, c)$ implies $R(a, c)$ for all $a, b, c \in A$.

[sTeX/MathTutorial]props/Antisymmetric.en.tex

Definition 3.4.10. A binary relation R on A is called **antisymmetric**, if $R(a, b)$ and $R(b, a)$ implies $a = b$ for all $a, b \in A$.

[sTeX/MathTutorial]orders/Graph.en.tex

Definition 3.4.11. A **directed graph** is a **structure** $\langle U, R \rangle$, where U is a **collection** and R a binary relation on U .

Definition 3.4.12. An **(undirected) graph** is a directed graph $\langle U, R \rangle$, where R is **symmetric**.

[sTeX/MathTutorial]orders/Preorder.en.tex

Definition 3.4.13. A structure $\langle U, \leq \rangle$ is called a **preorder** (or **quasiorder**, or **preordered set**; in short **proset**), if \leq is **reflexive** and **transitive**.

[sTeX/MathTutorial]orders/Poset.en.tex

Definition 3.4.14. A preorder $\langle U, \leq \rangle$ is called a **partial order** (or **poset**), if \leq is **antisymmetric**.

[sTeX/MathTutorial]orders/InfSup.en.tex

Definition 3.4.15. Let $\langle U, \leq \rangle$ a poset. An element $a \in U$ is called an **infimum** or **greatest lower bound** of x_1 and x_2 , if $a \leq x_1$, $a \leq x_2$, and for any x with $x \leq x_1$ and $x \leq x_2$, we have $x \leq a$.

Definition 3.4.16. Let $\langle U, \leq \rangle$ a poset. An element $a \in U$ is called a **supremum** or **least upper bound** of x_1 and x_2 , if $x_1 \leq a$, $x_2 \leq a$, and for any x with $x_1 \leq x$ and $x_2 \leq x$, we have $a \leq x$.



Note that **infima** and **suprema** are more generally defined on *sets* of elements. Doing so in **STEX** is significantly more complicated *for now*, and will require some amount of research to make convenient – especially if we want to subsequently define *operators* on pairs of elements, as below. We therefore opt for the simpler version where it is defined as binary from the get go.

[sTeX/MathTutorial]orders/MeetJoinSemilattice.en.tex

Definition 3.4.17. A poset $\langle U, \leq \rangle$ is called a **meet semilattice** if for every two elements a, b the infimum $a \wedge b$ exists.

Definition 3.4.18. A poset $\langle U, \leq \rangle$ is called a **join semilattice** if for every two elements a, b the supremum $a \vee b$ exists.

Definition 3.4.19. An **(order) semilattice** is a meet and join semilattice.

Exercise

Try to implement all of the above yourself!

3.4.1 Glueing Structures Together

We now want to progress towards **lattices**, i.e. the following:

Definition 3.4.20. A lattice is a structure $\langle U, \wedge, \vee \rangle$ such that $\langle U, \wedge \rangle$ and $\langle U, \vee \rangle$ are semilattices, and \vee absorbs \wedge and vice versa; i.e. $a \vee (a \wedge b) = a$ and $a \wedge (a \vee b) = a$.

The operations \wedge and \vee are called **meet** and **join**, respectively.

So we make a new `module`, open an `extstructure environment` and... realize two problems:

1. We can't just extend `semilattice`: We need *two* copies of `semilattice` that share a universe, and importing `semilattice` twice is of course redundant.
2. We also want to *rename* the operations of the two `semilattices` to be subsequently called `join` and `meet`.

What we need is a way to *inherit* from `semilattice` while a) *modifying* the `symbols` therein, and b) not be `idempotent` – i.e. two imports from the same `structure` or `module` should not be identified. We can do that with the `\copymod` macro, which takes three arguments:

1. A *name* for the copy,
2. the `structure` or `module` to copy, and
3. a comma-separated list of renamings and redefinitions of the `symbol`. `<symbol>=<def>` redefines `<symbol>`, `<symbol>@<newname>` renames it, `<symbol>=<def>@<newname>` (or `<symbol>@<newname>=<def>`) does both.

In our case, we want two copies of `semilattice`, which we will call `meetsl` and `joinsl`. In the first copy, we only want to rename `op` to `meet`. In the second, we want to rename `op` to `join`, and *also* redefine the universe to be the one from `meetsl`:

```
\copymod{meetsl}{semilattice}{
    op @ meet
}
\copymod{joinsl}{semilattice}{
    univ = \univ,
    op @ join
}
```

You might have already noticed some problem with that – which of the two universes does `\univ` refer to now? (They are *defined* as equal, but `LATEX` does not know that!) Or which of the two `commutative axioms` does “`commutative axiom`” refer to now? Everything is ambiguous now!

Not really - if you have wondered why the `\copymod` takes a *name* as argument: The name is prefixed to every `symbol` name. Hence, the `universe` in `joinsl` is now called `joinsl/universe`, and the one in `meetsl` is called `meetsl/universe`. Furthermore, `\copymod` by default generates no `semantic macros` for any of the imported `symbols` – except for those renamed with `@`. In fact, what the `@` syntax actually does, is to generate a `semantic macro` by that name. If we want to change the *name* (that is shown when using `\symname` et al), we add that new name in square brackets. Hence, what we really want to do is:

```
\copymod{meetsl}{semilattice}{
    univ @ univ,
    op @ [meet]meet
}
\copymod{joinsl}{semilattice}{
    univ = \univ,
    op @ [join]join
}
```

This now gives us two copies of `semilattice`, generates semantic macros `\univ` for `meetsl/universe`, `\meet` for `meetsl/op` and `\join` for `joinsl/op`, and renames `meetsl/op` to `meet` and `joinsl/op` to `join`.

That allows us to then add the `absorption` axioms, an `sdefinition` for `lattice` and subsequently `$\lattice!` produces $\langle U, \wedge, \vee \rangle$, with all axioms inherited (see [sTeX/MathTutorial]algebra/Lattice.en.tex).

3.4.2 Realizations

A very common situation we find in connection with `mathematical structures` is that “every *this* is a *that*” (or the concrete case “*this* is a *that*”).

With what we did so far, we are in this situation regarding the algebraic definition of `semilattices` and the order-theoretic one (exemplary `meet semilattice`).

In MMT parlance, this corresponds to a `total (implicit) theory morphism` from “*that*” to “*this*”.

In `STEX` words, we want to inherit from “*that*” by assigning all the `symbols` in “*that*” to concrete terms. In our case:

```
[sTeX/MathTutorial]algebra/SemiLatticeOrder.en.tex
```

Definition 3.4.21. Let $\langle U, \circ \rangle$ a `semilattice`. We let $a \leq b$ iff $a \circ b = a$.

Satz 3.4.22. $\langle U, \leq \rangle$ is a `meet semilattice`.

Proof: We need to prove the following

1. **reflexivity** ($a \leq a$): We need to show $a \circ a = a$. Follows from the `idempotent axiom`.
2. **antisymmetry** ($a \leq b$ and $b \leq a$ implies $a = b$): Assume $a \circ b = a$ and $b \circ a = b = a \circ b$ (by the `commutative axiom`). Hence, $a = b$
3. **transitivity** (If $a \leq b$ and $b \leq c$, then $a \leq c$): Assume $a \circ b = a$ and $b \circ c = b$. Then $a \circ c = (a \circ b) \circ c = a \circ (b \circ c) = a \circ b = a$. Hence, $a \leq c$.
4. $a \circ b$ is the `infimum` of $\{a, b\}$: By definition (and the `commutative axiom`), $a \circ b \leq a$ and $a \circ b \leq b$. We need to show, that if $x \leq a$ and $x \leq b$, then $x \leq a \circ b$. Assume $x \circ a = x$ and $x \circ b = x$. Then $x \circ (a \circ b) = (x \circ a) \circ b = x \circ b = x$. Hence $x \leq a \circ b$

So to be precise, we want to provide `definientia` for all undefined `symbols` in `meet semilattice` (i.e. the `relation` and `meet`) and `proofs` for all `axioms` (`reflexive axiom`, `anti-symmetric axiom`, `transitive axiom`, and `infimum axiom`), and by so obtain the fact that every `semilattice` is a `meet semilattice`.

For that purpose, we have the `\realize` macro. It behaves like `\copymod`, but does not take a name, and additionally requires that all undefined fields get assigned. So we could do the following:

Example 15

Input:

```

File [sTeX/MathTutorial]algebra/SemiLatticeOrder1.en.tex

8 \begin{extstructure*}{semilattice}
9   \realize{meets1}[
10     univ = \univ,
11     meet = \op!,
12     rel @ [order]order = \map{a,b}{\eq{\op{a,b},a}},
13     reflexive axiom = trivial,
14     transitive axiom = trivial,
15     antisymmetric axiom = trivial,
16     infimum axiom = trivial
17   }
18 \end{extstructure*}
19
20 \vardef{mysl}[return=\semilattice]{S}
21 $mysl{order}{a,b} \qquad \mysl{}[\univ,op,order]$
```

Output:

$$a \leq_S b \quad \langle U_S, \circ_S, \leq_S \rangle$$

As we can see, we can now access the field `order`, which is renamed from `relation` in `meet semilattice` and also has the desired definiens in `MMT`. But of course this approach is very “declarative”: We do all the assigning in one `macro`, rather than narratively as what they *should* be: definitions and proofs.

If we want to achieve the more narrative version at the beginning of the subsection, we can use the `realization environment` instead. It behaves like the `\realize` macro, but allows us to do the assignments and renamings individually somewhere in the body of the `environment`, interleaved with arbitrary text. Additionally, within the `environment`, all `sTeX` features that introduce *definientia* (like the `\definiens` macro) induce assignments instead.

To declaratively rename or assign fields, we can then use the `\assign` and `\renamedecl` macros instead. That allows us to do the following instead:

```

\begin{realization}{meets1}
\assign{univ}{\univ}
\assign{meet}{\op!}
\renamedecl{rel}[order]{order}
...
```

...and then use text to do the remaining assignments. For example, we can use the `sdefinition environment` to assign `rel` to the desired definiens:

```

\usestructure{meets1}
\begin{sdefinition}[for=order]
\varbind{va,vb}
Let $ \semilattice! [\univ,op] $ a \sn{semilattice}.
We let $ \rel{\va,\vb} $.
iff $ \definiens{\eq{\op{\va,\vb},\va}} $.
\end{sdefinition}
```

And now `sTeX` will use the `\definiens` to assign $a, b \mapsto a \circ b = a$ to the relation of `meet semilattice`.

Analogously, we can use the `sproof` and `subproof` environments to produce “definientia” (i.e. proofs) for the axioms (see [sTeX/MathTutorial]algebra/SemiLatticeOrder.en.tex)

Chapter 4

Extensions for Education

The last two chapters have shown generic markup and semantization facilities in $\text{\texttt{STEX}}$. As said before, investments in semantic markup pay off, iff the impact of a document is high, e.g. if there are many more readers than authors or if the semantic services afforded by the semantic markup can help reduce the help readers need to understand the material.

Educational documents constitute one category of high-impact documents which are supported by the $\text{\texttt{STEX}}$ ecosystem, we will cover them here. In fact, educational documents have been one of the initial document categories $\text{\texttt{STEX}}$ has been developed for. The idea is that if we can mark up the meaning and didactic role of learning objects, we can base learning support services on that and embed them into the documents.

Another reason educational documents are particularly interesting is that in a sense all academic communication is educational, as all documents try to “teach” the reader new concepts and results.

Concretely, we cover a document class for combining slides and course notes (??) and functionality for marking up problems and excercises (??) and for marking up homework assignments and exams (??).

4.1 Slides and Course Notes

TODO⁶

Contents

4.2 Problems and Exercises

4.2.1 Background

Problems/exercises are text fragments that contain a task assigned to the learner – e.g. computing the value of a specified quantity, simplifying an expression, modeling a described situation in a mathematical structure, or judging the veracity of a given statement. Problems/exercises are used in very different contexts, in

⁶TODO: `notesslides.sty`

- *homework assignments and formal exams*, where they are graded and points are awarded for course credit,
- *self-study materials* that allow learners explore applications of some concepts and/or practice,
- *automated tutoring systems* to determine learner competencies to trigger computer supported learning services.

In all of these contexts, diagnosis the correctness or suitability of an answer plays a great role, e.g. for grading, the generation of feedback, or the suggestion of remedial actions that may “heal” any diagnosed competency gaps or misconceptions. To support that we might want to specify “answer classes” that classify answers received for automating/triggering feedback and grading.

There are various types of problems/exercises in practical use. They are mainly distinguished by how they support diagnosis of student answers: there are

- open problems, where expected answers are free-form, and classification into answer classes is usually a manual process; see ??.
- single/multiple choice questions where the answer classes and thus feedback/grading correspond to the selection pattern of items; see ??.
- fill-in-the-blanks questions where we may want to specify how the answer string is interpreted; see ??.

Additionally, problems and exercises often come with text fragments that serve auxiliary functions: hints, notes, and grading specifications. Furthermore, we can specify how long solving a given problem is estimated to take and how many points will be awarded for a perfect solution – e.g. in an exam. See ?? for details.

The **problem** package gives semantic markup support for all these aspects of problems/exercises with a focus on problem libraries that collect carefully formulated and tested problems/exercises, so that they can be re-used in many contexts. It also tries to support all possible stakeholders in dealing with problems/exercises:

- *learners*, who try to solve problems and may profit from feedback
- *instructors*, who pose problems in homeworks, quiz, and exams,
- *graders*, who try to assess learner’s competencies from the answers given
- *authors* of learning objects and (automated) *course generators*, who may use problems to diagnose learner’s competencies (e.g. as prerequisites).

4.2.2 The Package, Options, and Configuration

The **problem** package provides functionality for marking up problems and exercises as semantic sources from which the presentations for the various contexts can be generated. E.g. documents without solutions for paper or online exams, and the corresponding exams with master solutions for exam reviews. Similarly with/without hints, or points. Their visibility is specified in the options of the **problem** package, which can be used in any L^AT_EX class. The following is a typical preamble for a problem file:

```
\documentclass[lang=de]{article}
\usepackage[solutions,hints,pts,min]{problem}
```

Here we have specified the options `solutions` (solutions should be shown), `hints` (hints should be given), `pts` (display the points awarded for solving the problem?), `min` (display the estimated minutes for problem solving). Leaving out the options would make the corresponding functionality invisible.

The `problem` package generates content and labels according to the language specified in the `lang` key of the main document¹, these can be localized by in the files `ldf/problem-<lang>.ldf`⁷.

The `problem` package also supplies the `test` option, which specifies that the content is intended for use in an assessment situation, where certain additional content (e.g. exam legalse) should be shown while other content (e.g. solutions) should not be.

Note that documents (or document fragments) with problems are often formatted in different versions that can be configured by these options. Therefore the following variant of the preamble prefix above can be very useful:

```
\documentclass[lang=de]{article}
\providecommand\probopts{,solutions,min}
\usepackage[hints,pts\probopts]{problem}
```

We add a level of indirection via the `\probopts` macro (`\providecommand` sets it unless it already exists). If the current file is `foo.tex`, then we can make a L^AT_EX file `foo-test.tex` that formats in test mode as

```
\newcommand\probopts{,test}\input{foo}
```

Alternatively, we do not need a file at all: we can just (e.g. in a Makefile or a PDF generation script) issue the following command in the terminal:

```
pdflatex "\def\probopts{,test}\input{foo}"
```

4.2.3 (Open) Problems

The main environment provided by the `problem` package is (surprise surprise) the `sproblem` environment. It is used to mark up problems and exercises. The following example shows the main functionality:

Example 16

Input:

¹EDNOTE: MK: document the attribute somewhere

⁷The current crop of language definition files is quite limited. Please feel free to contribute to the localization effort

```

File [sTeX/Documentation]tutorial/ext/simple-problem.en.tex

1 \begin{document}
2 \begin{sproblem}[id=prob.elefants,pts=10,min=2,title=Fitting Elefants]
3 How many Elefants can you fit into a Volkswagen beetle?
4 \begin{hint}
5   Think positively, this is simple!
6 \end{hint}
7 \begin{exnote}
8   Justify your answer
9 \end{exnote}
10 \begin{gnote}[title=deduct 5pts if justification is missing]
11   \anscls[feedback=you forget that elephants could be small]{none}
12   \anscls[feedback=you forget the back seats]{2}
13 \end{gnote}
14 \begin{solution}
15   Four, two in the front seats, and two in the back.

```

Output:

Exercise (Fitting Elefants)

How many Elefants can you fit into a Volkswagen beetle?

Lösung: Four, two in the front seats, and two in the back.

The `sproblem` environment takes an optional key/value argument with the keys `id` as an identifier that can be reference later, `pts` for the points to be gained from this exercise in homework or quiz situations, `min` for the estimated minutes needed to solve the problem, and finally `title` for an informative title of the problem.

The `sproblem` can contain several `solution` environments, which take the well-known `id`, `style`, and `title` attributes, and also the `testspace` and `answerclass`. The former

The additional functionality is specified in the `hint` `exnote` (notes in exercises), `solution`, and `gnote` (grading notes) environments. Here, the first three are shown whereas the grading notes are hidden, since the corresponding option was not given in the `\usepackage[...]{problem}`. All of these environments can occur any number of times in the `sproblem` environment. The `solution` environment takes an optional argument that is interpreted as the identifier.

4.2.4 Solutions and Answer Classes

Most problem libraries also contain (master) solutions, which show the intended way of solving a problem. The `solution` environment allows to specify solutions. It is only formatted if the `solutions` options is specified for the `problem` package.

We observe that many problems have more than one possible solution, so the `problem` package allows multiple `solution` environments; they can be distinguished by an `id` key in the optional first argument.

We also observe that in problem libraries we may want to keep track of classes of answers that have been observed e.g. in previous assessment situations, e.g. to provide

feedback. In this setting, the `solution` environment can be used to encode (typical representative of) **answer classes** (see also ?? below).

`solution` The `solution` environment takes an optional argument that allows the keys `id`, `title` and `style` keys that we have already seen above, further more

1. `testspace=`, which is equivalent to a `\testpsace{}` (see ??).
2. `answerclass=`, which gives the name of an answer class, this solution corresponds to.²

EdN:2

4.2.5 Grading Support

When problems are graded by multiple persons (e.g. for large university exams), then consistency of grading is a major issue. Both within a cohort and between cohorts e.g. in successive instances of a course.

Therefore keeping records of how to grade a problem (which problem is worth how many “points” and where to deduct how many points in which situations) and making them accessible to graders in a “master solution” – a version of an exam formatted to have the known solutions/answer classes (see above) and the grading specifications is a good practice. And it makes sense to include grading information into a problem library.

The simplest and arguably most effective measure is to specify the points that can be reached for a problem in the problem itself via the ‘`pts=`’ key (but see ??)

The `problem` package also supplies the `gnotes` environment, which is only formatted when the `gnotes=` key is given. This environment takes an optional argument that allows the `id`, `style`, and `title` keys. The latter is used to give a short version of the grading specification. More structured grading support can be expressed in `\anscls` markup which we discuss next.

We have already encountered the notion of answer classes – classes of answers that are supposed equivalent in terms of learner’s competencies inscribed in them – above. These are the natural carriers of both grading feedback and points.

In the `gnote` environment we use the `\anscls` macro for specifying such information using meta-level descriptions rather than typical representatives. It takes an optional keyword argument for the grading/feedback information and a required one for the answer class description.

We distinguish two usages of `\anscls[...]{<desc>}`: in

- *answer classes* `<desc>` is a specification of a class of answers and the `pts=` key the points to be awarded to that.
- *answer traits* `<desc>` specifies the deviation from an assumed correct solution, and the value of the `pts=` key specifies changes to the points specified in the problem itself.

Consider for instance the following situation:

```
\begin{sproblem}[pts=3]
  Give an example for a foo and explain it in one sentence.
\begin{gnote}[title=1 pt for the example and 2 for the explanation]
  \anscls[pts=0,feedback=I did not understand this]{complete gibberish}
  \anscls[pts=-2,feedback=You forgot the explanation]{no explanation}
\end{gnote}
\end{sproblem}
```

²EDNOTE: This mechanism needs to be further worked our or deprecated; if we keep it we probably need to add `pts` and `feedback` attributes for parity with `\anscls`.

Here we have a problem that is worth three points, and the grading note explains how to treat deviations from a correct solutions – which is not given, since there may be hundreds of examples for foos. The `\anscls` specify this out for grading support systems⁸. The first is an answer class description for the class of answers that cannot be made sense of by the graders and specifies a default grade and feedback. The second specifies the trait of a missing explanation and specifies a deduction commensurate with the grading note title and a feedback to the learner. Note that a grading support system can distinguish answer classes from traits by the form of the value of points: absolute values vs. differences like +5 or -3.

4.2.6 Structured Problems

Problems can be structured into subproblems via the `subproblem` environment. It takes the same arguments and allows the same content model as as `sproblem`.

Example 17

Input:

```
File [sTeX/Documentation]tutorial/ext/structured-problem.en.tex
1 \begin{document}
2 \begin{sproblem}[id=prob.elefants-mod,title=Fitting Elephants Modularly]
3 Consider the problem of fitting elephants into a Volkswagen beetle.
4 \begin{subproblem}[pts=1,min=2]
5 Estimate the number of elephants on the front seats.
6 \begin{solution}
7 Two on each side one.
8 \end{solution}
9 \end{subproblem}
10 \begin{subproblem}[pts=1,min=2]
11 Estimate the number of elephants on the back seats.
12 \begin{solution}
13 Three little elephants.
14 \end{solution}
15 \end{subproblem}
```

Output:

⁸like e.g. <https://gitos.rrze.fau.de/yn06uhoc/vollkorn-prototype>

Exercise (Fitting Elephants Modularly)

Consider the problem of fitting elephants into a Volkswagen beetle.

1. Estimate the number of elephants on the front seats.

Lösung: Two on each side one.

2. Estimate the number of elephants on the back seats.

Lösung: Three little elephants.

3. What is the total number?

Lösung: A family of five; we do not want elephants in the trunk.

By default, subproblems are numbered subordinate to the “mother problem”. The `problem` package does not make any assumptions about whether subproblems can be used independently from their sister problems, or on order requirements.

Note that neither `sproblem` nor `subproblem` can be nested, if you want to have further structure, you need to resort to regular L^AT_EX functionality. Note that you can add `solution` and `gnote` environments wherever you want, so that this need not force you to forego structure.

The `subproblem` environment is however very useful for modularizing problems: `subproblem` environments can appear outside of `sproblem` and in particular at the top-level of a file. This allows them to be shared between `sproblem` via `\inputref`.

4.2.7 Single/Multiple Choice Blocks

Multiple choice blocks can be formatted using the `mcb` environment, in which single choices are marked up with `\mcc` macro.

`\mcc` `\mcc[<keyvals>]{<text>}` takes an optional key/value argument `<keyvals>` for choice metadata and a required argument `<text>` for the proposed answer text. The following keys are supported

- `T` for correct answers, `F` (the default value) for false ones,
- `Ttext` the verdict for correct answers, `Ftext` for false ones, and
- `feedback` for a short feedback text given to the student.

What we see when this is formatted to PDF depends on the context. In solutions mode (we start the solutions in the code fragment below) we get

Example 18

Input:

```
1 \startsolution
2 \begin{sproblem}[title=Functions,name=functions1,autogradable]
3 What is the keyword to introduce a function definition in python?
4 \begin{mcb}
5   \mcc[T]{def} \mcc[F,feedback=that is for C and C++]{function}
6   \mcc[F,feedback=that is for Standard ML]{fun}
7   \mcc[F,Ftext=Nooooooooo,feedback=that is for Java]{public static void}
8 \end{mcb}
9 \end{sproblem}
```

Output:

Exercise (Functions)

What is the keyword to introduce a function definition in python?

- def
- function^a
- fun^b
- public static void^c

^a

that is for C and C++

^b

that is for Standard ML

^cNooooooooo

that is for Java

The problem has been marked as `autogradable` for semantic processing.

In “exam mode” where disable solutions (here via `\stopsolutions`) we get the questions without solutions (that is what the students see during the exam/quiz).

Example 19

Input:

```
1 \stopsolutions
2 \begin{sproblem}[title=Functions,name=functions1,autogradable]
3 What is the keyword to introduce a function definition in python?
4 \begin{mcb}
5   \mcc[T]{def} \mcc[F,feedback=that is for C and C++]{function}
6   \mcc[F,feedback=that is for Standard ML]{fun}
7   \mcc[F,Ftext=Nooooooooo,feedback=that is for Java]{public static void}
8 \end{mcb}
9 \end{sproblem}
```

Output:

Exercise (Functions)

What is the keyword to introduce a function definition in python?

- def
- function
- fun
- public static void

What we have seen is the default rendering of the multiple choice block, which is as an itemize like environment with check boxes. There are alternatives to that which we can choose with the `style` key in the optional attribute of the `mcb` environment. Currently the only alternative is `inline` style:

Example 20

Input:

```
1 \startSolutions
2 \begin{sproblem}[title=Functions, name=functions1]
3   What is the keyword to introduce a function definition in python?
4
5   \begin{mcb}[style=inline]
6     \mcc[T]{def} \mcc[F, feedback=that is for C and C++]{function}
7     \mcc[F, feedback=that is for Standard ML]{fun}
8     \mcc[F, Ftext=Noooooooooooo, feedback=that is for Java]{public static void}
9   \end{mcb}
10 \end{sproblem}
```

Output:

Exercise (Functions)

What is the keyword to introduce a function definition in python?

- [def function^a fun^b public static void^c]

^a

that is for C and C++

^b

that is for Standard ML

^c

Noooooooooooo

that is for Java

This is the solutions mode, i.e. with solutions, otherwise, the footnotes will fully disappear.

Analogously, single choice blocks are marked up by the `scc` environment and the individual choices by the `\scc` macro. In PDF, there is little difference to the multiple choice case. In interactive formats like HTML, single choice blocks are typically realized via radio buttons to enforce single choice.³

³EDNOTE: MK: are there any differences between `mcc` and `scc` we need to discuss here?

4.2.8 Filling-In Concrete Solutions

The next simplest situation, where we can implement auto-grading is the case where we have fill-in-the-blanks

The `\fillinsol` macro takes a single argument, which contains a concrete solution (i.e. a number, a string, ...), which generates a fill-in-box in test mode:

Example 21

Input:

```
1 \stopsolutions
2 \begin{sproblem}[id=elefants.fillin,title=Fitting Elefants]
3 How many Elefants can you fit into a Volkswagen beetle? \fillinsol{4}
4 \end{sproblem}
```

Output:

Exercise (Fitting Elefants)

How many Elefants can you fit into a Volkswagen beetle?



and the actual solution in solutions mode:

Example 22

Input:

```
1 \startsolutions
2 \begin{sproblem}[id=elefants.fillin,title=Fitting Elefants]
3 How many Elefants can you fit into a Volkswagen beetle? \fillinsol{4}
4 \end{sproblem}
```

Output:

Exercise (Fitting Elefants)

How many Elefants can you fit into a Volkswagen beetle?

4

If we do not want to leak information about the solution by the size of the blank we can also give `\fillinsol` an optional argument with a size: `\fillinsol[3cm]{12}` makes a box three cm wide.

Obviously, the required argument of `\fillinsol` can be used for auto-grading. For concrete data like numbers, this is immediate, for more complex data like strings “soft comparisons” might be in order. Indeed the default matching of answers against the string provided in the required argument of the `\fillinsol` is up to whitespace normalization.

The `problem` package allows to specify different behavior via three additional keys whose arguments are all triples of the form

```
{<target>}{{<TF>}}{<feedback>}
```

where `<target>` is the target specification, `<TF>` is the verdict T or F on the correctness of an answer that is accepted by `<target>`, and `<feedback>` a feedback string. In the following (somewhat contrived) example, we see all three at work:

Example 23

Input:

```

1 \begin{sproblem}
2   In 2019, Erlangen is a city of
3   \fillinsol[testspace=3cm,
4   exact={111962}T{Wow, you know your numbers, according to
5     Wikipedia that is exactly correct},
6   numrange={0-1000}F{No,that would be a village},
7   numrange={1001-100000}F{No,that would be a town},
8   numrange={100000-120000}T{Yes, it is close to 109000},
9   numrange={1200001-}F{No, that is too large},
10  regex={-.[0-9]+}F{What do negative Inhabitants even mean?},
11  regex={.*}F{Please give a natural number}
12  {111962}
13  inhabitants.
14 \end{sproblem}
```

Output:

Exercise

In 2019, Erlangen is a city of 111962^a inhabitants.

a

type	case	verdict	feedback
exact	111962	T	Wow, you know your numbers, according to Wikipedia that is exactly correct
numrange	0-1000	F	No,that would be a village
numrange	1001-100000	F	No,that would be a town
numrange	100000-120000	T	Yes, it is close to 109000
numrange	1200001-	F	No, that is too large
regex	-.[0-9]+	F	What do negative Inhabitants even mean?
regex	*	F	Please give a natural number

- `exact=` gives the exact string (no whitespace normalization).
- `numrange=` specifies a number range, i.e. a comma-separated list of pairs $\langle low \rangle - \langle high \rangle$, where $\langle low \rangle$ and $\langle high \rangle$ are number representations or empty.
- `regex=` gives a POSIX regular expression that specifies all acceptable strings.

4.2.9 Including Problems

`\includeproblem` The `\includeproblem` macro can be used to include a problem from another file. It takes an optional KeyVal argument and a second argument which is a path to the file containing the problem (the macro assumes that there is only one problem in the include file). The keys `title`, `min`, and `pts` specify the problem title, the estimated minutes for solving the problem and the points to be gained, and their values (if given) overwrite the ones specified in the `problem` environment in the included file.

The sum of the points and estimated minutes (that we specified in the `pts` and `min` keys to the `problem` environment or the `\includeproblem` macro) to the log file and the screen after each run. This is useful in preparing exams, where we want to make sure that the students can indeed solve the problems in an allotted time period.

The `\min` and `\pts` macros allow to specify (i.e. to print to the margin) the distribution of time and reward to parts of a problem, if the `pts` and `pts` options are set. This allows to give students hints about the estimated time and the points to be awarded.

4.2.10 Testing and Spacing

The `problem` package is often used by the `hwexam` package, which is used to create homework assignments and exams. Both of these have a “test mode” (invoked by the package option `test`), where certain information –master solutions or feedback – is not shown in the presentation.

```
\testspace      \testspace takes an argument that expands to a dimension, and leaves vertical space accordingly. Specific instances exist: \testsmallspace, \testmediumspace, \testlargepage \testsmallspace give small (1cm), medium (2cm), and big (3cm) vertical space.
\testsmallspace \testnewpage makes a new page in test mode, and \testemptypage generates an empty page with the cautionary message that this page was intentionally left empty.
\testemptypage  TODO9
```

4.3 Homework Assignments and Exams

4.3.1 Introduction

The `hwexam` package and class supplies an infrastructure that allows to format nice-looking assignment sheets by simply including problems from problem files marked up with the `problem` package. It is designed to be compatible with `problems.sty`, and inherits some of the functionality.

4.3.2 Package Options

<code>solutions</code>	The <code>hwexam</code> package and class take the options <code>solutions</code> , <code>notes</code> , <code>hints</code> , <code>gnotes</code> , <code>pts</code> , <code>min</code> , and <code>boxed</code> that are just passed on to the <code>problems</code> package (cf. its documentation for a description of the intended behavior).
------------------------	---

<code>multiple</code>	Furthermore, the <code>hwexam</code> package takes the option <code>multiple</code> that allows to combine multiple assignment sheets into a compound document (the assignment sheets are treated as section, there is a table of contents, etc.).
-----------------------	--

<code>test</code>	Finally, there is the option <code>test</code> that modifies the behavior to facilitate formatting tests. Only in <code>test</code> mode, the macros <code>\testspace</code> , <code>\testnewpage</code> , and <code>\testemptypage</code> have an effect: they generate space for the students to solve the given problems. Thus they can be left in the L ^A T _E X source.
-------------------	---

⁹TODO: check what is still undescribed `problem.sty` and make examples for it.

4.3.3 Assignments

assignment (*env.*) This package supplies the **assignment** environment that groups problems into assignment sheets. It takes an optional KeyVal argument with the keys **number** (for the assignment number; if none is given, 1 is assumed as the default or — in multi-assignment documents **title** — the ordinal of the **assignment** environment), **title** (for the assignment title; this is **type** referenced in the title of the assignment sheet), **type** (for the assignment type; e.g. “quiz”, **given** or “homework”), **given** (for the date the assignment was given), and **due** (for the date **due** the assignment is due).

4.3.4 Including Assignments

\includeassignment The **\includeassignment** macro can be used to input an assignment from another file. It takes an optional KeyVal argument and a second argument which is a path to the file containing the problem (the macro assumes that there is only one **assignment** environment in the included file). The keys **number**, **title**, **type**, **given**, and **due** are just as for the **assignment** environment and (if given) overwrite the ones specified in the **assignment** environment in the included file.

4.3.5 Typesetting Exams

testheading (*env.*) The **\testheading** takes an optional keyword argument where the keys **duration** specifies a string that specifies the duration of the test, **min** specifies the equivalent in number **min** of minutes, and **reqpts** the points that are required for a perfect grade.

```
reqpts1 \title{320101 General Computer Science (Fall 2010)}
2 \begin{testheading}[duration=one hour,min=60,reqpts=27]
3 Good luck to all students!
4 \end{testheading}
```

Will result in

Name:

Matriculation Number:

320101 General Computer Science (Fall 2010)

2024-11-20

You have one hour (sharp) for the test;

Write the solutions to the sheet.

The estimated time for solving this exam is 23 minutes, leaving you 37 minutes for revising your exam.

You can reach 23 points if you solve all problems. You will only need 27 points for a perfect score, i.e. -4 points are bonus points.

Different problems test different skills and knowledge, so do not get stuck on one problem.

	To be used for grading, do not write here								
prob.	4.1	1.1	1.2	2.1	2.2	2.3	2.4	2.5	2.6
total	0	0	0	10	0	0	0	0	0
reached									

good luck

10

¹⁰MK: The first three “problems” come from the stex examples above, how do we get rid of this?

Part II

User Manual

*The dynamic [HTML](#) version of this part can be found at
<https://stexmmt.mathhub.info/>:
<https://stexmmt.mathhub.info/stex/fullhtml?archive=stex/Documentation&filepath=manual.xhtml>*

Chapter 5

Basics

5.1 Package and Class Options

- `debug=(prefixes)`: (see Developer Manual)
- `lang=(languages)`: If set, `STEX` will load the `babel` package with the provided languages. Supported languages (currently) are:

<code>ar</code>	arabic
<code>bg</code>	bulgarian
<code>de</code>	german (with option <code>ngerman</code>)
<code>en</code>	english
<code>fi</code>	finnish
<code>fr</code>	french
<code>ro</code>	romanian
<code>ru</code>	russian
<code>tr</code>	turkish (with option <code>shorthands=:</code> !)

- `mathhub=(path)`: Uses the provided file path as MathHub directory (see ??).
- `usesms/writesms`: If `writesms` is set, content loaded from external `math` archives (i.e. `modules`) is persisted in the file `\jobname.sms`.

If `usesms` is set, the content of the `.sms`-file is loaded, obviating the need to reprocess the original files.

The options are not mutually exclusive, but care should be taken if dependencies have changed between builds.

This offers two advantages:

1. If a document has many (transitive) dependencies, `usesms` should significantly speed up the build process, and
2. setting `usesms` allows for distributing the `.sms`-file to make the document *standalone*, allowing for compilation without needing imported/used modules to be present.

The options `debug`, `mathhub`, `usesms` and `writesms` can also be set by the environment variables `STEX_DEBUG`, `MATHHUB`, `STEX_USESMS` and `STEX_WRITESMS`. In fact, the `MATHHUB` environment variable is the recommended way to set the `MathHub` directory. This is the only option where the *package option* overrides the environment variable.

The environment variables for `USE/WRITESMS` are particularly useful, in that they allow for convenient compilation workflows. For example, the `Build PDF/XHTML/OMDoc`-button in the `IDE` does the following:

```
STEX_WRITESMS=true pdflatex <job>.tex
[bibtex|biber] <job>
STEX_USESMS=true pdflatex <job>.tex
STEX_USESMS=true pdflatex <job>.tex
```

Guaranteeing (in the first run) that all dependencies are loaded from their respective sources and persisted, and in the two subsequent runs read from the generated `.sms` file, (likely) speeding up the subsequent runs significantly.

5.2 Math Archives and the MathHub Directory

`STEX` uses `math archives` to organize document content modularly, without a user having to specify absolute paths, which would differ between users and machines.

All `STEX` archives need to exist in the local `MathHub`-directory. `STEX` knows where this folder is via one of five means:

1. If the `STEX package` is loaded with the option `mathhub=/path/to/mathhub`, then `STEX` will consider `/path/to/mathhub` as the local `MathHub` directory.
2. If the `mathhub` package option is *not* set, but the macro `\mathhub` exists when the `STEX`-package is loaded, then this macro is assumed to point to the local `MathHub` directory; i.e. `\def\mathhub{/path/to/mathhub}\usepackage{stex}` will set the `MathHub` directory as `path/to/mathhub`.
3. Otherwise, `STEX` will attempt to retrieve the system variable `MATHHUB`, assuming it will point to the local `MathHub` directory. Since this variant needs setting up only *once* and is machine-specific (rather than defined in tex code), it is compatible with collaborating and sharing tex content, and hence recommended.
4. If that too fails, `STEX` will look for a file `~/.stex/mathhub.path`. If this file exists, `STEX` will assume that it contains the path to the local `MathHub`-directory. This method is recommended on systems where it is difficult to set environment variables, and is used by the `IDE` setup.
5. Finally, if all else fails, `STEX` considers `~/MathHub` to be the `MathHub` directory.

The `STEX IDE` allows you to directly download `math archives` from `gl.mathhub.info` – currently available `archives` there are:

- `sTeX/*` – a group of semi-experimental documents showcasing `STEX`3.3 features,
- `smgloM/*` – a vast collection of multilingual `modules` of concepts in mathematics and computer science. The SMGloM predates `STEX`3 and is thus largely “underannotated” with respect to (formal) semantics,
- `courses/*` – a vast collection of lecture slides and notes in computer science for courses held by Michael Kohlhase. They largely make use of SMGloM `modules`.

5.2.1 The Structure of Math Archives

An `archive` group/name is stored in the directory `<MathHub>/group/name`; e.g. assuming your local MathHub-directory is set as `/user/foo/MathHub`, then in order for the sTeX/Documentation `archive` to be found by the `sTeX` system, it needs to be in `/user/foo/MathHub/sTeX/Documentation`.

Each such `archive` needs two subdirectories:

- `/source` – this is where all your tex files go.
- `/META-INF` – a directory containing a single file `MANIFEST.MF`, the content of which we will consider shortly.

An additional `lib`-directory is optional, and is discussed in ??.

5.2.2 MANIFEST.MF-Files

The `MANIFEST.MF` in the `META-INF` directory consists of key-value-pairs, informing `sTeX` (and associated software, e.g. `MMT`) of various properties of an `archive`. For example, the `MANIFEST.MF` of the sTeX/Documentation archive looks like this:

```
id: sTeX/Documentation
ns: http://mathhub.info/sTeX/Documentation
narration-base: http://mathhub.info/sTeX/Documentation
format: stex
title: The sTeX Documentation
teaser: The full Documentation for the sTeX system
url-base: https://stexmmt.mathhub.info/:sTeX
dependencies:sTeX/ComputerScience/Software,sTeX/MathTutorial
ignore: */code/*|*/tikz/*|*/tutorial/solution/*
```

Many of these are in fact ignored by `sTeX`, but some are important:

`id`: The name of the `archive`, including its group (e.g. `sTeX/Document`). This is used by the `MMT` system in favor of the directory, but `TeX`'s limited access to the file system enforces the directory structure.

`source-base` or

`ns`: The namespace from which all `symbol` and `module MMT-URIs` in this `archive` are formed.

`narration-base`: The namespace from which all document `MMT-URI` in this repository are formed. It can safely match the `ns`-field.

`url-base`: A URL that is formed as a basis for *external references*. and hyperlinks. An `MMT` (or comparable system) instance should run there and host (`sTeX`-generated) `HTML`.

`dependencies`: All `archives` that this `archive` depends on. `sTeX` ignores this field, but `MMT` can pick up on them to resolve dependencies, e.g. when downloading `archives` in the `IDE`, which will also download all dependencies.

`ignore`: A regular expression of `.tex` files in the `source` directory that should be ignored; e.g. they will not be compiled when building a whole directory or archive in the `IDE`.

5.3 The lib-Directory

A `math archive`/`archive` may have a `lib` directory primarily intended for preamble code, `packages`, `.bib` files, etc., the files in which can be referenced in various ways.

Additionally, a *group* of archives `group` may have an additional `archive` `group/meta-inf`. If this `meta-inf` archive has a `/lib`-subdirectory, they too will be considered by the following.

`\libinput` `\libinput` {`some/file`} searches for a file `some/file[.tex]` in

- the `lib`-directory of the current archive, and
- the `lib`-directory of a `meta-inf`-archive in (any of) the archive groups containing the current archive

and `\inputs` all found files in reverse order; e.g. `\libinput{preamble}` in a `.tex`-file in `sTeX/Documentation` will *first* input `.../sTeX/meta-inf/lib/preamble.tex` and then `.../sTeX/Documentation/lib/preamble.tex`.

`\libinput` will throw an error if *no* candidate for `some/file` is found.

`\libinput[some/archive]{some/file}` will do the same, but starting in the `lib` directory of the `math archive` `some/archive`.

`\libusepackage` `\libusepackage` [`package-options`] {`some/file`} searches for a file `some/file.sty` in the same way that `\libinput` does, but will call `\usepackage[package-options]{path/to/some/file}` instead of `\input`.

`\libusepackage` throws an error if not *exactly one* candidate for `some/file.sty` is found.

`\addmhbibresource` `\addmhbibresource` [`some/archive`] {`some/file`} searches for a file like `some/file.bib` in `some/archive`'s `lib` directory and calls `\addbibresource` to the result.

`\libusetikzlibrary` `\libusetikzlibrary` behaves like `\libusepackage` but looks specifically for tikz libraries and calls `\usetikzlibrary` on the results.

throws an error if not *exactly one* candidate for the library is found.

A good practice is to have individual `sTEX` fragments follow basically this document frame:

```
\documentclass{sTEX}
\libinput{preamble}
\setsectionlevel{<your preference>}
\begin{document}
\IfInputref{}{
...
\maketitle
\ifstexhtml \else \tableofcontents \fi
}
...
\IfInputref{}{\libinput{postamble}}
\end{document}
```

Then the `preamble.tex` files can take care of loading the generally required `packages`, setting presentation customizations etc. (per archive or archive group or both), and a `postamble.tex` can e.g. print the bibliography, index etc.

`\libusepackage` is particularly useful in such a `preamble.tex` when we want to use custom packages that are not part of a `TeX` distribution, or on CTAN. In this case we commit the respective packages in one of the `lib` folders and use `\libusepackage` to load them.

5.4 Basic Macros

`\sTeX` The `\sTeX` macro produces this `STEX` logo. It is provided by the `sTEX-logo` package, `\stex` included with the `stex` package.

`\ifstexhtml` The `\ifstexhtml` conditional is *true* if the current compilation generates `HTML`, and *false* otherwise (i.e. generates `PDF`).

`\STEXinvisible` `\STEXinvisible{\langle code \rangle}`

Processes `\langle code \rangle`, but does not generate any output. In the `HTML`, `\langle code \rangle` is exported with `display:none`.

Can be used to declare formal content and preserve its semantics in `HTML` without generating output.

Chapter 6

Document Features

6.1 Document Fragments

sfragment (env.) To make reusability of document fragments more feasible, **STEX** provides the **sfragment** environment. `\begin{sfragment}[id=<id>,short=<short title>]<section title>` calls `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph` or `\ subparagraph` with argument `<section title>` depending on the current *section level* and availability, and increases the level accordingly.

The `<id>` can be used for cross-document references (see ??).

blindfragment (env.) In the case where we want to increase the section level *without* producing a corresponding section header, the **blindfragment** environment can be used. This allows e.g. typesetting `\sections` before the first `\chapter`.

\skipfragment The `\skipfragment` macro “skips an **sfragment**”, i.e. it just steps the respective sectioning counter. This macro is useful, when we want to keep two documents in sync structurally, so that section numbers match up: Any section that is left out in one becomes a `\skipfragment`.

\setsectionlevel The `\setsectionlevel` macro sets the current section level to that provided as argument. This is particularly useful in the preamble of a document, as to be ignored in e.g. `\inputref` and make sure that sectioning proceeds as desired; e.g. `\setsectionlevel{section}` make sure that the first **sfragment** will be typeset as a `\section` (rather than e.g. a `\part`).

\currentsectionlevel The `\currentsectionlevel` macro produces the literal string corresponding to the current section level – e.g. within a chapter (but outside of a section), `\currentsectionlevel` produces “chapter”.
The `\Currentsectionlevel` macro does the same, but capitalizes the first letter; e.g. in the above situation, `\Currentsectionlevel` produces “Chapter”.

6.2 Using and Referencing Document Fragments

`\inputref` `\inputref [<archive>]{<file>}`

Inputs the file `<file>` in `<archive>`'s source directory. If `[<archive>]` is empty, the current `archive`'s source directory is used. If there is no current `archive`, `<file>` is resolved relative to the current file.

The file's content is processed within a `TEX` group when using `pdflatex`. When converting to `HTML` however, the file is not processed *at all*, and instead, a reference to the file is inserted, that can be replaced by the `HTML` generated by the referenced file by e.g. the `MMT` system.

This is the recommended method to assemble documents from individual `.tex` files.

`\mhinput` Like `\inputref`, but actually calls `\input` in both `PDF` and `HTML` mode. Useful for small fragments or those without `modules`, but generally `\inputref` should be preferred.

`\ifinputref` `\ifinputref` is a `TEX` conditional for whether the current file is currently processed via `\IfInputref` `\inputref`.

`\IfInputref {<true code>}{<false code>}` behaves like `\ifinputref{true code}\else{false code}\fi` when using `pdflatex`; in `HTML` mode however, *both* arguments are processed and marked-up accordingly, so a hosting server (like `MMT`) can dynamically decide which parts to show or omit.

`\mhgraphics` `\mhgraphics` If the `graphicx` package is loaded, `\mhgraphics` takes the same arguments as `\includegraphics`, with the additional optional key `archive`. It then resolves the file path in

`\mhgraphics[archive=some/archive]{some/image}` relative to the `source`-folder of the `some/archive` `archive`. If no `archive` is provided, the file path `some/image` is resolved relative to the current `archive` (if existent).

`\cmhgraphics` additional wraps the image in a `center`-environment.

`\lstinputmhlisting` `\lstinputmhlisting` Like `\mhgraphics`, but for `\lstinputlisting` instead of `\includegraphics`. Only defined if the `listings` package is loaded.

6.3 Cross-Document References

If we take features like `\inputref` and `\mhinput` (and the `sfragment` environment) seriously and build large documents modularly from individually compiling documents for sections, chapters and so on, cross-referencing becomes an interesting problem.

Say, we have a document `main.tex`, which `\inputrefs` a section `section1.tex`, which references a definition with label `some_definition` in `section2.tex` (subsequently also `\inputrefed` in `main.tex`). Then the numbering of the definition will depend on the *document context* in which the document fragment `section2.tex` occurs - in `section2.tex` itself (as a standalone document), it might be *Definition 1*, in `main.tex`

it might be *Definition 3.1*, and in `section1.tex`, the definition *does not even occur*, so it needs to be referenced by some other text.

What we would want in that instance is an equivalent of `\autoref`, that takes the document context into account to yield something like *Definition 1*, *Definition 3.1* or “*Definition 1 in the section on Foo*” respectively.

For that to work, we need to supply (up to) three pieces of information:

- The *label* of the reference target (e.g. `some_definition`),
- (optionally) the *file/document* containing the reference target (e.g. `section2`). This is not strictly necessary if the reference target occurs in the *same* document, but if not, we need to know where to find the label,
- (optionally) the document context, in which we want to refer to the reference target (e.g. `main`).

Additionally, the document in which we want to reference a label needs a title for external references.

```
\sref
\sref [archive=<archive1>,file=<file1>]
{<label>}[archive=<archive2>,file=<file2>,title=<title>]
```

This `macro` references `<label>` (declared in `<file1>` in `math archive <archive1>`). If the object (section, figure, etc.) with that label occurs (eventually) in the same document, `\sref` will ignore the second set of optional arguments and simply defer to `\autoref` if that command exists, or `\ref` if the `hyperref` package is not included.

If the referenced object does *not* occur in the current document however, `\sref` will refer to it by the object’s name as it occurs in the file `<file2>` in `archive <archive2>`, followed by the title.

In `HTML` mode, the reference additionally links to the `HTML` of the `file1`.¹¹

This works by storing labels during compilation in a file `<jobname>.sref`, analogous to e.g. the `.toc`. Note that this consequently requires both `file1.tex` and `file2.tex` to have been compiled previously, to generate the `.sref` file.

For example, doing

```
\sref[file=tutorial/full.en]{sec:basics}[file=tutorial.en,title=the \stex Tutorial]
in this very document fragment ([sTeX/Documentation]macros/sref.en.tex) will yield
Part I (The Basics) in the \stex Tutorial if compiled itself, or if compiled as part of the
\stex manual, and will yield the \autoref link chapter 2 in the documentation (which
includes the tutorial).
```

```
\srefsetin
\srefsetin [<archive2>]{<file2>}{<title>}
```

Sets a default value for the optional arguments `<archive2>`, `<file2>` and `<title>` of `\sref`. If the second set of optional arguments in `\sref` are omitted, these default values are used. Particularly useful to set in a preamble.

```
\sreflabel
\sreflabel {<label>} sets a label analogous to \label{<label>}, but for use in \sref.
Note that for every \stex macro or environment that takes an optional id=<id>
argument, the <id> (if non-empty) generates an \sreflabel automatically.
For example, \begin{sfragment}[id=foo]{Foo} is equivalent to
\begin{sfragment}{Foo}\sreflabel{foo}.
```

```
\extref \extref [archive=<archive1>,file=<file1>]
{<label>} {archive=<archive2>,file=<file2>,title=<title>}
```

Like `\sref`, but with the third argument mandatory, `\extref` will *always* produce the output as if `<label>` would *not* occur in the current document.

Chapter 7

Modules and Symbols

7.1 Modules

A `module` is required to declare any new formal content such as `symbols` or `notations` (but not `variables`, which may be introduced anywhere).

↳ An `STEX module` corresponds to an `MMT/OMDOC theory`. As such
→ it gets assigned an `MMT-URI` (*universal resource identifier*) of the form
~~~`T <namespace>?<module-name>`.

`smodule (env.)` A new module is declared using the basic syntax

`\begin{smodule}[options]{ModuleName}... \end{smodule}`.

A module is required to declare any new formal content such as symbols or notations (but not variables, which may be introduced anywhere).

The `smodule`-environment takes several keyword arguments, all of which are optional:

`title (<token list>)` to display in customizations.

`style (<string>*)` for use in customizations, see ??

`id (<string>)` for cross-referencing, see `\sreflabel`.

`ns (<URI>)` the namespace to use. *Should not be used, unless you know precisely what you're doing.* If not explicitly set, is computed from the containing file and `archive`'s namespace.

`lang (<language>)` if not set, computed from the current file name (e.g. `foo.en.tex`).

`sig (<language>)` see below.

---

---

`\STEXexport` `\STEXexport{<code>}` executes `<code>` immediately and every time the current `module` is being used.



For technical reasons, `\STEXexport` processes its content in the `expl3` category code scheme – what this means is that all spaces are ignored entirely, and the characters `_` and `:` are valid characters in `macro` names.

In practice, this means you will have to use the `~` character for spaces, and if you want to use a subscript `_`, you should use the `macro` `\c_math_subscript_token` instead.

Also, note that no *global macro* definitions should happen in `\STEXexport`; this can lead to unexpected behaviour if the containing `module` has been used previously in the current document.

### 7.1.1 Signature Modules, Languages, and Multilinguality

if the current file is a translation of a file with the same base name but a different language suffix, setting `sig=<lang>` will preload the `module` from that language file. This helps ensuring that the (formal) content of both `modules` is (almost) identical across languages and avoids duplication.

For example, we can have a file `Foo.en.tex`, that declares and documents a `module` `Foo` (using `\begin{smodule}{Foo}`). If we put a file `Foo.de.tex` next to it, we can do `\begin{smodule}[sig=en]{Foo}` to have all the content in the `module` `Foo` (as declared in `Foo.en.tex`) available and translate its document content to german.

The `MMT` backend, when serving `STEX` content as `HTML`, will always attempt to find documentation in the language corresponding to the context; e.g. a user's preference.

## 7.2 Symbol Declarations

---

`\symdecl` `\symdecl {<mname>}[<options>]`

The `\symdecl` macro is the simplest way to introduce a new `symbol`. If `<options>` contains `name=<name>`, then `<name>` is the *name* of the `symbol`; otherwise, `<mname>` is used for the *name*. Additionally, a `semantic macro` `\mname` is generated.

The starred variant `\symdecl*` does not generate a `semantic macro`, in which case the `name`-option is superfluous.

→ `\symdecl` introduces a new `MMT/OMDoc constant` in the current `module` (i.e. → `MMT/OMDoc theory`). Correspondingly, they get assigned the `MMT-URI` ↵ `<module-URI>?<constant-name>`.

`\symdecl` takes the following optional arguments:

`name` see above,

`args` the arity of the `symbol` and its `semantic macro`; may be a number `0...9` or a string consisting of the characters `i`, `a`, `b` and `B` of length  $\leq 9$ ,

`type` the `symbol`'s `type`,

`def` the `symbol`'s `definiens`,

`return` the `symbol`'s *return code* (see below), most commonly the `semantic macro` of a `mathematical structure`,

`assoc` how to resolve arguments of `mode` `a` or `B`; may be `pre`, `bin`, `binl`, `binr` or `conj`,

`reorder` how to reorder the arguments in `OMDoc` (*advanced*),

`role` `symbols` with certain roles are treated in particular ways in `MMT/OMDoc` (*advanced*),

`argtypes` `TODO`<sup>12</sup>.

---

`\textsymdecl` `\textsymdecl{<mname>}{<options>}{<code>}`

Like `\symdecl`, but requires that the `symbol` has arity 0 (hence `\textsymdecl` does not take the `args`-option), and generates a `semantic macro` that takes no arguments in either text or math mode, and produces marked-up `<code>` as output.

Additionally, a `macro` `\<mname>name` is generated that produces `<code>` without any semantic markup.

---

`\symdef` `\symdef{<mname>}{<options>}{<notation>}`

Combines the functionalities and optional arguments of `\symdecl` and `\notation` in one.

### 7.2.1 Returns

Assume we have a `symbol` `foo` with `semantic macro` `\foo`, (exemplary) taking two arguments, and `return=<code>`. If we do `\foo{a}{b}!`, the return code is simply ignored. If we do `\foo{a}{b}` *without* the `!`, here is what happens:

1. `STEX` will replace `#1` and `#2` in `<code>` by `a` and `b`, yielding `<retcode>`.
2. `STEX` will insert `<retcode>{\<foo>{a}{b}!}` in the input token stream.

This means that `<code>` should contain at most `<arity of foo>` argument markers, and eat precisely one argument appended to `<code>`.

When in doubt, we recommend only using `semantic macros` for `mathematical structures` (with only optional arguments) and `\apply` (with only optional arguments) in `return`.

## 7.3 Referencing Symbols

---

`\symref` `\sr` `\symref{<symbol>}{{<text>}}`

The `\symref` macro (and its short version `\sr`) is the most general variant to mark-up arbitrary `LATeX` code `<text>` with the `symbol`.

---

<sup>12</sup>TODO: experimental

This is as good a place as any other to explain how  $\text{STEX}$  resolves a string `symbol` to an actual `symbol`.

If `\symbol` is a `semantic macro`, then  $\text{STEX}$  has no trouble resolving `symbolname` to the full `MMT-URI` of the `symbol` that is being invoked.

However, especially in `\symname` (or if a `symbol` was introduced using `\symdecl*` without generating a `semantic macro`), we might prefer to use the `name` of a symbol directly for readability – e.g. we would want to write A `\symname{natural number}` is... rather than A `\symname{Nat}` is....  $\text{STEX}$  attempts to handle this case thusly:



If `symbol` does *not* correspond to a `semantic macro` `\symbol` and does *not* contain a ?, then  $\text{STEX}$  checks all `symbols` currently in scope until it finds one, whose name is `symbol`. If `symbol` is of the form `pre?name`,  $\text{STEX}$  first looks through all `modules` currently in scope, whose full `MMT-URI` ends with `pre`, and then looks for a symbol with name `name` in those. This allows for disambiguating more precisely, e.g. by saying `\symname{Integers?addition}` or `\symname{RealNumbers?addition}` in the case where several additions are in scope.

 `\symref{\symbol}{\text}` in MMT/OMDoc generates the term  
`<OMS name="\symbol URI"/>`.

---

`\symname` `\symname[pre=\pre,post=\post]{\symbol}`

`\Symname` If the `symbol` referenced by `\symbol` has name `name`, this is a shortcut for

`\Sn` `\symref{\symbol}{\pre name \post}`.

`\sns` For example, given a symbol `agroup` with name `abelian group`, we can do `\symname[pre=Non-,post=s]{agroup}` to produce `Non-abelian groups`.

`\sn` is a shorter variant for `\symname`; `\Symname` and `\Sn` additionally capitalize the first letter. `\sns` and `\Sns` are short for `\sn [post=s]` and `\Sn [post=s]`, respectively.

---

`\srefsym` `\srefsym{\symbol}{\text}`

`\srefsymuri` turns `\text` into a link to

- The documentation of `\symbol`, if it occurs in the same document, or
- the `symbol`'s documentation online, based on the containing `math archive`'s `url-base`.

`\srefsymuri` does the same, but expects a `symbol`'s full `MMT-URI` as first argument. This is particularly useful for e.g. customizing highlighting (see ??).

---

`\symuse` `\symuse{\symbol}` behaves exactly like a `semantic macro` for `\symbol`.

## 7.4 Notations and Semantic Macros

---

`\notation` `\notation{<symbol>}[<options>]{<code>}`

introduces a new `notation` for the referenced `symbol`.

The starred variant `\notation*` sets this `notation` as the (new) default `notation`.

The optional arguments are:

- `prec=(opprec);<argprec 1>x...x<argprec n>`: An `operator precedence` and one `argument precedence` for each argument of the `semantic macro`. If no `argument precedences` are given, all `argument precedences` are equal to the `operator precedence`. By default, all `precedences` are 0, unless the `symbol` takes no argument, in which case the `operator precedence` is `\neginfpref` (negative infinity).
- `prec=nobrackets` is an abbreviation for `prec=\neginfpref;\infprec x...x\infprec`.
- `op=<code>`: An `operator notation`. If none is given, the notation component marked with `\maincomp` is used. If no `\maincomp` occurs in the `notation`, the default `operator notation` is `\symname{<symbol>}`.
- `variant=<id>`: An id for this `notation`. The key `variant=` can be omitted; i.e. `\notation[foo]` is equivalent to `\notation[variant=foo]`.

---

`\comp` `\maincomp`

`\comp` is used to mark notation components in a `\notation` to be highlighted. Additionally, each `notation` can use `\maincomp` at most once to mark the *primary* notation component.



Ideally, `\comp` would not be necessary: Everything in a `notation` that is *not* an argument should be a notation component. Unfortunately, it is computationally expensive to determine where an argument begins and ends, and the argument markers `#n` may themselves be nested in other `macro` applications or `TeX` groups, making it ultimately almost impossible to determine them automatically while also remaining compatible with arbitrary highlighting customizations (such as tooltips, hyperlinks, colors) that users might employ, and that are ultimately invoked by `\comp`.



Note that it is required that

1. the argument markers `#n` never occur inside a `\comp`, and
2. no `semantic macros` may ever occur inside a `notation`.

Both criteria are not just required for technical reasons, but conceptionally meaningful:

The underlying principle is that the arguments to a `semantic macro` represent *arguments to the mathematical operation* represented by a `symbol`. For example, a `semantic macro` application `\plus{a}{b}` would represent *the actual addition of (mathematical objects) a and b*. It should therefore be impossible for `a` or `b` to be part of a notation component of `\plus`.

Similarly, a `semantic macro` can not conceptually be part of the `notation` of `\plus`,

since a `symbol` represents a *distinct (mathematical) concept* with *its own semantics*, and `notations` are syntactic representations of the very `symbol` to which the `notation` belongs.



If you want an argument to a `semantic macro` to be a purely syntactic parameter, then you are likely somewhat confused with respect to the distinction between the precise *syntax* and *semantics* of the `symbol` you are trying to declare (which happens quite often even to experienced `STEX` users, like us), and might want to give those another thought - quite likely, the concept you aim to implement does not actually represent a semantically meaningful (mathematical) concept, and you will want to use `\def` and similar native `LATEX` macro definitions rather than `semantic macros`.

---

`\setnotation` The first `notation` provided will stay the default `notation` unless explicitly changed:  
`\setnotation{\langle symbol \rangle}{\langle id \rangle}` sets the default `notation` of `\langle symbol \rangle` to that with id `\langle id \rangle`.

#### 7.4.1 Precedences and Bracketing

---

`\infprec` and `\neginfprec` represent *infinitely large* and *infinitely small precedences*, respectively.



`STEX` decides whether to insert parentheses by comparing `operator precedences` to a *downward precedence*  $p_d$  with initial value `\infprec`. When encountering a `semantic macro`, `STEX` takes the `operator precedence`  $p_{op}$  of the `notation` used and checks whether  $p_{op} > p_d$ . If so, `STEX` inserts parentheses.

When `STEX` steps into an argument of a `semantic macro`, it sets  $p_d$  to the respective `argument precedence` of the `notation` used.

##### Example 24

Consider `semantic macros` `\plus` and `\mult` taking two arguments, with `notations`  $a+b$  and  $a \cdot b$  respectively, and `precedences` 100 for `\plus` and 50 for `\mult`.

Consider `$\plus{a, \mult{b, \plus{c, d}}}$` (i.e.  $a + b \cdot (c + d)$ ). Then:

1. `STEX` starts out with  $p_d = \infprec$ .
2. `STEX` encounters `\plus` with  $p_{op} = 100$ . Since  $100 \not> \infprec$ , it inserts no parentheses.
3. Next, `STEX` encounters the two arguments for `\plus`. Both have no specifically provided `argument precedence`, so `STEX` uses  $p_d = p_{op} = 100$  for both and recurses.
4. Next, `STEX` encounters `\mult{b, ...}`, whose `notation` has  $p_{op} = 50$ .
5. We compare to the current downward `precedence`  $p_d$  set by `\plus`, arriving at  $p_{op} = 50 \not> 100 = p_d$ , so `STEX` again inserts no parentheses.

6. Since the **notation** of `\mult` has no explicitly set **argument precedences**, **STEX** again uses the **operator precedence** for the arguments of `\mult`, hence sets  $p_d = p_{op} = 50$  and recurses.
7. Next, **STEX** encounters the inner `\plus{c, ...}` whose **notation** has  $p_{op} = 100$ . We compare to the current downward **precedence**  $p_d$  set by `\mult`, arriving at  $p_{op} = 100 > 50 = p_d$  – which finally prompts **STEX** to insert parentheses, and we proceed as before.

`\dobracket` `\dobraackets{\langle code\rangle}` wraps parentheses around `\{\langle code\rangle\}`.

`\withbrackets` `\withbrackets{\langle left\rangle}{\langle right\rangle}{\langle code\rangle}` uses the opening and closing parentheses `\langle left\rangle` and `\langle right\rangle` for the next pair of parentheses automatically inserted in `\{\langle code\rangle\}`.

#### 7.4.2 Notations for Argument Sequences

The following **macros** can be used in **notations** that take **mode a** or **B** arguments:

`\argssep` `\argssep{\langle parameter token\rangle}{\langle separator\rangle}`

takes the elements of the argument sequence in position `\langle parameter token\rangle` and separates them by `\langle separator\rangle`.

Note that the first argument *must* be a parameter token of the form `#k`, and the argument at position `k` of the **notation** has to have **argument mode a** or **B**.

`\argmap` `\argmap{\langle parameter token\rangle}{\langle code\rangle}{\langle separator\rangle}`

takes the elements of the argument sequence in position `\langle parameter token\rangle`, applies the code `\{\langle code\rangle\}` to each of them (which therefore should use `##1`) and separates them by `\langle separator\rangle`.

For example, the **notation** `\argmap{\#1}{X^{##1}}{++}` applied to the argument `{a,b,c}` produces  $X^a + X^b + X^c$ .

`\argarraymap` `TODO13`

#### 7.4.3 Semantic Macros

Assume we have a **semantic macro** `\smacro` taking (exemplary) two arguments. The precise behaviour of `\smacro` depends on whether we are in text or math mode.

**Math Mode** `\smacro!` produces the default **operator notation** of its **symbol**. Without `!`, `\smacro` expects at least two arguments, and `\smacro{a}{b}!` produces the default **notation** of its **symbol**.

If the **symbol** has a **return** code, then `\smacro{a}{b}` continues with executing the **return** code. Otherwise, `\smacro{a}{b}` also simply produces the default **operator notation**.

The starred variants `\smacro*` and `\smacro!*` behave as in *text mode*.

**Text Mode** `\smacro!{\langle arg\rangle}` marks up `\langle arg\rangle` similarly to how `\symref{\smacro}{\langle arg\rangle}` would.

Without the `!`, `\smacro` still only takes a single argument, but it is expected, that within `\langle arg\rangle`, the arguments for the `symbol` are explicitly marked up. The `\comp` macro is allowed in `\langle arg\rangle` to determine the components of `\langle arg\rangle` to be highlighted.

**\arg** The `\arg` macro can be used to explicitly mark the arguments of a `semantic macro` in text mode.

By default, they are numbered consecutively; e.g. `\smacro{... \arg{a} ... \arg{b}}` determines `a` and `b` to be the (first and second) arguments.

The starred variant `\arg*` allows for marking up the arguments, but does not produce any output. This can be used to provide arguments that are not mentioned in the text we want to mark up, because they are implicitly obvious or mentioned elsewhere.

If we want to change the order of the arguments, we can provide the precise argument number as an optional argument; e.g. `\smacro{... \arg[2]{a} ... \arg[1]{b}}` determines `b` to be the first and `a` to be the second argument.

An argument number may be used repeatedly, if the corresponding `argument mode` is `a` or `B`.

Applications of `semantic macros` with arguments are translated to  
M → MMT/OMDoc as OMA-terms with head `<OMS name="(symbol)" />`, or  
M → `<OMBIND name="(symbol)" />`, depending on the absence or presence of `arg`  
T → `ument mode` b or B arguments.  
Semantic macros with no arguments or invoked with `!` correspond to OMS directly.

## 7.5 Simple Inheritance

There are three `macros` that allow for opening a `module`, making its contents available for use:

**\usemodule** `\usemodule{\langle module\rangle}` is allowed anywhere and makes the `module`'s contents available up to the current `TeX` group. This is the right `macro` to use outside of `modules`, or when none of its contents use any of the used `module`'s `symbols` directly (e.g. in `types` or `definientia`).

**\requiremodule** `\requiremodule{\langle module\rangle}` is only allowed in `modules` and makes the required `module`'s contents available within the current `module`. The imported `symbols` can be safely used in `types` and `definientia`, but not in the `return` code of `symbols`, and the imported content is not exported further – i.e. using the current `module` does not also open the required `module`.

**\importmodule** `\importmodule{\langle module\rangle}` is only allowed in `modules` and makes the required `module`'s contents available within the current `module`. The imported `symbols` can be safely used anywhere, and the imported content exported to any `modules` subsequently importing the current one.

$\hookrightarrow M \rightarrow$  In **MMT**, every **document** and every **module** induces an **MMT theory**. `\usemodule`  
 $\dashv M \rightarrow$  induces and **MMT include** in the document **theory**, `\importmodule` and  
 $\rightsquigarrow T \rightsquigarrow \requiremodule$  both induce an **include** in the **module's theory**.

It is worth going into some detail how exactly `\usemodule`, `\importmodule` and `\requiremodule` resolve their arguments to find the desired **module** – which is closely related to the *namespace* generated for a **module**, that is used to generate its **MMT-URI**.

Ideally, **STEX** would allow for arbitrary **MMT-URIs** for **modules**, with no forced relationships between the *logical namespace* of a **module** and the *physical location* of the file declaring it – like **MMT** in fact allows for.

Unfortunately, **TEX** only provides very restricted access to the file system, so we are forced to generate namespaces systematically in such a way that they reflect the physical location of the associated files, so that **STEX** can resolve them accordingly. Largely, users need not concern themselves with namespaces at all, but for completeness sake, we describe how they are constructed:



- If `\begin{smodule}{Foo}` occurs in a file `/path/to/file/Foo[.(lang)].tex` which does not belong to an **math archive**, the namespace is `file://path/to/file`.
- If the same statement occurs in a file `/path/to/file/bar[.(lang)].tex`, the namespace is `file://path/to/file/bar`.

In other words: outside of **math archives**, the namespace corresponds to the file URI with the filename dropped iff it is equal to the **module** name, and ignoring the (optional) language suffix.

If the current file is in an **archive**, the procedure is the same except that the initial segment of the file path up to the **archive**'s **source** directory is replaced by the **archive**'s namespace URI.

Conversely, here is how namespaces/URIs and file paths are computed in import statements, exemplary `\importmodule`:



- `\importmodule{Foo}` outside of an **archive** refers to **module Foo** in the current namespace. Consequently, Foo must have been declared earlier in the same file or, if not, in a file `Foo[.(lang)].tex` in the same directory.
- The same statement *within* an **archive** refers to either the **module Foo** declared earlier in the same file, or otherwise to the **module Foo** in the **archive**'s top-level namespace. In the latter case, it has to be declared in a file `Foo[.(lang)].tex` directly in the **archive**'s **source** directory.
- Similarly, in `\importmodule{some/path?Foo}` the path `some/path` refers to either the sub-directory and relative namespace path of the current directory and namespace outside of an **archive**, or relative to the current **archive**'s top-level namespace and **source** directory, respectively.

The **module Foo** must either be declared in the file `(top-directory)/some/path/Foo[.(lang)].tex`, or in `(top-directory)/some/path[.(lang)].tex` (which are checked in that order).



- Similarly, `\importmodule[Some/Archive]{some/path?Foo}` is resolved like the previous cases, but relative to the archive `Some/Archive` in the `MathHub` directory.

## 7.6 Variables and Sequences

---

`\vardef` `\vardef{<mname>}[<options>]{<notation>}`

Takes the same arguments as `\symdef`, but produces a `variable` rather than a `symbol`. `Variables` definitions are always local to the current `TeX` group and are allowed anywhere (i.e. outside of `modules`).

`<options>` may include the additional keyword `bind`, in which case the `variable` will be appropriately abstracted away in statements (see also `\varbind`).

Unlike `\symdef`, there is no starred variant `\vardef*` – `variables` always generate a semantic macro.

The semantic macro for a `variable` behaves analogously to that of a `symbol`.

M → `Variables` induces the same `MMT/OMDoc` terms as `symbols` do, except for the head of the term being `<OMV name="..."/>` instead of `<OMS/>`.

---

`\varnotation` `\varnotation{<variable>}[<options>]{<notation>}`

Takes the exact same arguments as `\notation`, but attaches an additional `notation` to the `variable` `<variable>` rather than a `symbol`.

---

`\svar` `\svar{<name>}[<text>]`

Semantically marks up `<text>` as representing a `variable` `<name>`. The `variable` does not need to have been defined prior. If no `<name>` is given, `<text>` will be used as the name.

This is useful in situations like “throwaway expressions” or remarks; e.g.

`$\plus{\svar{n}, \svar{m}}$` means...

---

`\varseq` `\varseq{<mname>}[<options>]{<range>}{<notation>}`

Declares a new `variable` sequence. The `<options>` are the same as for `\vardef`. If not provided, `args=1` by default (a 0-ary sequence would just be a normal `variable`).

A `type` (given as `type=`) is interpreted to be the `type` of every element  $a_i$  of the sequence  $a_1, \dots, a_n$  (not of the sequence itself). If the `type` is itself a sequence  $A_1, \dots, A_n$ , the assumption is that its range is the same as the one of the new sequence, and the type of every  $a_i$  in the sequence is  $A_i$ .

`<range>` needs to be a comma-separated sequence of either `args` many arguments, or `\ellipses`.

The resulting semantic macro is allowed anywhere `STEX` expects an argument mode `a` or `B` argument.

\ellipses Represents ellipses in a range; produces `\ellipses` in math mode.

\seqmap `\seqmap{\langle code \rangle}{\langle sequence \rangle}`

Maps the function `\langle code \rangle` (containing #1) over every element of the `\langle sequence \rangle`. Is allowed anywhere STeX expects an argument mode a or B argument.

## 7.7 Structures

Mathematical structure bundle interdependent symbols together.

`mathstructure (env.)` `\begin{mathstructure}{\langle mname \rangle}{\langle name \rangle, this=\langle code \rangle}` opens a new mathematical structure with name `\langle mname \rangle` (if provided) or `\langle name \rangle` (otherwise), and semantic macro `\mname`. It subsequently behaves like the `smodule` environment.

\this The optional `this=\langle code \rangle` option allows for setting the typesetting of the `\this` macro within a `mathstructure`. In particular, `\this` can be used in notations for symbols declared in the structure. `\this` can be thought of as representing “the” (current) instance of this structure.

`extstructure (env.)` `\begin{extstructure}{\langle mname \rangle}{\langle name \rangle, this=\langle code \rangle}{\langle structs \rangle}` opens a new mathematical structure extending the structures given in `\langle structs \rangle` (a comma-separated list of names).

`extstructure* (env.)` `\begin{extstructure*}{\langle struct \rangle}` opens a new mathematical structure conservatively extending the (single) structure `\langle struct \rangle`. Conservative meaning: Every symbol newly introduced in this structure needs to have a definiens. The new symbols are attached as fields directly to `\langle struct \rangle`.

\usestructure The `\usestructure` macro behaves like `\usemodule` for mathematical structures, making the symbols available to use directly.

`mathstructure` make use of the *Theories-as-Types* paradigm (see [MueRabKoh:tat18]):  
M → `\begin{mathstructure}{\langle name \rangle}` creates a nested theory with name `\langle name \rangle`-module. The constant `\langle name \rangle` is defined as a dependent record type with manifest fields, the fields of which are generated from (and correspond to) the constants in `\langle name \rangle`-module.

### 7.7.1 Semantic Macros for Structures

Assume we have a mathematical structure with semantic macro `\struct`:

#### Example 25

```
\begin{mathstructure}{\struct}
\symdef{fielda}{a}
\symdef{fieldb}[args=2]{#1 \maincomp{b} #2}
\symdef{fieldc}[args=2,def=\sn{fieldb}]{#1 \maincomp{c} #2}
```

```

\inliness[name=axiom1]{\conclusion{some axiom}}
\end{mathstructure}
\notation{struct}{StRuCt}

```

- If `\struct` has no `notations`, then `\struct!` produces  $\langle a, b, c \rangle$ . Otherwise, it produces the notation, i.e. `StRuCt`. In both cases, `\struct{}{}` produces  $\langle a, b, c \rangle$  (for reasons that will become clearer in a moment).
- `\struct{}{}` (or `\struct!` in the case where no `notations` are around) can be modified in the following ways:
  - `\struct{}{}[\langle fieldname \rangle, ...]` lets you pick, which precise fields to show, so e.g. `\struct{}{}[fielda,fieldb]` produces  $\langle a, b \rangle$ . By default, `\struct{}{}` shows exactly the fields that have `semantic macros` (which are also used to access the fields in `\struct{...}{fieldname}`).
  - `\struct{}{}[(id)]` lets you pick the `notation` of the “`mathematical structure`” symbol to use to typeset the `structure`; e.g. `\struct{}{}[parens]` yields  $\langle a, b, c \rangle$ , and (combining both) `\struct{}{}[fielda,fieldb][angle]` yields  $\langle a, b \rangle$ .
- The two arguments in `\struct{first}{second}` represent 1. the term that is to be treated as an instance of `\struct`, and 2. the precise field to invoke. If the first is empty, then there is no instance. If the second is empty, `StTeX` will present all of them (that are not assertions). Hence, `\struct{S}{}{}` yields  $\langle a_S, b_S, c_S \rangle$ , `\struct{S}{fielda}` produces  $a_S$ , `\struct{}{fielda}` produces  $a$ , and `\struct{S}{fieldb}{x}{y}` produces  $x b_S y$ .
- For the sake of completion, `\struct{first}!` simply produces the given argument; e.g. `\struct{S}!` simply produces  $S$ .

More precisely: `\struct{\langle code \rangle}` acts like a “type coercion” of `\langle code \rangle` to be an instance of `\struct`.

Of course, it is (occasionally, but) rarely useful to use the `semantic macro` `\struct` by giving it two arguments *manually*; but this is what `StTeX` does when using `\struct` in the `return` of a `symbol` (or `variable`).

Continuing:

- `\struct[comp=\langle code \rangle]{...}{...}` applies `\langle code \rangle` (using #1) to every occurrence of `\maincomp` in the `notations` of the fields, as a replacement for the default modification `\#1_{\this}`. For example, `\struct[comp=\#1^{\langle Foo \rangle}]{S}{}{}` produces  $\langle a^{Foo}, b^{Foo}, c^{Foo} \rangle$ , and `\struct[comp=\#1^{\langle \this \rangle}]{S}{fieldb}{x}{y}` yields  $x b_S y$ .
- `\struct[this=\langle code \rangle]{...}{...}` modifies the way `\this` is being typeset; i.e. the presentation of the first `{...}` argument. For example `\struct[this=T]{S}{}{}` produces  $\langle a, b, c \rangle$  – since `\this` is not being used in the `notations` of the fields. Note that the `this=` and `comp=` variants are (as of yet) mutually incompatible.
- Finally, `\struct[\langle fieldname \rangle=\langle code \rangle]{...}{...}` assigns the field `\langle fieldname \rangle` to the term `\langle code \rangle`. `\langle code \rangle` is subsequently used when using `{...}{...}`, but not in fields directly. For example, `\struct[fielda=A]{S}{}{}` produces  $\langle A!, b_S, c_S \rangle$ , but `\struct[fielda=A]{S}{fielda}` produces  $a_S$ .

Note the insertion of ! behind the A – this is to make sure that assignments to [semantic macros](#) that takes arguments don't accidentally eat more than they should.

Also note that multiple assignments can be done in the same pair of [], or chained – i.e. both `$\struct{fielda=A,fieldb=B}...` and `$\struct{fielda=A}[fieldb=B]...` are valid and equivalent. `$\struct{fielda=A,fielda=B}...` however is not – every field may be assigned at most once.

# Chapter 8

## Statements

**STEX** provides four environments to semantically annotate various kinds of statements:

**sdefinition** (*env.*)

The **sdefinition environment** represents (primarily mathematical) *definitions*; in particular for *symbols*. The contents of the environment will be used as *documentation* for any *symbol* that either occurs as a `\definiendum` (or `\definename`) within the **sdefinition**, or that is listed in the optional `for=` argument of the **environment**.

If a `\definiens` occurs, this will be used by **MMT** as the formal definiens for the respective *symbol*.

**sassertion** (*env.*)

The **sassertion environment** represents *assertions*, i.e. *propositions* such as *theorems, lemmata, axioms*, etc. If a `\conclusion` occurs within the **sassertion**, its argument will be used as the formal *statement* of the assertion.

**sexample** (*env.*)

The **sexample environment** represents examples, the `for=` attribute specifies the concept this is an example for. For counterexamples use `style=counterexample`

**sparagraph** (*env.*)

The **sparagraph environment** represents all other kinds of (logical) paragraphs, such as remarks, comments, transitions between topics, recaps, reminders, etc.

All of these take the same arguments:

- `for=(cs1)`: a comma-separated list of *symbols*.
- The same optional arguments as `\symdecl`, with `macro=` replacing the name of the *semantic macro*. All of them are only relevant, if either `name=` or `macro=` are provided.

As with `\symdecl`, if no `name` is given, but `macro` is, then the same name is used for both the *semantic macro* and the *symbol* itself.

If `name` is given but `macro` isn't, no *semantic macro* is generated. Subsequently, the newly generated *symbol* is added the `for`-list.

- `style`: see ??.
- `title`: a title to use in various styles (see ??).
- `id`: a label to use for `\sref`.

\inlineass The macros `\inlineass`, `\inlinedef` and `\inlineex` behave like the `sassertion`, `sdefinition` and `sexample` environments respectively, but take the text to annotate as an argument, rather than as the body of an environment, and do not break paragraphs.

The same macros available in the environments are also available in the argument of the `\inline*` macros.

\varbind `\varbind{<cls>}`

retroactively attaches the `bind` option to every `variable` provided (as a comma-separated list).

## 8.1 More on Definitions

In `sdefinition` (and `sparagraph` with `style=symdoc`), the following additional macros are available:

\definiendum The `\definiendum` macro behaves largely like `\symref`, but it uses a dedicated highlighting for `definienda` and adds the referenced `symbol` to the `for=` list of the environment.  
\defname `\defname` is to `\definiendum` as `\symname` is to `\symref`. Analogously, `\Defname` behaves like `\Symname`.

`\defnotation` can be used in math mode to apply the `\definiendum` highlighting to `notations`.

\definiens The `\definiens` macro can be used to semantically annotate the `definiens` in a `sdefinition`.

If the `sdefinition` environment has several elements in its `for` list, an optional argument `\definiens[<symbol>]{...}` can be used to tell `STEX` which `symbol`'s definiens this is. By default, the *first* `symbol` in the `for` list is used.

Here is how `MMT` will treat the fragment marked up with `\definiens`:

Firstly, it will attempt to translate its contents into an `MMT/OMDOC` term. This succeeds easily if `\definiens` is some semantic macro (applied to arguments).

Secondly, it will check for all `variables` currently in scope, that were defined with the optional argument `bind`. It then will check, whether a `symbol` is in scope, that has `role=lambda`. If so, it will use that `lambda symbol` to bind all these variables (in the order in which they were defined) in the term. If no `lambda symbol` is found, it will use the `bind symbol` that ships with `STEX`.

The final term will be attached as definiens to the corresponding `MMT constant`, if it was declared in the same `module` as the `\definiens` occurrence.

## 8.2 More on Assertions

\premise  
\conclusion

The `\conclusion` macro can be used to mark up the actual statement within an `sassertion`. The `\premise` macro can be used to additionally mark up *premises*.

If the `sassertion` environment has several elements in its `for` list, an optional argument `\conclusion[⟨symbol⟩]{...}` can be used to tell `STEX` which `symbol`'s statement this is. By default, the *first symbol* in the `for` list is used.

Here is how `MMT` will treat the fragments marked up with `\conclusion` and `\premise`:

Firstly, it will attempt to translate the contents of `\conclusion` into an `MMT/OM-DOC` term  $c$ . This succeeds easily if the `\conclusion` is some semantic macro (applied to arguments).

Secondly, it will collect all fragments marked up with `\premise` and do the same to them ( $p_1, \dots, p_n$ ).

It will then check, whether a `symbol` is in scope, that has `role=implication`. If so, it will use that `implication symbol` to attach all the premises to the conclusion, resulting in  $t := \text{imply}(p_1, \dots, p_n, c)$ .

Next, it will check for all `variables` currently in scope, that were defined with the optional argument `bind`. It then will check, whether a `symbol` is in scope, that has `role forall`. If so, it will use that `forall symbol` to bind all these variables (in the order in which they were defined) in the term  $t$ .

Finally, it will check, whether a `symbol` is in scope, that has `role=judgment`. If so, it will set  $t := \text{judgment}(t)$ .

If no `forall symbol` is found, it will first apply the `judgment symbol` (if existent) and then use the `bind symbol` that ships with `STEX` to bind the `variables`.

The final term will be attached as `type` to the corresponding `MMT constant`, if it was declared in the same `module` as the `\definiens` occurrence.

`sproof (env.)`

TODO<sup>14</sup>

---

<sup>14</sup>TODO: proofs

# Chapter 9

# Customizing Typesetting

There are two kinds of typesetting that can be customized in **STEX**: symbol references (**notation** components, `\symref`, **variables**, etc.) are highlighted using a small set of **macros** that can be simply redefined by authors.

Other **macros** and **environments** usually have more complicated “typesetting rules” associated with them – often in the form of other already existing **environments** that should be used.

Lastly, in **HTML** we can provide custom CSS rules in **math archives** that determine the styling of certain **environments**, so that the actual presentation depends on the document in which the fragments are included (e.g. via `\inputref`), rather than the file the fragment is implemented in.

It is generally recommended to implement these customizations in a preamble in the `lib` directory (see ??)

## 9.1 Highlighting Symbol References

---

`\symrefemph`  
`\symrefemph@uri`

`\symrefemph` governs how references via `\symref` (or `\symname`, or their short variants) are highlighted;

Doing `\symref{<symbol>}{<text>}` ultimately expands to `\symrefemph@uri{<text>}{<symbol URI>}`, the default implementation of which is just `\symrefemph{<text>}`. The default implementation of `\symrefemph`, in turn, is just `\emph{<text>}`.

If you only want to change e.g. the color of `\symrefs`, you only need to redefine `\symrefemph`, e.g. using

```
\renewcommand\symrefemph[1]{\textcolor{red}{#1}}
```

the `@uri` variant is useful if you want to link somewhere, or show the URI in a tooltip. The stex-highlighting package does both, using:

```
\usepackage{pdfcomment}
\protected\def\symrefemph@uri#1#2{%
  \pdftooltip{%
    \srefsymuri{#2}{\symrefemph{#1}}%
  }%
  URI:~\detokenize{#2}%
}%
```

```

}
\def\symrefemph#1{%
  \ifcsname textcolor\endcsname
    \textcolor{teal}{#1}%
  \else#1\fi
}

```

---

`\compemph` Like `\symrefemph` and `\symrefemph@uri`, but governs the highlighting of components (marked with `\comp` or `\maincomp`) in `notations`.

---

`\defemph` Like `\symrefemph` and `\symrefemph@uri`, but governs the highlighting of definienda marked with `\definiendum` (or `\definename`).

---

`\varemph` Like `\compemph` and `\compemph@uri`, but governs the highlighting of components (marked with `\comp` or `\maincomp`) in the `notations` of `variables`.  
The second argument to `\varemph@uri` is the *name* of the `variable`.

## 9.2 Styling Environments and Macros

A variety of `environments` and `macros` provided by `STEX` are *stylistable* using the `macros` `\stextstyle{name}[\langle style\rangle]{...}`. These stylable `environments` and `macros` bind various of their parameters to `macros` `\this{parameter}`, which can then be used in the styles.

For example, if we have a `definition environment` that we would want to use to style our `sdefinitions`, we can do (in the simplest case)

```
\stextstyledefinition{\begin{definition}}{\end{definition}}
```

This tells `STEX` to insert `\begin{definition}` at the beginning of every `sdefinition environment`, and `\end{definition}` at the end.

If we have a `environments theorem` and `lemma`, we probably want the `sassertion environment` to use those for theorems and lemma. We can achieve that by doing

```
\stextstyleassertion[theorem]{\begin{theorem}}{\end{theorem}}
\stextstyleassertion[lemma]{\begin{lemma}}{\end{lemma}}
```

Now if we do `\begin{sassertion}[style=theorem]`, it will wrap the `environment` with `\begin{theorem}... \end{theorem}`.

Of course, many such statements might have a title, as e.g. in

**Satz 9.2.1** (Gödel's First Incompleteness Theorem). ...

In `sassertion`, we can provide that title as optional argument using `title=....` Before calling the styling provided, `sassertion` will store that title in the macro `\thistitle`, which we can use in the styling. For example, we might prefer to pass it on to the `theorem environment`:

```
\stextstyleassertion[theorem]{\ifx\thistitle\empty
  \begin{theorem}\else\begin{theorem}[\thistitle]\fi\end{theorem}}
```

---

```
\stexstylemodule           TODO15
```

```
\stexstylecopymodule
\stexstyleinterpretmodule
\stexstylerealization
\stexstylemathstructure
\stexstyleextstructure
\stexstyledefinition
\stexstyleassertion
\stexstyleexample
\stexstyleparagraph
\stexstyleproof
\stexstylesubproof
```

---

Additionally, we can style certain [macros](#), if we want them to produce output. For example, we might (for debuggin or documentation purposes) `\symdecl` to give a short summary of the symbol.

We can achieve that by doing, for example:

```
\stexstylesymdecl[debug]{
    Symbol \thisdeclname~(with arity \thisargs) of type \$\thistype$.
```

in which case

```
\symdecl{foo}[args=2,type=\mathbb{N},style=debug]
```

will yield

```
Symbol foo (with arity ii) of type N.
```

---

```
\stexstyleusemodule           TODO16
```

```
\stexstyleimportmodule
\stexstylerequiremodule
\stexstyleassign
\stexstylerenamedecl
\stexstyleassignMorphism
\stexstylecopymod
\stexstyleinterpretmod
\stexstylerealize
\stexstylesymdecl
\stexstyleextsymdecl
\stexstylenotation
\stexstylevarnotation
\stexstylesymdef
\stexstylevardef
\stexstylevarseq
\stexstylepsfsketch
\stexstyleMMTinclude
```

---

### 9.3 Custom CSS for Environments

# Chapter 10

## Additional Packages

### 10.1 NotesSlides Manual

#### 10.1.1 Introduction

The `notesslides` document class is derived from `beamer.cls` [`beamerclass:on`], it adds a “notes version” for course notes that is more suited to printing than the one supplied by `beamer.cls`.

The `notesslides` class takes the notion of a slide frame from Till Tantau’s excellent `beamer` class and adapts its notion of frames for use in the `STEX` and OMDoc. To support semantic course notes, it extends the notion of mixing frames and explanatory text, but rather than treating the frames as images (or integrating their contents into the flowing text), the `notesslides` package displays the slides as such in the course notes to give students a visual anchor into the slide presentation in the course (and to distinguish the different writing styles in slides and course notes).

In practice we want to generate two documents from the same source: the slides for presentation in the lecture and the course notes as a narrative document for home study. To achieve this, the `notesslides` class has two modes: *slides mode* and *notes mode* which are determined by the package option.

#### 10.1.2 Package Options

The `notesslides` class takes a variety of class options:

---

`slides` The options `slides` and `notes` switch between slides mode and notes mode (see ??).  
`notes`

---

`sectocframes` If the option `sectocframes` is given, then for the `sfragments`, special frames with the `sfragment` title (and number) are generated.

---

`frameimages` If the option `frameimages` is set, then slide mode also shows the `\frameimage`-generated  
`fiboxed` frames (see ??). If also the `fiboxed` option is given, the slides are surrounded by a box.

### 10.1.3 Notes and Slides

**frame** (*env.*) Slides are represented with the **frame** environment just like in the **beamer** class, see [Tantau:ugbc] for details.

**note** (*env.*) The **notesslides** class adds the **note** environment for encapsulating the course note fragments.



Note that it is essential to start and end the **notes** environment at the start of the line – in particular, there may not be leading blanks – else L<sup>A</sup>T<sub>E</sub>X becomes confused and throws error messages that are difficult to decipher.

By interleaving the **frame** and **note** environments, we can build course notes as shown here:

```
1 \ifnotes\maketitle\else
2 \frame[noframenumbering]\maketitle\fi
3
4 \begin{note}
5   We start this course with ...
6 \end{note}
7
8 \begin{frame}
9   \frametitle{The first slide}
10 ...
11 \end{frame}
12 \begin{note}
13   ... and more explanatory text
14 \end{note}
15
16 \begin{frame}
17   \frametitle{The second slide}
18 ...
19 \end{frame}
20 ...
```

---

**\ifnotes** Note the use of the **\ifnotes** conditional, which allows different treatment between **notes** and **slides** mode – manually setting **\notestrue** or **\notesfalse** is strongly discouraged however.



We need to give the title frame the **noframenumbering** option so that the frame numbering is kept in sync between the slides and the course notes.



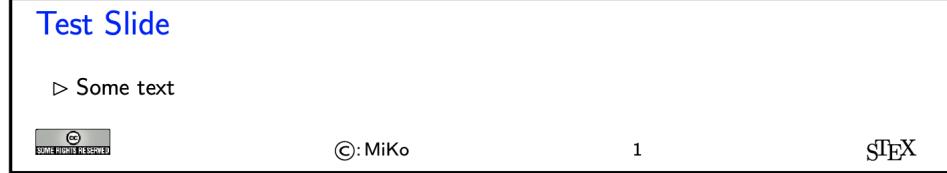
The **beamer** class recommends not to use the **allowframebreaks** option on frames (even though it is very convenient). This holds even more in the **notesslides** case: At least in conjunction with **\newpage**, frame numbering behaves funny (we have tried to fix this, but who knows).

**\inputref\*** If we want to transclude a the contents of a file as a note, we can use a new variant `\inputref*` of the `\inputref` macro: `\inputref*{foo}` is equivalent to `\begin{note}\inputref{foo}\end{note}`.

**nparagraph (env.)** There are some environments that tend to occur at the top-level of `note` environments.  
**nparagraph (env.)** We make convenience versions of these: e.g. the `nparagraph` environment is just an  
**ndefinition (env.)** `sparagraph` inside a `note` environment (but looks nicer in the source, since it avoids one  
**nexample (env.)** level of source indenting). Similarly, we have the `nfragment`, `ndefinition`, `nexample`,  
**nsproof (env.)** `nsproof`, and `nassertion` environments.  
**nassertion (env.)**

#### 10.1.4 Customizing Header and Footer Lines

The `notesslides` package and class comes with a simple default theme named `sTeX` that provided by the `beamterthemesTeX`. It is assumed as the default theme for `sTeX`-based notes and slides. The result in `notes` mode (which is like the `slides` version except that the slide height is variable) is



The footer line can be customized. In particular the logos.

**\setslidelogo** The default logo provided by the `notesslides` package is the `sTeX` logo it can be customized using `\setslidelogo{\langle logo name \rangle}`.

**\setsource** The default footer line of the `notesslides` package mentions copyright and licensing. In `notesslides` `\source` stores the author's name as the copyright holder. By default it is the author's name as defined in the `\author` macro in the preamble. `\setsource{\langle name \rangle}` can change the writer's name.

**\setlicensing** For licensing, we use the Creative Commons Attribution-ShareAlike license by default to strengthen the public domain. If package `hyperref` is loaded, then we can attach a hyperlink to the license logo. `\setlicensing[\langle url \rangle]{\langle logo name \rangle}` is used for customization, where `\langle url \rangle` is optional.

#### 10.1.5 Frame Images

Sometimes, we want to integrate slides as images after all – e.g. because we already have a PowerPoint presentation, to which we want to add `sTeX` notes.

---

```
\frameimage  
\mhframeimage
```

In this case we can use `\frameimage[<opt>]{<path>}`, where `<opt>` are the options of `\includegraphics` from the `graphicx` package [CarRah:tpp99] and `<path>` is the file path (extension can be left off like in `\includegraphics`). We have added the `label` key that allows to give a frame label that can be referenced like a regular `beamer` frame.

The `\mhframeimage` macro is a variant of `\frameimage` with repository support. Instead of writing

```
1 \frameimage{\MathHub{fooMH/bar/source/baz/foobar}}
```

we can simply write (assuming that `\MathHub` is defined as above)

```
1 \mhframeimage[fooMH/bar]{baz/foobar}
```

Note that the `\mhframeimage` form is more semantic, which allows more advanced document management features in `MathHub`.

If `baz/foobar` is the “current module”, i.e. if we are on the `MathHub` path `...MathHub/fooMH/bar...`, then stating the repository in the first optional argument is redundant, so we can just use

```
1 \mhframeimage{baz/foobar}
```

---

```
\textwarning
```

The `\textwarning` macro generates a warning sign: 

### 10.1.6 Ending Documents Prematurely

---

```
\prematurestop  
\afterprematurestop
```

For prematurely stopping the formatting of a document, S<sup>T</sup>E<sub>X</sub> provides the `\prematurestop` macro. It can be used everywhere in a document and ignores all input after that – backing out of the `sfragment` environments as needed. After that – and before the implicit `\end{document}` it calls the internal `\afterprematurestop`, which can be customized to do additional cleanup or e.g. print the bibliography.

`\prematurestop` is useful when one has a driver file, e.g. for a course taught multiple years and wants to generate course notes up to the current point in the lecture. Instead of commenting out the remaining parts, one can just move the `\prematurestop` macro. This is especially useful, if we need the rest of the file for processing, e.g. to generate a theory graph of the whole course with the already-covered parts marked up as an overview over the progress; see `import_graph.py` from the `lmhtools` utilities [lmhtools:github:on].

Text fragments and modules can be made more re-usable by the use of global variables. For instance, the admin section of a course can be made course-independent (and therefore re-usable) by using variables (actually token registers) `courseAcronym` and `courseTitle` instead of the text itself. The variables can then be set in the S<sup>T</sup>E<sub>X</sub> preamble of the course notes file.

### 10.1.7 Global Document Variables

To make document fragments more reusable, we sometimes want to make the content depend on the context. We use **document variables** for that.

---

`\setSGvar` `\setSGvar{<vname>}{<text>}` to set the global variable `<vname>` to `<text>` and `\useSGvar{<vname>}` to reference it.

---

`\ifSGvar` With `\ifSGvar` we can test for the contents of a global variable: the macro call `\ifSGvar{<vname>}{<val>}{<ctext>}` tests the content of the global variable `<vname>`, only if (after expansion) it is equal to `<val>`, the conditional text `<ctext>` is formatted.

### 10.1.8 Excursions

In course notes, we sometimes want to point to an “excursion” – material that is either presupposed or tangential to the course at the moment – e.g. in an appendix. The typical setup is the following:

```
1 \excursion{founif}{../fragments/founif.en}
2 {We will cover first-order unification in}
3 ...
4 \begin{appendix}\printexcursions\end{appendix}
```

It generates a paragraph that references the excursion whose source is in the file `../fragments/founif.en.tex` and automatically books the file for the `\printexcursions` command that is used here to put it into the appendix. We will look at the mechanics now.

---

`\excursion` The `\excursion{<ref>}{<path>}{<text>}` is syntactic sugar for

```
1 \begin{nparagraph}[title=Excursion]
2 \activateexcursion{founif}{../ex/founif}
3 We will cover first-order unification in \sref{founif}.
4 \end{nparagraph}
```

---

`\activateexcursion` Here `\activateexcursion{<path>}` augments the `\printexcursions` macro by a call `\inputref{<path>}`. In this way, the `\printexcursions` macro (usually in the appendix) will collect up all excursions that are specified in the main text.

Sometimes, we want to reference – in an excursion – part of another. We can use `\excursionref{<label>}` for that.

---

`\excursiongroup` Finally, we usually want to put the excursions into an `sfragment` environment and add an introduction, therefore we provide the a variant of the `\printexcursions` macro: `\excursiongroup[id=<id>, intro=<path>]` is equivalent to

```
1 \begin{note}
2 \begin{sfragment}[id=<id>]{Excursions}
3 \inputref{<path>}
4 \printexcursions
5 \end{sfragment}
6 \end{note}
```



When option `book` which uses `\pagestyle{headings}` is given and semantic macros are given in the `sfragment` titles, then they sometimes are not defined by the time the heading is formatted. Need to look into how the headings are made. This is a problem of the underlying document-structure package.

Local Variables: mode: latex TeX-master: .../**stex-manual** End:

## 10.2 Problem Manual

### 10.2.1 Introduction

The `problem` package supplies an infrastructure that allows specify problem. Problems are text fragments that come with auxiliary functions: hints, notes, and solutions. Furthermore, we can specify how long the solution to a given problem is estimated to take and how many points will be awarded for a perfect solution.

Finally, the `problem` package facilitates the management of problems in small files, so that problems can be re-used in multiple environment.

### 10.2.2 Problems and Solutions

---

`solutions` The `problem` package takes the options `solutions` (should solutions be output?), `notes` (should the problem notes be presented?), `hints` (do we give the hints?), `gnotes` (do we show grading notes?), `pts` (do we display the points awarded for solving the problem?), `min` (do we display the estimated minutes for problem soling). If theses are specified, then the corresponding auxiliary parts of the problems are output, otherwise, they remain invisible.

`boxed` The `boxed` option specifies that problems should be formatted in framed boxes so that they are more visible in the text. Finally, the `test` option signifies that we are in a test situation, so this option does not show the solutions (of course), but leaves space for the students to solve them.

`problem (env.)` The main environment provided by the `problem` package is (surprise surprise) the `problem` environment. It is used to mark up problems and exercises. The environment takes an optional KeyVal argument with the keys `id` as an identifier that can be reference later, `pts` for the points to be gained from this exercise in homework or quiz situations, `min` for the estimated minutes needed to solve the problem, and finally `title` for an informative title of the problem.

#### Example 26

Input:

```

1 \documentclass{article}
2 \usepackage[solutions,hints,pts,min]{problem}
3 \begin{document}
4 \begin{sproblem}[id=elefants,pts=10,min=2,title=Fitting Elefants]
5 How many Elefants can you fit into a Volkswagen beetle?
6 \begin{hint}
7 Think positively, this is simple!
8 \end{hint}
9 \begin{exnote}
10 Justify your answer
11 \end{exnote}
12 \begin{solution}
13 Four, two in the front seats, and two in the back.
14 \begin{gnote}
15 if they do not give the justification deduct 5 pts
16 \end{gnote}
17 \end{solution}
18 \end{sproblem}
19 \end{document}

```

Output:

### Exercise (Fitting Elefants)

How many Elefants can you fit into a Volkswagen beetle?

*Lösung:* Four, two in the front seats, and two in the back.

**solution (env.)** The **solution** environment can be used to specify a solution to a problem. If the package option **solutions** is set or **\solutionstrue** is set in the text, then the solution will be presented in the output. The **solution** environment takes an optional KeyVal argument with the keys **id** for an identifier that can be referenced for to specify which problem this is a solution for, and **height** that allows to specify the amount of space to be left in test situations (i.e. if the **test** option is set in the **\usepackage** statement).

**hint (env.)** The **hint** and **exnote** environments can be used in a **problem** environment to give hints  
**exnote (env.)** and to make notes that elaborate certain aspects of the problem. The **gnote** (grading  
**gnote (env.)** notes) environment can be used to document situations that may arise in grading.

---

**\start solutions** Sometimes we would like to locally override the **solutions** option we have given to  
**\stop solutions** the package. To turn on solutions we use the **\start solutions**, to turn them off,  
**\stop solutions**. These two can be used at any point in the documents.

---

**\if solutions** Also, sometimes, we want content (e.g. in an exam with master solutions) conditional  
on whether solutions are shown. This can be done with the **\if solutions** conditional.

### 10.2.3 Markup for Added-Value Services

The `problem` package is all about specifying the meaning of the various moving parts of practice/exam problems. The motivation for the additional markup is that we can base added-value services from these, for instance auto-grading and immediate feedback.

The simplest example of this are multiple-choice problems, where the `problem` package allows to annotate answer options with the intended values and possibly feedback that can be delivered to the users in an interactive setting. In this section we will give some infrastructure for these, we expect that this will grow over time.

#### Multiple Choice Blocks

`mcb (env.)` Multiple choice blocks can be formatted using the `mcb` environment, in which single choices are marked up with `\mcc` macro.

---

`\mcc` `\mcc[<keyvals>]{<text>}` takes an optional key/value argument `<keyvals>` for choice metadata and a required argument `<text>` for the proposed answer text. The following keys are supported

- `T` for true answers, `F` for false ones,
- `Ttext` the verdict for true answers, `Ftext` for false ones, and
- `feedback` for a short feedback text given to the student.

What we see when this is formatted to PDF depends on the context. In solutions mode (we start the solutions in the code fragment below) we get

#### Example 27

Input:

```
1 \startSolutions
2 \begin{sproblem}[title=Functions,name=functions1]
3 What is the keyword to introduce a function definition in python?
4 \begin{mcb}
5   \mcc[T]{def}
6   \mcc[F,feedback=that is for C and C++]{function}
7   \mcc[F,feedback=that is for Standard ML]{fun}
8   \mcc[F,Ftext=Noooooooooooo,feedback=that is for Java]{public static void}
9 \end{mcb}
10 \end{sproblem}
```

Output:

### Exercise (Functions)

What is the keyword to introduce a function definition in python?

- def
- function<sup>a</sup>
- fun<sup>b</sup>
- public static void<sup>c</sup>

<sup>a</sup>

*that is for C and C++*

<sup>b</sup>

*that is for Standard ML*

<sup>c</sup>Noooooooooooo

*that is for Java*

In “exam mode” where disable solutions (here via `\stopsolutions`)

### Example 28

Input:

```
1 \stopsolutions
2 \begin{sproblem}[title=Functions,name=functions1]
3 What is the keyword to introduce a function definition in python?
4 \begin{mcb}
5   \mcc[T]{def}
6   \mcc[F,feedback=that is for C and C++]{function}
7   \mcc[F,feedback=that is for Standard ML]{fun}
8   \mcc[F,Ftext=Noooooooooooo,feedback=that is for Java]{public static void}
9 \end{mcb}
10 \end{sproblem}
```

Output:

### Exercise (Functions)

What is the keyword to introduce a function definition in python?

- def
- function
- fun
- public static void

we get the questions without solutions (that is what the students see during the exam/quiz).

### Filling-In Concrete Solutions

The next simplest situation, where we can implement auto-grading is the case where we have fill-in-the-blanks

**\fillinsol** The `\fillinsol` macro takes a single argument, which contains a concrete solution (i.e. a number, a string, ...), which generates a fill-in-box in test mode:

### Example 29

Input:

```
1 \stopsolutions
2 \begin{sproblem}[id=elefants.fillin,title=Fitting Elefants]
3 How many Elefants can you fit into a Volkswagen beetle? \fillinsol{4}
4 \end{sproblem}
```

#### Exercise (Fitting Elefants)

and How many Elefants can you fit into a Volkswagen beetle?

### Example 30

Input:

```
1 \begin{sproblem}[id=elefants.fillin,title=Fitting Elefants]
2 How many Elefants can you fit into a Volkswagen beetle? \fillinsol{4}
3 \end{sproblem}
```

Output:

#### Exercise (Fitting Elefants)

How many Elefants can you fit into a Volkswagen beetle?

If we do not want to leak information about the solution by the size of the blank we can also give `\fillinsol` an optional argument with a size: `\fillinsol[3cm]{12}` makes a box three cm wide.

Obviously, the required argument of `\fillinsol` can be used for auto-grading. For concrete data like numbers, this is immediate, for more complex data like strings “soft comparisons” might be in order.<sup>17</sup>

#### 10.2.4 Including Problems

**\includeproblem** The `\includeproblem` macro can be used to include a problem from another file. It takes an optional KeyVal argument and a second argument which is a path to the file containing the problem (the macro assumes that there is only one problem in the included file). The keys `title`, `min`, and `pts` specify the problem title, the estimated minutes for solving the problem and the points to be gained, and their values (if given) overwrite the ones specified in the `problem` environment in the included file.

<sup>17</sup>For the moment we only assume a single concrete value as correct. In the future we will almost certainly want to extend the functionality to multiple answer classes that allow different feedback like in MCQ. This still needs a bit of design. Also we want to make the formatting of the answer in solutions/test mode configurable.

The sum of the points and estimated minutes (that we specified in the `pts` and `min` keys to the `problem` environment or the `\includeproblem` macro) to the log file and the screen after each run. This is useful in preparing exams, where we want to make sure that the students can indeed solve the problems in an allotted time period.

The `\min` and `\pts` macros allow to specify (i.e. to print to the margin) the distribution of time and reward to parts of a problem, if the `pts` and `pts` options are set. This allows to give students hints about the estimated time and the points to be awarded.

### 10.2.5 Testing and Spacing

The `problem` package is often used by the `hwexam` package, which is used to create homework assignments and exams. Both of these have a “test mode” (invoked by the package option `test`), where certain information –master solutions or feedback – is not shown in the presentation.

```
\testspace      \testspace takes an argument that expands to a dimension, and leaves vertical space accordingly. Specific instances exist: \testsmallspace, \testsmallspace, \testsmallspace \testsmallspace give small (1cm), medium (2cm), and big (3cm) vertical space.
\testsmallspace \testnewpage makes a new page in test mode, and \testemptypage generates an empty page with the cautionary message that this page was intentionally left empty.
\testemptypage Local Variables: mode: latex TeX-master: .../stex-manual End:
LocalWords: solutions,notes,hints,gnotes,pts,min,boxed,test gnotes elefants,pts gnote LocalWords: 2,title exnote hint,exnote,gnote ifsolutions mcb keyvals Ttext Ftext LocalWords: Functions,name F,feedback Noooooooooo,feedback 2,title includeproblem LocalWords: elefants.fillin,title
```

## 10.3 HWExam Manual

### 10.3.1 Introduction

The `hwexam` package and class supplies an infrastructure that allows to format nice-looking assignment sheets by simply including problems from problem files marked up with the `problem` package. It is designed to be compatible with `problems.sty`, and inherits some of the functionality.

### 10.3.2 Package Options

---

|                        |                                                                                                                                                                                                                                                                                                                                                                      |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>solutions</code> | The <code>hwexam</code> package and class take the options <code>solutions</code> , <code>notes</code> , <code>hints</code> , <code>gnotes</code> , <code>pts</code> , <code>notes</code> <code>min</code> , and <code>boxed</code> that are just passed on to the <code>problems</code> package (cf. its documentation for a description of the intended behavior). |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

---

|                     |  |
|---------------------|--|
| <code>gnotes</code> |  |
|---------------------|--|

---

|                  |  |
|------------------|--|
| <code>pts</code> |  |
|------------------|--|

---

|                  |  |
|------------------|--|
| <code>min</code> |  |
|------------------|--|

---

|                       |                                                                                                                                                                                                                                                    |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>multiple</code> | Furthermore, the <code>hwexam</code> package takes the option <code>multiple</code> that allows to combine multiple assignment sheets into a compound document (the assignment sheets are treated as section, there is a table of contents, etc.). |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                            |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>test</code> | Finally, there is the option <code>test</code> that modifies the behavior to facilitate formatting tests. Only in <code>test</code> mode, the macros <code>\testspace</code> , <code>\testnewpage</code> , and <code>\testemptypage</code> |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

have an effect: they generate space for the students to solve the given problems. Thus they can be left in the L<sup>A</sup>T<sub>E</sub>X source.

### 10.3.3 Assignments

**assignment** (*env.*) This package supplies the **assignment** environment that groups problems into assignment **number** sheets. It takes an optional KeyVal argument with the keys **number** (for the assignment number; if none is given, 1 is assumed as the default or — in multi-assignment documents **title** — the ordinal of the **assignment** environment), **title** (for the assignment title; this is **type** referenced in the title of the assignment sheet), **type** (for the assignment type; e.g. “quiz”, **given** or “homework”), **given** (for the date the assignment was given), and **due** (for the date **due** the assignment is due).

### 10.3.4 Including Assignments

---

**\inputassignment** The **\inputassignment** macro can be used to input an assignment from another file. It takes an optional KeyVal argument and a second argument which is a path to the file containing the problem (the macro assumes that there is only one **assignment** environment in the included file). The keys **number**, **title**, **type**, **given**, and **due** are just as for the **assignment** environment and (if given) overwrite the ones specified in the **assignment** environment in the included file.

### 10.3.5 Typesetting Exams

**testheading** (*env.*) The **\testheading** takes an optional keyword argument where the keys **duration** specifies a string that specifies the duration of the test, **min** specifies the equivalent in number **min** of minutes, and **reqpts** the points that are required for a perfect grade.

```
reqpts1 \title{320101 General Computer Science (Fall 2010)}
2 \begin{testheading}[duration=one hour,min=60,reqpts=27]
3   Good luck to all students!
4 \end{testheading}
```

Will result in

Name:

Matriculation Number:

## 320101 General Computer Science (Fall 2010)

2024-11-20

**You have one hour (sharp) for the test;**

Write the solutions to the sheet.

The estimated time for solving this exam is 23 minutes, leaving you 37 minutes for revising your exam.

You can reach 23 points if you solve all problems. You will only need 27 points for a perfect score, i.e. -4 points are bonus points.

Different problems test different skills and knowledge, so do not get stuck on one problem.

|         | To be used for grading, do not write here |     |     |     |     |     |     |     |     |
|---------|-------------------------------------------|-----|-----|-----|-----|-----|-----|-----|-----|
| prob.   | 4.1                                       | 1.1 | 1.2 | 2.1 | 2.2 | 2.3 | 2.4 | 2.5 | 2.6 |
| total   | 0                                         | 0   | 0   | 10  | 0   | 0   | 0   | 0   | 0   |
| reached |                                           |     |     |     |     |     |     |     |     |

good luck

18

Local Variables: mode: latex TeX-master: `../stex-manual` End:

LocalWords: hwexam solutions,notes,hints,gnotes,pts,min gnotes testemptypage reqpts LocalWords: inputassignment reqpts hour,min 60,reqpts

### 10.4 Tikzinput Manual

---

**image** The behavior of the `ikzinput` package is determined by whether the `image` option is given. If it is not, then the `tikz` package is loaded, all other options are passed on to it and `\tikzinput{<file>}` inputs the TIKZ file `<file>.tex`; if not, only the `graphicx` package is loaded and `\tikzinput{<file>}` loads an image file `<file>.(<ext>)` generated from `<file>.tex`.

The selective input functionality of the `tikzinput` package assumes that the TIKZ pictures are externalized into a standalone picture file, such as the following one

---

<sup>18</sup>MK: The first three “problems” come from the stex examples above, how do we get rid of this?

```

1 \documentclass{standalone}
2 \usepackage{tikz}
3 \usetikzlibrary{...}
4 \begin{document}
5   \begin{tikzpicture}
6     ...
7   \end{tikzpicture}
8 \end{document}

```

The `standalone` class is a minimal L<sup>A</sup>T<sub>E</sub>X class that when loaded in a document that uses the `standalone` package: the preamble and the `document` environment are disregarded during loading, so they do not pose any problems. In effect, an `\input` of the file above only sees the `tikzpicture` environment, but the file itself is standalone in the sense that we can run L<sup>A</sup>T<sub>E</sub>X over it separately, e.g. for generating an image file from it.

---

**\tikzinput** This is exactly where the `tikzinput` package comes in: it supplies the `\tikzinput` macro, **\ctikzinput** which – depending on the `image` option – either directly inputs the TIKZ picture (source) or tries to load an image file generated from it.

Concretely, if the `image` option is not set for the `tikzinput` package, then `\tikzinput[<opt>]{<file>}` disregards the optional argument `<opt>` and inputs `<file>.tex` via `\input` and resizes it to as specified in the `width` and `height` keys. If it is, `\tikzinput[<opt>]{<file>}` expands to `\includegraphics[<opt>]{<file>}`.

`\ctikzinput` is a version of `\tikzinput` that is centered.

---

**\mhtikzinput** **\cmhtikzinput** `\mhtikzinput` is a variant of `\tikzinput` that treats its file path argument as a relative path in a math archive in analogy to `\inputref`. To give the archive path, we use the `mrepos=` key. Again, `\cmhtikzinput` is a version of `\mhtikzinput` that is centered.

---

**\libusetikzlibrary** Sometimes, we want to supply archive-specific TIKZ libraries in the `lib` folder of the archive or the `meta-inf/lib` of the archive group. Then we need an analogon to `\libinput` for `\usetikzlibrary`. The `stex-tikzinput` package provides the `libusetikzlibrary` for this purpose.

Local Variables: mode: latex TeX-master: `../stex-manual` End:

## **Part III**

## **Documentation**

# Chapter 11

## sTeX Developer Manual



Keeping the implementation properly up-to-date is pretty much incompatible with the kinds of workflows systemically enforced in academia. Any of the following may be out of date.

\* indicates fully expandable functions, which can be used within an e-type argument (inside an @\_cmd:nomodule=TeX,replace=false,\expanded), x-type argument (in plain TeX terms, inside an @\_cmd:nomodule=TeX,replace=false,\edef), as well as within an f-type argument. ☆ indicates restricted expandable functions, which can be used within an x-type argument or an e-type argument, but cannot be fully expanded within an f-type argument. TF indicates conditional (if) functions whose variants with T, F and TF argument specifiers expect different “true”/“false” branches.

---

`\stex_debug:nn` `\stex_debug:nn {⟨prefix⟩} {⟨msg⟩}`

Logs the debug message {⟨msg⟩} under the prefix {⟨prefix⟩}. A message is shown if its prefix is in a list of prefixes given either via the package option `debug=⟨prefixes⟩` or the environment variable `STEX_DEBUG=⟨prefixes⟩`, where the latter overrides the former.

---

`\_stex_do_deprecation:n` TODO `\_stex_do_deprecation:n`

## 11.1 Documents

`\l_stex_docheader_sect` integer register keeping track of the current sectioning level:

- 0 part
- 1 chapter
- 2 section
- 3 subsection
- 4 subsubsection
- 5 paragraph
- > 5 subparagraph

`\setsectionlevel` sets `\l_stex_docheader_sect` to the corresponding integer value.

`\stex_ref_new_doc_target:n` internal variant of `\sreflabel`. If the argument is empty, the label is determined to be `REF<counter>` and `<counter>` is increased.

`\stex_ref_new_symbol:n` registers a new link target for the symbol with the given full uri (as string), using the `url-base`-field of the current archive's manifest file to link the symbol to `<url-base>/symbol?<uri>`.

`\stex_ref_new_sym_target:n` sets a new label for the symbol with the given full uri (as string). If called in sms-mode, defers to `\stex_ref_new_symbol:n`, if not already registered. Otherwise, sets a `\label` for the symbol.

`\stex_ref_new_sym_target:nn`  
`\stex_ref_new_sym_target:nn {<symbol>} {<target>}`  
redirects links for `<symbol>` to the one for the symbol `<target>`. Useful for e.g. symbols elaborated from structural features. Note that this acts as a *default*, in that previous or subsequent calls of `\stex_ref_new_sym_target:n{<symbol>}` are prioritized.  
Requires that either `\stex_ref_new_sym_target:n{<target>}` or `\stex_ref_new_sym_target:nn{<target>}{<other-target>}` have been called previously.

## 11.2 Modules

The contents of a module with full URI `<uri>` are represented as four macros:

- `\c_stex_module_<uri>_code`: code to be executed every time the module is *activated*; e.g. the contents of `\STEXexport`, defines semantic macros and macros for notations, activates dependency modules, etc.

- `\c_stex_module_<uri>_morphisms_prop`: property list containing all module dependencies of this module (see `\stex_module_add_morphism:nnn`).
- `\c_stex_module_<uri>symbols_prop`: property list containing all declarations in this module (see `\stex_module_add_symbol:nnnnnnN`).
- `\c_stex_module_<uri>_notations_prop`: property list containing all notations introduced in this module (see `\stex_add_module_notation:nnnn`).

---

`\l_stex_current_module_str` contains the full URI of the current module.

---

`\l_stex_all_modules_seq` contains the full URIs of all modules currently in scope.

---

`\stex_module_setup:n \stex_module_setup:n {<name>}`

Computes the full URI of a new module with name `<name>` in the current namespace, initializes the four macros above and sets `\l_stex_current_module_str` accordingly. Also takes care of correct naming for nested modules, activates the meta theory and loads the signature module if `sig=` was provided (according to `\l_stex_key_sig_str`).

---

`\stex_close_module:` closes the current module; checks whether we are currently in sms mode, and if so, calls `\stex_persist:n` to write the module contents to the sms-file.

---

`\stex_every_module:n \stex_every_module:n {<code>}`

executes `<code>` every time a new module is opened.

---

`\stex_if_in_module_p: *` tests whether we are currently in a module.  
`\stex_if_in_module:TF *`

---

`\stex_if_module_exists_p:n *`  
`\stex_if_module_exists:nTF *`

tests whether a module with the given full URI exists, in the sense of *has been parsed at some point in the current document*.

---

`\stex_activate_module:n \stex_activate_module:(o|x)` activates the module with the given full URI *if and only if* it has not already been activated in the current scope.

---

`\stex_execute_in_module:n \stex_execute_in_module:x` executes the provided code, adds it to the current module activation code, and makes sure the macros defined in it are valid in the current module TeX group level.

This macro is a combination of the following two macros:

---

`\stex_do_up_to_module:n` executes the provided code such that all definitions in it are valid in the current module  
`\stex_do_up_to_module:x` regardless of TeX group level (using `\stex_metagroup_do_in:nn`).

---

`\stex_module_add_code:n` adds the provided code to the module's `\c_stex_module_<uri>_code`-macro.  
`\stex_module_add_code:x`

### 11.3 Symbols

A symbol in **STEX** is represented as a tuple of several components:

---

`\stex_module_add_symbol:nnnnnnnN`

- #1 :  $\langle id \rangle$ : An *identifier* (possibly empty) that determines its semantic macro name, or e.g. in `mathstructure` its accessor-identifier (if empty, the name is used for that),
- #2 :  $\langle name \rangle$  a unique *name*, which in combination with the containing module determines its URI as  $\langle module-URI \rangle ? \langle name \rangle$ ,
- #3 :  $\langle arity \rangle$  a numeric string in the range 0..9,
- #4 :  $\langle args \rangle$  a list of argument specifiers of the form  $\langle i \rangle \langle mode \rangle \{ \langle argname \rangle \}$  (the length of  $\langle args \rangle$  must be  $3 \cdot \langle arity \rangle$ ),
- #5 :  $\langle definiens \rangle$  (or empty),
- #6 :  $\langle type \rangle$  (or empty),
- #7 :  $\langle return code \rangle$  (or empty), and
- #8 :  $\langle \text{\invocation\_macro} \rangle$ .

the arguments are stored in the property list `\c_stex_module_<\l_stex_current_module>_symbols_prop` under key  $\langle name \rangle$ .

If the identifier  $\langle id \rangle$  is non-empty, the macro `\id` is defined as `\stex_invoke-symbol:nnnnnnnN` with the arguments described there.

**TeXhackers note:** `\invocation_macro` must be `\protected`.

---

`\stex_iterate_symbols:n`  
`\stex_iterate_break:`  
`\stex_iterate_break:n`

`\stex_iterate_symbols:n {<code>}`

iterates over all symbols currently in scope and calls  $\langle code \rangle$  for all of them with arguments  $\{ \langle module \rangle \} \{ \langle name \rangle \} \{ \langle arity \rangle \} \{ \langle args \rangle \} \{ \langle definiens \rangle \} \{ \langle type \rangle \} \{ \langle return code \rangle \} \{ \langle \text{\invocation\_macro} \rangle \}$ .

Iteration can be stopped prematurely with `\stex_iterate_break:` and can stop and return code with `\stex_iterate_break:n`.

---

`\stex_iterate_symbols:nn`

`\stex_iterate_symbols:nn {<cs1>} {<code>}`

iterates over all symbols in the provided  $\langle cs1 \rangle$  of full module URIs.  $\langle code \rangle$  receives the same arguments as `\stex_iterate_symbols:n`, but iteration can not be stopped early.

---

```
\stex_get_symbol:n  
\l_stex_get_symbol_mod_str  
\l_stex_get_symbol_name_str  
\l_stex_get_symbol_arity_int  
\l_stex_get_symbol_args_tl  
\l_stex_get_symbol_def_tl  
\l_stex_get_symbol_type_tl  
\l_stex_get_symbol_return_tl  
\l_stex_get_symbol_invoke_cs
```

---

`\stex_get_symbol:n` attempts to find a symbol with the given name or id that is currently in scope. A name may be prefixed with a module name/path separated by `?`.

Throws an error if no such symbol is found; otherwise, sets the listed `\l_stex_get_symbol_...` macros to the components of the found symbol.

---

```
\stex_if_check_terms_p: * whether to typeset declaration components (notations, types, definientia etc.) in a throwaway box. Is true iff the backend is pdflatex and either the STEX_CHECKTERMS environment variable or checkterms package option is set.
```

---

`\stex_check_term:n` typesets the argument in a throwaway box in math mode iff `\stex_if_check_terms:` is true.

Is deactivated in sms-mode.

---

```
\stex_symdecl_do:  
\l_stex_key_name_str  
\l_stex_key_args_str  
\l_stex_key_argnames_clist  
\l_stex_assoc_args_count  
\l_stex_argnames_seq
```

---

`\stex_symdecl_do:` processes the shared (mandatory and optional) arguments of e.g. `\symdecl`, `\symdef`, `\vardef` etc.

Requires the following macros to be set

- `\l_stex_key_name_str`: the name of the symbol,
- `\l_stex_key_args_str`: the `args`-string (e.g. `3` or `ai`)
- `\l_stex_key_argnames_clist`: a list of *names* for the arguments, the length of which should be  $\leq$  the arity of the symbol

and will generate the following macros:

- `\l_stex_get_symbol_arity_int`: the arity of the symbol,
- `\l_stex_key_args_str`: the args string as a definite sequence of argument-mode characters, whose length is the arity of the symbol; e.g. `3` is turned into `iii`,
- `\l_stex_assoc_args_count`: the number of sequence arguments (i.e. `a` or `B` mode),
- `\l_stex_argnames_seq`: the full sequence of argument names; those not provided by `\l_stex_key_argnames_clist` are set to be `$j`, where `j` is the index of the argument,
- `\l_stex_get_symbol_args_tl`: a token list of triples `j m {argname}`, where `j` is the index and `m` the respective argument mode character (i.e. `i`, `a`, `b` or `B`).

---

`\_stex_symdecl_check_terms:`

calls `\stex_check_term:n` for the type and definiens stored in `\l_stex_key_type_tl` and `\l_stex_key_def_tl`

---

`\stex_symdecl_top:n \stex_symdecl_top:n {{maybename}}`

checks whether `\l_stex_key_name_str` is empty, and if so, sets it to be `<maybename>`. Then calls `\stex_symdecl_do:` and `\_stex_symdecl_check_terms:`, writes the components to the HTML (if applicable) and adds the symbol to the current module with invocation macro `\stex_invoke_symbol:` and id/macroname `\l_stex_mroname_str`.

Variables work very similar to symbols, except that their declarations are local to the current TeX-group rather than the current module, and they are not exported in modules.

---

`\l_stex_variables_prop` stores all variables currently in scope as a property list with key `<name>` and value `{{id}}{{name}}{{arity}}{{args}}{{definiens}}{{type}}{{return code}}{{\invocation\_macro}}`.

The invocation macro for “normal” variables declared with `\vardef` is `\stex_invoke_symbol::`.

---

`\_stex_vardecl_notation_macro:`

generates the notation macro for a variable, based on the values of the `\l_stex_key`-macros and `\l_stex_notation_macrocode_cs`.

---

`\stex_get_var:n` like `\stex_get_symbol:n`, but attempts to retrieve a variables and throws an error if none is found.

---

`\stex_get_symbol_or_var:n` like `\stex_get_symbol:n`, but first attempts to find a *variable*, and if none is found, defers to `\stex_get_symbol:n`.

## 11.4 Notations

---

`\stex_module_add_notation:nnnnn \stex_module_add_notation:eoexo`  
`\stex_module_add_notation:nnnnn {{symboluri}}{{id}}{{arity}}{{code}}{{op code}}`

stores the arguments in the property list `\c_stex_module_{\l_stex_current_module}_notations_prop` under key `<symboluri>!<id>` and calls `\stex_set_notation_macro:nnnnn`.

---

```
\stex_set_notation_macro:nnnnn \stex_set_notation_macro:nnnnn
\stex_set_notation_macro:eoexo {symboluri}{{id}{{arity}{{code}}{op code}}
```

Declares the following macros:

An active notation for a symbol with uri *<symboluri>* and id *<id>* is represented as a macro `\l_stex_notation_<symboluri>_<id>_cs`, that takes *<arity>* many argument and expands to *<code>*, and (if nonempty) an *operator* notation as a macro `\l_stex_notation_<symboluri>_op_<id>_cs` that expands to *<op code>*.

The default notation is represented as the *empty* id. If the corresponding macros `\l_stex_notation_<symboluri>_<id>_cs`, and `\l_stex_notation_<symboluri>_op_<id>_cs` do not yet exist, they are now defined as *<id>*.

---

```
\stex_iterate_notations:nn \stex_iterate_notations:nn {csl}{{code}}
```

iterates over all notations in the provided *<csl>* of full module URIs and calls *<code>* on each of them with arguments {*symboluri*}{{*id*}{{*arity*}{{*code*}}{*op code*}}.

---

```
\stex_notation_parse_and_then:nw \stex_notation_parse_and_then:nw {code}{{notations}}
\l_stex_key_prec_str
\l_stex_key_variant_str
\l_stex_notation_macrocode_cs
```

parses a notation (which may consist of multiple braced components, depending on the argument modes) and subsequently calls *<code>*.

Requires the following macros to be set:

- `\l_stex_get_symbol_arity_int`,
- `\l_stex_get_symbol_args_tl`,
- `\l_stex_key_prec_str`: The precedence string as provided by a user in the optional `precs`-argument,
- `\l_stex_key_variant_str`: the id of the notation variant.

Stores the final notation in the macro `\l_stex_notation_macrocode_cs` taking `\l_stex_get_symbol_arity_int` many arguments.

Some thing to consider when we generate a notation macro:

- The notation macro generated by the `\notation`-command should contain the variant identifier and the precedences, as these are intrinsic parts of the notation.
- It should *not* contain the symbol itself however, so that notations can be copied between symbols.
- Notations as provided by users will largely adhere to the standard L<sup>A</sup>T<sub>E</sub>X category code scheme, and
- notations need to be “persistable” in .deps-files.

We therefore want to augment the simple code provided by a user by various “annotations” that contain the relevant information (such as precedences) and to mark the

argument positions for semantic extractions, but we should adhere to the default L<sup>A</sup>T<sub>E</sub>X category code scheme in doing so.

---

```
\STEXInternalTermMathArgiii
\STEXInternalTermMathAssocArgiiii
\STEXInternalAssocArgMarkerI
\STEXInternalAssocArgMarkerII
\STEXInternalTermMathOMSOrOMViii
\STEXInternalTermMathOMAiii
\STEXInternalTermMathOMBiii
\STEXInternalSymbolAfterInvocationTL
```

---

In OPENMATH/OMDOC, there are (for our purposes) three kinds of expressions that an application of a semantic macro – and hence a notation macro – can represent, each of which corresponds to a macro taking care of the semantic annotations:

- OMS/OMV: a simple symbol (arity 0) (`\STEXInternalTermMathOMSOrOMViii`)
- OMA: an application of a symbol to argument (arity > 0, `\STEXInternalTermMathOMAiii`)
- OMB: a binding application of a symbol that binds/declares one or more variables (argument string contains b or B, `\STEXInternalTermMathOMBiii`).

The arguments are marked with `\STEXInternalTermMathArgiii` (i or b) or `\STEXInternalTermMathAssocArgiiii` (a or B). Finally, the notation is closed with `\STEXInternalSymbolAfterInvocationTL`.

How this works is best explained by example.

### Example 31

Assume we have a symbol and notation:

```
1 \symdecl{someSymbol}[args=iai]
2 \notation{someSymbol}[prec=10;20x30x40,variant=foo]
3 {First: #1; Second: #2; Third: #3; End}
4 {(#1 -- ##1 split ##2 -- #3)}
```

Since the symbol corresponds to an OMA, the whole notation is wrapped in `\STEXInternalTermMathOMAiii`, taking as arguments the variant identifier (foo), the operator precedence (10) and the body of the notation.

The second argument in the notation, being associative, is wrapped in a `\STEXInternalTermMathAssocArgiiii`, taking as arguments the argument number (2), the precedence (30), the T<sub>E</sub>X parameter token (#2) the notation body ((#1 -- ##1 split ##2 -- #3)), and finally the argument mode (a). Additionally, the markers ##1 and ##2 are replaced by `\STEXInternalAssocArgMarkerI` and `\STEXInternalAssocArgMarkerII`, respectively.

Subsequently, the non-sequence parameter tokens are wrapped in `\STEXInternalTermMathArgiii` with arguments mj (where m is the mode und j the index), the precedence (20 or 40 respectively), and the parameter token.

Finally, a `\STEXInternalSymbolAfterInvocationTL` is inserted.

The final expansion of `\l_stex_notation_macrocode_cs` is thus:

```
1 \STEXInternalTermMathOMAiii{foo}{10}{
2   First: \STEXInternalTermMathArgiii{i1}{20}{#1};
3   Second: \STEXInternalTermMathAssocArgiiii{2}{30}{#2}{
4     (\STEXInternalTermMathArgiii{i1}{20}{#1} --
```

```

5      \STEXInternalAssocArgMarkerI split
6      \STEXInternalAssocArgMarkerII --
7      \STEXInternalTermMathArgiii{i3}{40}{##3}
8  }{a};
9  Third: \STEXInternalTermMathArgiii {i3}{40}{#3}; End
10 }\STEXInternalSymbolAfterInvocationTL

```

---

`\stex_notation_top:nw` `\stex_notation_top:nw {<symboluri>}{<code>}`

calls `\stex_notation_parse_and_then:nw{<code>}` and, adds the notation for the symbol with URI `<symboluri>` to the current module and exports it to the HTML (if applicable).

## 11.5 Structural Features

---

`\stex_structural_feature_module:nn` `\stex_structural_feature_module:nn {<name>}{<typeid>}`

`\stex_structural_feature_module_end:`

`\g_stex_last_feature_str`

opens a new module-like structural feature of type `<typeid>` with name `<name>`, which needs to be closed with `\stex_structural_feature_module_end:`.

Its body behaves like a nested module with name `<modulename>/<name>`, the full URI of which is stored in `\g_stex_last_feature_str` for subsequent elaboration.

---

```
\stex_structural_feature_morphism:nnnnn
\stex_structural_feature_morphism_end: \stex_structural_feature_morphism:nnnnn
{\morphisname}{\typeid}{\archive}{\domain}{\annotations}
\l_stex_current_domain_str
\l_stex_feature_name_str
\l_stex_morphism_symbols_prop
\l_stex_morphism_renames_prop
\l_stex_morphism_morphisms_seq
```

---

opens a new morphism-like structural feature of type  $\langle typeid \rangle$  with name  $\langle morphisname \rangle$  and the module  $[\langle archive \rangle]{\langle domain \rangle}$  as domain, which needs to be closed with `\stex_structural_feature_morphism_end:`.

Deactivates `\symdecl`, `\textsymdecl`, `\symdef`, `\notation` and `smodule`, and activates `\assign`, `\assignMorphism` and `\renamedecl`.

Defines the following macros:

- `\l_stex_feature_name_str ={\langle name \rangle}.`
- `\l_stex_current_domain_str` = the full uri of  $\langle domain \rangle$ .
- `\l_stex_morphism_symbols_prop`: This property list is initialized as follows: For every symbol transitively included in  $\langle domain \rangle$  with data  $\langle module \rangle$ ,  $\langle name \rangle$ ,  $\langle id \rangle$ ,  $\langle arity \rangle$ ,  $\langle args \rangle$ ,  $\langle definiens \rangle$ ,  $\langle type \rangle$ , and  $\langle return code \rangle$ , the property list contains an entry with key  $[\langle module \rangle]/[\langle name \rangle]$  and value  $\{\langle id \rangle\}{\langle arity \rangle}{\langle args \rangle}{\langle definiens \rangle}{\langle type \rangle}{\langle return code \rangle}$ .
- `\l_stex_morphism_renames_prop`: An initially empty property list.
- `\l_stex_morphism_morphisms_seq`: TODO

At `\stex_structural_feature_morphism_end:`, the elaboration is computed from the above data thusly:

For every entry in `\l_stex_morphism_symbols_prop`, a new symbol is created with the values  $\langle arity \rangle$ ,  $\langle args \rangle$ ,  $\langle definiens \rangle$ ,  $\langle type \rangle$  and  $\langle return code \rangle$  from that property list, and either:

- if `\l_stex_morphism_renames_prop` does *not* contain an entry with key  $\langle module \rangle?{\langle name \rangle}$ , then the elaborated name is  $\langle morphisname \rangle/{\langle name \rangle}$  and its  $\langle id \rangle$  is empty (no semantic macro is generated), or
- if `\l_stex_morphism_renames_prop` contains an entry with key  $\langle module \rangle?{\langle name \rangle}$ , then its value needs to be of the form  $\{\langle id \rangle\}{\langle name \rangle}$ , which are used for the elaborated symbol.

All notations of the symbols transitively included in the domain are copied over to their elaborations.

## 11.6 Imports and Morphisms

---

```
\stex_module_add_morphism:nnnn
\stex_module_add_morphism:(nonn|ooox)
```

adds a new module morphism (i.e. inheritance, possibly with modification) to the current module

#1 : The name of the morphism (may be empty for e.g. `\importmodule`, but may be named for e.g. `copymodule`),

#2 : the URI of the module being “imported”,

#3 : the “type” of the morphism (e.g. `import` or `copymodule`),

#4 : a list of assignments as pairs  $\{\langle source \rangle\} \{\langle target \rangle\}$  that signify that the symbol with full URI  $\langle source \rangle$  is being mapped or elaborated to the new symbol with name  $\langle target \rangle$  in the current module. May be empty for e.g. `\importmodule`.

The provided arguments are stored in `\c_stex_module_<uri>_morphisms_prop` with key #1 (if non-empty) or [#2] (if #1 is empty) and value {#1}{#2}{#3}{#4}

Import-like statements in STEX are usually given as pairs  $[\langle archive \rangle] \{\langle path \rangle? \langle module \rangle\}$ , which be relative to the current archive and/or file path. For persistence and sms-mode, these pairs first need to be resolved into an *absolute* specification:

---

```
\stex_import_module_uri:nn
\l_stex_import_archive_str
\l_stex_import_name_str
\l_stex_import_uri_str
\l_stex_import_path_str
```

```
\stex_import_module_uri:nn {\langle archive \rangle} {\langle pathstring \rangle}
```

(where  $\langle archive \rangle$  may be empty) resolves the given arguments into:

- `\l_stex_import_archive_str` the given archive id (in which case `\stex_require_archive:n` is called), or the id of the current archive (if existent and  $\langle archive \rangle$  empty), or empty,
- `\l_stex_import_uri_str` if an archive id is given, or we currently are in an archive, its corresponding namespace; otherwise `{file:}`,
- `\l_stex_import_path_str` the path of the URI relative to the given (or current) archive, or (if not existent) the absolute path of  $\langle pathstring \rangle$  (without a module name) resolved relative to the current file’s parent directory, and
- `\l_stex_import_name_str` the name of the module.

If  $\langle pathstring \rangle$  does not contain the character ?, the whole pathstring is assumed to be the name of the module and the path is empty (or the current file’s parent directory, depending on the above).

---

```
\stex_import_require_module:nnn
```

takes as arguments values `\l_stex_import_archive_str`, `\l_stex_import_path_str` and `\l_stex_import_name_str` as computed by `\stex_import_module_uri:nn` and (optionally loads and) activates the corresponding module (or throws an error if it does not exist).

Most complex morphisms (`copymodule` et al) are implemented as structural features using `\stex_structural_feature_morphism:nnnn`.

---

```
\stex_get_in_morphism:n
\l_stex_get_symbol_macro_str
```

---

finds a symbol with the provided name or id in the domain of the current morphism. Sets the same macros as `\stex_get_symbol:n`, and additionally `\l_stex_get_symbol_macro_str` to the `<id>` of the symbol. Throws an error if no such symbol is found.

## 11.7 Expressions and Semantic Macros

---

```
\_stex_invoke_symbol:nnnnnnnN \l_stex_current_symbol_str \_stex_invoke_symbol:nnnnnnnN
\l_stex_current_arity_str {\langle module\rangle}{\langle name\rangle}{\langle arity\rangle}{\langle args\rangle}{\langle definiens\rangle}
\l_stex_current_args_tl {\langle type\rangle}
\l_stex_current_return_tl {\langle return code\rangle}
\l_stex_current_type_tl {\langle invocation_macro\rangle}
```

---

is how a semantic macro is/should be defined. `\_stex_invoke_symbol:nnnnnnnN` first checks whether semantic macros are currently allowed, and throws an error if not. Otherwise, it sets the `\comp`-controlled highlighting to `\compemph@uri`, initializes `\STEXInternalSymbolAfterInvocationTL`, defines the following macros for all of its arguments, and subsequently calls the `\invocation_macro`:

- `\l_stex_current_symbol_str ={\langle module\rangle}{\langle name\rangle}`
- `\l_stex_current_arity_str ={\langle arity\rangle}`
- `\l_stex_current_args_tl ={\langle args\rangle}`
- `\l_stex_current_type_tl ={\langle type\rangle}`
- `\l_stex_current_return_tl ={\langle return code\rangle}`

The simplest example for an `\invocation_macro` is `\stex_invoke_symbol:`.

---

```
\_stex_invoke_variable:nnnnnnN
```

---

analogous to `\_stex_invoke_symbol:nnnnnnnN`, but for variables; sets the `\comp`-controlled highlighting to `\varemph@uri`.

---

`\stex_invoke_symbol:` branches based on the mode and following characters:

- If math, check next character:
  - ! operator; defer to operator notation
  - else defer to notation and check the value of `\l_stex_current_return_tl={\langle return\rangle}`.
    - \* If `<return>` is empty, call the notation macro,
    - \* otherwise, call `<return>{\stex_invoke_symbol!}`.
- If text:

---

```
\stex_invoke_sequence_range:  
\stex_invoke_sequence_in:
```

TODO

## 11.8 Optional (Key-Value) Argument Handling

LATEX3 is surprisingly weak when it comes to handling optional (key-val) arguments in such a manner that *only* the freshly set macros are defined, and to modularly build up sets of argument keys. The following macros attempt to fix that:

---

```
\stex_keys_define:nnnn  
\stex_keys_set:nn
```

`\stex_keys_define:nnnn {<id>}{{<setup code>}}  
{<keyval setup code>}{{<parents>}}`

Defines a set of keys and their allowed values with identifier `stex/<id>`, that inherits from the sets with identifiers in `<parents>`.

`\stex_keys_set:nn {<id>}{{<CSL>}}` first executes `<setup code>` (e.g. to empty the macros holding the values) and then sets the keys in set `<id>` with the values provided in `<CSL>`.

---

`\_stex_do_id:` should be called whenever a macro or environment has a label id, i.e. calls `\stex_keys_set:nn{id}{...}`, after the title has been typeset. Sets a `\label` by calling `\stex_ref_new_doc_target:n{<id>}`.

### Example 32

If we define a set of keys with:

```
1 \stex_keys_define:nnnn{archive file}{  
2   \str_clear:N \l_stex_key_archive_str  
3   \str_clear:N \l_stex_key_file_str  
4 }{  
5   archive .str_set_x:N = \l_stex_key_archive_str ,  
6   file .str_set_x:N = \l_stex_key_file_str  
7 }{id}
```

then calling `\stex_keys_set:nn{archive file}{id=foo,file=bar}` sets `\l_stex_key_file_str={bar}`, assures that `\l_stex_key_archive_str` is empty, and executes the code associated with `id`, i.e. it sets `\l_stex_key_id_str={foo}`.

## 11.9 Stylistable Commands and Environments

---

```
\stex_new_stylistable_cmd:nnnn \stex_new_stylistable_cmd:nnnn {<name>} {<arity>} {<code>}<br/> {<default>}
```

Creates a new macro `\<name>` with expansion `<code>` taking `<arity>` many arguments, that is customizable in presentation by a user by calling `\stexstyle{<name>}`. On calling `\stex_style_apply:` executes the presentation code provided by a user.

`<code>` should:

- Call `\stex_keys_set:nn{style}{...}` (or a keyset inheriting from `style`),
- set macros with prefix `\this...` that a user might want to use for presentation (e.g. `\thistitle`),
- call `\stex_style_apply:` at some point.

---

```
\stex_new_stylistable_env:nnnnnnn \stex_new_stylistable_env:nnnnnnn {<name>} {<arity>} {<begincode>}<br/> {<endcode>} {<default begin>} {<default end>} {<prefix>}
```

Like `\stex_new_stylistable_cmd:nnnn`, but defines a new environment `{<prefix>}{<name>}` stylistable via `\stexstyle{<name>}`. Should call `\stex_style_apply:` twice; once in the `<begincode>` and once in `<endcode>`.

---

`\stex_style_apply:` Sets `\thisstyle` to be the head of the CSL `\l_stex_key_style_clist` and checks whether a style for the current stylistable macro/environment has been defined; if not, executes the code for the default style.

### Example 33

`\importmodule` is defined something like the following:

```
1 \stex_new_stylistable_cmd:nnnn{importmodule}{0{} m}{<br/> 2 ...<br/> 3 \def\thismoduleuri{...}<br/> 4 \def\thismodulename{...}<br/> 5 \stex_style_apply:<br/> 6 ...<br/> 7 }{}
```

A user can then customize the output generated by `\importmodule` (by default none) via `\stexstyle{importmodule}{...}{\thismodulename{...}}`.

### Example 34

`\smodule` does something like

```
1 \stex_new_stylistable_env:nnnnnnn{module}{0{} m}{<br/> 2 ...<br/> 3 \def\thismoduleuri{...}<br/> 4 \def\thismodulename{...}<br/> 5 \stex_style_apply:<br/> 6 ...<br/> 7 }{<br/> 8 ...}
```

```
9  \stex_style_apply:  
10 ...  
11 }{}{s}
```

which defines the environment name to be `smodule` and generates `\stextylemodule`.

## 11.10 Math Archives

Math archives are represented as L<sup>A</sup>T<sub>E</sub>X3 property lists, the keys/values of which correspond to the entries in the archive's manifest file. The most important fields are

- `id`,
- `narr` the document namespace,
- `ns` the content namespace, and
- `docurl` the document URL base.

---

`\stex_resolve_path_pair:Nnn`  
`\stex_resolve_path_pair:Nxx`

`\stex_resolve_path_pair:Nnn {<\target>}{{archive-id}}{<pathstring>}`

computes the absolute file path of `<pathstring>` relative to the source-folder of `<archive-id>` (if non-empty), or the current archive (if existent) or the parent working directory (otherwise), and stores the result in `\target`.

---

`\l_stex_current_archive_prop`

`\l_stex_current_archive_prop` always points to the current math archive or is `\undefined`, if the current file is not part of a math archive.

---

`\c_stex_main_archive_prop`

`\c_stex_main_archive_prop` represents the math archive in which the main file resides (if existent).

---

`\stex_require_archive:n`  
`\stex_require_archive:o`

`\stex_require_archive:n {<id>}`

looks for a math archive `<id>` in the MathHub directory, parses its manifest file, creates the corresponding property list `\c_stex_mathhub_<id>_manifest_prop`, and throws a fatal error if the archive is not found.

If the archive has been found and parsed before, does nothing, so it is cheap and safe to call repeatedly for the same id.

---

`\stex_set_current_archive:n`

`\stex_set_current_archive:n {<id>}`

Calls `\stex_require_archive:n{<id>}` and sets `\l_stex_current_archive_prop`.

---

`\stex_in_archive:nn`

`\stex_in_archive:nn {<opt-id>}{{code}}`

Executes `<code>` in the context of math archive `<opt-id>` (using `\stex_require_archive:n`), i.e. iff `<opt-id>` is non-empty, changes the current archive to the one with id `<opt-id>`, call `<code>` with `<opt-id>` as argument (in #1) and changes it back afterwards.

If `<opt-id>` is empty, `<code>` is called with the id of the *current* math archive as #1, or with #1 empty if there is no current math archive.

## 11.11 SMS-Mode

**STEX** has to extract formal content (i.e. modules and their symbols) from LATEX-files, that may otherwise contain arbitrary code, including macros that may not be defined unless the file is fully processed by TEX. Those modules and symbols also may depend on other modules that have not yet been loaded. The naive way to achieve this, which would be to just suppress output (e.g. by storing it in a box register) and then \input the required file, does not work thanks to TEX's limited *file stack*, which would overflow quickly for modules that have a deeply nested list of dependencies.

To solve those problems, STEX reads dependencies in what we call *sms mode*, which can be summarized thusly:

- In a first pass, we parse the file token by token, ignoring everything other than a select list of macros and environments that introduce dependencies (such as \importmodule and \begin{smodule}[sig=...]). Instead of loading those, we remember them for later.
- After the file has been fully parsed thusly, the dependencies found are loaded, again in sms-mode.
- In a second pass, we parse the file *again* in the same way, but this time execute all macros that are explicitly allowed in sms mode, such as \importmodule, \symdecl, \notation, \symdef, etc.
- all this parsing happens additionally in a \setbox\throwaway\vbox{...}-block to suppress any accidental output.

This means that TEX's input stack never grows by more than +1, but still behaves *as if* the dependencies were loaded recursively, at the detriment of being somewhat slow.

---

\stex\_if\_smsmode\_p: \* tests for whether we are currently in sms-mode.  
\stex\_if\_smsmode:TF \*

---

\stex\_file\_in\_smsmode:nn  
\stex\_file\_in\_smsmode:on    \stex\_file\_in\_smsmode:nn {\<filestring>} {\<setup-code>}  
sets up sms-mode and internal grouping, calls \<setup-code> and subsequently processes the file \<filestring> in sms-mode as described above.

### 11.11.1 Second Pass

---

\stex\_sms\_allow:N  
              \stex\_sms\_allow:N {\<macro>}

registers the \macro-command to be allowed in sms mode.

This only works, if \macro takes no arguments and/or does not touch the subsequent tokens.

For macros taking arguments, we can use

---

```
\stex_sms_allow_escape:N \stex_sms_allow_escape:N {⟨\macro⟩}
```

registers the `\macro`-command to be allowed in sms mode.

If `\macro` is subsequently encountered in sms-mode, parsing is halted and `\macro` can process arguments as desired. It then needs to continue parsing manually though, by calling `\stex_smsmode_do:` as (usually) its last token.

---

```
\stex_sms_allow_env:n \stex_sms_allow_env:n {⟨envname⟩}
```

registers the environment `⟨envname⟩` to be allowed in sms mode.

As with `\stex_sms_allow_escape:N`, the `\begin{⟨envname⟩}` is escaped, hence the begin-code of the environment needs to call `\stex_smsmode_do:`. Since `\end{⟨envname⟩}` never takes arguments, it does not need to be escaped.

---

`\stex_smsmode_do:` continues with sms-mode-style parsing. Does nothing if not in sms-mode, and is therefore safe to be called “just in case”.

### 11.11.2 First Pass

---

```
\stex_sms_allow_import:Nn  
\stex_sms_allow_import_env:nn
```

behave like `\stex_sms_allow_escape:N` and `\stex_sms_allow_env:n` respectively, but the macro or environment provided is now allowed in the *first* pass of sms-mode.

This macro should process arguments, add content to `\g_stex_sms_import_code`, and finally call `\stex_smsmode_do:`.

The code provided in the *second* argument is called before the first pass of sms-mode, as to set up functionality for these macros. For example, `\importmodule` provides code that redefines `\importmodule` to store the identified dependency in `\g_stex_sms_import_code` instead of activating it directly.

---

`\g_stex_sms_import_code` is built up in the first pass of sms mode and called subsequently; before the second pass.  
Code in this token list should load and activate dependencies found in the first pass.

## 11.12 Strings, File Paths, URIs

---

```
\stex_str_if_starts_with_p:nn * \stex_str_if_starts_with:nn {⟨first⟩}{⟨second⟩}  
\stex_str_if_starts_with:nnTF *
```

Checks whether the string `⟨first⟩` starts with the string `⟨second⟩` (i.e. `⟨second⟩` is a prefix of `⟨first⟩`).

---

```
\stex_str_if_ends_with_p:nn * \stex_str_if_ends_with:nn {⟨first⟩}{⟨second⟩}  
\stex_str_if_ends_with:nnTF *
```

Checks whether the string `⟨first⟩` ends with the string `⟨second⟩` (i.e. `⟨second⟩` is a suffix of `⟨first⟩`).

### 11.12.1 File Paths

*File paths* are represented as L<sup>A</sup>T<sub>E</sub>X3 sequences. The following methods make sure to

- canonicalize paths, i.e. resolve .. and . segments,
- set all category codes to 12 (other), and
- transform windows file paths containing \ uniformly to /.

---

\stex\_file\_resolve:Nn  
\stex\_file\_resolve:(No|Nx)      \stex\_file\_resolve:Nn {\(\macro\)}{\(string\)}

resolves and canonicalizes the file path string *string* and stores the result in \macro.

---

\stex\_file\_set:Nn  
\stex\_file\_set:(No|Nx)      \stex\_file\_set:Nn {\(\macro\)}{\(string\)}

represents an already canonicalized file path string as a L<sup>A</sup>T<sub>E</sub>X3 sequence and stores it in \macro.

---

\stex\_if\_file\_absolute\_p:N \*  
\stex\_if\_file\_absolute:NTF \*

\stex\_if\_file\_absolute:N tests whether the given file path (represented as a canonicalized L<sup>A</sup>T<sub>E</sub>X3 sequence) is an absolute file path.

---

\stex\_file\_use:N \*      \stex\_file\_use:N expands to a string representation of the given file path.

---

\stex\_if\_file\_starts\_with:NNTF      \stex\_if\_file\_starts\_with:NN {\(\first\)}{\(\second\)}

tests whether the file path \first is a child of \second. (*Not expandable*)

---

\stex\_file\_split\_off\_ext:NN      \stex\_file\_split\_off\_ext:NN {\(\target\)}{\(\source\)}

splits off the file extension of \source and stores the resulting file path in \target

---

\stex\_file\_split\_off\_lang:NN      \stex\_file\_split\_off\_lang:NN {\(\target\)}{\(\source\)}

checks whether the file path \source ends with a language abbreviation (e.g. .en), if so removes it, and stores the result in \target.

The following are primarily used in file hooks, but might occasionally be useful to call manually:

---

\stex\_filestack\_push:n      pushes the given file to the file stack, recomputing \g\_stex\_current\_file, the current language, document URI and namespace.

---

\stex\_filestack\_pop:      pops the current top entry of the file stack. If the file stack is empty, resets to \c\_stex\_main\_file.

## File Path Constants and Variables

\c\_stex\_pwd\_file store the parent working directory and the absolute path of the main file being processed  
\c\_stex\_main\_file (with guessed file extension .tex).

\c\_stex\_home\_file stores the user's home directory.

\c\_stex\_mathhub\_file stores the user's MathHub directory; its string representation is stored in \mathhub.

\g\_stex\_current\_file always points to the *current* file.

### 11.12.2 URIs

MMT URIs are represented as token lists of the form

{\\_\\_stex\_path\_auth:n{\{authority\}} \\_\\_stex\_path\_path:n{\{path\}} \\_\\_stex\_path\_module:n{\{modulename\}} \\_\\_stex\_path\_name:n{\{declname\}}},

all of which may be empty. Largely, URIs are used as strings only, but the above representation is used in \stex\_uri\_resolve:Nn to canonicalize URIs when they are computed the first time.

\stex\_map\_uri:Nnnnn \stex\_map\_uri:Nnnnn {\{uri\}}{\{authority code\}}{\{path code\}}{\{modulename code\}}{\{declname code\}}

executes the provided *code*s with the components of the *uri* as arguments.

\stex\_uri\_resolve:Nn behaves analogously to \stex\_file\_resolve:Nn.  
\stex\_uri\_resolve:(No|Nx)

\stex\_uri\_set:Nn behaves analogously to \stex\_file\_set:Nn.  
\stex\_uri\_set:(No|Nx)

\stex\_uri\_use:N \* behaves analogously to \stex\_file\_use:N.

A common usage of URIs is computing the namespace of content elements (modules or documents) from the namespace of a math archive and some relative file path within that archive.

\stex\_uri\_from\_repo\_file:NNNn \stex\_uri\_from\_repo\_file:NNNn {\{target\}}{\{repo\_prop\}}{\{filepath\}}{\{ns\_field\}}

computes the namespace URI from the property list *repo\_prop* of some math archive, the file path *filepath* and the archive field {\{ns\_field\}} (*narr* or *ns*), and stores the result in *target*.

---

`\stex_uri_from_repo_file_nolang:NNNn`

behaves like `\stex_uri_from_repo_file:NNNn`, but makes sure to split off language abbreviations from the file name (e.g. `.en`).

---

`\stex_uri_from_current_file:Nn`  
`\stex_uri_from_current_file_nolang:Nn`

Special cases for `\stex_uri_from_repo_file[_nolang]:NNNn`, for `\repo_prop=\l_stex_current_archive_prop` and `\filepath=\g_stex_current_file`.

---

`\stex_set_document_uri:` sets the current value of `\l_stex_current_doc_uri` based on the current file and archive.

---

`\stex_set_current_namespace:`

sets the current value of `\l_stex_current_ns_uri` based on the current file and archive.

---

`\stex_uri_add_module:NNn`  
`\stex_uri_add_module:NNo` `\stex_uri_add_module:NNn {\langle target \rangle}{\langle uri \rangle}{\langle name \rangle}`

Checks that URI `\uri` has no module name, adds the provided `\name` and stores the result in `\target`.

### URI Constants and Variables

---

`\l_stex_current_doc_uri` always points to the current document URI.

---

`\l_stex_current_ns_uri` always points to the current content namespace.

## 11.13 Language Handling

---

`\c_stex_languages_prop`  
`\c_stex_language_abrevs_prop`

Property lists converting babel languages to/from their abbreviations; e.g.

- `\prop_item:Nn \c_stex_languages_prop {de}` yields `ngerman`, and
- `\c_stex_language_abrevs_prop {ngerman}` yields `de`.

---

`\l_stex_current_language_str`

always stores the current language.

---

```
\stex_set_language:n  
\stex_set_language:(x|o)
```

```
\stex_set_language:n {⟨abbrev⟩}
```

Sets `\l_stex_current_language_str`, and, if the `babel` package is loaded, calls `\selectlanguage` on the language corresponding to `{⟨abbrev⟩}`.

Note that the package option `lang=` automatically loads the `babel` package.

If `{⟨abbrev⟩}=tr`, additionally call `\selectlanguage` with the option `shorthands=:!`.

Throws `error/unknownlanguage` if no language with abbreviation `{⟨abbrev⟩}` is known.

---

```
\stex_language_from_file:
```

infers the current language from file ending (e.g. `.en.tex`) and sets it appropriately.

Is called in a file hook, i.e. always switches language when inputting a file `.<lang>.<ext>`.

## 11.14 Inserting Annotations

`STEX` can be used to produce either `HTML` or `PDF`. In `HTML`-mode, multiple macros exist to insert annotations. The same macros are also valid in `PDF` mode but implemented as null operations.

---

```
\stex_suppress_html:n
```

```
\stex_suppress_html:n {⟨code⟩}
```

turns annotations off temporarily in `⟨code⟩` (e.g. as to not generate additional annotations for elaborated declarations, or in sms-mode).

For that to work, code that inserts annotations should use

---

```
\stex_if_do_html_p: * tests whether to generate HTML annotations.  
\stex_if_do_html:TF *
```

---

```
\stex@backend
```

should be set by a backend engine, such that a file `stex-backend-\stex@backend.cfg` exists.

### 11.14.1 Backend macros

Such a backend config file should provide the following:

---

```
\stex_if_html_backend_p: * can be used to determine whether the backend produces HTML (e.g. RuSTEX or LATEXML)  
\stex_if_html_backend:TF *
```

`\ifstexhtml` is set accordingly.

---

```
\stex_annotation:nnn
```

```
\stex_annotation:nnn {⟨attr⟩}{⟨value⟩}{⟨code⟩}
```

In `HTML` mode, annotates the output of `⟨code⟩` with the `XML`-attribute `⟨attr⟩="⟨value⟩"`.

In `PDF` mode, just calls `⟨code⟩`.

---

\stex\_annotation\_invisible:nnn  
\stex\_annotation\_invisible:n

\stex\_annotation\_invisible:n {<code>}

Should annotate <code> with data-shtml-visible="false" style="display:none;".  
In PDF mode, does nothing.

\stex\_annotation\_invisible:nnn combines \stex\_annotation\_invisible:n and \stex\_annotation:nnn.

\stex\_annotation\_env (env.) \begin{stex\_annotation\_env}{<attr>}{<value>} <code> \end{stex\_annotation\_env}  
should behave like \stex\_annotation:nnn{<attr>}{<value>}{<code>}

---

\stex\_mathml\_intent:nn MathML Intent (TODO)  
\stex\_mathml\_arg:nn

---

## 11.15 Persisting Content from Math Archives in sms-Files

---

\stex\_persist:n  
\stex\_persist:e

\stex\_persist:n {<code>}

writes <code> to the \jobname.sms-file, if writesms is active.

**TeXhackers note:** <code> is being read with expl3 category codes (except for spaces having catcode 10), but not pretokenized; i.e. <code> can safely change the current catcode scheme.

---

\c\_stex\_persist\_mode\_bool  
\c\_stex\_persist\_write\_mode\_bool

whether usesms or writesms are active.

---

## 11.16 Utility Methods

---

\stex\_macro\_body:N \*

expands to the *expansion* of the provided macro, including parameter tokens, with the original category codes intact; e.g. if \def\foo#1{First #1}, then \stex\_macro\_body:N \foo expands to First #1.

---

\stex\_macro\_definition:N \*

expands to the token list *defining* the provided macro, including parameters, command attributes (i.e. \long, \protected), with the original category codes intact; e.g. if \protected\def\foo#1{First #1}, then \stex\_macro\_definition:N \foo expands to \protected\def\foo#1{First #1}.

**TeXhackers note:** Does not work with “higher” parameter tokens, i.e. ##1, ####1 etc.

---

`\stex_deactivate_macro:Nn` `\stex_reactivate_macro:N`

`\stex_deactivate_macro:Nn {(\macro)} {(msg)}`

Makes `\macro` throw an error message `error/deactivated-macro{<msg>}`, notifying an author that the macro is only allowed in certain environments.

`\stex_reactivate_macro:N` restores the functionality of the macro.

---

`\stex_kpsewhich:Nn`

`\stex_kpsewhich:Nn {(\macro)} {(<args>)}`

Calls “`kpsewhich <args>`” and stores the result in `\macro`,

**TeXhackers note:** Does not require `shell-escape`

---

`\stex_get_env:Nn`

`\stex_get_env:Nn {(\macro)} {(<envvar>)}`

Stores the value of the environment variable `<envvar>` in `\macro`.

---

`\stex_fatal_error:n` `\stex_fatal_error:nn` `\stex_fatal_error:nxx`

Mimic the `\msg_error:-macros`, but make sure that TeX stops processing.

**TeXhackers note:** Calls `\input{non-existent file}`.

#### `\stex_ignore_spaces_and_pars:`

As the name suggests, ignores all subsequent spaces and `\pars` until the first non-expandable macro is encountered.

Useful for e.g. ending `\symdecl` and related macros with, so that formatting sources with empty lines does not cause paragraph breaks.

### 11.16.1 Group-like Behaviours

---

`\stex_pseudogroup_with:nn`

`\stex_pseudogroup_with:nn {(<macros>){(<code>)}}`

Calls `<code>` and subsequently restores the values of the `<macros>` given.

**TeXhackers note:** Does *not* work recursively!

---

`\stex_pseudogroup:nn`

`\stex_pseudogroup:nn {(<code1>){(<code2>)}}`

Expands `<code2>`, and inserts the result after `<code1>`. Works recursively and allows for restoring the values of macros in combination with `\stex_pseudogroup_restore:N`, but *only for macros that take no arguments*:

---

`\stex_pseudogroup_restore:N *` `\stex_pseudogroup_restore:N {(<macro>)}`

### Example 35

```
1 \stex_pseudogroup:nn{  
2   something changing \foo  
3   something changing \num  
4 }{  
5   \stex_pseudogroup_restore:N\foo  
6   \int_set:Nn \num {\int_use:N \num}  
7 }
```

restores the values of macro `\foo` and register `\num` after calling the first block.

Commands like `\symdecl` and `\importmodule` that generate (semantic) macros should be local *to the current module*, e.g. `smodule`. For that purpose, we open a new “metagroup” with some identifier (e.g. `\l_stex_current_module_str`) and then execute the relevant code *in the metagroup with that identifier*:

---

```
\stex_metagroup_new:n  
\stex_metagroup_new:o
```

`\stex_metagroup_new:n {⟨id⟩}`

Opens a new metagroup at the current TeX group level with identifier `⟨id⟩`.

---

```
\stex_metagroup_do_in:nn  
\stex_metagroup_do_in:nx
```

`\stex_metagroup_do_in:nn {⟨id⟩}{⟨code⟩}`

Executes `⟨code⟩` and adds its content to `\aftergroup` up until the TeX group level of the metagroup with identifier `⟨id⟩`.

# Chapter 12

## Additional Packages

### 12.1 NotesSlides Documentation

TODO

### 12.2 Problem Documentation

TODO

### 12.3 HWExam Documentation

TODO

### 12.4 Tikzinput Documentation

TODO

**Part IV**  
**Implementation**

# Chapter 13

## The sTeX Implementation

### 13.1 Setting up

Setup code for the document class

```
1  <*cls>
2  %%%%%%%%%%%%%%% stex.dtx %%%%%%%%%%%%%%%
3
4  \RequirePackage{expl3, l3keys2e}
5  \ProvidesExplClass{stex}{2023/10/13}{4.0.0}{sTeX document class}
6  \IfFileExists{stex-expl-compat.sty}{
7      \usepackage{stex-expl-compat}
8  }{}
9
10 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{stex}}
11 \ProcessOptions
12
13 \RequirePackage{stex}
14
15 \LoadClass{article}
16 </cls>
    Setup code for the package
17 <*package>
18 \RequirePackage{expl3, l3keys2e, ltxcmds}
19 \ProvidesExplPackage{stex}{2024/10/26}{4.0.0}{sTeX package}
20 \IfFileExists{stex-expl-compat.sty}{
21     \usepackage{stex-expl-compat}
22 }{}
23 \RequirePackage{stex-logo} % externalized for backwards-compatibility reasons
24 \RequirePackage{standalone}
25
26 \bool_new:N \c_stex_use_sref_bool
27 \message{^^J*~This~is~sTeX~version~4.0.0~*^^J}
    Package options:
28 \keys_define:nn { stex / package } {
29     debug      .str_set_x:N  = \c_stex_debug_clist ,
30     lang       .clist_set:N  = \c_stex_languages_clist ,
```

```

31  mathhub      .tl_set_x:N = \mathhub ,
32  usesms       .bool_set:N = \c_stex_persist_mode_bool ,
33  writesms     .bool_set:N = \c_stex_persist_write_mode_bool ,
34  forcenousesms .bool_set:N = \c_stex_persist_force_bool ,
35  checkterms   .bool_set:N = \c_stex_check_terms_bool ,
36  metadata      .bool_set:N = \c_stex_metadata_bool ,
37  image         .bool_set:N = \c_tikzinput_image_bool,
38  nofrontmatter .bool_set:N = \c_stex_no_frontmatter_bool,
39  unknown       .code:n      = {}
40 }
41 \exp_args:NNo \clist_set:Nn \c_stex_debug_clist \c_stex_debug_clist
42 \ProcessKeysOptions { stex / package }

    Error messages:
43 \input{stex-en.ldf}

```

## 13.2 Utilities

44 \cs\_set\_eq:NN \stex\_undefined: \undefined

### 13.2.1 Calling kpsewhich and Environment Variables

45 <@=stex\_envs>

\stex\_kpsewhich:Nn

```

46 \cs_new_protected:Nn \stex_kpsewhich:Nn {\group_begin:
47   \catcode`\|=12
48   \sys_get_shell:nnN { kpsewhich ~ #2 } { } \l_tmpa_tl
49   \tl_gset_eq:NN \l_tmpa_tl \l_tmpa_tl
50   \group_end:
51   \exp_args:NNo\str_set:Nn #1 {\l_tmpa_tl}
52   \tl_trim_spaces:N #1
53 }

```

(End of definition for \stex\_kpsewhich:Nn. This function is documented on page 145.)

\stex\_get\_env:Nn

```

54 \sys_if_platform_windows:TF{
55   \cs_new_protected:Nn \stex_get_env:Nn {\group_begin:
56     \escapechar=-1\catcode`\|=12
57     \exp_args:NNe \stex_kpsewhich:Nn #1 {-expand-var~\c_percent_str#2\c_percent_str}
58     \exp_args:NNx \str_if_eq:VnT #1 {\c_percent_str #2 \c_percent_str} {
59       \str_clear:N #1
60     }
61     \exp_args:NNx\use:nn\group_end: {
62       \str_set:Nn \exp_not:N #1 { #1 }
63     }
64   }
65 }{
66   \cs_new_protected:Nn \stex_get_env:Nn {
67     \stex_kpsewhich:Nn #1 {-var-value-#2}
68   }
69 }

```

(End of definition for \stex\_get\_env:Nn. This function is documented on page 145.)

### 13.2.2 Logging

```
70 <@=stex_debug>
\stex_debug:nn
71 \cs_new_protected:Nn \stex_debug:nn {
72   \exp_args:NNo \clist_if_in:NnTF \c_stex_debug_clist { \tl_to_str:n{all} }{
73     \__stex_debug_:nn{#1}{#2}
74   }{
75     \exp_args:NNo \clist_if_in:NnT \c_stex_debug_clist { \tl_to_str:n{#1} }{
76       \__stex_debug_:nn{#1}{#2}
77     }
78   }
79 }
80
81 \cs_new_protected:Nn \__stex_debug_:nn {
82   \msg_set:nnn{stex}{debug / #1}{
83     \\Debug~#1:~#2\\
84   }
85   \msg_none:nn{stex}{debug / #1}
86 }
```

(End of definition for `\stex_debug:nn`. This function is documented on page 123.)

We check an environment variable for debugging and set things up:

```
87 \stex_get_env:Nn\__stex_debug_env_str{STEX_DEBUG}
88 \str_if_empty:NTF\__stex_debug_env_str {
89   \clist_set_eq:NN \l__stex_debug_cl \c_stex_debug_clist
90 }{
91   \clist_set:No \l__stex_debug_cl {\__stex_debug_env_str}
92 }
93 \clist_clear:N \c_stex_debug_clist
94 \clist_map_inline:Nn \l__stex_debug_cl {
95   \exp_args:NNo \clist_put_right:Nn \c_stex_debug_clist
96   { \tl_to_str:n{#1} }
97 }
98
99 \exp_args:NNo \clist_if_in:NnTF \c_stex_debug_clist {\tl_to_str:n{all}} {
100   \msg_redirect_module:nnn{ stex }{ none }{ warning }
101   \stex_debug:nn{all}{Logging~everything!}
102 }{
103   \clist_map_inline:Nn \c_stex_debug_clist {
104     \msg_redirect_name:nnn{ stex }{ debug / #1 }{ warning }
105     \stex_debug:nn{#1}{Logging~#1}
106   }
107 }
```

### 13.2.3 File Paths

```
108 <@=stex_path>
\c_stex_filepath_sep_str
109 \sys_if_platform_windows:TF{
110   \str_set_eq:NN \c_stex_filepath_sep_str \c_backslash_str
111 }{
112   \str_const:Nn \c_stex_filepath_sep_str {}
```

113 }

(End of definition for `\c_stex_filepath_sep_str`. This variable is documented on page ??.)

### 13.2.4 Languages

114 `\c_stex_lang`

`\c_stex_languages_prop`

We store language abbreviations in two (mutually inverse) property lists:

```
115 \exp_args:NNx \prop_const_from_keyval:Nn \c_stex_languages_prop { \tl_to_str:n {
116   en = english ,
117   de = ngerman ,
118   ar = arabic ,
119   bg = bulgarian ,
120   ru = russian ,
121   fi = finnish ,
122   ro = romanian ,
123   tr = turkish ,
124   fr = french ,
125   sl = slovenian
126 } }
127
128 \exp_args:NNx \prop_const_from_keyval:Nn \c_stex_language_abbrevs_prop { \tl_to_str:n {
129   english = en ,
130   ngerman = de ,
131   arabic = ar ,
132   bulgarian = bg ,
133   russian = ru ,
134   finnish = fi ,
135   romanian = ro ,
136   turkish = tr ,
137   french = fr ,
138   slovenian = sl ,
139 } }
140 % todo: chinese simplified (zhs)
141 %       chinese traditional (zht)
```

(End of definition for `\c_stex_languages_prop` and `\c_stex_language_abbrevs_prop`. These variables are documented on page 142.)

`\l_stex_current_language_str`

142 `\str_new:N \l_stex_current_language_str`

(End of definition for `\l_stex_current_language_str`. This variable is documented on page 142.)

we use the `lang`-package option to load the corresponding babel languages:

`\stex_set_language:n`

`\stex_set_language:x`

`\stex_set_language:o`

143 `\cs_new_protected:Nn \stex_set_language:n {`

144  `\str_set:Nn \l_stex_current_language_str { #1 }`

145  `\prop_if_in:NnTF \c_stex_languages_prop {#1} {`

146  `\tl_set_rescan:Nnx \l_stex_lang_lang_str {}{\prop_item:Nn \c_stex_languages_prop {#1}}`

147  `\cs_if_eq:NNTF \onlypreamble \notprerr {`

148  `\ltx@ifpackageloaded{babel}{`

149  `\exp_args:No \selectlanguage \l_stex_lang_lang_str`

150  `}{}`

```

151    }{
152      \ltx@ifpackageloaded{babel}{}{
153        \str_if_eq:nnTF {#1} {tr} {
154          \RequirePackage[turkish,shorthands=:!]{babel}
155        }{
156          \RequirePackage[\l__stex_lang_lang_str]{babel}
157        }
158      }
159    }
160  }{
161    \msg_error:nnx{stex}{error/unknownlanguage}{#1}
162  }
163 }
164 \cs_generate_variant:Nn \stex_set_language:n {x,o}

```

(End of definition for `\stex_set_language:n`. This function is documented on page 143.)

#### `\stex_language_from_file:`

```

165 \cs_new_protected:Nn \stex_language_from_file: {
166   \seq_get_right:NN \g_stex_current_file \l_tmpa_str
167   \seq_set_split:NnV \l_tmpa_seq . \l_tmpa_str
168   \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str \% = ".tex/.dtx/.sty"
169   \exp_args:No \str_if_eq:nnF \l_tmpa_str {tex} {
170     \exp_args:No \str_if_eq:nnF \l_tmpa_str {dtx} {
171       \exp_args:No \str_if_eq:nnF \l_tmpa_str {ltx} {
172         \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq \l_tmpa_str
173       }
174     }
175   }
176   \seq_pop_left:NN \l_tmpa_seq \l_tmpa_str \% <filename>
177   \seq_if_empty:NF \l_tmpa_seq { %remaining element should be [<something>.]language
178     \seq_pop_right:NN \l_tmpa_seq \l__stex_lang_str
179     \str_if_eq:NNF \l__stex_lang_str \l_stex_current_language_str {
180       \exp_args:NNo \prop_if_in:NnT \c_stex_languages_prop \l__stex_lang_str {
181         \stex_set_language:o \l__stex_lang_str
182       }
183     }
184     \stex_debug:nn{lang} {Language~\l_stex_current_language_str~
185       inferred~from~file~name}
186   }
187 }

```

(End of definition for `\stex_language_from_file:`. This function is documented on page 143.)

Loading babel:

```

188 \clist_if_empty:NF \c_stex_languages_clist {
189   \bool_set_false:N \l__stex_lang_turkish_bool
190   \seq_clear:N \l_tmpa_seq
191   \clist_map_inline:Nn \c_stex_languages_clist {
192     \str_set:Nx \l_tmpa_str {#1}
193     \str_if_eq:nnT {#1}{tr} {
194       \bool_set_true:N \l__stex_lang_turkish_bool
195     }
196     \prop_get:NoNTF \c_stex_languages_prop \l_tmpa_str \l_tmpa_str {
197       \tl_set_rescan:Nno \l_tmpa_str {} \l_tmpa_str

```

```

198     \seq_put_right:Nn \l_tmpa_seq \l_tmpa_str
199   } {
200     \msg_error:nnx{stex}{error/unknownlanguage}{\l_tmpa_str}
201   }
202 }
203 \stex_debug:nn{lang} {Languages:~\seq_use:Nn \l_tmpa_seq {,~} }
204 \bool_if:NTF \l_stex_lang_turkish_bool {
205   \exp_args:NNe \use:nn \RequirePackage
206   {[main=\seq_use:Nn \l_tmpa_seq, ,shorthands=:!]}{babel}
207 }{
208   \exp_args:NNe \use:nn \RequirePackage
209   {[main=\seq_use:Nn \l_tmpa_seq, ]}{babel}
210 }
211 }

```

### 13.2.5 Group-like Behaviours

212 ⟨@=stex\_groups⟩

```

\stex_pseudogroup:nn
\stex_pseudogroup_restore:N
213 \cs_new_protected:Npn \stex_pseudogroup:nn {
214   \exp_args:Nne \use:nn
215 }
216 \cs_new:Nn \stex_pseudogroup_restore:N {
217   \tl_if_exist:NTF #1 {
218     \tl_set:Nn \exp_not:N #1 { \exp_args:No \exp_not:n #1 }
219   }{
220     \cs_undefine:N \exp_not:N #1
221   }
222 }

```

(End of definition for `\stex_pseudogroup:nn` and `\stex_pseudogroup_restore:N`. These functions are documented on page 145.)

`\stex_pseudogroup_with:nn`

```

223 \cs_new_protected:Nn \stex_pseudogroup_with:nn {
224   \tl_map_inline:nn{#1}{
225     \cs_set_eq:cN{\_stex_groups_\tl_to_str:n{##1}}{##1}
226   }
227   #2
228   \tl_map_inline:nn{#1}{
229     \cs_set_eq:Nc{##1}{\_stex_groups_\tl_to_str:n{##1}}
230     \cs_undefine:c{\_stex_groups_\tl_to_str:n{##1}}
231   }
232 }

```

(End of definition for `\stex_pseudogroup_with:nn`. This function is documented on page 145.)

`\stex_metagroup_new:n` List of all currently existing metagroup identifiers

`\stex_metagroup_new:o` 233 `\seq_new:N \l_stex_groups_ids_seq`

start a new metagroup at the current group level with id #1

```

234 \cs_new_protected:Nn \stex_metagroup_new:n {
235   \str_set:cx{l_stex_groups_#1_int} {\int_use:N\currentgrouplevel}
236   \seq_put_right:Nn \l_stex_groups_ids_seq {#1}

```

```

237 }
238 \cs_generate_variant:Nn \stex_metagroup_new:n {o}

(End of definition for \stex_metagroup_new:n. This function is documented on page 146.)

\stex_metagroup_do_in:nn
\stex_metagroup_do_in:nx
239 \prg_new_conditional:Nnn \__stex_groups_exists:n {TF} {
240     \str_if_exist:cTF{l__stex_groups_#1_int}
241         \prg_return_true: \prg_return_false:
242 }
243
244 \cs_new_protected:Nn \stex_metagroup_do_in:nn {
245     \__stex_groups_exists:nTF{#1}{
246         \__stex_groups_do_in:nn{#1}{#2}
247     }{
248         \msg_error:nnn{stex}{error/metagroup/missing}{#1}
249     }
250 }
251 \cs_generate_variant:Nn \stex_metagroup_do_in:nn {nx}
252
253 \cs_new_protected:Nn \__stex_groups_do_in:nn {
254     \exp_args:Nnx\stex_debug:nn{metagroup}{adding-to-\detokenize{#1}:^~\tl_to_str:n{#2}}
255     \tl_set:Nn\__stex_groups_tmp{#2}
256     \exp_args:Nx \int_compare:nNnF {\use:c{l__stex_groups_#1_int}}
257         = \currentgrouplevel {
258             \tl_if_exist:cTF{g__stex_groups_#1_\the\currentgrouplevel _content} {
259                 \exp_args:Nno \tl_gput_right:cn{g__stex_groups_#1_\the\currentgrouplevel _content}
260             }{
261                 \exp_args:Nno \tl_gset:cn{g__stex_groups_#1_\the\currentgrouplevel _content}
262             }\__stex_groups_tmp
263             \bool_if_exist:cF {l__stex_groups_\the\currentgrouplevel _bool} {
264                 \group_insert_after:N \__stex_groups_do:
265                 \bool_set_true:c {l__stex_groups_\the\currentgrouplevel _bool}
266             }
267         }
268         \__stex_groups_tmp
269     }
270
271 \cs_new_protected:Nn \__stex_groups_do: {
272     \seq_map_inline:Nn \l__stex_groups_ids_seq {
273         \tl_if_exist:cT{g__stex_groups_#1_int_eval:n{\currentgrouplevel+1}_content} {
274             \exp_args:NNno \exp_args:Nno \__stex_groups_do_in:nn{##1} {
275                 \csname g__stex_groups_##1_int_eval:n{\currentgrouplevel+1}_content\endcsname
276             }
277             \cs_undefine:c{g__stex_groups_##1_int_eval:n{\currentgrouplevel+1}_content}
278         }
279         \bool_if_exist:cF {l__stex_groups_\int_eval:n\currentgrouplevel _bool} {
280             \group_insert_after:N \__stex_groups_do:
281             \bool_set_true:c {l__stex_groups_\int_eval:n\currentgrouplevel _bool}
282         }
283     }
284 }

```

(End of definition for \stex\_metagroup\_do\_in:nn. This function is documented on page 146.)

### 13.2.6 HTML Annotations

285 ⟨@=stex\_annotation⟩

\stex\_if\_do\_html:TF Whether to (locally) produce HTML output

```

286 \bool_new:N \_stex_html_do_output_bool
287 \bool_set_true:N \_stex_html_do_output_bool
288
289 \prg_new_conditional:Nnn \stex_if_do_html: {p,T,F,TF} {
290   \bool_if:nTF \_stex_html_do_output_bool
291   \prg_return_true: \prg_return_false:
292 }
```

(End of definition for \stex\_if\_do\_html:TF. This function is documented on page 143.)

\stex\_suppress\_html:n Temporarily disable HTML output

```

293 \cs_new_protected:Nn \stex_suppress_html:n {
294   \stex_pseudogroup:nn{
295     \bool_set_false:N \_stex_html_do_output_bool
296     #1
297   }{
298     \stex_if_do_html:T {
299       \bool_set_true:N \_stex_html_do_output_bool
300     }
301   }
302 }
```

(End of definition for \stex\_suppress\_html:n. This function is documented on page 143.)

We determine the backend:

```

\stex_if_html_backend_p:
\stex_if_html_backend:TF
  \ifstexhtml
    \stex@backend
      303 \ifcsname if@rustex\endcsname\else
      304   \expandafter\newif\csname if@rustex\endcsname
      305   \crustexfalse
      306 \fi
      307
      308 \stex_get_env:Nn\__stex_annotation_env_str{STEX_FORCE_PDF}
      309 \exp_args:No \str_if_eq:nnTF \__stex_annotation_env_str {true} {
      310   \def\stex@backend{pdflatex}
      311 }{
      312   \tl_if_exist:N\stex@backend{
      313     \if@rustex
      314       \def\stex@backend{rustex}
      315     \else
      316       \cs_if_exist:NTF\HCode{
      317         \def\stex@backend{tex4ht}
      318       }{
      319         \def\stex@backend{pdflatex}
      320       }
      321     \fi
      322   }
      323 }
      324 \input{stex-backend-\stex@backend.cfg}
      325
      326
```

```

327 \newif\ifstexhtml
328 \stex_if_html_backend:TF {
329   \stexhtmltrue
330   \bool_set_true:N \_stex_html_do_output_bool
331 }{
332   \stexhtmlfalse
333   \bool_set_false:N \_stex_html_do_output_bool
334 }

```

(End of definition for `\stex_if_html_backend:TF`, `\ifstexhtml`, and `\stex@backend`. These functions are documented on page 143.)

```

\mmlintent
\mmlarg
335 \stex_if_html_backend:TF {
336   \cs_new_protected:Npn \mmlintent #1 #2 {
337     \stex_annotation:nn{mml:intent={#1}}{#2}
338   }
339   \cs_new_protected:Npn \mmlarg #1 #2 {
340     \stex_annotation:nn{mml:arg={#1}}{#2}
341   }
342 }{
343   \cs_new_protected:Npn \mmlintent #1 #2 { #2 }
344   \cs_new_protected:Npn \mmlarg #1 #2 { #2 }
345 }

```

(End of definition for `\mmlintent` and `\mmlarg`. These functions are documented on page ??.)

### 13.2.7 Auxiliary Methods

```
346 <@=stex_aux>
```

```

\stex_deactivate_macro:Nn
\stex_reactivate_macro:N
347 \cs_new_protected:Nn \stex_deactivate_macro:Nn {
348   \tl_set_eq:cN{\tl_to_str:n{#1}~~~orig}{#1}
349   \cs_set_protected:Npn#1{
350     \msg_error:nnnn{stex}{error/deactivated-macro}{#1}{#2}
351   }
352 }
353 \cs_new_protected:Nn \stex_reactivate_macro:N {
354   \exp_after:wn\let\exp_after:wn#1\csname \detokenize{#1}~~~orig\endcsname
355 }

```

(End of definition for `\stex_deactivate_macro:Nn` and `\stex_reactivate_macro:N`. These functions are documented on page 145.)

```
\stex_ignore_spaces_and_pars:
```

```

356 \protected\def\stex_ignore_spaces_and_pars: {
357   \begingroup\catcode13=10\relax
358   \@ifnextchar\parf
359     \endgroup\expandafter\stex_ignore_spaces_and_pars:\@gobble
360   }{
361     \endgroup
362   }
363 }

```

(End of definition for `\stex_ignore_spaces_and_pars:`. This function is documented on page 145.)

```

\stex_keys_define:nnnn
 364 \cs_new_nopar:Nn \stex_keys_define:nnnn {
 365   \tl_gset:cn {__stex_aux_keys_#1_pre_tl}{#2}
 366   \tl_gset:cn {__stex_aux_keys_#1_def_tl}{#3}
 367   \tl_if_empty:nF{#4} {
 368     \clist_map_inline:nn{#4} {
 369       \tl_set_eq:Nc \l__stex_aux_tl {__stex_aux_keys_##1_pre_tl}
 370       \tl_gput_left:co{__stex_aux_keys_#1_pre_tl} \l__stex_aux_tl
 371       \tl_set_eq:Nc \l__stex_aux_tl {__stex_aux_keys_##1_def_tl}
 372       \tl_gput_left:cn{__stex_aux_keys_#1_def_tl} ,
 373       \tl_gput_left:co{__stex_aux_keys_#1_def_tl} \l__stex_aux_tl
 374     }
 375   }
 376   \tl_set_eq:Nc \l__stex_aux_tl {__stex_aux_keys_#1_def_tl}
 377   \exp_args:Nno \keys_define:nn {stex / #1} {\l__stex_aux_tl}
 378 }

```

(End of definition for `\stex_keys_define:nnnn`. This function is documented on page 135.)

`\stex_keys_set:nn`

```

 379 \cs_new_nopar:Nn \stex_keys_set:nn {
 380   \use:c{\__stex_aux_keys_#1_pre_tl}
 381   \keys_set:nn {stex / #1} { #2 }
 382 }

```

(End of definition for `\stex_keys_set:nn`. This function is documented on page 135.)

Some ubiquitous key sets:

```

 383 \stex_keys_define:nnnn{archive file} {
 384   \str_clear:N \l_stex_key_archive_str
 385   \str_clear:N \l_stex_key_file_str
 386 } {
 387   archive .str_set_x:N = \l_stex_key_archive_str ,
 388   file .str_set_x:N = \l_stex_key_file_str
 389 } {}
 390
 391 \stex_keys_define:nnnn{id} {
 392   \str_clear:N \l_stex_key_id_str
 393 } {
 394   id .str_set_x:N = \l_stex_key_id_str
 395 } {}
 396
 397 \stex_keys_define:nnnn{title} {
 398   \tl_clear:N \l_stex_key_title_tl
 399 } {
 400   title .tl_set:N = \l_stex_key_title_tl
 401 } {}
 402
 403 \stex_keys_define:nnnn{style} {
 404   \clist_clear:N \l_stex_key_style_clist
 405 } {
 406   style .clist_set:N = \l_stex_key_style_clist
 407 } {}
 408
 409 \stex_keys_define:nnnn{deprecate} {

```

```

410   \str_clear:N \l_stex_key_deprecate_str
411 }{
412   deprecate     .str_set_x:N = \l_stex_key_deprecate_str
413 }{}
414
415 \stex_keys_define:nnnn{uses}{}
416   uses .code:n = {
417     \clist_map_inline:nn{#1}{%
418       \stex_if_starts_with:nTF{##1}[{%
419         \__stex_aux_split_at_bracket:w ##1 \_stex_end:
420       }{%
421         \usemodule{##1}
422       }
423     }
424   }
425 }{}
426
427 \cs_new_protected:Npn \__stex_aux_split_at_bracket:w [ #1 ] #2 \_stex_end: {%
428   \usemodule[#1]{#2}
429 }

```

#### \\_stex\_do\_deprecation:n

```

430 \cs_new:Nn \_stex_do_deprecation:n {
431   \str_if_empty:NF \l_stex_key_deprecate_str {
432     \msg_warning:nnxx{stex}{warning/deprecated}{#1}{\l_stex_key_deprecate_str}
433   }
434 }

```

(End of definition for `\_stex_do_deprecation:n`. This function is documented on page 123.)

#### \\_stex\_do\_id:

```

435 \cs_new_protected:Nn \_stex_do_id: {
436   \stex_if_smsmode:F {
437     \str_if_empty:NF \l_stex_key_id_str {
438       \exp_args:No \stex_ref_new_doc_target:n \l_stex_key_id_str
439     }
440   }
441 }

```

(End of definition for `\_stex_do_id:`. This function is documented on page 135.)

```

\stex_new_styable_env:nnnnnn
\stex_new_styable_cmd:nnn
\stex_style_apply:

```

```

442 \cs_new_protected:Nn \stex_new_styable_cmd:nnnn {
443   \exp_after:wN \newcommand \cs:w stexstyle#1 \cs_end:[2][]{%
444     \__stex_aux_patch:nnn{#1}{##1}{##2}
445   }
446   \exp_after:wN \NewDocumentCommand\cs:w #1\cs_end:{#2}%
447     \cs_set:Npn \stex_style_apply: {
448       \__stex_aux_apply_patch:n{#1}
449     }
450     #3
451   }
452   \tl_set:cn {\__stex_aux_style_#1:} { #4 }
453 }

```

```

454 \cs_new_protected:Nn \__stex_aux_apply_patch:n {
455   \clist_if_empty:NTF \l_stex_key_style_clist {
456     \tl_clear:N \thisstyle
457     \use:c{\__stex_aux_style_#1:}
458   }{
459     \clist_get:NN \l_stex_key_style_clist \thisstyle
460     \tl_if_exist:cTF{\__stex_aux_style_#1_\thisstyle :} {
461       \use:c{\__stex_aux_style_#1_\thisstyle :}
462     }{
463       \use:c{\__stex_aux_style_#1:}
464     }
465   }
466 }
467 }
468
469 \cs_new_protected:Nn \__stex_aux_patch:nnn {
470   \str_if_empty:nTF {#2}{%
471     \tl_set:cn{\__stex_aux_style_#1:}{#3}
472   }{
473     \tl_set:cn{\__stex_aux_style_#1_#2:}{#3}
474   }
475 }
476
477 \cs_new_protected:Nn \stex_new_stylable_env:nnnnnnn {
478   \exp_after:wN \newcommand \cs:w stexstyle#1 \cs_end:[3][]{
479     \__stex_aux_patch:nnnn{#1}{##1}{##2}{##3}
480   }
481   \NewDocumentEnvironment{#7#1}{#2}{%
482     \cs_set:Npn \stex_style_apply: {
483       \__stex_aux_apply_patch_begin:n{#1}
484     }
485     #3
486   }{
487     \cs_set:Npn \stex_style_apply: {
488       \__stex_aux_apply_patch_end:n{#1}
489     }
490     #4
491   }
492   \tl_set:cn {\__stex_aux_style_#1_start:} { #5 }
493   \tl_set:cn {\__stex_aux_style_#1_end:} { #6 }
494 }
495
496 \cs_new_protected:Nn \__stex_aux_patch:nnn {
497   \str_if_empty:nTF {#2}{%
498     \tl_set:cn{\__stex_aux_style_#1_start:}{#3}
499     \tl_set:cn{\__stex_aux_style_#1_end:}{#4}
500   }{
501     \tl_set:cn{\__stex_aux_style_#1_#2_start:}{#3}
502     \tl_set:cn{\__stex_aux_style_#1_#2_end:}{#4}
503   }
504 }
505
506 \cs_new_protected:Nn \__stex_aux_apply_patch_begin:n {
507   \clist_if_empty:NTF \l_stex_key_style_clist {

```

```

508     \tl_clear:N \thisstyle
509     \use:c{__stex_aux_style_#1_start:}
510 }{
511     \clist_get:NN \l_stex_key_style_clist \thisstyle
512     \stex_debug:nnf{styling}{dominant-style:~\thisstyle}
513     \tl_if_exist:cTF{__stex_aux_style_#1_\thisstyle_start:}{
514         \use:c{__stex_aux_style_#1_\thisstyle_start:}
515     }{
516         \use:c{__stex_aux_style_#1_start:}
517     }
518 }
519 }
520
521 \cs_new_protected:Nn \__stex_aux_apply_patch_end:n {
522     \tl_if_empty:NTF \thisstyle {
523         \use:c{__stex_aux_style_#1_end:}
524     }{
525         \tl_if_exist:cTF{__stex_aux_style_#1_\thisstyle_end:}{
526             \use:c{__stex_aux_style_#1_\thisstyle_end:}
527         }{
528             \use:c{__stex_aux_style_#1_end:}
529         }
530     }
531 }

```

(End of definition for `\stex_new_stytable_env:nnnnnnn`, `\stex_new_stytable_cmd:nnnn`, and `\stex_style_apply:`. These functions are documented on page 136.)

`\stex_str_if_ends_with_p:nn`

`\stex_str_if_ends_with:nnTF`

```

532 \prg_new_conditional:Nnn \stex_str_if_ends_with:nn {p,T,F,TF} {
533     \exp_args:Ne \str_if_eq:nnTF {
534         \str_range:nnn{#1}{- \str_count:n{#2}}{-1}
535     }{#2}\prg_return_true: \prg_return_false:
536 }

```

(End of definition for `\stex_str_if_ends_with:nnTF`. This function is documented on page 139.)

`\stex_str_if_starts_with_p:nn`

`\stex_str_if_starts_with:nnTF`

```

537 \prg_new_conditional:Nnn \stex_str_if_starts_with:nn {p,T,F,TF} {
538     \exp_args:Ne \str_if_eq:nnTF {
539         \str_range:nnn{#1}{1}{\str_count:n{#2}}
540     }{#2}\prg_return_true: \prg_return_false:
541 }

```

(End of definition for `\stex_str_if_starts_with:nnTF`. This function is documented on page 139.)

`\stex_macro_body:N`

```

542 \cs_new:Npn \__stex_aux_start:#1\__stex_aux_end: {\exp_not:n{#1}}
543 \cs_new_protected:Nn \__stex_aux_end: {}
544 \cs_new:Nn \stex_macro_body:N {
545     \exp_args:Nne\use:nn{\exp_after:wN \__stex_aux_start: #1} {
546         \__stex_aux_args:e {\cs_parameter_spec:N #1}\__stex_aux_end:
547     }
548 }
549

```

```

550 \cs_new:Nn \__stex_aux_args:n {
551   \tl_if_empty:nF{#1} {
552     {##\exp_args:N \tl_head:n {\tl_tail:n {#1}}}
553     \__stex_aux_args:e {\exp_args:N\tl_tail:n{\tl_tail:n{#1}}}
554   }
555 }
556 \cs_generate_variant:Nn \__stex_aux_args:n {e}

```

(End of definition for `\stex_macro_body:N`. This function is documented on page 144.)

```

\stex_macro_definition:N
557 \cs_new:Nn \stex_macro_definition:N {
558   \__stex_aux_prefix:e {\cs_prefix_spec:N #1}
559   \def\exp_not:N #1
560   \__stex_aux_params:e {\cs_parameter_spec:N #1}
561   {
562     \stex_macro_body:N #1
563   }
564 }
565
566 \cs_new:Nn \__stex_aux_prefix:n {
567   \tl_if_empty:nF{#1} {
568     \str_if_eq:eeTF {
569       \tl_range:nnn{#1}{1}{10}~
570     }{\tl_to_str:n{\protected}}{
571       \protected
572       \__stex_aux_prefix_long:e {
573         \str_range:nnn{#1}{11}{-1}
574       }
575     }{
576       \__stex_aux_prefix_long:n {#1}
577     }
578   }
579 }
580 \cs_generate_variant:Nn \__stex_aux_prefix:n {e}
581
582 \cs_new:Nn \__stex_aux_prefix_long:n {
583   \tl_if_empty:nF{#1} {
584     \str_if_eq:eeT {
585       \tl_range:nnn{#1}{1}{10}~
586     }{\tl_to_str:n{\long}}{\long}
587   }
588 }
589 \cs_generate_variant:Nn \__stex_aux_prefix_long:n {e}
590
591 \cs_new:Nn \__stex_aux_params:n {
592   \tl_if_empty:nF{#1} {
593     \exp_args:NNN \str_if_eq:VnTF \c_hash_str {\tl_head:n{#1}}{
594       #####
595     }{
596       \tl_head:n{#1}
597     }
598     \__stex_aux_params:e {\tl_tail:n{#1}}
599   }

```

```

600 }
601 \cs_generate_variant:Nn \__stex_aux_params:n {e}

```

(End of definition for \stex\_macro\_definition:N. This function is documented on page 144.)

### 13.2.8 Persistence

```

602 <@=stex_persist>

```

We check the environment variables:

```

603 \stex_get_env:Nn\__stex_persist_env_str{STEX_USESMS}
604 \str_if_empty:NF\__stex_persist_env_str{
605   \exp_args:No \str_if_eq:nnF \__stex_persist_env_str{false} {
606     \bool_set_true:N \c_stex_persist_mode_bool
607   }
608 }
609 \stex_get_env:Nn\__stex_persist_env_str{STEX_WRITESMS}
610 \str_if_empty:NF\__stex_persist_env_str{
611   \exp_args:No \str_if_eq:nnF \__stex_persist_env_str{false} {
612     \bool_set_true:N \c_stex_persist_write_mode_bool
613   }
614 }
615
616 \bool_if:NT \c_stex_persist_force_bool {
617   \bool_set_false:N \c_stex_persist_mode_bool
618 }

```

```

\stex_persist:n
\stex_persist:e
619 \iow_new:N \c__stex_persist_sms_iow
620
621 \bool_if:NTF \c_stex_persist_write_mode_bool {
622   \stex_if_html_backend:TF{
623     \cs_new:Npn \stex_persist:n #1 {}
624     \cs_new:Npn \stex_persist:e #1 {}
625   }{
626     \cs_new_protected:Nn \stex_persist:n {
627       \iow_now:Nn \c__stex_persist_sms_iow {#1}
628     }
629     \cs_generate_variant:Nn \stex_persist:n {e}
630   }
631 }{
632   \cs_new:Npn \stex_persist:n #1 {}
633   \cs_new:Npn \stex_persist:e #1 {}
634 }

```

(End of definition for \stex\_persist:n. This function is documented on page 144.)

Is called at the end of the .sty-file:

```

635
636 \cs_new_protected:Nn \__stex_persist_load_file:n {
637   \file_if_exist:nT{#1} {
638     \group_begin:
639     \cs_set:Npn \stex_persist:n ##1 {}
640     \cs_set:Npn \stex_persist:e ##1 {}
641     \stex_debug:nn{persist}{restoring~from~sms~file}
642     \catcode`\ =10\relax

```

```

643      \cs:w @ @ input \cs_end:#1\relax
644      \group_end:
645    }
646  }
647
648 \cs_new_protected:Nn \__stex_persist_write_only: {
649   \iow_open:Nn \c_stex_persist_sms_iow {\jobname.sms}
650   \AtEndDocument{ \iow_close:N \c_stex_persist_sms_iow }
651 }
652
653 \cs_new_protected:Nn \__stex_persist_read_and_write: {
654   \file_if_exist:nTF{\jobname.sms}{%
655     \ior_open:Nn \g_tmpa_ior {\jobname.sms}
656     \iow_open:Nn \g_tmpa_iow {\jobname sms2}
657     \ior_str_map_inline:Nn \g_tmpa_ior {
658       \iow_now:Nn \g_tmpa_iow {##1}
659     }
660     \iow_close:N \g_tmpa_iow
661     \ior_close:N \g_tmpa_ior
662     \__stex_persist_write_only:
663     \ior_open:Nn \g_tmpa_ior {\jobname sms2}
664     \ior_str_map_inline:Nn \g_tmpa_ior {
665       \iow_now:Nn \c_stex_persist_sms_iow {##1}
666     }
667     \ior_close:N \g_tmpa_ior
668     \__stex_persist_load_file:n{\jobname sms2}
669   }\__stex_persist_write_only:
670 }
671
672 \cs_new_protected:Nn \__stex_persist_read_now: {
673   \bool_if:NTF \c_stex_persist_mode_bool {
674     \bool_if:NTF \c_stex_persist_write_mode_bool
675       \__stex_persist_read_and_write:
676     {
677       \__stex_persist_load_file:n{\jobname sms}
678     }
679   }{
680     \bool_if:NT \c_stex_persist_write_mode_bool \__stex_persist_write_only:
681   }
682 }

```

### 13.2.9 Files, Paths and URIs

```

683 <@=stex_path>
\stex_file_set:Nn
\stex_file_set:Nn
\stex_file_set:Nx
684 \cs_new_protected:Nn \stex_file_set:Nn {
685   \str_if_empty:nTF {#2} { \seq_clear:N #1 }{%
686     \exp_args:NNno \seq_set_split:Nnn #1 / { \tl_to_str:n{#2} }%
687   }
688 }
689 \cs_generate_variant:Nn \stex_file_set:Nn {No, Nx}

```

(End of definition for \stex\_file\_set:Nn. This function is documented on page 140.)

```

\stex_file_resolve:Nn
\stex_file_resolve:No
\stex_file_resolve:Nx
 690 \sys_if_platform_windows:TF{
 691   \cs_new_protected:Npn \__stex_path_win_take:w #1#2#3 \__stex_path_ : {
 692     \uppercase{ \str_set:Nn \l__stex_path_str{#1}}
 693     \str_set:Nx \l__stex_path_drive {\l__stex_path_str #2}
 694     \str_set:Nn \l__stex_path_str{#3}
 695   }
 696   \cs_new_protected:Nn \stex_file_resolve:Nn {
 697     \str_set:Nn \l__stex_path_str {#2}
 698     \str_clear:N \l__stex_path_win_drive
 699     \exp_args:NNo \str_replace_all:Nnn \l__stex_path_str \c_backslash_str /
 700     \exp_args:Nx \str_if_eq:nnT {\str_item:Nn \l__stex_path_str 2} : {
 701       \exp_after:wN \__stex_path_win_take:w \l__stex_path_str \__stex_path_ :
 702     }
 703     \stex_file_set:No #1 \l__stex_path_str
 704     \__stex_path_canonicalize:N #1
 705     \str_if_empty:NF \l__stex_path_win_drive {
 706       \seq_pop_left:NN #1 \l__stex_path_str
 707       \seq_put_left:No #1 \l__stex_path_win_drive
 708     }
 709     \%stex_debug:nn{files}{Set~\tl_to_str:n{#1}~to~\stex_file_use:N #1}
 710   }
 711 }{
 712   \cs_new_protected:Nn \stex_file_resolve:Nn {
 713     \str_set:Nn \l__stex_path_str {#2}
 714     \stex_file_set:No #1 \l__stex_path_str
 715     \__stex_path_canonicalize:N #1
 716     \%stex_debug:nn{files}{Set~\tl_to_str:n{#1}~to~\stex_file_use:N #1}
 717   }
 718 }
 719 \cs_generate_variant:Nn \stex_file_resolve:Nn {No, Nx}
 720
 721 \cs_new_protected:Nn \__stex_path_canonicalize:N {
 722   \seq_if_empty:NF #1 {
 723     \seq_pop>NN #1 \l__stex_path_str
 724     \seq_clear:N \l__stex_path_seq
 725     \str_if_empty:NTF \l__stex_path_str {
 726       \seq_map_function:NN #1 \__stex_path_dodots:n
 727       \seq_put_left:Nn \l__stex_path_seq {}
 728     }{
 729       \seq_push:No #1 \l__stex_path_str
 730       \seq_map_function:NN #1 \__stex_path_dodots:n
 731     }
 732     \seq_set_eq:NN #1 \l__stex_path_seq
 733   }
 734 }
 735
 736 \cs_new_protected:Nn \__stex_path_dodots:n {
 737   \str_if_empty:nF{#1} {
 738     \str_if_eq:nnF {#1} {.} {
 739       \str_if_eq:nnTF {#1} {..} {
 740         \seq_if_empty:NF \l__stex_path_seq {
 741           \seq_pop_right>NN \l__stex_path_seq \l__stex_path_str
 742         }
 743       }
 744     }
 745   }

```

```

743     }{
744         \seq_put_right:Nn \l__stex_path_seq {#1}
745     }
746 }
747 }
748 }

```

(End of definition for `\stex_file_resolve:Nn`. This function is documented on page 140.)

```

\stex_if_file_absolute_p:N
\stex_if_file_absolute:NTF
749
750 \sys_if_platform_windows:TF {
751     \prg_new_conditional:Nnn \stex_if_file_absolute:N {p, T, F, TF} {
752         \seq_if_empty:NTF #1 \prg_return_false: {
753             \tl_set:Nx \l__stex_path_maybein_str {\seq_item:Nn #1 1}
754             \exp_args:No \tl_if_empty:nTF \l__stex_path_maybein_str \prg_return_true: {
755                 \exp_args:Nx \str_if_eq:nnTF {\str_item:Nn \l__stex_path_maybein_str 2} :
756                 \prg_return_true: \prg_return_false:
757             }
758         }
759     }
760 }{
761     \prg_new_conditional:Nnn \stex_if_file_absolute:N {p, T, F, TF} {
762         \seq_if_empty:NTF #1 \prg_return_false: {
763             \exp_args:Nx \tl_if_empty:nTF {\seq_item:Nn #1 1}
764             \prg_return_true: \prg_return_false:
765         }
766     }
767 }

```

(End of definition for `\stex_if_file_absolute:NTF`. This function is documented on page 140.)

```

\stex_file_use:N
768 \cs_new:Nn \stex_file_use:N {
769     \seq_use:Nn #1 /
770 }

```

(End of definition for `\stex_file_use:N`. This function is documented on page 140.)

```

stex_if_file_starts_with:NNTF
771 \prg_new_protected_conditional:Nnn \stex_if_file_starts_with:NN {T,F,TF} {
772     \seq_set_eq:NN \l__stex_path_a_seq #1
773     \seq_set_eq:NN \l__stex_path_b_seq #2
774     \tl_clear:N \l__stex_path_return_tl
775     \bool_while_do:nn{
776         \bool_not_p:n{
777             \bool_lazy_any_p:n{
778                 {\seq_if_empty_p:N \l__stex_path_a_seq}
779                 {\seq_if_empty_p:N \l__stex_path_b_seq}
780                 {\bool_not_p:n{\tl_if_empty_p:N \l__stex_path_return_tl}}
781             }
782         }
783     }{
784         \seq_pop_left:NN \l__stex_path_a_seq \l__stex_path_a_tl
785         \seq_pop_left:NN \l__stex_path_b_seq \l__stex_path_b_tl

```

```

786   \str_if_eq:NNF \l__stex_path_a_tl \l__stex_path_b_tl {
787     \tl_set:Nn \l__stex_path_return_tl {\prg_return_false:}
788   }
789 }
790 \tl_if_empty:NTF \l__stex_path_return_tl {
791   \seq_if_empty:NTF \l__stex_path_b_seq \prg_return_true: \prg_return_false:
792 } \l__stex_path_return_tl
793 }
```

(End of definition for `\stex_if_file_starts_with:NNTF`. This function is documented on page 140.)

```
\stex_file_split_off_ext:NN
\stex_file_split_off_lang:NN
```

```

794 \cs_new_protected:Nn \stex_file_split_off_ext:NN {
795   \seq_set_eq:NN #1 #2
796   \seq_pop_right:NN #1 \l__stex_path_str
797   \seq_set_split:NnV \l__stex_path_seq . \l__stex_path_str
798   \seq_pop_right:NN \l__stex_path_seq \l__stex_path_str
799   \seq_put_right:Nx #1 {\seq_use:Nn \l__stex_path_seq .}
800 }
801 \cs_new_protected:Nn \stex_file_split_off_lang:NN {
802   \seq_set_eq:NN #1 #2
803   \seq_pop_right:NN #1 \l__stex_path_str
804   \seq_set_split:NnV \l__stex_path_seq . \l__stex_path_str
805   \seq_pop_right:NN \l__stex_path_seq \l__stex_path_str
806
807   \seq_pop_right:NN \l__stex_path_seq \l__stex_path_str
808   \exp_args:NNo \prop_if_in:NnF \c_stex_languages_prop \l__stex_path_str {
809     \seq_put_right:No \l__stex_path_seq \l__stex_path_str
810   }
811
812   \seq_put_right:Nx #1 {\seq_use:Nn \l__stex_path_seq .}
813 }
```

(End of definition for `\stex_file_split_off_ext:NN` and `\stex_file_split_off_lang:NN`. These functions are documented on page 140.)

URIs:

```
\stex_map_uri:Nnnnn
```

```

814 \cs_set_protected:Nn \__stex_path_auth:n {
815   \msg_error:nnx{stex}{error/misused-uri}{\tl_to_str:n{#1}}
816 }
817 \cs_set_eq:NN \__stex_path_path:n \__stex_path_auth:n
818 \cs_set_eq:NN \__stex_path_module:n \__stex_path_auth:n
819 \cs_set_eq:NN \__stex_path_name:n \__stex_path_auth:n
820
821 \cs_set_protected:Nn \stex_map_uri:Nnnnn{
822   \stex_pseudogroup_with:nn{\__stex_path_auth:n\__stex_path_path:n\__stex_path_module:n\__st
823     \cs_set:Npn \__stex_path_auth:n {##1 {#2}}
824     \cs_set:Npn \__stex_path_path:n {##1 {#3}}
825     \cs_set:Npn \__stex_path_module:n {##1 {#4}}
826     \cs_set:Npn \__stex_path_name:n {##1 {#5}}
827     #1
828   }
829 }
```

(End of definition for `\stex_map_uri:Nnnnn`. This function is documented on page 141.)

```
\stex_uri_set:Nn
\stex_uri_set:Nn
\stex_uri_set:Nx
830 \str_set:Nx\__stex_path_colonslash{\cColonStr}
831 \cs_new_protected:Nn \__stex_path_uri_set:NnN {
832   \str_if_empty:nTF {#2} {
833     \msg_error:nnxx{stex}{error/invalid-uri}{\tl_to_str:n{#2}}{empty}
834   }{
835     \exp_args:NNNo \exp_args:NNno \seq_set_split:Nnn #1 \__stex_path_colonslash { \tl_to_str
836     \seq_pop_left:NN #1 \l__stex_path_auth_str
837     \seq_if_empty:NTF #1 {
838       \msg_error:nnxx{stex}{error/invalid-uri}{\tl_to_str:n{#2}}{missing~authority}
839     }{
840       \exp_args:NNnx \seq_set_split:Nnn #1 ? {\exp_args:NNo \seq_use:Nn #1 \__stex_path_colon
841       \seq_pop_left:NN #1 \l__stex_path_path
842       #3 \l__stex_path_path \l__stex_path_path
843       \seq_if_empty:NTF #1 {
844         \exp_args:NNo \__stex_path_uri_set:Nnxnn #1 \l__stex_path_auth_str
845         {\stex_file_use:N \l__stex_path_path} {} {}
846       }{
847         \seq_pop_left:NN #1 \l__stex_path_mod
848         \seq_if_empty:NTF #1 {
849           \exp_args:NNo \__stex_path_uri_set:Nnxon #1 \l__stex_path_auth_str
850           {\stex_file_use:N \l__stex_path_path} \l__stex_path_mod {}
851         }{
852           \seq_pop_left:NN #1 \l__stex_path_name
853           \seq_if_empty:NTF #1 {
854             \exp_args:NNo \__stex_path_uri_set:Nnxon #2 \l__stex_path_auth_str
855             {\stex_file_use:N \l__stex_path_path} \l__stex_path_mod \l__stex_path_name
856           }{
857             \msg_error:nnxx{stex}{error/invalid-uri}{\tl_to_str:n{#2}}{too~many~?s}
858           }
859         }
860       }
861     }
862   }
863   \stex_debug:nn{uris}{Set~\tl_to_str:n{#1}~to~\stex_uri_use:N #1}
864 }
865
866 \cs_new_protected:Nn \__stex_path_uri_set:Nnnnn{
867   \tl_set:Nn #1 {
868     \__stex_path_auth:n{ #2 }
869     \__stex_path_path:n{ #3 }
870     \__stex_path_module:n{ #4 }
871     \__stex_path_name:n{ #5 }
872   }
873 }
874 \cs_generate_variant:Nn \__stex_path_uri_set:Nnnnn {Nnxnn,Nnxon,Nnxoo}
875
876 \cs_new_protected:Nn \stex_uri_set:Nn {
877   \__stex_path_uri_set:NnN #1 {#2} \stex_file_set:Nn
878 }
879 \cs_generate_variant:Nn \stex_uri_set:Nn {No, Nx}
```

(End of definition for `\stex_uri_set:Nn`. This function is documented on page 141.)

```

\stex_uri_resolve:Nn
\stex_uri_resolve:No
\stex_uri_resolve:Nx
 880 \cs_new_protected:Nn \stex_uri_resolve:Nn {
 881   \__stex_path_uri_set:NnN #1 {#2} \stex_file_resolve:No
 882 }
 883 \cs_generate_variant:Nn \stex_uri_resolve:Nn {No, Nx}

```

(End of definition for `\stex_uri_resolve:Nn`. This function is documented on page 141.)

```

\stex_uri_use:N
 884 \cs_new:Npn \__stex_path_uri_use:w \__stex_path_auth:n #1 \__stex_path_path:n #2 \__stex_path_
 885   #1\c_colon_str/ #2 \tl_if_empty:nF { #3 }{ ? #3
 886     \tl_if_empty:nF { #4 }{ ? #4 } }
 887 }
 888 \cs_new:Nn \stex_uri_use:N {
 889   \exp_args:Ne \cs_if_eq:NNTF { \tl_head:N #1 } \__stex_path_auth:n {
 890     \exp_after:wn \__stex_path_uri_use:w #1
 891   }{
 892     \msg_error:nnnn{stex}{error/invalid-uri}{#1}{Not~a~URI}
 893   }
 894 }

```

(End of definition for `\stex_uri_use:N`. This function is documented on page 141.)

```

\stex_uri_from_repo_file>NNNn
\stex_uri_from_repo_file_nolang:NNNn
 895 \cs_new_protected:Npn \stex_uri_from_repo_file_nolang:NNNn {
 896   \__stex_path_from_repo_file:NNNNn \stex_file_split_off_lang:NN
 897 }
 898 \cs_new_protected:Npn \stex_uri_from_repo_file:NNNn {
 899   \__stex_path_from_repo_file:NNNNn \stex_file_split_off_ext:NN
 900 }
 901
 902 \cs_new_protected:Nn \__stex_path_from_repo_file:NNNNn {
 903   #1 \l__stex_path_file #4
 904   \prop_if_exist:NTF #3 {
 905     \str_clear:N \l__stex_path_uri
 906     \prop_get:NnNF #3 {#5} \l__stex_path_uri {
 907       \prop_get:NnNF #3 {ns} \l__stex_path_uri {
 908         \__stex_path_uri_set:Nnxnn #2 {file}
 909         {\stex_file_use:N \l__stex_path_file} {} {}
 910       }
 911     }
 912     \str_if_empty:NF \l__stex_path_uri {\__stex_path_relativize:N #2}
 913   }{
 914     \exp_args:NNx \__stex_path_uri_set:Nnxnn #2 {\tl_to_str:n{file}}
 915     {\stex_file_use:N \l__stex_path_file} {} {}
 916   }
 917 }
 918 \cs_new_protected:Nn \__stex_path_relativize:N {
 919   \seq_set_eq:NN \l__stex_path_seq \l__stex_path_file
 920   \seq_map_inline:Nn \c_stex_mathhub_file { % mathhub path
 921     \seq_pop_left:NN \l__stex_path_seq \l__stex_path_tl
 922   }
 923 \stex_file_set:Nx \l__stex_path_path {\prop_item:Nn \l_stex_current_archive_prop {id} }
```

```

925   \seq_map_inline:Nn \l__stex_path_path { % id
926     \seq_pop_left:NN \l__stex_path_seq \l__stex_path_tl
927   }
928   \seq_pop_left:NN \l__stex_path_seq \l__stex_path_tl % source
929
930   \stex_uri_set:Nx #1 { \l__stex_path_uri / \stex_file_use:N \l__stex_path_seq }
931 }

```

(End of definition for `\stex_uri_from_repo_file:Nnn` and `\stex_uri_from_repo_file_nolang:Nnn`.  
These functions are documented on page 141.)

```

\stex_uri_from_current_file:Nn
\stex_uri_from_current_file_nolang:Nn
932 \cs_new_protected:Nn \stex_uri_from_current_file:Nn {
933   \stex_debug:nn{docuri}{Here:~\stex_file_use:N \g_stex_current_file}
934   \stex_uri_from_repo_file:Nnn #1 \l_stex_current_archive_prop
935   \g_stex_current_file {#2}
936   \stex_debug:nn{docuri}{resolved:~\stex_uri_use:N #1}
937 }
938 \cs_new_protected:Nn \stex_uri_from_current_file_nolang:Nn {
939   \stex_uri_from_repo_file_nolang:Nnn #1 \l_stex_current_archive_prop
940   \g_stex_current_file {#2}
941 }

```

(End of definition for `\stex_uri_from_current_file:Nn` and `\stex_uri_from_current_file_nolang:Nn`.  
These functions are documented on page 142.)

```

\stex_uri_add_module:Nnn
\stex_uri_add_module:NNo
942 \cs_new_protected:Nn \stex_uri_add_module:Nnn {
943   \exp_args:Ne \cs_if_eq:NNTF { \tl_head:N #2 } \__stex_path_auth:n {
944     \stex_pseudogroup_with:nn
945     {\__stex_path_auth:n\__stex_path_path:n\__stex_path_module:n\__stex_path_name:n}
946     {
947       \cs_set:Npn \__stex_path_module:n ##1 {
948         \tl_if_empty:nTF{##1} {
949           \exp_not:N \__stex_path_module:n {#3}
950         }{
951           \msg_error:nnn{stex}{error/invalid-dpath}{#2}
952         }
953       }
954       \cs_set:Npn \__stex_path_name:n ##1 {
955         \tl_if_empty:nTF{##1} {
956           \exp_not:N \__stex_path_name:n {}
957         }{
958           \msg_error:nnn{stex}{error/invalid-dpath}{#2}
959         }
960       }
961       \tl_set:Nx #1 {#2}
962     }
963   }{
964     \msg_error:nnnn{stex}{error/invalid-uri}{#2}{Not~a~URI}
965   }
966 }
967 \cs_generate_variant:Nn \stex_uri_add_module:Nnn {NNo}

```

(End of definition for `\stex_uri_add_module:Nnn`. This function is documented on page 142.)

```
\l_stex_current_doc_uri
```

```
968 \tl_new:N \l_stex_current_doc_uri
```

(End of definition for `\l_stex_current_doc_uri`. This variable is documented on page 142.)

```
\stex_set_document_uri:
```

```
969 \cs_new_protected:Nn \stex_set_document_uri: {  
970     \stex_uri_from_current_file:Nn \l_stex_current_doc_uri {narr}  
971     \%stex_debug:nn{sref}{Document~URI:~\stex_uri_use:N \l_stex_current_doc_uri}  
972 }
```

(End of definition for `\stex_set_document_uri`. This function is documented on page 142.)

```
\stex_set_current_namespace:
```

```
973 \cs_new_protected:Nn \stex_set_current_namespace: {  
974     \stex_uri_from_current_file_nolang:Nn \l_stex_current_ns_uri {source-base}  
975     \%stex_debug:nn{modules}{Namespace~URI:~\stex_uri_use:N \l_stex_current_ns_uri}  
976 }
```

(End of definition for `\stex_set_current_namespace`. This function is documented on page 142.)

We determine the PWD

```
\c_stex_pwd_file
```

```
\c_stex_main_file
```

```
977 \sys_if_platform_windows:TF{  
978     \stex_get_env:Nn\l_stex_path_str{CD}  
979 }{  
980     \stex_get_env:Nn\l_stex_path_str{PWD}  
981 }  
982 \stex_file_resolve:No \c_stex_pwd_file \l_stex_path_str  
983 \seq_set_eq:NN \c_stex_main_file \c_stex_pwd_file  
984 \seq_put_right:Nx \c_stex_main_file {\jobname\tl_to_str:n{.tex}}  
985  
986 \stex_debug:nn {files} {PWD:~\stex_file_use:N \c_stex_pwd_file}
```

(End of definition for `\c_stex_pwd_file` and `\c_stex_main_file`. These variables are documented on page 141.)

### 13.2.10 File Hooks

keeps track of file changes:

```
987 \seq_gclear_new:N\g__stex_path_stack  
988 \seq_gclear_new:N\g_stex_current_file
```

```
\stex_filestack_push:n
```

```
989 \cs_new_protected:Nn \stex_filestack_push:n {  
990     \stex_str_if_ends_with:nnTF {#1}{.tex}{  
991         \stex_file_resolve:No \g_stex_current_file {#1}  
992     }{  
993         \stex_file_resolve:No \g_stex_current_file {#1.tex}  
994     }  
995     \stex_if_file_absolute:NF \g_stex_current_file {  
996         \stex_file_resolve:Nx \g_stex_current_file {  
997             \stex_file_use:N \c_stex_pwd_file / \stex_file_use:N \g_stex_current_file  
998         }
```

```

999 }
1000 \seq_gset_eq:NN \g_stex_current_file \g_stex_current_file
1001 \exp_args:NNx \seq_gpush:Nn \g__stex_path_stack {\stex_file_use:N \g_stex_current_file}
1002 \_stex_every_file:
1003 }
1004
1005 \cs_new_protected:Nn \_stex_every_file: {
1006   \stex_set_document_uri:
1007   \stex_language_from_file:
1008   \stex_set_current_namespace:
1009 }
1010 \%AtBeginDocument{\_stex_every_file:}

(End of definition for \stex_filestack_push:n. This function is documented on page 140.)

```

### \stex\_filestack\_pop:

```

1011 \cs_new_protected:Nn \stex_filestack_pop: {
1012   \seq_if_empty:NF \g__stex_path_stack {
1013     \seq_gpop>NN \g__stex_path_stack \l__stex_path_str
1014   }
1015   \seq_if_empty:NTF \g__stex_path_stack {
1016     \seq_gset_eq:NN \g_stex_current_file \c_stex_main_file
1017   }
1018   \seq_get>NN \g__stex_path_stack \l__stex_path_str
1019   \exp_args:NNo \stex_file_set:Nn \g_stex_current_file \l__stex_path_str
1020   \seq_gset_eq:NN \g_stex_current_file \g_stex_current_file
1021 }
1022 \_stex_every_file:
1023 }

(End of definition for \stex_filestack_pop:. This function is documented on page 140.)

```

Hooks for the current file:

```

1024 \cs_new_protected:Nn \stex_input_with_hooks:n {
1025   \tl_set:Nn \l__stex_path_do_hooks_pre_tl {
1026     \tl_gset:Nn \l__stex_path_do_hooks_pre_tl {}
1027     \stex_debug:nn{HERE}{Hook~for~#1^~J\meaning\CurrentFilePath^~J\CurrentFile}
1028     \tl_if_empty:NTF\CurrentFilePath{
1029       \exp_args:No \stex_filestack_push:n {\CurrentFile}
1030     }
1031     \exp_args:Ne \stex_filestack_push:n { \CurrentFilePath / \CurrentFile }
1032   }
1033 }
1034 \input{#1}
1035 \stex_debug:nn{HERE}{Hook~end~for~#1}
1036 \stex_filestack_pop:
1037 }
1038 \tl_new:N \l__stex_path_do_hooks_pre_tl {}
1039
1040 \AddToHook{file/before}{
1041   \l__stex_path_do_hooks_pre_tl
1042 }
1043 \%AddToHook{file/after}{ \stex_filestack_pop: }


```

### 13.3 Math Archives

```

1044 <@@=stex_mathhub>
\mathhub The path to the mathhub directory. If the \mathhub-macro is not set, we query
\c_stex_home_file kpsewhich for the MATHHUB system variable.
\c_stex_mathhub_file
1045
1046 \sys_if_platform_windows:TF{
1047   \stex_get_env:Nn \l_stex_mathhub_str {homedrive\c_percent_str\c_percent_str homepath}
1048 }{
1049   \stex_get_env:Nn \l_stex_mathhub_str {HOME}
1050 }
1051 \stex_file_resolve:No \c_stex_home_file \l_stex_mathhub_str
1052
1053 \str_if_empty:NTF\mathhub{
1054   \stex_get_env:Nn \l_stex_mathhub_str {MATHHUB}
1055   \str_if_empty:NTF \l_stex_mathhub_str {
1056     \ior_open:NnTF \g_tmpa_iор{\stex_file_use:N \c_stex_home_file/.stex/mathhub.path}{

1057       \group_begin:
1058         \escapechar=-\catcode'\\=12
1059         \ior_str_get:NN \g_tmpa_iор \l_stex_mathhub_str
1060         \str_gset_eq:NN \l_stex_mathhub_str \l_stex_mathhub_str
1061       \group_end:
1062         \ior_close:N \g_tmpa_iор
1063         \stex_debug:nn{mathhub}{MathHub~directory~determined~from~home~directory}
1064     }{
1065       \str_clear:N \l_stex_mathhub_str
1066     }
1067   }{
1068     \stex_debug:nn{mathhub}{MathHub~directory~determined~from~environment~variable}
1069   }
1070 }{
1071   \str_set_eq:NN \l_stex_mathhub_str \mathhub
1072 }
1073
1074 \str_if_empty:NTF \l_stex_mathhub_str {
1075   \msg_warning:nn{stex}{warning/nomathhub}
1076   \exp_args:NNe \stex_file_set:Nn \c_stex_mathhub_file {\stex_file_use:N \c_stex_home_file \
1077 }{
1078   \stex_file_resolve:No \c_stex_mathhub_file \l_stex_mathhub_str
1079   \stex_if_file_absolute:NF \c_stex_mathhub_file {
1080     \exp_args:NNe \stex_file_resolve:Nn \c_stex_mathhub_file {
1081       \stex_file_use:N \c_stex_main_file / .. / \l_stex_mathhub_str
1082     }
1083   }
1084 }
1085
1086 \exp_args:NNe \str_set:Nn \mathhub {\stex_file_use:N \c_stex_mathhub_file}
1087 \stex_debug:nn{mathhub}{MATHHUB:~\mathhub}

(End of definition for \mathhub, \c_stex_home_file, and \c_stex_mathhub_file. These variables are
documented on page ??.)
```

```

\l_stex_current_archive
\stex_set_current_archive:n
1088 \cs_new_protected:Nn \stex_set_current_archive:n {
```

```

1089 \stex_require_archive:n { #1 }
1090 \stex_debug:nn{mathhub}{switching-to-archive-#1}
1091 \prop_set_eq:Nc \l_stex_current_archive_prop {
1092   c_stex_mathhub_#1_manifest_prop
1093 }
1094 }
```

(End of definition for `\l_stex_current_archive` and `\stex_set_current_archive:n`. These variables are documented on page ??.)

`\stex_in_archive:nn`

```

1095 \cs_new_protected:Nn \stex_in_archive:nn {
1096   \cs_if_exist:NF \l__stex_mathhub_cs {
1097     \cs_set:Npn \l__stex_mathhub_cs ##1 {}
1098   }
1099   \stex_pseudogroup:nn{
1100     \cs_set:Npn \l__stex_mathhub_cs ##1 {#2}
1101     \tl_if_empty:nTF{#1} {
1102       \prop_if_exist:NTF \l_stex_current_archive_prop {
1103         \exp_args:Ne \l__stex_mathhub_cs {\prop_item:Nn \l_stex_current_archive_prop { id }
1104       }
1105       \l__stex_mathhub_cs {}
1106     }
1107   }{
1108     \stex_set_current_archive:n{#1}
1109     \l__stex_mathhub_cs {#1}
1110   }
1111 }{
1112   \stex_pseudogroup_restore:N \l_stex_current_archive_prop
1113   \cs_set:Npn \exp_not:N \l__stex_mathhub_cs ##1 {
1114     \exp_args:No \exp_not:n {\l__stex_mathhub_cs {##1}}
1115   }
1116 }
1117 }
```

(End of definition for `\stex_in_archive:nn`. This function is documented on page 137.)

`\stex_require_archive:n`

`\stex_require_archive:o`

```

1118 \cs_new_protected:Nn \stex_require_archive:n {
1119   \prop_if_exist:cF { c_stex_mathhub_#1_manifest_prop } {
1120     \seq_if_empty:NTF \c_stex_mathhub_file {
1121       \msg_fatal:nn{stex}{warning/nomathhub}
1122     }{
1123       \stex_debug:nn{mathhub}{Opening-archive:~#1}
1124       \__stex_mathhub_do_manifest:n { #1 }
1125     }
1126   }
1127 }
1128 \cs_generate_variant:Nn \stex_require_archive:n {o}
```

(End of definition for `\stex_require_archive:n`. This function is documented on page 137.)

Code for finding and parsing manifest files:

```

1129 \cs_new_protected:Nn \__stex_mathhub_do_manifest:n {
1130   \exp_args:Ne \__stex_mathhub_find_manifest:n {\stex_file_use:N \c_stex_mathhub_file / #1}
```

```

1131 \str_if_empty:NT \l_stex_mathhub_manifest_str {
1132   \msg_fatal:nne{stex}{error/noarchive}
1133   {#1}{\stex_file_use:N \c_stex_mathhub_file}
1134 }
1135 \__stex_mathhub_parse_manifest:n {#1}
1136 }

1137 \cs_new_protected:Nn \__stex_mathhub_find_manifest:n {
1138   \str_clear:N \l_stex_mathhub_manifest_str
1139   \seq_set_split:Nnn \l_stex_mathhub_seq / {#1}
1140   \bool_set_true:N \l_stex_mathhub_bool
1141   \bool_while_do:Nn \l_stex_mathhub_bool {
1142     \tl_if_eq:NNTF \l_stex_mathhub_seq \c_stex_mathhub_file {
1143       \bool_set_false:N \l_stex_mathhub_bool
1144     }{
1145       \__stex_mathhub_check_manifest:
1146       \bool_if:NT \l_stex_mathhub_bool {
1147         \seq_pop_right:NN \l_stex_mathhub_seq \l_stex_mathhub_t1
1148       }
1149     }
1150   }
1151 }
1152 }

1153 \cs_generate_variant:Nn \__stex_mathhub_find_manifest:n {x}
1154

1155 \cs_new_protected:Nn \__stex_mathhub_check_manifest: {
1156   \__stex_mathhub_check_manifest:n {MANIFEST.MF}
1157   \bool_if:NT \l_stex_mathhub_bool {
1158     \__stex_mathhub_check_manifest:n {META-INF/MANIFEST.MF}
1159     \bool_if:NT \l_stex_mathhub_bool {
1160       \__stex_mathhub_check_manifest:n {meta-inf/MANIFEST.MF}
1161     }
1162   }
1163 }

1164 \cs_new_protected:Nn \__stex_mathhub_check_manifest:n {
1165   \stex_debug:nn{mathhub}{Checking~\stex_file_use:N \l_stex_mathhub_seq / #1}
1166   \file_if_exist:nT {\stex_file_use:N \l_stex_mathhub_seq / #1} {
1167     \bool_set_false:N \l_stex_mathhub_bool
1168     \str_set:Nx \l_stex_mathhub_manifest_str {\stex_file_use:N \l_stex_mathhub_seq / #1}
1169   }
1170 }

1171 }

1172 \ior_new:N \c_stex_mathhub_manifest_ior
1173 \cs_new_protected:Nn \__stex_mathhub_parse_manifest:n {
1174   \ior_open:Nn \c_stex_mathhub_manifest_ior \l_stex_mathhub_manifest_str
1175   \prop_clear:N \l_stex_mathhub_prop
1176   \ior_map_inline:Nn \c_stex_mathhub_manifest_ior {
1177     \exp_args:NNNo \exp_args:NNN
1178     \seq_set_split:Nnn \l_stex_mathhub_seq \c_colon_str {\tl_to_str:n{##1}}
1179     \seq_pop_left:NNT \l_stex_mathhub_seq \l_stex_mathhub_key {
1180       \exp_args:NNo \str_set:Nn \l_stex_mathhub_key \l_stex_mathhub_key
1181       \str_set:Nx \l_stex_mathhub_val {\seq_use:Nn \l_stex_mathhub_seq :}
1182       \str_case:Nn \l_stex_mathhub_key {
1183         {id} \prop_put:Nno \l_stex_mathhub_prop { id } \l_stex_mathhub_
1184       }
1185     }
1186   }
1187 }
```

```

1185     {narration-base}  {\prop_put:Nno \l_stex_mathhub_prop { narr }      \l_stex_mathhub_
1186     {url-base}        {\prop_put:Nno \l_stex_mathhub_prop { docurl } } \l_stex_mathhub_
1187     {source-base}     {\prop_put:Nno \l_stex_mathhub_prop { ns } }       \l_stex_mathhub_
1188     {ns}              {\prop_put:Nno \l_stex_mathhub_prop { ns } }       \l_stex_mathhub_
1189   }
1190 }
1191 }
1192 \ior_close:N \c_stex_mathhub_manifest_ior
1193 \prop_gset_eq:cN { c_stex_mathhub_#1_manifest_prop } \l_stex_mathhub_prop
1194 \stex_debug:nn{mathhub}{Result:~\prop_to_keyval:N \l_stex_mathhub_prop}
1195 \stex_persist:e {
1196   \prop_gset_from_keyval:cn {c_stex_mathhub_#1_manifest_prop} {
1197     \prop_to_keyval:N \l_stex_mathhub_prop
1198   }
1199 }
1200 }

```

Current MathHub archive

```

\c_stex_main_archive_prop
\l_stex_current_archive_prop
1201 \cs_new_protected:Nn \stex_main_archive: {
1202   \stex_if_file_starts_with:NNTF \c_stex_pwd_file \c_stex_mathhub_file {
1203     \__stex_mathhub_find_manifest:x { \stex_file_use:N \c_stex_pwd_file }
1204     \str_if_empty:NTF \l_stex_mathhub_manifest_str {
1205       \stex_debug:nn{mathhub}{Not~currently~in~a~MathHub~archive}
1206     }{
1207       \__stex_mathhub_parse_manifest:n { main }
1208       \prop_set_eq:NN \c_stex_main_archive_prop \c_stex_mathhub_main_manifest_prop
1209       \cs_undefine:N \c_stex_mathhub_main_manifest_prop
1210       \prop_get:NnN \c_stex_main_archive_prop {id}
1211         \l_stex_mathhub_str
1212       \prop_set_eq:cN { c_stex_mathhub_\l_stex_mathhub_str _manifest_prop }
1213         \c_stex_main_archive_prop
1214       \exp_args:No \stex_set_current_archive:n { \l_stex_mathhub_str }
1215       \stex_debug:nn{mathhub}{Current~archive:~}
1216         \prop_item:Nn \l_stex_current_archive_prop {id}
1217     }
1218   \bool_if:NT \c_stex_persist_write_mode_bool {
1219     \tl_put_right:Nx \stex_persist_read_now: {
1220       \stex_persist:n {
1221         \prop_gset_from_keyval:cn {c_stex_mathhub_\l_stex_mathhub_str _manifest_prop} {
1222           \prop_to_keyval:N \c_stex_main_archive_prop
1223         }
1224         \prop_gset_eq:Nc \exp_not:N \l_stex_current_archive_prop {
1225           c_stex_mathhub_\l_stex_mathhub_str _manifest_prop
1226         }
1227         \prop_gset_eq:Nc \exp_not:N \c_stex_main_archive_prop {
1228           c_stex_mathhub_\l_stex_mathhub_str _manifest_prop
1229         }
1230       }
1231     }
1232   }
1233 }
1234 }
1235 \stex_debug:nn{mathhub}{Not~currently~in~the~MathHub~directory}

```

```

1236     }
1237 }
1238 \%\\bool_if:NF \\c_stex_persist_mode_bool {
1239   \\_stex_main_archive:
1240 }
1241 }
```

(End of definition for `\c_stex_main_archive_prop` and `\l_stex_current_archive_prop`. These variables are documented on page 137.)

## 13.4 Documents

### 13.4.1 Title

Stores the title, if it exists:

```

1242 <@=stex_doc>
1243 \\tl_new:N \\g__stex_doc_title_tl
```

`\stexdoctitle` Initial definition, will be changed at begin document:

```

1244 \\cs_new_protected:Npn \\stexdoctitle #1 {
1245   \\tl_gset:Nn \\g__stex_doc_title_tl { #1 }
1246   \\global\\def\\stexdoctitle##1{}
1247 }
```

At begin document, we switch to:

```

1248 \\cs_new_protected:Nn \\__stex_doc_set_title:n {
1249   \\stex_if_smsmode:F{
1250     \\global\\def\\stexdoctitle##1{}
1251     \\stex_debug:nn{title}{Setting~title~to:\\tl_to_str:n{#1}}
1252     \\tl_gset:Nn \\g__stex_doc_title_tl { #1 }
1253     \\__stex_doc_title_html:
1254   }
1255 }
```

Hooks, changes and HTML:

```

1256 \\cs_new_protected:Nn \\__stex_doc_title_html: {
1257   \\stex_if_do_html:T{\\stex_if_html_backend:T{
1258     \\stex_annotation_invisible:nn{data-shtml-doctype={}{{ \\hbox{\\g__stex_doc_title_tl} }}}
1259   }}
1260 }
1261
1262 \\AtBeginDocument {
1263   \\tl_if_empty:NTF \\g__stex_doc_title_tl {
1264     \\cs_set_eq:NN \\stexdoctitle \\__stex_doc_set_title:n
1265   }
1266   \\stex_debug:nn{title}{Setting~title~to:\\exp_args:No\\tl_to_str:n\\g__stex_doc_title_tl}
1267   \\global\\def\\stexdoctitle#1{}
1268   \\__stex_doc_title_html:
1269 }
1270
1271 \\cs_set_eq:NN \\__stex_doc_maketitle: \\maketitle
1272 \\global\\protected\\def\\maketitle{
1273   \\tl_if_empty:NF \\@title {
1274     \\exp_args:No \\stexdoctitle \\@title
```

```

1275     }
1276     \__stex_doc_maketitle:
1277   }
1278 }
```

(End of definition for `\stexdoctitle`. This function is documented on page ??.)

### 13.4.2 Sectioning

```

1279 \int_new:N \l_stex_docheader_sect
1280
1281 \tl_set:Nn \stex_current_section_level {document}
1282
1283 \cs_set_protected:Npn \currentsectionlevel {
1284   \stex_if_do_html:TF{
1285     \stex_annotate:nn{data-shtml-currentsectionlevel={},data-shtml-capitalize=false}{}
1286   }{
1287     \stex_current_section_level
1288   }
1289   \tl_if_exist:NT\xspace\xspace
1290 }
1291
1292 \cs_set_protected:Npn \Currentsectionlevel {
1293   \stex_if_do_html:TF{
1294     \stex_annotate:nn{data-shtml-currentsectionlevel={},data-shtml-capitalize=true}{}
1295   }{
1296     \exp_args:No \stex_capitalize:n \stex_current_section_level
1297   }
1298   \tl_if_exist:NT\xspace\xspace
1299 }
1300
1301 \stex_if_html_backend:TF {
1302   \cs_new_protected:Nn \_sfragment_do_level:nn {
1303     \stexdoctitle{#2}
1304     \par
1305     \begin{stex_env_node}{section}{data-shtml-section=\int_use:N \l_stex_docheader_sect}
1306       \noindent\stex_html_node:nnn{h1}{data-shtml-title={}}{
1307         \stex_annotate_force_break:n{#2}
1308       }\par
1309     }
1310     \cs_new_protected:Nn \_sfragment_end: {
1311       \end{stex_env_node}
1312     }
1313   }{
1314     \cs_new_protected:Nn \_sfragment_do_level:nn {
1315       \stexdoctitle{#2}
1316       \tl_if_empty:NTF \l_stex_key_short_tl {
1317         \use:c{#1}
1318       }{
1319         \exp_args:Nnx \use:nn{\use:c{#1}}{[\exp_args:No \exp_not:n \l_stex_key_short_tl]}
1320       }{#2}
1321       \int_incr:N \l_stex_docheader_sect
1322       \tl_set:Nn \stex_current_section_level{#1}
1323     }
1324 }
```

```

1324   \cs_new_protected:Nn \_sfragment_end: {}
1325 }
1326
1327
1328 \cs_new_protected:Npn \__stex_doc_do_section:n {
1329   \int_case:nnF \l_stex_docheader_sect {
1330     {0}{\cs_if_exist:NTF \thechapter {\_sfragment_do_level:nn{part}}{
1331       \int_incr:N \l_stex_docheader_sect
1332       \__stex_doc_do_section:n
1333     }}
1334     {1}{\cs_if_exist:NTF \thechapter {\_sfragment_do_level:nn{chapter}}{
1335       \int_incr:N \l_stex_docheader_sect
1336       \__stex_doc_do_section:n
1337     }}
1338     {2}{\_sfragment_do_level:nn{section}}
1339     {3}{\_sfragment_do_level:nn{subsection}}
1340     {4}{\_sfragment_do_level:nn{subsubsection}}
1341     {5}{\_sfragment_do_level:nn{paragraph}}
1342   }{\_sfragment_do_level:nn{subparagraph}}
1343 }
1344
1345 \stex_keys_define:nnnn{ sfragment }{
1346   \tl_clear:N \l_stex_key_short_tl
1347 }
1348   short .tl_set:N = \l_stex_key_short_tl
1349 }{id}
1350
1351 \NewDocumentEnvironment{sfragment}{ O{} m} {
1352   \stex_keys_set:nn{ sfragment }{#1}
1353   \__stex_doc_do_section:n{#2}
1354   \_stex_do_id:
1355 }
1356   \_sfragment_end:
1357 }
1358
1359 \bool_new:N \l__stex_doc_titlefragment_bool
1360
1361 \cs_new_protected:Nn \_stex_titlefragment: {
1362   \maketitle
1363 }
1364
1365
1366 \NewDocumentEnvironment{titlefragment}{ O{} m} {
1367   \stex_keys_set:nn{ sfragment }{#1}
1368   \cs_if_exist:NTF \maketitle {
1369     \bool_set_true:N \l__stex_doc_titlefragment_bool
1370     \title{#2}
1371     \tl_if_empty:NF \l_stex_key_short_tl {
1372       \cs_if_exist:NT \shorttitle {
1373         \exp_args:No \shorttitle \l_stex_key_short_tl
1374       }
1375     }
1376     \stexdoctitle{#2}
1377   \_stex_titlefragment:

```

```

1378 }{
1379   \bool_set_false:N \l__stex_doc_titlefragment_bool
1380   \__stex_doc_do_section:n{#2}
1381   \_stex_do_id:
1382 }
1383 }{
1384   \bool_if:NF \l__stex_doc_titlefragment_bool \_sfragment_end:
1385 }
1386
1387 \%int_incr:N \l_stex_docheader_sect
1388 \NewDocumentEnvironment{blindfragment}{}{
1389   \__stex_doc_skip_section:
1390 }{
1391   \stex_if_html_backend:T{
1392     \stex_annotation_invisible:n{~}
1393     \end{stex_annotation_env}
1394   }
1395 }
1396
1397
1398 \cs_new_protected:Nn \__stex_doc_skip_section_i: {
1399   \int_case:nn \l_stex_docheader_sect {
1400     {0}{\cs_if_exist:NF \thepart {
1401       \int_incr:N \l_stex_docheader_sect \__stex_doc_skip_section_i:
1402     }}
1403     {1}{\cs_if_exist:NF \thechapter {
1404       \int_incr:N \l_stex_docheader_sect \__stex_doc_skip_section_i:
1405     }}
1406   }
1407   \int_incr:N \l_stex_docheader_sect
1408 }
1409
1410 \stex_if_html_backend:TF {
1411   \cs_new_protected:Nn \__stex_doc_skip_section: {
1412     \__stex_doc_skip_section_i:
1413     \begin{stex_annotation_env}{data-shtml-skipsection=\int_use:N \l_stex_docheader_sect}
1414     \stex_annotation_invisible:n{~}
1415   }
1416 }{
1417   \cs_set_eq:NN \__stex_doc_skip_section: \__stex_doc_skip_section_i:
1418 }
1419
1420
1421 \cs_new_protected:Nn \__stex_doc_skip_fragment:n {
1422   \stepcounter{#1}
1423 }
1424
1425 \cs_new_protected:Npn \skipfragment {
1426   \int_case:nnF \l_stex_docheader_sect {
1427     {0}{\cs_if_exist:NTF \thepart {\__stex_doc_skip_fragment:n{part}}{
1428       \int_incr:N \l_stex_docheader_sect
1429       \skipfragment
1430     }}
1431     {1}{\cs_if_exist:NTF \thechapter {\__stex_doc_skip_fragment:n{chapter}}{

```

```

1432     \int_incr:N \l_stex_docheader_sect
1433     \skipfragment
1434   }}
1435   {2}{\_\_stex_doc_skip_fragment:n{section}}
1436   {3}{\_\_stex_doc_skip_fragment:n{subsection}}
1437   {4}{\_\_stex_doc_skip_fragment:n{subsubsection}}
1438   {5}{\_\_stex_doc_skip_fragment:n{paragraph}}
1439 }{\_\_stex_doc_skip_fragment:n{subparagraph}}
1440 }

\setsectionlevel

1441 \cs_new_protected:Npn \setsectionlevel #1 {
1442   \str_case:nnF{#1} {
1443     {part}{\int_set:Nn \l_stex_docheader_sect 0}
1444     {chapter}{\int_set:Nn \l_stex_docheader_sect 1}
1445     {section}{\int_set:Nn \l_stex_docheader_sect 2}
1446     {subsection}{\int_set:Nn \l_stex_docheader_sect 3}
1447     {subsubsection}{\int_set:Nn \l_stex_docheader_sect 4}
1448     {paragraph}{\int_set:Nn \l_stex_docheader_sect 5}
1449   }{
1450     \int_set:Nn \l_stex_docheader_sect 6
1451   }
1452 \cs_if_eq:NNTF\onlypreamble\@notprerr{
1453   \stex_annotation_invisible:nn{data-shtml-sectionlevel={\int_use:N\l_stex_docheader_sect}}{}}
1454 }{}
1455 }

1456 \stex_if_html_backend:T{
1457   \cs_new_protected:Nn \_\_stex_doc_check_topsect: {
1458     \int_case:nnF \l_stex_docheader_sect {
1459       {0}{\cs_if_exist:NTF \thepart {
1460         \stex_annotation_invisible:nn{data-shtml-sectionlevel=0}{}}
1461       }{
1462         \int_incr:N \l_stex_docheader_sect
1463         \_\_stex_doc_check_topsect:
1464       }
1465     }{
1466       {1}{\cs_if_exist:NTF \thechapter {
1467         \stex_annotation_invisible:nn{data-shtml-sectionlevel=1}{}}
1468       }{
1469         \int_incr:N \l_stex_docheader_sect
1470         \_\_stex_doc_check_topsect:
1471       }
1472     }{
1473       \stex_annotation_invisible:nn{data-shtml-sectionlevel={\int_use:N\l_stex_docheader_sect}}
1474     }
1475   }
1476   \AtBeginDocument{\_\_stex_doc_check_topsect:}
1477 }

1478 \AtBeginDocument{
1479   \bool_if:NF \c_stex_no_frontmatter_bool {
1480     \cs_if_exist:NTF\frontmatter{
1481       \let\_\_stex_doc_orig_frontmatter\frontmatter
1482       \let\frontmatter\relax

```

```

1484    }{
1485      \tl_set:Nn\__stex_doc_orig_frontmatter{
1486        \clearpage
1487        \%@\mainmatterfalse
1488        \pagenumbering{roman}
1489      }
1490    }
1491    \cs_if_exist:NTF\backmatter{
1492      \let\__stex_doc_orig_backmatter\backmatter
1493      \let\backmatter\relax
1494    }{
1495      \tl_set:Nn\__stex_doc_orig_backmatter{
1496        \clearpage
1497        \%@\mainmatterfalse
1498        \pagenumbering{roman}
1499      }
1500    }
1501    \newenvironment{frontmatter}{
1502      \__stex_doc_orig_frontmatter
1503    }{
1504      \cs_if_exist:NTF\mainmatter{
1505        \mainmatter
1506      }{
1507        \clearpage
1508        \%@\mainmattertrue
1509        \pagenumbering{arabic}
1510      }
1511    }
1512    \newenvironment{backmatter}{
1513      \__stex_doc_orig_backmatter
1514    }{
1515      \cs_if_exist:NTF\mainmatter{
1516        \mainmatter
1517      }{
1518        \clearpage
1519        \%@\mainmattertrue
1520        \pagenumbering{arabic}
1521      }
1522    }
1523  }
1524 }

```

(End of definition for `\setsectionlevel`. This function is documented on page 85.)

### 13.4.3 References

```
1525 <@=stex_refs>
```

References are stored in the file `\jobname.sref`, to enable cross-referencing external documents.

```

1526 \bool_if:NT \c_stex_use_sref_bool {
1527   \iow_new:N \c__stex_refs_iow
1528   \AtBeginDocument{\iow_open:Nn \c__stex_refs_iow {\jobname.sref}}
1529   \AtEndDocument{\iow_close:N \c__stex_refs_iow}
1530 }

```

The following macros are written to the .aux-file, and hence use L<sup>A</sup>T<sub>E</sub>X2e character code scheme:

```

1531 \cs_new_protected:Npn \STEXInternalSrefRestoreTarget #1#2#3#4#5 {}
1532
1533 \cs_new_protected:Npn \STEXInternalSetSrefSymURL #1 #2 {
1534   \str_gset:cn{g_stex_sref_sym_\tl_to_str:n{#1}_target}{#2}
1535 }
1536

\stex_ref_new_doc_target:n
  \sreflabel
    1537 \seq_new:N \g__stex_refs_files_seq
    1538 \int_new:N \l__stex_refs_unnamed_counter_int
    1539
    1540 \cs_new_protected:Nn \stex_ref_new_id:n {
      1541   \str_if_empty:nTF {#1} {
        1542     \int_gincr:N \l__stex_refs_unnamed_counter_int
        1543     \str_set:Nx \l__stex_refs_str {REF\int_use:N \l__stex_refs_unnamed_counter_int}
      1544   }{
        1545     \str_set:Nn \l__stex_refs_str {#1}
      1546   }
    1547   \str_set:Nx \l_stex_ref_url_str {\stex_uri_use:N \l_stex_current_doc_uri ? \l__stex_refs_s
    1548 }

    1549 \bool_if:NTF \c_stex_use_sref_bool {
    1550   \cs_new_protected:Nn \stex_ref_new_doc_target:n {
      1551     \stex_ref_new_id:n{#1}
      1552     \%stex_uri_add_module:NNo \l__stex_refs_uri \l_stex_current_doc_uri \l__stex_refs_str
      1553     \%stex_debug:nn{sref}{New-document-target:~\stex_uri_use:N \l__stex_refs_uri}
      1554     \__stex_refs_add_doc_ref:xo {\stex_uri_use:N \l_stex_current_doc_uri} \l__stex_refs_str
      1555     \stex_if_smsmode:F {
        1556       \iow_now:Nx \c__stex_refs_iow {
          1557         \STEXInternalSrefRestoreTarget
          1558         {\stex_uri_use:N \l_stex_current_doc_uri}
          1559         {\l__stex_refs_str}
          1560         {\@currentcounter}
          1561         {\@currentlabel}
          1562         {
            1563           \tl_if_exist:NT\@currentlabelname{
              1564             \exp_args:No\exp_not:n\@currentlabelname
            1565           }
          1566         }
        1567       }
      1568     }
      1569     \exp_args:Nx \label {sref@\l_stex_ref_url_str}
      1570     \stex_if_do_html:T {
        1571       \pdfdest name "sref@\l_stex_ref_url_str" xyz\relax
      1572     }
    1573   }
  1574 }
  1575 }{
  1576   \cs_new_protected:Nn \stex_ref_new_doc_target:n {
    1577     \stex_ref_new_id:n{#1}
  1578   }
  1579 }
  1580 \NewDocumentCommand \sreflabel {m} {\stex_ref_new_doc_target:n {#1}}

```

```

1581 \cs_new_protected:Nn \__stex_refs_add_doc_ref:nn {
1582   \seq_if_in:NnTF \g__stex_refs_files_seq {#1} {
1583     \seq_if_in:cnF {g__stex_refs_#1_seq}{#2} {
1584       \seq_gput_left:cn{g__stex_refs_#1_seq}{#2}
1585     }
1586   }{
1587     \seq_gput_right:Nn \g__stex_refs_files_seq {#1}
1588     \seq_new:c{g__stex_refs_#1_seq}
1589     \seq_gput_left:cn{g__stex_refs_#1_seq}{#2}
1590   }
1591 }
1592 }
1593 \cs_generate_variant:Nn \__stex_refs_add_doc_ref:nn {xo,xx}

```

(End of definition for `\stex_ref_new_doc_target:n` and `\sreflabel`. These functions are documented on page 124.)

`\sref` Optional arguments:

```

\extref 1594 \stex_keys_define:nnnn{sref / 1}{}{
1595   % TODO get rid of this
1596   fallback .code:n = {},
1597   pre .code:n = {},
1598   post .code:n = {}
1599 }{archive file}
1600 \stex_keys_define:nnnn{sref / 2}{}{}{archive file, title}
1601
1602 \str_new:N \l__stex_refs_default_archive_str
1603 \str_new:N \l__stex_refs_default_file_str
1604 \tl_new:N \l__stex_refs_default_title_tl
1605
1606 \cs_set_protected:Nn \__stex_refs_set_keys_b:n {
1607   \tl_if_empty:nTF{#1} {
1608     \str_set_eq:NN \l_stex_key_archive_str \l__stex_refs_default_archive_str
1609     \str_set_eq:NN \l_stex_key_file_str \l__stex_refs_default_file_str
1610     \tl_set_eq:NN \l_stex_key_title_tl \l__stex_refs_default_title_tl
1611   }{
1612     \stex_keys_set:nn{ sref / 2 }{ #1 }
1613   }
1614 }
1615
1616 \newcommand\srefsetin[3][]{%
1617   \str_set:Nx \l__stex_refs_default_archive_str {#1}
1618   \str_set:Nx \l__stex_refs_default_file_str {#2}
1619   \tl_set:Nn \l__stex_refs_default_title_tl {#3}
1620 }
1621

```

Auxiliary methods:

```

1622 \cs_new_protected:Nn \__stex_refs_find_uri:n {
1623   \str_clear:N \l__stex_refs_uri_str
1624   \stex_debug:nn{sref}{
1625     File:~\l_stex_key_file_str^^J
1626     Repo:\l_stex_key_archive_str
1627   }
1628   \str_if_empty:NTF \l_stex_key_file_str {

```

```

1629   \stex_debug:nn{sref}{Empty.\~{}Checking~current~file~for~#1}
1630   \seq_if_exist:cT{g__stex_refs_\stex_uri_use:N \l_stex_current_doc_uri _seq}{
1631     \exp_args:Nnx \__stex_refs_find_uri_in_file:nnn{#1}
1632       {\stex_uri_use:N \l_stex_current_doc_uri}\seq_map_break:
1633   }
1634   \str_if_empty:NT \l__stex_refs_uri_str {
1635     \seq_map_inline:Nn \g__stex_refs_files_seq {
1636       \__stex_refs_find_uri_in_file:nnn{#1}{##1}\{\seq_map_break:n{\seq_map_break:}\}
1637     }
1638   }
1639   \{
1640     \str_if_empty:NTF \l_stex_key_archive_str {
1641       \prop_if_exist:NTF \l_stex_current_archive_prop {
1642         \__stex_refs_find_uri_in_prop_file:N \l_stex_current_archive_prop
1643       }
1644         \stex_file_resolve:Nx \l__stex_refs_file
1645           { \stex_file_use:N \g_stex_current_file / .. / \l_stex_key_file_str }
1646         \str_set:Nx \l__stex_refs_uri_str { file:/ \stex_file_use:N \l__stex_refs_file }
1647       }
1648     \{
1649       \stex_require_archive:o \l_stex_key_archive_str
1650       \prop_set_eq:Nc \l__stex_refs_prop { c_stex_mathhub_\l_stex_key_archive_str _manifest_ }
1651       \__stex_refs_find_uri_in_prop_file:N \l__stex_refs_prop
1652     }
1653   }
1654 }
1655
1656 \cs_new_protected:Nn \__stex_refs_find_uri_in_prop_file:N {
1657   \str_set:Nx \l__stex_refs_uri_str {
1658     \stex_file_use:N \c_stex_mathhub_file /
1659     \prop_item:Nn #1 {id} /
1660       source / \l_stex_key_file_str .sref
1661   }
1662   \stex_file_resolve:No \l__stex_refs_file \l__stex_refs_uri_str
1663   \stex_uri_from_repo_file:NNNn \l__stex_refs_uri #1
1664     \l__stex_refs_file {narr}
1665   \str_set:Nx \l__stex_refs_uri_str {\stex_uri_use:N \l__stex_refs_uri}
1666 }
1667
1668 \cs_new_protected:Nn \__stex_refs_find_uri_in_file:nnn {
1669   \stex_debug:nn{sref}{Checking~file~#2}
1670   \seq_map_inline:cn{g__stex_refs_#2_seq}\{
1671     \str_if_eq:nnT{#1}{##1}\{
1672       \str_set:Nx \l__stex_refs_uri_str {\stex_uri_use:N \l_stex_current_doc_uri}
1673       \stex_debug:nn{sref}{Found.}
1674       #3
1675     }
1676   }
1677 }

```

Doing the actual referencing:

```

1678 \cs_new_protected:Nn \__stex_refs_do_autoref:n {
1679   \cs_if_exist:cTF{autoref}{
1680     \exp_args:Nx\autoref{sref@#1}

```

```

1681    }{
1682      \exp_args:Nx\ref{sref@#1}
1683    }
1684  }
1685
1686 \cs_new_protected:Nn \__stex_refs_do_sref:nn {
1687   \str_if_empty:NTF \l__stex_refs_uri_str {
1688     \str_if_empty:NTF \l_stex_key_file_str {
1689       \stex_debug:nn{sref}{autoref~on~\stex_uri_use:N \l_stex_current_doc_uri?#1}
1690       \exp_args:Ne \__stex_refs_do_autoref:n{\stex_uri_use:N \l_stex_current_doc_uri ? #1}
1691     }{
1692       \stex_debug:nn{sref}{srefin~on~#1}
1693       \__stex_refs_set_keys_b:n{ #2 }
1694       \__stex_refs_do_sref_in:n{#1}
1695     }
1696   }{
1697     \exp_args:NNo \seq_if_in:NnTF \g__stex_refs_files_seq \l__stex_refs_uri_str {
1698       \stex_debug:nn{sref}{Using~ref~file~\l__stex_refs_uri_str}
1699       \exp_args:Nnx \seq_if_in:cnTF{g_stex_ref}\l__stex_refs_uri_str _seq{\detokenize{#1}}{
1700         \stex_debug:nn{sref}{Reference~found~in~ref~files;~autoref~on~\l__stex_refs_uri_str?#1}
1701         \__stex_refs_do_autoref:n{\l__stex_refs_uri_str?#1}
1702       }{
1703         \str_if_empty:NTF \l_stex_key_file_str {
1704           \stex_debug:nn{sref}{in~empty;~autoref~on~\l__stex_refs_uri_str?#1}
1705           \__stex_refs_do_autoref:n{\l__stex_refs_uri_str?#1}
1706         }{
1707           \stex_debug:nn{sref}{in~non~empty;~srefin~on~\l__stex_refs_uri_str?#1}
1708           \__stex_refs_set_keys_b:n{ #2 }
1709           \__stex_refs_do_sref_in:n{#1}
1710         }
1711       }
1712     }{
1713       \stex_debug:nn{sref}{No~ref~file~found~for~\l__stex_refs_uri_str}
1714       \str_if_empty:NTF \l_stex_key_file_str {
1715         \stex_debug:nn{sref}{in~empty;~autoref~on~\l__stex_refs_uri_str?#1}
1716         \__stex_refs_do_autoref:n{\l__stex_refs_uri_str?#1}
1717       }{
1718         \stex_debug:nn{sref}{in~non~empty;~srefin~on~\l__stex_refs_uri_str?#1}
1719         \__stex_refs_set_keys_b:n{ #2 }
1720         \__stex_refs_do_sref_in:n{#1}
1721       }
1722     }
1723   }
1724 }
1725
1726 \cs_new_protected:Nn \__stex_refs_do_sref_in:n {
1727   \stex_debug:nn{sref}{In: \l_stex_key_file_str^JRepo:\l_stex_key_archive_str}
1728   \stex_debug:nn{sref}{URI: \l__stex_refs_uri_str?#1}
1729   \tl_if_exist:cTF{r@sref@\l__stex_refs_uri_str?#1}{
1730     \__stex_refs_do_autoref:n{\l__stex_refs_uri_str?#1}
1731   }{
1732     \%msg_warning:nnn{stex}{warning/smsmissing}{<filename>}
1733     \group_begin:\catcode13=9\relax\catcode10=9\relax
1734       \str_if_empty:NTF \l_stex_key_archive_str {

```

```

1735   \prop_if_exist:NTF \l_stex_current_archive_prop {
1736     \str_set:Nx \l_stex_refs_file_str {
1737       \stex_file_use:N \c_stex_mathhub_file /
1738       \prop_item:Nn \l_stex_current_archive_prop { id }
1739       / source / \l_stex_key_file_str .sref
1740     }
1741   }{
1742     \str_set:Nx \l_stex_refs_file_str {
1743       \stex_file_use:N \g_stex_current_file / .. / \l_stex_key_file_str .sref
1744     }
1745   }{
1746     \str_set:Nx \l_stex_refs_file_str {
1747       \stex_file_use:N \c_stex_mathhub_file / \l_stex_key_archive_str
1748       / source / \l_stex_key_file_str .sref
1749     }
1750   }
1751 \stex_file_resolve:No \l_stex_refs_file \l_stex_refs_file_str
1752 \str_set:Nx \l_stex_refs_file_str {\stex_file_use:N \l_stex_refs_file }
1753 \stex_debug:nn{sref}{File: \l_stex_refs_file_str }
1754 \exp_args:NNNx \exp_args:No \str_if_eq:nnTF \l_stex_refs_file_str {\stex_file_use:N\c
1755   \l_stex_refs_do_autoref:n{
1756     \str_if_empty:NF\l_stex_refs_uri_str{\l_stex_refs_uri_str?}#1
1757   }
1758 }{
1759   \exp_args:No \IfFileExists \l_stex_refs_file_str {
1760     \tl_clear:N \l_stex_refs_return_tl
1761     \str_set:Nn \l_stex_refs_id_str {#1}
1762     \let\STEXInternalSrefRestoreTarget\l_stex_refs_restore_target:nnnn
1763     \use:c{@ @ input}{\l_stex_refs_file_str}
1764     \exp_args:No \tl_if_empty:nTF \l_stex_refs_return_tl {
1765       \exp_args:Nnno \msg_warning:nnnn{stex}{warning/smslabelmissing}\l_stex_refs_file_
1766       \l_stex_refs_do_autoref:n{
1767         \str_if_empty:NF\l_stex_refs_uri_str{\l_stex_refs_uri_str?}#1
1768       }
1769     }
1770   }{
1771     \l_stex_refs_return_tl
1772   }
1773 }{
1774   \exp_args:Nnno \msg_warning:nnn{stex}{warning/smsmissing}\l_stex_refs_file_str
1775   \l_stex_refs_do_autoref:n{
1776     \str_if_empty:NF\l_stex_refs_uri_str{\l_stex_refs_uri_str?}#1
1777   }
1778 }
1779 \group_end:
1780 }
1781 }
1782 }
1783 \cs_new_protected:Nn \l_stex_refs_do_return:nnnn {
1784   \tl_set:Nn \l_stex_refs_return_tl {
1785     \stex_annotation:nn{data-shtml-sref={#4},data-shtml-srefin={\l_stex_refs_file_str}}{
1786       \use:c{#3autorefname}~#1\tl_if_empty:nF{#2}{~(#2)}
1787       \tl_if_empty:NF\l_stex_key_title_tl{

```

```

1789         {}~in~\l_stex_key_title_tl
1790     }
1791   }
1792 }
1793 }
1794
1795 \cs_new_protected:Nn \__stex_refs_restore_target:n { nnnnn {
1796   \str_if_empty:NTF \l_stex_refs_uri_str {
1797     \exp_args:No \str_if_eq:nnT \l_stex_refs_id_str {\#2} {
1798       \__stex_refs_do_return:nnnn{\#4}{\#5}{\#3}{\#1?\#2}
1799     }
1800   }{
1801     \stex_debug:nn{sref}{\l_stex_refs_uri_str{}~ == ~ #1 ~ ?}
1802     \exp_args:No \str_if_eq:nnT \l_stex_refs_uri_str {\#1} {
1803       \stex_debug:nn{sref}{\l_stex_refs_id_str~ == ~ #2 ~ ?}
1804       \exp_args:No \str_if_eq:nnT \l_stex_refs_id_str {\#2} {
1805         \stex_debug:nn{sref}{success!}
1806         \__stex_refs_do_return:nnnn{\#4}{\#5}{\#3}{\#1?\#2}
1807         \endinput
1808       }
1809     }
1810   }
1811 }

```

The actual macros:

```

1812 \NewDocumentCommand \sref { O{} m O{} }{
1813   \stex_keys_set:nn { sref / 1 }{ #1 }
1814   \__stex_refs_find_uri:n { #2 }
1815   \__stex_refs_do_sref:nn{#2}{#3}
1816 }
1817 \NewDocumentCommand \extref { O{} m m }{
1818   \stex_keys_set:nn { sref / 1 }{ #1 }
1819   \__stex_refs_find_uri:n { #2 }
1820   \__stex_refs_set_keys_b:n{ #3 }
1821   \str_if_empty:NT \l_stex_key_file_str {
1822     \msg_error:nn{stex}{error/extrefmissing}
1823   }
1824   \__stex_refs_do_sref_in:n{#2}
1825 }

```

(End of definition for `\sref` and `\extref`. These functions are documented on page 87.)

```

\sref_new_sym_target:n
1826 \cs_new_protected:Nn \stex_ref_new_symbol:n {
1827   \cs_if_exist:cF{r@sref@sym@\tl_to_str:n{\#1}}{
1828     \__stex_refs_new_symbol:n{\#1}
1829   }
1830 }
1831
1832 \cs_new_protected:Nn \__stex_sref_do_aux:n {
1833   #1 \iow_now:Nn \auxout {\#1}
1834 }
1835
1836 \cs_new_protected:Nn \__stex_refs_new_symbol:n {
1837   \prop_if_exist:NTF \l_stex_current_archive_prop {

```

```

1838 \prop_get:NnNTF \l_stex_current_archive_prop {docurl} \l__stex_refs_str {
1839   \exp_args:Ne \_stex_sref_do_aux:n {
1840     \STEXInternalSetSrefSymURL{\#1}{\l__stex_refs_str / symbol? #1}
1841   }
1842 }{
1843   \_stex_sref_do_aux:n { \STEXInternalSetSrefSymURL{\#1}{} }
1844 }
1845 }{
1846   \_stex_sref_do_aux:n { \STEXInternalSetSrefSymURL{\#1}{} }
1847 }
1848 }
1849
1850 \cs_new_protected:Nn \stex_ref_new_sym_target:n {
1851   \cs_if_exist:NT \hypertarget{
1852     \exp_args:Ne \hypertarget{\tl_to_str:n{sref@sym@ #1}}{}
1853     \str_gset:cx{\tl_to_str:n{r@sref@sym@ #1}}{\tl_to_str:n{sref@sym@ #1}}
1854   }
1855 }
1856
1857 \cs_new_protected:Nn \stex_ref_new_sym_target:nn {
1858   \str_if_eq:nnTF{\#1}{\#2} {
1859     \stex_ref_new_sym_target:n{\#1}
1860   }{
1861     \str_gset:cn{g_stex_sref_sym_ #1 _label}{\#2}
1862   }
1863 }

```

(End of definition for `\stex_ref_new_sym_target:n`. This function is documented on page 124.)

### \srefsym

```

1864 \NewDocumentCommand \srefsym { m m }{
1865   \stex_get_symbol:n { #1 }
1866   \exp_args:Ne
1867   \__stex_refs_sym_aux:nn{\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
1868 }
1869
1870 \cs_new_protected:Npn \__stex_refs_do_internal_link:nn #1 {
1871   \cs_if_exist:NTF \hyperlink {
1872     \hyperlink{\#1}
1873   }\use:n
1874 }
1875
1876 \cs_new_protected:Npn \__stex_refs_do_url_link:nn {
1877   \cs_if_exist:NTF \href \href \use_i:nn
1878 }
1879
1880 \cs_new_protected:Npn \__stex_refs_sym_aux:nn #1 {
1881   \cs_if_exist:cTF{\tl_to_str:n{r@sref@sym@#1}}{
1882     \exp_args:Ne \__stex_refs_do_internal_link:nn{\tl_to_str:n{sref@sym@#1}}
1883   }{
1884     \str_if_exist:cTF{g_stex_sref_sym_#1_label}{
1885       \exp_args:Ne \__stex_refs_sym_aux:nn{\use:c{g_stex_sref_sym_#1_label}}
1886     }{
1887       \str_if_empty:cTF{g_stex_sref_sym_#1_target}{
```

```

1888     \exp_args:Nn \__stex_refs_do_internal_link:nn{\tl_to_str:n{sref@sym@#1}}
1889     }{
1890         \exp_args:Nn \__stex_refs_do_url_link:nn{\use:c{g_stex_sref_sym_#1_target}}
1891     }
1892 }
1893 }
1894 }

```

(End of definition for `\srefsym`. This function is documented on page 92.)

```

\srefsymuri
1895 \cs_new_protected:Npn \srefsymuri #1 {
1896     \__stex_refs_sym_aux:nn{#1}
1897 }

```

(End of definition for `\srefsymuri`. This function is documented on page 92.)

### 13.4.4 Inputs

```
1898 <@@=stex_inputs>
```

```

\stex_resolve_path_pair:Nnn
\stex_resolve_path_pair:Nxx
1899 \cs_new_protected:Nn \stex_resolve_path_pair:Nnn {
1900     \stex_debug:nn{resolving-path}{#3-in-[#2]}
1901     \str_if_empty:nTF{#2} {
1902         \prop_if_exist:NTF \l_stex_current_archive_prop {
1903             \str_set:Nx #1 {\stex_file_use:N \c_stex_mathhub_file /
1904             \prop_item:Nn \l_stex_current_archive_prop { id } / source /
1905             #3}
1906             \stex_debug:nn{resolving-path}{In-current-archive-
1907             \prop_item:Nn \l_stex_current_archive_prop { id }
1908             ;~result:-#1}
1909         }{
1910             \str_set:Nx #1 {\stex_file_use:N \c_stex_pwd_file / .. / #3 }
1911             \stex_debug:nn{resolving-path}{No-current-archive;~result:-#1}
1912         }
1913     }{
1914         \stex_require_archive:n { #2 }
1915         \str_set:Nx #1 {\stex_file_use:N \c_stex_mathhub_file /
1916         \prop_item:cn {c_stex_mathhub_#2_manifest_prop} { id } / source /
1917         #3}
1918         \stex_debug:nn{resolving-path}{result:-#1}
1919     }
1920 }
1921 \cs_generate_variant:Nn \stex_resolve_path_pair:Nnn {Nxx}

```

(End of definition for `\stex_resolve_path_pair:Nnn`. This function is documented on page 137.)

```

\inputref
\mhinput
\ifinputref
1922 \newif \ifinputref \inputreffalse
1923
1924 \cs_new_protected:Nn \__stex_inputs_mhinput:nn {
1925     \stex_in_archive:nn {#1} {
1926         \ifinputref
1927             \stex_input_with_hooks:n{ \stex_file_use:N \c_stex_mathhub_file / ##1 / source / #2 }

```

```

1928     \else
1929         \inputreftrue
1930             \stex_input_with_hooks:n{ \stex_file_use:N \c_stex_mathhub_file / ##1 / source / #2
1931             \inputreffalse
1932         \fi
1933     }
1934 }
1935
1936 \NewDocumentCommand \mhinput { O{} m }{
1937     \exp_args:NNx\exp_args:Nnx\__stex_inputs_mhinput:nn{ \tl_to_str:n{#1} }{ \tl_to_str:n{#2} }
1938 }
1939
1940 \cs_new_protected:Nn \__stex_inputs_inputref_html:nn {
1941     \str_clear:N \l_tmpa_str
1942     \prop_get:NnNF \l_stex_current_archive_prop { narr } \l_tmpa_str {
1943         \prop_get:NnNF \l_stex_current_archive_prop { ns } \l_tmpa_str {}
1944     }
1945     \tl_if_empty:nTF{ #1 }{
1946         \IfFileExists{#2}{%
1947             \relax
1948             \exp_args:Ne\stex_html_literal:n{
1949                 \detokenize{<span~data-shtml-inputref="#" }\l_tmpa_str/#2\detokenize{"~ data-shtml-visib
1950             }
1951         }{%
1952             \stex_input_with_hooks:n{#2}
1953         }
1954     }{%
1955         \IfFileExists{ \stex_file_use:N \c_stex_mathhub_file / #1 / source / #2 }{%
1956             \relax
1957             \exp_args:Ne\stex_html_literal:n{
1958                 \detokenize{<span~data-shtml-inputref="#" }\l_tmpa_str/#2\detokenize{"~data-shtml-visib
1959             }
1960         }{%
1961             \input{ \stex_file_use:N \c_stex_mathhub_file / #1 / source / #2 }
1962         }
1963     }
1964 }
1965
1966 \cs_new_protected:Nn \__stex_inputs_inputref_pdf:nn {
1967     \begingroup
1968         \inputreftrue
1969         \tl_if_empty:nTF{ #1 }{
1970             \stex_input_with_hooks:n{#2}
1971         }{%
1972             \stex_input_with_hooks:n{ \stex_file_use:N \c_stex_mathhub_file / #1 / source / #2 }
1973         }
1974     \endgroup
1975 }
1976
1977 \cs_new_protected:Nn \__stex_inputs_inputref:nn {
1978     \stex_in_archive:nn {#1} {
1979         \stex_if_html_backend:TF
1980             \__stex_inputs_inputref_html:nn
1981             \__stex_inputs_inputref_pdf:nn

```

```

1982     {##1}{#2}
1983   }
1984 }
1985
1986 \NewDocumentCommand \inputref { O{} m}{
1987   \exp_args:NNx \exp_args:Nnx \__stex_inputs_inputref:nn{ \tl_to_str:n{#1} }{ \tl_to_str:n{#2} }
1988 }

(End of definition for \inputref, \mhinput, and \ifinputref. These functions are documented on page 86.)

```

### \addmhbibresource

```

1989 \cs_new_protected:Nn \__stex_inputs_bibresource:n {
1990   \__stex_inputs_up_archive:nn{#1}{bib}
1991   \seq_if_empty:NTF \l__stex_inputs_libinput_files_seq {
1992     \msg_error:nnxx{stex}{error/nofile}{\exp_not:N\addmhbibresource}{#1.bib}
1993   }{
1994     \seq_map_inline:Nn \l__stex_inputs_libinput_files_seq {
1995       \addbibrssource{ ##1 }
1996     }
1997   }
1998 }
1999 \newcommand\addmhbibresource[2][]{%
2000   \tl_if_empty:nTF{#1}{%
2001     \__stex_inputs_bibresource:n{#2}
2002   }{
2003     \stex_in_archive:nn{#1}{\__stex_inputs_bibresource:n{#2}}
2004   }
2005 }

(End of definition for \addmhbibresource. This function is documented on page 83.)

```

### \IfInputref

```

2006 \stex_if_html_backend:TF{
2007   \newcommand \IfInputref[2]{
2008     \stex_annotation:nn{data-shtml-ifinputref=true}{#1}
2009     \stex_annotation:nn{data-shtml-ifinputref=false}{#2}
2010   }
2011 }{
2012   \newcommand \IfInputref[2]{
2013     \ifinputref #1 \else #2 \fi
2014   }
2015 }

(End of definition for \IfInputref. This function is documented on page 86.)

```

### \libinput

```

2016 \cs_new_protected:Nn \__stex_inputs_up_archive:nn {
2017   \prop_if_exist:NF \l_stex_current_archive_prop {
2018     \msg_error:nnn{stex}{error/notinarchive}\libinput
2019   }
2020   \prop_get:NnNF \l_stex_current_archive_prop {id} \l__stex_inputs_id_str {
2021     \msg_error:nnn{stex}{error/notinarchive}\libinput
2022   }
2023   \seq_clear:N \l__stex_inputs_libinput_files_seq

```

```

2024 \seq_set_eq:NN \l__stex_inputs_path_seq \c_stex_mathhub_file
2025 \seq_set_split:NnV \l__stex_inputs_id_seq / \l__stex_inputs_id_str
2026
2027 \bool_while_do:nn { ! \seq_if_empty_p:N \l__stex_inputs_id_seq }{
2028   \str_set:Nx \l__stex_inputs_path_str {\stex_file_use:N \l__stex_inputs_path_seq / meta-i
2029   \IfFileExists{ \l__stex_inputs_path_str }{
2030     \exp_args:NNo \seq_if_in:NnF \l__stex_inputs_libinput_files_seq \l__stex_inputs_path_s
2031       \seq_put_right:No \l__stex_inputs_libinput_files_seq \l__stex_inputs_path_str
2032     }
2033   }{}}
2034   \seq_pop_left:NN \l__stex_inputs_id_seq \l__stex_inputs_path_str
2035   \seq_put_right:No \l__stex_inputs_path_seq \l__stex_inputs_path_str
2036 }
2037
2038 \str_set:Nx \l__stex_inputs_path_str {\stex_file_use:N \l__stex_inputs_path_seq / lib / #1
2039 \IfFileExists{ \l__stex_inputs_path_str }{
2040   \exp_args:NNo \seq_if_in:NnF \l__stex_inputs_libinput_files_seq \l__stex_inputs_path_str
2041     \seq_put_right:No \l__stex_inputs_libinput_files_seq \l__stex_inputs_path_str
2042   }
2043 }{}}
2044 }
2045
2046 \cs_new_protected:Nn \__stex_inputs_libinput:n {
2047   \__stex_inputs_up_archive:nn{#1}{tex}
2048   \seq_if_empty:NTF \l__stex_inputs_libinput_files_seq {
2049     \msg_error:nnxx{stex}{error/nofile}{\exp_not:N\libinput}{#1.tex}
2050   }{
2051     \seq_map_inline:Nn \l__stex_inputs_libinput_files_seq {
2052       \input{ ##1 }
2053     }
2054   }
2055 }
2056
2057 \newcommand \libinput [2] [] {
2058   \tl_if_empty:nTF{#1} {
2059     \__stex_inputs_libinput:n{#2}
2060   }{
2061     \stex_in_archive:nn{#1}{\__stex_inputs_libinput:n{#2}}
2062   }
2063 }

```

(End of definition for \libinput. This function is documented on page 83.)

```
\libusepackage
2064 \newcommand\libusepackage[2] [] {
2065   \__stex_inputs_up_archive:nn{#2}{sty}
2066   \int_compare:nNnTF {\seq_count:N \l__stex_inputs_libinput_files_seq} = 1 {
2067     \str_set:Nx \l__stex_inputs_tmp_str {\seq_item:Nn \l__stex_inputs_libinput_files_seq 1}
2068     \exp_args:Nne \use:n {\usepackage[#1]} {
2069       \str_range:Nnn\l__stex_inputs_tmp_str 1 {-5}
2070     }
2071   }{
2072     \msg_fatal:nnnn{stex}{error/nofile}{\libusepackage}{#1.sty}
2073   }
2074 }
```

(End of definition for `\libusepackage`. This function is documented on page 83.)

```
\mhgraphics
\cmhgraphics
\lstinputmhlisting
\clstinputmhlisting
\mhtikzinput
\cmhtikzinput
2075 \str_new:N \l__stex_inputs_gin_repo_str
2076 \ltx@ifpackageloaded{graphicx}{\use:n}{\AtEndOfPackageFile{graphicx}}{
2077   \define@key{Gin}{archive}{
2078     \tl_set:Nx\Gin@mrepos{\stex_file_use:N \c_stex_mathhub_file / #1 / source /}
2079   }
2080   \providecommand\mhgraphics[2][]{
2081     \tl_set:Nx\Gin@mrepos{
2082       \stex_file_use:N \c_stex_mathhub_file / \prop_item:Nn \l_stex_current_archive_prop {id}
2083     }
2084     \setkeys{Gin}{#1}
2085     \includegraphics[#1]{ \Gin@mrepos #2 }
2086   }
2087   \providecommand\cmhgraphics[2][]{{\begin{center}}\mhgraphics[#1]{#2}{\end{center}}}
2088 }
2089
2090 \ltx@ifpackageloaded{listings}{\use:n}{\AtEndOfPackageFile{listings}}{
2091   \define@key{lst}{archive}{
2092     \def\lst@mrepos{\stex_file_use:N \c_stex_mathhub_file / #1 / source /}
2093   }
2094   \newcommand\lstinputmhlisting[2][]{%
2095     \def\lst@mrepos{
2096       \stex_file_use:N \c_stex_mathhub_file / \prop_item:Nn \l_stex_current_archive_prop {id}
2097     }
2098     \setkeys{lst}{#1}%
2099     \lstinputlisting[#1]{\lst@mrepos #2}
2100   \newcommand\clstinputmhlisting[2][]{\begin{center}\lstinputmhlisting[#1]{#2}{\end{center}}}
2101 }
2102
2103 \ltx@ifpackageloaded{tikzinput}{\use:n}{\AtEndOfPackageFile{tikzinput}}{
2104   \define@key{Gin}{archive}{
2105     \str_set:Nn \l__stex_inputs_gin_repo_str {#1}
2106     \def\Gin@mrepos{\stex_file_use:N \c_stex_mathhub_file / #1 / source /}
2107   }
2108   \newcommand\mhtikzinput[2][]{%
2109     \str_clear:N \l__stex_inputs_gin_repo_str
2110     \def\Gin@mrepos{
2111       \stex_file_use:N \c_stex_mathhub_file / \prop_item:Nn \l_stex_current_archive_prop {id}
2112     }
2113     \setkeys{Gin}{#1}%
2114     \exp_args:No \stex_in_archive:nn \l__stex_inputs_gin_repo_str {
2115       \tikzinput[#1]{\Gin@mrepos #2}
2116     }
2117   }
2118   \newcommand\cmhtikzinput[2][]{\begin{center}\mhtikzinput[#1]{#2}{\end{center}}}
2119 }
```

(End of definition for `\mhgraphics` and others. These functions are documented on page 86.)

```
\libusetikzlibrary
2120 \cs_new_protected:Nn \__stex_inputs_usetikzlibrary_i:nn {
2121   \pgfkeys@spdef\pgf@temp{#1}
```

```

2122 \expandafter\ifx\csname tikz@library@\pgf@temp @loaded\endcsname\relax%
2123 \expandafter\global\expandafter\let\csname tikz@library@\pgf@temp @loaded\endcsname=\pgfutil@in@
2124 \expandafter\edef\csname tikz@library@#1@atcode\endcsname{\the\catcode`@}
2125 \expandafter\edef\csname tikz@library@#1@barcode\endcsname{\the\catcode`\|}%
2126 \expandafter\edef\csname tikz@library@#1@dollarcode\endcsname{\the\catcode`\$}
2127 \catcode`\@=11
2128 \catcode`\|=12
2129 \catcode`\$=3
2130 \pgfutil@InputIfFileExists{#2}{}{%
2131 \catcode`\@=\csname tikz@library@#1@atcode\endcsname
2132 \catcode`\|= \csname tikz@library@#1@barcode\endcsname
2133 \catcode`\$=\csname tikz@library@#1@dollarcode\endcsname
2134 }
2135
2136 \cs_new_protected:Nn \__stex_inputs_usetikzlibrary:n{%
2137   \__stex_inputs_up_archive:nn{tikzlibrary#1}{code.tex}
2138   \int_compare:nNnTF {\seq_count:N \l__stex_inputs_lbininput_files_seq} = 1 {
2139     \exp_args:Nne \__stex_inputs_usetikzlibrary_i:nn{#1}{ \seq_item:Nn \l__stex_inputs_lbininput_files_seq {1} }{tikzlibrary#1.code.tex}
2140   }
2141   \msg_fatal:nnnn{stex}{error/nofile}{\libusetikzlibrary}{tikzlibrary#1.code.tex}
2142 }
2143 }
2144
2145 \newcommand \libusetikzlibrary [2][]{%
2146   \cs_if_exist:NF \usetikzlibrary {%
2147     \msg_error:nnx{stex}{error/notikz}{\tl_to_str:n{\libusetikzlibrary}}
2148   }
2149   \tl_if_empty:nTF{#1}{%
2150     \__stex_inputs_usetikzlibrary:n{#2}
2151   }{%
2152     \stex_in_archive:nn{#1}{\__stex_inputs_usetikzlibrary:n{#2}}
2153   }
2154 }

```

(End of definition for `\libusetikzlibrary`. This function is documented on page 121.)

## 13.5 SMS Mode

```

2155 <@@=stex_smsmode>
      Macros and environments allowed in sms mode:
2156 \tl_new:N \g__stex_smsmode_allowed_tl
2157 \tl_new:N \g__stex_smsmode_allowed_escape_tl
2158 \seq_new:N \g__stex_smsmode_allowedenvs_seq
      \stex_sms_allow:N
\stex_sms_allow_escape:N
\stex_sms_allow_env:n
2159 \cs_new_protected:Nn \stex_sms_allow:N {
2160   \tl_gput_right:Nn \g__stex_smsmode_allowed_tl {#1}
2161 }
2162
2163 \cs_new_protected:Nn \stex_sms_allow_escape:N {
2164   \tl_gput_right:Nn \g__stex_smsmode_allowed_escape_tl {#1}
2165 }
2166

```

```

2167 \cs_new_protected:Nn \stex_sms_allow_env:n {
2168   \exp_args:NNx \seq_gput_right:Nn \g__stex_smsmode_allowedenvs_seq {\tl_to_str:n{#1}}
2169 }

```

(End of definition for `\stex_sms_allow:N`, `\stex_sms_allow_escape:N`, and `\stex_sms_allow_env:n`. These functions are documented on page 138.)

Some initial allowed macros:

```

2170 \stex_sms_allow:N \makeatletter
2171 \stex_sms_allow:N \makeatother
2172 \stex_sms_allow:N \ExplSyntaxOn
2173 \stex_sms_allow:N \ExplSyntaxOff
2174 \stex_sms_allow:N \rustexBREAK

```

`\stex_if_smsmode_p:`  
`\stex_if_smsmode:TF`

```

2175 \bool_new:N \g__stex_smsmode_bool
2176 \bool_set_false:N \g__stex_smsmode_bool
2177 \prg_new_conditional:Nnn \stex_if_smsmode: { p, T, F, TF } {
2178   \bool_if:NTF \g__stex_smsmode_bool \prg_return_true: \prg_return_false:
2179 }

```

(End of definition for `\stex_if_smsmode:TF`. This function is documented on page 138.)

`\stex_sms_allow_import:Nn`  
`\stex_sms_allow_import_env:nn`

```

2180 \tl_new:N \g__stex_smsmode_allowed_import_tl
2181 \seq_new:N \g__stex_smsmode_allowed_import_env_seq
2182 \cs_new_protected:Nn \stex_sms_allow_import:Nn {
2183   \tl_gput_right:Nn \g__stex_smsmode_allowed_import_tl {#1}
2184   \tl_gset:cn{\tl_to_str:n{#1}~~~smsemode} {#2}
2185 }
2186 \cs_new_protected:Nn \stex_sms_allow_import_env:nn {
2187   \exp_args:NNx \seq_gput_right:Nn \g__stex_smsmode_allowed_import_env_seq {\tl_to_str:n{#1}
2188   \tl_gset:cn{\tl_to_str:n{#1}~~~env~~~smsemode} {#2}
2189 }
2190
2191 \tl_new:N \g_stex_sms_import_code

```

(End of definition for `\stex_sms_allow_import:Nn` and `\stex_sms_allow_import_env:nn`. These functions are documented on page 139.)

`\stex_file_in_smsmode:nn`  
`\stex_file_in_smsmode:on`

```

2192 \cs_new_protected:Nn \__stex_smsmode_in_smsmode:n { \stex_suppress_html:n {
2193   \vbox_set:Nn \l_tmpa_box {
2194     \bool_set_true:N \g__stex_smsmode_bool
2195     \bool_set_false:N \stex_html_do_output_bool
2196     #1
2197   }
2198   \%box_clear:N \l_tmpa_box
2199 } }
2200
2201 \quark_new:N \q__stex_smsmode_break
2202
2203 \cs_new_protected:Nn \__stex_smsmode_start_smsmode:n {
2204   \everyeof{\q__stex_smsmode_break\exp_not:N}
2205   \let\stex_smsmode_do:\__stex_smsmode_smsmode_do:
2206   \exp_after:wN \exp_after:wN \exp_after:wN

```

```

2207 \stex_smsmode_do:
2208   \cs:w @ @ input\cs_end: "#1" \relax
2209 }
2210
2211 \cs_new_protected:Nn \stex_file_in_smsmode:nn {
2212   \seq_gclear:N \l__stex_smsmode_importmodules_seq
2213   \seq_gclear:N \l__stex_smsmode_sigmodules_seq
2214   \tl_clear:N \g_stex_sms_import_code
2215   \group_begin:
2216     \let \l_stex_metatheory_uri \c_stex_default_metalanguage
2217     \cs_set:Npn \stex_check_term:n ##1 {}
2218     \seq_clear:N \l_stex_all_modules_seq
2219     \str_clear:N \l_stex_current_module_str
2220     #2
2221     \stex_filestack_push:n{#1}
2222     \__stex_smsmode_in_smsmode:n {
2223       \let \__stex_smsmode_do_aux_curr:N \__stex_smsmode_do_aux_imports:N
2224       \tl_map_inline:Nn \g_stex_smsmode_allowed_import_tl {
2225         \use:c{\tl_to_str:n{##1}---smsmode}
2226       }
2227       \seq_map_inline:Nn \g_stex_smsmode_allowed_import_env_seq {
2228         \use:c{\tl_to_str:n{##1}---env---smsmode}
2229       }
2230       \__stex_smsmode_start_smsmode:n{#1}
2231     }
2232     \__stex_smsmode_in_smsmode:n \g_stex_sms_import_code
2233     \__stex_smsmode_in_smsmode:n {
2234       \let \__stex_smsmode_do_aux_curr:N \__stex_smsmode_do_aux_normal:N
2235       \__stex_smsmode_start_smsmode:n{#1}
2236     }
2237     \stex_filestack_pop:
2238   \group_end:
2239 }
2240 \cs_generate_variant:Nn \stex_file_in_smsmode:nn {on}

```

(End of definition for `\stex_file_in_smsmode:nn`. This function is documented on page 138.)

```

\stex_smsmode_do:
2241 \cs_new_protected:Nn \__stex_smsmode_smsmode_do: {
2242   \%stex_if_smsmode:T {
2243     \__stex_smsmode_do:w
2244   %}
2245 }
2246 \let\stex_smsmode_do:\relax
2247
2248 \cs_new:Nn \__stex_smsmode_check_cs:NNn {
2249   \exp_after:wN\if\exp_after:wN\relax\exp_not:N#3
2250   \exp_after:wN#1\exp_after:wN#3\else
2251   \exp_after:wN#2\fi
2252 }
2253
2254 \cs_new_protected:Npn \__stex_smsmode_do:w #1 {
2255   \exp_args:Nx \tl_if_empty:nTF { \tl_tail:n{ #1 }}{

```

```

2257     \__stex_smsemode_check_cs:NNn \__stex_smsemode_do_aux:N \__stex_smsemode_do:w { #1 }
2258     }{
2259         \__stex_smsemode_do:w
2260     }
2261 }
2262
2263 \cs_new_protected:Nn \__stex_smsemode_do_aux:N {
2264     \cs_if_eq:NNF #1 \q__stex_smsemode_break {
2265         \__stex_smsemode_do_aux_curr:N #1
2266     }
2267 }
2268
2269 \cs_new_protected:Nn \__stex_smsemode_do_aux_imports:N {
2270     % \stex_debug:nn{sms}{Checking~\tl_to_str:n{#1}-in~import}
2271     \tl_if_in:NnTF \g__stex_smsemode_allowed_import_tl {#1} {
2272         \stex_debug:nn{sms}{Executing~\tl_to_str:n{#1}-in~import}
2273         #1
2274     }{
2275         \cs_if_eq:NNTF \begin {#1} {
2276             \__stex_smsemode_check_begin:Nn \g__stex_smsemode_allowed_import_env_seq
2277         }{
2278             \cs_if_eq:NNTF \end {#1} {
2279                 \__stex_smsemode_check_end:Nn \g__stex_smsemode_allowed_import_env_seq
2280             }{
2281                 \__stex_smsemode_do:w
2282             }
2283         }
2284     }
2285 }
2286
2287 \cs_new_protected:Nn \__stex_smsemode_do_aux_normal:N {
2288     % \stex_debug:nn{sms}{Checking~\tl_to_str:n{#1}-in-sms-mode}
2289     \tl_if_in:NnTF \g__stex_smsemode_allowed_tl {#1} {
2290         \stex_debug:nn{sms}{Executing~\tl_to_str:n{#1}}
2291         #1\__stex_smsemode_do:w
2292     }{
2293         \tl_if_in:NnTF \g__stex_smsemode_allowed_escape_tl {#1} {
2294             \stex_debug:nn{sms}{Executing~escaped~\tl_to_str:n{#1}}
2295             #1
2296         }{
2297             \cs_if_eq:NNTF \begin {#1} {
2298                 \__stex_smsemode_check_begin:Nn \g__stex_smsemode_allowedenvs_seq
2299             }{
2300                 \cs_if_eq:NNTF \end {#1} {
2301                     \__stex_smsemode_check_end:Nn \g__stex_smsemode_allowedenvs_seq
2302                 }{
2303                     \__stex_smsemode_do:w
2304                 }
2305             }
2306         }
2307     }
2308 }
2309
2310 \cs_new_protected:Nn \__stex_smsemode_check_begin:Nn {

```

```

2311 % \stex_debug:nn{sms}{Checking~environment~#2}
2312   \seq_if_in:NxTF #1 { \tl_to_str:n{#2} }{
2313     \stex_debug:nn{sms}{Environment~#2}
2314     \begin{#2}
2315   }{
2316     \__stex_smsmode_do:w
2317   }
2318 }
2319 \cs_new_protected:Nn \__stex_smsmode_check_end:Nn {
2320 % \stex_debug:nn{sms}{Checking~end~environment~#2}
2321   \seq_if_in:NxTF #1 { \tl_to_str:n{#2} }{
2322     \stex_debug:nn{sms}{End-Environment~#2}
2323     \end{#2}\__stex_smsmode_do:w
2324   }{
2325     \%str_if_eq:nnTF{#2}{document} \endinput
2326     \__stex_smsmode_do:w
2327   }
2328 }

```

(End of definition for `\stex_smsmode_do:`. This function is documented on page 139.)

## 13.6 Modules

### 13.6.1 The smodule-environment

`2329 <@@=stex_modules>`

`\l_stex_current_module_str` The current module:

`2330 \str_new:N \l_stex_current_module_str`

(End of definition for `\l_stex_current_module_str`. This variable is documented on page 125.)

`\l_stex_all_modules_seq` Stores all modules currently in scope

`2331 \seq_new:N \l_stex_all_modules_seq`

(End of definition for `\l_stex_all_modules_seq`. This variable is documented on page 125.)

`\stex_every_module:n`

```

2332 <@@=stex_module_setup>
2333 \tl_clear:N \g_stex_every_module_tl {
2334 }
2335 \cs_new_protected:Nn \stex_every_module:n {
2336   \tl_gput_right:Nn \g_stex_every_module_tl { #1 }
2337 }

```

(End of definition for `\stex_every_module:n`. This function is documented on page 125.)

`\stex_module_setup:n` Sets up a new module:

```

2338 \cs_new_protected:Npn \stex_module_setup:n {
2339   \stex_if_in_module:TF \__stex_module_setup_setup_nested:n \__stex_module_setup_setup_top:n
2340 }
2341 \cs_new_protected:Nn \__stex_module_setup_setup_top:n {
2342   \__stex_module_setup_get_uri_str:n{#1}
2343   \stex_debug:nn{module}{Module~URI:~\__stex_module_setup_ns_str?#1}
2344 }

```

```

2345 \str_if_empty:NTF \l_stex_key_sig_str
2346 \_stex_module_setup_top_nosig:n \_stex_module_setup_top_sig:n {\l__stex_module_setu
2347 \stex_metagroup_new:o \l_stex_current_module_str
2348 \g_stex_every_module_tl
2349 \stex_execute_in_module:x {
2350     \_stex_do_deprecation:n{#1}
2351 }
2352 \_\_stex_module_setup_load_meta:
2353 }
2354
2355 \cs_new_protected:Nn \_stex_module_setup_top_nosig:n {
2356     \stex_if_module_exists:nTF{#1} {
2357         \stex_debug:nn{modules}{(already exists)}
2358     }{
2359         \tl_gclear:c{c_stex_module_ #1 _code}
2360         \prop_gclear:c{c_stex_module_ #1 _morphisms_prop }
2361         \prop_gclear:c{c_stex_module_ #1 _symbols_prop }
2362         \prop_gclear:c{c_stex_module_ #1 _notations_prop }
2363     }
2364     \str_set:Nx \l_stex_current_module_str {#1}
2365     \seq_put_right:No \l_stex_all_modules_seq \l_stex_current_module_str
2366 }
2367
2368 \cs_new_protected:Nn \_\_stex_module_setup_setup_top_sig:n {
2369     \stex_if_module_exists:nTF{#1} {
2370         \stex_debug:nn{modules}{(already exists)}
2371     }{
2372         \stex_debug:nn{modules}{(needs loading)}
2373         \_\_stex_module_setup_load_sig:
2374     }
2375 \% \stex_if_smsmode:F { % WHY?
2376     \stex_activate_module:x {
2377         #1
2378     }
2379     %}
2380     \str_set:Nx \l_stex_current_module_str{#1}
2381 }
2382
2383 \cs_new_protected:Nn \_\_stex_module_setup_load_sig: {
2384     \stex_file_split_off_ext:NN \l__stex_module_setup_sigfile \g_stex_current_file
2385     \stex_file_split_off_lang:NN \l__stex_module_setup_sigfile \l__stex_module_setup_sigfile
2386     \exp_args:Ne \stex_file_in_smsmode:nn {
2387         \stex_file_use:N \l__stex_module_setup_sigfile . \l_stex_key_sig_str . tex
2388     }{}
2389 }
2390
2391 \cs_new_protected:Nn \_\_stex_module_setup_setup_nested:n {
2392     \exp_after:wN
2393         \_\_stex_module_setup_split_module:n \l_stex_current_module_str \_\_stex_module_setup_end:
2394     \stex_debug:nn{module}{Nested~Module~URI:~\l_stex_current_module_str}
2395     \seq_put_right:No \l_stex_all_modules_seq \l_stex_current_module_str
2396     \stex_metagroup_new:o \l_stex_current_module_str
2397 }
2398

```

```

2399 \cs_new_protected:Nn \__stex_module_setup_get_uri_str:n {
2400   \str_clear:N \l__stex_module_setup_ns_str
2401   \stex_map_uri:Nnnnn \l_stex_current_ns_uri {
2402     \str_set:Nx \l__stex_module_setup_ns_str{##1\cColonStr/}
2403   }{
2404     \seq_set_split:Nnn \l__stex_module_setup_seq / {##1}
2405     \seq_pop_right:NN \l__stex_module_setup_seq \l__stex_module_setup_seg
2406     \exp_args:No \str_if_eq:nnF \l__stex_module_setup_seg {#1} {
2407       \seq_put_right:No \l__stex_module_setup_seq \l__stex_module_setup_seg
2408     }
2409     \tl_put_right:Nx \l__stex_module_setup_ns_str {\seq_use:Nn \l__stex_module_setup_seq /}
2410   }{}{}
2411 }
2412 }
2413
2414 \cs_new_protected:Npn \__stex_module_setup_split_module:n #1?#2 \__stex_module_setup_end: #3
2415   \stex_module_setup_top_nosig:n { #1 ? #2 / #3}
2416 }
2417
2418 \bool_new:N \l_stex_in_meta_bool
2419 \bool_set_false:N \l_stex_in_meta_bool
2420
2421 \cs_new_protected:Nn \__stex_module_setup_load_meta: {
2422   \tl_if_empty:NF \l_stex_metatheory_uri {
2423     \stex_execute_in_module:x{
2424       \stex_pseudogroup_with:nn{\l_stex_in_meta_bool}{%
2425         \stex_activate_module:n {\stex_uri_use:N \l_stex_metatheory_uri }%
2426       }
2427     }
2428   }
2429 }
2430
2431 <@@=stex_modules>

```

(End of definition for `\stex_module_setup:n`. This function is documented on page 125.)

`\stex_close_module:`

```

2432 \cs_new:Nn \stex_close_module: {
2433   \bool_if:NT \c_stex_persist_write_mode_bool \__stex_modules_persist_module:
2434   \stex_debug:nn{module}{%
2435     Closing~module~\l_stex_current_module_str^~J
2436     Code:~\expandafter\meaning\csname c_stex_module_\l_stex_current_module_str _code\endcsna
2437     Imports:\exp_after:wN \prop_to_keyval:N \cs:w c_stex_module_\l_stex_current_module_str
2438     Declarations:\exp_after:wN \prop_to_keyval:N \cs:w c_stex_module_\l_stex_current_module_
2439     Notations:\exp_after:wN \prop_to_keyval:N \cs:w c_stex_module_\l_stex_current_module_str
2440   }
2441 }
2442
2443 \cs_new_protected:Nn \__stex_modules_persist_module: {
2444   \stex_persist:e {
2445     \__stex_modules_restore_module:nnnn {\l_stex_current_module_str}{%
2446       \exp_after:wN \prop_to_keyval:N \cs:w
2447         c_stex_module_\l_stex_current_module_str _morphisms_prop
2448       \cs_end:
2449     }
2450   }
2451 }

```

```

2449 }{
2450   \exp_after:wN \prop_to_keyval:N \cs:w
2451     c_stex_module_\l_stex_current_module_str _symbols_prop
2452   \cs_end:
2453 }{
2454   \exp_after:wN \exp_after:wN \exp_after:wN \exp_not:n
2455   \exp_after:wN \exp_after:wN \exp_after:wN
2456   { \cs:w c_stex_module_\l_stex_current_module_str _code \cs_end: }
2457 }{}
2458 \prop_map_function:cN{c_stex_module_\l_stex_current_module_str _notations_prop}
2459   \_stex_modules_persist_nots_i:nn
2460   \exp_not:N \STEXRestoreNotsEnd {}
2461 }
2462 }
2463
2464 \cs_new_protected:Nn \_stex_modules_restore_module:nnnn {
2465   \prop_gset_from_keyval:cn{c_stex_module_\tl_to_str:n{#1}_morphisms_prop}{#2}
2466   \cs_set:Npn \_stex_modules_tl {#3}
2467   \exp_args:Nno \prop_gset_from_keyval:cn{c_stex_module_\tl_to_str:n{#1}_symbols_prop}\_stex_
2468   \prop_map_inline:cn{c_stex_module_\tl_to_str:n{#1}_symbols_prop}%
2469   \stex_ref_new_symbol:n{#1?##1}
2470 }
2471 \cs_gset:cpn{c_stex_module_\tl_to_str:n{#1}_code}{#4}
2472 \prop_gclear:c{c_stex_module_\tl_to_str:n{#1} _notations_prop}
2473 \str_set:Nn \l__stex_modules_restore_mod_str {#1}
2474 \group_begin:
2475   \catcode`_=8\relax
2476   \catcode`:=12\relax
2477   \_stex_modules_restore_nots:n
2478 }
2479
2480 \cs_new:Nn \_stex_modules_persist_nots_i:nn {
2481   \exp_not:n{#2}
2482 }
2483
2484 \quark_new:N \STEXRestoreNotsEnd
2485
2486 \cs_new_protected:Nn \_stex_modules_restore_nots:n {
2487   \_stex_modules_restore_nots_i:n
2488 }
2489
2490 \cs_new_protected:Nn \_stex_modules_restore_nots_i:n {
2491   \tl_if_eq:nnTF{#1}{\STEXRestoreNotsEnd}%
2492   \group_end:
2493 }{
2494   \_stex_modules_restore_nots_ii:nnnn {#1}
2495 }
2496 }
2497
2498 \cs_new_protected:Nn \_stex_modules_restore_nots_ii:nnnnn {
2499   \cs_set:Npn \l__stex_modules_tl {{#4}{#5}}
2500   \exp_args:NNe\use:nn\prop_gput:cnn{
2501     {c_stex_module_\l__stex_modules_restore_mod_str _notations_prop}
2502     {\tl_to_str:n{#1!#2}}%

```

```

2503     {\tl_to_str:n{#1}}{\tl_to_str:n{#2}}{#3}
2504     \exp_args:No \exp_not:n \l__stex_modules_tl
2505   }
2506 }
2507 \__stex_modules_restore_nots_i:n
2508 }

```

(End of definition for `\stex_close_module`. This function is documented on page 125.)

`\l_stex_metatheory_uri`

```
2509 \tl_new:N \l_stex_metatheory_uri
```

(End of definition for `\l_stex_metatheory_uri`. This variable is documented on page ??.)

`\setmetatheory`

```

2510 \cs_new_protected:Nn \__stex_modules_set_matatheory:nn {
2511   \group_begin:
2512     \stex_debug:nn{metatheory}{Setting-metatheory~[#1]#2}
2513     \stex_import_module_uri:nn { #1 } { #2 }
2514     \stex_debug:nn{metatheory}{Here:^^J
2515       \l_stex_import_archive_str^^J
2516       \l_stex_import_path_str^^J
2517       \l_stex_import_name_str^^J
2518   }
2519   \stex_import_require_module:ooo
2520     \l_stex_import_archive_str
2521     \l_stex_import_path_str
2522     \l_stex_import_name_str
2523   \stex_debug:nn{metatheory}{Found:-\l_stex_import_ns_str}
2524   \exp_args:Nne \use:nn {
2525     \group_end: \stex_uri_resolve:Nn \l_stex_metatheory_uri
2526     }{\l_stex_import_ns_str}
2527 }
2528
2529 \NewDocumentCommand \setmetatheory {O{} m}{
2530   \__stex_modules_set_matatheory:nn { #1 }{ #2 }
2531   \stex_smsmode_do:
2532 }
2533 \stex_sms_allow_escape:N \setmetatheory

```

(End of definition for `\setmetatheory`. This function is documented on page ??.)

Keys and key handling:

```

2534 \stex_keys_define:nnnn{smodule}{
2535   \str_clear:N \l_stex_key_sig_str
2536 }{
2537   meta .code:n = {
2538     \str_if_empty:nTF {#1} {
2539       \tl_clear:N \l_stex_metatheory_uri
2540     }{
2541       \stex_uri_resolve:Nx \l_stex_metatheory_uri { #1 }
2542     }
2543   },
2544   ns .code:n = {
2545     \stex_uri_resolve:Nx \l_stex_current_ns_uri { #1 }

```

```

2546   } ,
2547   lang .code:n      = {
2548     \stex_set_language:n { #1 }
2549   } ,
2550   sig .str_set_x:N  = \l_stex_key_sig_str ,
2551   creators .code:n  = {} , % todo ?
2552   contributors .code:n = {} , % todo ?
2553   srccite .code:n    = {} % todo ?
2554 }{id, title, style, deprecate}

smodule (env.)
2555 \stex_new_stylable_env:nnnnnnn {module} {O{} m}{
2556   \stex_keys_set:nn { smodule }{ #1 }
2557   \tl_set_eq:NN \thistitle \l_stex_key_title_tl
2558   \tl_if_empty:NF \thistitle {
2559     \exp_args:No \stexdoctitle \thistitle
2560   }
2561   \exp_args:Nx \stex_module_setup:n { \tl_to_str:n{ #2 } }
2562
2563   \stex_if_do_html:T {
2564     \exp_args:Nne \begin{stex_annotation_env} {
2565       data-shtml-theory={\l_stex_current_module_str},
2566       data-shtml-language={ \l_stex_current_language_str},
2567       data-shtml-signature={\l_stex_key_sig_str}
2568       \tl_if_empty:NF \l_stex_metatheory_uri {
2569         data-shtml-metatheory={\stex_uri_use:N \l_stex_metatheory_uri}
2570       }
2571     }
2572     \stex_annotation_invisible:n
2573   }
2574   \stex_if_smsmode:F {
2575     \str_set_eq:NN \thismoduleuri \l_stex_current_module_str
2576     \tl_set:Nn \thismodulename {#2}
2577     \stex_style_apply:
2578   }
2579   \stex_smsmode_do:
2580 }{
2581   \stex_close_module:
2582   \stex_if_smsmode:F \stex_style_apply:
2583   \stex_if_do_html:T{ \end{stex_annotation_env} }
2584 }{}{}{s}
2585
2586 \stex_sms_allow_env:n{smodule}

```

\stex\_if\_in\_module\_p: Are we currently in a module?

\stex\_if\_in\_module:TF 2587 \prg\_new\_conditional:Nnn \stex\_if\_in\_module: {p, T, F, TF} {
2588 \str\_if\_empty:NTF \l\_stex\_current\_module\_str
2589 \prg\_return\_false: \prg\_return\_true:
2590 }

(End of definition for \stex\_if\_in\_module:TF. This function is documented on page 125.)

\stex\_if\_module\_exists\_p:n Does a module with this URI exist?

\stex\_if\_module\_exists:nTF 2591 \prg\_new\_conditional:Nnn \stex\_if\_module\_exists:n {p, T, F, TF} {

```

2592   \tl_if_exist:cTF { c_stex_module_#1_code }
2593     \prg_return_true: \prg_return_false:
2594 }

```

(End of definition for `\stex_if_module_exists:nTF`. This function is documented on page 125.)

`\stex_do_up_to_module:n` Execute code in the current module (i.e. as if the `\begin{smodule}` was the current tex group)

```

2595 \cs_new_protected:Nn \stex_do_up_to_module:n {
2596   \exp_args:No \stex_metagroup_do_in:nn \l_stex_current_module_str {#1}
2597 }
2598 \cs_generate_variant:Nn \stex_do_up_to_module:n {x}

```

(End of definition for `\stex_do_up_to_module:n`. This function is documented on page 126.)

`\stex_module_add_code:n`

`\stex_module_add_code:x`

```

2599 \cs_new_protected:Nn \stex_module_add_code:n {
2600   \tl_gput_right:cn {c_stex_module_\l_stex_current_module_str _code} { #1 }
2601 }
2602 \cs_generate_variant:Nn \stex_module_add_code:n {x}

```

(End of definition for `\stex_module_add_code:n`. This function is documented on page 126.)

`\stex_execute_in_module:n`

`\stex_execute_in_module:x`

```

2603 \cs_new_protected:Nn \stex_execute_in_module:n { \stex_if_in_module:TF {
2604   \stex_module_add_code:n { #1 }
2605   \stex_do_up_to_module:n { #1 }
2606 }{ #1 } }
2607 \cs_generate_variant:Nn \stex_execute_in_module:n {x}

```

(End of definition for `\stex_execute_in_module:n`. This function is documented on page 125.)

`\STEXexport`

```

2608 \NewDocumentCommand \STEXexport {} {
2609   \ExplSyntaxOn
2610   \__stex_modules_export:n
2611 }
2612 \cs_new_protected:Nn \__stex_modules_export:n {
2613   \stex_ignore_spaces_and_pars:#1\ExplSyntaxOff
2614   \stex_module_add_code:n { \stex_ignore_spaces_and_pars:#1}
2615   \stex_smsmode_do:
2616 }

```

Only allowed in modules, and allowed (escaped) in sms mode:

```

2617 \stex_deactivate_macro:Nn \STEXexport {module-environments}
2618 \stex_sms_allow_escape:N \STEXexport
2619 \stex_every_module:n {\stex_reactivate_macro:N \STEXexport}

```

(End of definition for `\STEXexport`. This function is documented on page 89.)

`\stex_module_add_morphism:nnnn`

`\stex_module_add_morphism:nonn`

`\stex_module_add_morphism:ooox`

```

2620 \cs_new_protected:Nn \stex_module_add_morphism:nnnn {
2621   \exp_args:Nne \prop_gput:cnn{c_stex_module_\l_stex_current_module_str _morphisms_prop}{%
2622     \tl_if_empty:nTF{#1}{[#2]}{#1}
2623   }{#1}{#2}{#3}{#4}}
2624 }
2625 \cs_generate_variant:Nn \stex_module_add_morphism:nnnn {nonn,ooox}

```

(End of definition for \stex\_module\_add\_morphism:nnnn. This function is documented on page 133.)

```
\stex_module_add_symbol:nnnnnnN #1 : {⟨Macro name⟩}
#2 : {⟨Name⟩}
#3 : {⟨arity⟩}
#4 : {(⟨Arg num⟩){⟨Arg str⟩})*}
#5 : Definiens
#6 : type
#7 : Return
#8 : Command

2626 \cs_new_protected:Nn \stex_module_add_symbol:nnnnnnnnN {
2627   \stex_debug:nn{declaration}{New~declaration:~\l_stex_current_module_str?#2^J
2628     Macro:#1^JArity:#3~(#4)^J
2629     Def:~\tl_to_str:n{#5}^J
2630     Type:~\tl_to_str:n{#6}^J
2631     Returns:~\tl_to_str:n{#7}
2632   }
2633   \%prop_gput:cnx{c_stex_module_}\l_stex_current_module_str _symbols_prop}
2634   %{\#2}{\exp_not:n{#1}{#2}{#3}{#4}{#5}{#6}}{\#7}\exp_not:n{#8}}
2635   \prop_gput:cnn{c_stex_module_}\l_stex_current_module_str _symbols_prop}
2636   {\#2}{\#1}{\#2}{\#3}{\#4}{\#5}{\#6}{\#7}{\#8}}
2637   \tl_if_empty:nF{#1}{
2638     \stex_execute_in_module:n {
2639       \__stex_modules_activate_sym:n {#2}
2640     }
2641   }
2642 }
2643
2644 \cs_new_protected:Nn \__stex_modules_activate_sym:n {
2645   \prop_map_inline:cn{c_stex_module_}\l_stex_current_module_str _symbols_prop} {
2646     \str_if_eq:nnT{#1}{##1} {
2647       \__stex_modules_activate_i:nnnnnnnn ##2
2648     }
2649   }
2650 }
2651 \cs_new_protected:Nn \__stex_modules_activate_i:nnnnnnnn {
2652   \stex_debug:nn{activating}{#1:\l_stex_current_module_str^J
2653     \tl_to_str:n{#2}{#3}{#4}{#5}{#6}{#7}{#8}}
2654   }
2655   \cs_set:cpx{#1} {
2656     \stex_invoke_symbol:nnnnnnN
2657     {\l_stex_current_module_str}
2658     \exp_not:n{#2}{#3}{#4}{#5}{#6}{#7}{#8}
2659   }
2660   \stex_debug:nn{activating}{done}
2661   \prop_map_break:
2662 }
```

(End of definition for \stex\_module\_add\_symbol:nnnnnnN. This function is documented on page ??.)

```
\stex_module_add_notation:nnnn #1 : URI
\stex_module_add_notation:eoeeoo #2 : variant
\stex_set_notation_macro:nnnnn #3 : arity
```

```

#4 : macro body
#5 : op

2663 \cs_new_protected:Nn \stex_module_add_notation:nnnnn {
2664   \stex_debug:nn{notations}{Adding~notation:^^J
2665     #1~\c_hash_str#2~#3^^J
2666     to~\l_stex_current_module_str
2667   }
2668   \prop_gput:cnn{c_stex_module_\l_stex_current_module_str _notations_prop}
2669   {#1!#2}{##1}{#2}{#3}{#4}{#5}
2670   \stex_execute_in_module:n {
2671     \__stex_modules_activate_not:nn{#1}{#2}
2672   }
2673 }
2674 \cs_generate_variant:Nn \stex_module_add_notation:nnnnn {eoeoo}
2675
2676
2677 \cs_new_protected:Nn \__stex_modules_activate_not:nn {
2678   \prop_map_inline:cn{c_stex_module_\l_stex_current_module_str _notations_prop} {
2679     \str_if_eq:nnT{#1!#2}{##1} {
2680       \prop_map_break:n{\stex_set_notation_macro:nnnnn ##2 }
2681     }
2682   }
2683 }

```

(End of definition for `\stex_module_add_notation:nnnnn` and `\stex_set_notation_macro:nnnnn`. These functions are documented on page 128.)

```

\stex_set_notation_macro:nnnnn
\stex_set_notation_macro:eoexo
2684 \cs_new_protected:Nn \stex_set_notation_macro:nnnnn {
2685   \tl_set:cn {l_stex_notation_#1_#2_cs}{#4}
2686   \cs_if_exist:cF{l_stex_notation_#1__cs} {
2687     \tl_set:cn {l_stex_notation_#1__cs}{#4}
2688   }
2689   \tl_if_empty:nF{#5} {
2690     \tl_set:cn{l_stex_notation_#1_op_#2_cs}{#5}
2691     \cs_if_exist:cF{l_stex_notation_#1_op__cs} {
2692       \cs_set_eq:cc {l_stex_notation_#1_op__cs}{l_stex_notation_#1_op_#2_cs}
2693     }
2694   }
2695 }
2696 \cs_generate_variant:Nn \stex_set_notation_macro:nnnnn {eoexo}

```

(End of definition for `\stex_set_notation_macro:nnnnn`. This function is documented on page 129.)

```

\stex_activate_module:n
\stex_activate_module:o
\stex_activate_module:x
2697 \cs_new_protected:Nn \stex_activate_module:n {
2698   \seq_if_in:NnF \l_stex_all_modules_seq { #1 } {
2699     \stex_debug:nn{modules}{Activating~module~#1^^J\expandafter\meaning\csname c_stex_module
2700     \seq_put_right:Nn \l_stex_all_modules_seq { #1 }
2701     \stex_pseudogroup:nn{
2702       \str_set:Nn \l_stex_current_module_str {#1}
2703       \use:c{ c_stex_module_#1_code }
2704     }{
2705       \stex_pseudogroup_restore:N \l_stex_current_module_str

```

```

2706     }
2707   }
2708 }
2709 \cs_generate_variant:Nn \stex_activate_module:n {o,x}

```

(End of definition for \stex\_activate\_module:n. This function is documented on page 125.)

Iterating:

```
2710 (@@=stex_iterate)
```

\stex\_iterate\_symbols:n

```

2711 \cs_new_protected:Nn \stex_iterate_symbols:n {
2712   \stex_pseudogroup_with:nn{\__stex_iterate_sym_cs:nnnnnnnnN\stex_iterate_break:\stex_iterat
2713     \cs_set:Npn \__stex_iterate_sym_cs:nnnnnnnnN
2714     ##1 ##2 ##3 ##4 ##5 ##6 ##7 ##8 ##9 { #1 }
2715     \cs_set:Npn \stex_iterate_break: {
2716       \prop_map_break:n{\seq_map_break:}
2717     }
2718     \cs_set:Npn \stex_iterate_break:n ##1 {
2719       \prop_map_break:n{\seq_map_break:n{##1}}
2720     }
2721     \seq_map_inline:Nn \l_stex_all_modules_seq {
2722       \prop_map_inline:cn{c_stex_module_#1_symbols_prop}{
2723         \__stex_iterate_sym_cs:nnnnnnnnN {##1} #####2
2724       }
2725     }
2726   }
2727 }
```

(End of definition for \stex\_iterate\_symbols:n. This function is documented on page 126.)

\stex\_iterate\_symbols:nn

```

2728 \cs_new_protected:Nn \stex_iterate_symbols:nn {
2729   \seq_clear:N \l__stex_iterate_mods_seq
2730   \stex_pseudogroup_with:nn{\__stex_iterate_sym_cs:nnnnnnnnN}{%
2731     \cs_set:Npn \__stex_iterate_sym_cs:nnnnnnnnN
2732     ##1 ##2 ##3 ##4 ##5 ##6 ##7 ##8 ##9 { #2 }
2733     \clist_map_function:nN {#1} \__stex_iterate_it_decl_i:n
2734   }
2735 }
2736
2737 \cs_new_protected:Nn \__stex_iterate_it_decl_i:n {
2738   \seq_if_in:NnF \l__stex_iterate_mods_seq {#1} {
2739     \seq_put_left:Nn \l__stex_iterate_mods_seq {#1}
2740     \prop_map_inline:cn{c_stex_module_#1_morphisms_prop}{
2741       \__stex_iterate_it_decl_check:nnnn ##2
2742     }
2743     \prop_map_inline:cn{c_stex_module_#1_symbols_prop}{
2744       \__stex_iterate_sym_cs:nnnnnnnnN {#1} ##2
2745     }
2746   }
2747 }
2748 \cs_new_protected:Nn \__stex_iterate_it_decl_check:nnnn {
2749   \tl_if_empty:nT{#1}{%
2750     \__stex_iterate_it_decl_i:n {#2}

```

```

2751     }
2752 }

(End of definition for \stex_iterate_symbols:nn. This function is documented on page 126.)
```

```
\stex_iterate_notations:nn
2753 \cs_new_protected:Nn \stex_iterate_notations:nn {
2754   \seq_clear:N \l__stex_iterate_mods_seq
2755   \stex_pseudogroup_with:nn{\l__stex_iterate_not_cs:nnnnn}{%
2756     \cs_set:Npn \__stex_iterate_not_cs:nnnnn{%
2757       ##1 ##2 ##3 ##4 ##5 { #2 }%
2758       \clist_map_function:nN {#1} \__stex_iterate_it_not_i:n
2759     }%
2760   }%
2761
2762 \cs_new_protected:Nn \__stex_iterate_it_not_i:n {%
2763   \seq_if_in:NnF \l__stex_iterate_mods_seq {#1} {%
2764     \seq_put_left:Nn \l__stex_iterate_mods_seq {#1}%
2765     \prop_map_inline:cn{c_stex_module_#1_notations_prop}{%
2766       \__stex_iterate_not_cs:nnnnn ##2
2767     }%
2768     \prop_map_inline:cn{c_stex_module_#1_morphisms_prop}{%
2769       \__stex_iterate_it_not_check:nnnn ##2
2770     }%
2771   }%
2772 }%
2773 \cs_new_protected:Nn \__stex_iterate_it_not_check:nnnn {%
2774   \tl_if_empty:nT{#1}{%
2775     \__stex_iterate_it_not_i:n {#2}
2776   }%
2777 }
```

(End of definition for \stex\_iterate\_notations:nn. This function is documented on page 129.)

```
\stex_iterate_morphisms:nn
2778 \cs_new_protected:Nn \stex_iterate_morphisms:nn {
2779   \seq_clear:N \l__stex_iterate_mods_seq
2780   \bool_set_true:N \l__stex_iterate_continue_bool
2781   \cs_set:Npn \__stex_iterate_morphism_cs:nnnn ##1 ##2 ##3 ##4 ##5 {%
2782     #2
2783     \bool_if:NT \l__stex_iterate_continue_bool {%
2784       \str_if_eq:nnTF{##1}{##2}{%
2785         \tl_put_right:Nn \l__stex_iterate_todo_tl {%
2786           \__stex_iterate_morphism:nn{##5}{##2}
2787         }%
2788       }{%
2789         \tl_put_right:Nn \l__stex_iterate_todo_tl {%
2790           \__stex_iterate_morphism:nn{##5 / ##1}{##2}
2791         }%
2792       }%
2793     }%
2794   }%
2795   \cs_set:Npn \stex_iterate_break:n ##1 {%
2796     \bool_set_false:N \l__stex_iterate_continue_bool
2797     \prop_map_break:n{##1}
```

```

2798     }
2799     \__stex_iterate_iterate_morphism:nn{}{#1}
2800 }
2801
2802 \cs_new_protected:Nn \__stex_iterate_iterate_morphism:nn {
2803     \tl_clear:N \l__stex_iterate_todo_tl
2804     \seq_if_in:NnF \l__stex_iterate_mods_seq {#1 #2} {
2805         \seq_put_right:Nn \l__stex_iterate_mods_seq {#1 #2}
2806         \prop_map_inline:cn{c_stex_module_#2_morphisms_prop} {
2807             \__stex_iterate_morphism:cs:nnnn ##2 {#1}
2808             % TODO
2809             % ##1: name or [mpath]
2810             % ##2 = {#####1}{#####2}{#####3}{#####4}
2811             % #####1 = name
2812             % #####2 = mpath
2813             % #####3 = type
2814             % #####4 = {origname}{newname}*
2815         }
2816         \bool_if:NT \l__stex_iterate_continue_bool \l__stex_iterate_todo_tl
2817     }
2818 }
```

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

### 13.6.2 Structural Features

```

2819 <@=stex_features>
2820
\stex_structural_feature_module:nn
\stex_structural_feature_module_end:
2820 \cs_new_protected:Nn \stex_structural_feature_module:nn {
2821     \stex_if_do_html:TF {
2822         \exp_args:Nne \begin{stex_annotation_env} {
2823             data-shtml-feature-#2={
2824                 \l_stex_current_module_str/#1}
2825             \str_if_empty:NF \l_stex_mroname_str {,
2826                 data-shtml-mroname={\l_stex_mroname_str}
2827             }
2828         }
2829         \stex_annotation_invisible:n{}
2830     }\group_begin:
2831     \stex_module_setup:n {#1-module}
2832 }
2833
2834 \cs_new_protected:Nn \stex_structural_feature_module_end: {
2835     \tl_gset_eq:NN \g_stex_last_feature_str \l_stex_current_module_str
2836     \stex_close_module:
2837     \stex_if_do_html:TF{
2838         \end{stex_annotation_env}
2839     }\group_end:
2840 }
```

(End of definition for `\stex_structural_feature_module:nn` and `\stex_structural_feature_module_end:`. These functions are documented on page 131.)

```
\stex_structural_feature_morphism:nnn
\stex_structural_feature_morphism_end:
```

```

2841 \bool_new:N \l__stex_features_implicit_bool
2842 \cs_new_protected:Nn \stex_structural_feature_morphism:nnnnn {
2843   \str_clear:N \l_stex_current_domain_str
2844   \tl_if_empty:nT{#3} {
2845     \stex_get_mathstructure:n{#4}
2846     \str_set_eq:NN \l_stex_current_domain_str \l_stex_get_structure_module_str
2847   }
2848   \str_if_empty:NT \l_stex_current_domain_str {
2849     \stex_import_module_uri:nn { #3 }{ #4 }
2850     \group_begin:
2851     \stex_import_require_module:ooo
2852       \l_stex_import_archive_str
2853       \l_stex_import_path_str
2854       \l_stex_import_name_str
2855     \exp_args:Nnx \use:nn \group_end: {
2856       \str_set:Nn \exp_not:N\l_stex_current_domain_str {\l_stex_import_ns_str}
2857     }
2858   }
2859   \tl_if_empty:nTF{#1} {
2860     \bool_set_true:N \l__stex_features_implicit_bool
2861     \str_set:Nx \l_tmpa_str {[ \l_stex_current_domain_str ] }
2862   }{
2863     \bool_set_false:N \l__stex_features_implicit_bool
2864     \str_set:Nn \l_tmpa_str {#1}
2865   }
2866
2867 \stex_if_do_html:TF {
2868   \begin{stex_annotation_env} {
2869     data-shtml-feature-#2={\l_stex_current_module_str?\l_tmpa_str},
2870     data-shtml-domain={\l_stex_current_domain_str}
2871     #5
2872   }
2873   \stex_annotation_invisible:n{ }
2874 } \group_begin:
2875 \str_set:Nn \l__stex_features_feature_str {#2}
2876 \str_set_eq:NN \l_stex_feature_name_str \l_tmpa_str
2877 \__stex_features_setup:
2878 \__stex_features_reactivate:
2879 \%~\A\stex_activate_module:o \l_stex_current_domain_str
2880 \exp_args:Ne \stex_metagroup_new:n {\l_stex_current_module_str / \l_stex_feature_name_str}
2881 }
2882
2883 \cs_new_protected:Nn \__stex_features_do_for_list: {
2884   \seq_clear:N \l_stex_fors_seq
2885   \clist_map_inline:Nn \l_stex_key_for_clist {
2886     \exp_args:Ne\stex_get_in_morphism:n{\tl_to_str:n{##1}}
2887     \seq_put_right:Nx \l_stex_fors_seq
2888     {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
2889   }
2890 }
2891
2892 \cs_new_protected:Nn \__stex_features_add_definiens:nn {
2893   \__stex_features_set_definiens_macros: #1\__stex_features_break:
2894   \stex_assign_do:n{#2}

```

```

2895      #2
2896  }
2897 \cs_new_protected:Npn \__stex_features_set_definiens_macros: #1?#2?#3\__stex_features_break:
2898   \str_set:Nn \l_stex_get_symbol_mod_str {#1?#2}
2899   \str_set:Nn \l_stex_get_symbol_name_str {#3}
2900   \exp_args:Nne\use:nn{\__stex_features_set_definiens_macros_i:nnnnnnn}{%
2901     \prop_item:Nn \l_stex_morphism_symbols_prop {[#1?#2]/[#3]}}
2902   }
2903 }
2904 \cs_new_protected:Nn \__stex_features_set_definiens_macros_i:nnnnnnn {
2905   \tl_set:Nn \l_stex_get_symbol_def_tl{#4}
2906 }
2907
2908 \cs_new_protected:Nn \stex_structural_feature_morphism_end: {
2909   \str_gset_eq:NN \l_stex_feature_name_str \l_stex_feature_name_str
2910   \str_gset_eq:NN \l_stex_current_domain_str \l_stex_current_domain_str
2911   \seq_gset_eq:NN \l_stex_morphism_symbols_prop \l_stex_morphism_symbols_prop
2912   \seq_gset_eq:NN \l_stex_morphism_renames_prop \l_stex_morphism_renames_prop
2913   \seq_gset_eq:NN \l_stex_morphism_morphisms_seq \l_stex_morphism_morphisms_seq
2914   \__stex_features_do_elaboration:
2915   \stex_if_do_html:TF{
2916     \end{stex_annotation_env}
2917   }\group_end:
2918 }
2919
2920 \cs_new_protected:Nn \__stex_features_setup: {
2921   \prop_clear:N \l_stex_morphism_symbols_prop
2922   \prop_clear:N \l_stex_morphism_renames_prop
2923   \seq_clear:N \l_stex_morphism_morphisms_seq
2924   \__stex_features_do_decls:
2925   \exp_args:No \__stex_features_do_morphisms:n \l_stex_current_domain_str
2926 }
2927
2928 \cs_new_protected:Nn \__stex_features_rename_all: {
2929
2930 }
2931
2932 \cs_new:Npn \__stex_features_clean:nnw [#1] / #2 \stex_end: {
2933   [#1]/[#2]
2934 }
2935
2936 \cs_new_protected:Nn \__stex_features_do_decls: {
2937   \exp_args:No \stex_iterate_symbols:nn \l_stex_current_domain_str {
2938     \stex_if_starts_with:nnTF{##3}[{
2939       \exp_args:NNe \prop_put:Nnn \l_stex_morphism_symbols_prop {
2940         \__stex_features_clean:nnw ##3 \stex_end:
2941       }
2942     }{
2943       \prop_put:Nnn \l_stex_morphism_symbols_prop
2944         {[##1]/[##3]}
2945     }{
2946       {[##2]{##4}{##5}{##6}{##7}{##8}##9
2947     }
2948 }

```

```

2949 }
2950
2951 \cs_new_protected:Nn \stex_structural_feature_morphism_check_total: {
2952     \prop_map_inline:Nn \l_stex_morphism_symbols_prop {
2953         \__stex_features_total_check: ##1 ##2
2954     }
2955 }
2956
2957 \cs_new_protected:Npn \__stex_features_total_check: [#1]/[#2] #3 #4 #5 #6 #7 #8 #9 {
2958     \tl_if_empty:nT{#6} {
2959         \msg_error:nnxx{stex}{error/needsdefiniens}{#1?#2}{total~morphism}
2960     }
2961 }
2962
2963 \cs_new:Npn \__stex_features_split_qm:w #1 ? #2 ? #3 { #3 }
2964 \cs_new_protected:Nn \__stex_features_do_elaboration: {
2965     \stex_debug:nn{morphisms} {
2966         Elaborating:^^J\prop_to_keyval:N \l_stex_morphism_symbols_prop
2967         ^^J
2968         Renamings:^^J
2969         \prop_to_keyval:N \l_stex_morphism_renames_prop
2970     }
2971     \prop_map_inline:Nn \l_stex_morphism_symbols_prop {
2972         \__stex_features_elab_check: ##1 ##2
2973     }
2974     \exp_args:No\stex_iterate_notations:nn\l_stex_current_domain_str{
2975         \prop_get:NnNTF \l_stex_morphism_renames_prop {##1}\l_stex_features_tmp {
2976             \exp_args:Ne \stex_module_add_notation:nnnnn
2977             {\l_stex_current_module_str ? \exp_after:wN \use_i:nn \l__stex_features_tmp}
2978         }{
2979             \exp_args:Ne \stex_module_add_notation:nnnnn
2980             {\l_stex_current_module_str ? \l_stex_feature_name_str
2981                 / \__stex_features_split_qm:w ##1}
2982             }{##2}{##3}{##4}{##5}
2983     }
2984     \stex_module_add_morphism:ooox
2985         \l_stex_feature_name_str
2986         \l_stex_current_domain_str
2987         \l__stex_features_feature_str
2988         {\prop_map_function:NN \l_stex_morphism_renames_prop \__stex_features_rename:nn}
2989     }
2990
2991 \cs_new:Nn \__stex_features_rename:nn{
2992     {#1}{\use_i:nn#2}
2993 }
2994
2995 \cs_new_protected:Npn \__stex_features_elab_check: [#1]/[#2] #3 {
2996     \prop_get:NnNTF \l_stex_morphism_renames_prop {#1?#2}\l_stex_features_tmp {
2997         \stex_debug:nn{morphisms}{Generating~\l_stex_features_tmp}
2998         \exp_after:wN \stex_module_add_symbol:nnnnnnnnN \l_stex_features_tmp
2999     }{
3000         \bool_if:NTF \l__stex_features_implicit_bool {
3001             \stex_debug:nn{morphisms}{Generating~#3:~\l_stex_feature_name_str / #2}
3002             \exp_args:Nno \stex_module_add_symbol:nnnnnnnnN {#3}{\l_stex_feature_name_str / #2}

```

```

3003     }{
3004         \stex_debug:nn{morphisms}{Generating~\l_stex_feature_name_str / #2}
3005         \exp_args:Nno \stex_module_add_symbol:nnnnnnnN {}{\l_stex_feature_name_str / #2}
3006     }
3007 }
3008 }
3009
3010 \cs_new_protected:Nn \__stex_features_do_morphisms:n {
3011     \prop_map_inline:cn {c_stex_module_#1_morphisms_prop} {
3012         \__stex_features_do_morph:nnnn ##2
3013     }
3014 }
3015
3016 \cs_new_protected:Nn \__stex_features_do_morph:nnnn {
3017     \tl_if_empty:nF{#3} {
3018         \seq_put_right:Nn \l_stex_morphism_morphisms_seq {{#1}{#2}{#3}}
3019     }
3020     \__stex_features_do_morphisms:n{#2}
3021 }
3022
3023 \cs_new_protected:Npn \__stex_features_reactivate: {
3024     \stex_deactivate_macro:Nn \symdecl {module~environments}
3025     \stex_deactivate_macro:Nn \textsymdecl {module~environments}
3026     \stex_deactivate_macro:Nn \symdef {module~environments}
3027     \stex_deactivate_macro:Nn \notation {module~environments}
3028     \stex_deactivate_macro:Nn \importmodule {module~environments}
3029     \stex_deactivate_macro:Nn \requiremodule {module~environments}
3030     \stex_deactivate_macro:Nn \smodule {outside~of~morphisms}
3031     \stex_reactivate_macro:N \assign
3032     \stex_reactivate_macro:N \assignMorphism
3033     \stex_reactivate_macro:N \renamedecl
3034     \cs_set_eq:NN \stex_do_for_list: \__stex_features_do_for_list:
3035     \cs_set_eq:NN \stex_add_definiens:nn \__stex_features_add_definiens:nn
3036 }

```

(End of definition for `\stex_structural_feature_morphism:nnn` and `\stex_structural_feature_morphism_end::`. These functions are documented on page ??.)

`\stex_get_in_morphism:n`

```

3037 \cs_new_protected:Nn \stex_get_in_morphism:n {
3038     \str_clear:N \l_stex_get_symbol_name_str
3039     \prop_map_inline:Nn \l_stex_morphism_symbols_prop {
3040         \exp_args:Nx \__stex_features_get_check:nnnn{\tl_to_str:n{#1}}##1##2
3041     }
3042     \str_if_empty:NT \l_stex_get_symbol_name_str {
3043         \prop_map_inline:NN \l_stex_morphism_renames_prop {
3044             \__stex_features_renamed_check:nnnnn{#1}##1##2
3045         }
3046         \str_if_empty:NT \l_stex_get_symbol_name_str {
3047             \msg_error:nnxx{stex}{error/unknownsymbolin}{#1}{
3048                 morphism~\l_stex_feature_name_str
3049             }
3050         }
3051 }

```

```

3052 }
3053
3054 \cs_new_protected:Npn \__stex_features_renamed_check:nnnnn #1#2#3#4=#5#6 {
3055     \str_if_eq:nnTF{#1}{#5}{
3056         \exp_args:Nnx \use:nn{\__stex_features_check_break:nnnnnnnnn{#2?#3}{#4}}{
3057             \prop_item:Nn \l_stex_morphism_symbols_prop {[#2?#3]/[#4]}
3058         }
3059     }{
3060         \str_if_eq:nnT{#1}{#6}{
3061             \exp_args:Nnx \use:nn{\__stex_features_check_break:nnnnnnnnn{#2?#3}{#4}}{
3062                 \prop_item:Nn \l_stex_morphism_symbols_prop {[#2?#3]/[#4]}
3063             }
3064         }
3065     }
3066 }
3067
3068 \cs_new_protected:Npn \__stex_features_get_check:nnnn #1[#2]/[#3]#4 {
3069     \str_if_eq:nnTF{#1}{#3}{
3070         \__stex_features_check_break:nnnnnnnnn{#2}{#3}{#4}
3071     }{
3072         \str_if_eq:nnTF{#1}{#4}{
3073             \__stex_features_check_break:nnnnnnnnn{#2}{#3}{#4}
3074         }{
3075             \use_none:nnnnn
3076         }
3077     }
3078 }
3079
3080 \cs_new_protected:Nn \__stex_features_check_break:nnnnnnnnn {
3081     \prop_map_break:n{
3082         \str_set:Nn \l_stex_get_symbol_mod_str{#1}
3083         \str_set:Nn \l_stex_get_symbol_name_str{#2}
3084         \str_set:Nn \l_stex_get_symbol_macro_str{#3}
3085         \int_set:Nn \l_stex_get_symbol_arity_int {#4}
3086         \tl_set:Nn \l_stex_get_symbol_args_tl {#5}
3087         \tl_set:Nn \l_stex_get_symbol_def_tl {#6}
3088         \tl_set:Nn \l_stex_get_symbol_type_tl {#7}
3089         \tl_set:Nn \l_stex_get_symbol_return_tl {#8}
3090         \tl_set:Nn \l_stex_get_symbol_invoke_cs {#9}
3091     }
3092 }

```

(End of definition for `\stex_get_in_morphism:n`. This function is documented on page 134.)

## 13.7 Inheritance

### 13.7.1 \importmodule/\usemodule

```
3093 <@=stex_importmodule>
```

```
\usemodule
3094 \stex_new_stylable_cmd:nnnn {usemodule} { 0{} m } {
3095     \stex_import_module_uri:nn { #1 }{ #2 }
3096     \stex_import_require_module:ooo
```

```

3097   \l_stex_import_archive_str
3098   \l_stex_import_path_str
3099   \l_stex_import_name_str
3100 \stex_if_do_html:T {
3101   \hbox{\stex_annotation_invisible:nn
3102     {data-shtml-usemodule=\l_stex_import_ns_str} {}}
3103 }
3104 \stex_if_smsmode:F{
3105   \group_begin:
3106   \tl_set_eq:NN \thismoduleuri \l_stex_import_ns_str
3107   \tl_set_eq:NN \thismodulename \l_stex_import_name_str
3108   \tl_clear:N \thisstyle
3109   \stex_style_apply:
3110   \group_end:
3111 }
3112 }{\aftergroup\ignorespaces}

```

(End of definition for `\usemodule`. This function is documented on page 96.)

`\stex_import_module_uri:nn`

```

3113 \cs_new_protected:Nn \stex_import_module_uri:nn {
3114   \stex_debug:nn{importmodule}{URI:~#1<~#2~}
3115   \exp_args:NNnx \seq_set_split:Nnn \__stex_importmodule_seq ? { \tl_to_str:n{ #2 } }
3116   \seq_pop_right:NN \__stex_importmodule_seq \l_stex_import_name_str
3117   \str_set:Nx \l_stex_import_path_str { \seq_use:Nn \__stex_importmodule_seq ? }
3118   \tl_if_empty:nTF { #1 } {
3119     \stex_debug:nn{importmodule}{No~archive}
3120     \prop_if_exist:NTF \l_stex_current_archive_prop {
3121       \stex_debug:nn{importmodule}{Picking~current~archive}
3122       \str_set:Nx \l_stex_import_archive_str {
3123         \prop_item:Nn \l_stex_current_archive_prop { id }
3124       }
3125     }{
3126       \str_clear:N \l_stex_import_archive_str
3127       \str_set:Nn \l_stex_import_uri_str {file:}
3128       \str_if_empty:NTF \l_stex_import_path_str {
3129         \stex_debug:nn{importmodule}{Empty~Path}
3130         \stex_file_split_off_ext:NN \l__stex_importmodule_path_seq \g_stex_current_file
3131         \stex_file_split_off_lang:NN \l__stex_importmodule_path_seq \l__stex_importmodule_pa
3132         \str_set:Nx \l_stex_import_path_str {
3133           \stex_file_use:N \l__stex_importmodule_path_seq
3134         }
3135       }{
3136         \stex_debug:nn{importmodule}{Resolving~path~\l_stex_import_path_str~relative~to~\ste
3137         \stex_file_resolve:Nx \l__stex_importmodule_seq { \stex_file_use:N \g_stex_current_f
3138         \str_set:Nx \l_stex_import_path_str {
3139           \stex_file_use:N \l__stex_importmodule_seq
3140         }
3141         \stex_debug:nn{importmodule}{...yields~\l_stex_import_path_str}
3142       }
3143     }
3144   }{
3145     \stex_debug:nn{importmodule}{Archive~#1}
3146     \str_set:Nx \l_stex_import_archive_str { #1 }

```

```

3147   \stex_require_archive:o \l_stex_import_archive_str
3148   \str_set:Nx \l_stex_import_uri_str { \prop_item:cn{ c_stex_mathhub_ \l_stex_import_archive_str }
3149 }
3150 }

(End of definition for \stex_import_module_uri:nn. This function is documented on page 133.)
```

```

\stex_import_require_module:nnn
\stex_import_require_module:ooo
3151 \cs_new_protected:Npn \stex_import_require_module:nnn #1 {
3152   \tl_if_empty:nTF { #1 } {
3153     \str_clear:N \l__stex_importmodule_archive_str
3154     \str_set:Nn \l_stex_import_uri_str {file:{}}
3155     \__stex_importmodule_get_module:nnn {}
3156   }{
3157     \stex_require_archive:n { #1 }
3158     \str_set:Nx \l__stex_importmodule_archive_str {#1}
3159     \str_set:Nx \l_stex_import_uri_str { \prop_item:cn{ c_stex_mathhub_ #1 _manifest_prop}{}
3160     \str_set:Nx \l__stex_importmodule_str { \stex_file_use:N \c_stex_mathhub_file / #1 / sou
3161     \exp_args:No \__stex_importmodule_get_module:nnn \l__stex_importmodule_str
3162   }
3163 }
3164 \cs_generate_variant:Nn \stex_import_require_module:nnn {ooo}
3165
3166 \cs_new_protected:Npn \stex_import_require_module_safe:nnn #1 {
3167   \tl_if_empty:nTF { #1 } {
3168     \str_clear:N \l__stex_importmodule_archive_str
3169     \str_set:Nn \l_stex_import_uri_str {file:{}}
3170     \__stex_importmodule_get_module_safe:nnn {}
3171   }{
3172     \stex_require_archive:n { #1 }
3173     \str_set:Nx \l__stex_importmodule_archive_str {#1}
3174     \str_set:Nx \l_stex_import_uri_str { \prop_item:cn{ c_stex_mathhub_ #1 _manifest_prop}{}
3175     \str_set:Nx \l__stex_importmodule_str { \stex_file_use:N \c_stex_mathhub_file / #1 / sou
3176     \exp_args:No \__stex_importmodule_get_module_safe:nnn \l__stex_importmodule_str
3177   }
3178 }
3179
3180 \cs_new_protected:Nn \__stex_importmodule_get_module_uri:nnn {
3181   \tl_if_empty:nF {#2} {
3182     \str_set:Nx \l_stex_import_uri_str {\l_stex_import_uri_str / #2}
3183   }
3184 \stex_debug:nn{importmodule}{~>#1<^J>#2<^J>#3<^J>\l_stex_import_uri_str<^J
3185   Current~file:\stex_file_use:N \g_stex_current_file^J
3186   Current~namespace:\stex_uri_use:N \l_stex_current_ns_uri
3187 }
3188 \stex_if_module_exists:nTF {\l_stex_import_uri_str?#3} {
3189   \str_set:Nx \l_stex_import_ns_str {\l_stex_import_uri_str?#3}
3190 }{
3191   \stex_if_module_exists:nTF{\stex_uri_use:N \l_stex_current_ns_uri ? #3} {
3192     \str_set:Nx \l_stex_import_ns_str {\stex_uri_use:N \l_stex_current_ns_uri ? #3}
3193   }{
3194     \__stex_importmodule_get_from_file:nnn{#1}{#2}{#3}
3195     \str_set:Nx \l_stex_import_ns_str {\l_stex_import_uri_str?#3}
3196   }
3197 }
```

```

3197   }
3198 }
3199 \cs_new_protected:Nn \__stex_importmodule_get_module_uri_safe:nnn {
3200   \tl_if_empty:nF {#2} {
3201     \str_set:Nx \l_stex_import_uri_str {\l_stex_import_uri_str / #2}
3202   }
3203   \stex_debug:nn{importmodule}{~>#1<^^J>#2<^^J>#3<^^J>\l_stex_import_uri_str<^^J
3204     Current~file:\stex_file_use:N \g_stex_current_file<^^J
3205     Current~namespace:\stex_uri_use:N \l_stex_current_ns_uri
3206   }
3207   \stex_if_module_exists:nTF {\l_stex_import_uri_str?#3} {
3208     \str_set:Nx \l_stex_import_ns_str {\l_stex_import_uri_str?#3}
3209   }{
3210     \stex_if_module_exists:nTF{\stex_uri_use:N \l_stex_current_ns_uri ? #3} {
3211       \str_set:Nx \l_stex_import_ns_str {\stex_uri_use:N \l_stex_current_ns_uri ? #3}
3212     }{
3213       \__stex_importmodule_get_from_file_safe:nnn{#1}{#2}{#3}
3214       \str_set:Nx \l_stex_import_ns_str {\l_stex_import_uri_str?#3}
3215     }
3216   }
3217 }
3218
3219 \cs_new_protected:Nn \__stex_importmodule_get_module:nnn {
3220   \stex_debug:nn{importmodule}{Requiring~>[#1]#2?#3<}
3221   \__stex_importmodule_get_module_uri:nnn{#1}{#2}{#3}
3222   \stex_activate_module:o \l_stex_import_ns_str
3223 }
3224
3225 \cs_new_protected:Nn \__stex_importmodule_get_module_safe:nnn {
3226   \stex_debug:nn{importmodule}{Requiring~>[#1]#2?#3<}
3227   \__stex_importmodule_get_module_uri_safe:nnn{#1}{#2}{#3}
3228 }
3229
3230 \cs_new_protected:Nn \__stex_importmodule_get_from_file:nnn {
3231   \stex_file_resolve:Nx \l_stex_importmodule_seq {\tl_if_empty:nF{ #1 }{ #1 / } #2 }
3232   \str_set:Nx \l_stex_importmodule_str {\stex_file_use:N \l_stex_importmodule_seq}
3233   \stex_debug:nn{imports}{Looking~for~\l_stex_import_uri_str?#3...}
3234   \__stex_importmodule_check_file:nn{ /#3.tex }{
3235     \__stex_importmodule_check_file:nn{/#3.\l_stex_current_language_str .tex}{
3236       \__stex_importmodule_check_file:nn{/#3.en.tex}{
3237         \__stex_importmodule_check_file:nn{.tex}{
3238           \__stex_importmodule_check_file:nn{.\l_stex_current_language_str .tex}{
3239             \__stex_importmodule_check_file:nn{.en.tex}{
3240               \msg_error:nnx{stex}{error/unknownmodule}{\l_stex_import_uri_str?#3}
3241             }
3242           }
3243         }
3244       }
3245     }
3246   }
3247   \stex_if_smsmode:TF{
3248     \exp_args:NNo \exp_args:Nnx \str_if_eq:nnTF{\l__stex_importmodule_str}{\stex_file_use:N
3249       \stex_debug:nn{imports}{Skipping~current~file}
3250   }

```

```

3251     \__stex_importmodule_load_file:n{#3}
3252   }
3253   }{
3254     \__stex_importmodule_load_file:n{#3}
3255   }
3256 }
3257 \cs_new_protected:Nn \__stex_importmodule_get_from_file_safe:nnn {
3258   \stex_file_resolve:Nx \l__stex_importmodule_seq { \tl_if_empty:nF{ #1 }{ #1 / } #2 }
3259   \str_set:Nx \l__stex_importmodule_str { \stex_file_use:N \l__stex_importmodule_seq }
3260   \stex_debug:nn{imports}{Looking-for-\l_stex_import_uri_str?#3...}
3261   \__stex_importmodule_check_file:nn{ /#3.tex }{
3262     \__stex_importmodule_check_file:nn{/#3.\l_stex_current_language_str .tex}{
3263       \__stex_importmodule_check_file:nn{/#3.en.tex}{
3264         \__stex_importmodule_check_file:nn{.tex}{
3265           \__stex_importmodule_check_file:nn{.\l_stex_current_language_str .tex}{
3266             \__stex_importmodule_check_file:nn{.en.tex}{

3267           }
3268         }
3269       }
3270     }
3271   }
3272 }
3273 \stex_if_smsmode:TF{
3274   \exp_args:NNo \exp_args:Nnx \str_if_eq:nnTF{\l__stex_importmodule_str}{\stex_file_use:N
3275     \stex_debug:nn{imports}{Skipping-current-file}}
3276   }{
3277     \IfFileExists{ \l__stex_importmodule_str }{
3278       \__stex_importmodule_load_file:n{#3}
3279     }{}
3280   }
3281 }{
3282   \IfFileExists{ \l__stex_importmodule_str }{
3283     \__stex_importmodule_load_file:n{#3}
3284   }{}
3285 }
3286 }
3287
3288 \cs_new_protected:Nn \__stex_importmodule_load_file:n {
3289   \stex_file_in_smsmode:on \l__stex_importmodule_str {
3290     \str_if_empty:NF \l__stex_importmodule_archive_str {
3291       \stex_set_current_archive:n \l__stex_importmodule_archive_str
3292     }
3293     \stex_debug:nnf{modules}{Loading~\l__stex_importmodule_str}
3294   }
3295   \stex_if_module_exists:nF {\l_stex_import_uri_str?#1} {
3296     \msg_error:nnx{stex}{error/unknownmodule}{\l_stex_import_uri_str?#1}
3297   }
3298 }
3299
3300 \cs_new_protected:Npn \__stex_importmodule_check_file:nn #1 {
3301   \stex_debug:nn{imports}{Checking~ \l__stex_importmodule_str #1}
3302   \IfFileExists{ \l__stex_importmodule_str #1 }{
3303     \stex_debug:nn{imports}{Success}
3304     \str_set:Nx \l__stex_importmodule_str { \l__stex_importmodule_str #1 }

```

```

3305   }
3306 }

(End of definition for \stex_import_require_module:nnn. This function is documented on page 133.)

\importmodule
3307 \stex_new_stylable_cmd:nnnn{importmodule} { 0{} m } {
3308   \__stex_importmodule_import_module:nn {#1}{#2}
3309   \stex_smsmode_do:
3310 }()
3311 \stex_deactivate_macro:Nn \importmodule {module~environments}
3312
3313 \cs_new_protected:Nn \__stex_importmodule_import_module:nn {
3314   \stex_import_module_uri:nn { #1 }{ #2 }
3315   \stex_import_require_module:ooo
3316   \l_stex_import_archive_str
3317   \l_stex_import_path_str
3318   \l_stex_import_name_str
3319   \stex_execute_in_module:x{
3320     \stex_activate_module:n{\l_stex_import_ns_str}
3321   }
3322   \stex_module_add_morphism:nonn
3323   {}{\l_stex_import_ns_str}{import}={}
3324   \stex_if_do_html:T {
3325     \stex_annotation_invisible:nn
3326     {data-shtml-import=\l_stex_import_ns_str} {}
3327   }
3328   \stex_if_smsmode:F{
3329     \group_begin:
3330     \tl_set:Nn \thisarchive {#1}
3331     \tl_set_eq:NN \thismoduleuri \l_stex_import_ns_str
3332     \tl_set_eq:NN \thismodulename \l_stex_import_name_str
3333     \tl_clear:N \thisstyle
3334     \stex_style_apply:
3335     \group_end:
3336   }
3337 }
3338
3339 \cs_new_protected:Nn \__stex_importmodule_import_module_presms:nn {
3340   \stex_import_module_uri:nn { #1 }{ #2 }
3341   \tl_gput_right:Nx \g_stex_sms_import_code {
3342     \stex_import_require_module_safe:nnn
3343     {\l_stex_import_archive_str}
3344     {\l_stex_import_path_str}
3345     {\l_stex_import_name_str}
3346   }
3347 }
3348
3349 \stex_sms_allow_escape:N \importmodule
3350 \stex_every_module:n {\stex_reactivate_macro:N \importmodule}
3351 \stex_sms_allow_import:Nn \importmodule {
3352   \stex_reactivate_macro:N \importmodule
3353   \let \__stex_importmodule_import_module:nn \__stex_importmodule_import_module_presms:nn
3354 }

```

```

3355 \stex_new_stylable_cmd:nnnn{requiremodule} { 0{} m } {
3356   \stex_import_module_uri:nn { #1 }{ #2 }
3357   \stex_import_require_module:ooo
3358     \l_stex_import_archive_str
3359     \l_stex_import_path_str
3360     \l_stex_import_name_str
3361   \stex_do_up_to_module:x{
3362     \stex_activate_module:n{\l_stex_import_ns_str}
3363   }
3364   \stex_if_do_html:T {
3365     \stex_annotation_invisible:nn
3366       {data-shtml-import=\l_stex_import_ns_str} {}
3367   }
3368   \stex_if_smsmode:F{
3369     \group_begin:
3370     \tl_set:Nn \thisarchive {#1}
3371     \tl_set_eq:NN \thismoduleuri \l_stex_import_ns_str
3372     \tl_set_eq:NN \thismodulenname \l_stex_import_name_str
3373     \tl_clear:N \thisstyle
3374     \stex_style_apply:
3375     \group_end:
3376   }
3377   \stex_smsmode_do:
3378 }{}
3380 \stex_deactivate_macro:Nn \requiremodule {module-environments}

```

(End of definition for `\importmodule`. This function is documented on page 96.)

### 13.7.2 Theory Morphisms

```

3381 <@@=stex_morphisms>
3382 \stex_new_stylable_cmd:nnnn {assign} { m m }{
3383   \stex_get_in_morphism:n{#1}
3384   \_stex_assign_do:n{#2}
3385   \stex_smsmode_do:
3386 }{}
3387 \stex_sms_allow_escape:N\assign
3388
3389 \cs_new_protected:Nn \_stex_assign_do:n{
3390   \stex_debug:nn{assign}{Assigning~\l_stex_get_symbol_name_str~to~\tl_to_str:n{#1}}
3391   \tl_if_empty:NF \l_stex_get_symbol_def_tl {
3392     \%msg_error:nnxx{stex}{error/symbolalreadydefined}{\l_stex_get_symbol_name_str}{
3393       % morphism~\l_stex_feature_name_str
3394     %}
3395   }
3396   \stex_check_term:n{#1}
3397   \stex_debug:nn{HERE!}{
3398     \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str^~J
3399     \tl_to_str:n{#1}
3400   }
3401   \stex_if_do_html:T{
3402     \stex_annotation_invisible:nn{data-shtml-assign={\l_stex_get_symbol_mod_str?\l_stex_get_sy
3403       \_stex_annotation_force_break:n{

```

```

3404     \mode_if_math:T\hbox{\$\\stex_annotate:nn{data-shtml-definiens={}{}#1$}
3405     }
3406   }
3407 }
3408 \exp_args:Ne \stex_metagroup_do_in:nx{
3409   \l_stex_current_module_str / \l_stex_feature_name_str
3410 }{
3411   \prop_put:Nnn \exp_not:N \l_stex_morphism_symbols_prop
3412   {[\\l_stex_get_symbol_mod_str]/[\\l_stex_get_symbol_name_str]}
3413   {
3414     {\\l_stex_get_symbol_macro_str}
3415     {int_use:N \\l_stex_get_symbol_arity_int}
3416     {\\l_stex_get_symbol_args_t1}
3417     {\exp_not:n#1}
3418     {\exp_args:No\exp_not:n\\l_stex_get_symbol_type_t1}
3419     {\exp_args:No\exp_not:n\\l_stex_get_symbol_return_t1}
3420     {\\l_stex_get_symbol_invoke_cs}
3421   }
3422 }
3423 }
3424
3425
3426 \stex_new_stylable_cmd:nnnn {renamedecl} { m 0{} m }{
3427   \stex_get_in_morphism:n#1
3428   \stex_renamedecl_do:nn#2#3
3429   \stex_smsmode_do:
3430 }{}
3431 \stex_sms_allow_escape:N\renamedecl
3432
3433 \cs_new_protected:Nn \stex_renamedecl_do:nn {
3434   \stex_debug:nn{renamedecl}{Renaming\\l_stex_get_symbol_name_str-to-[#1]{#2}}
3435   \stex_if_do_html:T{
3436     \exp_args:Ne \stex_annotate_invisible:nn{
3437       data-shtml-rename={\\l_stex_get_symbol_mod_str}?\\l_stex_get_symbol_name_str},
3438       data-shtml-macroname=#2
3439       \str_if_empty:nF{#1}{ ,data-shtml-to={#1} }
3440   }{}
3441 }
3442 \exp_args:Ne \stex_metagroup_do_in:nx{
3443   \l_stex_current_module_str / \l_stex_feature_name_str
3444 }{
3445   \prop_put:Nnn \exp_not:N \l_stex_morphism_renames_prop
3446   {[\\l_stex_get_symbol_mod_str]?\\l_stex_get_symbol_name_str}{#2}[
3447     \tl_if_empty:nTF{#1}{\\l_stex_feature_name_str/\\l_stex_get_symbol_name_str}{#1}
3448   ]}
3449 }
3450 }
3451
3452 \stex_new_stylable_cmd:nnnn {assignMorphism} { m m }{
3453   \str_clear:N \\l_stex_morphisms_morphism_dom_str
3454   \stex_iterate_morphisms:nn\\l_stex_current_domain_str{
3455     \stex_debug:nn{assignMorphism}{%
3456       Checking:~#1-vs:^^J##1^^J##2^^J##3^^J##4
3457     }

```

```

3458 \str_if_eq:nnTF{#1}{##1}{
3459     \__stex_morphisms_do_morph_assign:nnn{##1}{##2}{##4}
3460 }{
3461     \stex_if_ends_with:nnT{##2}{#1}{
3462         \__stex_morphisms_do_morph_assign:nnn{##1}{##2}{##4}
3463     }
3464 }
3465 }
3466 \str_if_empty:NT \l__stex_morphisms_morphism_dom_str {
3467     \msg_error:nnn{stex}{error/nomorphism}{#1}
3468 }
3469 \bool_set_false:N \l_tmpa_bool
3470 \stex_iterate_morphisms:nn \l_stex_current_module_str {
3471     \stex_debug:nn{assignMorphism}{}
3472     Checking:~#2~vs:^^J##1^^J##2^^J##3^^J##4
3473 }
3474 \str_if_eq:nnTF{#2}{##1}{
3475     \stex_debug:nn{assignMorphism}{match!}
3476     \stex_iterate_break:n{
3477         \stex_annotation_invisible:nn{
3478             data-shtml-assignmorphismfrom={\l__stex_morphisms_morphism_dom_str}
3479             data-shtml-assignmorphismto={\l_stex_current_module_str?##1}
3480         }{}
3481         \bool_set_true:N \l_tmpa_bool
3482     }
3483 }
3484 \stex_if_ends_with:nnT{##2}{#2}{
3485     \stex_debug:nn{assignMorphism}{match!}
3486     \stex_iterate_break:n{
3487         \stex_annotation_invisible:nn{
3488             data-shtml-assignmorphismfrom={\l__stex_morphisms_morphism_dom_str},
3489             data-shtml-assignmorphismto={\l_stex_current_module_str?##1}
3490         }{}
3491         \bool_set_true:N \l_tmpa_bool
3492     }
3493 }
3494 }
3495 }
3496 \bool_if:NF \l_tmpa_bool {
3497     \msg_error:nnn{stex}{error/nomorphism}{#2}
3498 }
3499 }{}}
3500 \cs_new_protected:Nn \__stex_morphisms_do_morph_assign:nnn {
3501     \stex_iterate_break:n{
3502         \str_set:Nx \l__stex_morphisms_morphism_dom_str { \l_stex_current_domain_str ? #1 }
3503         \stex_debug:nn{assignMorphism}{match!}
3504         \stex_iterate_symbols:nn{#2}{{
3505             \stex_debug:nn{assignMorphism}{removing~##1?##3}
3506             % TODO: non-trivial assignments
3507             \prop_remove:Nn \l_stex_morphism_symbols_prop {
3508                 [##1]/[##3]
3509             }
3510         }}
3511     }

```

```

3512 }
3513
3514 \stex_deactivate_macro:Nn \assign {morphism~environments}
3515 \stex_deactivate_macro:Nn \renamedecl {morphism~environments}
3516 \stex_deactivate_macro:Nn \assignMorphism {morphism~environments}
3517 \stex_new_stylable_env:nnnnnnn {copymodule}{m 0{} m}{
3518
3519 \stex_structural_feature_morphism:nnnnn{#1}{morphism}{#2}{#3}{,data-shtml-total=false}
3520
3521 \stex_if_smsmode:F {
3522   \tl_set:Nn \thiscopyname { #1 }
3523   \tl_set_eq:NN \thismoduleuri \l_stex_current_domain_str
3524   \stex_style_apply:
3525 }
3526 \stex_smsmode_do:
3527 }{
3528 \stex_if_smsmode:F {
3529   \stex_style_apply:
3530 }
3531 \stex_structural_feature_morphism_end:
3532 }{}{}{}
3533 \stex_deactivate_macro:Nn \copymodule {module~environments}
3534 \stex_every_module:n {
3535   \stex_reactivate_macro:N \copymodule
3536 }
3537 \stex_sms_allow_env:n{copymodule}
3538
3539 \stex_new_stylable_env:nnnnnnn {interpretmodule}{m 0{} m}{
3540 \stex_structural_feature_morphism:nnnnn{#1}{morphism}{#2}{#3}{,data-shtml-total=true}
3541 \stex_if_smsmode:F {
3542   \tl_set:Nn \thiscopyname { #1 }
3543   \tl_set_eq:NN \thismoduleuri \l_stex_current_domain_str
3544   \stex_style_apply:
3545 }
3546 \stex_smsmode_do:
3547 }{
3548 \stex_structural_feature_morphism_check_total:
3549 \stex_if_smsmode:F {
3550   \stex_style_apply:
3551 }
3552 \stex_structural_feature_morphism_end:
3553 }{}{}{}
3554 \stex_deactivate_macro:Nn \interpretmodule {module~environments}
3555 \stex_every_module:n {
3556   \stex_reactivate_macro:N \interpretmodule
3557 }
3558 \stex_sms_allow_env:n{interpretmodule}
3559 \stex_new_stylable_env:nnnnnnn {realization}{0{} m}{
3560
3561 \stex_structural_feature_morphism:nnnnn{}{morphism}{#1}{#2}{,data-shtml-total=true}
3562 \%stex_execute_in_module:x{
3563 \% \stex_activate_module:n{\l_stex_current_domain_str}
3564 \%}

```

```

3565 \stex_if_smsmode:F {
3566   \tl_set:Nn \thiscopyname { #2 }
3567   \tl_set_eq:NN \thismoduleuri \l_stex_current_domain_str
3568   \stex_style_apply:
3569 }
3570 \stex_smsmode_do:
3571 }{
3572   \stex_structural_feature_morphism_check_total:
3573   \stex_if_smsmode:F {
3574     \stex_style_apply:
3575   }
3576   \stex_structural_feature_morphism_end:
3577 }{}{}{}
3578 \stex_deactivate_macro:Nn \realization {module-environments}
3579 \stex_every_module:n {
3580   \stex_reactivate_macro:N \realization
3581 }
3582 \stex_sms_allow_env:n{\realization}

3583 \cs_new_protected:Nn \__stex_morphisms_parse_assign:n {
3584   \str_clear:N \l__stex_morphisms_name_str
3585   \str_clear:N \l__stex_morphisms_newname_str
3586   \tl_clear:N \l__stex_morphisms_ass_tl
3587   \stex_debug:nn{morphisms}{Parsing-#1}
3588   \exp_args:NNN \seq_set_split:Nnn \l__stex_morphisms_seq {\tl_to_str:n{@}} {#1}
3589   \int_compare:nNnTF {\seq_count:N \l__stex_morphisms_seq} = 1 {
3590     \stex_debug:nn{morphisms}{No-@}
3591     \seq_pop_left:NN \l__stex_morphisms_seq \l__stex_morphisms_next_tl
3592   }{
3593     \seq_pop_left:NN \l__stex_morphisms_seq \l__stex_morphisms_name_str
3594     \stex_debug:nn{morphisms}{Name:~\l__stex_morphisms_name_str}
3595     \exp_args:NNN \str_set:Nn \l__stex_morphisms_name_str \l__stex_morphisms_name_str
3596     \tl_set:Nx \l__stex_morphisms_next_tl {\seq_use:Nn \l__stex_morphisms_seq @}
3597   }
3598   \exp_args:NNNN \seq_set_split:Nnn \l__stex_morphisms_seq = \l__stex_morphisms_next_tl
3599   \str_if_empty:NTF \l__stex_morphisms_name_str {
3600     \seq_pop_left:NN \l__stex_morphisms_seq \l__stex_morphisms_name_str
3601     \exp_args:NNN \str_set:Nn \l__stex_morphisms_name_str \l__stex_morphisms_name_str
3602     \tl_set:Nx \l__stex_morphisms_ass_tl {\seq_use:Nn \l__stex_morphisms_seq =}
3603   }{
3604     \seq_pop_left:NN \l__stex_morphisms_seq \l__stex_morphisms_newname_str
3605     \exp_args:NNN \str_set:Nn \l__stex_morphisms_newname_str \l__stex_morphisms_newname_str
3606     \tl_set:Nx \l__stex_morphisms_ass_tl {\seq_use:Nn \l__stex_morphisms_seq =}
3607   }
3608   \__stex_morphisms_do_parsed_assign:
3609 }
3610
3611 \cs_new_protected:Nn \__stex_morphisms_do_parsed_assign: {
3612   \exp_args:No \stex_get_in_morphism:n \l__stex_morphisms_name_str
3613   \str_if_empty:NF \l__stex_morphisms_newname_str {
3614     \exp_after:wN \__stex_morphisms_do_parsed_newname: \l__stex_morphisms_newname_str \__ste
3615   }
3616   \tl_if_empty:NF \l__stex_morphisms_ass_tl {
3617     \exp_args:No \stex_assign_do:n \l__stex_morphisms_ass_tl
3618   }

```

```

3619 }
3620
3621 \cs_new_protected:Nn \__stex_morphisms_do_parsed_newname: {
3622   \peek_charcode:NTF [ {
3623     \__stex_morphisms_do_parsed_newname:w
3624   }{
3625     \__stex_morphisms_do_parsed_newname:w []
3626   }
3627 }
3628
3629 \cs_new_protected:Npn \__stex_morphisms_do_parsed_newname:w [#1] #2 \__stex_morphisms_end: {
3630   \stex_renamedecl_do:nn{#1}{#2}
3631 }
3632
3633 \stex_new_stylable_cmd:nnnn{copymod}{m O{} m m}{
3634   \stex_structural_feature_morphism:nnnnn{#1}{morphism}{#2}{#3}{,data-shtml-total=false}
3635
3636 \clist_map_function:nN{#4}\__stex_morphisms_parse_assign:n
3637
3638 \stex_if_smsmode:F {
3639   \tl_set:Nn \thiscopyname { #1 }
3640   \tl_set_eq:NN \thismoduleuri \l_stex_current_domain_str
3641   \stex_style_apply:
3642 }
3643 \stex_structural_feature_morphism_end:
3644 \stex_smsmode_do:
3645 }{}
3646 \stex_deactivate_macro:Nn \copymod {module~environments}
3647 \stex_every_module:n {
3648   \stex_reactivate_macro:N \copymod
3649 }
3650 \stex_sms_allow_escape:N\copymod
3651
3652
3653 \stex_new_stylable_cmd:nnnn{interpretmod}{m O{} m m}{
3654   \stex_structural_feature_morphism:nnnnn{#1}{morphism}{#2}{#3}{,data-shtml-total=true}
3655
3656 \clist_map_function:nN{#4}\__stex_morphisms_parse_assign:n
3657
3658 \stex_if_smsmode:F {
3659   \tl_set:Nn \thiscopyname { #1 }
3660   \tl_set_eq:NN \thismoduleuri \l_stex_current_domain_str
3661   \stex_style_apply:
3662 }
3663 \stex_structural_feature_morphism_check_total:
3664 \stex_structural_feature_morphism_end:
3665 \stex_smsmode_do:
3666 }{}
3667 \stex_deactivate_macro:Nn \interpretmod {module~environments}
3668 \stex_every_module:n {
3669   \stex_reactivate_macro:N \interpretmod
3670 }
3671 \stex_sms_allow_escape:N\interpretmod
3672

```

```

3673   \stex_new_stylable_cmd:nnnn{realize}{0{} m m}{
3674     \stex_structural_feature_morphism:nnnnn{}{morphism}{#1}{#2}{,data-shtml-total=true}
3675
3676   \clist_map_function:nN{#3}\_stex_morphisms_parse_assign:n
3677
3678   \stex_if_smsmode:F {
3679     \tl_set:Nn \thiscopyname { #1 }
3680     \tl_set_eq:NN \thismoduleuri \l_stex_current_domain_str
3681     \stex_style_apply:
3682   }
3683
3684   \stex_structural_feature_morphism_check_total:
3685   \stex_structural_feature_morphism_end:
3686   \stex_smsmode_do:
3687 }{}
3688 \stex_deactivate_macro:Nn \realize {module~environments}
3689 \stex_every_module:n {
3690   \stex_reactivate_macro:N \realize
3691 }
3692 \stex_sms_allow_escape:N\realize

```

## 13.8 Symbols

### 13.8.1 Declarations

3693 `<@=stex_symdecl>`

Some setup:

```

\stex_if_check_terms_p:
\stex_if_check_terms:TF
3694 \stex_if_html_backend:TF {
3695   \prg_new_conditional:Nnn \stex_if_check_terms: {p, T, F, TF} {
3696     \prg_return_false:
3697   }
3698 }{
3699   \stex_get_env:Nn\__stex_symdecl_env_str{STEX_CHECKTERMS}
3700   \str_if_empty:NF\__stex_symdecl_env_str{
3701     \exp_args:No \str_if_eq:nnF \__stex_symdecl_env_str{false} {
3702       \bool_set_true:N \c_stex_check_terms_bool
3703     }
3704   }
3705   \bool_if:NTF \c_stex_check_terms_bool {
3706     \prg_new_conditional:Nnn \stex_if_check_terms: {p, T, F, TF} {
3707       \prg_return_true:
3708     }
3709   }{
3710     \prg_new_conditional:Nnn \stex_if_check_terms: {p, T, F, TF} {
3711       \prg_return_false:
3712     }
3713   }
3714 }

```

(End of definition for `\stex_if_check_terms:TF`. This function is documented on page 127.)

```

\stex_check_term:n
3715 \stex_if_check_terms:TF{

```

```

3716   \cs_new_protected:Nn \stex_check_term:n {
3717     \hbox_set:Nn \l_tmpa_box {
3718       \group_begin:
3719         $$#1$$
3720       \group_end:
3721     }
3722   }
3723 }{
3724   \cs_new_protected:Nn \stex_check_term:n {}
3725 }

```

(End of definition for `\stex_check_term:n`. This function is documented on page 127.)  
`symdecl` arguments:

```

3726 \stex_keys_define:nnnn{symargs}{
3727   \str_clear:N \l_stex_key_args_str
3728   \str_clear:N \l_stex_key_role_str
3729   \str_clear:N \l_stex_key_reorder_str
3730   \str_clear:N \l_stex_key_assoc_str
3731 }{
3732   args      .str_set:N = \l_stex_key_args_str ,
3733   reorder   .str_set:N = \l_stex_key_reorder_str ,
3734   assoc     .choices:nn = {bin,binl,binr,pre,conj,pwconj}
3735   {\str_set:Nx \l_stex_key_assoc_str \l_keys_choice_tl},
3736   role      .str_set:N      = \l_stex_key_role_str
3737 }{}
3738
3739 \stex_keys_define:nnnn{decl}{
3740   \str_clear:N \l_stex_key_name_str
3741   \str_clear:N \l_stex_key_args_str
3742   \tl_clear:N \l_stex_key_type_tl
3743   \tl_clear:N \l_stex_key_def_tl
3744   \tl_clear:N \l_stex_key_return_tl
3745   \str_clear:N \l_stex_key_wikidata_str
3746   \clist_clear:N \l_stex_key_argtypes_clist
3747 }{
3748   name      .str_set:N = \l_stex_key_name_str ,
3749
3750   return    .tl_set:N      = \l_stex_key_return_tl ,
3751   argtypes  .clist_set:N = \l_stex_key_argtypes_clist ,
3752   wikidata  .str_set:N      = \l_stex_key_wikidata_str ,
3753
3754   type      .tl_set:N      = \l_stex_key_type_tl ,
3755   def       .tl_set:N      = \l_stex_key_def_tl ,
3756
3757   align     .code:n        = {},
3758   gf       .code:n        = {}
3759 }{style,deprecate,symargs}
3760 % \stex_do_deprecation:n{#2}

```

```

\symdecl
3761 \str_new:N \l_stex_mroname_str
3762 \stex_new_stylable_cmd:nnnn {symdecl} { s m O{} } {
3763   \stex_keys_set:nn{decl}{#3}
3764   \IfBooleanTF #1 {

```

```

3765     \str_clear:N \l_stex_mroname_str
3766 }{
3767     \str_set:Nx \l_stex_mroname_str { #2 }
3768 }
3769 \stex_symdecl_top:n{#2}
3770
3771 \stex_if_smsmode:F{
3772     \group_begin:
3773     \tl_set:Nx \thisdecluri {\l_stex_current_module_str ? \l_stex_key_name_str}
3774     \tl_set_eq:NN \thisdeclname \l_stex_key_name_str
3775     \tl_set_eq:NN \thistype \l_stex_key_type_tl
3776     \tl_set_eq:NN \thisdefiniens \l_stex_key_def_tl
3777     \tl_set_eq:NN \thisargs \l_stex_key_args_str
3778     \tl_clear:N \thisstyle
3779     \stex_style_apply:
3780     \group_end:
3781 }
3782 \stex_smsmode_do:
3783 }{}
3784 \stex_deactivate_macro:Nn \symdecl {module~environments}
3785 \stex_every_module:n {\stex_reactivate_macro:N \symdecl}
3786 \stex_sms_allow_escape:N \symdecl

```

(End of definition for \symdecl. This function is documented on page 90.)

```

\stex_symdecl_top:n
3787 \cs_new_protected:Nn \stex_symdecl_top:n {
3788     \str_if_empty:NT \l_stex_key_name_str {
3789         \str_set:Nx \l_stex_key_name_str { #1 }
3790     }
3791     \stex_symdecl_do:
3792     \stex_symdecl_check_terms:
3793     \__stex_symdecl_add_decl:
3794     \stex_if_do_html:T {
3795         \stex_symdecl_html:
3796     }
3797 }
3798
3799 \cs_new_protected:Nn \__stex_symdecl_add_decl: {
3800     \exp_args:Nnx \use:nn {\stex_module_add_symbol:nnnnnnnN} {
3801         {\l_stex_mroname_str}
3802         {\l_stex_key_name_str}
3803         {\int_use:N \l_stex_get_symbol_arity_int}
3804         {\l_stex_get_symbol_args_tl}
3805         {\tl_if_empty:NF \l_stex_key_def_tl{DEFED} }%{\exp_args:No \exp_not:n \l_stex_key_def_tl
3806         {}%{\exp_args:No \exp_not:n \l_stex_key_type_tl}
3807         {\exp_args:No \exp_not:n \l_stex_key_return_tl}
3808         \stex_invoke_symbol:
3809     }
3810     \exp_args:Ne \stex_ref_new_symbol:n
3811         {\l_stex_current_module_str?\l_stex_key_name_str}
3812 }
3813
3814 \cs_new:Nn \stex_return_args:nn {

```

```

3815   {\svar{ARGUMENT_#1}\_stex_eat_exclamation_point:}
3816 }
3817
3818 \cs_new_protected:Nn \stex_symdecl_html: {
3819   \exp_args:Ne \stex_annotation_invisible:nn {
3820     data-shtml-symdecl = {\l_stex_current_module_str ? \l_stex_key_name_str},
3821     data-shtml-args = {\l_stex_key_args_str}
3822     \str_if_empty:NF \l_stex_mroname_str {,
3823       data-shtml-macroname={\l_stex_mroname_str}
3824     }
3825     \str_if_empty:NF \l_stex_key_wikidata_str {,
3826       data-shtml-wikidata={\l_stex_key_wikidata_str}
3827     }
3828     \str_if_empty:NF \l_stex_key_assoc_str {,
3829       data-shtml-assotype={\l_stex_key_assoc_str}
3830     }
3831     \str_if_empty:NF \l_stex_key_reorder_str {,
3832       data-shtml-reorderargs={\l_stex_key_reorder_str}
3833     }
3834     \str_if_empty:NF \l_stex_key_role_str {,
3835       data-shtml-role={\l_stex_key_role_str}
3836     }
3837 }{\hbox\bgroup\stex_annotation_force_break:n{
3838   \bool_set_true:N \stex_in_invisible_html_bool
3839   \tl_if_empty:NF \l_stex_key_type_tl {
3840     $\stex_annotation:nn{data-shtml-type={}}{\l_stex_key_type_tl}$
3841   }
3842   \tl_if_empty:NF \l_stex_key_def_tl {
3843     $\stex_annotation:nn{data-shtml-definiens={}}{\l_stex_key_def_tl}$
3844   }
3845   \tl_if_empty:NF \l_stex_key_return_tl{
3846     \exp_args:Nno \use:n{
3847       \cs_generate_from_arg_count:NNnn \l_stex_symdecl_cs
3848       \cs_set:Npn \l_stex_get_symbol_arity_int} \l_stex_key_return_tl
3849       \tl_set:Nx \l_stex_symdecl_args_tl {\l_stex_map_args:N \l_stex_return_args:nn}
3850       $\stex_annotation:nn{data-shtml-returntype={}}{
3851         \exp_after:wN \l_stex_symdecl_cs \l_stex_symdecl_args_tl!
3852       }$
3853     }
3854     \clist_if_empty:NF \l_stex_key_argtypes_clist {
3855       \stex_annotation:nn{data-shtml-argtypes={}}{\l_stex_annotation_force_break:n{
3856         \clist_map_inline:Nn \l_stex_key_argtypes_clist {
3857           $\stex_annotation:nn{data-shtml-type={}}{\##1}$
3858         }
3859       }}
3860     }
3861   }\egroup}
3862 }

```

(End of definition for `\stex_symdecl_top:n`. This function is documented on page 128.)

`\stex_symdecl_do:` Requires the above keys and `\l_stex_mroname_str` to be set first

```

3863 \cs_new_protected:Nn \stex_symdecl_do: {
3864   \stex_do_deprecation:n \l_stex_key_name_str

```

```

3865     \_\_stex\_symdecl\_parse\_arity:
3866     \_\_stex\_symdecl\_do\_args:
3867 }
3868
3869 \int_new:N \l_stex_assoc_args_count
3870
3871 \cs_new_protected:Nn \_\_stex\_symdecl\_parse\_arity: {
3872     \int_zero:N \l_stex_get_symbol_arity_int
3873     \int_zero:N \l_stex_assoc_args_count
3874     \str_map_inline:Nn \l_stex_key_args_str {
3875         \str_case:nnF ##1 {
3876             0 { \str_map_break: }
3877             1 { \str_map_break:n{
3878                 \int_set:Nn \l_stex_get_symbol_arity_int {1}
3879                 \str_set:Nn \l_stex_key_args_str {i}
3880             } }
3881             2 { \str_map_break:n{
3882                 \int_set:Nn \l_stex_get_symbol_arity_int {2}
3883                 \str_set:Nn \l_stex_key_args_str {ii}
3884             } }
3885             3 { \str_map_break:n{
3886                 \int_set:Nn \l_stex_get_symbol_arity_int {3}
3887                 \str_set:Nn \l_stex_key_args_str {iiii}
3888             } }
3889             4 { \str_map_break:n{
3890                 \int_set:Nn \l_stex_get_symbol_arity_int {4}
3891                 \str_set:Nn \l_stex_key_args_str {iiiii}
3892             } }
3893             5 { \str_map_break:n{
3894                 \int_set:Nn \l_stex_get_symbol_arity_int {5}
3895                 \str_set:Nn \l_stex_key_args_str {iiiiii}
3896             } }
3897             6 { \str_map_break:n{
3898                 \int_set:Nn \l_stex_get_symbol_arity_int {6}
3899                 \str_set:Nn \l_stex_key_args_str {iiiiiii}
3900             } }
3901             7 { \str_map_break:n{
3902                 \int_set:Nn \l_stex_get_symbol_arity_int {7}
3903                 \str_set:Nn \l_stex_key_args_str {iiiiiiii}
3904             } }
3905             8 { \str_map_break:n{
3906                 \int_set:Nn \l_stex_get_symbol_arity_int {8}
3907                 \str_set:Nn \l_stex_key_args_str {iiiiiiiii}
3908             } }
3909             9 { \str_map_break:n{
3910                 \int_set:Nn \l_stex_get_symbol_arity_int {9}
3911                 \str_set:Nn \l_stex_key_args_str {iiiiiiiiii}
3912             } }
3913             i {\int_incr:N \l_stex_get_symbol_arity_int}
3914             b {\int_incr:N \l_stex_get_symbol_arity_int}
3915             a {\int_incr:N \l_stex_get_symbol_arity_int \int_incr:N \l_stex_assoc_args_count}
3916             B {\int_incr:N \l_stex_get_symbol_arity_int \int_incr:N \l_stex_assoc_args_count}
3917 }{
3918     \msg_error:nnnx{stex}{error/wrongargs}{
```

```

3919     \l_stex_current_module_str ? \l_stex_key_name_str
3920     }{##1}
3921   }
3922 }
3923 }
3924
3925 \cs_new_protected:Nn \__stex_symdecl_do_args: {
3926   \tl_clear:N \l_stex_get_symbol_args_tl
3927   \int_step_inline:nn \l_stex_get_symbol_arity_int {
3928     \tl_put_right:Nn \l_stex_get_symbol_args_tl {##1}
3929     \tl_put_right:Nx \l_stex_get_symbol_args_tl {
3930       \str_item:Nn \l_stex_key_args_str {##1}
3931     }
3932   }
3933 }

```

(End of definition for `\stex_symdecl_do:`. This function is documented on page 127.)

`\_stex_symdecl_check_terms:`

```

3934 \cs_new_protected:Nn \_stex_symdecl_check_terms: {
3935   \stex_check_term:n{
3936     \stex_debug:nn{check_terms}{Checking~type...}
3937     \group_begin:\l_stex_key_type_tl\group_end:
3938     \stex_debug:nn{check_terms}{Checking~definiens...}
3939     \group_begin:\l_stex_key_def_tl\group_end:
3940     \stex_debug:nn{check_terms}{Checking~return...}
3941     \group_begin:\l_stex_key_return_tl!\group_end:
3942     \stex_debug:nn{check_terms}{Checking~argument~types...}
3943     \group_begin:\l_stex_key_argtypes_clist\group_end:
3944   }
3945 }

```

(End of definition for `\_stex_symdecl_check_terms:`. This function is documented on page 128.)

`\textsymdecl`

```

3946
3947 \stex_keys_define:nnnn{textsymdecl}{
3948   \str_clear:N \l_stex_key_name_str
3949   \tl_clear:N \l_stex_key_type_tl
3950   \tl_clear:N \l_stex_key_def_tl
3951 }{
3952   name      .str_set:N  = \l_stex_key_name_str ,
3953   type      .tl_set:N   = \l_stex_key_type_tl ,
3954   def       .tl_set:N   = \l_stex_key_def_tl,
3955   gf        .code:n    = {}
3956 }{style,deprecate}
3957
3958 \stex_new_stylable_cmd:nnnn {textsymdecl} {m 0{} m} {
3959   \stex_keys_set:nn{symdef}{}
3960   \stex_keys_set:nn{textsymdecl}{#2}
3961   \str_set:Nx \l_stex_macroname_str { #1 }
3962   \str_if_empty:NT \l_stex_key_name_str {
3963     \str_set:Nn \l_stex_key_name_str {#1}
3964   }%
3965   % \str_set:Nx \l_stex_key_name_str {\l_stex_key_name_str-sym}

```

```

3966    %}
3967    \str_set:Nn \l_stex_key_role_str {textsymdecl}
3968
3969    \stex_symdecl_do:
3970    \stex_symdecl_check_terms:
3971    \exp_args:Nnx \use:nn {\stex_module_add_symbol:nnnnnnN}{
3972        {\l_stex_mroname_str}
3973        {\l_stex_key_name_str}
3974        {O}{}
3975        {\tl_if_empty:NF \l_stex_key_def_tl{DEFED} }
3976        {}% type
3977        {\use:c{\#1name_nospace}}% return
3978        \stex_invoke_text_symbol:
3979    }
3980    \exp_args:Ne \stex_ref_new_symbol:n
3981        {\l_stex_current_module_str?\l_stex_key_name_str}
3982    \stex_if_do_html:T {
3983        \stex_symdecl_html:
3984    }
3985
3986    \int_set:Nn \l_stex_get_symbol_arity_int 0
3987    \tl_clear:N \l_stex_key_op_tl
3988    \str_clear:N \l_stex_key_intent_str
3989    \str_clear:N \l_stex_key_prec_str
3990    \str_set_eq:NN \l_stex_get_symbol_mod_str \l_stex_current_module_str
3991    \str_set_eq:NN \l_stex_get_symbol_name_str \l_stex_key_name_str
3992    \stex_notation_parse:n{\hbox{\#3}}
3993    \stex_notation_add:
3994    \stex_if_do_html:T {
3995        \def\comp{\_comp}
3996        \stex_notation_do_html:n{\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
3997    }
3998    \stex_execute_in_module:x{
3999        \__stex_symdecl_set_textsymdecl_macro:nnn{\#1}{\l_stex_current_module_str?\l_stex_key_name_str}
4000        \exp_not:n{\#3}
4001    }
4002
4003    \stex_if_smsmode:F{
4004        \group_begin:
4005        \tl_set:Nx \thisdecluri {\l_stex_current_module_str ? \l_stex_key_name_str}
4006        \tl_set_eq:NN \thisdeclname \l_stex_key_name_str
4007        \tl_clear:N \thisstyle
4008        \stex_style_apply:
4009        \group_end:
4010    }
4011    \stex_smsmode_do:
4012 }{}
4013 \stex_deactivate_macro:Nn \textsymdecl {module-environments}
4014 \stex_every_module:n {\stex_reactivate_macro:N \textsymdecl}
4015 \stex_sms_allow_escape:N \textsymdecl
4016
4017 \cs_new_protected:Nn \__stex_symdecl_set_textsymdecl_macro:nnn {
4018     \cs_set_protected:cpn{\#1name_nospace}{\#3}
4019     \cs_set_protected:cpn{\#1name}{\#3}

```

```

4020   \mode_if_vertical:T{\hbox_unpack:N\c_empty_box}
4021   \mode_if_math:T\hbox{\let\xspace\relax #3}
4022   \mode_if_math:F{\cs_if_exist:NT\xspace\xspace}
4023 }
4024 }
4025
4026 \cs_new_protected:Nn \stex_invoke_text_symbol: {
4027   \mode_if_vertical:T{\hbox_unpack:N\c_empty_box}
4028   \stex_term_oms_or_omv:nnn{}{}{\maincomp{\let\xspace\relax\l_stex_current_return_tl}}
4029   \group_end:\mode_if_math:F{\cs_if_exist:NT\xspace\xspace}
4030 }

```

(End of definition for `\textsymdecl`. This function is documented on page 91.)

```

\stex_get_symbol:n
4031 \cs_new_protected:Nn \stex_get_symbol:n {
4032   \stex_get_symbol:n{ #1 }
4033   \str_if_empty:NT \l_stex_get_symbol_name_str {
4034     \msg_error:nnn{\stex}{error/unknownsymbol}{#1}
4035   }
4036 }
4037
4038 \cs_new_protected:Nn \stex_get_symbol:n {
4039   \str_clear:N \l_stex_get_symbol_mod_str
4040   \str_clear:N \l_stex_get_symbol_name_str
4041   \cs_if_exist:cTF { #1 }{
4042     \cs_set_eq:Nc \l_stex_symdecl_cs { #1 }
4043     % command name
4044     \exp_args:Ne \tl_if_empty:nTF { \cs_argument_spec:N \l_stex_symdecl_cs }{
4045       % ...that takes no arguments
4046       \exp_args:Ne \cs_if_eq:NNTF {\tl_head:N \l_stex_symdecl_cs}
4047         \stex_invoke_symbol:nnnnnnnn
4048         \stex_symdecl_get_symbol_from_cs:
4049         \stex_symdecl_get_symbol_from_string:n { #1 }}
4050     }{
4051       \stex_symdecl_get_symbol_from_string:n { #1 }
4052     }
4053   }{
4054     \stex_symdecl_get_symbol_from_string:n { #1 }
4055   }
4056 }
4057
4058 \int_new:N \l_stex_get_symbol_arity_int
4059 \cs_new_protected:Nn \stex_symdecl_get_symbol_from_cs: {
4060   \stex_debug:nn{symbols}{Getting~from~cs...}
4061   \stex_pseudogroup_with:nn{\stex_invoke_symbol:nnnnnnnN}{
4062     \cs_set:Npn \stex_invoke_symbol:nnnnnnnN ##1 ##2 ##3 ##4 ##5 ##6 ##7 ##8 {
4063       \str_set:Nn \l_stex_get_symbol_mod_str {##1}
4064       \str_set:Nn \l_stex_get_symbol_name_str {##2}
4065       \int_set:Nn \l_stex_get_symbol_arity_int {##3}
4066       \tl_set:Nn \l_stex_get_symbol_args_tl {##4}
4067       \tl_set:Nn \l_stex_get_symbol_def_tl {##5}
4068       \tl_set:Nn \l_stex_get_symbol_type_tl {##6}
4069       \tl_set:Nn \l_stex_get_symbol_return_tl {##7}

```

```

4070     \tl_set:Nn \l_stex_get_symbol_invoke_cs {##8}
4071   }
4072   \l_stex_symdecl_cs
4073 }
4074 }
4075
4076 \cs_new_protected:Nn \__stex_symdecl_get_symbol_from_string:n {
4077   \stex_debug:nn{symbols}{Getting~from~string~#1...}
4078   \seq_set_split:Nnn \l_stex_symdecl_seq ? {#1}
4079   \seq_pop_right:NN \l_stex_symdecl_seq \l_stex_symdecl_name
4080   \seq_if_empty:NTF \l_stex_symdecl_seq {
4081     \exp_args:No \__stex_symdecl_get_from_one_string:n {#1}
4082   }{
4083     \exp_args:NNe \exp_args:Nno \__stex_symdecl_get_symbol_from_modules:nn {
4084       \seq_use:Nn \l_stex_symdecl_seq ?
4085     } \l_stex_symdecl_name
4086   }
4087 }
4088
4089 \cs_new_protected:Nn \__stex_symdecl_sym_from_str_i:nnnn {
4090   \bool_lazy_any:nTF{
4091     {\str_if_eq_p:nn{#2}{#3}}
4092     {\str_if_eq_p:nn{#2}{#4}}
4093     {\str_if_ends_with_p:nn{#4}{/#2}}
4094   }{
4095     \__stex_symdecl_sym_i_finish:nnnnnnN{#1}{#4}
4096   }{
4097     \__stex_symdecl_sym_i_gobble:nnnnn
4098   }
4099 }
4100 \cs_new_protected:Nn \__stex_symdecl_sym_i_gobble:nnnnn {}
4101
4102 \cs_new_protected:Nn \__stex_symdecl_sym_i_finish:nnnnnnN {
4103   \prop_map_break:n{\seq_map_break:n{
4104     \str_set:Nn \l_stex_get_symbol_mod_str {#1}
4105     \str_set:Nn \l_stex_get_symbol_name_str {#2}
4106     \int_set:Nn \l_stex_get_symbol_arity_int {#3}
4107     \tl_set:Nn \l_stex_get_symbol_args_tl {#4}
4108     \tl_set:Nn \l_stex_get_symbol_def_tl {#5}
4109     \tl_set:Nn \l_stex_get_symbol_type_tl {#6}
4110     \tl_set:Nn \l_stex_get_symbol_return_tl {#7}
4111     \tl_set:Nn \l_stex_get_symbol_invoke_cs {#8}
4112   }}
4113 }
4114
4115 \cs_new_protected:Nn \__stex_symdecl_get_symbol_from_modules:nn {
4116   \stex_debug:nn{symbols}{Getting~#2-in~#1...}
4117   \seq_map_inline:Nn \l_stex_all_modules_seq {
4118     \str_if_ends_with:nnT{##1}{#1} {
4119       \prop_map_inline:cN{c_stex_module_##1_symbols_prop} {
4120         \__stex_symdecl_sym_from_str_i:nnnn{##1}{#2} ####2
4121       }
4122     }
4123 }

```

```

4124 }
4125
4126 \cs_new_protected:Nn \__stex_symdecl_get_from_one_string:n {
4127   \stex_debug:nn{symbols}{Getting~#1~anywhere...}
4128   \stex_iterate_symbols:n{
4129     \%stex_debug:nn{symbols}{>#1==##2~|~#1==##3<...}
4130     \bool_lazy_any:nT{
4131       {\str_if_eq_p:nn{#1}{##2}}
4132       {\str_if_eq_p:nn{#1}{##3}}
4133       {\stex_str_if_ends_with_p:nn{##3}{/#1}}
4134     }{
4135       \stex_iterate_break:n{
4136         \str_set:Nn \l_stex_get_symbol_mod_str {##1}
4137         \str_set:Nn \l_stex_get_symbol_name_str {##3}
4138         \int_set:Nn \l_stex_get_symbol_arity_int {##4}
4139         \tl_set:Nn \l_stex_get_symbol_args_tl {##5}
4140         \tl_set:Nn \l_stex_get_symbol_def_tl {##6}
4141         \tl_set:Nn \l_stex_get_symbol_type_tl {##7}
4142         \tl_set:Nn \l_stex_get_symbol_return_tl {##8}
4143         \tl_set:Nn \l_stex_get_symbol_invoke_cs {##9}
4144       }
4145     }
4146   }
4147 }

```

(End of definition for `\stex_get_symbol:n`. This function is documented on page 127.)

### 13.8.2 Notations

```

4148 <@=stex_notations>
\__stex_map_args:N
\__stex_map_notation_args:N
4149 \cs_new:Nn \__stex_map_args:N {
4150   \tl_if_empty:NF \l_stex_get_symbol_args_tl {
4151     \exp_after:wN \__stex_notations_map_args_i:w \exp_after:wN
4152     #1 \l_stex_get_symbol_args_tl \__stex_notations_args_end:
4153   }
4154 }
4155 \cs_new:Npn \__stex_notations_map_args_i:w #1 #2 #3 #4 \__stex_notations_args_end: {
4156   #1 #2 #3
4157   \tl_if_empty:nF{#4}{
4158     \__stex_notations_map_args_i:w #1 #4 \__stex_notations_args_end:
4159   }
4160 }
4161
4162 \cs_new:Nn \__stex_map_notation_args:N {
4163   \tl_if_empty:NF \l_stex_notation_args_tl {
4164     \exp_after:wN \__stex_notations_map_args_ii:w \exp_after:wN
4165     #1 \l_stex_get_symbol_args_tl \__stex_notations_args_end:
4166   }
4167 }
4168 \cs_new:Npn \__stex_notations_map_args_ii:w #1 #2 #3 #4 #5 #6 \__stex_notations_args_end: {
4169   #1 #2 #3 #4 #5
4170   \tl_if_empty:nF{#6}{
4171     \__stex_notations_map_args_ii:w #1 #6 \__stex_notations_args_end:

```

```

4172     }
4173 }

(End of definition for \_stex_map_args:N and \_stex_map_notation_args:N. These functions are documented on page ??.)

notation arguments:
4174 \stex_keys_define:nnnn{notation}{
4175   \str_clear:N \l_stex_key_variant_str
4176   \str_clear:N \l_stex_key_prec_str
4177   \str_clear:N \l_stex_key_op_tl
4178   \str_clear:N \l_stex_key_intent_str
4179   \clist_clear:N \l_stex_key_intent_args_clist
4180 }{
4181   variant    .str_set_x:N = \l_stex_key_variant_str ,
4182   prec       .str_set_x:N = \l_stex_key_prec_str ,
4183   op         .tl_set:N    = \l_stex_key_op_tl ,
4184   intent      .str_set:N = \l_stex_key_intent_str ,
4185   argnames    .clist_set:N = \l_stex_key_intent_args_clist ,
4186   unknown     .code:n    = {
4187     \str_if_empty:NTF \l_keys_key_str {
4188       \str_set:Nx \l_stex_key_variant_str {\l_keys_key_tl}
4189     }{
4190       \str_set_eq:NN \l_stex_key_variant_str \l_keys_key_str
4191     }
4192   }
4193 }{style}

\notation
4194 \stex_new_stylable_cmd:nnnn {notation} { s m 0{} m} {
4195   \stex_keys_set:nn{notation}{#3}
4196   \stex_get_symbol:n{#2}
4197   \stex_notation_parse:n{#4}
4198   \stex_if_check_terms:T{ \_stex_notation_check: }
4199   \_stex_notation_add:
4200   \stex_if_do_html:T {
4201     \def\comp{\_comp}
4202     \_stex_notation_do_html:n{\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
4203   }
4204   \IfBooleanTF#1{
4205     \_stex_notation_set_default:n{
4206       \l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str
4207     }
4208   }{}
4209   \stex_if_smsmode:F{
4210     \group_begin:
4211     \__stex_notations_styledefs:
4212     \stex_style_apply:
4213     \group_end:
4214   }
4215   \stex_smsmode_do:
4216 }{}}

4217
4218 \cs_new_protected:Nn \__stex_notations_styledefs: {
4219   \str_set_eq:NN \thisnotationvariant \l_stex_key_variant_str

```

```

4220 \str_set:Nn \thisdeclname \l_stex_get_symbol_name_str
4221 \tl_set:Nx \thisdecluri {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
4222 \def\thisnotation{
4223   $
4224   \tl_set_eq:NN \l_stex_current_symbol_str\thisdecluri
4225   \exp_args:Nne \use:nn{\l_stex_notation_macrocode_cs} {
4226     \l_stex_notation_make_args:
4227   }$
4228 }
4229 }

4230 \stex_deactivate_macro:Nn \notation {module~environments}
4231 \stex_every_module:n {\stex_reactivate_macro:N \notation}
4232 \stex_sms_allow_escape:N \notation

```

(End of definition for \notation. This function is documented on page 93.)

\stex\_notation\_parse:n requires the above keys, \l\_stex\_get\_symbol\_arity\_int, and \l\_stex\_get\_symbol\_args\_tl

```

4234 \cs_new_protected:Nn \stex_notation_parse:n {
4235   \tl_if_empty:NF \l_stex_key_op_tl {
4236     \tl_set:Nx \l_stex_key_op_tl { \exp_not:N\maincomp {
4237       \exp_args:No \exp_not:n \l_stex_key_op_tl
4238     } }
4239   }
4240   \seq_clear:N \l__stex_notations_precs_seq
4241   \tl_clear:N \l_stex_notation_args_tl
4242   \int_compare:nNnTF \l_stex_get_symbol_arity_int = 0 {
4243     \__stex_notations_const_precs:
4244     \tl_if_empty:NT \l_stex_key_op_tl {
4245       \tl_set:Nn \l_stex_key_op_tl { \maincomp{#1} }
4246     }
4247   }{
4248     \__stex_notations_fun_precs:
4249     \str_set:Nn \l__stex_notations_missing_str {#1}
4250     \tl_clear:N \l__stex_notations_missing_tl
4251     \l_stex_map_args:N \__stex_notations_add_missing_args:nn
4252     \tl_if_empty:NT \l_stex_key_op_tl {
4253       \hbox_set:Nn \l_tmpa_box {
4254         \str_set:Nn \l_stex_current_symbol_str {}
4255         \cs_set:Npn \l_tmpa cs ##1 ##2 ##3 ##4 ##5 ##6 ##7 ##8 ##9 { #1 }
4256         \cs_set:Npn \maincomp ##1 {
4257           \tl_gset:Nn \l_stex_key_op_tl { \maincomp{##1} }
4258           ##1
4259         }
4260         \cs_set:Npn \argsep ##1 ##2 {##1 ##2}
4261         \cs_set:Npn \argmap ##1 ##2 ##3 {##1 ##3}
4262         \cs_set:Npn \argarraymap ##1 ##2 ##3 ##4 {
4263           ##1 ##2
4264         }
4265         \stex_suppress_html:n{$\l_tmpa cs abcdefghj$}
4266       }
4267     }
4268   }

```

```

4269 \exp_args:NNe
4270 \tl_set:Nn \l_stex_notation_macrocode_cs {
4271   \STEXInternalNotation
4272   { \l_stex_key_variant_str }
4273   { \l_stex_notations_oppref_tl }
4274   { \l_stex_key_intent_str }
4275   { \l_stex_notation_args_tl }
4276   {
4277     \int_compare:nNnTF \l_stex_get_symbol_arity_int = 0
4278     { \exp_not:n { \maincomp{ #1 } } }
4279     { \exp_not:n { #1 } \l_stex_notations_missing_tl }
4280   }
4281 }
4282 \stex_debug:nn{notation}{Notation:\meaning\l_stex_notation_macrocode_cs}
4283 }
4284
4285 \cs_new_protected:Nn \__stex_notations_const_precs: {
4286   \str_if_empty:NTF \l_stex_key_prec_str {
4287     \tl_set:No \l_stex_notations_oppref_tl { \neginfpref }
4288   }{
4289     \str_if_eq:onTF \l_stex_key_prec_str {nobrackets} {
4290       \tl_set:No \l_stex_notations_oppref_tl { \neginfpref }
4291     }{
4292       \tl_set_eq:NN \l_stex_notations_oppref_tl \l_stex_key_prec_str
4293     }
4294   }
4295 }
4296
4297 \cs_new_protected:Nn \__stex_notations_fun_precs: {
4298   \str_if_empty:NTF \l_stex_key_prec_str {
4299     \tl_set:No \l_stex_notations_oppref_tl { \neginfpref }
4300   }{
4301     \str_if_eq:onTF \l_stex_key_prec_str {nobrackets} {
4302       \tl_set:No \l_stex_notations_oppref_tl { \neginfpref }
4303     }{
4304       \tl_set_eq:NN \l_stex_notations_oppref_tl \l_stex_key_prec_str
4305     }
4306   }
4307 \str_if_empty:NTF \l_stex_key_prec_str {
4308   \tl_set:Nn \l_stex_notations_oppref_tl { 0 }
4309   \int_step_inline:nn \l_stex_get_symbol_arity_int {
4310     \seq_put_right:Nn \l_stex_notations_precs_seq {0}
4311   }
4312 }{
4313   \str_if_eq:onTF \l_stex_key_prec_str {nobrackets} {
4314     \stex_debug:nn{notation}{No~brackets}
4315     \tl_set:No \l_stex_notations_oppref_tl { \neginfpref }
4316     \int_step_inline:nn \l_stex_get_symbol_arity_int {
4317       \exp_args:NNo \seq_put_right:Nn \l_stex_notations_precs_seq \infpref
4318     }
4319     } \__stex_notations_parse_precs:
4320 }
4321 \__stex_notations_do_argnames:
4322 }

```

```

4323 \cs_new_protected:Nn \__stex_notations_parse_precs: {
4324   \stex_debug:nn{notation}{parsing-precedence~\l_stex_key_prec_str}
4325   \seq_set_split:NnV \l__stex_notations_seq ; \l_stex_key_prec_str
4326   \seq_pop_left:NNTF \l__stex_notations_seq \l__stex_notations_str {
4327     \tl_set_eq:NN \l__stex_notations_oppref_tl \l__stex_notations_str
4328     \seq_pop_left:NNT \l__stex_notations_seq \l__stex_notations_str {
4329       \exp_args:NNo \seq_set_split:NnV \l__stex_notations_seq
4330       {\tl_to_str:n{x}} \l__stex_notations_str
4331     }
4332   }
4333   }{
4334     \tl_set:Nno \l__stex_notations_oppref_tl { 0 }
4335   }
4336 \int_step_inline:nn \l_stex_get_symbol_arity_int {
4337   \seq_pop_left:NNTF \l__stex_notations_seq \l__stex_notations_str {
4338     \seq_put_right:Nno \l__stex_notations_precs_seq \l__stex_notations_str
4339   }{
4340     \seq_put_right:Nno \l__stex_notations_precs_seq \l__stex_notations_oppref_tl
4341   }
4342 }
4343 }
4344
4345 \cs_new_protected:Nn \__stex_notations_do_argnames: {
4346   \tl_clear:N \l_stex_notation_args_tl
4347   \stex_map_args:N \__stex_notations_do_argname:nn
4348 }
4349
4350 \cs_new_protected:Nn \__stex_notations_do_argname:nn {
4351   \clist_if_empty:NTF \l_stex_key_intent_args_clist {
4352     \tl_put_right:Nx \l_stex_notation_args_tl {
4353       #1#2{\seq_item:Nn \l__stex_notations_precs_seq #1} {
4354         \str_if_empty:NF \l_stex_key_intent_str {#1}
4355       }
4356     }
4357   }{
4358     \tl_put_right:Nx \l_stex_notation_args_tl {
4359       #1#2{\seq_item:Nn \l__stex_notations_precs_seq #1}
4360       {\c_dollar_str\clist_item:Nn \l_stex_key_intent_args_clist 1}
4361     }
4362     \clist_pop:NN \l_stex_key_intent_args_clist \l_tmpa_tl
4363   }
4364 }
4365
4366 \cs_new:Nn \__stex_notations_add_missing_args:nn {
4367   % TODO this skips arguments if e.g. ##1 occurs rather than #1!!
4368   \exp_args:NNe \str_if_in:NnF \l__stex_notations_missing_str {\c_hash_str\c_hash_str#1} {
4369     \tl_put_right:Nn \l__stex_notations_missing_tl{\STEXinvis{## #1}}
4370   }
4371 }

```

(End of definition for `\stex_notation_parse:n`. This function is documented on page ??.)

```

\__stex_notation_check:
\__stex_notation_add:
\__stex_notation_do_html:n
\__stex_notation_make_args:
4372 \cs_new_protected:Nn \__stex_notation_check: {

```

```

4373 \stex_check_term:n{
4374   \str_set:Nn \l_stex_current_symbol_str {test}
4375   \cs_set:Npn \comp ##1 {##1}
4376   \stex_debug:nn{check_terms}{Checking~notation...}
4377   \exp_args:Nne \use:nn{\l_stex_notation_macrocode_cs}{}{
4378     \l_stex_notation_make_args:
4379   }
4380 }
4381 }
4382 }
4383 \cs_new_protected:Nn \l_stex_notation_add: {
4384   \stex_module_add_notation:oeoof{
4385     \l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str
4386   } \l_stex_key_variant_str
4387   {\int_use:N \l_stex_get_symbol_arity_int}
4388   \l_stex_notation_macrocode_cs
4389   \l_stex_key_op_tl
4390 }
4391
4392 \cs_new_protected:Nn \l_stex_notation_do_html:n {
4393   \hbox{\l_stex_annotation_invisible:nn {
4394     data-shtml-notation={#1},
4395     data-shtml-notationfragment={\l_stex_key_variant_str},
4396     data-shtml-precedence={\l_stex_notations_opprec_t1},
4397     data-shtml-argprecs={\seq_use:Nn \l_stex_notations_precs_seq ,}
4398   }{
4399     \cs_set_protected:Npn \argsep ##1 ##2 {
4400       \l_stex_annotation:nn{data-shtml-argsep={}}{
4401         ##1 ##2
4402       }
4403     }
4404     \cs_set_protected:Npn \argmap ##1 ##2 ##3 {
4405       \cs_set:Npn \l_stex_notations_map_cs: #####1 { ##2 }
4406       \l_stex_annotation:nn{data-shtml-argmap={}}{
4407         \l_stex_notations_map_cs:{##1} \l_stex_annotation:nn{data-shtml-argmap-sep={}}{##3}
4408       }
4409     }
4410     \cs_set_protected:Npn \maincomp {
4411       \do_comp:nNn {maincomp}\compemph@uri
4412     }
4413   $ 
4414   \str_set:Nx \l_stex_current_symbol_str {#1}
4415   \l_stex_annotation:nn{data-shtml-notationcomp={}}{
4416     \exp_args:Nne \use:nn {
4417       \l_stex_notation_macrocode_cs {}
4418     }{
4419       \l_stex_map_args:N \l_stex_notations_make_arg_html:nn
4420     }
4421   }
4422   $ 
4423   \tl_if_empty:NF \l_stex_key_op_tl {
4424     $
4425     \str_set:Nx \l_stex_current_symbol_str {#1}
4426     \l_stex_annotation:nn{data-shtml-notationopcomp={}}{

```

```

4427          \_stex_term_oms:nnn{\l_stex_key_variant_str}{}{\l_stex_key_op_t}
4428      }
4429      $
4430  }
4431 }
4432 }
4433
4434 \cs_new:Nn \__stex_notations_make_arg_html:nn {
4435 %   \str_case:nnF #2 {
4436 %     a {{%
4437 %       \stex_annotation:nn{data-shtml-argnum=#1a}{x},
4438 %       \stex_annotation:nn{data-shtml-argnum=#1b}{x}
4439 %     }%
4440 %     B {{%
4441 %       \stex_annotation:nn{data-shtml-argnum=#1a}{x},
4442 %       \stex_annotation:nn{data-shtml-argnum=#1b}{x}
4443 %     }%
4444 %   }%
4445 %   {%
4446 %     \stex_annotation:nn{data-shtml-argnum=#1}{x}
4447 %   }%
4448 % }
4449 }
450
451 \cs_new:Nn \_stex_notation_make_args: {
452   \stex_map_notation_args:N \__stex_notations_make_arg:nnnn
453 }
454
455
456 \cs_new:Nn \__stex_notations_make_arg:nnnn {
457   \str_case:nnF #2 {
458     a {{%
459       a\c_math_subscript_token{#1,1},
460       a\c_math_subscript_token{#1,2}
461     }%
462     B {{%
463       B\c_math_subscript_token{#1,1},
464       B\c_math_subscript_token{#1,2}
465     }%
466   }%
467   \_stex_term_arg:nnnn{#1}{#2}{#3}{#4}
468   {{#2}\c_math_subscript_token{#1}}
469 }
470 }

```

(End of definition for `\_stex_notation_check:` and others. These functions are documented on page ??.)

```

\setnotation
\_stex_notation_set_default:n
4471 \cs_new_protected:Npn \setnotation #1 #2 {
4472   \stex_get_symbol:n{#1}
4473   \cs_if_exist:cTF{\l_stex_notation_
4474     \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4475     _#2_cs

```

```

4476 }{
4477   \tl_set_eq:Nc \l_stex_notation_macrocode cs {l_stex_notation_
4478     \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4479     _#2_cs
4480   }
4481   \cs_if_exist:cTF{l_stex_notation_
4482     \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4483     _op_#2_cs
4484   }{
4485     \tl_set_eq:Nc \l_stex_key_op_tl {l_stex_notation_
4486       \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4487       _op_#2_cs
4488     }
4489   }{
4490     \tl_clear:N \l_stex_key_op_tl
4491   }
4492   \_stex_notation_set_default:n{
4493     \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4494   }
4495 }{
4496   \msg_error:nnxx{stex}{unknownnotation}{#2}{
4497     \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4498   }
4499 }
4500 }
4501
4502 \cs_new_protected:Nn \_stex_notation_set_default:n{
4503   \stex_if_in_module:TF{
4504     \stex_module_add_notation:oeooo{#1}{}{%
4505       \int_use:N \l_stex_get_symbol_arity_int}
4506     \l_stex_notation_macrocode cs
4507     \l_stex_key_op_tl
4508   }{
4509     \cs_set_eq:cN {l_stex_notation_
4510       \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4511       _cs}\l_stex_notation_macrocode cs
4512     \tl_if_empty:NF \l_stex_key_op_tl {
4513       \cs_set_eq:cN{l_stex_notation_
4514         \l_stex_get_symbol_mod_str?\l_stex_get_symbol_name_str
4515         _op__cs}\l_stex_key_op_tl
4516     }
4517   }
4518 }

```

(End of definition for `\setnotation` and `\_stex_notation_set_default:n`. These functions are documented on page 94.)

### `\varnotation`

```

4519 \stex_new_stylable_cmd:nnnn {varnotation} { s m 0{} m} {
4520   \stex_keys_set:nn{notation}{#3}
4521   \stex_get_var:n{#2}
4522   \str_set_eq:NN \l_stex_key_name_str \l_stex_get_symbol_name_str
4523   \stex_notation_parse:n{#4}
4524   \stex_if_check_terms:T{ \_stex_notation_check: }

```

```

4525   \_stex_vardecl_notation_macro:
4526   \IfBooleanTF#1{
4527     \_stex_notation_set_default:n{\l_stex_get_symbol_name_str}
4528   }{}
4529   \group_begin:
4530   \tl_set_eq:NN \thisvarname \l_stex_get_symbol_name_str
4531   \tl_clear:N \thisstyle
4532   \str_set_eq:NN \thisnotationvariant \l_stex_key_variant_str
4533   \def\thisnotation{
4534     $ \let\l_stex_current_symbol_str \thisvarname
4535     \def\comp{\_varcomp}\exp_args:Nne \use:nn{\l_stex_notation_macrocode_cs}{}
4536     \_stex_notation_make_args:
4537   }$}
4538 }
4539 \stex_style_apply:
4540 \group_end:
4541 }{}
```

(End of definition for `\varnotation`. This function is documented on page 98.)

```

\symdef
4542 \stex_keys_define:nnnn{symdef}{}{}{decl,notation}
4543
4544 \cs_new_protected:Nn \stex_symdef_styledefs: {
4545   \tl_set:Nx \thisdecluri {\l_stex_current_module_str ? \l_stex_key_name_str}
4546   \tl_set_eq:NN \thisdeclname \l_stex_key_name_str
4547   \tl_set_eq:NN \thistype \l_stex_key_type_tl
4548   \tl_set_eq:NN \thisdefiniens \l_stex_key_def_tl
4549   \tl_set_eq:NN \thisargs \l_stex_key_args_str
4550   \tl_clear:N \thisstyle
4551   \str_set_eq:NN \thisnotationvariant \l_stex_key_variant_str
4552   \def\thisnotation{
4553     $ \let\l_stex_current_symbol_str \thisdecluri
4554     \def\comp{\_comp}\exp_args:Nne \use:nn{\l_stex_notation_macrocode_cs}{}
4555     \_stex_notation_make_args:
4556   }$}
4557 }
4558 }
4559
4560 \stex_new_stylable_cmd:nnnn {symdef} { m 0{} m} {
4561   \stex_keys_set:nn{symdef}{#2}
4562   \str_set:Nx \l_stex_mroname_str { #1 }
4563   \stex_symdecl_top:n{#1}
4564   \stex_debug:nn{symdef}{Doing~\l_stex_current_module_str ? \l_stex_key_name_str}
4565   \str_set_eq:NN \l_stex_get_symbol_mod_str \l_stex_current_module_str
4566   \str_set_eq:NN \l_stex_get_symbol_name_str \l_stex_key_name_str
4567   \stex_notation_parse:n{#3}
4568   \stex_debug:nn{Here!}{\meaning\l_stex_notation_args_tl}
4569   \stex_notation_check:
4570   \stex_notation_add:
4571   \stex_if_do_html:T{
4572     \_stex_notation_do_html:n{\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
4573   }
4574 \stex_if_smsmode:F{
```

```

4575   \group_begin:
4576   \_stex_symdef_styledefs:
4577   \stex_style_apply:
4578   \group_end:
4579 }
4580 \stex_smsmode_do:
4581 }{}
4582
4583 \stex_deactivate_macro:Nn \symdef {module~environments}
4584 \stex_every_module:n {\stex_reactivate_macro:N \symdef}
4585 \stex_sms_allow_escape:N \symdef

```

(End of definition for `\symdef`. This function is documented on page 91.)

`\stex_do_default_notation_op:`

```

4586 \cs_new_protected:Nn \stex_do_default_notation: {
4587   \stex_do_default_notation_op:
4588   \tl_if_empty:NTF \l_stex_current_args_tl {
4589     \tl_clear:N \l__stex_notations_args_tl
4590   }{
4591     \__stex_notations_make_name:
4592     \tl_set_eq:NN \l_stex_get_symbol_args_tl \l_stex_current_args_tl
4593     \tl_set:Nx \l__stex_notations_args_tl {
4594       \stex_map_args:N \__stex_notations_augment_arg:nn
4595     }
4596     \tl_put_right:Nn \l_stex_default_notation {\comp{}}
4597     \seq_clear:N \l_tmpa_seq
4598     \int_step_inline:nn \l_stex_current_arity_str {
4599       \seq_put_right:Nn \l_tmpa_seq {#### ##1}
4600     }
4601     \tl_put_right:Nx \l_stex_default_notation {
4602       \seq_use:Nn \l_tmpa_seq {\mathpunct{\comp{,}}}}
4603     }
4604     \tl_put_right:Nn \l_stex_default_notation {\comp{}}
4605   }
4606   \tl_set:Nx \l_stex_default_notation {\STEXInternalNotation{}{0}{}{\l__stex_notations_args_}
4607   \exp_args:No \exp_not:n \l_stex_default_notation
4608 }
4609 }
4610
4611 \cs_new:Nn \__stex_notations_augment_arg:nn {
4612   #1#2{0}{}
4613 }
4614
4615 \cs_new_protected:Nn \__stex_notations_make_name: {
4616   \exp_args:NNNo \seq_set_split:Nnn \l_tmpa_seq ? \l_stex_current_symbol_str
4617   \seq_pop_right:NN \l_tmpa_seq \l__stex_notations_name_str
4618   \exp_args:NNNo \seq_set_split:Nnn \l_tmpa_seq / \l__stex_notations_name_str
4619   \seq_pop_right:NN \l_tmpa_seq \l__stex_notations_name_str
4620 }
4621
4622 \cs_new_protected:Nn \stex_do_default_notation_op: {
4623   \__stex_notations_make_name:
4624   \tl_set:Nx \l_stex_default_notation {\exp_not:N \maincomp{ \exp_not:N \mathrm {\l__stex_no
4625 }
```

(End of definition for \stex\_do\_default\_notation\_op:. This function is documented on page ??.)

```
\STEXInternalNotation
4626 % 1: variant 2: operator precedence 3: intent 4: arguments 5: code 6: next
4627
4628 \cs_new_protected:Npn \STEXInternalNotation #1 #2 #3 #4 #5 #6 {
4629   \__stex_notations_process_notation:nnnnnn{#1}{#2}{#3}{#4}{#5}{#
4630     \l__stex_notations_code_tl
4631     #6
4632   }
4633 }
4634
4635 \cs_new_protected:Npn \__stex_notations_process_notation:nnnnnn #1 #2 #3 #4 {
4636   \tl_if_empty:nTF{#4} {
4637     \__stex_notations_simple:nnnnn{#1}{#2}{#3}
4638   }{
4639     \__stex_notations_complex:nnnnnn{#1}{#2}{#3}{#4}
4640   }
4641 }
4642
4643 \cs_new_protected:Nn \__stex_notations_simple:nnnnn {
4644   \stex_debug:nn{Notation~code}{\tl_to_str:n{#4}}
4645   \tl_set:Nn \l__stex_notations_code_tl {
4646     \cs_set:Npn \l__stex_notations_cs {
4647       \stex_maybe_brackets:nn{#2}{#
4648         \stex_term_oms_or_omv:nnn{#1}{#3}{#4}
4649       }
4650     }
4651     \l__stex_notations_cs
4652   }
4653   \stex_debug:nn{Here:Notation}{\meaning \l__stex_notations_code_tl }
4654   #5
4655 }
4656
4657 \cs_new_protected:Nn \__stex_notations_complex:nnnnnn {
4658   \stex_debug:nn{Notation~code}{\tl_to_str:n{#5}}
4659   \int_zero:N \l_tmpa_int
4660   \tl_set:Nn \l__stex_notations_pre_tl f\cs_set_eq:NN \stex_term_oma_or_omb:nnn \stex_term_
4661   \tl_set:Nn \l__stex_notations_code_tl {
4662     \cs_generate_from_arg_count:NNnn \l__stex_notations_cs \cs_set:Npn \l_tmpa_int
4663   {
4664     \stex_maybe_brackets:nn{#2}{#
4665       \stex_term_oma_or_omb:nnn{#1}{#3}{#
4666         \bool_set_false:N \l_stex_brackets_dones_bool
4667         #5
4668       }
4669     }
4670   }
4671   \l__stex_notations_cs
4672 }
4673 \tl_set:Nn \l__stex_notations_after_tl{
4674   \exp_args:NNo
4675   \tl_put_left:Nn \l__stex_notations_code_tl \l__stex_notations_pre_tl
4676   \tl_put_left:Nx \l__stex_notations_code_tl {
```

```

4677     \int_set:Nn \l_tmpa_int {\int_use:N \l_tmpa_int}
4678 }
4679 \stex_debug:nn{Here:Notation}{\meaning \l__stex_notations_code_tl }
4680 #6
4681 }
4682 \__stex_notations_parse_notation_args:nnnw #4 \__stex_notations_args_end:
4683 }
4684
4685 \cs_new_protected:Npn \__stex_notations_parse_notation_args:nnnw #1 #2 #3 #4 #5 \__stex_not
4686   \tl_if_empty:nTF{#5} {
4687     \__stex_notations_add_last:nnnn{#1}{#2}{#3}{#4}
4688   }{
4689     \__stex_notations_add_next:nnnnn{#1}{#2}{#3}{#4}{#5}
4690   }
4691 }
4692
4693 \cs_new_protected:Nn \__stex_notations_add_next:nnnnn {
4694   \__stex_notations_add:nnnn{#1}{#2}{#3}{#4}{#6}
4695   \__stex_notations_parse_notation_args:nnnw #5 \__stex_notations_args_end:
4696 }
4697
4698 \cs_new_protected:Nn \__stex_notations_add_last:nnnnn {
4699   \__stex_notations_add:nnnnn{#1}{#2}{#3}{#4}{#5}
4700   \l__stex_notations_after_tl
4701 }
4702
4703 \cs_new_protected:Nn \__stex_notations_add:nnnnn {
4704   \int_incr:N \l_tmpa_int
4705   \str_case:nn{#2} {
4706     i {
4707       \tl_put_right:Nn \l__stex_notations_code_tl {
4708         {\__stex_term_arg:nnnn{#1}{#2}{#3}{#4}{#5}}
4709     }
4710   }
4711   b {
4712     \tl_set:Nn \l__stex_notations_pre_tl {
4713       \cs_set_eq:NN \stex_term_oma_or_omb:nnn \stex_term_omb:nnn
4714     }
4715     \tl_put_right:Nn \l__stex_notations_code_tl {
4716       {\__stex_term_arg:nnnnn{#1}{#2}{#3}{#4}{#5}}
4717     }
4718   }
4719   a {
4720     \tl_put_right:Nn \l__stex_notations_code_tl {
4721       {\__stex_term_arg_aB:nnnnn{#1}{#2}{#3}{#4}{#5}}
4722     }
4723   }
4724   B {
4725     \tl_set:Nn \l__stex_notations_pre_tl {
4726       \cs_set_eq:NN \stex_term_oma_or_omb:nnn \stex_term_omb:nnn
4727     }
4728     \tl_put_right:Nn \l__stex_notations_code_tl {
4729       {\__stex_term_arg_aB:nnnnn{#1}{#2}{#3}{#4}{#5}}
4730     }

```

```

4731     }
4732   }
4733 }
```

(End of definition for \STEXInternalNotation. This function is documented on page ??.)

### a/B-mode argument handling

\argsep

```

4734 \cs_new_protected:Nn \__stex_notations_check_aB_arg:Nn {
4735   \exp_args:Ne \cs_if_eq:NNF {\tl_head:n{#2}}
4736     \__stex_term_arg_aB:nnnnn {
4737       \msg_error:nnx{\stex}{error/assocarg}{\tl_to_str:n{#1}}
4738     }
4739   }
4740 
4741 \cs_new_protected:Npn \argsep #1 #2 {
4742   \__stex_notations_check_aB_arg:Nn\argsep{#1}
4743   \stex_pseudogroup_with:nn{\__stex_term_do_aB_clist:}{
4744     \tl_set:Nn \__stex_term_do_aB_clist: {
4745       \seq_use:Nn \l_stex_aB_args_seq {#2}
4746     }
4747   #1
4748 }
4749 }
```

(End of definition for \argsep. This function is documented on page 95.)

\argmap

```

4750 \cs_new_protected:Npn \argmap #1 #2 #3 {
4751   \__stex_notations_check_aB_arg:Nn\argmap{#1}
4752   \stex_pseudogroup_with:nn{
4753     \__stex_term_do_aB_clist:
4754     \__stex_notations_map_cs:
4755   }{
4756     \cs_set:Npn \__stex_notations_map_cs: ##1 { #2 }
4757     \tl_set:Nn \__stex_term_do_aB_clist: {
4758       \seq_clear:N \l_tmpa_seq
4759       \seq_map_inline:Nn \l_stex_aB_args_seq {
4760         \tl_if_eq:nnTF{##1}{\ellipses}{
4761           \seq_put_right:Nn \l_tmpa_seq \ellipses
4762         }{
4763           \seq_put_right:Nx \l_tmpa_seq {
4764             \exp_after:wN \exp_not:n \exp_after:wN { \__stex_notations_map_cs: {##1} }
4765           }
4766         }
4767       }
4768       \seq_set_eq:NN \l_stex_aB_args_seq \l_tmpa_seq
4769       \seq_use:Nn \l_stex_aB_args_seq {#3}
4770     }
4771   #1
4772 }
4773 }
```

(End of definition for \argmap. This function is documented on page 95.)

```

\argarraymap
4774 \int_new:N \l__stex_notations_clist_count_int
4775 \cs_new_protected:Npn \argarraymap #1 #2 #3 #4 {
4776   \__stex_notations_check_aB_arg:Nn\argarraymap{#1}
4777   \stex_pseudogroup_with:nn{
4778     \__stex_term_do_aB_clist:
4779     \__stex_notations_map_cs:
4780   }{
4781     \cs_set:Npn \__stex_notations_map_cs: ##1 { #3 }
4782     \int_set:Nn \l__stex_notations_clist_count_int {\exp_args:No\clist_count:n{\tl_to_str:n}
4783     \tl_set:Nn \l_stex_term_do_aB_clist: {
4784       \tl_clear:N \l_tmpa_tl
4785       \int_zero:N \l_tmpa_int
4786       \seq_map_inline:Nn \l_stex_aB_args_seq {
4787         \int_incr:N \l_tmpa_int
4788         \int_compare:nNnT \l_tmpa_int > \l__stex_notations_clist_count_int {
4789           \int_set:Nn \l_tmpa_int 1
4790         }
4791         \tl_put_right:Nx \l_tmpa_tl {
4792           \exp_after:wN \exp_not:n \exp_after:wN { \__stex_notations_map_cs: {##1} }
4793           \clist_item:nn{#4}\l_tmpa_int
4794         }
4795       }
4796       \seq_set_eq:NN \l_stex_aB_args_seq \l_tmpa_seq
4797       \begin{array}{#2}
4798         \l_tmpa_tl
4799       \end{array}
4800     }
4801     #1
4802   }
4803 }

```

(End of definition for `\argarraymap`. This function is documented on page 95.)

### 13.8.3 Variables

```

4804 <@=stex_vars>
\vardef
4805 \prop_new:N \l_stex_variables_prop
4806 \bool_new:N \l__stex_vars_bind_bool
4807 \cs_new_protected:Nn \stex_variable:nnnnnnnN {}
4808 \stex_keys_define:nnnn{vardef} {
4809   \bool_set_false:N \l__stex_vars_bind_bool
4810 }
4811   bind .bool_set:N = \l__stex_vars_bind_bool
4812 }{symdef}
4813
4814 \stex_new_stylistable_cmd:nnnn {vardef} { m 0{} m} {
4815   \stex_keys_set:nn{vardef}{#2}
4816   \str_set:Nx \l_stex_macroname_str { #1 }
4817   \str_if_empty:NT \l_stex_key_name_str {
4818     \str_set:Nx \l_stex_key_name_str { #1 }
4819   }

```

```

4820   \stex_symdecl_do:
4821   \_stex_symdecl_check_terms:
4822   \__stex_vars_add:
4823   \__stex_vars_macro:
4824   \stex_if_do_html:T \__stex_vars_html:
4825
4826
4827   \int_set:Nn \l_stex_get_symbol_arity_int {\l_stex_get_symbol_arity_int}
4828   \stex_debug:nn{vardef}{Doing~\l_stex_key_name_str}
4829   \tl_set_eq:NN \l_stex_get_symbol_return_tl \l_stex_key_return_tl
4830   \stex_notation_parse:n{#3}
4831   \stex_if_check_terms:T{ \_stex_notation_check: }
4832   \_stex_vardecl_notation_macro:
4833   \stex_if_do_html:T {
4834     \def\comp{\_varcomp}
4835     \_stex_notation_do_html:n \l_stex_key_name_str
4836   }
4837   \group_begin:
4838   \tl_set_eq:NN \thisvarname \l_stex_key_name_str
4839   \tl_clear:N \thisstyle
4840   \str_set_eq:NN\thisnotationvariant\l_stex_key_variant_str
4841   \def\thisnotation{
4842     $ \let\l_stex_current_symbol_str\thisvarname
4843     \def\comp{\_varcomp}\exp_args:Nne \use:nn{\l_stex_notation_mmacrocode_{}}{
4844       \_stex_notation_make_args:
4845     }$}
4846   }
4847   \stex_style_apply:
4848   \group_end:\ignorespaces
4849 }{}}
4850
4851 \cs_new_protected:Nn \__stex_vars_add: {
4852   \exp_args:NNNo \exp_args:NNnx
4853   \prop_put:Nnn \l_stex_variables_prop \l_stex_key_name_str {
4854     {\l_stex_mroname_str}
4855     {\l_stex_key_name_str}
4856     {\int_use:N \l_stex_get_symbol_arity_int}
4857     {\l_stex_get_symbol_args_tl}
4858     {\exp_args:No \exp_not:n \l_stex_key_def_tl}
4859     {\exp_args:No \exp_not:n \l_stex_key_type_tl}
4860     {\exp_args:No \exp_not:n \l_stex_key_return_tl}
4861   \stex_invoke_symbol:
4862 }
4863 }
4864
4865 \cs_new_protected:Nn \__stex_vars_macro: {
4866   \tl_set:cx{\l_stex_mroname_str} {
4867     \_stex_invoke_variable:nnnnnnN
4868     {\l_stex_key_name_str}
4869     {\int_use:N \l_stex_get_symbol_arity_int}
4870     {\l_stex_get_symbol_args_tl}
4871     {\exp_args:No \exp_not:n \l_stex_key_def_tl}
4872     {\exp_args:No \exp_not:n \l_stex_key_type_tl}
4873     {\exp_args:No \exp_not:n \l_stex_key_return_tl}

```

```

4874     \stex_invoke_symbol:
4875 }
4876 }
4877
4878 \cs_new_protected:Nn \__stex_vars_html: {
4879     \stex_if_do_html:T {
4880         \hbox\bgroup\exp_args:Ne \stex_annotation_invisible:nn {
4881             data-shtml-vardef = {\l_stex_key_name_str},
4882             data-shtml-args = {\l_stex_key_args_str}
4883             \str_if_empty:NF \l_stex_mroname_str {,
4884                 data-shtml-macroname={\l_stex_mroname_str}
4885             }
4886             \str_if_empty:NF \l_stex_key_assoc_str {,
4887                 data-shtml-assoc_type={\l_stex_key_assoc_str}
4888             }
4889             \str_if_empty:NF \l_stex_key_role_str {,
4890                 data-shtml-role={\l_stex_key_role_str}
4891             }
4892             \str_if_empty:NF \l_stex_key_reorder_str {,
4893                 data-shtml-reorderargs={\l_stex_key_reorder_str}
4894             }
4895             \bool_if:NT \l_stex_vars_bind_bool {,
4896                 data-shtml-bind={}
4897             }
4898 }{
4899     \stex_annotation_force_break:n{
4900         \bool_set_true:N \stex_in_invisible_html_bool
4901         \tl_if_empty:NF \l_stex_key_type_tl {
4902             \stex_annotation:nn{data-shtml-type={}}{\$ \l_stex_key_type_tl \$}
4903         }
4904         \tl_if_empty:NF \l_stex_key_def_tl {
4905             \stex_annotation:nn{data-shtml-definiens={}}{\$ \l_stex_key_def_tl \$}
4906         }
4907         \tl_if_empty:NF \l_stex_key_return_tl{
4908             \exp_args:Nno \use:n{
4909                 \cs_generate_from_arg_count:NNnn \l__stex_vars_cs
4910                 \cs_set:Npn \l_stex_get_symbol_arity_int} \l_stex_key_return_tl
4911                 \tl_set:Nx \l_stex_vars_args_tl {\l_stex_map_args:N \stex_return_args:nn}
4912                 \$\stex_annotation:nn{data-shtml-return_type={}}{%
4913                     \exp_after:wN \l__stex_vars_cs \l_stex_vars_args_tl!}$
4914             }
4915             \tl_if_empty:NF \l_stex_key_argtypes_clist {
4916                 \stex_annotation:nn{data-shtml-argtypes={}}{%
4917                     \stex_annotation_force_break:n{
4918                         \clist_map_inline:Nn \l_stex_key_argtypes_clist {
4919                             \$\stex_annotation:nn{data-shtml-type={}}{\#\#1\$}
4920                         }
4921                     }
4922                 }
4923             }
4924         }
4925     }\egroup
4926 }
4927 }

```

(End of definition for \vardef. This function is documented on page 98.)

```
\_stex_vardecl_notation_macro:
4928 \cs_new_protected:Nn \_stex_vardecl_notation_macro: {
4929   \tl_set_eq:cN {l_stex_notation_
4930     \l_stex_key_name_str _
4931     \l_stex_key_variant_str _cs
4932   }\l_stex_notation_macrocode_cs
4933   \cs_if_exist:cF {l_stex_notation_\l_stex_key_name_str __cs} {
4934     \tl_set_eq:cN{l_stex_notation_\l_stex_key_name_str __cs}
4935     \l_stex_notation_macrocode_cs
4936   }
4937   \tl_if_empty:NF \l_stex_key_op_tl {
4938     \tl_set_eq:cN {l_stex_notation_\l_stex_key_name_str _op_
4939       \l_stex_key_variant_str _cs}\l_stex_key_op_tl
4940     \cs_if_exist:cF {l_stex_notation_\l_stex_key_name_str _op__cs} {
4941       \cs_set_eq:cN{l_stex_notation_\l_stex_key_name_str _op__cs}
4942       \l_stex_key_op_tl
4943     }
4944   }
4945 }
```

(End of definition for \\_stex\_vardecl\_notation\_macro:. This function is documented on page 128.)

```
\stex_get_symbol_or_var:n
\stex_get_var:n
4946 \cs_new_protected:Nn \__stex_vars_set_vars:nnnnnnN {
4947   \stex_debug:nn{symbols}{Variable~#1-found}
4948   \cs_set:Npn \stex_variable:nnnnnnN ##1 ##2 ##3 ##4 ##5 ##6 ##7 ##8 {}
4949   \str_clear:N \l_stex_get_symbol_mod_str
4950   \str_set:Nn \l_stex_get_symbol_name_str {#1}
4951   \int_set:Nn \l_stex_get_symbol_arity_int {#2}
4952   \tl_set:Nn \l_stex_get_symbol_args_tl {#3}
4953   \tl_set:Nn \l_stex_get_symbol_def_tl {#4}
4954   \tl_set:Nn \l_stex_get_symbol_type_tl {#5}
4955   \tl_set:Nn \l_stex_get_symbol_return_tl {#6}
4956   \tl_set:Nn \l_stex_get_symbol_invoke_cs {#7}
4957 }
4958
4959 \cs_new_protected:Nn \__stex_vars_get_var:n {
4960   \prop_map_inline:Nn \l_stex_variables_prop {
4961     \__stex_vars_check_var:nnnnnnnn {#1} ##2
4962   }
4963 }
4964
4965 \cs_new_protected:Nn \__stex_vars_check_var:nnnnnnnn {
4966   \str_if_eq:nnTF{#1}{#2} {
4967     \prop_map_break:n{\__stex_vars_set_vars:nnnnnnN {#3}{#4}{#5}{#6}{#7}{#8}{#9}}
4968   }{
4969     \str_if_eq:nnT{#1}{#3} {
4970       \prop_map_break:n{\__stex_vars_set_vars:nnnnnnN {#3}{#4}{#5}{#6}{#7}{#8}{#9}}
4971     }
4972   }
4973 }
```

```

4975 \cs_new_protected:Nn \stex_get_var:n {
4976   \str_clear:N \l_stex_get_symbol_name_str
4977   \__stex_vars_get_var:n{#1}
4978   \str_if_empty:NT \l_stex_get_symbol_name_str {
4979     \msg_error:nnn{stex}{error/unknownsymbol}{#1}
4980   }
4981 }
4982
4983 \cs_new_protected:Nn \stex_get_symbol_or_var:n {
4984   \str_clear:N \l_stex_get_symbol_name_str
4985   \__stex_vars_get_var:n{#1}
4986   \str_if_empty:NT \l_stex_get_symbol_name_str {
4987     \stex_debug:nn{symbols}{No~variable~#1~found}
4988     \stex_get_symbol:n{#1}
4989   }
4990 }

```

(End of definition for `\stex_get_symbol_or_var:n` and `\stex_get_var:n`. These functions are documented on page 128.)

### \svar

```

4991 \NewDocumentCommand \svar {O{} m}{
4992   \group_begin:
4993   \tl_if_empty:nTF{#1} {
4994     \str_set:Nn \l_stex_current_symbol_str {#2}
4995   }{
4996     \str_set:Nn \l_stex_current_symbol_str {#1}
4997   }
4998   \bool_if:NTF \l_stex_allow_semantic_bool {
4999     \tl_clear:N \l_stex_current_term_tl
5000     \stex_term_omv:nnn{}{}{\l_varcomp{#2}}
5001   }{
5002     \msg_error:nnxx{stex}{error/notallowed}{Variable}{\l_stex_current_symbol_str}
5003   }
5004   \group_end:
5005 }

```

(End of definition for `\svar`. This function is documented on page 98.)

### 13.8.4 Sequences

```
5006 <@=stex_seqs>
```

#### \varseq

```

5007 \stex_new_stylable_cmd:nnnn {varseq}{m O{} m m} {
5008   \stex_keys_set:nn{symdef}{#2}
5009   \str_set:Nx \l_stex_macroname_str { #1 }
5010   \str_if_empty:NT \l_stex_key_name_str {
5011     \str_set:Nx \l_stex_key_name_str { #1 }
5012   }
5013   \str_if_empty:NT \l_stex_key_args_str {
5014     \str_set:Nn \l_stex_key_args_str {1}
5015   }
5016   \stex_symdecl_do:
5017 
```

```

5018 \tl_set_eq:NN \l_stex_get_symbol_return_tl \l_stex_key_return_tl
5019 \clist_set:Nn \l__stex_seqs_range_clist {#3}
5020 \tl_if_empty:NTF \l_stex_key_op_tl {
5021   \stex_notation_parse:n{#4}
5022   \tl_clear:N \l_stex_key_op_tl
5023 }{
5024   \stex_notation_parse:n{#4}
5025 }
5026 \stex_if_do_html:T \__stex_seqs_html:
5027 \stex_if_check_terms:T \__stex_seqs_check_terms:
5028 \__stex_seqs_add:
5029 \__stex_seqs_macro:
5030 \stex_if_check_terms:T \_stex_notation_check:
5031 \stex_vardecl_notation_macro:
5032 \group_begin:
5033 \tl_set_eq:NN \thisvarname \l_stex_key_name_str
5034 \tl_clear:N \thisstyle
5035 \str_set_eq:NN\thisnotationvariant\l_stex_key_variant_str
5036 \def\thisnotation{
5037   \$\let\l_stex_current_symbol_str\thisvarname
5038   \def\comp{\_varcomp}\exp_args:Nne \use:nn{\l_stex_notation_macrocode_cs{}}
5039   \__stex_seqs_make_args:
5040 }
5041 }
5042 \stex_style_apply:
5043 \group_end:\ignorespaces
5044 }{}

5045 \cs_new_protected:Nn \__stex_seqs_add: {
5046   \exp_args:NNNo \exp_args:NNnx
5047   \prop_put:Nnn \l_stex_variables_prop \l_stex_key_name_str {
5048     {\l_stex_mroname_str}
5049     {\l_stex_key_name_str}
5050     {\int_use:N \l_stex_get_symbol_arity_int}
5051     {\l_stex_get_symbol_args_tl}
5052     {\exp_args:No \exp_not:n \l_stex_key_def_tl}
5053     {\exp_args:No \exp_not:n \l_stex_seqs_range_clist}
5054     {\exp_args:No \exp_not:n \l_stex_key_return_tl}
5055     \stex_invoke_sequence:
5056   }
5057 }
5058 }

5059 \cs_new_protected:Nn \__stex_seqs_macro: {
5060   \tl_set:cx{\l_stex_mroname_str}{%
5061     \stex_invoke_variable:nnnnnnN
5062     {\l_stex_key_name_str}
5063     {\int_use:N \l_stex_get_symbol_arity_int}
5064     {\l_stex_get_symbol_args_tl}
5065     {\exp_args:No \exp_not:n \l_stex_key_def_tl}
5066     {\exp_args:No \exp_not:n \l_stex_seqs_range_clist}
5067     {\exp_args:No \exp_not:n \l_stex_key_return_tl}
5068     \stex_invoke_sequence:
5069   }
5070 }
5071 }

```

```

5072
5073 \cs_new_protected:Nn \__stex_seqs_make_args: { \TODO }
5074 \cs_new_protected:Nn \__stex_seqs_check_terms: { \TODO }
5075
5076 \cs_new_protected:Nn \__stex_seqs_html: {
5077     \exp_args:Ne \stex_annotation_invisible:nn {
5078         data-shtml-varseq = {\l_stex_key_name_str},
5079         data-shtml-args = {\l_stex_key_args_str}
5080         \str_if_empty:NF \l_stex_mroname_str {,
5081             data-shtml-macroname=\l_stex_mroname_str}
5082     }
5083     \str_if_empty:NF \l_stex_key_assoc_str {,
5084         data-shtml-assoc=\l_stex_key_assoc_str}
5085     }
5086     \str_if_empty:NF \l_stex_key_role_str {,
5087         data-shtml-role=\l_stex_key_role_str}
5088     }
5089     \str_if_empty:NF \l_stex_key_reorder_str {,
5090         data-shtml-reorderargs=\l_stex_key_reorder_str}
5091     }
5092 }{\hbox\bgroup
5093     \stex_annotation_force_break:n{
5094         \tl_if_empty:NF \l_stex_key_type_tl {
5095             \stex_annotation:nn{data-shtml-type={}}{$\l_stex_key_type_tl$}
5096         }
5097         \tl_if_empty:NF \l_stex_key_def_tl {
5098             \stex_annotation:nn{data-shtml-definiens={}}{$\l_stex_key_def_tl$}
5099         }
5100         \tl_if_empty:NF \l_stex_key_return_tl{
5101             \exp_args:Nno \use:n{
5102                 \cs_generate_from_arg_count:NNnn \l__stex_seqs_cs
5103                 \cs_set:Npn \l_stex_get_symbol_arity_int} \l_stex_key_return_tl
5104                 \tl_set:Nx \l__stex_seqs_args_tl {\l_stex_map_args:N \stex_return_args:nn}
5105                 \stex_annotation:nn{data-shtml-returntype={}}{
5106                     $ \exp_after:wN \l__stex_seqs_cs \l__stex_seqs_args_tl!$}
5107             }
5108             \tl_if_empty:NF \l_stex_key_argtypes_clist {
5109                 \stex_annotation:nn{data-shtml-argtypes={}}{
5110                     \stex_annotation_force_break:n{
5111                         \clist_map_inline:Nn \l_stex_key_argtypes_clist {
5112                             \stex_annotation:nn{data-shtml-type={}}{$##1$}
5113                         }
5114                     }
5115                 }
5116             }
5117         }
5118     \egroup}
5119 }

```

(End of definition for `\varseq`. This function is documented on page 98.)

`\stex_invoke_sequence:`

```

5120 \cs_new_protected:Nn \stex_invoke_sequence: {
5121     \peek_charcode_remove:NTF ! {

```

```

5122     \peek_charcode:NTF [ \__stex_seqs_do_op:w { \__stex_seqs_do_op:w [] } ]
5123     }\__stex_seqs_do_first:
5124 }
5125
5126 \cs_new_protected:Npn \__stex_seqs_do_op:w [#1] {
5127     \cs_if_exist:cTF {l_stex_notation_\l_stex_current_symbol_str _op_#1_cs} {
5128         \stex_maybe_brackets:n{ \neginfprefc }{
5129             \stex_term_oms_or_omv:n{ \neginfprefc }{
5130                 \use:c{l_stex_notation_\l_stex_current_symbol_str _op_#1_cs} }
5131             }
5132         \group_end:
5133     }{
5134         \__stex_seqs_get_index_notation:n{ \neginfprefc }
5135         \peek_charcode:NTF [ \__stex_seqs_doop_range:w { \__stex_seqs_doop_range:w[] } ]
5136     }
5137 }
5138
5139 \cs_new_protected:Npn \__stex_seqs_doop_range:w [#1] {
5140     \bool_set_true:N \l_stex_allow_semantic_bool
5141     \clist_clear:N \l__stex_seqs_clist
5142     \clist_map_function:NN \l_stex_current_type_tl \__stex_seqs_doop_arg:n
5143         \stex_annotation:n{
5144             data-shtml-term=OMV,
5145             data-shtml-head={\l_stex_current_symbol_str},
5146             data-shtml-notationid={}
5147         }{
5148         \l__stex_seqs_clist
5149     }
5150     \group_end:
5151 }
5152
5153 \cs_new_protected:Nn \__stex_seqs_doop_arg:n {
5154     \tl_if_eq:nnTF{ \ellipses }{
5155         \clist_put_right:Nn \l__stex_seqs_clist {
5156             \ellipses
5157         }
5158     }{
5159         \clist_put_right:Nn \l__stex_seqs_clist {
5160             \exp_args:Nno \str_if_eq:nnTF \l_stex_current_arity_str {1} {
5161                 \group_begin:
5162                 \l__stex_seqs_CS \group_end: { \ellipses }
5163             }{
5164                 \group_begin:
5165                 \l__stex_seqs_CS \group_end: #1
5166             }
5167         }
5168     }
5169 }
5170 }
5171
5172 \cs_new_protected:Nn \__stex_seqs_get_index_notation:n {
5173     \cs_if_exist:cTF {l_stex_notation_\l_stex_current_symbol_str _#1_cs} {
5174         \cs_set_eq:Nc \l__stex_seqs_CS {l_stex_notation_\l_stex_current_symbol_str _#1_cs}
5175     }

```

```

5176     \stex_do_default_notation:
5177     \cs_set_eq:NN \l__stex_seqs_cs \l_stex_default_notation
5178   }
5179 }
5180
5181
5182 \cs_new:Nn \__stex_seqs_do_first_arg:n {{\exp_not:n{## #1}}}
5183
5184 \cs_new_protected:Nn \__stex_seqs_do_first: {
5185   \exp_args:Nnx \use:nn{
5186     \cs_generate_from_arg_count:NNnn \l__stex_seqs_cs \cs_set:Npn
5187     \l_stex_current_arity_str {{
5188       \tl_set:Nn \exp_not:N \l__stex_seqs_first_args_tl {
5189         \int_step_function:nN \l_stex_current_arity_str \__stex_seqs_do_first_arg:n
5190       }
5191       \exp_not:N \__stex_seqs_do_first_next:
5192     }}
5193   \l__stex_seqs_cs
5194 }
5195
5196 \cs_new_protected:Nn \__stex_seqs_do_first_next: {
5197   \peekCharCode_remove:NTF ! {
5198     \peekCharCode:NTF [ \__stex_seqs_do_one:w {\__stex_seqs_do_one:w []}]
5199   }{
5200     \peekCharCode:NTF [ \__stex_seqs_do_all:w {\__stex_seqs_do_all:w []}]
5201   }
5202 }
5203
5204 \cs_new_protected:Npn \__stex_seqs_do_one:w [#1] {
5205   \__stex_seqs_get_index_notation:n{#1}
5206   \stex_debug:nn{HERE~seq~one}{\meaning\l__stex_seqs_cs^{\meaning\l__stex_seqs_first_args_t1}}
5207   \exp_args:Nno\use:nn{\l__stex_seqs_cs\group_end:}\l__stex_seqs_first_args_t1
5208 }
5209
5210 \cs_new_protected:Npn \__stex_seqs_do_all:w [#1] {
5211   \stex_debug:nn{HERE~seq~all}{\meaning\l__stex_seqs_first_args_t1}
5212   \exp_args:Nno\use:nn{\l_stex_invoke_notation:w [#1]}\l__stex_seqs_first_args_t1
5213 }

```

(End of definition for `\stex_invoke_sequence`. This function is documented on page ??.)

`\seqmap`

```

5214 \cs_new_protected:Npn \seqmap #1 #2 {
5215   \symuse{Metatheory?sequence-expression}{\seqmap{#1}{#2}}%\l_tmpa_t1 {#2}
5216 }

```

(End of definition for `\seqmap`. This function is documented on page 99.)

### 13.8.5 Expressions

5217 `<@@=stex_expr>`

Various variables:

```

5218 \bool_new:N \l_stex_allow_semantic_bool
5219 \bool_set_true:N \l_stex_allow_semantic_bool

```

```

5220 \tl_new:N \l_stex_current_term_tl
5221 \tl_set:Nn \l_stex_every_symbol_tl {
5222   \bool_set_false:N \l_stex_allow_semantic_bool
5223 }

\stex_next_symbol:n

5224 \tl_new:N \l__stex_expr_reset_tl
5225 \cs_new_protected:Nn \stex_next_symbol:n {
5226   \tl_set:Nx \l_stex_every_symbol_tl {
5227     \tl_gset:Nn \exp_not:N \l__stex_expr_reset_tl {
5228       \tl_set:Nn \exp_not:N \l_stex_every_symbol_tl {
5229         \exp_args:No \exp_not:n \l_stex_every_symbol_tl
5230       }
5231       \tl_gset:Nn \exp_not:N \l__stex_expr_reset_tl {
5232         \exp_args:No \exp_not:n \l__stex_expr_reset_tl
5233       }
5234     }
5235     \tl_set:Nn \exp_not:N \l_stex_every_symbol_tl {
5236       \exp_args:No \exp_not:n \l_stex_every_symbol_tl
5237     }
5238     \exp_not:n{ \aftergroup \l__stex_expr_reset_tl }
5239     \exp_not:N \l_stex_every_symbol_tl
5240     \exp_not:n{ #1 }
5241   }
5242 }
5243 \cs_generate_variant:Nn \stex_next_symbol:n {e}

(End of definition for \stex_next_symbol:n. This function is documented on page ??.)

```

#### \STEXinvisible

```

5244 \cs_new_protected:Npn \STEXinvisible #1 {
5245   \stex_annotation_invisible:n { #1 }
5246 }

```

(End of definition for \STEXinvisible. This function is documented on page 84.)

### Invoking Semantic Macros

#### \stex\_invoke\_symbol:nnnnnnN

```

5247 \cs_new_protected:Nn \stex_invoke_symbol:nnnnnnN {
5248   \bool_if:NTF \l_stex_allow_semantic_bool{
5249     \stex_if_html_backend:T{\relax\ifvmode\indent\fi}
5250     \__stex_expr_setup:nnnnnf\comp{\#1?\#2}{\#3}{\#4}{\#7}{\#6}
5251     \cs_set_eq:NN \stex_term_oms_or_omv:nnn \stex_term_oms:nnn
5252     \tl_put_right:Nn \l_stex_current_redo_tl{
5253       \cs_set_eq:NN \stex_term_oms_or_omv:nnn \stex_term_oms:nnn
5254     }
5255     #8
5256   }{
5257     \msg_error:nnxx{stex}{error/notallowed}{\#1?\#2}{\l_stex_current_symbol_str}
5258   }
5259 }
5260 \cs_generate_variant:Nn \stex_invoke_symbol:nnnnnnN {ooxooooN}
5261
5262 \cs_new_protected:Nn \__stex_expr_setup:nnnnn {

```

```

5263 \group_begin:
5264 \tl_clear:N \l_stex_return_notation_tl
5265 \tl_set:Nn \l_stex_current_redo_tl {
5266   \let \this \stex_current_this:
5267   \def\comp{\#1}
5268   \def\maincomp{\comp}
5269   \str_set:Nn \l_stex_current_symbol_str {\#2}
5270   \str_set:Nn \l_stex_current_arity_str{ #3 }
5271   \tl_set:Nn \l_stex_current_args_tl{ #4 }
5272   \tl_set:Nn \l_stex_current_return_tl{ #5 }
5273   \tl_set:Nn \l_stex_current_type_tl{ #6 }
5274   \tl_clear:N \l_stex_current_term_tl
5275 }
5276 \tl_put_right:Nx \l_stex_current_redo_tl {
5277   \exp_args:No \exp_not:n \l_stex_every_symbol_tl
5278 }
5279 \l_stex_current_redo_tl
5280 }

```

(End of definition for `\_stex_invoke_symbol:nnnnnnN`. This function is documented on page ??.)

```

\_stex_invoke_variable:nnnnnn
5281 \cs_new_protected:Nn \_stex_invoke_variable:nnnnnnN {
5282   \bool_if:NTF \l_stex_allow_semantic_bool{
5283     \stex_if_html_backend:T{\relax\ifvmode\indent\fi}
5284     \__stex_expr_setup:nnnnnn{\varcomp{\#1}{\#2}{\#3}{\#6}{\#5}}
5285     \cs_set_eq:NN \stex_term_oms_or_omv:nnn \stex_term_omv:nnn
5286     \tl_put_right:Nn \l_stex_current_redo_tl {
5287       \cs_set_eq:NN \stex_term_oms_or_omv:nnn \stex_term_omv:nnn
5288     }
5289     #7
5290   }{
5291     \msg_error:nnxx{stex}{error/notallowed}{\#1}{\l_stex_current_symbol_str}
5292   }
5293 }

```

(End of definition for `\_stex_invoke_variable:nnnnnn`. This function is documented on page ??.)

```

\symuse
5294 \cs_new_protected:Npn \symuse #1 {
5295   \stex_get_symbol:n{\#1}
5296   \exp_args:Nno \use:n {\_stex_invoke_symbol:ooooooN
5297   \l_stex_get_symbol_mod_str
5298   \l_stex_get_symbol_name_str
5299   {\int_use:N \l_stex_get_symbol_arity_int}
5300   \l_stex_get_symbol_args_tl
5301   \l_stex_get_symbol_def_tl
5302   \l_stex_get_symbol_type_tl
5303   \l_stex_get_symbol_return_tl}
5304   \l_stex_get_symbol_invoke_cs
5305 }

```

(End of definition for `\symuse`. This function is documented on page 92.)

\stex\_invoke\_symbol: Top-Level: Check whether text/math mode or custom notation, whether delimited by !, return code etc.

```

5306 \cs_new_protected:Nn \stex_invoke_symbol: {
5307   \stex_debug:nn{expressions}{Invoking-\l_stex_current_symbol_str}
5308   \mode_if_math:TF \__stex_expr_invoke_math: \__stex_expr_invoke_text:
5309 }
5310
5311 \cs_new_protected:Nn \__stex_expr_invoke_text: {
5312   \stex_debug:nn{expressions}{text-mode}
5313   \peek_charcode_remove:NTF ! \__stex_expr_invoke_op_custom:n \__stex_expr_invoke_custom:n
5314 }
5315
5316 \cs_new_protected:Nn \__stex_expr_invoke_math: {
5317   \stex_debug:nn{expressions}{math-mode}
5318   \peek_charcode_remove:NTF ! {
5319     % operator
5320     \peek_charcode_remove:NTF * \__stex_expr_invoke_op_custom:n {
5321       % op notation
5322       \peek_charcode:NTF [ \__stex_expr_invoke_op_notation:w {
5323         \__stex_expr_invoke_op_notation:w []
5324       }
5325     }
5326   }
5327   \peek_charcode_remove:NTF * \__stex_expr_invoke_custom:n {
5328     % normal
5329     \peek_charcode:NTF [ \__stex_invoke_notation:w {
5330       \__stex_invoke_notation:w []
5331     }
5332   }
5333 }
5334 }
```

Notations:

```

5335 \cs_new_protected:Npn \stex_invoke_notation:w [#1] {
5336   \peek_charcode_remove:NTF !
5337   \__stex_expr_invoke_op_notation:w [#1]
5338 }{
5339   \stex_debug:nn{expressions}{using-notation-#1-for-\l_stex_current_symbol_str}
5340   \cs_if_exist:cTF{\l_stex_notation_\l_stex_current_symbol_str _#1_cs} {
5341     \tl_if_empty:NTF \l_stex_current_return_tl {
5342       \stex_debug:nn{expressions}{return-empty}
5343       \use:c{\l_stex_notation_\l_stex_current_symbol_str _#1_cs}{\group_end:\l_stex_eat_excl}
5344     }{
5345       \stex_debug:nn{expressions}{return?}
5346       \exp_after:wN\exp_after:wN\exp_after:wN
5347       \__stex_expr_invoke_return_maybe:n
5348       \exp_after:wN\exp_after:wN\exp_after:wN
5349       {\cs:w \l_stex_notation_\l_stex_current_symbol_str _#1_cs \cs_end: {}}
5350     }
5351   }{
5352     \stex_do_default_notation:
5353     \tl_if_empty:NTF \l_stex_current_return_tl {
5354       \l_stex_default_notation{\group_end:\l_stex_eat_exclamation_point:}
5355     }
5356 }
```

```

5356     \exp_after:wN
5357     \_stex_expr_invoke_return_maybe:n
5358     \exp_after:wN
5359     {\l_stex_default_notation {}}
5360   }
5361 }
5362 }
5363 }
5364
5365 \cs_new_protected:Npn \__stex_expr_invoke_op_notation:w [#1] {
5366   \stex_debug:nn{expressions}{op~notation~for~\l_stex_current_symbol_str}
5367   \cs_if_exist:cTF{\l_stex_notation_\l_stex_current_symbol_str _op_#1_cs} {
5368     \stex_maybe_brackets:nn{\neginfpref}{%
5369       \stex_term_oms_or_omv:nnn{#1}{}%
5370       {\use:c{\l_stex_notation_\l_stex_current_symbol_str _op_#1_cs}}%
5371     }
5372     \group_end:
5373   }{
5374     \int_compare:nNnTF \l_stex_current_arity_str = 0 {
5375       \tl_clear:N \l_stex_current_return_tl
5376       \stex_invoke_notation:w [#1]
5377     }{
5378       \stex_do_default_notation_op:
5379       \stex_maybe_brackets:nn{\neginfpref}{%
5380         \stex_term_oms_or_omv:nnn{#1}{}%
5381         {\l_stex_default_notation}%
5382       }
5383       \group_end:
5384     }
5385   }
5386 }

```

Return:

```

5387 \cs_new_protected:Nn \__stex_expr_invoke_return_maybe:n {
5388   \tl_clear:N \l__stex_expr_return_args_tl
5389   \tl_set:Nn \l__stex_expr_return_this_tl {#1}
5390   \exp_args:Nnx \use:n {
5391     \cs_generate_from_arg_count:NNnn \__stex_expr_ret_cs
5392     \cs_set:Npn \l_stex_current_arity_str } {
5393       \int_step_function:nN \l_stex_current_arity_str \__stex_expr_return_arg:n
5394       \__stex_expr_invoke_return_next:
5395     }
5396     \__stex_expr_ret_cs
5397   }
5398
5399 \cs_new:Nn \__stex_expr_return_arg:n {
5400   \tl_put_right:Nn \exp_not:N \l__stex_expr_return_args_tl {{#### #1}}
5401 }
5402
5403 \cs_new_protected:Nn \__stex_expr_invoke_return_next: {
5404   \peek_charcode_remove:NTF ! {
5405     \exp_after:wN \l__stex_expr_return_this_tl \l__stex_expr_return_args_tl \group_end:
5406   }\__stex_expr_invoke_return:
5407 }

```

```

5408 \cs_new_protected:Nn \__stex_expr_invoke_return: {
5409   \tl_set:Nx \l__stex_expr_return_this_tl {
5410     \__stex_expr_return_notation:n {
5411       \exp_after:wN \exp_after:wN \exp_after:wN
5412       \exp_not:n \exp_after:wN \exp_after:wN \exp_after:wN {
5413         \exp_after:wN \l__stex_expr_return_this_tl \l__stex_expr_return_args_tl
5414       }
5415     }
5416   }
5417 }
5418 \stex_debug:nn{return}{Notation:~\meaning\l__stex_expr_return_this_tl}
5419 \tl_put_left:Nx \l__stex_expr_return_this_tl {
5420   \group_begin:\exp_args:No \exp_not:n \l_stex_current_redo_tl
5421 }
5422 \exp_args:Nnx \use:n {
5423   \cs_generate_from_arg_count:NNnn \__stex_expr_ret_cs
5424   \cs_set:Npn \l_stex_current_arity_str } {
5425     \exp_args:No \exp_not:n \l_stex_current_return_tl
5426   }
5427 \stex_debug:nn{return}{
5428   \meaning\__stex_expr_ret_cs^~J
5429   \meaning\l__stex_expr_return_this_tl^~J
5430   \exp_args:No \exp_not:n \l__stex_expr_return_args_tl^~J
5431 }
5432 \exp_args:Nnx \use:nn {
5433   \exp_after:wN \group_end: \__stex_expr_ret_cs
5434 }{
5435   \exp_args:No \exp_not:n \l__stex_expr_return_args_tl
5436   {
5437     \exp_args:No \exp_not:n \l__stex_expr_return_this_tl
5438     \group_end:
5439   }
5440 }
5441 }
5442
5443
5444 \cs_new_protected:Nn \__stex_expr_return_notation:n {
5445   \tl_if_empty:NTF \l_stex_return_notation_tl { #1 }{
5446     \l_stex_return_notation_tl
5447   }
5448 }
5449

```

#### Custom Notations:

```

5450 \cs_new_protected:Nn \__stex_expr_invoke_op_custom:n {
5451   \stex_debug:nn{expressions}{custom~op}
5452   \bool_set_true:N \l_stex_allow_semantic_bool
5453   \stex_term_oms_or_omv:nnn{}{}{\maincomp{#1}}
5454   \group_end:
5455 }
5456
5457 \int_new:N \l__stex_expr_arg_counter_int
5458 \cs_new_protected:Nn \__stex_expr_invoke_custom:n {
5459   \stex_debug:nn{custom}{custom~notation~for~\l_stex_current_symbol_str}

```

```

5460 \stex_pseudogroup:nn{
5461   \bool_set_true:N \l_stex_allow_semantic_bool
5462   \prop_gclear:N \l__stex_expr_customs_prop
5463   \seq_gclear:N \l__stex_expr_customs_seq
5464   \int_gzero:N \l__stex_expr_arg_counter_int
5465   \tl_if_empty:NF \l_stex_current_args_tl {
5466     \exp_after:wN \__stex_expr_add_prop_arg:nw \l_stex_current_args_tl \_stex_args_end:
5467     \cs_set_eq:NN \arg \__stex_expr_arg:n
5468   }
5469   \tl_set_eq:NN \l_stex_get_symbol_args_tl \l_stex_current_args_tl
5470   \cs_set_eq:NN \__stex_expr_do_ab_next:nnn \_stex_term_oma:nnn
5471   \__stex_map_args:N \__stex_expr_check_b:nn
5472   \__stex_expr_do_ab_next:nnn{}{}{#1}
5473 }{
5474   \prop_if_exist:NT \l__stex_expr_customs_prop {
5475     \prop_gset_from_keyval:Nn \exp_not:N \l__stex_expr_customs_prop {
5476       \prop_to_keyval:N \l__stex_expr_customs_prop
5477     }
5478   }
5479   \int_gset:Nn \l__stex_expr_arg_counter_int { \int_use:N \l__stex_expr_arg_counter_int}
5480   \seq_if_exist:NT \l__stex_expr_customs_seq {
5481     \seq_gset_split:Nnn \exp_not:N \l__stex_expr_customs_seq , {
5482       \seq_use:Nn \l__stex_expr_customs_seq ,
5483     }
5484   }
5485 }
5486 % TODO check that all arguments are present
5487 \group_end:
5488 }
5489
5490 \cs_new_protected:Npn \__stex_expr_add_prop_arg:nw #1 #2 #3\stex_args_end: {
5491   \prop_gput:Nnn \l__stex_expr_customs_prop {#1} {}
5492   \seq_gput_right:Nn \l__stex_expr_customs_seq {#2}
5493   \tl_if_empty:nF{#3}{\__stex_expr_add_prop_arg:nw #3 \stex_args_end:}
5494 }
5495
5496 \cs_new:Nn \__stex_expr_check_b:nn {
5497   \str_case:nn #2 {
5498     b {\cs_set_eq:NN \__stex_expr_do_ab_next:nnn \_stex_term_omb:nnn}
5499     B {\cs_set_eq:NN \__stex_expr_do_ab_next:nnn \_stex_term_omb:nnn}
5500   }
5501 }
5502
5503 \NewDocumentCommand \__stex_expr_arg:n { s O{} m } {
5504   \IfBooleanTF #1 {
5505     \stex_annotation_invisible:n{
5506       \__stex_expr_arg_inner:nn{#2}{#3}
5507     }
5508   }{
5509     \__stex_expr_arg_inner:nn{#2}{#3}
5510   }
5511 }
5512
5513 \cs_new_protected:Nn \__stex_expr_arg_inner:nn {

```

```

5514 \tl_if_empty:nTF{#1}{
5515   \int_gincr:N \l__stex_expr_arg_counter_int
5516   \exp_args:Ne \__stex_expr_check:nTF{ \int_use:N \l__stex_expr_arg_counter_int }{
5517     \__stex_expr_arg_do:oon \l_tmpa_tl \l_tmpb_tl
5518   }{
5519     \__stex_expr_arg_inner:nn{}}
5520   }{ #2 }
5521 }{
5522   \__stex_expr_check:nTF {#1}{%
5523     \__stex_expr_arg_do:oon \l_tmpa_tl \l_tmpb_tl { #2 }
5524   }{
5525     \exp_args:No \str_case:nnTF \l_tmpb_tl {
5526       {a} {
5527         \exp_args:NNnx \prop_gput:Nnn \l__stex_expr_customs_prop {#1}{%
5528           \l_tmpa_tl X
5529         }
5530         \tl_set:Nx \l_tmpa_tl { #1 \int_eval:n {\tl_count:N \l_tmpa_tl + 1} }
5531       }
5532       {B} {
5533         \exp_args:NNnx \prop_gput:Nnn \l__stex_expr_customs_prop {#1}{%
5534           \l_tmpa_tl X
5535         }
5536         \tl_set:Nx \l_tmpa_tl { #1 \int_eval:n {\tl_count:N \l_tmpa_tl + 1} }
5537       }
5538     }{
5539       \__stex_expr_arg_do:oon \l_tmpa_tl \l_tmpb_tl { #2 }
5540     }{
5541       \msg_error:nnxx{stex}{error/invalidarg}{#1}{\l_stex_current_symbol_str}
5542     }
5543   }
5544 }
5545 }
5546
5547 \prg_new_conditional:Nnn \__stex_expr_check:n {TF} {
5548   \exp_args:NNe \prop_get:NnNTF \l__stex_expr_customs_prop {#1} \l_tmpa_tl {
5549     \tl_set:Nx \l_tmpb_tl {\seq_item:Nn \l__stex_expr_customs_seq {#1} }
5550     \tl_if_empty:NTF \l_tmpa_tl {
5551       \exp_args:NNe \prop_gput:Nnn \l__stex_expr_customs_prop
5552         { #1 }{X}
5553       \exp_args:No \str_case:nnF \l_tmpb_tl {
5554         {a} {
5555           \tl_set:Nx \l_tmpa_tl{ #1 1 }
5556         }
5557         {B} {
5558           \tl_set:Nx \l_tmpa_tl{ #1 1 }
5559         }
5560         \tl_set:Nx \l_tmpa_tl{ #1 }
5561       }
5562       \prg_return_true:
5563     }{
5564       \prg_return_false:
5565     }
5566   }
5567 }

```

```

5568     \msg_error:nxxx{stex}{error/invalidarg}{#1}{\l_stex_current_symbol_str}
5569     \prg_return_false:
5570   }
5571 }
5572 %
5573 % #1 argnum #2 argmode #3 code
5574 \cs_new_protected:Nn \__stex_expr_arg_do:nnn {
5575   \stex_debug:nn{custom}{Doing~argument~#1~of~mode~#2:~\tl_to_str:n{#3}}
5576   \group_begin:
5577     \bool_set_true:N \l_stex_allow_semantic_bool
5578     \__stex_term_arg:nnn {#2}{#1}{#3}
5579   \group_end:
5580 }
5581 \cs_generate_variant:Nn \__stex_expr_arg_do:nnn {oon}

```

(End of definition for `\stex_invoke_symbol`. This function is documented on page 134.)

## Argument Handling and Annotating

```

\__stex_term_arg:nnnnn
\__stex_term_arg:nnn
5582 % 1: argnum 2: argmode 3: precedence 4: argname 5: code
5583 \cs_new_protected:Nn \__stex_term_arg:nnnnn {
5584   \group_begin:
5585     \str_clear:N \l_stex_current_symbol_str
5586     \tl_clear:N \l_stex_current_term_tl
5587     \int_set:Nn \l_stex_notation_downprec { #3 }
5588     \bool_set_true:N \l_stex_allow_semantic_bool
5589     \__stex_term_arg:nnn {#2}{#1}%
5590       \tl_if_empty:nTF{#4}%
5591         #5
5592       }%
5593         \stex_annotation:nn{mml:arg={#4}}{#5}
5594       }
5595     }
5596   \group_end:
5597 }
5598 %
5599 \cs_new_protected:Nn \__stex_term_arg:nnn {
5600   \__stex_annotation_force_break:n{ \stex_annotation:nn{ data-shtml-arg={#2}, data-shtml-argmode={#3 } } }
5601 }
5602 }
```

(End of definition for `\__stex_term_arg:nnnnn` and `\__stex_term_arg:nnn`. These functions are documented on page ??.)

```

\__stex_term_arg_aB:nnnnn
5603 \tl_set:Nn \__stex_expr_do_aB_clist: {
5604   \seq_use:Nn \l_stex_aB_args_seq {
5605     \mathpunct{\comp{,}}
5606   }
5607 }
5608 \tl_set_eq:NN \__stex_term_do_aB_clist: \__stex_expr_do_aB_clist:
5609 %
5610 \int_new:N \l__stex_expr_count_int
5611 \cs_new_protected:Nn \__stex_term_arg_aB:nnnnn {
```

```

5612 \tl_if_empty:nTF{#5}{
5613   \_stex_term_arg:nnnnn{#1}{#2}{#3}{#4}{}
5614 }{
5615   \seq_clear:N \l_stex_aB_args_seq
5616   \int_zero:N \l_stex_expr_count_int
5617   \clist_map_inline:nn{#5}{
5618     \__stex_expr_aB_arg:nnnnn{##1}{#1}{#2}{#3}{#4}
5619   }
5620   \_stex_term_do_aB_clist:
5621 }
5622 }
5623
5624 % 1: code 2: argnum 3: argmode 4: precedence 5: argname
5625 \cs_new_protected:Npn \__stex_expr_aB_arg:nnnnn #1 {
5626   \int_incr:N \l_stex_expr_count_int
5627   \__stex_expr_is_varseq:nTF{#1} {
5628     \exp_after:wN \exp_after:wN \exp_after:wN
5629     \__stex_expr_assoc_seq:nnnnnnn
5630     \exp_after:wN
5631     \__stex_expr_gobble:nnnnnnnn #1 \__stex_expr_end:
5632   }
5633   \__stex_expr_is_seqmap:nTF{#1} {
5634     \exp_args:NNe \use:nn \__stex_expr_do_seqmap:nnnnnn {\tl_tail:n{#1}}
5635   }
5636   \__stex_expr_aB_simple_arg:nnnnn{#1}
5637 }
5638 }
5639 }
5640
5641 \cs_new:Npn \__stex_expr_gobble:nnnnnnnn #1 #2 #3 #4 #5 #6 #7 #8 #9 \__stex_expr_end: {
5642   {#2} #3 {#6}
5643 }
5644
5645 \cs_new_protected:Nn \__stex_expr_aB_simple_arg:nnnnn{
5646   \seq_put_right:Nx \l_stex_aB_args_seq {
5647     \_stex_term_arg:nnnnn{#2}\int_use:N\l_stex_expr_count_int}{#3}{#4}{#5}{#6}{#7}{#8}{#9}{#10}{#11}{#12}{#13}{#14}{#15}{#16}{#17}{#18}{#19}{#20}{#21}{#22}{#23}{#24}{#25}{#26}{#27}{#28}{#29}{#30}{#31}{#32}{#33}{#34}{#35}{#36}{#37}{#38}{#39}{#40}{#41}{#42}{#43}{#44}{#45}{#46}{#47}{#48}{#49}{#50}{#51}{#52}{#53}{#54}{#55}{#56}{#57}{#58}{#59}{#60}{#61}{#62}{#63}{#64}{#65}{#66}{#67}{#68}{#69}{#70}{#71}{#72}{#73}{#74}{#75}{#76}{#77}{#78}{#79}{#80}{#81}{#82}{#83}{#84}{#85}{#86}{#87}{#88}{#89}{#90}{#91}{#92}{#93}{#94}{#95}{#96}{#97}{#98}{#99}{#100}{#101}{#102}{#103}{#104}{#105}{#106}{#107}{#108}{#109}{#110}{#111}{#112}{#113}{#114}{#115}{#116}{#117}{#118}{#119}{#120}{#121}{#122}{#123}{#124}{#125}{#126}{#127}{#128}{#129}{#130}{#131}{#132}{#133}{#134}{#135}{#136}{#137}{#138}{#139}{#140}{#141}{#142}{#143}{#144}{#145}{#146}{#147}{#148}{#149}{#150}{#151}{#152}{#153}{#154}{#155}{#156}{#157}{#158}{#159}{#160}{#161}{#162}{#163}{#164}{#165}{#166}{#167}{#168}{#169}{#170}{#171}{#172}{#173}{#174}{#175}{#176}{#177}{#178}{#179}{#180}{#181}{#182}{#183}{#184}{#185}{#186}{#187}{#188}{#189}{#190}{#191}{#192}{#193}{#194}{#195}{#196}{#197}{#198}{#199}{#200}{#201}{#202}{#203}{#204}{#205}{#206}{#207}{#208}{#209}{#210}{#211}{#212}{#213}{#214}{#215}{#216}{#217}{#218}{#219}{#220}{#221}{#222}{#223}{#224}{#225}{#226}{#227}{#228}{#229}{#230}{#231}{#232}{#233}{#234}{#235}{#236}{#237}{#238}{#239}{#240}{#241}{#242}{#243}{#244}{#245}{#246}{#247}{#248}{#249}{#250}{#251}{#252}{#253}{#254}{#255}{#256}{#257}{#258}{#259}{#260}{#261}{#262}{#263}{#264}{#265}{#266}{#267}{#268}{#269}{#270}{#271}{#272}{#273}{#274}{#275}{#276}{#277}{#278}{#279}{#280}{#281}{#282}{#283}{#284}{#285}{#286}{#287}{#288}{#289}{#290}{#291}{#292}{#293}{#294}{#295}{#296}{#297}{#298}{#299}{#300}{#301}{#302}{#303}{#304}{#305}{#306}{#307}{#308}{#309}{#3010}{#3011}{#3012}{#3013}{#3014}{#3015}{#3016}{#3017}{#3018}{#3019}{#3020}{#3021}{#3022}{#3023}{#3024}{#3025}{#3026}{#3027}{#3028}{#3029}{#3030}{#3031}{#3032}{#3033}{#3034}{#3035}{#3036}{#3037}{#3038}{#3039}{#3040}{#3041}{#3042}{#3043}{#3044}{#3045}{#3046}{#3047}{#3048}{#3049}{#3050}{#3051}{#3052}{#3053}{#3054}{#3055}{#3056}{#3057}{#3058}{#3059}{#3060}{#3061}{#3062}{#3063}{#3064}{#3065}{#3066}{#3067}{#3068}{#3069}{#3070}{#3071}{#3072}{#3073}{#3074}{#3075}{#3076}{#3077}{#3078}{#3079}{#3080}{#3081}{#3082}{#3083}{#3084}{#3085}{#3086}{#3087}{#3088}{#3089}{#30810}{#30811}{#30812}{#30813}{#30814}{#30815}{#30816}{#30817}{#30818}{#30819}{#30820}{#30821}{#30822}{#30823}{#30824}{#30825}{#30826}{#30827}{#30828}{#30829}{#30830}{#30831}{#30832}{#30833}{#30834}{#30835}{#30836}{#30837}{#30838}{#30839}{#30840}{#30841}{#30842}{#30843}{#30844}{#30845}{#30846}{#30847}{#30848}{#30849}{#30850}{#30851}{#30852}{#30853}{#30854}{#30855}{#30856}{#30857}{#30858}{#30859}{#30860}{#30861}{#30862}{#30863}{#30864}{#30865}{#30866}{#30867}{#30868}{#30869}{#30870}{#30871}{#30872}{#30873}{#30874}{#30875}{#30876}{#30877}{#30878}{#30879}{#30880}{#30881}{#30882}{#30883}{#30884}{#30885}{#30886}{#30887}{#30888}{#30889}{#30890}{#30891}{#30892}{#30893}{#30894}{#30895}{#30896}{#30897}{#30898}{#30899}{#308100}{#308101}{#308102}{#308103}{#308104}{#308105}{#308106}{#308107}{#308108}{#308109}{#308110}{#308111}{#308112}{#308113}{#308114}{#308115}{#308116}{#308117}{#308118}{#308119}{#308120}{#308121}{#308122}{#308123}{#308124}{#308125}{#308126}{#308127}{#308128}{#308129}{#308130}{#308131}{#308132}{#308133}{#308134}{#308135}{#308136}{#308137}{#308138}{#308139}{#308140}{#308141}{#308142}{#308143}{#308144}{#308145}{#308146}{#308147}{#308148}{#308149}{#308150}{#308151}{#308152}{#308153}{#308154}{#308155}{#308156}{#308157}{#308158}{#308159}{#308160}{#308161}{#308162}{#308163}{#308164}{#308165}{#308166}{#308167}{#308168}{#308169}{#308170}{#308171}{#308172}{#308173}{#308174}{#308175}{#308176}{#308177}{#308178}{#308179}{#308180}{#308181}{#308182}{#308183}{#308184}{#308185}{#308186}{#308187}{#308188}{#308189}{#308190}{#308191}{#308192}{#308193}{#308194}{#308195}{#308196}{#308197}{#308198}{#308199}{#3081910}{#3081911}{#3081912}{#3081913}{#3081914}{#3081915}{#3081916}{#3081917}{#3081918}{#3081919}{#3081920}{#3081921}{#3081922}{#3081923}{#3081924}{#3081925}{#3081926}{#3081927}{#3081928}{#3081929}{#30819210}{#30819211}{#30819212}{#30819213}{#30819214}{#30819215}{#30819216}{#30819217}{#30819218}{#30819219}{#30819220}{#30819221}{#30819222}{#30819223}{#30819224}{#30819225}{#30819226}{#30819227}{#30819228}{#30819229}{#30819230}{#30819231}{#30819232}{#30819233}{#30819234}{#30819235}{#30819236}{#30819237}{#30819238}{#30819239}{#308192310}{#308192311}{#308192312}{#308192313}{#308192314}{#308192315}{#308192316}{#308192317}{#308192318}{#308192319}{#308192320}{#308192321}{#308192322}{#308192323}{#308192324}{#308192325}{#308192326}{#308192327}{#308192328}{#308192329}{#308192330}{#308192331}{#308192332}{#308192333}{#308192334}{#308192335}{#308192336}{#308192337}{#308192338}{#308192339}{#308192340}{#308192341}{#308192342}{#308192343}{#308192344}{#308192345}{#308192346}{#308192347}{#308192348}{#308192349}{#308192350}{#308192351}{#308192352}{#308192353}{#308192354}{#308192355}{#308192356}{#308192357}{#308192358}{#308192359}{#308192360}{#308192361}{#308192362}{#308192363}{#308192364}{#308192365}{#308192366}{#308192367}{#308192368}{#308192369}{#308192370}{#308192371}{#308192372}{#308192373}{#308192374}{#308192375}{#308192376}{#308192377}{#308192378}{#308192379}{#308192380}{#308192381}{#308192382}{#308192383}{#308192384}{#308192385}{#308192386}{#308192387}{#308192388}{#308192389}{#308192390}{#308192391}{#308192392}{#308192393}{#308192394}{#308192395}{#308192396}{#308192397}{#308192398}{#308192399}{#3081923100}{#3081923101}{#3081923102}{#3081923103}{#3081923104}{#3081923105}{#3081923106}{#3081923107}{#3081923108}{#3081923109}{#3081923110}{#3081923111}{#3081923112}{#3081923113}{#3081923114}{#3081923115}{#3081923116}{#3081923117}{#3081923118}{#3081923119}{#3081923120}{#3081923121}{#3081923122}{#3081923123}{#3081923124}{#3081923125}{#3081923126}{#3081923127}{#3081923128}{#3081923129}{#3081923130}{#3081923131}{#3081923132}{#3081923133}{#3081923134}{#3081923135}{#3081923136}{#3081923137}{#3081923138}{#3081923139}{#3081923140}{#3081923141}{#3081923142}{#3081923143}{#3081923144}{#3081923145}{#3081923146}{#3081923147}{#3081923148}{#3081923149}{#3081923150}{#3081923151}{#3081923152}{#3081923153}{#3081923154}{#3081923155}{#3081923156}{#3081923157}{#3081923158}{#3081923159}{#3081923160}{#3081923161}{#3081923162}{#3081923163}{#3081923164}{#3081923165}{#3081923166}{#3081923167}{#3081923168}{#3081923169}{#3081923170}{#3081923171}{#3081923172}{#3081923173}{#3081923174}{#3081923175}{#3081923176}{#3081923177}{#3081923178}{#3081923179}{#3081923180}{#3081923181}{#3081923182}{#3081923183}{#3081923184}{#3081923185}{#3081923186}{#3081923187}{#3081923188}{#3081923189}{#3081923190}{#3081923191}{#3081923192}{#3081923193}{#3081923194}{#3081923195}{#3081923196}{#3081923197}{#3081923198}{#3081923199}{#30819231100}{#30819231111}{#30819231122}{#30819231133}{#30819231144}{#30819231155}{#30819231166}{#30819231177}{#30819231188}{#30819231199}{#30819231200}{#30819231211}{#30819231222}{#30819231233}{#30819231244}{#30819231255}{#30819231266}{#30819231277}{#30819231288}{#30819231299}{#30819231300}{#30819231311}{#30819231322}{#30819231333}{#30819231344}{#30819231355}{#30819231366}{#30819231377}{#30819231388}{#30819231399}{#30819231400}{#30819231411}{#30819231422}{#30819231433}{#30819231444}{#30819231455}{#30819231466}{#30819231477}{#30819231488}{#30819231499}{#30819231500}{#30819231511}{#30819231522}{#30819231533}{#30819231544}{#30819231555}{#30819231566}{#30819231577}{#30819231588}{#30819231599}{#30819231600}{#30819231611}{#30819231622}{#30819231633}{#30819231644}{#30819231655}{#30819231666}{#30819231677}{#30819231688}{#30819231699}{#30819231700}{#30819231711}{#30819231722}{#30819231733}{#30819231744}{#30819231755}{#30819231766}{#30819231777}{#30819231788}{#30819231799}{#30819231800}{#30819231811}{#30819231822}{#30819231833}{#30819231844}{#30819231855}{#30819231866}{#30819231877}{#30819231888}{#30819231899}{#30819231900}{#30819231911}{#30819231922}{#30819231933}{#30819231944}{#30819231955}{#30819231966}{#30819231977}{#30819231988}{#30819231999}{#30819231101}{#30819231112}{#30819231123}{#30819231134}{#30819231145}{#30819231156}{#30819231167}{#30819231178}{#30819231189}{#30819231190}{#308192311101}{#308192311112}{#308192311123}{#308192311134}{#308192311145}{#308192311156}{#308192311167}{#308192311178}{#308192311189}{#308192311190}{#3081923111101}{#3081923111112}{#3081923111123}{#3081923111134}{#3081923111145}{#3081923111156}{#3081923111167}{#3081923111178}{#3081923111189}{#3081923111190}{#30819231111101}{#30819231111112}{#30819231111123}{#30819231111134}{#30819231111145}{#30819231111156}{#30819231111167}{#30819231111178}{#30819231111189}{#30819231111190}{#308192311111101}{#308192311111112}{#308192311111123}{#308192311111134}{#308192311111145}{#308192311111156}{#308192311111167}{#308192311111178}{#308192311111189}{#308192311111190}{#3081923111111101}{#3081923111111112}{#3081923111111123}{#3081923111111134}{#3081923111111145}{#3081923111111156}{#3081923111111167}{#3081923111111178}{#3081923111111189}{#3081923111111190}{#30819231111111101}{#30819231111111112}{#30819231111111123}{#30819231111111134}{#30819231111111145}{#30819231111111156}{#30819231111111167}{#30819231111111178}{#30819231111111189}{#30819231111111190}{#308192311111111101}{#308192311111111112}{#308192311111111123}{#308192311111111134}{#308192311111111145}{#308192311111111156}{#308192311111111167}{#308192311111111178}{#308192311111111189}{#308192311111111190}{#3081923111111111101}{#3081923111111111112}{#3081923111111111123}{#3081923111111111134}{#3081923111111111145}{#3081923111111111156}{#3081923111111111167}{#3081923111111111178}{#3081923111111111189}{#3081923111111111190}{#30819231111111111101}{#30819231111111111112}{#30819231111111111123}{#30819231111111111134}{#30819231111111111145}{#30819231111111111156}{#30819231111111111167}{#30819231111111111178}{#30819231111111111189}{#30819231111111111190}{#308192311111111111101}{#308192311111111111112}{#308192311111111111123}{#308192311111111111134}{#308192311111111111145}{#308192311111111111156}{#308192311111111111167}{#308192311111111111178}{#308192311111111111189}{#308192311111111111190}{#3081923111111111111101}{#3081923111111111111112}{#3081923111111111111123}{#3081923111111111111134}{#3081923111111111111145}{#3081923111111111111156}{#3081923111111111111167}{#3081923111111111111178}{#3081923111111111111189}{#3081923111111111111190}{#30819231111111111111101}{#30819231111111111111112}{#30819231111111111111123}{#30819231111111111111134}{#30819231111111111111145}{#30819231111111111111156}{#30819231111111111111167}{#30819231111111111111178}{#30819231111111111111189}{#30819231111111111111190}{#308192311111111111111101}{#308192311111111111111112}{#308192311111111111111123}{#308192311111111111111134}{#308192311111111111111145}{#308192311111111111111156}{#308192311111111111111167}{#308192311111111111111178}{#308192311111111111111189}{#308192311111111111111190}{#3081923111111111111111101}{#3081923111111111111111112}{#3081923111111111111111123}{#3081923111111111111111134}{#3081923111111111111111145}{#3081923111111111111111156}{#3081923111111111111111167}{#3081923111111111111111178}{#3081923111111111111111189}{#3081923111111111111111190}{#30819231111111111111111101}{#30819231111111111111111112}{#30819231111111111111111123}{#30819231111111111111111134}{#30819231111111111111111145}{#30819231111111111111111156}{#30819231111111111111111167}{#30819231111111111111111178}{#30819231111111111111111189}{#30819231111111111111111190}{#308192311111111111111111101}{#308192311111111111111111112}{#308192311111111111111111123}{#308192311
```

```

5664     \stex_invoke_sequence:
5665         \prg_return_true:\prg_return_false:
5666     }\prg_return_false:
5667 }\prg_return_false:
5668 }
5669
5670 \prg_new_conditional:Nnn \__stex_expr_is_seqmap:n {TF} {
5671     \int_compare:nNnTF {\tl_count:n{#1}} = 3 {
5672         \exp_args:Ne \tl_if_eq:nnTF {\tl_head:n{#1}} {\seqmap}
5673         \prg_return_true:\prg_return_false:
5674     }\prg_return_false:
5675 }

Sequence variable:

5676 % 1: name 2: arity 3: clist 4: argnum 5: argmode 6: precedence 7: argname
5677 \cs_new_protected:Nn \__stex_expr_assoc_seq:nnnnnnn {
5678     \group_begin:
5679         \seq_clear:N \l_stex_aB_args_seq
5680         \__stex_expr_assoc_make_seq:nnn{#1}{#3}{#2}
5681     \exp_args:NNe \use:nn \group_end:
5682         \seq_put_right:Nn \exp_not:N \l_stex_aB_args_seq {
5683             \_stex_is_sequentialized:n{
5684                 \_stex_term_arg:nnnnn{#4}\int_use:N\l_stex_expr_count_int}{#5}{#6}{#7}{#8}
5685                 \bool_set_true:N \l_stex_allow_semantic_bool
5686                 \str_set:Nn \exp_not:N \l_stex_current_symbol_str
5687                     {\l_stex_current_symbol_str}
5688                 \tl_if_empty:NF \l_stex_current_term_tl {
5689                     \tl_set:Nn \exp_not:N \l_stex_current_term_tl {
5690                         \exp_args:No \exp_not:n \l_stex_current_term_tl
5691                     }
5692                 }
5693                 \stex_annotation:nn{
5694                     data-shtml-term=OMV,
5695                     data-shtml-head={#1},
5696                     data-shtml-notationid={}
5697                 }{
5698                     \_stex_annotation_force_break:n{
5699                         \_stex_term_do_aB_clist:
5700                     }
5701                 }
5702             }
5703         }
5704     }
5705 }
5706 }

5707 % #1: name, #2: clist, #3:arity
5708 \cs_new_protected:Nn \__stex_expr_assoc_make_seq:nnn {
5709     \cs_if_exist:cTF{\l_stex_notation_#1__cs} {
5710         \cs_set_eq:Nc \l_stex_expr_cs {\l_stex_notation_#1__cs}
5711     }{
5712         \stex_do_default_notation:
5713         \cs_set_eq:NN \l_stex_expr_cs \l_stex_default_notation
5714     }
5715 }
```

```

5716 \clist_map_inline:nn{#2}{
5717   \tl_if_eq:nnTF{##1}{\ellipses}{
5718     \seq_put_right:Nn \l_stex_aB_args_seq { ##1 }
5719   }{
5720     \int_compare:nNnTF {#3} = 1 {
5721       \tl_set:Nn \l__stex_expr_iarg_tl { ##1 }
5722     }{
5723       \tl_set:Nn \l__stex_expr_iarg_tl { ##1 }
5724     }
5725     \seq_put_right:Nx \l_stex_aB_args_seq {
5726       \group_begin:
5727       \exp_not:n {
5728         \tl_set_eq:NN \stex_term_do_aB_clist: \__stex_expr_do_aB_clist:
5729         \def\comp{\varcomp}
5730         \str_set:Nn \l_stex_current_symbol_str{#1}
5731       }
5732       \exp_after:wN \exp_after:wN \exp_after:wN \exp_not:n
5733       \exp_after:wN \exp_after:wN \exp_after:wN {
5734         \exp_after:wN \l__stex_expr_cs \exp_after:wN \group_end: \l__stex_expr_iarg_tl
5735       }
5736     }
5737   }
5738 }
5739 }

\seqmap:

5740 % 1: fun 2: clist 3: argnum 4: argmode 5: precedence 6: argname
5741 \cs_new_protected:Nn \__stex_expr_do_seqmap:nnnnnn {
5742   \group_begin:
5743     \cs_set:Npn \l_tmpa_cs {##1 {#1}}
5744     \seq_clear:N \l_stex_aB_args_seq
5745     \clist_map_inline:nn{#2} {
5746       \__stex_expr_is_varseq:nTF{##1} {
5747         \exp_after:wN
5748         \__stex_expr_varseq_in_map:nnnnnnnn {#1}
5749       }{
5750         \seq_put_right:Nn \l_stex_aB_args_seq {##1}
5751       }
5752     }
5753     \seq_clear:N \l__stex_expr_old_seq
5754     \seq_map_inline:Nn \l_stex_aB_args_seq {
5755       \tl_if_eq:nnTF{##1}{\ellipses} {
5756         \seq_put_right:Nn \l__stex_expr_old_seq {##1}
5757       }
5758       % TODO \stex_is_sequentialized:n
5759     {
5760       \exp_args:NNo \seq_put_right:Nn \l__stex_expr_old_seq {
5761         \l_tmpa_cs {##1}
5762       }
5763     }
5764   }
5765   \seq_set_eq:NN \l_stex_aB_args_seq \l__stex_expr_old_seq
5766 \exp_args:NNe \use:nn \group_end: {
5767   \seq_put_right:Nn \exp_not:N \l_stex_aB_args_seq {

```

```

5768     \_stex_is_sequentialized:n{
5769         \_stex_term_arg:nnnnn{#3}\int_use:N\l_stex_expr_count_int}{#4}{#5}{#6}){
5770             \bool_set_true:N \l_stex_allow_semantic_bool
5771             \str_set:Nn \exp_not:N \l_stex_current_symbol_str
5772                 {\l_stex_current_symbol_str}
5773             \tl_set:Nn \exp_not:N \l_stex_current_term_tl {
5774                 \symuse{Metatheory?sequence-map}
5775                 {\exp_args:No \exp_not:n \l_tmpa_tl}
5776                 {\_stex_term_do_aB_clist:}
5777             }
5778             \_stex_expr_do_headterm:nn{}{
5779                 \_stex_term_do_aB_clist:
5780             }
5781         }
5782     }
5783 }
5784 }
5785 }
5786
5787 \cs_new_protected:Nn \_stex_expr_varseq_in_map:nnnnnnn {
5788     \_stex_expr_assoc_make_seq:nnn{#2}{#6}{#3}
5789 }

```

(End of definition for `\_stex_term_arg_aB:nnnnn`. This function is documented on page ??.)

## Term HTML Annotations

```

\_stex_term_oms_or_omv:nnn
    \_stex_term_oms:nnn
    \_stex_term_omv:nnn
5790 \cs_new_protected:Nn \_stex_eat_exclamation_point: {
5791     \peek_charcode_remove:NT ! {
5792         \_stex_eat_exclamation_point:
5793     }
5794 }
5795
5796 \bool_new:N \stex_in_invisible_html_bool
5797 \bool_set_false:N \stex_in_invisible_html_bool
5798 \stex_if_html_backend:TF {
5799     % 1: variant 2: intent 3: code
5800     \cs_new_protected:Nn \_stex_term_oms:nnn {
5801         \tl_if_empty:NTF \l_stex_current_term_tl {
5802             \stex_annotation:nn{
5803                 data-shtml-term=OMID,
5804                 data-shtml-head={\l_stex_current_symbol_str},
5805                 data-shtml-notationid={#1},
5806             }{
5807                 \_stex_annotation_force_break:n{#3}
5808             }
5809         }{
5810             \_stex_expr_do_headterm:nn{#1}{#3}
5811         }
5812     }
5813     \cs_new_protected:Nn \_stex_term_omv:nnn {
5814         \tl_if_empty:NTF \l_stex_current_term_tl {
5815             \stex_annotation:nn{

```

```

5816     data-shtml-term=OMV,
5817     data-shtml-head={\l_stex_current_symbol_str},
5818     data-shtml-notationid={#1}
5819   }{
5820     \_stex_annotation_force_break:n{#3}
5821   }
5822   }{
5823     \_\_stex_expr_do_headterm:nn{#1}{#3}
5824   }
5825 }
5826 \cs_new_protected:Nn \_\_stex_expr_do_headterm:nn {
5827   \bool_if:NTF \stex_in_invisible_html_bool {
5828     {\bool_set_true:N \l_stex_allow_semantic_bool
5829       \ensuremath{\l_stex_current_term_tl}}
5830     }
5831   }{
5832     \stex_annotation:nn{
5833       data-shtml-term=complex,
5834       data-shtml-head={\l_stex_current_symbol_str},
5835       data-shtml-notationid={#1}
5836     }{
5837       \_stex_annotation_force_break:n{
5838         \stex_annotation_invisible:nn{data-shtml-headterm={}){
5839           {\bool_set_true:N \l_stex_allow_semantic_bool
5840             \ensuremath{\l_stex_current_term_tl}}
5841           }
5842         }
5843       }
5844       #2
5845     }
5846   }
5847 }
5848 }{
5849   \cs_new_protected:Nn \_stex_term_oms:nnn {#3}
5850   \cs_new_protected:Nn \_stex_term_omv:nnn {#3}
5851   \cs_new_protected:Nn \_\_stex_expr_do_headterm:nn { #2 }
5852 }
5853 \cs_set_eq:NN \_stex_term_oms_or_omv:nnn \_stex_term_oms:nnn

```

(End of definition for `\_stex_term_oms_or_omv:nnn`, `\_stex_term_oms:nnn`, and `\_stex_term_omv:nnn`.  
These functions are documented on page ??.)

```

\_stex_term_oma:nnn
5854 \stex_if_html_backend:TF {
5855   \cs_new_protected:Nn \_stex_term_oma:nnn {
5856     \tl_if_empty:NTF \l_stex_current_term_tl {
5857       \stex_annotation:nn{
5858         data-shtml-term=OMA,
5859         data-shtml-head={\l_stex_current_symbol_str},
5860         data-shtml-notationid={#1}
5861       }{
5862         \_stex_annotation_force_break:n{#3}
5863       }
5864     }

```

```

5865     \stex_annotate:nn{
5866         data-shtml-term=OMA,
5867         data-shtml-head={\l_stex_current_symbol_str},
5868         data-shtml-notationid={#1}
5869     }{
5870         \_stex_annotate_force_break:n{
5871             \stex_annotate_invisible:nn{data-shtml-headterm={}{{}
5872                 \bool_set_true:N \l_stex_allow_semantic_bool
5873                 \l_stex_current_term_tl
5874             }{}}
5875             #3
5876         }
5877     }
5878 }
5879 }
5880 }{
5881     \cs_new_protected:Nn \_stex_term_oma:nnn {#3}
5882 }

```

(End of definition for `\_stex_term_oma:nnn`. This function is documented on page ??.)

```

\_stex_term_omb:nnn
5883 \stex_if_html_backend:TF {
5884     \cs_new_protected:Nn \_stex_term_omb:nnn {
5885         \tl_if_empty:NTF \l_stex_current_term_tl {
5886             \stex_annotate:nn{
5887                 data-shtml-term=OMBIND,
5888                 data-shtml-head={\l_stex_current_symbol_str},
5889                 data-shtml-notationid={#1}
5890             }{
5891                 \_stex_annotate_force_break:n{#3}
5892             }
5893         }{
5894             \stex_annotate:nn{
5895                 data-shtml-term=OMBIND,
5896                 data-shtml-head={\l_stex_current_symbol_str},
5897                 data-shtml-notationid={#1}
5898             }{
5899                 \_stex_annotate_force_break:n{
5900                     \stex_annotate_invisible:nn{data-shtml-headterm={}{{}
5901                         \bool_set_true:N \l_stex_allow_semantic_bool
5902                         \l_stex_current_term_tl
5903                     }{}}
5904                     #3
5905                 }
5906             }
5907         }
5908     }{
5909     }{
5910         \cs_new_protected:Nn \_stex_term_omb:nnn { #3 }
5911     }

```

(End of definition for `\_stex_term_omb:nnn`. This function is documented on page ??.)

## Automated Bracketing

```
\infprec
\neginfprec 5912 \tl_const:Nx \infprec {\int_use:N \c_max_int}
5913 \tl_const:Nx \neginfprec {-\int_use:N \c_max_int}

(End of definition for \infprec and \neginfprec. These functions are documented on page 94.)

\dobrackets
5914 \int_new:N \l_stex_notation_downprec
5915 \int_set:Nn \l_stex_notation_downprec \infprec
5916 \tl_set:Nn \l_stex_expr_left_bracket_str (
5917 \tl_set:Nn \l_stex_expr_right_bracket_str )
5918 \bool_new:N \l_stex_brackets_dones_bool
5919
5920 \cs_new_protected:Nn \stex_maybe_brackets:nn {
5921     \bool_if:NTF \l_stex_brackets_dones_bool {
5922         \bool_set_false:N \l_stex_brackets_dones_bool
5923         #2
5924     } {
5925         \stex_debug:nn{brackets}{#1}\int_eval:n \l_stex_notation_downprec?
5926         \int_compare:nNnTF { #1 } > \l_stex_notation_downprec {
5927             \%bool_if:NTF \l_stex_inpararray_bool { #2 }{
5928                 \dobrackets {
5929                     #2
5930                 }
5931                 %}
5932             }{
5933                 #2
5934             }
5935         }
5936     }
5937
5938 \%RequirePackage{scalerel}
5939 \cs_new_protected:Npn \dobrackets #1 {
5940     \%ThisStyle{\if D\m@switch
5941     \% \exp_args:Nnx \use:nn
5942     \% { \exp_after:wN \left\l_ \stex_expr_left_bracket_str #1 }
5943     \% { \exp_not:N\right\r_ \stex_expr_right_bracket_str }
5944     \% \else
5945     \group_begin:
5946     \%stex_pseudogroup_with:nn{\l_stex_brackets_dones_bool\l_stex_notation_downprec} {
5947         \bool_set_true:N \l_stex_brackets_dones_bool
5948         \%int_set:Nn \l_stex_notation_downprec \infprec
5949         \mathopen{\cs_if_exist:NT\l_stex_current_symbol_str\comp
5950             \l_stex_expr_left_bracket_str
5951         }
5952         #1
5953     \group_end: \%}
5954     \mathclose{\cs_if_exist:NT\l_stex_current_symbol_str\comp
5955         \l_stex_expr_right_bracket_str
5956     }
5957 \%fi}
5958 }
```

(End of definition for \dobrackets. This function is documented on page ??.)

### \withbrackets

```
5959 \cs_new_protected:Npn \withbrackets #1 #2 #3 {
5960   \stex_pseudogroup_with:nn{\l_stex_expr_left_bracket_str\l_stex_expr_right_bracket_str}{%
5961     \tl_set:Nn \l_stex_expr_left_bracket_str { #1 }
5962     \tl_set:Nn \l_stex_expr_right_bracket_str { #2 }
5963     #3
5964   }
5965 }
```

(End of definition for \withbrackets. This function is documented on page 95.)

### \dowithbrackets

```
5966 \cs_new_protected:Npn \dowithbrackets #1 #2 #3 {
5967   \withbrackets{#1}{#2}{\dobrackets{#3}}
5968 }
```

(End of definition for \dowithbrackets. This function is documented on page ??.)

## Symname and Variants

```
\symname
  \sn 5969 \def\maincomp{\comp}
  \sns 5970
\Symname 5971 \stex_keys_define:nnnn{symname}%
  \Sn 5972   \tl_clear:N \l_stex_key_pre_tl
  \Sns 5973   \tl_clear:N \l_stex_key_post_tl
\symref 5974   \% \tl_clear:N \l_stex_key_proot_tl
  \sr 5975 }%
\varref 5976   pre .tl_set:N = \l_stex_key_pre_tl ,
\varname 5977   post .tl_set:N = \l_stex_key_post_tl ,
  5978   root .code:n = {}%.tl_set:N = \l_stex_key_root_tl
\Varname 5979 }{}%
5980
5981 \NewDocumentCommand \symref { O{} m m } {
5982   \group_begin:
5983   \stex_keys_set:nn{symname}{#1}
5984   \stex_get_symbol:n{#2}
5985   \stex_do_ref:nNn{#3}\symrefemph@uri\_stex_term_oms:nnn
5986 }
5987 \let\sr\symref
5988
5989 \NewDocumentCommand \symname { O{} m } {
5990   \group_begin:
5991   \stex_keys_set:nn{symname}{#1}
5992   \stex_get_symbol:n{#2}
5993   \stex_do_ref:nNn{
5994     \l_stex_key_pre_tl\l_stex_get_symbol_name_str\l_stex_key_post_tl
5995   }\symrefemph@uri\_stex_term_oms:nnn
5996 }
5997 \let\sn\symname
5998 \protected\def\sns{\symname[post=s]}
5999
```

```

6000 \NewDocumentCommand \Symname { O{} m} {
6001   \group_begin:
6002   \stex_keys_set:nn{symname}{#1}
6003   \stex_get_symbol:n{#2}
6004   \stex_do_ref:nNn{
6005     \l_stex_key_pre_tl\exp_after:wN\_stex_capitalize:n\l_stex_get_symbol_name_str\l_stex_key
6006   }\symrefemph@uri\stex_term_oms:nnn
6007 }
6008 \cs_new_protected:Nn \stex_capitalize:n {
6009   \uppercase{#1}
6010 }
6011 \let\Sn\Symname
6012 \protected\def\Sns{\Symname[post=s]}
6013
6014 \cs_new:Npn \stex_split_slash: #1/#2/#3\_stex_args_end: {
6015   \tl_if_empty:nTF{#3}{
6016     #2
6017   }{
6018     \stex_split_slash: #2 / #3 \stex_args_end:
6019   }
6020 }
6021
6022 \NewDocumentCommand \varref { O{} m m} {
6023   \group_begin:
6024   \stex_keys_set:nn{symname}{#1}
6025   \stex_get_var:n{#2}
6026   \stex_do_ref:nNn{#3}\varempath@uri{
6027     \str_set_eq:NN \l_stex_current_symbol_str\l_stex_get_symbol_name_str
6028     \def\comp{\_varcomp}
6029     \stex_term_omv:nnn
6030   }
6031 }
6032
6033 \NewDocumentCommand \varname { O{} m} {
6034   \group_begin:
6035   \stex_keys_set:nn{symname}{#1}
6036   \stex_get_var:n{#2}
6037   \stex_do_ref:nNn{
6038     \l_stex_key_pre_tl\l_stex_get_symbol_name_str\l_stex_key_post_tl
6039   }\varempath@uri{
6040     \str_set_eq:NN \l_stex_current_symbol_str\l_stex_get_symbol_name_str
6041     \def\comp{\_varcomp}
6042     \stex_term_omv:nnn
6043   }
6044 }
6045
6046 \NewDocumentCommand \Varname { O{} m} {
6047   \group_begin:
6048   \stex_keys_set:nn{symname}{#1}
6049   \stex_get_var:n{#2}
6050   \stex_do_ref:nNn{
6051     \l_stex_key_pre_tl\exp_after:wN\_stex_capitalize:n\l_stex_get_symbol_name_str\l_stex_key
6052   }\varempath@uri{
6053     \str_set_eq:NN \l_stex_current_symbol_str\l_stex_get_symbol_name_str

```

```

6054     \def\comp{\_varcomp}
6055     \_stex_term_omv:nnn
6056 }
6057 }
6058
6059
6060 \cs_new_protected:Nn \_stex_do_ref:nNn {
6061   \stex_if_html_backend:T{ \relax \ifvmode \indent \fi }
6062   \bool_if:NTF \l_stex_allow_semantic_bool{
6063     \str_set:Nx \l_stex_current_symbol_str
6064       {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
6065     \str_if_in:NnT \l_stex_get_symbol_name_str / {
6066       \str_set:Nx \l_stex_get_symbol_name_str {
6067         \exp_after:wN \_stex_split_slash: \l_stex_get_symbol_name_str
6068         /\_stex_args_end:
6069       }
6070     }
6071     \tl_clear:N \l_stex_current_term_tl
6072     \def\comp{\_comp}
6073     \let\compemph@uri#2
6074     #3{}{}{\comp{#1}}
6075   }{
6076     \msg_error:nnxx{stex}{error/notallowed}{#1}{\l_stex_current_symbol_str}
6077   }
6078   \group_end:
6079 }

```

(End of definition for \symname and others. These functions are documented on page 92.

## Highlighting

```
6080 〈@=stex_notationcomps〉

  \comp
\compemph@uri
  \compemph
  \defemph
\defemph@uri
  \symrefemph
\symrefemph@uri
  \varempm
\varempm@uri
  \varempm
\cs_new_protected:Nn \_do_comp:nNn {
  \stex_pseudogroup_with:nn{\comp}{%
    \def\comp##1{##1}
    \str_if_empty:NTF \l_stex_current_symbol_str {
      #3
    }{
      \stex_if_html_backend:TF {
        \stex_annotation:nn { data-shtml-#1 = \l_stex_current_symbol_str}{ #3 }
      }{
        \exp_args:Nno #2 { #3 } \l_stex_current_symbol_str
      }
    }
  }
\cs_new_protected:Npn \_comp {
  \_do_comp:nNn {comp}\compemph@uri
}
\cs_new_protected:Npn \_varcomp {
  \_do_comp:nNn {varcomp}\varempm@uri
```

```

6102 }
6103 \cs_new_protected:Npn \_defcomp {
6104     \do_comp:nNn {definiendum}\defemph@uri
6105 }
6106 }
6107
6108 \cs_set_protected:Npn \comp {}
6109
6110 \cs_new_protected:Npn \compemph@uri #1 #2 {
6111     \compemph{#1 }
6112 }
6113
6114 \cs_new_protected:Npn \compemph #1 {
6115     #1
6116 }
6117
6118 \cs_new_protected:Npn \defemph@uri #1 #2 {
6119     \defemph{#1}
6120 }
6121
6122 \cs_new_protected:Npn \defemph #1 {
6123     \ifmmode\else\expandafter\textbf\fi{#1}
6124 }
6125
6126 \cs_new_protected:Npn \symrefemph@uri #1 #2 {
6127     \symrefemph{#1}
6128 }
6129
6130 \cs_new_protected:Npn \symrefemph #1 {
6131     \emph{#1}
6132 }
6133
6134 \cs_new_protected:Npn \varemph@uri #1 #2 {
6135     \varemph{#1}
6136 }
6137
6138 \cs_new_protected:Npn \varemph #1 {
6139     #1
6140 }

```

(End of definition for `\comp` and others. These functions are documented on page 93.)

## 13.9 Mathematical Structures

```

6141 <@@=stex_structures>
6142 \this
6143 \cs_new_protected:Npn \stex_current_this: {
6144     \bool_set_true:N \l_stex_allow_semantic_bool
6145     \tl_if_empty:NTF \l_stex_current_this_tl {{}}{
6146         \str_set:Nx \l_stex_current_symbol_str {
6147             \l_stex_current_module_str ? \l__stex_structures_name_str
6148         }
6149         \maincomp{\l_stex_current_this_tl}

```

```

6149     }
6150   }
6151 }
6152 \let \this \stex_current_this:
(End of definition for \this. This function is documented on page 99.)

mathstructure (env.)
6153 \stex_new_stylable_env:nnnnnnn {mathstructure}{m 0{}}
6154   \__stex_structures_begin:nn{#1}{#2}
6155   \stex_smsmode_do:
6156 }{
6157   \stex_structural_feature_module_end:
6158   \__stex_structures_do_externals:
6159 }{}{}{}
6160
6161 \stex_keys_define:nnnn{mathstructure}{
6162   \tl_clear:N \l_stex_current_this_tl
6163   \str_clear:N \l__stex_structures_name_str
6164 }{
6165   this .tl_set:N = \l_stex_current_this_tl ,
6166   unknown .code:n = {
6167     \str_if_empty:NTF \l_keys_key_str {
6168       \str_set:Nx \l__stex_structures_name_str {\l_keys_key_tl}
6169     }{
6170       \str_set_eq:NN \l__stex_structures_name_str \l_keys_key_str
6171     }
6172   }
6173 }{}}
6174
6175 \cs_new_protected:Nn \__stex_structures_begin:nn {
6176   \stex_keys_set:nn {mathstructure}{#2}
6177   \str_if_empty:NT \l__stex_structures_name_str {
6178     \str_set:Nn \l__stex_structures_name_str {#1}
6179   }
6180   \def\comp{\_comp}
6181
6182 \exp_args:Nne \use:nn { \stex_module_add_symbol:nnnnnnnN }
6183   { [#1]{\l__stex_structures_name_str}{0}{}{defed}{}
6184     \l_stex_current_module_str / \l__stex_structures_name_str-module
6185   }
6186   {} \stex_invoke_structure:
6187   \str_set:Nx \l_stex_mroname_str {#1}
6188   \stex_execute_in_module:x{
6189     \seq_clear:c{\l_stex_structure_macros_\l_stex_current_module_str / \l__stex_structures_na
6190     \seq_put_right:cn{\l_stex_structure_macros_\l_stex_current_module_str / \l__stex_structur
6191   }
6192   \exp_args:No \stex_structural_feature_module:nn
6193     {\l__stex_structures_name_str}{structure}
6194 }
6195
6196 \stex_sms_allow_env:n{mathstructure}
6197 \stex_deactivate_macro:Nn \mathstructure {module-environments}
6198 \stex_every_module:n {\stex_reactivate_macro:N \mathstructure}

```

```

6199 \cs_new_protected:Nn \__stex_structures_do_externals: {
6200   \tl_set:Nn \l__stex_structures_replace_this_tl {####1}
6201   \exp_args:No \stex_iterate_symbols:n{g_stex_last_feature_str} {
6202     \__stex_structures_external_decl:nnnn{##5}{##4}{##3}{##8}
6203   }
6204 }
6205 }
6206
6207 \cs_new_protected:Nn \__stex_structures_external_decl:nnnn {
6208   \%stex_debug:nn{structure}{%
6209     % Generating-external-declaration-\l__stex_structures_name_str/#3-in-
6210     % \l_stex_current_module_str^~J
6211     % \tl_to_str:n{#1}^~J\tl_to_str:n{#2}^~J\tl_to_str:n{#4}
6212   }%
6213   \%tl_set:Nn \l_stex_get_symbol_args_tl {#1}
6214   \%exp_args:Nnx \use:nn {\stex_module_add_symbol:nnnnnnN} {
6215     {}{\l__stex_structures_name_str/#3}\int_eval:n{#2 + 1}}
6216     {ii\tl_if_empty:nF{#1}{\stex_map_args:N \__stex_structures_shift_argls:nn}}
6217     {defed}{typed}
6218   }{#4}\stex_invoke_outer_field:
6219 }
6220
6221 \cs_new:Nn \__stex_structures_shift_argls:nn {
6222   \int_eval:n{#1+1}#2
6223 }

```

\stex\_get\_mathstructure:n

```

6224 \cs_new_protected:Nn \stex_get_mathstructure:n {
6225   \stex_get_mathstructure:n{#1}
6226   \str_if_empty:NT \l_stex_get_structure_module_str {
6227     \msg_error:nnn{stex}{error/unknownstructure}{#1}
6228   }
6229 }
6230 \cs_new_protected:Nn \__stex_get_mathstructure:n {
6231   \str_clear:N \l_stex_get_structure_module_str
6232   \stex_get_symbol:n{#1}
6233   \str_if_empty:NF \l_stex_get_symbol_name_str {
6234     \exp_args:No \tl_if_eq:NNT \l_stex_get_symbol_invoke_cs \stex_invoke_structure: {
6235       \str_set_eq:NN \l_stex_get_structure_module_str \l_stex_get_symbol_type_tl
6236     }
6237   }
6238 }

```

(End of definition for \stex\_get\_mathstructure:n. This function is documented on page ??.)

extstructure (env.)

```

6239 \stex_new_stylable_env:nnnnnn {extstructure}{m 0{} m}{%
6240   \seq_clear:N \l__stex_structures_imports_seq
6241   \clist_map_inline:nn{#3}{%
6242     \stex_get_mathstructure:n{##1}
6243     \clist_map_inline:Nn \l_stex_get_symbol_type_tl {%
6244       \exp_args:Ne \stex_if_module_exists:nT{\tl_to_str:n{####1}}{%
6245         \seq_put_right:Nn \l__stex_structures_imports_seq{####1}
6246       }
6247     }
6248   }
6249 }

```

```

6247 }
6248 \clist_set:Nn \l_tmpa_clist \l_stex_get_structure_module_str
6249 \clist_reverse:N \l_tmpa_clist
6250 \clist_pop:NN \l_tmpa_clist \l_stex_get_structure_module_str
6251 \stex_execute_in_module:x{
6252     \seq_put_right:cn{\l_stex_structure_macros_\l_stex_get_structure_module_str _seq}{\#1}
6253 }
6254 }
6255 \__stex_structures_begin:nn{\#1}{\#2}
6256 \seq_map_inline:Nn\l__stex_structures_imports_seq{
6257     \stex_if_do_html:T {
6258         \hbox{\stex_annotation_invisible:nn
6259             {data-shtml-import=\#\#1} {}}
6260     }
6261     \stex_module_add_morphism:nonn
6262         {\#1\{import\}{}}
6263     \stex_execute_in_module:x{
6264         \stex_activate_module:n{\#1}
6265     }
6266 }
6267 \stex_smsmode_do:
6268 }{
6269     \stex_structural_feature_module_end:
6270     \__stex_structures_do_externals:
6271 }{}{}{}
6272
6273 \stex_sms_allow_env:n{extstructure}
6274 \stex_deactivate_macro:Nn \extstructure {module-environments}
6275 \stex_every_module:n {
6276     \stex_reactivate_macro:N \extstructure
6277 }
6278
6279 \cs_new:Nn \__stex_structures_extend_structure_i:NnnnnnnnN {
6280     \exp_not:n{\#1\{#2\}{#3\{#4\{#5\{defed\}}\}\{l__stex_structures_extmod_str,\#7\}\exp_not:n{\{#8\}\#9}}
6281 }
6282 \cs_new_protected:Nn \__stex_structures_extend_structure:nn {
6283     \stex_debug:nn{ext}{Extending~#1~by~#2}
6284     \str_set:Nn \l__stex_structures_extmod_str{\#2}
6285     \tl_set:cx{\#1} {
6286         \exp_after:wN \exp_after:wN \exp_after:wN
6287         \__stex_structures_extend_structure_i:NnnnnnnnN \cs:w #1 \cs_end:
6288     }
6289 }
6290
6291 \stex_new_stylable_env:nnnnnnn {extstructure*}{m} {
6292     \__stex_structures_new_extstruct_name:
6293     \seq_clear:N \l__stex_structures_imports_seq
6294     \stex_get_mathstructure:n{\#1}
6295
6296     \clist_map_inline:Nn \l_stex_get_symbol_type_tl {
6297         \exp_args:Ne \stex_if_module_exists:nT{\tl_to_str:n{\#1}}{
6298             \seq_put_right:Nn \l__stex_structures_imports_seq{\#1}
6299         }
6300 }

```

```

6301  \clist_set:No \l_tmpa_clist \l_stex_get_structure_module_str
6302  \clist_reverse:N \l_tmpa_clist
6303  \clist_pop:NN \l_tmpa_clist \l_stex_get_structure_module_str
6304
6305
6306  \stex_execute_in_module:x{
6307    \seq_map_inline:cn{\l_stex_structure_macros_\l_stex_get_structure_module_str _seq} {
6308      \exp_not:N \tl_if_exist:cT{###1} {
6309        \__stex_structures_extend_structure:nn{###1}{\l_stex_current_module_str/\l_stex_st
6310      }
6311    }
6312  }
6313
6314  \exp_args:No \__stex_structures_begin:nn\l_stex_structures_exstruct_name_str{}
6315
6316  \seq_map_inline:Nn \l_stex_structures_imports_seq {
6317    \stex_if_do_html:T {
6318      \stex_annotation_invisible:nn
6319      {data-shtml-import= {##1}} {}
6320    }
6321    \stex_module_add_morphism:nonn
6322    {}{##1}{import} {}
6323    \stex_execute_in_module:x{
6324      \stex_activate_module:n{##1}
6325    }
6326  }
6327
6328  \stex_smsmode_do:
6329 }{
6330  \prop_map_inline:cn{
6331    c_stex_module_ \l_stex_current_module_str _symbols_prop
6332  }{
6333    \__stex_structures_check_def:nnnnnnnn ##2
6334  }
6335  \stex_structural_feature_module_end:
6336  \__stex_structures_do_externals:
6337 }{}{}{}
6338
6339 \stex_sms_allow_env:n{extstructure*}
6340 \exp_after:wN \stex_deactivate_macro:Nn
6341   \cs:w extstructure*\cs_end: {module-environments}
6342 \stex_every_module:n {
6343   \exp_after:wN \stex_reactivate_macro:N \cs:w extstructure*\cs_end:
6344 }
6345
6346 \cs_new_protected:Nn \__stex_structures_check_def:nnnnnnnn {
6347   \tl_if_empty:nT{#5} {
6348     \msg_error:nnnn{stex}{error/needsdefiniens}{#2}{extstructure*}
6349   }
6350 }
6351
6352 \stex_every_module:n{ \str_set:Nn \l_stex_structures_extname_count 0}
6353
6354 \cs_new_protected:Nn \__stex_structures_new_extstruct_name: {

```

```

6355   \stex_do_up_to_module:n {
6356     \str_set:Nx \l__stex_structures_extname_count {\int_eval:n{\l__stex_structures_extname_c
6357   }
6358   \str_set:Nx \l__stex_structures_exstruct_name_str {EXTSTRUCT_\l__stex_structures_extname_c
6359 }
6360

    Invoking structures:

6361 \cs_new_protected:Nn \stex_invoke_structure: {
6362   \tl_set:Nn \l__stex_structures_set_comp_tl {\__stex_structures_set_thiscomp:}
6363   \__stex_structures_invoke_top:n {}
6364 }

6365
6366 \cs_new_protected:Nn \__stex_structures_invoke_top:n {
6367   \stex_debug:nn{structure} {
6368     invoking~structure~{\l_stex_current_type_tl}<\tl_to_str:n{#1}>
6369   }
6370   \peek_charcode:NTF [ {
6371     \__stex_structures_merge:nw{#1}
6372   }{
6373     \__stex_structures_invocation_type:n {#1}
6374     \tl_set:Nn \l__stex_structures_this_tl {}
6375     \peek_charcode_remove:NTF ! {
6376       \peek_charcode:NTF [ {
6377         \__stex_structures_maybe_notation:w
6378       }{
6379         \__stex_structures_maybe_notation:w []
6380       }
6381     }{
6382       \__stex_structures_invoke_this:n
6383     }
6384   }
6385 }

6386
6387 \cs_new_protected:Npn \__stex_structures_merge:nw #1 [ #2 ] {
6388   \exp_args:Ne \stex_if_starts_with:nnTF {\tl_to_str:n{#2}}{comp} {
6389     \__stex_structures_set_customcomp: #2 \__stex_structures_end:
6390     \__stex_structures_invoke_top:n{#1}
6391   }{
6392     \exp_args:Ne \stex_if_starts_with:nnTF {\tl_to_str:n{#2}}{this} {
6393       \__stex_structures_set_thisnotation: #2 \__stex_structures_end:
6394       \__stex_structures_invoke_top:n{#1}
6395     }{
6396       \tl_if_empty:nTF{#1} {
6397         \__stex_structures_invoke_top:n{#2}
6398       }{
6399         \tl_if_empty:nTF{#2} {
6400           \__stex_structures_invoke_top:n{#1}
6401         }{
6402           \__stex_structures_invoke_top:n{#1,#2}
6403         }
6404       }
6405     }
6406   }

```

```

6407 }
6408
6409 \cs_new_protected:Npn \__stex_structures_set_thisnotation: this= #1 \__stex_structures_end:
6410   \tl_set:Nn \l_stex_return_notation_tl { \comp{#1} }
6411   \tl_set:Nn \l__stex_structures_set_comp_tl {}
6412 }
6413
6414 \cs_new_protected:Npn \__stex_structures_set_customcomp: comp= #1 \__stex_structures_end: {
6415   \tl_set:Nn \l__stex_structures_set_comp_tl {
6416     \__stex_structures_set_custom_comp:n{#1}
6417   }
6418   \tl_set:Nn \l_stex_return_notation_tl { \comp{} }
6419 }

```

The structure type:

```

6420 \cs_new_protected:Nn \__stex_structures_invokation_type:n {
6421   \__stex_structures_do_assign_list:n{#1}
6422   \clist_if_empty:NTF \l__stex_structures_fields_clist {
6423     \int_compare:nNnTF {\clist_count:N \l_stex_current_type_tl}
6424       = 1 {
6425         \tl_set:Nx \l_stex_structures_current_type_tl {
6426           \exp_args:No \exp_not:n \l_stex_current_redo_tl
6427           \stex_term_oms_or_omv:nnn{}{}{}
6428         }
6429       }{
6430         \exp_args:No \__stex_structures_make_type:n \l_stex_current_type_tl
6431       }
6432     }{
6433       \int_compare:nNnTF {\clist_count:N \l_stex_current_type_tl}
6434         = 1 {
6435           \__stex_structures_make_type:n {}
6436         }{
6437           \exp_args:No \__stex_structures_make_type:n \l_stex_current_type_tl
6438         }
6439     }
6440   }
6441
6442 \cs_new_protected:Nn \__stex_structures_do_assign_list:n {
6443   \clist_clear:N \l__stex_structures_fields_clist
6444   \tl_if_empty:nF {#1} {
6445     \keyval_parse>NNn\TODO\__stex_structures_do_assign:nn{#1}
6446   }
6447 }
6448
6449 \cs_new_protected:Nn \__stex_structures_do_assign:nn {
6450   \clist_put_right:Nn \l__stex_structures_fields_clist {{#1}{#2}}
6451 }
6452
6453 \cs_new_protected:Nn \__stex_structures_make_type:n {
6454   \tl_if_empty:nTF{#1} {
6455     \seq_clear:N \l_tmpa_seq
6456   }{
6457     \seq_set_split:Nnn \l_tmpa_seq ,{#1}
6458     \seq_pop_right:NN \l_tmpa_seq \l_tmpa_tl
6459     \seq_reverse:N \l_tmpa_seq

```

```

6460 }
6461 \tl_set:Nx \l__stex_structures_current_type_tl {
6462   \symuse{Metatheory?module-type~merge}{
6463     {
6464       \exp_args:No \exp_not:n \l_stex_current_redo_tl
6465       \stex_term_oms_or_omv:nnn{}{}{}
6466     }
6467     \seq_map_function:NN \l_tmpa_seq \__stex_structures_make_mod:n
6468     \clist_if_empty:NF \l_stex_structures_fields_clist {
6469       ,\symuse{Metatheory?anonymous~record}{
6470         \exp_args:Ne \tl_tail:n{
6471           \clist_map_function:NN \l_stex_structures_fields_clist \__stex_structures_make_
6472           }
6473         }
6474       }
6475     }
6476   }
6477 }
6478
6479 \cs_new:Nn \__stex_structures_make_mod:n {
6480   ,\symuse{Metatheory?module-type}{
6481     \stex_annotation:nn{data-shtml-term=OMMOD,data-shtml-head={#1}}{}
6482   }
6483 }
6484
6485 \cs_new:Nn \__stex_structures_make_oml:n {
6486   \__stex_structures_make_oml:nn #1
6487 }
6488 \cs_new:Nn \__stex_structures_make_oml:nn {
6489   ,\stex_annotation:nn{
6490     data-shtml-term=OML,
6491     data-shtml-head={#1}
6492   }{
6493     \__stex_annotation_force_break:n{
6494       \stex_annotation:nn{data-shtml-definiens={}{}\exp_not:n{#2!}}
6495     }
6496   }
6497 }

```

Insert the structure type as a term:

```

6498 \cs_new:Nn \__stex_structures_current_type: {
6499   \%exp_args:No \exp_not:n \l_stex_current_redo_tl
6500   \tl_set:Nn \exp_not:N \l_stex_current_term_tl {
6501     \exp_args:No\exp_not:n\l_stex_structures_current_type_tl
6502   }
6503   \stex_term_oms_or_omv:nnn{}{}{}
6504 }

```

The structure type itself:

```

6505 \cs_new_protected:Npn \__stex_structures_maybe_notation:w [ #1 ] {
6506   \tl_set_eq:NN \l_stex_current_term_tl \l_stex_structures_current_type_tl
6507   \cs_if_exist:cTF{\l_stex_notation_\l_stex_current_symbol_str _#1_cs} {
6508     \use:c{\l_stex_notation_\l_stex_current_symbol_str _#1_cs}\group_end:
6509   }{
6510     \__stex_structures_make_prop:

```

```

6511     \__stex_structures_make_prop_assign:
6512     \__stex_structures_present_i:w [#1]
6513   }
6514 }
6515
6516 \cs_new_protected:Nn \__stex_structures_present: {
6517   \peekCharCode:NTF [ {
6518     \__stex_structures_present_i:w
6519   }{
6520     \__stex_structures_present:nn{}{}
6521   }
6522 }
6523
6524 \cs_new_protected:Npn \__stex_structures_present_i:w [#1] {
6525   \int_compare:nNnTF{\clist_count:n{#1}} = 1 {
6526     \__stex_structures_present:nn{}{#1}
6527   }{
6528     \peekCharCode:NTF [ {
6529       \__stex_structures_present_ii:nw{#1}
6530     }{
6531       \__stex_structures_present:nn{#1}{}
6532     }
6533   }
6534 }
6535
6536 %First: clist, second:notation-id
6537 \cs_new_protected:Npn \__stex_structures_present_ii:nw #1 [#2] {
6538   \__stex_structures_present:nn{#1}{#2}
6539 }
6540
6541 \cs_new_protected:Nn \__stex_structures_present:nn {
6542   \clist_clear:N \l__stex_structures_clist
6543   \tl_if_empty:nTF{#1} {
6544     \cs_set:Npn \l__stex_structures_cs ##1 ##2 ##3 {
6545       \tl_if_empty:nF{##2} {
6546         \__stex_structures_present_entry:nn {##1}{##3}
6547       }
6548     }
6549   }{
6550     \cs_set:Npn \l__stex_structures_cs ##1 ##2 ##3 {
6551       \exp_args:Ne \clist_if_in:nnT{\tl_to_str:n{#1}}{##1} {
6552         \__stex_structures_present_entry:nn {##1}{##3}
6553       }
6554     }
6555   }
6556 \prop_map_inline:Nn \l__stex_structures_prop {
6557   \l__stex_structures_cs {##1} ##2
6558 }
6559 \__stex_term_oms_or_omv:nnn{}{}{
6560   \exp_args:Nno \use:n{
6561     \bool_set_true:N \l_stex_allow_semantic_bool
6562     \symuse{Metatheory?mathematical-structure}[#2]
6563   }{\l__stex_structures_clist}
6564 }\group_end:

```

```

6565 }
6566 \cs_new_protected:Nn \__stex_structures_present_entry:nn {
6567   \seq_if_in:NnTF \l__stex_structures_assigned_seq {#1} {
6568     \clist_put_right:Nn \l__stex_structures_clist {#2!}
6569   }{
6570     \exp_args:NNe \clist_put_right:Nn \l__stex_structures_clist {
6571       \__stex_next_symbol:n {
6572         \exp_args:No \exp_not:n \l__stex_structures_set_comp_tl
6573         \tl_set:Nn \exp_not:N \l__stex_structures_this_tl {
6574           \exp_args:No \exp_not:n \l__stex_structures_this_tl
6575         }
6576       }
6577       \exp_not:n {
6578         \tl_set_eq:NN \this \l__stex_structures_this_tl
6579       }
6580       \tl_set:Nn \exp_not:N \l_stex_return_notation_tl {
6581         \exp_args:No \exp_not:n \l_stex_return_notation_tl
6582       }
6583     }
6584     \exp_not:n{#2!}
6585   }
6586 }
6587 }
6588 \cs_new_protected:Npn \_thiscomp #1 #2 {
6589   {\tl_set:cn{this}{{}}#1#2}\c_math_subscript_token{
6590     \group_begin:
6591     \bool_set_true:N \l_stex_allow_semantic_bool
6592     \l__stex_structures_this_tl
6593     \group_end:
6594   }
6595 }
6596 }
6597 }
6598 \cs_new_protected:Nn \__stex_structures_set_thiscomp: {
6599   \exp_args:Ne \tl_if_eq:NNF {\tl_head:N \maincomp} \_thiscomp {
6600     \edef\maincomp {\_thiscomp{\comp}}
6601   }
6602 }
6603 }
6604 \cs_new_protected:Nn \__stex_structures_set_custom_comp:n {
6605   \exp_args:Ne \tl_if_eq:NNF {\tl_head:N \maincomp} \_customthiscomp {
6606     \cs_set_protected:Npx \_customthiscomp ##1 {
6607       \group_begin:
6608         \bool_set_true:N \l_stex_allow_semantic_bool
6609         \exp_not:n{
6610           \cs_set:Npn \l__stex_structures_comp_cs ##1 {
6611             #1
6612           }
6613           \def\maincomp
6614             }{\comp}
6615             \exp_not:N\l__stex_structures_comp_cs{\comp{##1}}
6616           \group_end:
6617         }
6618   }

```

```

6619     \def\maincomp {\_customthiscomp}
6620   }
6621 }
6622
this (of type structure):
6623
6624 \cs_new_protected:Nn \__stex_structures_invoke_this:n {
6625   \peek_charcode_remove:NTF ! {
6626     \exp_args:Nne\use:nn{
6627       \group_end:\symuse{Metatheory?of-type}[invisible]{
6628         \tl_if_empty:nTF{\#1}{\__stex_annotation_force_break:n{}{\#1}
6629           }
6630         }{
6631           {\__stex_structures_current_type:}
6632         }
6633       }{
6634         \__stex_structures_invoke_maybe_field:nn{\#1}
6635       }
6636   }
6637
6638 \cs_new_protected:Nn \__stex_structures_invoke_maybe_field:nn {
6639   \__stex_structures_make_prop:
6640   \__stex_structures_set_this:n{\#1}
6641   \tl_if_empty:nTF{\#2} {
6642     \__stex_structures_make_prop_assign:
6643     \__stex_structures_present:
6644   }{
6645     \__stex_structures_invoke_field:n{\#2}
6646   }
6647 }
6648
6649 \cs_new_protected:Nn \__stex_structures_set_this:n {
6650   \tl_if_empty:nTF{\#1} {
6651     \% \tl_put_right:Nn \l_stex_current_redo_tl {
6652     \% \tl_clear:N \l__stex_structures_this_tl
6653     \%}
6654   }{
6655     \tl_set:Nx \l__stex_structures_this_tl {
6656       \bool_set_true:N \l_stex_allow_semantic_bool
6657       \tl_set:Nn \exp_not:N \this {
6658         \exp_args:No \exp_not:n \this
6659       }
6660       \exp_not:n{\#1}
6661     }
6662     \tl_set_eq:NN \this \l__stex_structures_this_tl
6663     \% \l_stex_return_notation_tl
6664   }
6665 }
6666
6667 \cs_new_protected:Nn \__stex_structures_get_field_name:n {
6668   \str_set:Nx \l__stex_structures_field_name_str {
6669     \exp_args:Nne \use:n {\exp_after:wN \use_i:nn \use:n}
6670     {\prop_item:Nn \l__stex_structures_prop {\#1}}
6671   }

```

```

6672 \str_if_empty:NNT \l__stex_structures_field_name_str {
6673   \str_set:Nn \l__stex_structures_field_name_str {\#1}
6674 }
6675 }

6676 \cs_new_protected:Nn \__stex_structures_invoke_field:n {
6677   \prop_if_in:NnTF \l__stex_structures_prop {\#1} {
6678     \__stex_structures_get_field_name:n{\#1}
6679     \tl_clear:N \l__stex_structures_more_nextsymbol_tl
6680     \%exp_args:NNe \seq_if_in:NnF \l__stex_structures_assigned_seq {\tl_to_str:n{\#1}}{
6681       \tl_set:Nx \l__stex_structures_more_nextsymbol_tl {
6682         \tl_set:Nn \exp_not:N \l__stex_structures_this_tl {
6683           \exp_args:No \exp_not:n \l__stex_structures_this_tl
6684         }
6685       \exp_not:n {
6686         \tl_set_eq:NN \this \l__stex_structures_this_tl
6687       }
6688       \tl_set:Nn \exp_not:N \l_stex_return_notation_tl {
6689         \exp_args:No \exp_not:n \l_stex_return_notation_tl
6690       }
6691       \exp_args:No \exp_not:n \l__stex_structures_set_comp_tl
6692     }
6693   }
6694 }

6695 \exp_args:NNx \use:nn \group_end: {
6696   \stex_next_symbol:n {
6697     \exp_args:No \exp_not:n \l__stex_structures_redo_tl
6698     \tl_set:Nn \exp_not:N \l_stex_current_term_tl {
6699       \symuse{Metatheory?record~field}{%
6700         \symuse{Metatheory?of~type}{%
6701           \exp_args:No \tl_if_empty:nTF \l__stex_structures_this_tl {\stex_annotation_force_%
6702             }{ \__stex_structures_current_type: }%
6703           }%
6704           \stex_annotation:nn{data-shtml-term=OML,data-shtml-head={\l__stex_structures_field_%
6705             }%
6706           }%
6707           \exp_args:No \exp_not:n \l__stex_structures_more_nextsymbol_tl
6708         }%
6709         \exp_not:N \use_i:nn
6710         \prop_item:Nn \l__stex_structures_prop {\#1}
6711       }%
6712     }%
6713     \msg_error:nnn{stex}{error/unknownfield}{\#1}
6714   }
6715 }

6716 \cs_new_protected:Nn \__stex_structures_make_prop: {
6717   \prop_clear:N \l__stex_structures_prop
6718   \seq_clear:N \l__stex_structures_seq
6719   \seq_clear:N \l__stex_structures_assigned_seq
6720   \tl_clear:N \l__stex_structures_redo_tl
6721   \__stex_structures_prop_do_decls:
6722   \__stex_structures_prop_do_notations:
6723 }
6724 }

6725

```

```

6726 \cs_new_protected:Nn \__stex_structures_make_prop_assign: {
6727   \clist_if_empty:NF \l__stex_structures_fields_clist {
6728     \clist_map_inline:Nn \l__stex_structures_fields_clist {
6729       \__stex_structures_make_prop_assign:nn ##1
6730     }
6731   }
6732 }
6733
6734 \cs_new_protected:Nn \__stex_structures_make_prop_assign:nn {
6735   \prop_if_in:NnTF \l__stex_structures_prop {#1} {
6736     \exp_args:NNe \seq_put_right:Nn \l__stex_structures_assigned_seq {\tl_to_str:n{#1}}
6737     \exp_args:Nne \use:nn {\__stex_structures_make_prop_assign_replace:nnnn {#1}{#2}}
6738     {\prop_item:Nn \l__stex_structures_prop {#1}}
6739   }{
6740     \msg_error:nnn{stex}{error/unknownfieldass}{#1}
6741   }
6742 }
6743 \cs_new_protected:Nn \__stex_structures_make_prop_assign_replace:nnnn {
6744   \prop_put:Nnn \l__stex_structures_prop {#1}{#3}{#2}
6745   \tl_if_empty:nF{#3} {
6746     \tl_set:cn{#1}{#2}
6747     \tl_put_right:Nn \l__stex_structures_redo_tl {
6748       \tl_set:cn{#1}{#2}
6749     }
6750   }
6751 }
6752
6753 \cs_new_protected:Nn \__stex_structures_prop_do_decls: {
6754   \exp_args:No \stex_iterate_symbols:nn \l_stex_current_type_tl {
6755     \tl_if_empty:nTF{##2} {
6756       \__stex_structures_do_decl_nomacro:nnnnnnnn{##3}
6757     }{
6758       \__stex_structures_do_decl:nnnnnnnn{##2}
6759     }
6760     {##1}{##3}{##4}{##5}{##6}{##7}{##8}{##9}
6761   }
6762 }
6763
6764 \cs_new_protected:Nn \__stex_structures_do_decl_nomacro:nnnnnnnn {
6765   \prop_if_in:NnF \l__stex_structures_prop {#1} {
6766     \seq_put_left:Nx \l__stex_structures_seq {\tl_to_str:n{#2?#3}}
6767     \prop_put:Nnn \l__stex_structures_prop {#1} {
6768       {}{
6769         \stex_invoke_symbol:nnnnnnnN
6770         {#2}
6771         {#3}
6772         {#4}{#5}{#6}{#7}{#8}#9
6773       }
6774     }
6775   }
6776 }
6777
6778 \cs_new_protected:Nn \__stex_structures_do_decl:nnnnnnnn {
6779   \prop_if_in:NnF \l__stex_structures_prop {#1} {

```

```

6780   \seq_put_left:Nx \l__stex_structures_seq {\tl_to_str:n{#2##3}}
6781   \prop_put:Nnn \l__stex_structures_prop {#1}{
6782     {#3} {
6783       \_stex_invoke_symbol:nnnnnnnN
6784       {#2}
6785       {#3}
6786       {#4}{#5}{#6}{#7}{#8}#9
6787     }
6788   }
6789 }
6790 \%tl_set:cn{#1} {
6791 % \_stex_invoke_symbol:nnnnnnnN
6792 % {#2}{#3}{#4}{#5}{#6}{#7}{#8}#9
6793 %}
6794 \%tl_put_right:Nn \l__stex_structures_redo_tl {
6795 % \tl_set:cn{#1} {
6796 %   \_stex_invoke_symbol:nnnnnnnN
6797 %   {#2}{#3}{#4}{#5}{#6}{#7}{#8}#9
6798 % }
6799 %}
6800 }
6801
6802 \cs_new_protected:Nn \__stex_structures_prop_do_notations: {
6803   \exp_args:No \stex_iterate_notations:nn \l_stex_current_type_tl {
6804     \exp_args:NNe \seq_if_in:NnT \l__stex_structures_seq {\tl_to_str:n{##1}}{
6805       \tl_put_right:Nn \l__stex_structures_redo_tl {
6806         \cs_if_exist:cF{\l_stex_notation_##1 _##2_cs} {
6807           \tl_set:cn{\l_stex_notation_##1 _##2_cs}{##4}
6808         }
6809         \cs_if_exist:cF{\l_stex_notation_##1 __cs} {
6810           \tl_set:cn{\l_stex_notation_##1 __cs}{##4}
6811         }
6812       }
6813       \cs_if_exist:cF{\l_stex_notation_##1 _##2_cs} {
6814         \tl_set:cn{\l_stex_notation_##1 _##2_cs}{##4}
6815       }
6816       \cs_if_exist:cF{\l_stex_notation_##1 __cs} {
6817         \tl_set:cn{\l_stex_notation_##1 __cs}{##4}
6818       }
6819       \tl_if_empty:nF{##5} {
6820         \tl_put_right:Nn \l__stex_structures_redo_tl {
6821           \cs_if_exist:cF{\l_stex_notation_##1 _op_##2_cs} {
6822             \tl_set:cn{\l_stex_notation_##1 _op_##2_cs}{##5}
6823           }
6824           \cs_if_exist:cF{\l_stex_notation_##1 _op__cs} {
6825             \tl_set:cn{\l_stex_notation_##1 _op__cs}{##5}
6826           }
6827         }
6828         \cs_if_exist:cF{\l_stex_notation_##1 _op_##2_cs} {
6829           \tl_set:cn{\l_stex_notation_##1 _op_##2_cs}{##5}
6830         }
6831         \cs_if_exist:cF{\l_stex_notation_##1 _op__cs} {
6832           \tl_set:cn{\l_stex_notation_##1 _op__cs}{##5}
6833         }

```

```

6834         }
6835     }
6836 }
6837 }

\usestructure

6838 \cs_new_protected:Npn \usestructure #1 {
6839     \stex_get_mathstructure:n{ #1 }
6840     \seq_clear:N \l__stex_structures_imports_seq
6841     \clist_map_inline:Nn \l_stex_get_symbol_type_tl {
6842         \exp_args:Ne \stex_if_module_exists:nT{\tl_to_str:n{##1}}{
6843             \seq_put_right:Nn \l__stex_structures_imports_seq{##1}
6844         }
6845     }
6846     \seq_map_inline:Nn \l__stex_structures_imports_seq {
6847         \stex_if_do_html:T {
6848             \hbox{\stex_annotation_invisible:nn
6849                 {data-shtml-usemodule=##1} {}}
6850         }
6851         \stex_activate_module:n {##1}
6852     }
6853 }

```

(End of definition for `\usestructure`. This function is documented on page 99.)

## 13.10 Statements

```

6854 <@=stex_statements>
6855
6856 \stex_keys_define:nnnn{statement}{
6857     \str_clear:N \l_stex_key_name_str
6858     \str_clear:N \l_stex_key_mroname_str
6859     \clist_clear:N \l_stex_key_for_clist
6860     \str_clear:N \l_stex_key_args_str
6861     \tl_clear:N \l_stex_key_type_tl
6862     \tl_clear:N \l_stex_key_def_tl
6863     \tl_clear:N \l_stex_key_return_tl
6864     \clist_clear:N \l_stex_key_argtypes_clist
6865 }{
6866     name      .str_set:N = \l_stex_key_name_str ,
6867     for       .clist_set:N = \l_stex_key_for_clist ,
6868     macro     .str_set:N = \l_stex_key_mroname_str ,
6869     % start   .str_set:N = \l_stex_key_title_str , % TODO remove
6870     type     .tl_set:N = \l_stex_key_type_tl ,
6871     judgment  .code:n = {},
6872     from     .code:n= {}, % TODO remove
6873     to      .code:n={} % TODO remove
6874 }{id,title,style,symargs}

\stex_new_statement:nn
6875 \cs_new_protected:Npn \stex_do_for_list: {
6876     \seq_clear:N \l_stex_fors_seq
6877     \clist_map_inline:Nn \l_stex_key_for_clist {
6878         \exp_args:Ne\stex_get_symbol:n{\tl_to_str:n{##1}}

```

```

6879     \seq_put_right:Nx \l_stex_fors_seq
6880         {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
6881     }
6882 }
6883
6884 \cs_new_protected:Nn \__stex_statements_setup:nn {
6885     \str_if_empty:NF \l_stex_key_mroname_str {
6886         \str_if_empty:NT \l_stex_key_name_str {
6887             \str_set_eq:NN \l_stex_key_name_str \l_stex_key_mroname_str
6888         }
6889     }
6890     \stex_do_for_list:
6891     \str_if_empty:NF \l_stex_key_name_str {
6892         \__stex_statements_force_id:
6893         \seq_put_right:Nx \l_stex_fors_seq {
6894             \l_stex_current_module_str ? \l_stex_key_name_str
6895         }
6896         \str_set_eq:NN \l_stex_mroname_str \l_stex_key_mroname_str
6897         \str_set:Nn \l_stex_key_role_str {#2}
6898         \stex_symdecl_do:
6899         \exp_args:Nnx \use:nn {\stex_module_add_symbol:nnnnnnnN} {
6900             {\l_stex_key_mroname_str}{\l_stex_key_name_str}
6901             {\int_use:N \l_stex_get_symbol_arity_int}
6902             {\l_stex_get_symbol_args_tl}
6903             {#1}{}{}\stex_invoke_symbol:
6904         }
6905         \stex_if_do_html:T \stex_symdecl_html:
6906     }
6907     \str_clear:N \l__stex_statements_uri_str
6908     \str_if_empty:NTF \l_stex_key_name_str {
6909         \stex_debug:nfn{statement}{no~name}
6910         \int_compare:nNnTF {\seq_count:N \l_stex_fors_seq} = 1 {
6911             \str_set:Nx \l__stex_statements_uri_str {\seq_item:Nn \l_stex_fors_seq 1}
6912             \stex_debug:nn{statement}{for:~\l__stex_statements_uri_str}
6913         }{
6914             \stex_debug:nn{statement}{no~for}
6915         }
6916     }{
6917         \str_set:Nx \l__stex_statements_uri_str {\l_stex_current_module_str ? \l_stex_key_name_s
6918         \stex_debug:nn{statement}{name:~\l__stex_statements_uri_str}
6919     }
6920 }
6921
6922 \cs_new:Nn \__stex_statements_html_keyvals:nn {
6923     data-shtmlt-#1={},
6924     data-shtmlt-inlinet={#2},
6925     \seq_if_empty:NF \l_stex_fors_seq {
6926         data-shtmlt-fors={\seq_use:Nn \l_stex_fors_seq ,}
6927     }
6928     \str_if_empty:NF \l_stex_key_id_str {
6929         data-shtmlt-id={\stex_uri_use:N \l_stex_current_doc_uri ? \l_stex_key_id_str}
6930     }
6931     \clist_if_empty:NF \l_stex_key_style_clist {
6932         data-shtmlt-styles={\l_stex_key_style_clist}

```

```

6933     }
6934 }
6935
6936 \cs_new_protected:Nn \stex_new_statement:nnn {
6937     \stex_new_stylable_env:nnnnnnn {#1}{0{}}
6938     \stex_keys_set:nn{statement}{##1}
6939     #3
6940
6941     \stex_if_smsmode:F {
6942         \exp_args:Nne \begin{stex_annotation_env}{
6943             \__stex_statements_html_keyvals:nn{#1}{false}
6944         }
6945         \tl_set_eq:NN \thistitle \l_stex_key_title_tl
6946         \str_set_eq:NN \thisname \l_stex_key_name_str
6947         \clist_set_eq:NN \thisfor \l_stex_key_for_str
6948         \stex_if_html_backend:TF {
6949             \noindent
6950             \stex_annotation:nn{data-shtml-title={}}{\_stex_annotation_force_break:n\l_stex_key_title}
6951         }
6952         \stex_style_apply:
6953     }
6954     \_stex_do_id:
6955     \stex_smsmode_do:
6956 }{
6957     \stex_if_smsmode:F {
6958         \stex_if_html_backend:F \stex_style_apply:
6959         \end{stex_annotation_env}
6960     }
6961 }{}{}{s}
6962 \stex_sms_allow_env:n{s#1}
6963
6964 \tl_if_empty:nF{#2}{

6965     \exp_after:wN \NewDocumentCommand \cs:w inline#2\cs_end: { 0{} m}{

6966         \group_begin:
6967         \stex_keys_set:nn{statement}{##1}
6968         #3
6969         \_stex_do_id:
6970         \stex_if_smsmode:F{
6971             \exp_args:Ne \stex_annotation:nn{\__stex_statements_html_keyvals:nn{#1}{true}}{
6972                 \stex_annotation_force_break:n{##2}
6973             }
6974         }
6975         \group_end:
6976         \stex_smsmode_do:
6977     }
6978     \exp_after:wN \stex_sms_allow_escape:N\cs:w inline#2\cs_end:
6979 }
6980 }

6981 \cs_new_protected:Nn \__stex_statements_setup_def: {

6982     \stex_if_smsmode:F{
6983         \seq_map_inline:Nn \l_stex_fors_seq {
6984             \stex_ref_new_sym_target:n{##1}
6985         }
6986     }

```

```

6987   }
6988   \stex_reactivate_macro:N \definiendum
6989   \stex_reactivate_macro:N \defnotation
6990   \stex_reactivate_macro:N \defname
6991   \stex_reactivate_macro:N \Defname
6992   \stex_reactivate_macro:N \varbind
6993 }
6994
6995 \cs_new_protected:Nn \__stex_statements_force_id: {
6996   \str_if_empty:NT \l_stex_key_id_str {
6997     \__stex_ref_new_id:n{}
6998     \str_set_eq:NN \l_stex_key_id_str \l__stex_refs_str
6999   }
7000 }
7001
7002 \stex_new_statement:nnn{definition}{def}{
7003   \__stex_statements_force_id:
7004   \__stex_statements_setup:nn{}{}
7005   \__stex_statements_setup_def:
7006   \stex_reactivate_macro:N \definiens
7007 }
7008 \stex_new_statement:nnn{assertion}{ass}{
7009   \__stex_statements_setup:nn{}{assertion}
7010   \stex_if_smsmode:F{
7011     \seq_map_inline:Nn \l_stex_fors_seq {
7012       \stex_ref_new_sym_target:n{\##1}
7013     }
7014   }
7015   \stex_reactivate_macro:N \varbind
7016   \stex_reactivate_macro:N \conclusion
7017   \stex_reactivate_macro:N \premise
7018   \stex_reactivate_macro:N \definiendum
7019   \stex_reactivate_macro:N \defnotation
7020   \stex_reactivate_macro:N \defname
7021   \stex_reactivate_macro:N \Defname
7022 }
7023 \stex_new_statement:nnn{example}{ex}{\stex_if_smsmode:F {\__stex_statements_setup:nn{}{example}}
7024 \stex_new_statement:nnn{paragraph}{}{
7025   \clist_if_in:NnTF \l_stex_key_style_clist {syndoc} {
7026     \__stex_statements_force_id:
7027     \__stex_statements_setup:nn{}{}
7028     \__stex_statements_setup_def:
7029   }{
7030     \__stex_statements_setup:nn{}{}
7031   }
7032 }

```

(End of definition for \stex\_new\_statement:nn. This function is documented on page ??.)

```

definiendum
7033 \cs_new_protected:Nn \__stex_statements_do_defref:nn {
7034   \stex_if_html_backend:T{\relax\ifvmode\indent\fi}
7035   \group_begin:
7036   \stex_get_symbol:n{\#1}

```

```

7037 \bool_if:NTF \l_stex_allow_semantic_bool{
7038   \str_set:Nx\l_stex_current_symbol_str
7039     {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
7040   \str_if_in:NnT \l_stex_get_symbol_name_str / {
7041     \str_set:Nx \l_stex_get_symbol_name_str {
7042       \exp_after:wN \_stex_split_slash: \l_stex_get_symbol_name_str
7043       /\_stex_args_end:
7044     }
7045   }
7046   \exp_args:No \stex_ref_new_sym_target:n \l_stex_current_symbol_str
7047   \def\comp{\_defcomp}
7048   \stex_annotate:nn{data-shtml-definiendum=\l_stex_current_symbol_str}{\comp{#2}}
7049   {
7050     \msg_error:nnnx{stex}{error/notallowed}{#1}{\l_stex_current_symbol_str}
7051   }
7052   \group_end:
7053 }

7054 \stex_keys_define:nnnn{defname}{}{
7055   gf .code:n = {}
7056 }{symname}
7058 \NewDocumentCommand \defnotation{ m } {
7059   \_stex_next_symbol:n { \def\comp{\_defcomp}}#1
7060 }
7062 \stex_deactivate_macro:Nn \defnotation {definition~environments}
7063
7064 \NewDocumentCommand \definiendum { O{} m m} {
7065   \stex_keys_set:nn{defname}{ #1 }
7066   \__stex_statements_do_defref:nn{#2}{#3}
7067 }
7068 \stex_deactivate_macro:Nn \definiendum {definition~environments}
7069
7070 \NewDocumentCommand \defname { O{} m } {
7071   \stex_keys_set:nn{defname}{#1}
7072   \__stex_statements_do_defref:nn{#2}(
7073     \l_stex_key_pre_tl\l_stex_get_symbol_name_str\l_stex_key_post_tl
7074   )
7075 }
7076 \stex_deactivate_macro:Nn \defname {definition~environments}
7077
7078 \NewDocumentCommand \Defname { O{} m } {
7079   \stex_keys_set:nn{defname}{#1}
7080   \__stex_statements_do_defref:nn{#2}(
7081     \l_stex_key_pre_tl\exp_after:wN\_stex_capitalize:n\l_stex_get_symbol_name_str\l_stex_key
7082   )
7083 }
7084 \stex_deactivate_macro:Nn \Defname {definition~environments}
7085
7086 \NewDocumentCommand \definiens { O{} m }{
7087   \group_begin:
7088   \str_clear:N \l_stex_get_symbol_name_str
7089   \tl_if_empty:nF {#1} {

```

```

7091   \stex_get_symbol:n { #1 }
7092   \str_set:Nx \l__stex_statements_uri_str
7093     {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
7094   }
7095   \str_if_empty:NT \l__stex_statements_uri_str {
7096     \msg_error:nn{stex}{error/definiensfor}
7097   }
7098   \stex_debug:nn{definiens}{Checking-\l__stex_statements_uri_str}
7099
7100   \exp_args:No \_stex_add_definiens:nn \l__stex_statements_uri_str{#2}
7101
7102   \group_end:
7103   \stex_smsmode_do:
7104 }
7105 \stex_deactivate_macro:Nn \definiens {definition-environments}
7106 \stex_sms_allow_escape:N \definiens
7107
7108 \cs_new_protected:Nn \_stex_add_definiens:nn {
7109   \exp_args:Nno \stex_str_if_starts_with:nnT{#1} \l_stex_current_module_str {
7110     \prop_map_inline:cN{\c_stex_module_\l_stex_current_module_str _symbols_prop} {
7111       \stex_debug:nn{definiens}{#1 == \l_stex_current_module_str?##1}
7112       \str_if_eq:NOT {#1} {\l_stex_current_module_str?##1} {
7113         \prop_map_break:n{\_stex_add_definiens_inner:nnnnnnnn ##2}
7114       }
7115     }
7116   }
7117   \stex_if_smsmode:F{
7118     \stex_annotation:nn{ data-shtml-definiens={#1}}{
7119       #2 \%_stex_annotation_force_break:n{ #2 }
7120     }
7121   }
7122 }
7123
7124 \cs_new_protected:Nn \_stex_add_definiens_inner:nnnnnnnn {
7125   \stex_debug:nn{definiens}{Adding-definiens-to-\l_stex_current_module_str?#2}
7126   \prop_gput:cnn{\c_stex_module_\l_stex_current_module_str _symbols_prop}
7127     {#2}{##1}{#2}{#3}{#4}{defed}{#6}{#7}{#8}}
7128 }
7129
7130 \NewDocumentCommand \varbind {m} {
7131   \clist_map_inline:nn {#1} {
7132     \stex_get_var:n {##1}
7133     \stex_if_do_html:T {
7134       \stex_annotation_invisible:nn {data-shtml-bind=\l_stex_get_symbol_name_str}{}
7135     }
7136   }
7137 }
7138 \stex_deactivate_macro:Nn \varbind {definition-or~assertion~environments}
7139
7140 \NewDocumentCommand \conclusion { O{} m } {
7141   \group_begin:
7142   \str_clear:N \l_stex_get_symbol_name_str
7143   \tl_if_empty:nF {#1} {
7144     \stex_get_symbol:n { #1 }

```

```

7145   \str_set:Nx \l__stex_statements_uri_str
7146     {\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
7147   }
7148 \str_if_empty:NT \l__stex_statements_uri_str {
7149   \msg_error:nn{stex}{error/conclusionfor}
7150 }
7151 \stex_annotate:nn{ data-shtml-conclusion=\l__stex_statements_uri_str}{
7152   #2 \%_stex_annotate_force_break:n{ #2 }
7153 }
7154 \group_end:
7155 }
7156 \stex_deactivate_macro:Nn \conclusion {assertion-environments}
7157
7158 \NewDocumentCommand \premise {O{} m} {
7159   \tl_if_empty:nF {#1} {
7160     \stex_debug:nn{Here:}{Variable~#1}
7161     \exp_args:Nne\use:nn{\vardef}{{v#1}[name=#1]{#1}}
7162   }
7163   \stex_annotate:nn{data-shtml-premise={#1}}{#2}
7164 }
7165 \stex_deactivate_macro:Nn \premise {assertion-environments}

```

(End of definition for `definiendum`. This function is documented on page 103.)

## 13.11 Proofs

We first define some keys for the `sproof` environment.

```

7166 <@@=stex_proof>
7167 \stex_keys_define:nnnn{ spf }{
7168   \tl_clear:N \l_stex_key_for_clist
7169   \tl_clear:N \l_stex_key_from_tl
7170   \tl_set_eq:NN \l_stex_key_proofend_tl \__stex_proof_proof_box_tl
7171   \tl_clear:N \l_stex_key_continues_tl
7172   \tl_clear:N \l_stex_key_term_tl
7173   \tl_clear:N \l_stex_key_functions_tl
7174   \tl_clear:N \l_stex_key_method_tl
7175   \bool_set_false:N \l_stex_key_hide_bool
7176 }{
7177   for      .clist_set:N = \l_stex_key_for_clist ,
7178   from     .tl_set:N   = \l_stex_key_from_tl ,
7179   proofend .tl_set:N   = \l_stex_key_proofend_tl,
7180   continues .tl_set:N = \l_stex_key_continues_tl,
7181   functions .tl_set:N = \l_stex_key_functions_tl,
7182   term     .tl_set:N   = \l_stex_key_term_tl,
7183   method    .tl_set:N = \l_stex_key_method_tl,
7184   hide      .bool_set:N = \l_stex_key_hide_bool
7185 }{id,style,title}
7186
7187 \bool_set_true:N \l__stex_proof_inc_counter_bool

```

For proofs, we will have to have deeply nested structures of enumerated list-like environments. However, L<sup>A</sup>T<sub>E</sub>X only allows `enumerate` environments up to nesting depth 4 and general list environments up to listing depth 6. This is not enough for us. Therefore we have decided to go along the route proposed by Leslie Lamport to use a single top-level

list with dotted sequences of numbers to identify the position in the proof tree. Unfortunately, we could not use his `pf.sty` package directly, since it does not do automatic numbering, and we have to add keyword arguments all over the place, to accomodate semantic information.

```

7188 \intarray_new:Nn\l__stex_proof_counter_intarray{50}
7189 \cs_new_protected:Npn \__stex_proof_insert_number: {
7190   \int_set:Nn \l_tmpa_int {1}
7191   \bool_while_do:nn {
7192     \int_compare_p:nNn {
7193       \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int
7194     } > 0
7195   }{
7196     \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int .
7197     \int_incr:N \l_tmpa_int
7198   }
7199 }
7200 \cs_new_protected:Nn \__stex_proof_number_as_string:N {
7201   \str_clear:N #1
7202   \int_set:Nn \l_tmpa_int {1}
7203   \bool_while_do:nn {
7204     \int_compare_p:nNn {
7205       \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int
7206     } > 0
7207   }{
7208     \str_put_right:Nx #1 {\intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int .}
7209     \int_incr:N \l_tmpa_int
7210   }
7211 }
7212 \cs_new_protected:Npn \__stex_proof_inc_counter: {
7213   \int_set:Nn \l_tmpa_int {1}
7214   \bool_while_do:nn {
7215     \int_compare_p:nNn {
7216       \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int
7217     } > 0
7218   }{
7219     \int_incr:N \l_tmpa_int
7220   }
7221   \int_compare:nNnF \l_tmpa_int = 1 {
7222     \int_decr:N \l_tmpa_int
7223   }
7224   \intarray_gset:Nnn \l__stex_proof_counter_intarray \l_tmpa_int {
7225     \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int + 1
7226   }
7227 }
7228 }
7229 \cs_new_protected:Npn \__stex_proof_add_counter: {
7230   \int_set:Nn \l_tmpa_int {1}
7231   \bool_while_do:nn {
7232     \int_compare_p:nNn {
7233       \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int
7234     } > 0
7235   }{

```

```

7237     \int_incr:N \l_tmpa_int
7238 }
7239 \intarray_gset:Nnn \l__stex_proof_counter_intarray \l_tmpa_int { 1 }
7240 }
7241
7242 \cs_new_protected:Npn \__stex_proof_remove_counter: {
7243     \int_set:Nn \l_tmpa_int {1}
7244     \bool_while_do:nNn {
7245         \int_compare_p:nNn {
7246             \intarray_item:Nn \l__stex_proof_counter_intarray \l_tmpa_int
7247         } > 0
7248     }{
7249         \int_incr:N \l_tmpa_int
7250     }
7251     \int_decr:N \l_tmpa_int
7252     \intarray_gset:Nnn \l__stex_proof_counter_intarray \l_tmpa_int { 0 }
7253 }

spfsketch
7254 \newenvironment{spfsketchenv}{}{}
7255 \stex_new_stylable_cmd:nnnn{spfsketch}{0{}} m}{\par
7256     \begin{spfsketchenv}
7257     \stex_keys_set:nn{spf}{#1}
7258     \stex_do_for_list:
7259     \stex_do_id:
7260     \exp_args:Ne \stex_annotation:nn{
7261         data-shtml-proofsketch={
7262             \seq_if_empty:NF \l_stex_fors_seq {
7263                 \seq_use:Nn \l_stex_fors_seq ,
7264             }
7265         }{
7266             \stex_style_apply:
7267             #2
7268         }
7269     }
7270     \end{spfsketchenv}
7271 }{
7272     \noindent\emph{\spfsketchenv\autorefname :}-
7273 }

```

(End of definition for `spfsketch`. This function is documented on page ??.)

`\sproofend` This macro places a little box at the end of the line if there is space, or at the end of the next line if there isn't

```

7274 \tl_set:Nn \__stex_proof_box_tl {
7275     \ltx@ifpackageloaded{amssymb}{$\square$}{
7276         \hbox{\vrule\vbox{\hrule width 6 pt\vskip 6pt\hrule}\vrule}
7277     }
7278 }
7279
7280 \tl_set:Nn \sproofend {
7281     \tl_if_empty:NF \l_stex_key_proofend_tl {
7282         \hfil\null\nobreak\hfill\l_stex_key_proofend_tl\par\smallskip
7283     }
7284 }

```

(End of definition for \sproofend. This function is documented on page ??.)

```
\stexcommentfont
```

```
7285 \cs_new_protected:Npn \stexcommentfont {
7286     \small\itshape
7287 }
```

(End of definition for \stexcommentfont. This function is documented on page ??.)

sproof (env.)

```
7288 \cs_new_protected:Nn \__stex_proof_start_list:n {
7289     \begin{list}{}{
7290         \setlength\topsep{0pt}
7291         \setlength\parsep{0pt}
7292         \setlength\rightmargin{0pt}
7293     }\item[#1]
7294 }
7295 \cs_new_protected:Nn \__stex_proof_end_list: {
7296     \end{list}
7297 }
7298 \cs_new_protected:Nn \__stex_proof_html: {
7299     \stex_annotation_invisible:n{\hbox{
7300         \tl_if_empty:NF \l_stex_key_term_tl {
7301             $ \stex_annotation:nn{data-shtml-proofterm={}}{\l_stex_key_term_tl}$
7302         }
7303         \tl_if_empty:NF \l_stex_key_method_tl {
7304             \stex_annotation:nn{data-shtml-proofmethod={}}{\l_stex_key_method_tl}
7305         }
7306     }}
7307 }
7308 }
7309 \cs_new_protected:Nn \__stex_proof_html_env:n {
7310     \exp_args:Nne \begin{stex_annotation_env}{
7311         data-shtml-#1={
7312             \seq_if_empty:NF \l_stex_fors_seq {
7313                 \seq_use:Nn \l_stex_fors_seq ,
7314             }
7315         }
7316         \bool_if:NT \l_stex_key_hide_bool {
7317             data-shtml-proofhide=true
7318         }
7319     }
7320     \__stex_proof_html:
7321 }
7322 }
7323 \bool_set_false:N \l_stex_proof_in_spfblock_bool
7324 \cs_new_protected:Nn \__stex_proof_begin_proof:nn {\par
7325     \intarray_gzero:N \l_stex_proof_counter_intarray
7326     \intarray_gset:Nnn \l_stex_proof_counter_intarray 1 1
7327     \stex_keys_set:nn{spfsteps}{#1}
7328     \stex_do_for_list:
7329     \stex_if_do_html:T {
7330         \__stex_proof_html_env:n{proof}
```

```

7332 }
7333 \seq_map_inline:Nn \l_stex_fors_seq {
7334   \stex_debug:nn{definiens}{Adding-definiens-to-##1}
7335   \_stex_add_definiens:nn {##1}{\STEXinvisibl{proven}}
7336 }
7337 \stex_style_apply:
7338 \_stex_do_id:
7339 \stex_reactivate_macro:N \subproof
7340 \stex_reactivate_macro:N \spfstep
7341 \stex_reactivate_macro:N \conclude
7342 \stex_reactivate_macro:N \assumption
7343 \stex_reactivate_macro:N \eqstep
7344 \stex_reactivate_macro:N \yield
7345 \stex_reactivate_macro:N \spfblock
7346 \stex_reactivate_macro:N \spfjust
7347 \stex_annotation:nn{data-shtml-title={}}{#2}
7348 \stex_if_do_html:T{
7349   \begin{stex_annotation_env}{data-shtml-proofbody={}}
7350 }
7351 }
7352 \stex_new_stylable_env:nnnnnnn{proof}{0{} m}){
7353   \__stex_proof_begin_proof:nn{#1}{#2}
7354   \bool_set_true:N\l_stex_proof_in_spfblock_bool\__stex_proof_start_list:n{}
7355   \group_begin:\stexcommentfont
7356 }{
7357   \stex_style_apply:
7358   \stex_if_do_html:T{\end{stex_annotation_env}\end{stex_annotation_env}}
7359 }{
7360   \emph{\sproofautorefname :}-
7361 }{
7362   \sproofend
7363 }{s}
7364 \AddToHook{env/sproof/end}{
7365   \bool_if:NT\l_stex_proof_in_spfblock_bool {
7366     \group_end:\__stex_proof_end_list:
7367   }
7368 }
7369
7370 \stex_new_stylable_env:nnnnnnn{proof*}{0{} }{
7371   \__stex_proof_begin_proof:nn{#1}{}
7372   \bool_set_false:N\l_stex_proof_in_spfblock_bool
7373 }{
7374   \stex_style_apply:
7375   \stex_if_do_html:T{\end{stex_annotation_env}\end{stex_annotation_env}}
7376 }{
7377   \emph{Proof:}-
7378 }{
7379   \sproofend
7380 }{s}

subproof (env.)
7381 \str_set_eq:NN \subproofautorefname \spfstepautorefname
7382 \stex_new_stylable_env:nnnnnnn{subproof}{s 0{} m}{\par
7383   \stex_keys_set:nn{spf}{#2}

```

```

7384   \_stex_do_for_list:
7385   \stex_if_do_html:T {
7386     \_\_stex_proof_html_env:n{subproof}
7387   }
7388   \seq_map_inline:Nn \l_stex_fors_seq {
7389     \stex_debug:nn{definiens}{Adding-definiens-to-##1}
7390     \_stex_add_definiens:nn {##1}{\STEXinvisible{proven}}
7391   }
7392
7393   \IfBooleanTF #1 {
7394     \stex_style_apply:
7395     \str_if_empty:NF \l_stex_key_id_str {
7396       \_\_stex_proof_number_as_string:N \@currentlabel
7397       \str_set:Nx \@currentHref{subproof.\@currentlabel}
7398       \_stex_do_id:
7399     }
7400     \bool_set_false:N \l\_stex_proof_in_spfblock_bool
7401     \stex_annotation:nn{data-shtml-prooftitle={}}{#3}
7402   }{
7403     \bool_if:NTF \l\_stex_proof_in_spfblock_bool {
7404       \str_if_empty:NF \l_stex_key_id_str {
7405         \_\_stex_proof_number_as_string:N \@currentlabel
7406         \str_set:Nx \@currentHref{subproof.\@currentlabel}
7407         \_stex_do_id:
7408       }
7409       \_\_stex_proof_start_list:n\_\_stex_proof_insert_number:
7410         \stex_annotation:nn{data-shtml-prooftitle={}}{#3}
7411         \_\_stex_proof_add_counter:
7412         \stex_style_apply:
7413   }{
7414     \stex_annotation:nn{data-shtml-prooftitle={}}{#3}
7415     \stex_style_apply:
7416     \_stex_do_id:
7417   }
7418 }
7419 \stex_if_do_html:T{
7420   \begin{stex_annotation_env}{data-shtml-proofbody={}}
7421 }
7422 \bool_if:NT \l\_stex_proof_in_spfblock_bool {\group_begin:\stexcommentfont}
7423 }{
7424   \stex_style_apply:
7425   \bool_if:NT \l\_stex_proof_in_spfblock_bool \_\_stex_proof_inc_counter:
7426   \stex_if_do_html:T{\end{stex_annotation_env}}
7427   \bool_if:NT\l\_stex_proof_in_spfblock_bool \_\_stex_proof_end_list:
7428   \stex_if_do_html:T{\end{stex_annotation_env}}
7429   \aftergroup \_\_stex_proof_inblock_restore:
7430 }{}{}{}
7431 \AddToHook{env/subproof/before}{
7432   \bool_if:NT \l\_stex_proof_in_spfblock_bool \group_end:
7433 }
7434 \AddToHook{env/subproof/end}{
7435   \bool_if:NT\l\_stex_proof_in_spfblock_bool {
7436     \group_end:\_\_stex_proof_remove_counter:
7437     \% \_\_stex_proof_end_list:

```

```

7438     }
7439 }
7440 \stex_deactivate_macro:Nn \subproof {sproof~environments}
7441
7442 \cs_new_protected:Nn \__stex_proof_inblock_restore: {
7443     \bool_if:NT\l_stex_proof_in_spfblock_bool {
7444         \group_begin:\stexcommentfont
7445     }
7446 }

\spfstep
\conclude
7447
\assumption
7448 \stex_keys_define:nnnn { spfsteps } {
7449     \clist_clear:N \l_stex_key_for_clist
\eqstep
7450     \str_clear:N \l_stex_key_name_str
7451     \tl_clear:N \l_stex_key_method_tl
7452     \tl_clear:N \l_stex_key_term_tl
7453 }{
7454     for .clist_set:N = \l_stex_key_for_clist ,
7455     method .tl_set:N = \l_stex_key_method_tl,
7456     term .tl_set:N = \l_stex_key_term_tl,
7457     name .str_set_x:N = \l_stex_key_name_str
7458     % todo: style=inline
7459 }{id,style,title}
7460
7461 \newenvironment{spfstepenv}{
7462     \str_set_eq:NN \spfstepenvautorefname \spfstepautorefname
7463 }{}
7464
7465 \cs_new_protected:Nn \__stex_proof_step_html:nn {
7466     \stex_if_do_html:TF{
7467         \exp_args:Ne \stex_annotation:n{%
7468             data-shtml-spf#1=%
7469             \seq_if_empty:NF \l_stex_fors_seq {%
7470                 \seq_use:Nn \l_stex_fors_seq ,
7471             }%
7472         }%
7473         \str_if_empty:NF \l_stex_key_name_str {%
7474             data-shtml-stepname={\l_stex_key_name_str}%
7475         }%
7476     }%
7477     \__stex_proof_html:
7478     #2
7479 }
7480 }{ #2 }
7481
7482
7483 \cs_new_protected:Nn \__stex_proof_make_step_macro:Nnnnn {
7484     \NewDocumentCommand #1 {s O{} +m} {
7485         \bool_if:NT \l_stex_proof_in_spfblock_bool \group_end:
7486         \stex_keys_set:nn{spfsteps}{##2}
7487         \str_if_empty:NF \l_stex_key_name_str {
7488             \stex_debug:nn{Here:}{Variable-\l_stex_key_name_str}
7489             \exp_args:Nne\use:nn{\vardef}{\v\l_stex_key_name_str}[name=\l_stex_key_name_str]{\l_stex_key_name_str}
    }
}

```

```

7490     }
7491
7492 \begin{spfstepenv}
7493   \str_if_empty:N \l_stex_key_id_str {
7494     \__stex_proof_number_as_string:N \@currentlabel
7495     \str_set:Nx \@currentHref{spfstep.\@currentlabel}
7496     \__stex_do_id:
7497   }
7498
7499   \bool_if:NTF \l__stex_proof_in_spfblock_bool {
7500     \IfBooleanTF ##1 {
7501       \__stex_proof_step_html:nn{##2}{##3}
7502     }{
7503       \__stex_proof_step_html:nn{##2}{\__stex_proof_start_list:n{##3} ##3 \__stex_proof_end_#5
7504       }
7505     }
7506     \end{spfstepenv}
7507     \group_begin:\stexcommentfont
7508   }{
7509     \__stex_proof_step_html:nn{##2}{##3}
7510     \end{spfstepenv}
7511   }
7512 }
7513 \stex_deactivate_macro:Nn #1 {sproof~environments}
7514 }
7515
7516 \__stex_proof_make_step_macro:Nnnnn \assumption {assumption} \__stex_proof_insert_number: {}
7517 \__stex_proof_make_step_macro:Nnnnn \conclude {conclusion} {\$Rightarrow\$} {} {}
7518 \__stex_proof_make_step_macro:Nnnnn \spfstep {step} \__stex_proof_insert_number: {} \__stex_
7519
7520 \NewDocumentCommand \eqstep {s m}{
7521   \bool_if:NTF \l__stex_proof_in_spfblock_bool {
7522     \group_end:
7523     \IfBooleanTF #1 {
7524       \__stex_proof_step_html:nn{eqstep}{$= #2$}
7525     }{
7526       \__stex_proof_step_html:nn{eqstep}{\__stex_proof_start_list:n{$=$} $#2$ \__stex_proof_
7527     }
7528     \group_begin:\stexcommentfont
7529   }{
7530     \__stex_proof_step_html:nn{eqstep}{$= #2$}
7531   }
7532 }
7533 \stex_deactivate_macro:Nn \eqstep {sproof~environments}
7534
7535 \NewDocumentCommand \yield {+m}{
7536   \stex_annotation:nn{data-shtml-proofterm={}}{\#1}
7537 }
7538 \stex_deactivate_macro:Nn \yield {sproof~environments}
7539
7540 \NewDocumentEnvironment{spfblock}{}{
7541   \bool_set_false:N \l__stex_proof_in_spfblock_bool
7542 }{
7543   \aftergroup\__stex_proof_inblock_restore:

```

```

7544 }
7545 \stex_deactivate_macro:Nn \spfblock {sproof~environments}
7546 \AddToHook{env/spfblock/before} {
7547   \bool_if:NT \l__stex_proof_in_spfblock_bool \group_end:
7548 }
7549
7550
7551 \newcommand\spfjust[1]{
7552   \stex_annotate:nn{spfjust={}}{ #1 }
7553 }
7554 \stex_deactivate_macro:Nn \spfjust {sproof~environments}

(End of definition for \spfstep and others. These functions are documented on page ??.)

```

## 13.12 Metatheory

```

7555 <@=stex_meta>
7556 \group_begin:
7557   \cs_set:Npn \__stex_modules_persist_module: {}
7558   \cs_set:Npn \stex_check_term:n #1 {}
7559   \cs_set:Npn \__stex_sref_do_aux:n #1 { #1 }
7560   \bool_set_false:N \stex_html_do_output_bool
7561   \bool_set_false:N \c_stex_check_terms_bool
7562   \stex_uri_resolve:Nn \l_stex_current_ns_uri {http://mathhub.info/sTeX/meta}
7563   \stex_module_setup:n{Metatheory}
7564
7565   \symdef{of~type}[args=ii,invisible]{#1}
7566   \notation{of~type}[colon]{#1 \mathbin{\comp{:}} #2}
7567
7568   \symdef{apply}[args=ia,prec=0;\infprec x\infprec]{#1\mathopen{\comp{} #2 \mathclose{\comp}}}
7569   \notation{apply}[lambda]{#1\mathrel{\lambda} \argsep{#2}\mathrel{;}}
7570   \notation{apply}[infixop]{\argsep{#2}\mathbin{#1}}
7571   \notation{apply}[infixrel]{\argsep{#2}\mathrel{#1}}
7572
7573 % structures
7574   \symdef{module~type}[args=i,op=\mathtt{MOD}]{\mathopen{\comp{\mathtt{MOD}}}\mathclose{\comp{}}}
7575   \symdef{module~type~merge}[args=a,op=\oplus]{\argsep{#1}\mathbin{\comp{\oplus}}}
7576   \symdef{anonymous~record}[args=a]{\mathopen{\comp{[]}}\mathclose{\comp{}}}
7577   \symdef{record~field}[args=2]{#1\comp{.}\#2}
7578   \symdecl*{record-type}
7579
7580   \symdecl*{mathstruct}[name=mathematical-structure,args=a] % TODO
7581   \notation{mathstruct}[angle,prec=nobrackets]{\mathopen{\comp{\langle}}\mathclose{\comp{\rangle}}}
7582   \notation{mathstruct}[parens,prec=nobrackets]{\mathopen{\comp{()}}\mathclose{\comp{}}}
7583
7584 % sequences
7585   \symdef{ellipses}[ldots]{\ldots}
7586   \symdef{sequence~expression}[comma,args=a]{#1}
7587   \symdef{sequence~type}[args=1]{#1^{\comp{\ast}}}

```

```

7593 \symdef{sequence~map}[args=ia]{%
7594   \comp{\mathrm{map}}\mathopen{\comp{}}\mathclose{\comp{}}
7595   #2\mathclose{\comp{}}}
7596 }
7597 \iffalse
7598 % binder (\forall, \Pi, \lambda etc.)
7599 \symdef{pibind}[name=dependent function type,prec=nobrackets,
7600   op=(\cdot\cdot\cdot)\cdot\cdot\cdot; \cdot\cdot\cdot, args=Bi, assoc=pre]
7601 {\argmap{#1}{%
7602   \mathopen{\comp{}} ##1 \mathclose{\comp{}}}
7603   }{\mathbin{\comp{\cdot\cdot\cdot}}}\mathbin{\comp{\cdot\cdot\cdot}} ##2}
7604 \notation{pibind}[\forall]{\comp{\forall} #1\mathopen{\comp{.}} #2}
7605 \notation{pibind}[\Pi]{\mathop{\comp{\prod}}\c_math_subscript_token{#1}#2}
7606
7607 \symdef{mapbind}[name=lambda, mapsto, prec=nobrackets, op=\mapsto, args=Bi, assoc=pre]
7608 { #1 \mathrel{\comp{\mapsto}} #2}
7609 \notation{mapbind}[\lambda, prec=nobrackets, op=\lambda]
7610 { \comp{\lambda} #1 \mathopen{\comp{.}} #2}
7611 \fi
7612 \symdecl{bind}[args=Bi,assoc=pre]
7613 \notation{bind}[depfun,prec=nobrackets,op=(\cdot\cdot\cdot)\cdot\cdot\cdot;\cdot\cdot\cdot]
7614 { \mathopen{\comp{}} #1 \mathclose{\comp{}}}\mathbin{\comp{\cdot\cdot\cdot}} ##2}
7615 \notation{bind}[\forall]{\comp{\forall} #1.\cdot;#2}
7616 \notation{bind}[\Pi]{\mathop{\comp{\prod}}\c_math_subscript_token{#1}#2}
7617
7618 \symdef{implicit~bind}[args=Bi,assoc=pre]{\mathopen{\comp{}} #1 \mathclose{\comp{}}\c_math_
7619
7620 \symdecl*{integer~literal}
7621 \notation{integer~literal}{\mathbb{Z}}
7622
7623 \symdecl*{ordinal}
7624 \notation{ordinal}{\mathhtt{Ord}}
7625
7626 % propositions
7627 \symdef{prop}[name=proposition]{\mathhtt{Prop}}
7628 \symdef{judgment~holds}[args=i,role=judgment]{\comp{\vdash};#1}
7629
7630 % any object
7631 \symdef{object}{\mathhtt{Obj}}
7632
7633 % TODO DELETE
7634 \symdef{aseqdots}[args=a,prec=nobrackets]
7635 { #1\comp{,\ldots}##1\comp{},##2}
7636 \symdef{aseqfromto}[args=ai,prec=nobrackets]
7637 { #1\comp{,\ldots,}##2##1\comp{},##2}
7638 \symdef{aseqfromtovia}[args=a ii,prec=nobrackets]
7639 { #1\comp{,\ldots,}##2\comp{,\ldots,}##3##1\comp{},##2}
7640
7641
7642 \stex_close_module:
7643 \stex_uri_add_module:NNn \l_stex_metatheory_uri \l_stex_current_ns_uri {Metatheory}
7644 \global \let \l_stex_metatheory_uri \l_stex_metatheory_uri
7645 \global \let \c_stex_default_metatheory \l_stex_metatheory_uri
7646 \group_end:

```

### 13.13 MMT Interfaces

```

7647 <@@=todo>
7648 \cs_new_protected:Npn \MSC #1 {}

\MMTinclude

7649 \stex_new_styable_cmd:nnnn{MMTinclude}{m}{
7650   \stex_annotation_invisible:nn{data-shtml-import={#1}}{}
7651 }{}
7652 \stex_deactivate_macro:Nn \MMTinclude {module~environments}
7653 \stex_every_module:n {\stex_reactivate_macro:N \MMTinclude}

(End of definition for \MMTinclude. This function is documented on page ??.)
```

```

\MMTrule

7654 \NewDocumentCommand \MMTrule {m m} {
7655   \tl_if_empty:nTF{#2}{\seq_clear:N \l_tmpa_seq}{\seq_set_split:Nnn \l_tmpa_seq , {#2}}
7656 }
7657 \int_zero:N \l_tmpa_int
7658 \stex_annotation_invisible:n{
7659   $
7660   \stex_annotation:nn{data-shtml-rule={scala://#1}}{
7661     \stex_annotation_force_break:n{
7662       \seq_if_empty:NF \l_tmpa_seq {
7663         \seq_map_inline:Nn \l_tmpa_seq {
7664           \int_incr:N \l_tmpa_int
7665           \stex_annotation:nn{
7666             data-shtml-argmode=i,
7667             data-shtml-arg={\int_use:N \l_tmpa_int}
7668           }{ ##1 }
7669         }
7670       }
7671     }
7672   ~}
7673   }$}
7674 }
7675 }

7676 \stex_deactivate_macro:Nn \MMTrule {module~environments}
7677 \stex_every_module:n{\stex_reactivate_macro:N \MMTrule}
```

(End of definition for \MMTrule. This function is documented on page ??.)

```

mmtinterface (env.)

7678 \NewDocumentEnvironment { mmtinterface } { O{} m m } {
7679   \stex_module_setup_top_nosig:n { #3 }
7680   \str_set_eq:NN \l__todo_mmt_module_str \l_stex_current_module_str
7681   \str_clear:N \l_stex_current_module_str
7682   \stex_keys_set:nn { smodule }{ #1 }
7683   \stex_module_setup:n{ #2 }
7684   \str_set_eq:NN \l__todo_stex_module_str \l_stex_current_module_str
7685   \stex_debug:nn{mmt}{Interface~\l__todo_stex_module_str^~Jfor~\l__todo_mmt_module_str}
7686
7687   \stex_if_do_html:T {
7688     \exp_args:Nne \begin{stex_annotation_env} {
7689       data-shtml-theory={\l_stex_current_module_str},
```

```

7690     data-shtml-language={ \l_stex_current_language_str},
7691     data-shtml-signature={}
7692     \tl_if_empty:NF \l_stex_metatheory_uri {
7693         data-shtml-metatheory={\stex_uri_use:N \l_stex_metatheory_uri}
7694     }
7695 }
7696 \stex_annotation_invisible:n{ }
7697 \stex_annotation_invisible:nn
7698     {data-shtml-import=\l_todo_mmt_module_str} {}
7699 }
7700 \stex_module_add_code:x{
7701     \stex_activate_module:nf \l_todo_mmt_module_str }
7702 }
7703 \stex_module_add_morphism:nonn
7704     {}{\l_todo_mmt_module_str}{import}{}}
7705 \stex_reactivate_macro:N \mmtdef
7706 \stex_smsmode_do:
7707 }{
7708     \str_set_eq:NN \l_stex_current_module_str \l_todo_mmt_module_str
7709     \stex_close_module:
7710     \str_set_eq:NN \l_stex_current_module_str \l_todo_stex_module_str
7711     \stex_close_module:
7712     \stex_if_do_html:T { \end{stex_annotation_env} }
7713 }
7714 \stex_sms_allow_env:n{mmtinterface}

\mmtdef
7715 \NewDocumentCommand \mmtdef {m 0{} m} {
7716     \stex_keys_set:nn{symdef}{#2}
7717     \str_set:Nx \l_stex_macroname_str { #1 }
7718     \str_if_empty:NT \l_stex_key_name_str {
7719         \str_set:Nx \l_stex_key_name_str { #1 }
7720     }
7721     \stex_symdecl_do:
7722
7723     \str_set_eq:NN \l_stex_current_module_str \l_todo_mmt_module_str
7724     \cs_set_eq:NN \l_todo_old_metagroup_cd \stex_metagroup_do_in:nn
7725     \cs_set_protected:Npn \stex_metagroup_do_in:nn ##1 ##2 {##2}
7726     \exp_args:Nnx \use:nn {\stex_module_add_symbol:nnnnnnN} {
7727         {\l_stex_macroname_str}
7728         {\l_stex_key_name_str}
7729         {\int_use:N \l_stex_get_symbol_arity_int}
7730         {\l_stex_get_symbol_args_tl}
7731         {}
7732         {}
7733         {}
7734         \stex_invoke_symbol:
7735     }
7736     \cs_set_eq:NN \stex_metagroup_do_in:nn \l_todo_old_metagroup_cd
7737     \str_set_eq:NN \l_stex_current_module_str \l_todo_stex_module_str
7738
7739     \str_set_eq:NN \l_stex_get_symbol_mod_str \l_todo_mmt_module_str
7740     \str_set_eq:NN \l_stex_get_symbol_name_str \l_stex_key_name_str
7741     \stex_notation_parse:n{#3}

```

```

7742   \_stex_notation_check:
7743   \_stex_notation_add:
7744   \stex_if_do_html:T{
7745     \_stex_notation_do_html:n{\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str}
7746   }
7747   \stex_smsmode_do:
7748 }
7749 \stex_deactivate_macro:Nn \mmtdef {mmtinterface~environments}
7750 \stex_sms_allow_escape:N \mmtdef

```

(End of definition for `\mmtdef`. This function is documented on page ??.)

#### VoLL-KI Annotations

```

7751 \newcommand\precondition[2]{
7752   \str_clear:N \l_stex_get_symbol_name_str
7753   \stex_get_symbol:n{#2}
7754   \str_if_empty:NTF \l_stex_get_symbol_name_str{
7755     \errmessage{Unknown~symbol~#2}
7756   }
7757   \str_case:nnTF {#1} {
7758     {remember}{}
7759     {understand}{}
7760     {analyze}{}
7761     {evaluate}{}
7762     {apply}{}
7763     {create}{}
7764   }
7765   \ifstexhtml \else \bool_if:NT \c_stex_metadata_bool {
7766     \marginpar{\tiny \textbf{Precondition:}~#1~}
7767     \exp_args:Ne \symrefemph@urif{\l_stex_get_symbol_name_str}{\l_stex_get_symbol_mod_}
7768   }
7769 } \fi
7770 \stex_annotation_invisible:nn {
7771   data-shtml-preconditionsymbol={\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_st}
7772   data-shtml-preconditiondimension={#1}
7773 } {}
7774 } {\errmessage{Unknown~cognitive~dimension~#1}}
7775 }

7776 }
7777 \newcommand\objective[2]{
7778   \str_clear:N \l_stex_get_symbol_name_str
7779   \stex_get_symbol:n{#2}
7780   \str_if_empty:NTF \l_stex_get_symbol_name_str{
7781     \errmessage{Unknown~symbol~#2}
7782   }
7783   \str_case:nnTF {#1} {
7784     {remember}{}
7785     {understand}{}
7786     {analyze}{}
7787     {evaluate}{}
7788     {apply}{}
7789     {create}{}
7790   }
7791   \ifstexhtml \else \bool_if:NT \c_stex_metadata_bool {
7792     \marginpar{\tiny \textbf{Objective:}~#1~}

```

```

7793     \exp_args:N\symrefemph@uri{\l_stex_get_symbol_name_str}{\l_stex_get_symbol_mod_}
7794     }
7795   }\fi
7796 \stex_annotation_invisible:nn{
7797   data-shtml-objectivesymbol={\l_stex_get_symbol_mod_str ? \l_stex_get_symbol_name_str},
7798   data-shtml-objectivedimension={#1}
7799   }{}
7800 }{\errmessage{Unknown~cognitive~dimension~#1}}
7801 }
7802 }

7803 \seq_if_empty:NT \g_stex_current_file {
7804   \seq_gset_eq:NN \g_stex_current_file \c_stex_main_file
7805 }
7806 \_stex_persist_read_now:
7807 \_stex_every_file:
7808 \cs_new_protected:Nn \__todo_newlabel:n {
7809   \exp_args:N\__todo_old_newlabel:{\tl_to_str:n{#1}}
7810 }
7811 \AtBeginDocument{
7812   \iow_now:Nn \@auxout {
7813     \ExplSyntaxOn
7814     \let\__todo_old_newlabel:\newlabel
7815     \let\newlabel\__todo_newlabel:n
7816     \ExplSyntaxOff
7817   }
7818 }
7819 
```

# Chapter 14

## Additional Packages

### 14.1 Implementation: The `notesslides` Package

#### 14.1.1 Class and Package Options

We define some Package Options and switches for the `notesslides` class and activate them by passing them on to `beamer.cls` and `omdoc.cls` and the `notesslides` package. We pass the `nontheorem` option to the `statements` package when we are not in notes mode, since the `beamer` package has its own (overlay-aware) theorem environments.

```
7820 <*cls>
7821 <@=notesslides>
7822 \ProvidesExplClass{notesslides}{2024/10/26}{4.0.0}{notesslides Class}
7823 \RequirePackage{l3keys2e}
7824
7825 \str_const:Nn \c__notesslides_class_str {article}
7826
7827 \keys_define:nn{notesslides / cls}{
7828   class .str_set_x:N = \c__notesslides_class_str,
7829   notes .bool_set:N = \c__notesslides_notes_bool ,
7830   slides .code:n    = { \bool_set_false:N \c__notesslides_notes_bool },
7831   %docopt .str_set_x:N = \c__notesslides_docopt_str,
7832   unknown .code:n   = {
7833     \PassOptionsToClass{\CurrentOption}{beamer}
7834     \PassOptionsToClass{\CurrentOption}{\c__notesslides_class_str}
7835     \PassOptionsToPackage{\CurrentOption}{notesslides}
7836     \PassOptionsToPackage{\CurrentOption}{stex}
7837   }
7838 }
7839 \ProcessKeysOptions{ notesslides / cls }
7840
7841 \RequirePackage{stex}
7842 \stex_if_html_backend:T {
7843   \bool_set_true:N \c__notesslides_notes_bool
7844 }
7845
7846 \bool_if:NTF \c__notesslides_notes_bool {
7847   \PassOptionsToPackage{notes=true}{notesslides}
7848   \message{notesslides.cls:-Formatting-document-in-notes-mode}
```

```

7849 }{
7850   \PassOptionsToPackage{notes=false}{notesslides}
7851   \message{notesslides.cls:~Formatting~document~in~slides~mode}
7852 }
7853
7854 \bool_if:NTF \c_notesslides_notes_bool {
7855   \LoadClass{\c__notesslides_class_str}
7856 }{
7857   \LoadClass[10pt,notheorems,xcolor={dvipsnames,svgnames}]{beamer}
7858   \%newcounter{Item}
7859   \%newcounter{paragraph}
7860   \%newcounter{ subparagraph}
7861   \%newcounter{Hfootnote}
7862 }
7863 \RequirePackage{notesslides}
7864 
```

now we do the same for the `notesslides` package.

```

7865 <*package>
7866 \ProvidesExplPackage{notesslides}{2024/10/26}{4.0.0}{notesslides Package}
7867 \RequirePackage{l3keys2e}
7868
7869 \keys_define:nn{notesslides / pkg}{
7870   notes          .bool_set:N  = \c_notesslides_notes_bool ,
7871   slides         .code:n    = { \bool_set_false:N \c_notesslides_notes_bool },
7872   sectocframes  .bool_set:N  = \c_notesslides_sectocframes_bool ,
7873   topsect        .str_set_x:N = \c_notesslides_topsect_str,
7874   unknown        .code:n    = {
7875     \PassOptionsToPackage{\CurrentOption}{stex}
7876     \PassOptionsToPackage{\CurrentOption}{tikzinput}
7877   }
7878 }
7879 \ProcessKeysOptions{ notesslides / pkg }
7880
7881 \RequirePackage{stex}
7882 \stex_if_html_backend:T {
7883   \bool_set_true:N\c_notesslides_notes_bool
7884 }
7885
7886 \cs_set:Npn \sectiontitleemph #1 {
7887   \textbf{\Large #1}
7888 }
7889
7890 \newif\ifnotes
7891 \bool_if:NTF \c_notesslides_notes_bool {
7892   \notestrue
7893   \PassOptionsToPackage{dvipsnames,svgnames}{xcolor}
7894   \RequirePackage[noamsthm,hyperref]{beamerarticle}
7895   \RequirePackage{mdframed}
7896   \str_if_empty:NTF \c_notesslides_topsect_str{
7897     \%setsectionlevel{section}
7898   }{
7899     \exp_args:No \setsectionlevel \c_notesslides_topsect_str
7900   }

```

```

7901 }{
7902   \notesfalse
7903
7904   \cs_new_protected:Nn \__notesslides_do_sectocframes: {
7905     \cs_set_protected:Nn \__notesslides_do_label:n {
7906       \str_case:nnF{##1} {
7907         {part} {
7908           \tl_set:Nx\l__notesslides_num{\the part}
7909           \tl_set:cx{@ @ label}{%
7910             \cs_if_exist:NTF\parttitlename{\exp_not:N\parttitlename}{\exp_not:N\partname}{-}}
7911         }
7912         {chapter} {
7913           \tl_set:Nx\l__notesslides_num{\the chapter}
7914           \tl_set:cx{@ @ label}{%
7915             \cs_if_exist:NTF\chapertitlename{\exp_not:N\chapertitlename}{\exp_not:N\chaptername}{-}}
7916         }
7917         {section} {
7918           \tl_set:Nx\l__notesslides_num{\cs_if_exist:NT\thechapter{\the chapter.}\thesection.}
7919           \tl_set:cx{@ @ label}{\l__notesslides_num\quad}
7920         }
7921         {subsection} {
7922           \tl_set:Nx\l__notesslides_num{\cs_if_exist:NT\thechapter{\the chapter.}\thesubsection.}
7923           \tl_set:cx{@ @ label}{\l__notesslides_num\quad}
7924         }
7925         {subsubsection} {
7926           \tl_set:Nx\l__notesslides_num{\cs_if_exist:NT\thechapter{\the chapter.}\thesubsubsection.}
7927           \tl_set:cx{@ @ label}{\l__notesslides_num\quad}
7928         }
7929         {paragraph} {
7930           \tl_set:Nx\l__notesslides_num{\cs_if_exist:NT\thechapter{\the chapter.}\thesubparagraph.}
7931           \tl_set:cx{@ @ label}{\l__notesslides_num\quad}
7932         }
7933       }{
7934         \tl_set:Nx\l__notesslides_num{\cs_if_exist:NT\thechapter{\the chapter.}\thesubsubsubsection.}
7935         \tl_set:cx{@ @ label}{\l__notesslides_num\quad}
7936       }
7937     }
7938     \cs_set_protected:Nn \_sfragment_do_level:nn {
7939       \tl_if_exist:cT{c@##1}{\stepcounter{##1}}
7940       \addcontentsline{toc}{##1}{\protect\numberline{\use:c{the##1}}##2}
7941       \__notesslides_do_label:n{##1}
7942       \pdfbookmark[\int_use:N \l_stex_docheader_sect]{\l__notesslides_num##2}{##1.\l__note}
7943       \begin{frame}[noframenumbering]
7944         \vfill\centering
7945         \sectiontitleemph{%
7946           \use:c{@ @ label} ##2
7947         }
7948       \end{frame}
7949       \int_incr:N \l_stex_docheader_sect
7950       \tl_set:Nn \stex_current_section_level{##1}
7951     }
7952   }
7953
7954   \cs_set_protected:Nn \_stex_titlefragment: {

```

```

7955   \begin{frame}[noframenumbering]\maketitle\end{frame}
7956   \let\maketitle\relax
7957 }
7958
7959 \AtBeginDocument{
7960   \str_if_empty:NTF \c_notesslides_topsect_str {
7961     \setsectionlevel{section}
7962   } {
7963     \exp_args:No \setsectionlevel \c_notesslides_topsect_str
7964     \exp_args:No \str_if_eq:nnTF \c_notesslides_topsect_str {chapter} {
7965       \__notesslides_define_chapter:
7966     }{
7967       \exp_args:No \str_if_eq:nnT \c_notesslides_topsect_str {part} {
7968         \__notesslides_define_chapter:
7969         \__notesslides_define_part:
7970       }
7971     }
7972   }
7973 }
7974
7975 \bool_if:NT \c_notesslides_sectocframes_bool {
7976   \__notesslides_do_sectocframes:
7977 }
7978 }

7979 \cs_new_protected:Nn \__notesslides_define_chapter: {
7980   \cs_if_exist:NF \chaptername {
7981     \cs_set_protected:Npn \chaptername {Chapter}
7982   }
7983   \cs_if_exist:NF \chapter {
7984     \cs_set_protected:Npn \chapter {INVALID}
7985   }
7986   \cs_if_exist:NF \c@chapter {
7987     \newcounter{chapter}\counterwithin*{section}{chapter}
7988   }
7989 }
7990 }

7991 \cs_new_protected:Nn \__notesslides_define_part: {
7992   \cs_if_exist:NF \partname {
7993     \cs_set_protected:Npn \partname {Part}
7994   }
7995   \cs_if_exist:NF \part {
7996     \cs_set_protected:Npn \part {INVALID}
7997   }
7998   \cs_if_exist:NF \c@part {
7999     \newcounter{part}\counterwithin*{chapter}{part}
8000   }
8001 }
8002 }

```

\prematurestop We initialize \afterprematurestop, and provide \prematurestop@endsfragment which looks up \sfragment@level and recursively ends enough {sfragment}s.

```

8003 \def \c_notesslides_document_str{document}
8004 \newcommand\afterprematurestop{}
8005 \def\prematurestop@endsfragment{

```

```

8006 \unless\ifx\@currenvir\c__notesslides_document_str
8007   \expandafter\expandafter\expandafter\end\expandafter\expandafter\expandafter{\expandafter
8008   \expandafter\prematurestop@endsfragment
8009 \fi
8010 }
8011 \providecommand\prematurestop{
8012   \stex_if_html_backend:F{
8013   \message{Stopping~sTeX~processing~prematurely}
8014   \prematurestop@endsfragment
8015   \afterprematurestop
8016   \end{document}
8017 }
8018 }

```

(End of definition for `\prematurestop`. This function is documented on page 111.)

### 14.1.2 Notes and Slides

For the notes case, we also provide the `\usetheme` macro that would otherwise come from the the `beamer` class.

```

8019 \bool_if:NT \c__notesslides_notes_bool {
8020   \renewcommand\usetheme[2][]{\usepackage[#1]{beamertheme#2}}
8021 }
8022 \NewDocumentCommand \libusetheme {O{} m} {
8023   \libusepackage[#1]{beamertheme#2}
8024 }

```

We define the sizes of slides in the notes. Somehow, we cannot get by with the same here.

```

8025 \newlength{\slidewidth}\setlength{\slidewidth}{13.5cm}
8026 \newlength{\slideheight}\setlength{\slideheight}{9cm}

```

We first set up the slide boxes in `notes` mode. We set up sizes and provide a box register for the frames and a counter for the slides.

```

8027 \ifnotes
8028
8029 \newlength{\slideframewidth}
8030 \setlength{\slideframewidth}{1.5pt}

```

`frame (env.)` We first define the keys.

```

8031 \cs_new_protected:Nn \__notesslides_do_yes_param:Nn {
8032   \exp_args:Nx \str_if_eq:nnTF { \str_uppercase:n{ #2 } }{ yes }{
8033     \bool_set_true:N #1
8034   }{
8035     \bool_set_false:N #1
8036   }
8037 }
8038
8039 \stex_keys_define:nnnn{notesslides / frame}{
8040   \str_clear:N \l__notesslides_frame_label_str
8041   \bool_set_true:N \l__notesslides_frame_allowframebreaks_bool
8042   \bool_set_true:N \l__notesslides_frame_allowdisplaybreaks_bool
8043   \bool_set_true:N \l__notesslides_frame_fragile_bool
8044   \bool_set_true:N \l__notesslides_frame_shrink_bool

```

```

8045 \bool_set_true:N \l__notesslides_frame_squeeze_bool
8046 \bool_set_true:N \l__notesslides_frame_t_bool
8047 }{
8048   label           .str_set_x:N = \l__notesslides_frame_label_str,
8049   allowframebreaks .code:n      = {
8050     \__notesslides_do_yes_param:Nn \l__notesslides_frame_allowframebreaks_bool { #1 }
8051   },
8052   allowdisplaybreaks .code:n     = {
8053     \__notesslides_do_yes_param:Nn \l__notesslides_frame_allowdisplaybreaks_bool { #1 }
8054   },
8055   fragile         .code:n      = {
8056     \__notesslides_do_yes_param:Nn \l__notesslides_frame_fragile_bool { #1 }
8057   },
8058   shrink          .code:n      = {
8059     \__notesslides_do_yes_param:Nn \l__notesslides_frame_shrink_bool { #1 }
8060   },
8061   squeeze         .code:n      = {
8062     \__notesslides_do_yes_param:Nn \l__notesslides_frame_squeeze_bool { #1 }
8063   },
8064   t               .code:n      = {
8065     \__notesslides_do_yes_param:Nn \l__notesslides_frame_t_bool { #1 }
8066   },
8067   unknown         .code:n      = {}
8068 }{}}

```

We redefine the `itemize` environment so that it looks more like the one in `beamer`.

```

8069 \cs_new_protected:Nn \__notesslides_setup_itemize: {
8070   \def\itemize@level{outer}
8071   \def\itemize@outer{outer}
8072   \def\itemize@inner{inner}
8073   \%newcommand\metakeys@show@keys[2]{\marginnote{{\scriptsize ##2}}}
8074   \renewenvironment{itemize}){
8075     \ifx\itemize@level\itemize@outer
8076       \def\itemize@label{$\rhd$}
8077     \fi
8078     \ifx\itemize@level\itemize@inner
8079       \def\itemize@label{$\scriptstyle\rhd$}
8080     \fi
8081     \begin{list}
8082       {\itemize@label}
8083       {\setlength{\labelsep}{.3em}}
8084       {\setlength{\labelwidth}{.5em}}
8085       {\setlength{\leftmargin}{1.5em}}
8086     }
8087     \edef\itemize@level{\itemize@inner}
8088   }{
8089     \end{list}
8090   }
8091 }

```

We create the box with the `mdframed` environment from the equinymous package.

```

8092 \stex_if_html_backend:TF {
8093   \stex_css_literal:n{
8094     .stex-frame {

```

```

8095     background-color:#fff;
8096     margin:5px ~ 0;
8097     border:2px ~ solid ~ #000;
8098     border-radius:5px;
8099     padding:5px;
8100     padding-right:10px;
8101     width: var(--rustex-curr-width);
8102     display:block;
8103   }
8104 }
8105
8106 \cs_new_protected:Nn \__notesslides_frame_box_begin: {
8107   \vbox\bgroun
8108   \begin{stex_annotation_env}{class:stex-frame={}}
8109     \mdf@patchamsthm\notesslidesfont
8110   }
8111 \cs_new_protected:Nn \__notesslides_frame_box_end: {
8112   %^^A \notesslides@slidelabel
8113   \medskip\par\noindent\tiny\notesslidesfooter
8114   \end{stex_annotation_env}\egroup
8115 }
8116 }{
8117 \cs_new_protected:Nn \__notesslides_frame_box_begin: {
8118   \begin{mdframed}[
8119     linewidth=\slideframewidth,
8120     skipabove=1ex,
8121     skipbelow=1ex,
8122     userdefinedwidth=\slidewidth,
8123     align=center
8124   ]\notesslidesfont
8125 }
8126 \cs_new_protected:Nn \__notesslides_frame_box_end: {
8127   \medskip\par\noindent\tiny\notesslidesfooter%^^A\notesslides@slidelabel
8128   \end{mdframed}
8129 }
8130 }

```

We define the environment, read them, and construct the slide number and label.

```

8131 \renewenvironment{frame}[1][]{
8132   \stex_keys_set:nn{notesslides / frame}{#1}
8133   \stepcounter{framenumber}
8134   \renewcommand\newpage{\addtocounter{framenumber}{1}}
8135   \def\@currentlabel{\theframenumber}
8136   \str_if_empty:NF \l__notesslides_frame_label_str {
8137     \label{\l__notesslides_frame_label_str}
8138   }
8139   \__notesslides_setup_itemize:
8140   \__notesslides_frame_box_begin:
8141 }{
8142   \__notesslides_frame_box_end:
8143 }

```

Now, we need to redefine the frametitle (we are still in course notes mode).

```

\frametitle
8144 \renewcommand{\frametitle}[1]{
8145   \stexdoctitle { #1 }
8146   \notesslidestitleref{\#1}\medskip
8147 }

(End of definition for \frametitle. This function is documented on page ??.)
```

```

\pause
8148 \newcommand\pause{}
```

(End of definition for \pause. This function is documented on page ??.)  
We redefine the `columns` and `column` environments:

```

8149 \renewenvironment{columns}[1] []{
8150   \par\noindent
8151   \begin{minipage}
8152     \slidewidth\centering\leavevmode
8153 %   \stex_if_html_backend:T{
8154 %     \cs_if_exist:NT \rustex_if:T {
8155 %       \rustex_if:T {\par
8156 %         \rustex_direct_HTML:n{<table><tr><td>}
8157 %       }
8158 %     }
8159 %   }
8160 }{
8161 %   \stex_if_html_backend:T{
8162 %     \cs_if_exist:NT \rustex_if:T {
8163 %       \rustex_if:T {\par
8164 %         \rustex_direct_HTML:n{</td></tr></table>}
8165 %       }
8166 %     }
8167 %   }
8168   \end{minipage}\par\noindent
8169 }
8170 \newsavebox\columnbox
8171 \renewenvironment<>{column}[2] []{
8172   \begin{lrbox}{\columnbox}
8173 %   \stex_if_html_backend:T{
8174 %     \cs_if_exist:NT \rustex_if:T {
8175 %       \rustex_if:T {\par
8176 %         \rustex_direct_HTML:n{</td><td>}
8177 %       }
8178 %     }
8179 %   }
8180   \begin{minipage}{#2}
8181 }{
8182   \end{minipage}
8183 %   \stex_if_html_backend:T{
8184 %     \cs_if_exist:NT \rustex_if:T {
8185 %       \rustex_if:T {\par
8186 %         \rustex_direct_HTML:n{</td><td>}
8187 %       }
8188 %     }
8189 %   }
```

```

8190     \end{lrbox}\usebox\columnbox
8191 }
8192 \fi

```

### 14.1.3 Environment and Macro Patches

The `note` environment is used to leave out text in the `slides` mode. It does not have a counterpart in OMDoc. So for course notes, we define the `note` environment to be a no-operation otherwise we declare the `note` environment to produce no output.

```

8193 \bool_if:NTF \c_notesslides_notes_bool {
8194     \renewenvironment{note}{\ignorespaces}{}
8195 }{
8196     \renewenvironment{note}{\setbox \l_tmpa_box\vbox\bgroup}{\egroup}
8197 }

```

For other environments we introduce variants prefixed with `n`, which are excluded in `slides` mode.

```

8198 \cs_new_protected:Nn \__notesslides_notes_env:nnnn {
8199     \bool_if:NTF \c_notesslides_notes_bool {
8200         \newenvironment{\#1}\#2{\#3}{\#4}
8201 }{
8202     \newenvironment{\#1}\#2{
8203         \cs_set:Npn \__notesslides_eat: #####1 \end #####2 {
8204             \str_if_eq:nnTF{\#1}{#####2} {
8205                 \end{\#1}
8206             }{
8207                 \__notesslides_eat:
8208             }
8209         }
8210         \__notesslides_eat:
8211         \%setbox\l_tmpa_box\vbox\bgroup\#3
8212 }{
8213     \%#4\egroup
8214 }
8215 }
8216 }
8217
8218 \__notesslides_notes_env:nnnn{nparagraph}{[1]}{(\begin{sparagraph}{\#1})}{(\end{sparagraph})}
8219 \__notesslides_notes_env:nnnn{nfragment}{[2]}{(\begin{sfragment}{\#1}{\#2})}{(\end{sfragment})}
8220 \__notesslides_notes_env:nnnn{ndefinition}{[1]}{(\begin{sdefinition}{\#1})}{(\end{sdefinition})}
8221 \__notesslides_notes_env:nnnn{nassertion}{[1]}{(\begin{sassertion}{\#1})}{(\end{sassertion})}
8222 \__notesslides_notes_env:nnnn{nproof}{[2]}{(\begin{sproof}{\#1}{\#2})}{(\end{sproof})}
8223 \__notesslides_notes_env:nnnn{nexample}{[1]}{(\begin{sexample}{\#1})}{(\end{sexample})}

8224
8225 \RequirePackage{graphicx}
8226
8227 \NewDocumentCommand\frameimage{s O{} m} {
8228     \IfBooleanTF #1 {
8229         \begin{frame}[plain]
8230     }{
8231         \begin{frame}
8232     }
8233     \bool_if:NTF \c_notesslides_notes_bool {

```

```

8234     \slidewidth=\dimexpr\slidewidth-(2\slideframewidth)\relax
8235     }{
8236         \slidewidth=\textwidth\relax
8237     }
8238     \def\Gin@ewidth{}\setkeys{Gin}{#2}
8239     \tl_if_empty:NTF \Gin@ewidth {
8240         \mhgraphics[width=\slidewidth,#2]{#3}
8241     }{
8242         \mhgraphics[#2]{#3}
8243     }
8244     \end{frame}
8245 }

```

hacking inputref:

```

\inputref*
8246 \cs_set_eq:NN\__notesslides_inputref:\inputref
8247 \cs_set_protected:Npn\inputref{\@ifstar\ninputref\__notesslides_inputref:}
8248 \bool_if:NTF \c_notesslides_notes_bool {
8249     \newcommand\ninputref[2][]{\__notesslides_inputref:[#1]{#2}}
8250     \__notesslides_inputref:[#1]{#2}
8251 }
8252 }{
8253     \newcommand\ninputref[2][]{}
8254 }

```

(End of definition for `\inputref*`. This function is documented on page 110.)

#### 14.1.4 Styling Across Notes/Slides

```

8255 \def\notesslidestitleemph#1{
8256     {\Large\bf\sf#1}
8257     \vskip0.1\baselineskip
8258     \leaders\hrule width \textwidth
8259     \vskip0.4pt%
8260     \nointerlineskip
8261 }
8262
8263 \def\notesslidesfooter{}
8264
8265 \let\notesslidesfont\sffamily

```

#### 14.1.5 Beamer Compatibility

All of this should be removed and made part of a template

```

8266 \stex_if_do_html:TF{
8267     \NewDocumentEnvironment{slideshow}{}{
8268         \begin{stex_annotation_env}{data-shtml-slideshow={}}
8269         \cs_set_protected:Npn \nextslide ##1 {
8270             \begin{stex_annotation_env}{data-shtml-slideshow-slide={}} ##1
8271             \end{stex_annotation_env}
8272         }
8273     \cs_set_protected:Npn \lastslide ##1 {
8274         \begin{stex_annotation_env}{data-shtml-slideshow-slide={}} ##1
8275     }

```

```

8276      \end{stex_annotation_env}
8277    }
8278  }{
8279    \end{stex_annotation_env}
8280  }
8281 }{
8282   \int_new:N \l__notesslides_slideshow_counter_int
8283   \NewDocumentEnvironment{slideshow}{}
8284     \int_zero:N \l__notesslides_slideshow_counter_int
8285     \cs_set_protected:Npn \nextslide ##1 {
8286       \int_incr:N \l__notesslides_slideshow_counter_int
8287       \only<\int_use:N \l__notesslides_slideshow_counter_int>{##1}
8288     }
8289     \cs_set_protected:Npn \lastslide ##1 {
8290       \int_incr:N \l__notesslides_slideshow_counter_int
8291       \only<\int_use:N \l__notesslides_slideshow_counter_int ->{##1}
8292     }
8293 }{
8294
8295 }
8296 }
8297
8298 \bool_if:NT \c_notesslides_notes_bool {
8299   \def\author{\@dblarg\ns@author}
8300   \long\def\ns@author[#1]#2{%
8301     \tl_if_empty:nTF{#1}{
8302       \def\beamer@shortauthor{#2}
8303     }{
8304       \def\beamer@shortauthor{#1}
8305     }
8306     \def\@author{#2}
8307   }
8308   \def\title{\@dblarg\ns@title}
8309   \long\def\ns@title[#1]#2{%
8310     \tl_if_empty:nTF{#1}{
8311       \def\beamer@shorttitle{#2}
8312     }{
8313       \def\beamer@shorttitle{#1}
8314     }
8315     \def\@title{#2}
8316     \stexdotitle{#2}
8317   }
8318   \def\insertshortauthor{
8319     \hbox\bgroup\def\\{}\cs_if_exist:NT\beamer@shortauthor\beamer@shortauthor\egroup
8320   }
8321   \def\insertshorttitle{
8322     \hbox\bgroup\def\\{}\cs_if_exist:NT\beamer@shorttitle\beamer@shorttitle\egroup
8323   }
8324   \stex_if_html_backend:TF{
8325     \def\insertframenumber{\stex_annotation:nn{data-shtml-framenumber={}{}}
8326   }{
8327     \def\insertframenumber{\@arabic\c@framenumber}
8328   }
8329   \def\insertshortdate{\today}

```

```
8330 }
```

### 14.1.6 TODO Excursions

- \excursion The excursion macros are very simple, we define a new internal macro \excursionref and use it in \excursion, which is just an \inputref that checks if the new macro is defined before formatting the file in the argument.

```
8331 \gdef\printexcursions{}  
8332 \newcommand\excursionref[2]{% label, text  
8333   \bool_if:NT \c_notesslides_notes_bool {  
8334     \begin{sparagraph}[title=Excursion]  
8335       #2 \sref[fallback=the appendix]{#1}.  
8336     \end{sparagraph}  
8337   }  
8338 }  
8339 \newcommand\activate@excursion[2][]{  
8340   \tl_gput_right:Nn\printexcursions{\inputref[#1]{#2}}  
8341 }  
8342 \newcommand\excursion[4][]{% repos, label, path, text  
8343   \bool_if:NT \c_notesslides_notes_bool {  
8344     \activate@excursion[#1]{#3}  
8345     \excursionref[#2]{#4}  
8346   }  
8347 }
```

(End of definition for \excursion. This function is documented on page 112.)

```
\excursiongroup
```

```
8348 \keys_define:nn{notesslides / excursiongroup }{  
8349   id      .str_set_x:N = \l__notesslides_excursion_id_str,  
8350   intro    .tl_set:N    = \l__notesslides_excursion_intro_tl,  
8351   archive  .str_set_x:N = \l__notesslides_excursion_mhrepos_str  
8352 }  
8353 \cs_new_protected:Nn \__notesslides_excursion_args:n {  
8354   \tl_clear:N \l__notesslides_excursion_intro_tl  
8355   \str_clear:N \l__notesslides_excursion_id_str  
8356   \str_clear:N \l__notesslides_excursion_mhrepos_str  
8357   \keys_set:nn {notesslides / excursiongroup }{ #1 }  
8358 }  
8359 \newcommand\excursiongroup[1][]{  
8360   \__notesslides_excursion_args:n{ #1 }  
8361   \tl_if_empty:NF\printexcursions  
8362   {\IfInputref{}{\begin{note}  
8363     \begin{sfragment}{Excursions}% TODO pass on id  
8364     \ifdefempty\l__notesslides_excursion_intro_tl{}{  
8365       \exp_args:NNe \use:nn \inputref{[\l__notesslides_excursion_mhrepos_str]}{  
8366         \l__notesslides_excursion_intro_tl  
8367       }  
8368     }  
8369     \printexcursions%  
8370     \end{sfragment}  
8371   \end{note}}}  
8372 }  
8373 \ifcsname beameritemnestingprefix\endcsname\else\def\beameritemnestingprefix{}\fi
```

(End of definition for \excursiongroup. This function is documented on page 112.)

```
8374 \prop_new:N \g__notesslides_variables_prop
8375 \cs_set_protected:Npn \setSGvar #1 #2 {
8376   \prop_gput:Nnn \g__notesslides_variables_prop {#1}{#2}
8377 }
8378 \cs_set_protected:Npn \useSGvar #1 {
8379   \prop_item:Nn \g__notesslides_variables_prop {#1}
8380 }
8381 \cs_set_protected:Npn \ifSGvar #1 #2 #3 {
8382   \prop_get:NnNF \g__notesslides_variables_prop {#1} \l__notesslides_tmp {
8383     \PackageError{document-structure}
8384     {The sTeX Global variable #1 is undefined}
8385     {set it with \protect\setSGvar}\TODO better error
8386   }
8387   \tl_if_eq:NnT \l__notesslides_tmp {#2}{#3}
8388 }
8389
8390
8391 </package>
```

## 14.2 Implementation: The problem Package

### 14.2.1 Package Options

The first step is to declare (a few) package options that handle whether certain information is printed or not. They all come with their own conditionals that are set by the options.

```
8392 <*package>
8393 <@=problems>
8394 \ProvidesExplPackage{problem}{2024/10/26}{4.0.0}{Semantic Markup for Problems}
8395 \RequirePackage{l3keys2e}
8396
8397 \keys_define:nn { problem / pkg }{
8398   notes      .default:n    = { true },
8399   notes      .bool_set:N  = \c__problems_notes_bool,
8400   gnotes     .default:n    = { true },
8401   gnotes     .bool_set:N  = \c__problems_gnotes_bool,
8402   hints      .default:n    = { true },
8403   hints      .bool_set:N  = \c__problems_hints_bool,
8404   solutions  .default:n    = { true },
8405   solutions  .bool_set:N  = \c__problems_solutions_bool,
8406   pts        .default:n    = { true },
8407   pts        .bool_set:N  = \c__problems_pts_bool,
8408   min        .default:n    = { true },
8409   min        .bool_set:N  = \c__problems_min_bool,
8410   %boxed     .default:n    = { true },
8411   %boxed     .bool_set:N  = \c__problems_boxed_bool,
8412   test       .default:n    = { true },
8413   test       .bool_set:N  = \c__problems_test_bool,
8414   unknown    .code:n      = {
8415     \PassOptionsToPackage{\CurrentOption}{stex}
8416 }
```

```

8417 }
8418 \newif\ifsolutions
8419
8420 \ProcessKeysOptions{ problem / pkg }
8421 \bool_if:NTF \c__problems_solutions_bool {
8422   \solutionstrue
8423 }{
8424   \solutionsfalse
8425 }
8426 \newif\ifintest
8427 \bool_if:NTF \c__problems_test_bool {
8428   \intesttrue
8429 }{
8430   \intestfalse
8431 }
8432
8433 \RequirePackage{stex}

```

\problem@kw@\* For multilinguality, we define internal macros for keywords that can be specialized in \*.ldf files.

```

8434 \AddToHook{begindocument}{
8435   \ExplSyntaxOn\makeatletter
8436   \input{problem-english.ldf}
8437   \ltx@ifpackageloaded{babel}{
8438     \clist_set:Nx \l_tmpa_clist {\exp_args:No \tl_to_str:n \bbl@loaded}
8439     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{ngerman}}{
8440       \input{problem-ngerman.ldf}
8441     }
8442     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{finnish}}{
8443       \input{problem-finnish.ldf}
8444     }
8445     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{french}}{
8446       \input{problem-french.ldf}
8447     }
8448     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{russian}}{
8449       \input{problem-russian.ldf}
8450     }
8451   }{}}
8452   \makeatother\ExplSyntaxOff
8453 }

```

(End of definition for \problem@kw@\*. This function is documented on page ??.)

### 14.2.2 Problems and Solutions

We now prepare the KeyVal support for problems. The key macros just set appropriate internal macros.

```

8454 \bool_new:N \l_stex_key_autogradable_bool
8455 \stex_keys_define:nnnn{ problem }{
8456   \tl_set:Nn \l_stex_key_pts_tl 0
8457   \tl_set:Nn \l_stex_key_min_tl 0
8458   \str_clear:N \l_stex_key_name_str
8459   \str_clear:N \l_stex_key_mhrepos_str

```

```

8460   \bool_set_false:N \l_stex_key_autogradable_bool
8461 }{
8462   pts .tl_set:N = \l_stex_key_pts_tl,
8463   min .tl_set:N = \l_stex_key_min_tl,
8464   name .str_set:N = \l_stex_key_name_str,
8465   autogradable .bool_set:N = \l_stex_key_autogradable_bool,
8466   archive .code:n = {},
8467   %archive .str_set:N = \l_stex_key_mhrepos_str,
8468   creators .code:n = {}
8469   %imports .tl_set:N = \l__problems_prob_imports_tl,
8470   %refnum .int_set:N = \l__problems_prob_refnum_int,
8471 }{id,title,style,uses}

```

Then we set up a counter for problems.

\numberproblemsin

```

8472 \newcounter{sproblem}[section]
8473 \newcommand\numberproblemsin[1]{
8474   \@addtoreset{sproblem}{#1}
8475   \def\thesproblem{\arabic{#1}.\arabic{sproblem}}
8476 }
8477 \numberproblemsin{section}
8478 %\def\theplainsproblem{\arabic{sproblem}}
8479 %\def\thesproblem{\thesection.\theplainsproblem}

```

(End of definition for \numberproblemsin. This function is documented on page ??.)

sproblem (env.)

```

8480 \cs_new:Nn \__problems_activate_macros: {
8481   \stex_reactivate_macro:N \solution
8482   \stex_reactivate_macro:N \mcb
8483   \stex_reactivate_macro:N \scb
8484   \stex_reactivate_macro:N \fillinsol
8485   \stex_reactivate_macro:N \hint
8486   \stex_reactivate_macro:N \exnote
8487   \stex_reactivate_macro:N \gnote
8488 }
8489
8490 \fp_new:N \g_problem_total_pts_fp
8491 \fp_new:N \g_problem_total_min_fp
8492
8493 \bool_new:N \l__problems_in_problem_bool
8494 \bool_new:N \l__problems_has_pts_bool
8495 \bool_new:N \l__problems_has_min_bool
8496 \bool_set_false:N \l__problems_in_problem_bool
8497 \stex_new_stylable_env:nnnnnn {problem} {O{}}
8498   \bool_if:NT \l__problems_in_problem_bool {
8499     \msg_error:nn{stex}{error/nestedproblem}
8500   }
8501   \cs_if_exist:NTF \l_problem_inputproblem_keys_tl {
8502     \tl_put_left:Nn \l_problem_inputproblem_keys_tl {#1,}
8503     \exp_args:Nno \stex_keys_set:nn {problem} {
8504       \l_problem_inputproblem_keys_tl
8505     }
8506   }

```

```

8507     \stex_keys_set:nn{problem}{#1}
8508 }
8509 \refstepcounter{sproblem}
8510
8511 \stex_if_do_html:T {
8512   \str_if_empty:NT \l_stex_key_name_str {
8513     \stex_file_split_off_lang:NN \l__problems_path_seq \g_stex_current_file
8514     \seq_get_right:NN \l__problems_path_seq \l_stex_key_name_str
8515   }
8516   \exp_args:Nne \begin{stex_annotation_env} {
8517     data-shtml-problem={\l_stex_key_name_str},
8518     data-shtml-language={\l_stex_current_language_str},
8519     data-shtml-autogradable={\bool_if:NTF \l_stex_key_autogradable_bool {true}{false}}
8520   }
8521   \tl_if_empty:NF \l_stex_key_title_tl {
8522     \exp_args:No \stexdoctitle \l_stex_key_title_tl
8523   }
8524   \stex_annotation_invisible:nn{
8525     data-shtml-problempoints={\l_stex_key_pts_tl}
8526   }{ \l_stex_key_pts_tl }
8527 }
8528 \tl_set_eq:NN \thistitle \l_stex_key_title_tl
8529
8530
8531 \bool_set_true:N \l__problems_in_problem_bool
8532 \tl_set_eq:NN \l__problems_pts_tl \l_stex_key_pts_tl
8533 \tl_set_eq:NN \l__problems_min_tl \l_stex_key_min_tl
8534 \tl_if_eq:NnTF \l__problems_pts_tl {0}
8535   {\bool_set_false:N \l__problems_has_pts_bool}
8536   {\bool_set_true:N \l__problems_has_pts_bool}
8537 \tl_if_eq:NnTF \l__problems_min_tl {0}
8538   {\bool_set_false:N \l__problems_has_min_bool}
8539   {\bool_set_true:N \l__problems_has_min_bool}
8540 \int_gzero:N \g__problems_subproblem_int
8541
8542 \stex_style_apply:
8543 \stex_do_id:
8544 \__problems_activate_macros:
8545 }{
8546   \fp_gadd:Nn \g_problem_total_pts_fp { \l__problems_pts_tl}
8547   \fp_gadd:Nn \g_problem_total_min_fp { \l__problems_min_tl}
8548 \__problems_record_problem:
8549 \stex_style_apply:
8550 \stex_if_do_html:T{ \end{stex_annotation_env} }
8551 }{
8552 \par\noindent\problemheader
8553 \stex_if_do_html:F{
8554   \bool_if:NT \c__problems_pts_bool {
8555     \tl_if_eq:NnF \l__problems_pts_tl {0} {
8556       \marginpar{\l__problems_pts_tl{}~\problem@kw@pts\smallskip}
8557     }
8558   }
8559   \bool_if:NT \c__problems_min_bool {
8560     \tl_if_eq:NnF \l__problems_min_tl {0} {

```

```

8561      \marginpar{\l__problems_min_tl{}~\problem@kw@minutes\smallskip}
8562    }
8563  }
8564 }
8565 \par
8566 \stex_ignore_spaces_and_pars:
8567 }{
8568 \par\bigskip
8569 % \bool_if:NT \c__problems_test_bool \pagebreak
8570 }{s}

8571 \tl_set:Nn \problemheader {
8572   \stex_if_do_html:TF{
8573     \tl_if_empty:NF \thistitle {
8574       \stex_annotate:nn{data-shtml-title={}}{\textbf{\thistitle}}
8575     }
8576   }{
8577     \textbf{\sproblemautorefname{}\thesproblem}
8578     \tl_if_empty:NF \thistitle {
8579       {\~{}}(\thistitle)
8580     }
8581   }
8582 }
8583 }
8584 }
8585
8586 \cs_new_protected:Nn \__problems_record_problem: {
8587   \exp_args:NNe \iow_now:Nn \@auxout {
8588     \problem@restore {\thesproblem}{\l__problems_pts_tl}{\l__problems_min_tl}
8589   }
8590 }
8591
8592 \cs_new_protected:Npn \problem@restore #1 #2 #3 {}

subproblem (env.)
8593 \int_new:N \g__problems_subproblem_int
8594
8595 \stex_new_stylable_env:nnnnnn {subproblem} {0{}){
8596   \stex_keys_set:nn{problem}{#1}
8597   \bool_if:NF \l__problems_in_problem_bool{
8598     \ifstexhtml\else
8599       \par{\bfseries WARNING~subproblem~to~be~used~in~some~problem}\par
8600     \fi
8601   \__problems_activate_macros:
8602   \bool_set_true:N \l__problems_in_problem_bool
8603   \tl_set:Nn \l__problems_pts_tl{0}
8604   \tl_set:Nn \l__problems_min_tl{0}
8605 }
8606 \str_if_empty:NT \l_stex_key_name_str {
8607   \stex_file_split_off_lang:NN \l__problems_path_seq \g_stex_current_file
8608   \seq_get_right:NN \l__problems_path_seq \l_stex_key_name_str
8609 }
8610 \stex_if_do_html:T {
8611   \str_if_empty:NT \l_stex_key_name_str {
8612     \stex_file_split_off_lang:NN \l__problems_path_seq \g_stex_current_file

```

```

8613     \seq_get_right:NN \l__problems_path_seq \l_stex_key_name_str
8614 }
8615 \exp_args:Nne \begin{stex_annotation_env} {
8616     data-shtml-subproblem={\l_stex_key_name_str},
8617     data-shtml-language={\l_stex_current_language_str},
8618     data-shtml-autogradable={\bool_if:NTF \l_stex_key_autogradable_bool {true}{false}}
8619 }
8620 \stex_annotation_invisible:n{}
8621 \tl_if_empty:NF \l_stex_key_title_tl {
8622     \exp_args:No \stexdoctitle \l_stex_key_title_tl
8623 }
8624 }
8625 \int_gincr:N \g__problems_subproblem_int
8626 \bool_if:NF \l__problems_has_pts_bool {
8627     \tl_gset:Nx \l__problems_pts_tl {\fp_to_decimal:n {\l__problems_pts_tl + \l_stex_key_pts}
8628 }
8629 \bool_if:NF \l__problems_has_min_bool {
8630     \tl_gset:Nx \l__problems_min_tl {\fp_to_decimal:n {\l__problems_min_tl + \l_stex_key_min}
8631 }
8632 \stex_if_smsmode:F \stex_style_apply:
8633 }{
8634 \stex_if_smsmode:F \stex_style_apply:
8635 \stex_if_do_html:T{ \end{stex_annotation_env} }
8636 }{
8637 \begin{list}{}{
8638     \setlength\topsep{0pt}
8639     \setlength\parsep{0pt}
8640     \setlength\rightmargin{0pt}
8641 }\item[\int_use:N \g__problems_subproblem_int .]
8642 \bool_if:NT \c__problems_pts_bool {
8643     \bool_if:NF \l__problems_has_pts_bool {
8644         \marginpar{\smallskip\l_stex_key_pts_tl{}~\problem@kw@pts}
8645     }
8646 }
8647 \bool_if:NT \c__problems_min_bool {
8648     \bool_if:NF \l__problems_has_min_bool{
8649         \marginpar{\smallskip\l_stex_key_min_tl{}~\problem@kw@minutes}
8650     }
8651 }
8652 }{
8653     \end{list}
8654 }{}}

\includeproblem

8655 \stex_keys_define:nnnn{ includeproblem }{
8656     \str_clear:N \l_stex_key_mhrepos_str
8657 }{
8658     archive .str_set:N      = \l_stex_key_mhrepos_str,
8659     unknown .code:n = {}
8660 }{}

8661 \NewDocumentCommand\includeproblem{O{} m}{
8662     \group_begin:
8663     \tl_set:Nn \l_problem_inputproblem_keys_tl {#1}

```

```

8665   \stex_keys_set:nn{includeproblem}{#1}
8666   \exp_args:Nno \use:nn{\inputref[]}\l_stex_key_mhrepos_str}{#2}
8667   \group_end:
8668 }
8669

```

(End of definition for `\includeproblem`. This function is documented on page 117.)

`solution (env.)`

```

8670 \int_new:N \g_problem_id_counter
8671 \dim_new:N \l_stex_key_testsphere_dim
8672 \stex_keys_define:nmmn{ solution }{
8673   \str_clear:N \l_stex_key_answerclass_str
8674   \dim_zero:N \l_stex_key_testsphere_dim
8675 }{
8676   testspace .dim_set:N = \l_stex_key_testsphere_dim,
8677   answerclass .str_set:N = \l_stex_key_answerclass_str
8678 }{id,title,style}
8679
8680 \cs_new_protected:Nn \__problems_solution_start:n {
8681   \stex_keys_set:nn{ solution }{#1}
8682   \str_if_empty:NT \l_stex_key_id_str {
8683     \int_gincr:N \g_problem_id_counter
8684     \str_set:Nx \l_stex_key_id_str {
8685       SOLUTION_ \int_use:N \g_problem_id_counter
8686     }
8687   }
8688   \stex_if_do_html:T{
8689     \begin{stex_annotation_env}{
8690       data-shtml-solution=\l_stex_key_id_str,
8691       data-shtml-answerclass={\l_stex_key_answerclass_str}
8692     }
8693   }
8694   \stex_style_apply:
8695 }
8696
8697 \stex_new_stylable_env:nnnnnnn { solution }{ 0{} }{
8698   \stex_if_do_html:TF{
8699     \__problems_solution_start:n{#1}
8700   }{
8701     \if solutions
8702       \__problems_solution_start:n{#1}
8703     \else
8704       \stex_keys_set:nn{ solution }{#1}
8705       \testspace{\l_stex_key_testsphere_dim}
8706       \setbox\l_tmpa_box\vbox\bgroup
8707       \fi
8708   }
8709 }
8710 \stex_if_do_html:TF{
8711   \stex_style_apply:
8712   \end{stex_annotation_env}
8713 }{
8714   \if solutions

```

```

8715     \stex_style_apply:
8716     \stex_if_do_html:T{
8717         \end{stex_annotation_env}
8718     }
8719     \else
8720         \egroup
8721     \fi
8722 }
8723 }{
8724     \par\smallskip\rule[.3em]{\linewidth}{0.4pt}\newline\smallskip
8725     \noindent\emph{\problem@kw@solution\tl_if_empty:N \l_stex_key_title_tl{
8726         \l_stex_key_title_tl \str_if_empty:N \l_stex_key_answerclass_str {
8727             (Answer Class: \l_stex_key_answerclass_str)
8728         }
8729     } :~}
8730 }{
8731     \par\rule[.3em]{\linewidth}{0.4pt}\newline
8732 }){}
8733
8734 \stex_deactivate_macro:Nn \solution {sproblem~environments}

\startproblems
\stopproblems
8735 \cs_new_protected:Npn \startproblems{
8736     \global\solutiontrue
8737 }
8738 \cs_new_protected:Npn \stopproblems{
8739     \global\solutionsfalse
8740 }

```

(End of definition for `\startproblems` and `\stopproblems`. These functions are documented on page 114.)

`hint (env.)`

```

8741
8742 \stex_keys_define:nnnn{ problemenv }{}{}{id,title,style}
8743
8744 \cs_new_protected:Nn \__problems_hint_start:n {
8745     \stex_keys_set:nn{ problemenv }{\#1}
8746     \str_if_empty:NT \l_stex_key_id_str {
8747         \int_gincr:N \g_problem_id_counter
8748         \str_set:Nx \l_stex_key_id_str {
8749             HINT_\int_use:N \g_problem_id_counter
8750         }
8751     }
8752     \stex_if_do_html:T{
8753         \begin{stex_annotation_env}{
8754             data-shtml-problemhint=\l_stex_key_id_str
8755         }
8756     }
8757     \stex_style_apply:
8758 }
8759
8760 \stex_new_stylable_env:nnnnnnn { hint }{ 0{} }{
8761     \stex_if_do_html:TF{

```

```

8762     \__problems_hint_start:n{#1}
8763 }{
8764     \bool_if:NTF \c__problems_hints_bool {
8765         \__problems_hint_start:n{#1}
8766     }{
8767         \setbox\l_tmpa_box\vbox\bgroup
8768     }
8769 }
8770 }{
8771     \stex_if_do_html:TF{
8772         \stex_style_apply:
8773         \end{stex_annotation_env}
8774 }{
8775     \bool_if:NTF \c__problems_hints_bool {
8776         \stex_style_apply:
8777         \stex_if_do_html:T{
8778             \end{stex_annotation_env}
8779         }
8780     }{
8781         \egroup
8782     }
8783 }
8784 }{
8785     \par\smallskip\rule[.3em]{\linewidth}{0.4pt}\newline\smallskip
8786     \noindent\emph{\problem@kw@hint\tl_if_empty:N \l_stex_key_title_tl{
8787         \~{}\l_stex_key_title_tl
8788     }\~{}}
8789 }{
8790     \par\rule[.3em]{\linewidth}{0.4pt}\newline
8791 }{}
8792 \stex_deactivate_macro:Nn \hint {sproblem~environments}

exnote (env.)
8793 \cs_new_protected:Nn \__problems_exnote_start:n {
8794     \stex_keys_set:nn{ problemenv }{#1}
8795     \str_if_empty:NT \l_stex_key_id_str {
8796         \int_gincr:N \g_problem_id_counter
8797         \str_set:Nx \l_stex_key_id_str {
8798             EXNOTE_\int_use:N \g_problem_id_counter
8799         }
8800     }
8801     \stex_if_do_html:T{
8802         \begin{stex_annotation_env}{
8803             data-shtml-problemnote=\l_stex_key_id_str
8804         }
8805     }
8806     \stex_style_apply:
8807 }
8808
8809 \stex_new_stylable_env:nnnnnnn { exnote }{ 0{} }{
8810     \stex_if_do_html:TF{
8811         \__problems_exnote_start:n{#1}
8812 }{
8813     \bool_if:NTF \c__problems_notes_bool {

```

```

8814     \__problems_exnote_start:n{#1}
8815     }{
8816         \setbox\l_tmpa_box\vbox\bgroup
8817     }
8818 }
8819 }{
8820     \stex_if_do_html:TF{
8821         \stex_style_apply:
8822         \end{stex_annotation_env}
8823 }{
8824     \bool_if:NTF \c__problems_notes_bool {
8825         \stex_style_apply:
8826         \stex_if_do_html:T{
8827             \end{stex_annotation_env}
8828         }
8829     }{
8830         \egroup
8831     }
8832 }
8833 }{
8834     \par\smallskip\rule[.3em]{\linewidth}{0.4pt}\newline\smallskip
8835     \noindent\emph{\problem@kw@note\tl_if_empty:N \l_stex_key_title_tl{
8836         {-}\l_stex_key_title_tl
8837     } :~}
8838 }{
8839     \par\rule[.3em]{\linewidth}{0.4pt}\newline
8840 }{}}
8841 \stex_deactivate_macro:Nn \exnote {sproblem~environments}

gnote (env.)
8842 \int_new:N \l__problems_anscls_int
8843
8844 \cs_new_protected:Nn \__problems_gnote_start:n {
8845     \stex_keys_set:nn{ problemenv }{#1}
8846     \str_if_empty:NT \l_stex_key_id_str {
8847         \int_gincr:N \g_problem_id_counter
8848         \str_set:Nx \l_stex_key_id_str {
8849             GNOTE_\int_use:N \g_problem_id_counter
8850         }
8851     }
8852
8853     \stex_if_do_html:T{
8854         \begin{stex_annotation_env}{
8855             data-shtml-problemgnote=\l_stex_key_id_str
8856         }
8857     }
8858     \stex_style_apply:
8859 }
8860
8861 \stex_new_stylable_env:nnnnnnn { gnote }{ 0{} }{
8862     \stex_if_do_html:TF{
8863         \__problems_gnote_start:n{#1}
8864 }{
8865     \bool_if:NTF \c__problems_gnotes_bool {

```

```

8866     \__problems_gnote_start:n{#1}
8867     }{
8868         \setbox\l_tmpa_box\vbox\bgroup
8869     }
8870 }
8871 \stex_reactivate_macro:N \anscls
8872 }{
8873 \stex_if_do_html:TF{
8874     \stex_style_apply:
8875     \end{stex_annotation_env}
8876 }{
8877     \bool_if:NTF \c__problems_gnotes_bool {
8878         \stex_style_apply:
8879         \stex_if_do_html:T{
8880             \end{stex_annotation_env}
8881         }
8882     }{
8883         \egroup
8884     }
8885 }
8886 }{
8887     \par\smallskip\rule[.3em]{\linewidth}{0.4pt}\newline\smallskip
8888     \noindent\emph{\problem@kw@grading\str_if_empty:NF \l_stex_key_title_tl{
8889         \l_stex_key_title_tl
8890     }} :~}
8891 }{
8892     \par\rule[.3em]{\linewidth}{0.4pt}\newline
8893 }{}}
8894 \stex_deactivate_macro:Nn \gnote {sproblem~environments}
8895
8896
8897 \stex_keys_define:nnnn{ anscls }{
8898     \str_clear:N \l_stex_key_pts_str
8899     \tl_clear:N \l_stex_key_feedback_tl
8900 }{
8901     pts .str_set:N = \l_stex_key_pts_str,
8902     feedback .tl_set:N = \l_stex_key_feedback_tl,
8903     update .code:n = {}
8904 }{id}
8905 \newcommand \anscls [2][]{ {
8906     \stex_keys_set:nn{ anscls }{#1}
8907     \str_if_empty:NT \l_stex_key_id_str {
8908         \int_incr:N \l__problems_anscls_int
8909         \str_set:Nx \l_stex_key_id_str {
8910             AC\int_use:N \l__problems_anscls_int
8911         }
8912     }
8913     \begin{list}{}{
8914         \setlength\topsep{0pt}
8915         \setlength\parsep{0pt}
8916         \setlength\rightmargin{0pt}
8917     }\item[\l_stex_key_id_str]
8918     \stex_if_do_html:TF{
8919         \exp_args:Ne \stex_annotation:nn{

```

```

8920     data-shtml-answerclass={\l_stex_key_id_str}
8921     \str_if_empty:NF \l_stex_key_pts_str{
8922         ,data-shtml-answerclass-pts={\l_stex_key_pts_str}
8923     }
8924     }{
8925         #2
8926         \tl_if_empty:NF \l_stex_key_feedback_tl{
8927             \stex_annotation_invisible:nn{
8928                 data-shtml-answerclass-feedback={true}
8929             }{\l_stex_key_feedback_tl}
8930         }
8931     }
8932     }{
8933         #2
8934     }
8935     \str_if_empty:NF \l_stex_key_pts_str {\par
8936         ~ \problem@kw@points :~\l_stex_key_pts_str
8937     }
8938     \tl_if_empty:NF \l_stex_key_feedback_tl {\par
8939         ~ \problem@kw@feedback :~\l_stex_key_feedback_tl
8940     }
8941     \end{list}
8942 }
8943 \stex_deactivate_macro:Nn \anscls {gnote~environments}

```

The margin pars are reader-visible, so we need to translate

```

8944 \def\pts#1{
8945     \bool_if:NT \c__problems_pts_bool {
8946         \stex_annotation:nn{data-shtml-problempoints={#1}}{\marginpar{#1~\problem@kw@pts}}
8947     }\hbox_unpack:N\c_empty_box
8948 }
8949 \def\min#1{
8950     \bool_if:NT \c__problems_min_bool {
8951         \stex_annotation:nn{data-shtml-problemminutes={}}{\marginpar{#1~\problem@kw@minutes}}
8952     }\hbox_unpack:N\c_empty_box
8953 }

```

*mcb (env.)*

```

8954 \stex_new_stylable_env:nnnnnn{mcb}{0{}>{
8955     \stex_keys_set:nn{style}{#1}
8956     \stex_if_do_html:T{
8957         \tl_set:Nn\problem_mcc_box_tl{}
8958         \exp_args:Nne \begin{stex_annotation_env}{
8959             data-shtml-multiple-choice-block={}
8960             \clist_if_empty:NF \l_stex_key_style_clist ,,
8961                 data-shtml-styles={\l_stex_key_style_clist}
8962             }
8963         }
8964     }
8965     \stex_deactivate_macro:Nn \mcb {sproblem~environments}
8966     \stex_deactivate_macro:Nn \scb {sproblem~environments}
8967     \stex_deactivate_macro:Nn \solution {sproblem~environments}
8968     \stex_deactivate_macro:Nn \hint {sproblem~environments}
8969     \stex_deactivate_macro:Nn \exnote {sproblem~environments}

```

```

8970 \stex_deactivate_macro:Nn \gnote {sproblem~environments}
8971 \stex_reactivate_macro:N \mcc
8972 \cs_set:Nn \__problems_mccline:n{
8973   \begin{list}{}{
8974     \setlength\topsep{0pt}
8975     \setlength\parsep{0pt}
8976     \setlength\rightmargin{0pt}
8977   }\item[\problem_mcc_box_tl] ##1 \end{list}
8978 }
8979 \stex_style_apply:
8980 }{
8981   \stex_style_apply:
8982   \stex_if_do_html:T{
8983     \end{stex_annotation_env}
8984   }
8985 }{\par}{}
8986 \stexstylemcb[inline]{
8987   {\Large[]}\cs_set:Nn \__problems_mccline:n{\problem_mcc_box_tl{~} #1}
8988 }{\Large[]}}
8989
8990 \stex_deactivate_macro:Nn \mcc {sproblem~environments}
we define the keys for the mcc macro
8991 \cs_new_protected:Nn \__problems_do_yes_param:Nn {
8992   \exp_args:Nx \str_if_eq:nnTF { \str_lowercase:n{ #2 } }{ yes }{
8993     \bool_set_true:N #1
8994   }{
8995     \bool_set_false:N #1
8996   }
8997 }
8998 \stex_keys_define:nnnn{mcc}{
8999   \tl_clear:N \l_stex_key_feedback_tl
9000   \bool_set_false:N \l_stex_key_T_bool
9001   \tl_clear:N \l_stex_key_Ttext_tl
9002   \tl_clear:N \l_stex_key_Ftext_tl
9003 }{
9004   feedback .tl_set:N      = \l_stex_key_feedback_tl ,
9005   T       .code:n      = {\bool_set_true:N \l_stex_key_T_bool} ,
9006   F       .code:n      = {\bool_set_false:N \l_stex_key_T_bool} ,
9007   Ttext   .tl_set:N      = \l_stex_key_Ttext_tl ,
9008   Ftext   .tl_set:N      = \l_stex_key_Ftext_tl ,
9009 }{id}
9010
\mcc
9011
9012 \stex_if_html_backend:TF{
9013   \tl_set:Nn \problem_mcc_box_tl {
9014     \ltx@ifpackageloaded{amssymb}{$\square$}{
9015       \hbox{\vrule\vbox{\hrule width 6 pt\vskip 6pt\hrule}\vrule}
9016     }
9017   }
9018 \newcommand\mcc[2][]{\stex_keys_set:nn{mcc}{#1}}
9019

```

```

9020   \tl_set:Nn \l_tmpa_tl{
9021     \bool_if:NTF \l_stex_key_T_bool {
9022       \tl_if_empty:NTF \l_stex_key_Ttext_tl \problem@kw@correct \l_stex_key_Ttext_tl
9023     }{
9024       \tl_if_empty:NTF \l_stex_key_Ftext_tl \problem@kw@wrong \l_stex_key_Ftext_tl
9025     }
9026     \tl_if_empty:NF \l_stex_key_feedback_tl {
9027       \\\emph{\l_stex_key_feedback_tl}
9028     }
9029   }
9030
9031   \__problems_mccline:n{
9032     \stex_if_do_html:T{
9033       \stex_annotate:nn{data-shtml-mcc={%
9034         \bool_if:NTF \l_stex_key_T_bool {true}{false}
9035       }{}%
9036         #2\stex_annotate:nn{data-shtml-mcc-solution={}}{\l_tmpa_tl}
9037       }%
9038     }
9039   }
9040 }
9041 }{
9042   \tl_set:Nn \problem_mcc_box_default_tl {
9043     \ltx@ifpackageloaded{amssymb}{$\square$}%
9044       \hbox{\vrule\vbox{\hrule width 6 pt\vskip 6pt\hrule}\vrule}%
9045   }
9046 }
9047 \tl_set:Nn \problem_mcc_box_tl {
9048   \ifsolutions
9049     \bool_if:NTF \l_stex_key_T_bool {
9050       \makebox[0pt][1]{\problem_mcc_box_default_tl}
9051       \hspace{0.1em}\cs_if_exist:NTF\checkmark{%
9052         \raisebox{.15ex}{\checkmark}
9053       } X%
9054     }{\problem_mcc_box_default_tl}
9055   \else
9056     \problem_mcc_box_default_tl
9057   \fi
9058 }
9059 \newcommand\mcc[2][]{%
9060   \stex_keys_set:nn{mcc}{#1}
9061   \ifsolutions
9062     \tl_set:Nx \l_tmpa_tl{
9063       \bool_if:NTF \l_stex_key_T_bool {
9064         \tl_if_empty:NF \l_stex_key_Ttext_tl {\exp_not:N \l_stex_key_Ttext_tl}
9065       }{
9066         \tl_if_empty:NF \l_stex_key_Ftext_tl {\exp_not:N \l_stex_key_Ftext_tl}
9067       }
9068       \tl_if_empty:NF \l_stex_key_feedback_tl {
9069         \exp_not:n{\\\emph{\l_stex_key_feedback_tl}}
9070       }
9071     }
9072   \fi
9073 }
```

```

9074     \__problems_mccline:n{ #2
9075         \ifsolutions
9076             \tl_if_empty:NF \l_tmpa_tl { \footnote{\l_tmpa_tl} }
9077             \fi
9078         }
9079     }
9080 }
9081 \stex_deactivate_macro:Nn \mcc {mcb~environments}

```

(End of definition for \mcc. This function is documented on page 115.)

**scb** (*env.*)

```

9082
9083 \cs_new_protected:Nn \__problems_maybe_inline:n {
9084     \clist_if_in:NnTF \l_stex_key_style_clist {inline} {
9085         \let\__problems_oldpar:\stex_par:
9086         \cs_set:Nn\stex_par:{}}
9087         #1
9088         \let\stex_par:\__problems_oldpar:
9089     }{#1}
9090 }
9091
9092 \stex_new_stylable_env:nnnnnnn{scb}{0{}}
9093     \stex_keys_set:nn{style}{#1}
9094     \stex_if_do_html:T{
9095         \__problems_maybe_inline:n{
9096             \exp_args:Nne\begin{stex_annotation_env}{
9097                 data-shtml-single-choice-block={}
9098                 \clist_if_empty:NF \l_stex_key_style_clist ,,
9099                     data-shtml-styles={\l_stex_key_style_clist}
9100                 }
9101             }
9102         }
9103         \tl_set:Nn\problem_scc_box_tl{}
9104     }
9105 \stex_deactivate_macro:Nn \mcb {sproblem~environments}
9106 \stex_deactivate_macro:Nn \scb {sproblem~environments}
9107 \stex_deactivate_macro:Nn \solution {sproblem~environments}
9108 \stex_deactivate_macro:Nn \hint {sproblem~environments}
9109 \stex_deactivate_macro:Nn \exnote {sproblem~environments}
9110 \stex_deactivate_macro:Nn \gnote {sproblem~environments}
9111 \stex_reactivate_macro:N \scc
9112 \cs_set:Nn \__problems_sccline:n{
9113     \begin{list}{}{
9114         \setlength\topsep{0pt}
9115         \setlength\parsep{0pt}
9116         \setlength\rightmargin{0pt}
9117         }\item[\problem_scc_box_tl] ##1 \end{list}
9118     }
9119     \stex_style_apply:
9120 }
9121 \stex_style_apply:
9122 \stex_if_do_html:T{
9123     \__problems_maybe_inline:n { \end{stex_annotation_env} }

```

```

9124     }
9125 }{\par}{}
9126 \stexstylescb[inline]{
9127   {\Large[]}\cs_set:Nn \__problems_sccline:n{\problem_scc_box_tl{~} #1}
9128 }{\Large]}
9129
9130 \stex_deactivate_macro:Nn \scb {sproblem-environments}

\scc

9131
9132 \stex_if_html_backend:TF{
9133   \tl_set:Nn\problem_scc_box_tl{$\bigcirc$}
9134   \newcommand\scc[2][]{%
9135     \stex_keys_set:nn{mcc}{#1}
9136     \tl_set:Nn \l_tmpa_tl{
9137       \bool_if:NTF \l_stex_key_T_bool {
9138         \tl_if_empty:NTF \l_stex_key_Ttext_tl \problem@kw@correct \l_stex_key_Ttext_tl
9139       }{
9140         \tl_if_empty:NTF \l_stex_key_Ftext_tl \problem@kw@wrong \l_stex_key_Ftext_tl
9141       }
9142       \tl_if_empty:NF \l_stex_key_feedback_tl {
9143         \\\emph{\l_stex_key_feedback_tl}
9144       }
9145     }
9146
9147   \__problems_sccline:n{
9148     \stex_if_do_html:T{
9149       \stex_annotation:nn{data-shtml-scc={%
9150         \bool_if:NTF \l_stex_key_T_bool {true}{false}
9151       }}{
9152         #2\stex_annotation:nn{data-shtml-scc-solution={}}{\l_tmpa_tl}
9153       }
9154     }
9155   }
9156 }
9157 }{
9158   \tl_set:Nn\problem_scc_box_default_tl{$\bigcirc$}
9159   \tl_set:Nn\problem_scc_box_tl{
9160     \if solutions
9161       \bool_if:NTF \l_stex_key_T_bool {
9162         \makebox[0pt][l]{\problem_scc_box_default_tl}
9163         \hspace{0.1em}$\backslash$cs_if_exist:NTF\checkmark{
9164           \raisebox{.15ex}{\checkmark}
9165           } X$
9166         }{\problem_scc_box_default_tl}
9167       \else
9168         \problem_scc_box_default_tl
9169       \fi
9170   }
9171   \newcommand\scc[2][]{%
9172     \stex_keys_set:nn{mcc}{#1}
9173     \if solutions
9174       \tl_set:Nx \l_tmpa_tl{
9175         \bool_if:NTF \l_stex_key_T_bool {

```

```

9176     \tl_if_empty:N \l_stex_key_Ttext_tl {\exp_not:N \l_stex_key_Ttext_tl}
9177     }{
9178         \tl_if_empty:N \l_stex_key_Ftext_tl {\exp_not:N \l_stex_key_Ftext_tl}
9179     }
9180     \tl_if_empty:N \l_stex_key_feedback_tl {
9181         \exp_not:n{ \\\emph{\l_stex_key_feedback_tl} }
9182     }
9183 }
9184 \fi
9185
9186 \__problems_sccline:n{ #2
9187     \ifsolutions
9188         \tl_if_empty:N \l_tmpa_tl { \footnote{\l_tmpa_tl} }
9189     \fi
9190 }
9191 }
9192 }
9193
9194 \stex_deactivate_macro:Nn \scc {scb~environments}
9195
9196
9197 \newcommand\yesTnoF{
9198     \begin{scb}[style=inline]
9199         \scc[T]{yes}~\scc[F]{no}
9200     \end{scb}
9201 }
9202 \newcommand\yesFnoT{
9203     \begin{scb}[style=inline]
9204         \scc[F]{yes}~\scc[T]{no}
9205     \end{scb}
9206 }
9207 \newcommand\trueTfalseF{
9208     \begin{scb}[style=inline]
9209         \scc[T]{true}~\scc[F]{false}
9210     \end{scb}
9211 }
9212 \newcommand\trueFfalseT{
9213     \begin{scb}[style=inline]
9214         \scc[F]{true}~\scc[T]{false}
9215     \end{scb}
9216 }

```

(End of definition for \scc. This function is documented on page ??.)

```
\fillinsol
9217 \stex_keys_define:nnnn{fillinsol}{
9218     \tl_clear:N \l__problems_fillin_solution_tl
9219     \dim_zero:N \l_stex_key_testsphere_dim
9220 }{
9221     testspace .dim_set:N = \l_stex_key_testsphere_dim,
9222     exact .code:n = {\__problems_parse_fillin_arg:nnnn{exact}\#1 },
9223     numrange .code:n = {\__problems_parse_fillin_arg:nnnn{numrange}\#1 },
9224     regex .code:n = {\__problems_parse_fillin_arg:nnnn{regex}\#1 }
9225 }{}{}
```

```

9226 \stex_if_html_backend:TF{
9227   \cs_new:Nn \__problems_parse_fillin_arg:nnnn {
9228     \tl_set:Nn \l_tmpa_tl{#3}
9229     \tl_set:Nn \l_tmpb_tl{T}
9230     \stex_annotate_invisible:nn{
9231       data-shtml-fillin-case={#1},
9232       data-shtml-fillin-case-value={#2},
9233       data-shtml-fillin-case-verdict={
9234         \tl_if_eq:NNTF\l_tmpa_tl\l_tmpb_tl{true}{false}
9235       },
9236     }{#4}
9237   }
9238 }
9239 }{
9240   \cs_new:Nn \__problems_parse_fillin_arg:nnnn {
9241     \tl_put_right:Nn \l__problems_fillin_solution_tl {
9242       #1 & #2 & #3 & #4 \crr
9243     }
9244   }
9245 }
9246
9247
9248 \newcommand\fillinsol[2][]{%
9249   \stex_if_do_html:F \quad
9250   \mode_if_math:TF{
9251     \hbox{\__problems_fillinsol:nn{#1}{#2}}
9252   }{
9253     \__problems_fillinsol:nn{#1}{#2}
9254   }
9255   \stex_if_do_html:F \quad
9256 }
9257
9258 \stex_if_html_backend:TF{
9259   \cs_new_protected:Nn \__problems_fillinsol:nn {
9260     \exp_args:Ne \stex_annotate:nn{data-shtml-fillinsol=true}{}{
9261       \stex_keys_set:nn{fillinsol}{#1}
9262       \stex_annotate_force_break:n{
9263         #2
9264       }
9265       \l__problems_fillin_solution_tl
9266     }
9267   }{
9268   }{
9269   \cs_new_protected:Nn \__problems_fillinsol:nn {
9270     \stex_keys_set:nn{fillinsol}{#1}
9271     \if solutions
9272       \textcolor{red}{\fbox{#2}}
9273       \tl_if_empty:NF \l__problems_fillin_solution_tl {
9274         \footnote{
9275           \halign{ ~##~\hfil & ~##~\hfil & ~##~\hfil & ~##~\hfil \cr
9276             \textbf{type}&\textbf{case}&\textbf{verdict}&\textbf{feedback}\cr
9277             \l__problems_fillin_solution_tl
9278           }
9279         }
9300       }
9301     }
9302   }
9303 }
```

```

9280     }
9281 \else
9282     \fbox{\dim_compare:nNnTF\l_stex_key_testspace_dim={0pt}{
9283         \phantom{\huge{#2}}
9284     }{
9285         \hspace{\l_stex_key_testspace_dim}\vphantom{\huge{A #2}}
9286     }}
9287 \fi
9288 }
9289 }
9290 \stex_deactivate_macro:Nn \fillinsol {sproblem-environments}

```

(End of definition for `\fillinsol`. This function is documented on page 117.)

### `\testemptypage`

```

9292 \newcommand\testemptypage[1][]{%
9293   \bool_if:NT \c__problems_test_bool {\vfill\begin{center}\hwexam@kw@testemptypage\end{center}}
9294 }

```

(End of definition for `\testemptypage`. This function is documented on page ??.)

### `\testspace`

```

9295 \newcommand\testspace[1]{\bool_if:NT \c__problems_test_bool {\vspace*{#1}}}
9296 \newcommand\testsmallspace{\testspace{1cm}}
9297 \newcommand\testmedspace{\testspace{2cm}}
9298 \newcommand\testbigspace{\testspace{3cm}}

```

(End of definition for `\testspace`. This function is documented on page ??.)

### `\testnewpage`

```

9299 \newcommand\testnewpage{\bool_if:NT \c__problems_test_bool {\newpage}}

```

(End of definition for `\testnewpage`. This function is documented on page ??.)

```

9300 
```

## 14.3 Implementation: The `hwexam` Package

### 14.3.1 Package Options

The first step is to declare (a few) package options that handle whether certain information is printed or not. Some come with their own conditionals that are set by the options, the rest is just passed on to the `problems` package.

```

9301 <*package>
9302 \ProvidesExplPackage{hwexam}{2024/10/26}{4.0.0}{homework assignments and exams}
9303 \RequirePackage{l3keys2e}
9304
9305 \keys_define:nn {hwexam / pkg}{
9306   multiple .default:n = { false },
9307   multiple .bool_set:N = \c_hwexam_multiple_bool,
9308   qrcode .default:n = { false },
9309   qrcode .bool_set:N = \c_hwexam_qrcode_bool,
9310   unknown .code:n = {

```

```

9311     \PassOptionsToPackage{\CurrentOption}{problem}
9312 }
9313 }
9314 \ProcessKeysOptions{ hwexam /pkg }
9315 \RequirePackage{problem}

\hwexam_kw_* For multilinguality, we define internal macros for keywords that can be specialized in *.ldf files.
9316 \AddToHook{begindocument}{
9317   \ExplSyntaxOn\makeatletter
9318   \input{hwexam-english.ldf}
9319   \ltx@ifpackageloaded{babel}{
9320     \clist_set:Nx \l_tmpa_clist {\exp_args:No \tl_to_str:n \bblobloaded}
9321     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{ngerman}}{
9322       \input{hwexam-ngerman.ldf}
9323     }
9324     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{finnish}}{
9325       \input{hwexam-finnish.ldf}
9326     }
9327     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{french}}{
9328       \input{hwexam-french.ldf}
9329     }
9330     \exp_args:NNx \clist_if_in:NnT \l_tmpa_clist {\detokenize{russian}}{
9331       \input{hwexam-russian.ldf}
9332     }
9333   }{}
9334   \makeatother\ExplSyntaxOff
9335 }

```

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

### 14.3.2 QR Codes

```

9336 \group_begin:
9337   \escapechar=-1
9338   \xdef\_\@_qr_backslash{\string\\}
9339 \group_end:
9340
9341 \bool_if:NT \c_hwexam_qrcode_bool {
9342   \RequirePackage{qrcode}
9343   \RequirePackage{marginnote}
9344   \str_new:N \g_@@_qr_json_str
9345   \bool_new:N \g_@@_qr_in_problems_json_bool
9346   \bool_set_false:N \g_@@_qr_in_problems_json_bool
9347   \bool_new:N \g_@@_qr_in_subproblems_json_bool
9348   \bool_set_false:N \g_@@_qr_in_subproblems_json_bool
9349   \bool_new:N \g_@@_qr_in_anscls_json_bool
9350   \bool_set_false:N \g_@@_qr_in_anscls_json_bool
9351   \bool_if:NTF \c__problems_gnotes_bool {
9352     \gdef \qrjson { \str_gput_right:Nx \g_@@_qr_json_str}
9353   }{
9354     \gdef \qrjson #1 {}
9355   }
9356 \qrjson{[]}

```

```

9357 \AtEndDocument{\qrjson{}}
9358 \def\_\_qr\_escape\_char:n #1 {
9359   #1\exp_args:Nno\use:nn{\if_charcode:w#1}\_\_qr\_backslash#1\fi
9360 }
9361
9362 \gdef\_\_qr\_escape:n #1{\exp_args:Ne\str_map_function:nN{\tl_to_str:n{#1}}\_\_qr\_escape_c
9363 \gdef \_\_qr\_escape:o #1{\exp_args:No\_\_qr\_escape:n#1}
9364
9365 \def\qrschema{T0:D0:\examnumber}
9366
9367 \bool_if:NTF \c__problems_test_bool {
9368   \def\doproblemqr{
9369     \ifstexhtml\else{
9370       \reversemarginpar\marginnote{\qrcode[height=1.5cm]{\qrschema}}
9371       \normalmarginpar
9372     }\fi
9373   }
9374   \def\insertexamnumber{
9375     \ifstexhtml\else
9376       \tl_if_exist:NTF \examnumber {
9377         {\Large\bfseries ID:~\examnumber}
9378       }{
9379         {\color{red}\Large\bfseries!!!!~ WARNING:~NO~{\string\examnumber}~SET~!!!!}\gdef\examnumb
9380       }
9381       \global\def\insertexamnumber{}
9382     }\fi
9383   }
9384 }{
9385   \def\doproblemqr{}
9386   \def\insertexamnumber{}
9387 }
9388
9389 \stextyleproblem[noqr]{
9390   \par\noindent\problemheader \xdef\qrdf{\thesproblem}
9391   \bool_if:NT \c__problems_pts_bool {
9392     \tl_if_eq:NnF \l__problems_pts_tl {0} {
9393       \marginpar{\l__problems_pts_tl{}~\problem@kw@points\smallskip}
9394     }
9395   }
9396   \bool_if:NT \c__problems_min_bool {
9397     \tl_if_eq:NnF \l__problems_min_tl {0} {
9398       \marginpar{\l__problems_min_tl{}~\problem@kw@minutes\smallskip}
9399     }
9400   }
9401   \par
9402   \stex_ignore_spaces_and_pars:
9403 }{
9404   \par\bigskip
9405 }
9406
9407 \stextyleproblem{
9408   \par\noindent\problemheader \xdef\qrdf{\thesproblem}
9409   \doproblemqr
9410   \bool_if:NT \c__problems_pts_bool {

```

```

9411   \tl_if_eq:NnF \l__problems_pts_tl {0}{
9412     \marginpar{\l__problems_pts_tl{}~\problem@kw@points\smallskip}
9413   }
9414 }
9415 \bool_if:NT \c__problems_min_bool {
9416   \tl_if_eq:NnF \l__problems_min_tl {0} {
9417     \marginpar{\l__problems_min_tl{}~\problem@kw@minutes\smallskip}
9418   }
9419 }
9420
9421 \bool_if:NTF \g_@@_qr_in_problems_json_bool {
9422   \qrjson{,}
9423 }{
9424   \bool_gset_true:N \g_@@_qr_in_problems_json_bool
9425 }
9426
9427 \qrjson {
9428   \c_left_brace_str
9429   "id": "\_@@_qr_escape:o\l_stex_key_id_str", "title": "\_@@_qr_escape:o\l_stex_key_title_t1"
9430   "number": "\thesproblem"
9431 }
9432
9433 \tl_if_eq:NnF \l__problems_pts_tl {0}{
9434   \qrjson {
9435     , "pts": \l__problems_pts_tl
9436   }
9437 }
9438 \par
9439 \stex_ignore_spaces_and_pars:
9440 }{
9441 \bool_if:NT \g_@@_qr_in_subproblems_json_bool {
9442   \qrjson []
9443 }
9444 \bool_gset_false:N \g_@@_qr_in_subproblems_json_bool
9445 \qrjson {\c_right_brace_str}
9446 \par\bigskip
9447 }
9448
9449 \stexstylesubproblem[noqr]{
9450   \begin{list}{}{
9451     \setlength\topsep{0pt}
9452     \setlength\parsep{0pt}
9453     \setlength\rightmargin{0pt}
9454   }\item[\int_use:N \g__problems_subproblem_int .]
9455   \xdef\qrid{\thesproblem.\int_use:N \g__problems_subproblem_int}
9456 \bool_if:NT \c__problems_pts_bool {
9457   \bool_if:NF \l__problems_has_pts_bool {
9458     \marginpar{\smallskip\l_stex_key_pts_tl{}~\problem@kw@points}
9459   }
9460 }
9461 \bool_if:NT \c__problems_min_bool {
9462   \bool_if:NF \l__problems_has_min_bool{
9463     \marginpar{\smallskip\l_stex_key_min_tl{}~\problem@kw@minutes}
9464 }

```

```

9465   }
9466 }{\end{list}}
9467
9468 \stexstylesubproblem{
9469   \begin{list}{}{
9470     \setlength\topsep{0pt}
9471     \setlength\parsep{0pt}
9472     \setlength\rightmargin{0pt}
9473     }\item[\int_use:N \g__problems_subproblem_int .]
9474     \xdef\qrid{\thesproblem.\int_use:N \g__problems_subproblem_int}
9475     \doproblemqr
9476     \bool_if:NT \c__problems_pts_bool {
9477       \bool_if:NF \l__problems_has_pts_bool {
9478         \marginpar{\smallskip\l_stex_key_pts_tl{}~\problem@kw@points}
9479       }
9480     }
9481     \bool_if:NT \c__problems_min_bool {
9482       \bool_if:NF \l__problems_has_min_bool{
9483         \marginpar{\smallskip\l_stex_key_min_tl{}~\problem@kw@minutes}
9484       }
9485     }
9486     \bool_if:NTF \g_@@_qr_in_subproblems_json_bool {
9487       \qrjson {,}
9488     }{
9489       \qrjson {
9490         , "subproblems": [
9491           }
9492           \bool_gset_true:N \g_@@_qr_in_subproblems_json_bool
9493         }
9494       \qrjson {
9495         \c_left_brace_str
9496         "id": "\_@@_qr_escape:o\l_stex_key_id_str", "title": "\_@@_qr_escape:o\l_stex_key_title_tl"
9497         "number": "\thesproblem.\int_use:N \g__problems_subproblem_int"
9498       }
9499       \bool_if:NF \l__problems_has_pts_bool {
9500         \qrjson {
9501           , "pts": \l_stex_key_pts_tl
9502         }
9503       }{
9504         \qrjson {\c_right_brace_str}
9505       \end{list}
9506     }
9507   }
9508
9509 \stexstylelegnote{
9510   \par\smallskip\rule[.3em]{\linewidth}{0.4pt}\newline\smallskip
9511   \noindent\emph{\problem@kw@grading\str_if_empty:NF \l_stex_key_title_tl{}}\smallskip
9512   \{\sim\}\l_stex_key_title_tl
9513   \} : \{\sim\}
9514   \qrjson {
9515     , "gnote": \c_left_brace_str
9516     "id": "\_@@_qr_escape:o\l_stex_key_id_str", "title": "\_@@_qr_escape:o\l_stex_key_title_tl",
9517     "anscls": [
9518   }

```

```

9519  }{
9520   \qrjson {
9521     ]\c_right_brace_str
9522   }
9523   \par\rule[.3em]{\linewidth}{0.4pt}\newline
9524 }{}

9525
9526 \renewcommand \anscls [2][]{%
9527   \stex_keys_set:nn{ anscls }{\#1}
9528   \str_if_empty:NT \l_stex_key_id_str {
9529     \int_incr:N \l__problems_anscls_int
9530     \str_set:Nx \l_stex_key_id_str {
9531       AC\int_use:N \l__problems_anscls_int
9532     }
9533   }
9534   \bool_if:NTF \g_@@_qr_in_anscls_json_bool {
9535     \qrjson{,}
9536   }{
9537     \bool_set_true:N \g_@@_qr_in_anscls_json_bool
9538   }
9539 \qrjson{
9540   \c_left_brace_str
9541   "id": "\_@@_qr_escape:o\l_stex_key_id_str", "description": "\_@@_qr_escape:n{\#2}",
9542   "feedback": "\_@@_qr_escape:o\l_stex_key_feedback_tl",
9543   "pts": "\l_stex_key_pts_str"
9544   \c_right_brace_str
9545 }
9546 \begin{list}{}{%
9547   \setlength\topsep{0pt}
9548   \setlength\parsep{0pt}
9549   \setlength\rightmargin{0pt}
9550 } \item[\l_stex_key_id_str]
9551   \stex_if_do_html:TF{%
9552     \exp_args:Ne \stex_annotation:nn{%
9553       data-shtml-answerclass={\l_stex_key_id_str}
9554       \str_if_empty:NF \l_stex_key_pts_str{%
9555         ,data-shtml-answerclass-pts={\l_stex_key_pts_str}
9556       }
9557     }{
9558       #2
9559       \tl_if_empty:NF \l_stex_key_feedback_tl{%
9560         \stex_annotation_invisible:nn{%
9561           data-shtml-answerclass-feedback={true}
9562           }\{\l_stex_key_feedback_tl}
9563         }
9564       }
9565     }{\#2}
9566   \str_if_empty:NF \l_stex_key_pts_str { \par
9567     ~ \problem@kw@points :~\l_stex_key_pts_str
9568   }
9569   \str_if_empty:NF \l_stex_key_feedback_tl { \par
9570     ~ \problem@kw@feedback :~\l_stex_key_feedback_tl
9571   }
9572 \end{list}

```

```

9573   }
9574   \stex_deactivate_macro:Nn \anscls {gnote~environments}
9575
9576   \AtEndDocument{
9577     \message{^^J^^J\g_@@_qr_json_str^^J^^J}
9578     \bool_if:NT \c__problems_gnotes_bool {
9579       \iow_new:N \c_@@_qr_json_iow
9580       \iow_open:Nn \c_@@_qr_json_iow {\jobname-vollkorn.json}
9581       \iow_now:Nx \c_@@_qr_json_iow {\g_@@_qr_json_str}
9582       \iow_close:N \c_@@_qr_json_iow
9583     }
9584   }
9585
9586   \renewcommand\testemptypage[1][]{%
9587     \bool_if:NT \c__problems_test_bool {\
9588       \xdef\qrid{P\thepage}
9589       \doproblemqr
9590       \vfill\begin{center}\hwexam@kw@testemptypage\end{center}\eject
9591     }
9592   }
9593 }

```

### 14.3.3 Assignments

Then we set up a counter for problems and make the problem counter inherited from `problem.sty` depend on it. Furthermore, we specialize the `\prob@label` macro to take the assignment counter into account.

`assignment (env.)`

```

9594 \stex_keys_define:nnnn{ assignment }{
9595   \tl_clear:N \l_stex_key_number_tl
9596   \tl_clear:N \l_stex_key_given_tl
9597   \tl_clear:N \l_stex_key_due_tl
9598 }{
9599   number .tl_set:N      = \l_stex_key_number_tl,
9600   given  .tl_set:N      = \l_stex_key_given_tl,
9601   due    .tl_set:N      = \l_stex_key_due_tl,
9602   unknown .code:n = {}
9603 }{id,title,style}
9604
9605 \newcounter{assignment}
9606 \stex_new_stylable_env:nnnnnnn {assignment}{0{}){
9607   \cs_if_exist:NTF \l_hwexam_includeassignment_keys_tl {
9608     \tl_put_left:Nn \l_hwexam_includeassignment_keys_tl {#1,}
9609     \exp_args:Nno \stex_keys_set:nn{assignment}{
9610       \l_hwexam_includeassignment_keys_tl
9611     }
9612   }{
9613     \stex_keys_set:nn{assignment}{#1}
9614   }
9615   \tl_if_empty:NF \l_stex_key_number_tl {
9616     \global\setcounter{assignment}{\int_eval:n{\l_stex_key_number_tl-1}}
9617   }
9618   \global\refstepcounter{assignment}

```

```

9619 \setcounter{sproblem}{0}
9620 \def\thesproblem{\theassignment.\arabic{sproblem}}
9621 \stex_style_apply:
9622 \stex_do_id:
9623 }{
9624 \stex_style_apply:
9625 }{
9626 \par\begin{center}
9627 \textbf{\Large\assignmentautorefname~\theassignment}
9628 \tl_if_empty:NF \l_stex_key_title_tl {
9629 {\~}---\l_stex_key_title_tl
9630 }
9631 }\par\smallskip
9632 \textbf{
9633 \tl_if_empty:NF \l_stex_key_given_tl {
9634 \hwexam@kw@given :~\l_stex_key_given_tl\quad
9635 }
9636 \tl_if_empty:NF \l_stex_key_due_tl {
9637 \hwexam@kw@due :~\l_stex_key_due_tl\quad
9638 }
9639 }
9640 \end{center}
9641 \par\bigskip
9642 }{
9643 \par\pagebreak
9644 }{}}

```

#### \includeassignment

```

9645 \NewDocumentCommand\includeassignment{O{} m}{
9646 \group_begin:
9647 \tl_set:Nn \l_hwexam_includeassignment_keys_tl {#1}
9648 \stex_keys_set:nn{includeproblem}{#1}
9649 \exp_args:Nno \use:nn{\inputref[]}\l_stex_key_mhrepos_str]{#2}
9650 \group_end:
9651 }

```

(End of definition for \includeassignment. This function is documented on page 77.)

Restoring information about problems:

```

9652 \prop_new:N \c_@@_problems_prop
9653 \tl_set:Nn \c_@@_total_mins_tl {0}
9654 \tl_set:Nn \c_@@_total_pts_tl {0}
9655 \int_new:N \c_@@_total_problems_int
9656 \cs_set_protected:Npn \problem@restore #1 #2 #3 {
9657 \int_gincr:N \c_@@_total_problems_int
9658 \prop_gput:Nnn \c_@@_problems_prop {#1}{#2}{#3}
9659 \tl_gset:Nx \c_@@_total_pts_tl { \int_eval:n { \c_@@_total_pts_tl + #2 } }
9660 \tl_gset:Nx \c_@@_total_mins_tl { \int_eval:n { \c_@@_total_mins_tl + #2 } }
9661 }

```

\correction@table This macro generates the correction table

```

9662 \newcommand\correction@table{
9663 \int_compare:nNnT \c_@@_total_problems_int = 0 {
9664 \int_incr:N \c_@@_total_problems_int
9665 \prop_put:Nnn \c_@@_problems_prop {-}{-}{-}}

```

```

9666   }
9667   \tl_clear:N \l_tmpa_tl
9668   \tl_clear:N \l_tmpb_tl
9669   \tl_clear:N \l_tmpc_tl
9670   \prop_map_inline:Nn \c_@@_problems_prop {
9671     \tl_put_right:Nn \l_tmpa_tl { ##1 & }
9672     \tl_put_right:Nx \l_tmpb_tl { \use_i:nn ##2 & }
9673     \tl_put_right:Nn \l_tmpc_tl { & }
9674   }
9675   \resizebox{\textwidth}{!}{%
9676     \exp_args:Nne \begin{tabular}{|l|*{\int_use:N \c_@@_total_problems_int}{c|}c||l|}\hline
9677       &\exp_args:Ne \multicolumn{\int_eval:n{ \c_@@_total_problems_int + 1}}{c||}
9678       {\footnotesize\hwexam@kw@forgrading} &\hline
9679       \hwexam@kw@probs & \l_tmpa_tl \hwexam@kw@sum & \hwexam@kw@grade\hline
9680       \hwexam@kw@pts & \l_tmpb_tl \c_@@_total_pts_tl &\hline
9681       \hwexam@kw@reached & \l_tmpc_tl & [7cm]\hline
9682     \end{tabular}}}

```

(End of definition for `\correction@table`. This function is documented on page ??.)

### \testheading

```

9683 \def\hwexamheader{\input{hwexam-default.header}}
9684
9685 \def\hwexamminutes{
9686   \tl_if_empty:NTF \hwexam@duration {
9687     {\hwexam@min}~\hwexam@minutes@kw
9688   }{
9689     \hwexam@duration
9690   }
9691 }
9692
9693 \stex_keys_define:nnnn{ hwexam / testheading }{
9694   \tl_clear:N \hwexam@min
9695   \tl_clear:N \hwexam@duration
9696   \tl_clear:N \hwexam@reqpts
9697   \tl_clear:N \hwexam@tools
9698 }{
9699   min .tl_set:N = \hwexam@min,
9700   duration .tl_set:N = \hwexam@duration,
9701   reqpts .tl_set:N = \hwexam@reqpts,
9702   tools .tl_set:N = \hwexam@tools
9703 }{}
9704
9705 \newenvironment{testheading}[1][]{%
9706   \stex_keys_set:nn { hwexam / testheading }{#1}
9707
9708   \tl_set_eq:NN \hwexam@totalpts \c_@@_total_pts_tl
9709   \tl_set_eq:NN \hwexam@totalmin \c_@@_total_mins_tl
9710   \tl_set:Nx \hwexam@checktime {\int_eval:n{ \hwexam@min - \hwexam@totalmin }}%
9711
9712   \newif\if@bonuspoints
9713   \tl_if_empty:NTF \hwexam@reqpts {
9714     \@bonuspointsfalse
9715   }%

```

```

9716   \tl_set:Nx \hwexam@bonuspts {
9717     \int_eval:n{ \hwexam@totalpts - \hwexam@reqpts}
9718   }
9719   \!@bonuspointstrue
9720 }
9721
9722   \makeatletter\hwexamheader\makeatother
9723 }{
9724   \newpage
9725 }

```

(End of definition for `\testheading`. This function is documented on page ??.)

9726 `</package>`

#### 14.3.4 Leftovers

at some point, we may want to reactivate the logos font, then we use

```

here we define the logos that characterize the assignment
\font\bierfont=../assignments/bierglas
\font\denkerfont=../assignments/denker
\font\uhrfont=../assignments/uhr
\font\warnschildfont=../assignments/achtung

```

```

\newcommand\bierglas{{\bierfont\char65}}
\newcommand\denker{{\denkerfont\char65}}
\newcommand\uhr{{\uhrfont\char65}}
\newcommand\warnschild{{\warnschildfont\char 65}}
\newcommand\hardA{\warnschild}
\newcommand\longA{\uhr}
\newcommand\thinkA{\denker}
\newcommand\discussA{\bierglas}

```

### 14.4 Tikzinput Implementation

```

9727 <@@=tikzinput>
9728 <*package>
9729
9730 %%%%%%%%%% tikzinput.dtx %%%%%%
9731
9732 \ProvidesExplPackage{tikzinput}{2024/10/26}{4.0.0}{tikzinput package}
9733 \RequirePackage{l3keys2e}
9734
9735 \keys_define:nn { tikzinput } {
9736   image .bool_set:N    = \c_tikzinput_image_bool,
9737   image .default:n     = false ,
9738   unknown .code:n      = {}
9739 }
9740
9741 \ProcessKeysOptions { tikzinput }
9742

```

```

9743 \bool_if:NTF \c_tikzinput_image_bool {
9744   \RequirePackage{graphicx}
9745
9746   \providetcommand\usetikzlibrary[]{}
9747   \newcommand\tikzinput[2][]{\includegraphics[#1]{#2}}
9748 }{
9749   \RequirePackage{tikz}
9750   \RequirePackage{standalone}
9751
9752   \newcommand\tikzinput[2][]{%
9753     \setkeys{Gin}{#1}
9754     \ifx \Gin@ewidth \Gin@exclamation
9755       \ifx \Gin@eheight \Gin@exclamation
9756         \input{#2}
9757       \else
9758         \resizebox{!}{\Gin@eheight}{\input{#2}}
9759       \else
9760         \input{#2}
9761       \fi
9762     \else
9763       \ifx \Gin@eheight \Gin@exclamation
9764         \resizebox{\Gin@ewidth}{!}{\input{#2}}
9765       \else
9766         \resizebox{\Gin@ewidth}{\Gin@eheight}{\input{#2}}
9767       \fi
9768     \fi
9769   }
9770 }
9771 \fi
9772 \fi
9773 }
9774 }
9775
9776 \newcommand\ctikzinput[2][]{%
9777   \begin{center}
9778     \tikzinput[#1]{#2}
9779   \end{center}
9780 }
9781 </package>

```

# Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

| Symbols                                      |                                                                                            |
|----------------------------------------------|--------------------------------------------------------------------------------------------|
| \\$ .....                                    | 2109, 2112, 2116                                                                           |
| \; .....                                     | <u>7549</u> , 7580, 7593, 7595, 7608                                                       |
| @@ commands:                                 |                                                                                            |
| \c_@@_problems_prop .....                    | 9572, 9578, 9585, 9590                                                                     |
| \_@@_qr_escape:n .....                       | 9319, 9320, 9368, 9416, 9436, 9461, 9462                                                   |
| \_@@_qr_escape_char:n ..                     | 9315, 9319                                                                                 |
| \g_@@_qr_in_anscls_json_bool ..              | 9306, 9307, 9454, 9457                                                                     |
| \g_@@_qr_in_problems_json_bool ..            | 9302, 9303, 9360, 9363                                                                     |
| \g_@@_qr_in_subproblems_json_-<br>bool ..... | 9304, 9305, 9380, 9383, 9406, 9412                                                         |
| \c_@@_qr_json_iow .....                      | 9499, 9500, 9501, 9502                                                                     |
| \g_@@_qr_json_str .....                      | 9301, 9309, 9497, 9501                                                                     |
| \c_@@_total_mins_t1 ..                       | 9573, 9580, 9629                                                                           |
| \c_@@_total_problems_int .....               | 9575, 9577, 9583, 9584, 9596, 9597                                                         |
| \c_@@_total_pts_t1 .....                     | 9574, 9579, 9600, 9628                                                                     |
| \\ .....                                     | 53, 80, 1051, 7880, 7885, 8276, 8279, 8984, 9026, 9100, 9138, 9295, 9598, 9599, 9600, 9601 |
| \{ .....                                     | 7598                                                                                       |
| \} .....                                     | 7598                                                                                       |
| \_ .....                                     | 44, 635, 7912, 9250, 9507                                                                  |
| \_@@_qr_backslash .....                      | 9295, 9316                                                                                 |
| \_comp .....                                 | 3978, 4184, 4536, 5232, 6054, 6078, 6162                                                   |
| \_customthiscomp .....                       | 6588, 6589, 6601                                                                           |
| \_defcomp .....                              | 6086, 7027, 7040                                                                           |
| \_stex_html_do_output_bool ..                | 283, 284, 287, 292, 296, 327, 330, 2178, 7540                                              |
| \_thiscomp .....                             | 6571, 6582, 6583                                                                           |
| \_varcomp .....                              | 4517, 4816, 4825, 4982, 5020, 5266, 5711, 6010, 6023, 6036, 6082                           |
| \  .....                                     | 2108, 2111, 2115                                                                           |
| <b>A</b>                                     |                                                                                            |
| \activateexcursion .....                     | <u>109</u>                                                                                 |
| \addbibresource .....                        | <u>80</u> , 1978                                                                           |
| <b>B</b>                                     |                                                                                            |
| \addcontentsline .....                       | 7910                                                                                       |
| \addmhbibresource .....                      | <u>80</u> , <u>1972</u>                                                                    |
| \addtocounter .....                          | 8091                                                                                       |
| \AddToHook .....                             | 1033, 1036, 7344, 7411, 7414, 7526, 8391, 9273                                             |
| \aftergroup ...                              | 144, 3095, 5220, 7409, 7523                                                                |
| \afterprematurestop .....                    | 108, 7974, 7985                                                                            |
| \anscls .....                                | 8828, 8862, 8900, 9446, 9494                                                               |
| \apply .....                                 | 88                                                                                         |
| \arabic .....                                | 8432, 8435, 9540                                                                           |
| \arg .....                                   | 46, 93, 5449                                                                               |
| \argarraymap .....                           | 92, 4245, <u>4756</u>                                                                      |
| \argmap .....                                | 92, 4244, 4386, <u>4732</u> , 7581                                                         |
| \argsep .....                                | 40, 92, 4243, 4381, <u>4716</u> , 7549, 7550, 7551, 7557                                   |
| \assign .....                                | 64, 130, 3014, 3370, 3497                                                                  |
| assignment (env.) .....                      | <u>73</u> , <u>116</u> , <u>9514</u>                                                       |
| \assignmentautorefname .....                 | 9547                                                                                       |
| \assignMorphism .....                        | 130, 3015, 3499                                                                            |
| \assumption .....                            | 7322, <u>7427</u>                                                                          |
| \ast .....                                   | 7572                                                                                       |
| \AtBeginDocument .....                       | 1003, 1255, 1469, 1472, 1520, 7781, 7929                                                   |
| \AtEndDocument .....                         | 643, 1521, 9314, 9496                                                                      |
| \AtEndOfFile .....                           | 2059, 2073, 2086                                                                           |
| \author .....                                | 8256                                                                                       |
| \autoref .....                               | 84, 1665                                                                                   |

|                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| \bfseries              | 8556, 9334, 9336                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| \bgroup                | 3820, 4862, 5074, 8064, 8153, 8168, 8276, 8279, 8663, 8724, 8773, 8825                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| \bigcirc               | 9090, 9115                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \bigskip               | 8525, 9385, 9561                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| blindfragment (env.)   | 82                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| bool commands:         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \bool_gset_false:N     | 9383                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \bool_gset_true:N      | 9363, 9412                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \bool_if:NTF           | 201, 614, 666, 667, 673, 1140, 1150, 1152, 1211, 1232, 1377, 1473, 2161, 2416, 2766, 2799, 2983, 3479, 3688, 4877, 4980, 5230, 5264, 5809, 5903, 5909, 6044, 7017, 7297, 7345, 7383, 7402, 7405, 7407, 7412, 7415, 7423, 7465, 7479, 7501, 7527, 7816, 7824, 7861, 7945, 7989, 8150, 8156, 8190, 8205, 8255, 8290, 8300, 8378, 8384, 8455, 8476, 8511, 8516, 8526, 8554, 8575, 8583, 8586, 8599, 8600, 8604, 8605, 8721, 8732, 8770, 8781, 8822, 8834, 8902, 8907, 8978, 8991, 9006, 9020, 9094, 9107, 9118, 9132, 9250, 9252, 9256, 9298, 9308, 9324, 9349, 9354, 9360, 9380, 9396, 9397, 9401, 9402, 9406, 9419, 9454, 9498, 9507, 9663 |
| \bool_if:nTF           | 287                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| \bool_if_exist:NTF     | 260, 276                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| \bool_lazy_any:nTF     | 4073, 4113                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \bool_lazy_any_p:n     | 770                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| \bool_new:N            | 283, 1352, 2158, 2401, 2824, 4788, 5200, 5778, 5900, 8411, 8450, 8451, 8452, 9302, 9304, 9306                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| \bool_not_p:n          | 769, 773                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| \bool_set_false:N      | 186, 292, 330, 1137, 1161, 1372, 2159, 2178, 2402, 2779, 2846, 3452, 4648, 4791, 5204, 5779, 5904, 7155, 7304, 7352, 7380, 7521, 7540, 7541, 7800, 7841, 8005, 8417, 8453, 8492, 8495, 8952, 8957, 8963, 9303, 9305, 9307                                                                                                                                                                                                                                                                                                                                                                                                                 |
| \bool_set_true:N       | 191, 262, 278, 284, 296, 327, 603, 609, 1134, 1362, 2177, 2763, 2843, 3464, 3474, 3685, 3821, 4882, 5122, 5201, 5434, 5443, 5559, 5570, 5667, 5752, 5810, 5821, 5854, 5883, 5929, 6125, 6543, 6574, 6591, 6636, 7167, 7334, 7813, 7853, 8003, 8011, 8012, 8013, 8014, 8015, 8016, 8488, 8493, 8496, 8559, 8950, 8962, 9457                                                                                                                                                                                                                                                                                                                |
| \bool_while_do:Nn      | 1135                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \bool_while_do:nn      | 768, 2010, 7171, 7183, 7195, 7212, 7224                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| \l_tmpa_bool           | 3452, 3464, 3474, 3479                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| box commands:          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \box_clear:N           | 2181                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \c_empty_box           | 4003, 4010, 8904, 8909                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| \l_tmpa_box            | 2176, 2181, 3700, 4236, 8153, 8168, 8663, 8724, 8773, 8825                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| boxed                  | 110                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| C                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \catcode               | 44, 53, 354, 635, 1051, 1718, 2107, 2108, 2109, 2110, 2111, 2112, 2114, 2115, 2116, 2458, 2459                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| \cdot                  | 7580, 7593                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \centering             | 7914, 8109                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \chapter               | 27, 82, 7954, 7955                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| \chaptername           | 7885, 7951, 7952                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| \chapertitlename       | 7885                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \checkmark             | 9008, 9009, 9120, 9121                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| \circ                  | 54                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| \clearpage             | 1479, 1489, 1500, 1511                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| clist commands:        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \clist_clear:N         | 90, 401, 3729, 4162, 5123, 6425, 6524, 6839, 6844, 7429                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| \clist_count:N         | 6405, 6415                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \clist_count:n         | 4764, 6507                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \clist_get>NN          | 457, 508                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| \clist_if_empty:NTF    | 185, 453, 504, 3837, 4334, 6404, 6450, 6707, 6911, 8917, 9055                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| \clist_if_in:NnTF      | 69, 72, 96, 7005, 8396, 8399, 8402, 8405, 9041, 9278, 9281, 9284, 9287                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| \clist_if_in:nnTF      | 6533                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \clist_item:Nn         | 4343                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \clist_item:nn         | 4775                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \clist_map_function>NN | 5124, 6453                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \clist_map_function:nN | 2716, 2741, 3619, 3639, 3660                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| \clist_map_inline:Nn   | .. 91, 100, 188, 2868, 3839, 4900, 5093, 6225, 6278, 6708, 6821, 6857                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| \clist_map_inline:nn   | 365, 414, 5599, 5698, 5727, 6223, 7111                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| \clist_pop>NN          | 4345, 6232, 6286                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| \clist_put_right:Nn    | .. 92, 5137, 5141, 6432, 6551, 6553                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| \clist_reverse:N       | 6231, 6285                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| \clist_set:Nn          | .. 38, 88, 5001, 6230, 6284, 8395, 9277                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| \clist_set_eq>NN       | 86, 6927                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

\l\_tmpa\_clist 6230, 6231, 6232, 6284,  
     6285, 6286, 8395, 8396, 8399, 8402,  
     8405, 9277, 9278, 9281, 9284, 9287  
 \clstinputmhlisting ..... 83, 2058  
 \cmhgraphics ..... 83, 2058  
 \cmhtikzinput ..... 118, 2058  
 \color ..... 9336  
 \columnbox ..... 8127, 8129, 8147  
 \comp ..... 32, 33, 36, 46, 90, 93,  
     103, 132, 3978, 4184, 4357, 4517,  
     4536, 4578, 4584, 4586, 4816, 4825,  
     5020, 5249, 5250, 5587, 5711, 5931,  
     5936, 5951, 6010, 6023, 6036, 6054,  
     6056, 6063, 6162, 6392, 6400, 6583,  
     6597, 6598, 7027, 7028, 7040, 7546,  
     7548, 7555, 7557, 7559, 7560, 7565,  
     7567, 7572, 7574, 7575, 7582, 7583,  
     7584, 7585, 7588, 7590, 7594, 7595,  
     7596, 7598, 7608, 7615, 7617, 7619  
 \compemph ..... 103, 6063  
 \conclude ..... 7321, 7427  
 \conclusion 50–52, 99, 101, 6996, 7120, 7136  
 \copymod ..... 62, 63, 3629, 3631, 3633  
 \copymodule ..... 3516, 3518  
 \counterwithin ..... 7958, 7970  
 \cr ..... 9232, 9233  
 \crcr ..... 9199  
 cs commands:  
     \cs:w ..... 440, 443, 475, 636, 2191,  
         2420, 2421, 2422, 2429, 2433, 2439,  
         5331, 6269, 6323, 6325, 6945, 6958  
     \cs\_argument\_spec:N ..... 4027  
     \cs\_end: .. 440, 443, 475, 636, 2191,  
         2420, 2421, 2422, 2431, 2435, 2439,  
         5331, 6269, 6323, 6325, 6945, 6958  
     \cs\_generate\_from\_arg\_count>NNnn  
         ..... 3830,  
         4644, 4891, 5084, 5168, 5373, 5405  
     \cs\_generate\_variant:Nn .....  
         ..... 161, 235, 248, 553, 577,  
         586, 598, 622, 682, 712, 867, 872,  
         876, 960, 1121, 1146, 1578, 1906,  
         2223, 2581, 2585, 2590, 2608, 2657,  
         2679, 2692, 3147, 5225, 5242, 5563  
     \cs\_gset:Npn ..... 2454  
     \cs\_if\_eq:NNTF ..... 144,  
         882, 936, 1445, 2247, 2258, 2261,  
         2280, 2283, 4029, 4717, 5643, 5645  
     \cs\_if\_exist:NTF .....  
         .... 313, 1089, 1323, 1327, 1361,  
         1365, 1393, 1396, 1420, 1424, 1453,  
         1459, 1474, 1484, 1497, 1508, 1664,  
         1812, 1836, 1856, 1862, 1866, 2129,  
         2669, 2674, 4005, 4012, 4024, 4455,

4957, 4965, 5028, 5042, 5055, 5056,  
 5058, 5102, 5135, 5154, 5166, 5178,  
 5207, 5229, 5244, 5263, 5288, 5293,  
 5298, 5369, 5385, 5391, 5426, 5432,  
 5440, 5495, 5556, 5565, 5581, 5593,  
 5627, 5638, 5659, 5691, 5723, 5769,  
 5772, 5782, 5795, 5808, 5831, 5832,  
 5833, 5837, 5863, 5866, 5892, 5902,  
 5990, 6042, 6063, 6157, 6182, 6189,  
 6206, 6212, 6264, 6328, 6336, 6343,  
 6348, 6402, 6424, 6431, 6435, 6498,  
 6523, 6549, 6581, 6587, 6606, 6618,  
 6629, 6647, 6657, 6697, 6706, 6714,  
 6723, 6733, 6744, 6758, 6782, 6864,  
 6916, 6962, 6975, 7013, 7088, 7104,  
 7180, 7268, 7275, 7279, 7290, 7305,  
 7422, 7445, 7463, 7778, 7874, 7950,  
 7962, 8001, 8039, 8063, 8068, 8074,  
 8083, 8155, 8310, 8543, 8637, 8701,  
 8750, 8801, 8948, 9040, 9216, 9226  
`\cs_new_protected:Npn` ..... 210, 333, 336, 340,  
 341, 424, 684, 888, 891, 1237, 1321,  
 1418, 1434, 1522, 1524, 1855, 1861,  
 1865, 1880, 2238, 2321, 2397, 2880,  
 2940, 2978, 3006, 3037, 3051, 3134,  
 3149, 3283, 3612, 4453, 4610, 4617,  
 4667, 4723, 4732, 4757, 5108, 5121,  
 5186, 5192, 5196, 5226, 5276, 5317,  
 5347, 5472, 5607, 5921, 5941, 5948,  
 6078, 6082, 6086, 6092, 6096, 6100,  
 6104, 6108, 6112, 6116, 6120, 6124,  
 6369, 6391, 6396, 6487, 6506, 6519,  
 6571, 6818, 6855, 7169, 7193, 7210,  
 7222, 7265, 7628, 8549, 8692, 8695  
`\cs_parameter_spec:N` ..... 543, 557  
`\cs_prefix_spec:N` ..... 555  
`\cs_set:Nn` 8929, 8944, 9043, 9069, 9084  
`\cs_set:Npn` ..... 444, 479, 484,  
 632, 633, 816, 817, 818, 819, 940,  
 947, 1090, 1093, 1106, 2200, 2449,  
 2482, 2696, 2698, 2701, 2714, 2739,  
 2764, 2778, 3831, 4045, 4238, 4239,  
 4243, 4244, 4245, 4357, 4387, 4628,  
 4644, 4738, 4763, 4892, 4930, 5085,  
 5168, 5374, 5406, 5725, 6526, 6532,  
 6593, 7537, 7538, 7539, 7856, 8160  
`\cs_set:Npx` ..... 2638  
`\cs_set_eq:NN` 41, 222, 226, 810, 811,  
 812, 1257, 1264, 1410, 2675, 3017,  
 3018, 4025, 4491, 4495, 4642, 4695,  
 4708, 4923, 5156, 5159, 5233, 5235,  
 5267, 5269, 5449, 5452, 5480, 5481,  
 5693, 5696, 5835, 7704, 7716, 8203  
`\cs_set_protected:Nn` ..... 807, 814, 1591, 7875, 7908, 7924  
`\cs_set_protected:Npn` ..... 346, 1276, 1285, 4001, 4002,  
 4381, 4386, 4392, 6090, 7705, 7952,  
 7955, 7964, 7967, 8204, 8227, 8231,  
 8242, 8246, 8332, 8335, 8338, 9576  
`\cs_set_protected:Npx` ..... 6589  
`\cs_undefine:N` ... 217, 227, 274, 1202  
`\l_tmpa_cs` ..... 4238, 4248, 5725, 5743  
`\csname` ..... 272, 301, 351, 2105, 2106, 2107, 2108,  
 2109, 2114, 2115, 2116, 2419, 2682  
`\ctikzinput` ..... 118, 9696  
`\CurrentFile` ..... 1020, 1022, 1024  
`\CurrentFilePath` ..... 1020, 1021, 1024  
`\currentgrouplevel` . 232, 254, 255, 256,  
 258, 260, 262, 270, 272, 274, 276, 278  
`\CurrentOption` ..... 10, 7803, 7804,  
 7805, 7806, 7845, 7846, 8372, 9268  
`\Currentsectionlevel` ..... 82, 1285  
`\currentsectionlevel` ..... 82, 1276

## D

`\DeclareOption` ..... 10  
`\def` ..... 91, 142, 307, 311, 314, 316, 353, 556, 1239, 1243,  
 1260, 1265, 2075, 2078, 2089, 2093,  
 3978, 4184, 4205, 4515, 4517, 4534,  
 4536, 4816, 4823, 4825, 5018, 5020,  
 5249, 5250, 5711, 5951, 5980, 5994,  
 6010, 6023, 6036, 6054, 6065, 6162,  
 6596, 6601, 7027, 7040, 7973, 7975,  
 8040, 8041, 8042, 8046, 8049, 8092,  
 8195, 8212, 8220, 8256, 8257, 8259,  
 8261, 8263, 8265, 8266, 8268, 8270,  
 8272, 8275, 8276, 8278, 8279, 8282,  
 8284, 8286, 8330, 8432, 8435, 8436,  
 8901, 8906, 9315, 9322, 9325, 9331,  
 9338, 9342, 9343, 9540, 9603, 9605  
`\defemph` ..... 103, 6063  
`\Definename` .... 100, 6971, 7001, 7058, 7064  
`\definename` ..... 24, 36, 45, 99, 100, 103, 6970, 7000, 7050, 7056  
`\definiendum` ..... 24, 36, 45, 99, 100, 103, 6968, 6998, 7044, 7048  
`\definiendum` ..... 7013  
`\definiens` ..... 47, 57, 64, 99–101, 6986, 7067, 7085, 7086  
`\defnotation` ..... 36, 45, 100, 6969, 6999, 7039, 7042  
`\detokenize` 251, 351, 1684, 8396, 8399,  
 8402, 8405, 9278, 9281, 9284, 9287

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| dim commands:       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \dim_compare:nNnTF  | 9239                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| \dim_new:N          | 8628                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| \dim_zero:N         | 8631, 9176                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \dimexpr            | 8191                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| do commands:        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \_do_comp:nNn       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| .....               | 4393, 6063, 6079, 6083, 6087                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| \dobracket          | 92                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| \dobrackets         | 92, 5896, 5949                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| \doproblemqr        | 9325, 9342, 9348, 9395, 9509                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| \dowithbrackets     | 5948                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| \due                | 73, 116                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| \duration           | 74, 116                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| E                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \edef               | 2107, 2108, 2109, 6583, 8057                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| \egroup             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| .....               | 3844, 4907, 5100, 8071, 8153, 8170, 8276, 8279, 8677, 8738, 8787, 8840                                                                                                                                                                                                                                                                                                                                                                                               |
| \eject              | 9250, 9510                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \ellipses           | 95, 96, 4742, 4743, 5136, 5138, 5699, 5737                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \else               | 300, 312, 1913, 1996, 2234, 5926, 6105, 8330, 8555, 8660, 8676, 9012, 9124, 9238, 9326, 9332, 9677, 9682, 9687                                                                                                                                                                                                                                                                                                                                                       |
| \emph               | 17, 19, 102, 6113, 7252, 7340, 7357, 8682, 8743, 8792, 8845, 8984, 9026, 9100, 9138, 9431                                                                                                                                                                                                                                                                                                                                                                            |
| \end                | 137, 142, 1304, 1386, 2070, 2083, 2101, 2261, 2283, 2306, 2566, 2821, 2899, 4781, 6939, 7250, 7276, 7338, 7355, 7406, 7408, 7486, 7490, 7692, 7918, 7925, 7977, 7986, 8059, 8071, 8085, 8125, 8139, 8147, 8160, 8162, 8175, 8176, 8177, 8178, 8179, 8180, 8201, 8229, 8233, 8236, 8293, 8327, 8328, 8507, 8592, 8610, 8669, 8674, 8730, 8735, 8779, 8784, 8832, 8837, 8898, 8934, 8940, 9074, 9080, 9157, 9162, 9167, 9172, 9250, 9426, 9492, 9510, 9560, 9602, 9699 |
| \endcsname          | 272, 300, 301, 351, 2105, 2106, 2107, 2108, 2109, 2114, 2115, 2116, 2419, 2682, 8330                                                                                                                                                                                                                                                                                                                                                                                 |
| \endgroup           | 356, 358, 1957                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| \endinput           | 1792, 2308                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \ensuremath         | 5811, 5822                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| environments:       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| assignment          | 73, 116, 9514                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| blindfragment       | 82                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| exnote              | 1, 8750                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| extstructure        | 96, 6221                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| extstructure*       | 96                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| frame               | 1, 8001                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| gnote               | 1, 8799                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| hint                | 1, 8698                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| mathstructure       | 96, 6135                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| mcb                 | 1, 8911                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| mmtinterface        | 7658                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| nassertion          | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| ndefinition         | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| nexample            | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| note                | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| nparagraph          | 1, 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| nsproof             | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| problem             | 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| sassertion          | 99                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| scb                 | 9039                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| sdefinition         | 99                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| sexample            | 99                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| sfragment           | 82                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| smodule             | 86, 2538                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| solution            | 1, 8627                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| sparagraph          | 99                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| sproblem            | 8437                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| sproof              | 101, 7268                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| stex_annotation_env | 142                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| subproblem          | 8550                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| subproof            | 7361                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| testheading         | 74, 116                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| \eq                 | 32, 42                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| \eqstep             | 7323, 7427                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| \equal              | 31                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| \errmessage         | 7735, 7749, 7756, 7770                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| \escapechar         | 53, 1051, 9294                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| \everyeof           | 2187                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| \examnumber         | 9322, 9333, 9334, 9336                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| \excursion          | 109, 8288                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| \excursiongroup     | 109, 8305                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| \excursionref       | 109, 8289, 8302                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \exnote (env.)      | 1, 8750                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| \exnote             | 8443, 8798, 8926, 9066                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| exp commands:       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| \exp_after:wN       | 351, 440, 443, 475, 542, 694, 883, 2189, 2233, 2234, 2235, 2375, 2420, 2421, 2422, 2429, 2433, 2437, 2438, 2960, 2981, 3597, 3834, 4134, 4147, 4746, 4774, 4895, 5088, 5328, 5330, 5338, 5340, 5387, 5394, 5395, 5396, 5415, 5448, 5610, 5612, 5714, 5715, 5716, 5729, 5924, 5987, 6033, 6049, 6268, 6322, 6325, 6649, 6945, 6958, 7022, 7061                                                                                                                        |
| \exp_args:Ne        | 530, 535, 549, 550, 882, 936, 1024, 1096, 1123, 1675, 1824, 1837,                                                                                                                                                                                                                                                                                                                                                                                                    |

1851, 1867, 1870, 1873, 1875, 2369,  
 2863, 2869, 2959, 2962, 3391, 3419,  
 3425, 3793, 3802, 3963, 4027, 4029,  
 4717, 4862, 5059, 5498, 5643, 5645,  
 5654, 6226, 6279, 6370, 6374, 6452,  
 6533, 6582, 6588, 6822, 6858, 6951,  
 7240, 7447, 7779, 9217, 9319, 9597  
**\exp\_args:NNe** .....  
 ... 54, 202, 205, 590, 1069, 1073,  
 1079, 2483, 2922, 3571, 4066, 4252,  
 4350, 5530, 5533, 5616, 5663, 5748,  
 6553, 6661, 6716, 6784, 8322, 8544  
**\exp\_args:Nne** .....  
 . 211, 542, 2051, 2122, 2507, 2547,  
 2604, 2805, 2883, 4208, 4359, 4398,  
 4517, 4536, 4825, 5020, 6164, 6608,  
 6649, 6717, 6922, 7141, 7291, 7469,  
 7668, 8473, 8572, 8915, 9053, 9596  
**\exp\_args:NNNo** .....  
 828, 1171, 3581, 4598, 4600, 4834, 5029  
**\exp\_args:NNno** ..... 271, 679, 828  
**\exp\_args:Nnno** ..... 1751, 1759  
**\exp\_args:NNNx** ..... 1171, 1740  
**\exp\_args:NNnx** .....  
 ... 833, 3098, 4834, 5029, 5509, 5515  
**\exp\_args:NNo** ..... 38,  
 48, 69, 72, 92, 96, 169, 177, 692,  
 801, 833, 837, 842, 847, 1012, 1174,  
 1682, 2013, 2023, 3231, 3257, 3578,  
 3584, 3588, 4300, 4313, 4656, 5742  
**\exp\_args:Nno** .. 256, 258, 271, 374,  
 2450, 2985, 2988, 3829, 4066, 4890,  
 5083, 5189, 5194, 5278, 6072, 6542,  
 7089, 8460, 8623, 9316, 9529, 9569  
**\exp\_args:NNx** .....  
 . 55, 58, 112, 125, 907, 994, 1922,  
 1970, 2151, 2170, 6675, 8396, 8399,  
 8402, 8405, 9278, 9281, 9284, 9287  
**\exp\_args:Nnx** ... 251, 1312, 1616,  
 1684, 1922, 1970, 2838, 3039, 3044,  
 3231, 3257, 3783, 3954, 5167, 5372,  
 5404, 5414, 5923, 6196, 6879, 7706  
**\exp\_args:No** 146, 166, 167, 168, 215,  
 306, 435, 602, 608, 747, 1022, 1107,  
 1207, 1259, 1267, 1289, 1312, 1366,  
 1555, 1740, 1745, 1750, 1782, 1787,  
 1789, 2097, 2390, 2487, 2542, 2579,  
 2908, 2920, 2957, 3144, 3159, 3401,  
 3402, 3595, 3600, 3684, 3788, 3789,  
 3790, 4064, 4220, 4589, 4764, 4840,  
 4841, 4842, 4853, 4854, 4855, 5035,  
 5036, 5037, 5048, 5049, 5050, 5142,  
 5211, 5214, 5218, 5259, 5402, 5407,  
 5412, 5417, 5419, 5507, 5535, 5643,  
 5645, 5672, 5757, 6174, 6184, 6216,  
 6296, 6408, 6412, 6419, 6446, 6481,  
 6483, 6555, 6557, 6563, 6638, 6664,  
 6670, 6672, 6677, 6681, 6687, 6734,  
 6783, 7026, 7080, 7869, 7933, 7934,  
 7937, 8395, 8479, 8579, 9277, 9320  
**\exp\_args:Nx** .....  
 ... 253, 693, 748, 756, 1559, 1665,  
 1667, 2239, 2544, 3023, 8002, 8949  
**\exp\_not:N** ..... 59, 215, 217, 556,  
 942, 949, 1106, 1217, 1220, 1975,  
 2032, 2187, 2233, 2443, 2839, 3394,  
 3428, 4219, 4606, 5170, 5173, 5209,  
 5210, 5213, 5217, 5221, 5382, 5457,  
 5463, 5664, 5668, 5671, 5749, 5753,  
 5755, 5925, 6290, 6482, 6556, 6562,  
 6598, 6637, 6663, 6669, 6678, 6689,  
 7880, 7885, 9021, 9023, 9133, 9135  
**\exp\_not:n** ... 215, 539, 1107, 1312,  
 1555, 2437, 2464, 2487, 2617, 2641,  
 3400, 3401, 3402, 3788, 3789, 3790,  
 3983, 4220, 4261, 4262, 4589, 4746,  
 4774, 4840, 4841, 4842, 4853, 4854,  
 4855, 5035, 5036, 5037, 5048, 5049,  
 5050, 5164, 5211, 5214, 5218, 5220,  
 5222, 5259, 5395, 5402, 5407, 5412,  
 5417, 5419, 5630, 5672, 5709, 5714,  
 5757, 6262, 6408, 6446, 6476, 6481,  
 6483, 6555, 6557, 6559, 6563, 6566,  
 6592, 6638, 6640, 6664, 6666, 6670,  
 6672, 6677, 6681, 6687, 9026, 9138  
**\expandafter** .....  
 . 301, 356, 2105, 2106, 2107, 2108,  
 2109, 2419, 2682, 6105, 7977, 7978  
**\ExplSyntaxOff** 2156, 2596, 7786, 8409, 9291  
**\ExplSyntaxOn** 2155, 2592, 7783, 8392, 9274  
**\extref** ..... 85, 1579  
**\extstructure** (env.) ..... 96, 6221  
**\extstructure** ..... 6256, 6258  
**\extstructure\*** (env.) ..... 96

## F

**\fbox** ..... 9229, 9239  
**\fi** ..... 303, 318, 1917, 1932, 1940,  
 1996, 2235, 5231, 5265, 5939, 6043,  
 6105, 7014, 7591, 7979, 8047, 8050,  
 8149, 8330, 8557, 8664, 8678, 9014,  
 9029, 9034, 9126, 9141, 9146, 9244,  
 9316, 9329, 9339, 9681, 9691, 9692  
**\fiboxed** ..... 105  
 file commands:  
     **\file\_if\_exist:nTF** ... 630, 647, 1160  
**\filepath** ..... 139, 140  
**\fillinsol** ..... 114, 8441, 9174

|                                 |                                     |                                                                    |
|---------------------------------|-------------------------------------|--------------------------------------------------------------------|
| \first . . . . .                | 138                                 | <b>H</b>                                                           |
| \fn . . . . .                   | 54                                  | \halign . . . . . 9232                                             |
| \foo . . . . .                  | 15, 54, 88, 142, 144                | \have . . . . . 7427                                               |
| \fooname . . . . .              | 15                                  | \hbox 1251, 3084, 3387, 3820, 3975, 4004,                          |
| \footnote . . . . .             | 9033, 9145, 9231                    | 4375, 4862, 5074, 6240, 6828, 7256,                                |
| \footnotesize . . . . .         | 9598                                | 7280, 8276, 8279, 8972, 9001, 9208                                 |
| \foral . . . . .                | 46, 51                              | hbox commands:                                                     |
| \forallal . . . . .             | 46, 7578, 7584, 7595                | \hbox_set:Nn . . . . . 3700, 4236                                  |
| fp commands:                    |                                     | \hbox_unpack:N 4003, 4010, 8904, 8909                              |
| \fp_gadd:Nn . . . . .           | 8503, 8504                          | \HCode . . . . . 313                                               |
| \fp_new:N . . . . .             | 8447, 8448                          | \hfil . . . . . 7262, 9232                                         |
| \fp_to_decimal:n . . . . .      | 8584, 8587                          | \hfill . . . . . 7262                                              |
| frame (env.) . . . . .          | 1, <u>8001</u>                      | hint (env.) . . . . . 1, <u>8698</u>                               |
| \frameimage . . . . .           | 108, 8184                           | \hint . . . . . 8442, 8749, 8925, 9065                             |
| frameimages . . . . .           | 105                                 | hints . . . . . 73, 110, 115                                       |
| \frametitle . . . . .           | 8101                                | \hline . . . . . 9596, 9598, 9599, 9600, 9601                      |
| \frontmatter . . . . .          | 1474, 1475, 1476                    | \href . . . . . 1862                                               |
| \fun . . . . .                  | 45                                  | \hrule . . . . . 7256, 8972, 9001                                  |
| \funspace . . . . .             | 37                                  | \hspace . . . . . 9008, 9120, 9242                                 |
| <b>G</b>                        |                                     |                                                                    |
| \gdef . . . . .                 | 8288, 9309, 9311, 9319, 9320, 9336  | \HTML . . . . . 15, 26                                             |
| \given . . . . .                | 73, 116                             | \huge . . . . . 9240, 9242                                         |
| \global . . . . .               | 1239, 1243, 1260, 1265, 2106, 7624, | hwexam commands:                                                   |
|                                 | 7625, 8693, 8696, 9338, 9536, 9538  | \l_hwexam_includeassignment_-                                      |
| \gnote (env.) . . . . .         | 1, <u>8799</u>                      | keys_tl . . . . . 9527, 9528, 9530, 9567                           |
| \gnote . . . . .                | 8444, 8851, 8927, 9067              | \hwexam_kw_* . . . . . 9273                                        |
| \gnote . . . . .                | 73, 110, 115                        | \c_hwexam_multiple_bool . . . . . 9264                             |
| group commands:                 |                                     | \c_hwexam_qrcode_bool . . . . . 9266, 9298                         |
| \group_begin: . . . . .         | 43, 52, 631, 1050, 1718,            | \hwexamheader . . . . . 9603, 9642                                 |
|                                 | 2198, 2457, 2494, 2813, 2833, 2857, | \hwexamminutes . . . . . 9605                                      |
|                                 | 3088, 3312, 3353, 3701, 3755, 3920, | \hyperlink . . . . . 1856, 1857                                    |
|                                 | 3922, 3924, 3926, 3987, 4193, 4511, | \hypertarget . . . . . 1836, 1837                                  |
|                                 | 4557, 4819, 4974, 5014, 5143, 5147, |                                                                    |
|                                 | 5245, 5402, 5558, 5566, 5639, 5660, |                                                                    |
|                                 | 5708, 5724, 5927, 5964, 5972, 5983, |                                                                    |
|                                 | 6005, 6016, 6029, 6573, 6590, 6946, |                                                                    |
|                                 | 7015, 7068, 7121, 7335, 7402, 7424, |                                                                    |
|                                 | 7487, 7508, 7536, 8620, 9293, 9566  |                                                                    |
| \group_end: . . . . .           | 47, 58, 637, 1054,                  | <b>I</b>                                                           |
|                                 | 1765, 2221, 2475, 2508, 2822, 2838, | \id . . . . . 123                                                  |
|                                 | 2900, 3093, 3318, 3359, 3703, 3763, | \if . . . . . 2233, 5922                                           |
|                                 | 3920, 3922, 3924, 3926, 3992, 4012, | if commands:                                                       |
|                                 | 4196, 4522, 4560, 4830, 4986, 5025, | \if_charcode:w . . . . . 9316                                      |
|                                 | 5114, 5132, 5144, 5148, 5189, 5325, | \IfBooleanTF . . . . . 3747, 4187,                                 |
|                                 | 5336, 5354, 5365, 5387, 5415, 5420, | 4508, 5486, 7373, 7480, 7503, 8185                                 |
|                                 | 5436, 5469, 5561, 5578, 5639, 5663, | \ifcsname . . . . . 300, 8330                                      |
|                                 | 5716, 5748, 5935, 6060, 6490, 6546, | \ifdefempty . . . . . 8321                                         |
|                                 | 6576, 6599, 6609, 6675, 6955, 7032, | \iffalse . . . . . 7577                                            |
|                                 | 7082, 7134, 7346, 7412, 7416, 7465, | \IfFileExists . . . . . 6, 20, 1745, 1931,                         |
|                                 | 7502, 7527, 7626, 8624, 9296, 9570  | 1939, 2012, 2022, 3260, 3265, 3285                                 |
| \group_insert_after:N . . . . . | 261, 277                            | \IfInputref . . . . . 28, 83, <u>1989</u> , 8319                   |
|                                 |                                     | \ifininputref . . . . . 28, 83, <u>1907</u> , 1996                 |
|                                 |                                     | \ifintest . . . . . 8383                                           |
|                                 |                                     | \ifmmode . . . . . 6105                                            |
|                                 |                                     | \ifnotes . . . . . 106, 7860, 7997                                 |
|                                 |                                     | \ifSGvar . . . . . 109, 8338                                       |
|                                 |                                     | \ifsolutions . . . . . 111, 8375, 8658, 8671, 9005,                |
|                                 |                                     | 9018, 9032, 9117, 9130, 9144, 9228                                 |
|                                 |                                     | \ifstexhtml . . . . . . 28, 81, 141, <u>300</u> , 8555, 9326, 9332 |

\ifvmode 1932, 1940, 5231, 5265, 6043, 7014  
 \ifx 2105, 7976, 8045, 8048, 9674, 9675, 9683  
 \ignorespaces .... 3095, 4830, 5025, 8151  
 image ..... 117  
 \imply ..... 51  
 \importmodule ..... 38, 49, 55, 93–95,  
     131, 134, 136, 137, 144, 213, 3011, 3290  
 \includeassignment ..... 9565  
 \includegraphics ..... 83, 2068, 9667  
 \includeproblem ..... 72, 114, 8612  
 \indent ..... 5231, 5265, 6043, 7014  
 \infprec 90, 91, 4300, 5894, 5897, 5930, 7548  
 \inline\* ..... 100  
 \inlineass ..... 50, 55, 58, 100  
 \inlinedef ..... 50, 100  
 \inlineex ..... 100  
 \input ..... 27, 80, 136, 40, 322,  
     1027, 1944, 2035, 8393, 8397, 8400,  
     8403, 8406, 9275, 9279, 9282, 9285,  
     9288, 9603, 9676, 9679, 9685, 9689  
 \inputassignment ..... 74, 116  
 \inputref ... 27, 28, 82, 83, 102, 1907,  
     8203, 8204, 8297, 8322, 8623, 9569  
 \inputref\* ..... 107, 8203  
 \inputreffalse ..... 1907, 1916  
 \inputreftrue ..... 1914, 1951  
 \insertexamnumber ..... 9331, 9338, 9343  
 \insertframenumber ..... 8282, 8284  
 \insertshortauthor ..... 8275  
 \insertshortdate ..... 8286  
 \insertshorttitle ..... 8278  
 \inset ..... 45  
 int commands:  
     \int\_case:nn ..... 1392  
     \int\_case:nnTF ..... 1322, 1419, 1452  
     \int\_compare:nNnTF .....  
         253, 2049, 2121, 3572, 4225, 4260,  
         4770, 5356, 5642, 5653, 5702, 5908,  
         6405, 6415, 6507, 6890, 7202, 9583  
     \int\_compare\_p:nNn .....  
         7172, 7184, 7196, 7213, 7225  
     \int\_decr:N ..... 7203, 7231  
     \int\_eval:n ..... 270, 272, 274, 276, 278,  
         5512, 5518, 5907, 6197, 6204, 6338,  
         9536, 9579, 9580, 9597, 9630, 9637  
     \int\_gincr:N ..... 1533, 5497,  
         8582, 8640, 8704, 8753, 8804, 9577  
     \int\_gset:Nn ..... 5461  
     \int\_gzero:N ..... 5446, 8497  
     \int\_incr:N .....  
         1314, 1324, 1328, 1380, 1394, 1397,  
         1400, 1421, 1425, 1456, 1462, 3896,  
         3897, 3898, 3899, 4686, 4769, 5608,  
             7177, 7189, 7200, 7217, 7229, 7645,  
             7919, 8243, 8247, 8865, 9449, 9584  
 \int\_new:N ..... 1272,  
     1529, 3852, 4041, 4756, 5439, 5592,  
     5896, 8239, 8550, 8627, 8799, 9575  
 \int\_set:Nn 1436, 1437, 1438, 1439,  
     1440, 1441, 1443, 3068, 3861, 3865,  
     3869, 3873, 3877, 3881, 3885, 3889,  
     3893, 3969, 4048, 4089, 4121, 4659,  
     4764, 4771, 4809, 4933, 5569, 5897,  
     5930, 7170, 7182, 7194, 7211, 7223  
 \int\_step\_function:nN ... 5171, 5375  
 \int\_step\_inline:nn .....  
     3910, 4292, 4299, 4319, 4580  
 \int\_use:N .. 232, 1298, 1406, 1446,  
     1466, 1534, 3398, 3786, 4369, 4487,  
     4659, 4838, 4851, 5033, 5046, 5281,  
     5461, 5498, 5629, 5666, 5751, 5894,  
     5895, 6881, 7648, 7709, 7912, 8244,  
     8248, 8598, 8642, 8706, 8755, 8806,  
     8867, 9393, 9394, 9417, 9451, 9596  
 \int\_zero:N ..... 3855,  
     3856, 4641, 4767, 5598, 7638, 8241  
 \c\_max\_int ..... 5894, 5895  
 \l\_tmpa\_int .....  
     4641, 4644, 4659, 4686, 4767, 4769,  
     4770, 4771, 4775, 7170, 7173, 7176,  
     7177, 7182, 7185, 7188, 7189, 7194,  
     7197, 7200, 7202, 7203, 7205, 7206,  
     7211, 7214, 7217, 7219, 7223, 7226,  
     7229, 7231, 7232, 7638, 7645, 7648  
 intarray commands:  
     \intarray\_gset:Nnn .....  
         7205, 7219, 7232, 7307  
     \intarray\_gzero:N ..... 7306  
     \intarray\_item:Nn .... 7173, 7176,  
         7185, 7188, 7197, 7206, 7214, 7226  
     \intarray\_new:Nn ..... 7168  
 \interpretmod ..... 3650, 3652, 3654  
 \interpretmodule ..... 3537, 3539  
 \intestfalse ..... 8387  
 \intesttrue ..... 8385  
 invocation commands:  
     \invokation\_macro ..... 123, 126, 132  
 ior commands:  
     \ior\_close:N .... 654, 660, 1055, 1185  
     \ior\_map\_inline:Nn ..... 1170  
     \ior\_new:N ..... 1166  
     \ior\_open:Nn ..... 648, 656, 1168  
     \ior\_open:NnTF ..... 1049  
     \ior\_str\_get:NN ..... 1052  
     \ior\_str\_map\_inline:Nn ..... 650, 657  
     \g\_tmpa\_ior ..... 648, 650,  
         654, 656, 657, 660, 1049, 1052, 1055

|                      |                                                                                                                                                                                                                    |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| iow commands:        |                                                                                                                                                                                                                    |
| \iow_close:N         | 643, 653, 1521, 9502                                                                                                                                                                                               |
| \iow_new:N           | 612, 1519, 9499                                                                                                                                                                                                    |
| \iow_now:Nn          | 620,<br>651, 658, 1547, 1818, 7782, 8544, 9501                                                                                                                                                                     |
| \iow_open:Nn         | 642, 649, 1520, 9500                                                                                                                                                                                               |
| \g_tmpa_iow          | 649, 651, 653                                                                                                                                                                                                      |
| \isassociative       | 49                                                                                                                                                                                                                 |
| \iscommutative       | 49                                                                                                                                                                                                                 |
| \item                | 7273, 8598, 8874, 8934, 9074, 9393, 9470                                                                                                                                                                           |
| \itshape             | 7266                                                                                                                                                                                                               |
| <b>J</b>             |                                                                                                                                                                                                                    |
| \jobname             | 77, 142, 642, 647, 648, 649,<br>656, 661, 670, 977, 1520, 1740, 9500                                                                                                                                               |
| \join                | 63                                                                                                                                                                                                                 |
| <b>K</b>             |                                                                                                                                                                                                                    |
| keys commands:       |                                                                                                                                                                                                                    |
| \l_keys_choice_tl    | 3718                                                                                                                                                                                                               |
| \keys_define:nn      | 27,<br>374, 7797, 7839, 8305, 8354, 9262, 9655                                                                                                                                                                     |
| \l_keys_key_str      | 4170, 4173, 6149, 6152                                                                                                                                                                                             |
| \l_keys_key_tl       | 4171, 6150                                                                                                                                                                                                         |
| \keys_set:nn         | 378, 8314                                                                                                                                                                                                          |
| keyval commands:     |                                                                                                                                                                                                                    |
| \keyval_parse>NNn    | 6427                                                                                                                                                                                                               |
| <b>L</b>             |                                                                                                                                                                                                                    |
| \label               | 84, 121, 133, 1559, 8094                                                                                                                                                                                           |
| \labelsep            | 8053                                                                                                                                                                                                               |
| \labelwidth          | 8054                                                                                                                                                                                                               |
| \lambda              | 7578, 7589, 7590                                                                                                                                                                                                   |
| \langle              | 7565                                                                                                                                                                                                               |
| \Large               | 7857, 8213, 9334, 9336, 9547                                                                                                                                                                                       |
| \lastslide           | 8231, 8246                                                                                                                                                                                                         |
| \LaTeX               | 23                                                                                                                                                                                                                 |
| \latex               | 23                                                                                                                                                                                                                 |
| \ldots               | 7570, 7615, 7617, 7619                                                                                                                                                                                             |
| \leaders             | 8215                                                                                                                                                                                                               |
| \leavevmode          | 8109                                                                                                                                                                                                               |
| \left                | 5924                                                                                                                                                                                                               |
| \leftmargin          | 8055                                                                                                                                                                                                               |
| \let                 | 351, 1475, 1476,<br>1485, 1486, 1748, 2106, 2188, 2199,<br>2206, 2217, 2229, 3336, 4004, 4011,<br>4516, 4535, 4824, 5019, 5248, 5969,<br>5979, 5993, 6055, 6134, 7624, 7625,<br>7784, 7785, 7926, 8222, 9042, 9045 |
| \libinput            | 20, 80, <u>1999</u>                                                                                                                                                                                                |
| \libusepackage       | 80, 81, <u>2047</u> , 7993                                                                                                                                                                                         |
| \libusetHEME         | 7992                                                                                                                                                                                                               |
| \libusetikZlibrary   | 80, 118, <u>2103</u>                                                                                                                                                                                               |
| \linewidth           | 8681, 8688, 8742, 8747,<br>8791, 8796, 8844, 8849, 9430, 9443                                                                                                                                                      |
| \LoadClass           | 15, 7825, 7827                                                                                                                                                                                                     |
| \long                | 142, 583, 8257, 8266                                                                                                                                                                                               |
| \lstinputlisting     | 83, 2082                                                                                                                                                                                                           |
| \lstinputmhlisting   | 83, <u>2058</u>                                                                                                                                                                                                    |
| <b>M</b>             |                                                                                                                                                                                                                    |
| \macro               | 136–138, 143                                                                                                                                                                                                       |
| \macroname           | 32                                                                                                                                                                                                                 |
| \magma               | 53                                                                                                                                                                                                                 |
| \maincomp            | 32, 33, 36, 54, 90, 97,<br>103, 4011, 4219, 4228, 4239, 4240,<br>4261, 4392, 4606, 5250, 5435, 5951,<br>6130, 6582, 6583, 6588, 6596, 6601                                                                         |
| \mainmatter          | 1497, 1498, 1508, 1509                                                                                                                                                                                             |
| \makeatletter        | 2153, 8392, 9274, 9642                                                                                                                                                                                             |
| \makeatother         | 2154, 8409, 9291, 9642                                                                                                                                                                                             |
| \makebox             | 9007, 9119                                                                                                                                                                                                         |
| \maketitle           | 1264, 1265, 1355, 1361, 7925, 7926                                                                                                                                                                                 |
| \mapsto              | 7587, 7588                                                                                                                                                                                                         |
| \marginnote          | 8043, 9327                                                                                                                                                                                                         |
| \marginpar           | 8513, 8518, 8601, 8606,<br>8903, 8908, 9351, 9356, 9398, 9403                                                                                                                                                      |
| \mathbb              | 7601                                                                                                                                                                                                               |
| \mathbin             | 33, 7546, 7550, 7557, 7583, 7594                                                                                                                                                                                   |
| \mathclose           | 33, 5936, 7548, 7555, 7559,<br>7565, 7567, 7575, 7582, 7594, 7598                                                                                                                                                  |
| \mathhub             | 78, 139, 171, 30, <u>1038</u>                                                                                                                                                                                      |
| \mathop              | 33, 7585, 7596                                                                                                                                                                                                     |
| \mathopen            | 33, 5931, 7548, 7555, 7559,<br>7565, 7567, 7574, 7582, 7594, 7598                                                                                                                                                  |
| \mathord             | 33                                                                                                                                                                                                                 |
| \mathpunct           | 33, 4584, 5587, 7574, 7584, 7590                                                                                                                                                                                   |
| \mathrel             | 32, 33, 7551, 7588                                                                                                                                                                                                 |
| \mathrm              | 4606, 7574                                                                                                                                                                                                         |
| mathstructure (env.) | 96, <u>6135</u>                                                                                                                                                                                                    |
| \mathstructure       | 6179, 6180                                                                                                                                                                                                         |
| \mathtt              | 7554, 7555, 7604, 7607, 7611                                                                                                                                                                                       |
| \mcb (env.)          | 1, <u>8911</u>                                                                                                                                                                                                     |
| \mcb                 | 8439, 8922, 8947, 9062                                                                                                                                                                                             |
| \mcc                 | 69, 112, 8928, <u>8968</u>                                                                                                                                                                                         |
| \meaning             | 1020, 2419, 2682, 4265, 4550, 4635,<br>4661, 5188, 5193, 5400, 5410, 5411                                                                                                                                          |
| \medskip             | 8070, 8084, 8103                                                                                                                                                                                                   |
| \meet                | 63                                                                                                                                                                                                                 |
| \message             | 26, 7818, 7821, 7983, 9497                                                                                                                                                                                         |
| \mhframeimage        | 108                                                                                                                                                                                                                |
| \mhgraphics          | 83, <u>2058</u> , 8197, 8199                                                                                                                                                                                       |
| \mhinput             | 83, <u>1907</u>                                                                                                                                                                                                    |
| \mhtikzinput         | 118, <u>2058</u>                                                                                                                                                                                                   |
| \min                 | 74, 116                                                                                                                                                                                                            |
| \min                 | 8906                                                                                                                                                                                                               |
| min                  | 73, 110, 115                                                                                                                                                                                                       |
| \mmlarg              | <u>332</u>                                                                                                                                                                                                         |

\mmlintent ..... 332  
 \mmtdef ..... 7685, 7695  
 \MMTinclude ..... 7629  
 mmtinterface (env.) ..... 7658  
 \MMTrule ..... 7634  
 \mname ..... 87, 96  
 mode commands:  
     \mode\_if\_math:TF .....  
         .. 3387, 4004, 4005, 4012, 5290, 9207  
     \mode\_if\_vertical:TF ..... 4003, 4010  
 \MSC ..... 7628  
 msg commands:  
     \msg\_error: ..... 143  
     \msg\_error:nn ..... 1807, 7076, 7129, 8456  
     \msg\_error:nnn .....  
         158, 197, 245, 808, 944, 951, 2001,  
         2004, 2130, 3223, 3279, 3450, 3480,  
         4017, 4719, 4961, 6209, 6693, 6720  
     \msg\_error:nnnn ..... 347, 826, 831,  
         850, 885, 957, 1975, 2032, 2942,  
         3030, 3375, 3901, 4478, 4984, 5239,  
         5273, 5523, 5550, 6058, 6330, 7030  
     \msg\_fatal:nn ..... 1114  
     \msg\_fatal:nnnn ..... 1125, 2055, 2124  
     \msg\_none:nn ..... 82  
     \msg\_redirect\_module:nnn ..... 97  
     \msg\_redirect\_name:nnn ..... 101  
     \msg\_set:nnn ..... 79  
     \msg\_warning:nn ..... 1068  
     \msg\_warning:nnn ..... 1717, 1759  
     \msg\_warning:nnnn ..... 429, 1751  
 \mult ..... 40, 41, 91, 92  
 \multicolumn ..... 9597  
 \multiple ..... 73, 115

**N**

nassertion (env.) ..... 1  
 \Nat ..... 36  
 ndefinition (env.) ..... 1  
 \neginfprec 41, 90, 91, 4270, 4273, 4282,  
     4285, 4298, 5110, 5350, 5361, 5894  
 \newcommand ..... 440, 475, 1601, 1982,  
     1990, 1995, 2040, 2047, 2077, 2083,  
     2091, 2101, 2128, 7531, 7731, 7752,  
     7974, 8043, 8105, 8206, 8210, 8289,  
     8296, 8299, 8316, 8430, 8862, 8975,  
     9016, 9091, 9128, 9154, 9159, 9164,  
     9169, 9205, 9249, 9252, 9253, 9254,  
     9255, 9256, 9582, 9667, 9672, 9696  
 \newcounter ..... 7828, 7829,  
     7830, 7831, 7958, 7970, 8429, 9525  
 \NewDocumentCommand .....  
     .... 443, 1565, 1797, 1802, 1849,  
     1921, 1969, 2512, 2591, 4973, 5485,

5963, 5971, 5982, 6004, 6015, 6028,  
 6945, 7039, 7044, 7050, 7058, 7067,  
 7110, 7120, 7138, 7464, 7500, 7515,  
 7634, 7695, 7992, 8184, 8619, 9565  
 \NewDocumentEnvironment .. 478, 1344,  
     1359, 1381, 7520, 7658, 8225, 8240  
 \newenvironment ..... 1494,  
     1505, 7234, 7441, 8157, 8159, 9625  
 \newif 301, 324, 1907, 7860, 8375, 8383, 9632  
 \newlabel ..... 7784, 7785  
 \newlength ..... 7995, 7996, 7999  
 \newline ..... 8681, 8688, 8742, 8747,  
     8791, 8796, 8844, 8849, 9430, 9443  
 \newpage ..... 8091, 9256, 9644  
 \newsavebox ..... 8127  
 nexample (env.) ..... 1  
 \nextslide ..... 8227, 8242  
 \ninputref ..... 8204, 8206, 8210  
 \nobreak ..... 7262  
 \noindent ..... 1299, 1932, 1940, 6929,  
     7252, 8070, 8084, 8107, 8125, 8509,  
     8682, 8743, 8792, 8845, 9347, 9431  
 \nointerlineskip ..... 8217  
 \normalmarginpar ..... 9328  
 \notation .. 32, 33, 41, 88, 90, 95, 127,  
     130, 136, 3010, 4177, 7546, 7549,  
     7550, 7551, 7564, 7566, 7584, 7585,  
     7589, 7593, 7595, 7596, 7601, 7604  
 note (env.) ..... 1  
 notes ..... 73, 105, 110, 115  
 \notesfalse ..... 7872  
 notesslides commands:  
     \c\_notesslides\_notes\_bool .....  
         ... 7799, 7800, 7813, 7816, 7824,  
         7840, 7841, 7853, 7861, 7989, 8150,  
         8156, 8190, 8205, 8255, 8290, 8300  
     \c\_notesslides\_sectocframes\_bool  
         ..... 7842, 7945  
     \c\_notesslides\_topsect\_str 7843,  
         7866, 7869, 7930, 7933, 7934, 7937  
 notesslides internal commands:  
     \c\_\_notesslides\_class\_str .....  
         .... 7795, 7798, 7804, 7825  
     \\_\_notesslides\_define\_chapter: ...  
         .... 7935, 7938, 7950  
     \\_\_notesslides\_define\_part: ...  
         .... 7939, 7962  
     \\_\_notesslides\_do\_label:n 7875, 7911  
     \\_\_notesslides\_do\_sectocframes: ...  
         .... 7874, 7946  
     \\_\_notesslides\_do\_yes\_param:Nn ..  
         .... 8001,  
         8020, 8023, 8026, 8029, 8032, 8035  
     \c\_\_notesslides\_docopt\_str ... 7801

|                                                         |                                                                                                |                                                                               |             |
|---------------------------------------------------------|------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|-------------|
| \c__notesslides_document_str . . . . .                  | 7973, 7976                                                                                     | \number . . . . .                                                             | 73, 116     |
| \__notesslides_eat: . . . . .                           | 8160, 8164, 8167                                                                               | \numberline . . . . .                                                         | 7910        |
| \__notesslides_excursion_args:n . . . . .               | 8310, 8317                                                                                     | \numberproblemsin . . . . .                                                   | <u>8429</u> |
| \l__notesslides_excursion_id_str . . . . .              | 8306, 8312                                                                                     |                                                                               |             |
| \l__notesslides_excursion_intro_tl . . . . .            | 8307, 8311, 8321, 8323                                                                         | <b>O</b>                                                                      |             |
| \l__notesslides_excursion_mhrepos_str . . . . .         | 8308, 8313, 8322                                                                               | obeyedline † commands:                                                        |             |
| \l__notesslides_frame_allowdisplaybreaks_bool . . . . . | 8012, 8023                                                                                     | \obeyedline\uuuuuu\l_stex_-<br>current_domain_str . . . . .                   | 130         |
| \l__notesslides_frame_allowframebreaks_bool . . . . .   | 8011, 8020                                                                                     | \obeyedline\uuuuuu\l_stex_-<br>feature_name_str . . . . .                     | 130         |
| \__notesslides_frame_box_begin: . . . . .               | 8063, 8074, 8097                                                                               | \obeyedline\uuuuuu\l_stex_-<br>morphism_morphisms_seq\obeyedline<br>. . . . . | 130         |
| \__notesslides_frame_box_end: . . . . .                 | 8068, 8083, 8099                                                                               | \obeyedline\uuuuuu\l_stex_-<br>morphism_renames_prop . . . . .                | 130         |
| \l__notesslides_frame_fragile_bool . . . . .            | 8013, 8026                                                                                     | \obeyedline\uuuuuu\l_stex_-<br>morphism_symbols_prop . . . . .                | 130         |
| \l__notesslides_frame_label_str . . . . .               | 8010, 8018, 8093, 8094                                                                         | obeyedline <b>STEX</b> commands:                                              |             |
| \l__notesslides_frame_shrink_bool . . . . .             | 8014, 8029                                                                                     | \obeyedline\uuuuuu\stex_invoke_-<br>sequence_in:\obeyedline . . . . .         | 133         |
| \l__notesslides_frame_squeeze_bool . . . . .            | 8015, 8032                                                                                     | \obeyedline\uuuuuu\stex_invoke_-<br>sequence_range: . . . . .                 | 133         |
| \l__notesslides_frame_t_bool . . . . .                  | 8016, 8035                                                                                     | \obeyedline\uuuuuu\stex_structural_-<br>feature_morphism:nnnn . . . . .       | 130         |
| \__notesslides_inputref: . . . . .                      | 8203, 8204, 8207                                                                               | \obeyedline\uuuuuu\stex_structural_-<br>feature_morphism_end: . . . . .       | 130         |
| \__notesslides_notes_env:nnnn . . . . .                 | 8155,<br>8175, 8176, 8177, 8178, 8179, 8180                                                    | obeyedline commands:                                                          |             |
| \l__notesslides_num . . . . .                           | 7878, 7880, 7883,<br>7885, 7888, 7889, 7892, 7893, 7896,<br>7897, 7900, 7901, 7904, 7905, 7912 | \obeyedline\uuuuu\c_stex_persist_-<br>mode_bool . . . . .                     | 142         |
| \__notesslides_setup_itemize: . . . . .                 | 8039, 8096                                                                                     | obeyedline † commands:                                                        |             |
| \l__notesslides_slideshow_counter_int . . . . .         | 8239, 8241, 8243, 8244, 8247, 8248                                                             | \obeyedline\uuuuu\l_stex_argnames_-<br>seq\obeyedline . . . . .               | 125         |
| \l__notesslides_tmp . . . . .                           | 8339, 8344                                                                                     | \obeyedline\uuuuu\l_stex_assoc_-<br>args_count . . . . .                      | 125         |
| \g__notesslides_variables_prop . . . . .                | 8331, 8333, 8336, 8339                                                                         | \obeyedline\uuuuu\l_stex_current_-<br>args_tl . . . . .                       | 132         |
| \notesslidesfont . . . . .                              | 8066, 8081, 8222                                                                               | \obeyedline\uuuuu\l_stex_current_-<br>arity_str . . . . .                     | 132         |
| \notesslidesfooter . . . . .                            | 8070, 8084, 8220                                                                               | \obeyedline\uuuuu\l_stex_current_-<br>return_tl . . . . .                     | 132         |
| \notesslidestitleemph . . . . .                         | 8103, 8212                                                                                     | \obeyedline\uuuuu\l_stex_current_-<br>type_tl\obeyedline . . . . .            | 132         |
| \notestrue . . . . .                                    | 7862                                                                                           | \obeyedline\uuuuu\l_stex_get_-<br>symbol_args_tl . . . . .                    | 124         |
| nparagraph (env.) . . . . .                             | 1, 1                                                                                           | \obeyedline\uuuuu\l_stex_get_-<br>symbol_arity_int . . . . .                  | 124         |
| nsproof (env.) . . . . .                                | 1                                                                                              | \obeyedline\uuuuu\l_stex_get_-<br>symbol_def_tl . . . . .                     | 124         |
| \null . . . . .                                         | 7262                                                                                           | \obeyedline\uuuuu\l_stex_get_-<br>symbol_invoke_cs\obeyedline .               | 124         |
| \num . . . . .                                          | 144                                                                                            | \obeyedline\uuuuu\l_stex_get_-<br>symbol_macro_str . . . . .                  | 132         |
|                                                         |                                                                                                | \obeyedline\uuuuu\l_stex_get_-<br>symbol_mod_str . . . . .                    | 124         |

|                                                  |                                           |                           |
|--------------------------------------------------|-------------------------------------------|---------------------------|
| \obeyedline\l_stex_get_-                         | \only .....                               | 8244, 8248                |
| symbol_name_str .....                            | \oplus .....                              | 7556, 7557                |
| \obeyedline\l_stex_get_-                         |                                           |                           |
| symbol_return_tl .....                           | P                                         |                           |
| \obeyedline\l_stex_get_-                         | \PackageError .....                       | 8340                      |
| symbol_type_tl .....                             | \pagebreak .....                          | 8526, 9563                |
| \obeyedline\l_stex_import_-                      | \pagenumbering ...                        | 1481, 1491, 1502, 1513    |
| archive_str .....                                | \par .....                                | 143,                      |
| \obeyedline\l_stex_import_-                      | 355, 1297, 1301, 7235, 7262, 7305,        |                           |
| name_str .....                                   | 7362, 8070, 8084, 8107, 8112, 8120,       |                           |
| \obeyedline\l_stex_import_-                      | 8125, 8132, 8142, 8509, 8522, 8525,       |                           |
| path_str\obeyedline .....                        | 8556, 8681, 8688, 8742, 8747, 8791,       |                           |
| \obeyedline\l_stex_import_-                      | 8796, 8844, 8849, 8892, 8895, 8942,       |                           |
| uri_str .....                                    | 9082, 9347, 9377, 9385, 9430, 9443,       |                           |
| \obeyedline\l_stex_key_-                         | 9486, 9489, 9546, 9551, 9561, 9563        |                           |
| argnames_clist .....                             | \paragraph .....                          | 27, 82                    |
| \obeyedline\l_stex_key_-                         | \parsep .....                             | 7271,                     |
| args_str .....                                   | 8596, 8872, 8932, 9072, 9391, 9468        |                           |
| \obeyedline\l_stex_key_-                         | \part .....                               | 27, 82, 7966, 7967        |
| name_str .....                                   | \partname .....                           | 7880, 7963, 7964          |
| \obeyedline\l_stex_key_-                         | \parttitle .....                          | 7880                      |
| prec_str .....                                   | \PassOptionsToClass .....                 | 7803, 7804                |
| \obeyedline\l_stex_key_-                         | \PassOptionsToPackage .....               |                           |
| variant_str .....                                | 10, 7805, 7806, 7817,                     |                           |
| \obeyedline\l_stex_notation_-                    | 7820, 7845, 7846, 7863, 8372, 9268        |                           |
| macrocode_cs\obeyedline .....                    | \pause .....                              | 8105                      |
| obeyedline <i>STEX</i> commands:                 | \PDF .....                                | 15, 26                    |
| \obeyedline\stex_import_-                        | \pdfbookmark .....                        | 7912                      |
| module_uri:nn .....                              | \pdfdest .....                            | 1561                      |
| \obeyedline\STEXInternalAssocArgMarkerI          | peek commands:                            |                           |
| .....                                            | \peekCharCode:NTF .....                   |                           |
| \obeyedline\STEXInternalAssocArgMarkerII         | ... 3605, 5104, 5117, 5180, 5182,         |                           |
| .....                                            | 5304, 5311, 6352, 6358, 6499, 6510        |                           |
| \obeyedline\STEXInternalSymbolAfterInvokationNTF | \obeyedline\peekCharCode_remove:NTF ..... |                           |
| .....                                            | ... 5103, 5179, 5295, 5300, 5302,         |                           |
| \obeyedline\STEXInternalTermMathArgiii           | 5309, 5318, 5386, 5773, 6357, 6607        |                           |
| .....                                            | \phantom .....                            | 9240                      |
| \obeyedline\STEXInternalTermMathAssocArgIII      | \phantom \Pi .....                        | 7578                      |
| .....                                            | 128 \precondition .....                   | 39–41, 44, 90–92          |
| \obeyedline\STEXInternalTermMathOMAii            | 128 \prematurestop .....                  | 7731                      |
| plus .....                                       | 108, 7973                                 |                           |
| .....                                            | 128 \premise .....                        | 52, 101, 6997, 7138, 7145 |
| \obeyedline\STEXInternalTermMathOMBii            | \prg_new_commands:                        |                           |
| .....                                            | 128 \prg_new_conditional:Nnn .....        |                           |
| obeyedline commands:                             | ... 236, 286, 529,                        |                           |
| \obeyedline\g_stex_last_-                        | 534, 744, 754, 2160, 2570, 2574,          |                           |
| feature_str\obeyedline .....                     | 3678, 3689, 3693, 5529, 5641, 5652        |                           |
| obeyedline <i>STEX</i> commands:                 | \prg_new_protected_conditional:Nnn .....  |                           |
| \obeyedline\stex_structural_-                    | 764                                       |                           |
| feature_module_end: .....                        | \prg_return_false: ..                     | 238, 288, 532,            |
| \objective .....                                 | 537, 745, 749, 755, 757, 780, 784,        |                           |
| \OMDoc .....                                     | 2161, 2572, 2576, 3679, 3694, 5547,       |                           |
| \omdoc .....                                     | 5551, 5647, 5648, 5649, 5655, 5656        |                           |

```

\prg_return_true: ..... 238, 288,
      532, 537, 747, 749, 757, 784, 2161,
      2572, 2576, 3690, 5545, 5647, 5655
\printexcursion ..... 109
\printexcursions . 8288, 8297, 8318, 8326
problem (env.) ..... 1
problem commands:
  \g_problem_id_counter ..... .
      ..... 8627, 8640, 8642,
      8704, 8706, 8753, 8755, 8804, 8806
  \l_problem_inputproblem_keys_tl .
      ..... 8458, 8459, 8461, 8621
  \problem_mcc_box_default_tl ...
      ..... 8999, 9007, 9011, 9013
  \problem_mcc_box_tl .....
      ..... 8914, 8934, 8944, 8970, 9004
  \problem_scc_box_default_tl ...
      ..... 9115, 9119, 9123, 9125
  \problem_scc_box_tl .....
      ..... 9060, 9074, 9084, 9090, 9116
  \g_problem_total_min_fp .. 8448, 8504
  \g_problem_total_pts_fp .. 8447, 8503
\problemheader ..... 8509, 8529, 9347
problems internal commands:
  \__problems_activate_macros: ...
      ..... 8437, 8501, 8558
  \l__problems_anscls_int .....
      ..... 8799, 8865, 8867, 9449, 9451
  \c__problems_boxed_bool .....
      ..... 8368
  \__problems_do_yes_param:Nn .. 8948
  \__problems_exnote_start:n .....
      ..... 8750, 8768, 8771
  \l__problems_fillin_solution_tl .
      ..... 9175, 9198, 9222, 9230, 9234
  \__problems_fillinsol:nm .....
      ..... 9208, 9210, 9216, 9226
  \__problems_gnote_start:n .....
      ..... 8801, 8820, 8823
  \c__problems_gnotes_bool .....
      ..... 8358, 8822, 8834, 9308, 9498
  \l__problems_has_min_bool .....
      .. 8452, 8495, 8496, 8586, 8605, 9402
  \l__problems_has_pts_bool . 8451,
      8492, 8493, 8583, 8600, 9397, 9419
  \__problems_hint_start:n .....
      ..... 8701, 8719, 8722
  \c__problems_hints_bool .....
      ..... 8360, 8721, 8732
  \l__problems_in_problem_bool ...
      .. 8450, 8453, 8455, 8488, 8554, 8559
  \__problems_maybe_inline:n .....
      ..... 9040, 9052, 9080
  \__problems_mccline:n .....
      ..... 8929, 8944, 8988, 9031
  \c__problems_min_bool .....
      .. 8366, 8516, 8604, 8907, 9354, 9401
  \l__problems_min_tl .....
      ..... 8490, 8494, 8504, 8517,
      8518, 8545, 8561, 8587, 9355, 9356
  \c__problems_notes_bool .....
      ..... 8356, 8770, 8781
  \__problems_oldpar: .....
      ..... 9042, 9045
  \__problems_parse_fillin_-
      arg:nnnn 9179, 9180, 9181, 9185, 9197
  \l__problems_path_seq .....
      .. 8470, 8471, 8564, 8565, 8569, 8570
  \l__problems_prob_imports_tl . 8426
  \l__problems_prob_refnum_int . 8427
  \c__problems_pts_bool .....
      .. 8364, 8511, 8599, 8902, 9349, 9396
  \l__problems_pts_tl .....
      8489, 8491, 8503, 8512, 8513, 8545,
      8560, 8584, 9350, 9351, 9372, 9374
  \__problems_record_problem: .....
      ..... 8505, 8543
  \__problems_sscline:n .....
      ..... 9069, 9084, 9104, 9143
  \__problems_solution_start:n ...
      ..... 8637, 8656, 8659
  \c__problems_solutions_bool .....
      ..... 8362, 8378
  \__problems_subproblem_int 8497,
      8550, 8582, 8598, 9393, 9394, 9417
  \c__problems_test_bool 8370, 8384,
      8526, 9250, 9252, 9256, 9324, 9507
\ProcessKeysOptions .....
      ... 39, 7809, 7849, 8377, 9271, 9661
\ProcessOptions .....
      ..... 11
\prod ..... 7585, 7596
\prop ..... 42
prop commands:
  \prop_clear:N . 1169, 2904, 2905, 6698
  \prop_const_from_keyval:Nn . 112, 125
  \prop_gclear:N .....
      ..... 2343, 2344, 2345, 2455, 5444
  \prop_get:NnN .....
      ..... 1203
  \prop_get:NnNTF .....
      ..... 193, 899, 900, 1823, 1927,
      1928, 2003, 2958, 2979, 5530, 8339
  \prop_gput:Nnn .....
      2483, 2604, 2616, 2618, 2651, 5473,
      5509, 5515, 5533, 7106, 8333, 9578
  \prop_gset_eq:NN .....
      1186, 1217, 1220
  \prop_gset_from_keyval:Nn .....
      ..... 1189, 1214, 2448, 2450, 5457
  \prop_if_exist:NTF .....
      ..... 897, 1095, 1112, 1626,
      1720, 1822, 1887, 2000, 3103, 5456

```

|                          |                                                                                                                                        |                       |                                                                                                                                                         |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| \prop_if_in:NnTF         | 142, 177, 801, 6658, 6715, 6745, 6759                                                                                                  | \realize              | 63, 64, 3671, 3673, 3675                                                                                                                                |
| \prop_item:Nn            | 143, 917, 1096, 1209, 1644, 1723, 1889, 1892, 1901, 2065, 2079, 2094, 2884, 3040, 3045, 3106, 3131, 3142, 3157, 6650, 6690, 6718, 8336 | \ref                  | 84, 1667                                                                                                                                                |
| \prop_map_break:         | 2644                                                                                                                                   | \refstepcounter       | 8466, 9538                                                                                                                                              |
| \prop_map_break:n        | 2663, 2699, 2702, 2780, 3064, 4086, 4949, 4952, 7093                                                                                   | \relax                | . 354, 635, 636, 1476, 1486, 1561, 1718, 1932, 1940, 2105, 2191, 2229, 2233, 2458, 2459, 4004, 4011, 5231, 5265, 6043, 7014, 7926, 8191, 8193           |
| \prop_map_function:NN    | 2441, 2971                                                                                                                             | \renamedecl           | 64, 130, 3016, 3414, 3498                                                                                                                               |
| \prop_map_inline:Nn      | 2451, 2628, 2661, 2705, 2723, 2726, 2748, 2751, 2789, 2935, 2954, 2994, 3022, 3026, 4102, 4942, 6312, 6538, 7090, 9590                 | \renewcommand         | 7990, 8091, 8101, 9446, 9506                                                                                                                            |
| \prop_new:N              | 4787, 8331, 9572                                                                                                                       | \renewenvironment     | .. 8044, 8088, 8106, 8128, 8151, 8153                                                                                                                   |
| \prop_put:Nnn            | 1177, 1178, 1179, 1180, 1181, 2922, 2926, 3394, 3428, 4835, 5030, 6724, 6747, 6761, 9585                                               | repo commands:        |                                                                                                                                                         |
| \prop_remove:Nn          | 3490                                                                                                                                   | \repo_prop            | 139, 140                                                                                                                                                |
| \prop_set_eq:NN          | 1084, 1201, 1205, 1635                                                                                                                 | \reqpts               | 74, 116                                                                                                                                                 |
| \prop_to_keyval:N        | 1187, 1190, 1215, 2420, 2421, 2422, 2429, 2433, 2949, 2952, 5458                                                                       | \requiremodule        | 93, 94, 3012, 3363                                                                                                                                      |
| \protect                 | 7910, 8342                                                                                                                             | \RequirePackage       | 4, 13, 18, 23, 24, 151, 153, 202, 205, 5920, 7793, 7811, 7833, 7837, 7851, 7864, 7865, 8182, 8352, 8390, 9260, 9272, 9299, 9300, 9653, 9664, 9669, 9670 |
| \protected               | 123, 142, 353, 567, 568, 1265, 5980, 5994                                                                                              | \resizebox            | 9595, 9678, 9684, 9688                                                                                                                                  |
| \providecommand          | 2063, 2070, 7981, 9666                                                                                                                 | \reversemarginpar     | 9327                                                                                                                                                    |
| \ProvidesExplClass       | 5, 7792                                                                                                                                | \rhd                  | 8046, 8049                                                                                                                                              |
| \ProvidesExplPackage     | 19, 7836, 8351, 9259, 9652                                                                                                             | \right                | 5925                                                                                                                                                    |
| \pts                     | 8901                                                                                                                                   | \Rightarrow           | 7497                                                                                                                                                    |
| pts                      | 73, 110, 115                                                                                                                           | \rightmargin          | 7272, 8597, 8873, 8933, 9073, 9392, 9469                                                                                                                |
| <b>Q</b>                 |                                                                                                                                        |                       |                                                                                                                                                         |
| \qrcode                  | 9327                                                                                                                                   | \rule                 | 8681, 8688, 8742, 8747, 8791, 8796, 8844, 8849, 9430, 9443                                                                                              |
| \qrid                    | 9347, 9394, 9508                                                                                                                       | rustex commands:      |                                                                                                                                                         |
| \qrjson                  | 9309, 9311, 9313, 9314, 9361, 9366, 9373, 9381, 9384, 9407, 9409, 9414, 9420, 9425, 9434, 9440, 9455, 9459                             | \rustex_direct_HTML:n | 8113, 8121, 8133, 8143                                                                                                                                  |
| \qrschema                | 9322, 9327                                                                                                                             | \rustex_if:TF         | 8111, 8112, 8119, 8120, 8131, 8132, 8141, 8142                                                                                                          |
| \quad                    | 7889, 7893, 7897, 7901, 7905, 9206, 9212, 9554, 9557                                                                                   | \rustexBREAK          | 2157                                                                                                                                                    |
| quark commands:          |                                                                                                                                        | <b>S</b>              |                                                                                                                                                         |
| \quark_new:N             | 2184, 2467                                                                                                                             | sassertion (env.)     | 99                                                                                                                                                      |
| quark internal commands: |                                                                                                                                        | scb (env.)            | 9039                                                                                                                                                    |
| \q__stex_smsmode_break   | 2184, 2187, 2247                                                                                                                       | \scb                  | 8440, 8923, 9063, 9087                                                                                                                                  |
| <b>R</b>                 |                                                                                                                                        |                       |                                                                                                                                                         |
| \raisebox                | 9009, 9121                                                                                                                             | \scc                  | 9068, 9088                                                                                                                                              |
| \rangleangle             | 7565                                                                                                                                   | \scriptsize           | 8043                                                                                                                                                    |
| \realization             | 3561, 3563                                                                                                                             | \scriptstyle          | 8049                                                                                                                                                    |
| seq commands:            |                                                                                                                                        | sdefinition (env.)    | 99                                                                                                                                                      |
| \seq_clear:N             | 187, 678, 717, 2006, 2201, 2712, 2737, 2762, 2867, 2906, 4223, 4579, 4740, 5597,                                                       | \second               | 138                                                                                                                                                     |

5661, 5726, 5735, 6171, 6222, 6275,  
 6437, 6699, 6700, 6820, 6856, 7635  
`\seq_count:N` . . . . . 2049, 2121, 3572, 6890  
`\seq_gclear:N` . . . . . 2195, 2196, 5445  
`\seq_gclear_new:N` . . . . . 980, 981  
`\seq_get>NN` . . . . . 1011  
`\seq_get_right>NN` 163, 8471, 8565, 8570  
`\seq_gpop>NN` . . . . . 1006  
`\seq_gpush:Nn` . . . . . 994  
`\seq_gput_left:Nn` . . . . . 1570, 1575  
`\seq_gput_right:Nn` . . . . .  
                        1573, 2151, 2170, 5474  
`\seq_gset_eq>NN` . . . . .  
                        993, 1009, 1013, 2894, 2895, 2896, 7774  
`\seq_gset_split:Nnn` . . . . . 5463  
`\seq_if_empty:NTF` . . . . . 174, 715, 733,  
                        745, 755, 784, 830, 836, 841, 846,  
                        1005, 1008, 1113, 1974, 2031, 4063,  
                        6905, 7242, 7293, 7449, 7643, 7773  
`\seq_if_empty_p:N` . . . . . 771, 772, 2010  
`\seq_if_exist:NTF` . . . . . 1615, 5462  
`\seq_if_in:NnTF` . . . . . 1568, 1569, 1682,  
                        1684, 2013, 2023, 2295, 2304, 2681,  
                        2721, 2746, 2787, 6550, 6661, 6784  
`\seq_item:Nn` . . . . . 746,  
                        756, 2050, 2122, 4336, 4342, 5531, 6891  
`\seq_map_break:` . . . . . 1617, 1621, 2699  
`\seq_map_break:n` . . . . . 1621, 2702, 4086  
`\seq_map_function>NN` . . . . . 719, 723, 6449  
`\seq_map_inline:Nn` . . . . .  
                        269, 914, 918, 1620,  
                        1655, 1977, 2034, 2210, 2704, 4100,  
                        4741, 4768, 5736, 6238, 6289, 6298,  
                        6826, 6964, 6991, 7313, 7368, 7644  
`\seq_new:N` . . . . .  
                        . . . . . 230, 1528, 1574, 2141, 2164, 2314  
`\seq_pop>NN` . . . . . 716  
`\seq_pop_left>NN` . . . . . 173, 699,  
                        777, 778, 829, 834, 840, 845, 915,  
                        919, 921, 2017, 3574, 3576, 3583, 3587  
`\seq_pop_left:NNTF` . . . . .  
                        . . . . . 1173, 4310, 4312, 4320  
`\seq_pop_right>NN` . . . . . 165, 175,  
                        734, 789, 791, 796, 798, 800, 1141,  
                        2389, 3099, 4062, 4599, 4601, 6440  
`\seq_push:Nn` . . . . . 722  
`\seq_put_left:Nn` . . . . .  
                        . . . . . 700, 720, 2722, 2747, 6746, 6760  
`\seq_put_right:Nn` . . . . .  
                        . . . . . 169, 195, 233, 737, 792,  
                        802, 805, 977, 2014, 2018, 2024,  
                        2348, 2378, 2391, 2683, 2788, 2870,  
                        3001, 4293, 4300, 4321, 4323, 4581,  
                        4743, 4745, 5628, 5664, 5700, 5707,

5732, 5738, 5742, 5749, 6172, 6227,  
 6234, 6280, 6716, 6823, 6859, 6873  
`\seq_reverse:N` . . . . . 6441  
`\seq_set_eq>NN` . . . . . 725, 765, 766, 788,  
                        795, 913, 976, 2007, 4750, 4778, 5747  
`\seq_set_split:Nnn` . . . . . 164,  
                        679, 790, 797, 828, 833, 1133, 1172,  
                        2008, 2388, 3098, 3571, 3581, 4061,  
                        4309, 4313, 4598, 4600, 6439, 7636  
`\seq_use:Nn` . . . . .  
                        . . . . . 200, 203, 206, 762, 792, 805,  
                        833, 1175, 2393, 3100, 3579, 3585,  
                        3589, 4067, 4379, 4584, 4727, 4751,  
                        5464, 5586, 6906, 7243, 7294, 7450  
`\l_tmpa_seq` . . . . . 164,  
                        165, 169, 173, 174, 175, 187, 195,  
                        200, 203, 206, 4579, 4581, 4584,  
                        4598, 4599, 4600, 4601, 4740, 4743,  
                        4745, 4750, 4778, 6437, 6439, 6440,  
                        6441, 6449, 7635, 7636, 7643, 7644  
`\seqmap` . . . . . 96, 5196, 5654  
`\setbox` . . . . .  
                        136, 8153, 8168, 8663, 8724, 8773, 8825  
`\setcounter` . . . . . 9536, 9539  
`\setkeys` . . . . . 2067, 2081, 2096, 8195, 9673  
`\setlength` . . . . . 7270, 7271, 7272,  
                        7995, 7996, 8000, 8053, 8054, 8055,  
                        8595, 8596, 8597, 8871, 8872, 8873,  
                        8931, 8932, 8933, 9071, 9072, 9073,  
                        9390, 9391, 9392, 9467, 9468, 9469  
`\setlicensing` . . . . . 107  
`\setmetatheory` . . . . . 2493  
`\setnotation` . . . . . 33, 91, 4453  
`\setsectionlevel` . . . . . 27,  
                        82, 121, 1434, 7867, 7869, 7931, 7933  
`\setSGvar` . . . . . 109, 8332, 8342  
`\setsidelogo` . . . . . 107  
`\setsource` . . . . . 107  
`sexample (env.)` . . . . . 99  
`\sf` . . . . . 8213  
`\sffamily` . . . . . 8222  
`sfragment (env.)` . . . . . 82  
`sfragment commands:`  
     `\_sfragment_do_level:nn` . . . . .  
                        . . . . . 1295, 1307, 1323, 1327,  
                        1331, 1332, 1333, 1334, 1335, 7908  
     `\_sfragment_end:` 1303, 1317, 1349, 1377  
`\shorttitle` . . . . . 1365, 1366  
`\skipfragment` . . . . . 82, 1418, 1422, 1426  
`\slideframewidth` . . . . . 7999, 8000, 8076, 8191  
`\slideheight` . . . . . 7996  
`slides` . . . . . 105  
`\slidewidth` . . . . .  
                        . . . . . 7995, 8079, 8109, 8191, 8193, 8197

\smacro ..... 92, 93  
 \small ..... 7266  
 \smallskip ..... 7262, 8513, 8518,  
     8601, 8606, 8681, 8742, 8791, 8844,  
     9351, 9356, 9398, 9403, 9430, 9551  
 smodule (env.) ..... 86, 2538  
 \smodule ..... 3013  
 \Sn ..... 20, 89, 5951  
 \sn ..... 20, 24, 89, 5951  
 \Sns ..... 20, 89, 5951  
 \sns ..... 20, 89, 5951  
 \solution ..... 73  
 solution (env.) ..... 1, 8627  
 \solution ..... 8438, 8691, 8924, 9064  
 solutions ..... 73, 110, 115  
 \solutionsfalse ..... 8381, 8696  
 \solutionstrue ..... 8379, 8693  
 \source ..... 138  
 sparagraph (env.) ..... 99  
 \spfblock ..... 7325, 7525  
 \spfjust ..... 7326, 7531, 7534  
 spfsketch ..... 7234  
 \spfsketchenvautorefname ..... 7252  
 \spfstep ..... 7320, 7427  
 \spfstepautorefname ..... 7361, 7442  
 \spfstepenvautorefname ..... 7442  
 sproblem (env.) ..... 8437  
 \sproblemautorefname ..... 8535  
 sproof (env.) ..... 101, 7268  
 \sproofautorefname ..... 7340  
 \sproofend ..... 7254, 7342, 7359  
 \square ..... 7255, 8971, 9000  
 \sr ..... 20, 24, 88, 5951  
 \sref ..... 84, 85, 99, 1579, 8292  
 \sreflabel ..... 84, 86, 121, 1528  
 \srefsetin ..... 84, 1601  
 \srefsym ..... 89, 1849  
 \srefsymuri ..... 89, 1880  
 \startsolutions ..... 111, 8692  
 \stepcounter ..... 1415, 7909, 8090  
 \sTeX ..... 14, 15, 81  
 \stex ..... 15, 24, 81  
 stex commands:  
     \l\_stex\_aB\_args\_seq .....  
         4727, 4741, 4750, 4751, 4768, 4778,  
         5586, 5597, 5628, 5661, 5664, 5700,  
         5707, 5726, 5732, 5736, 5747, 5749  
     \l\_stex\_activate\_module:n .....  
         122, 2359, 2408,  
         2680, 2680, 2692, 2862, 3205, 3303,  
         3346, 3546, 6246, 6306, 6831, 7681  
     \l\_stex\_add\_definiens:nn .....  
         3018, 7080, 7088, 7315, 7370  
     \l\_stex\_add\_definiens\_inner:nnnnnnnnn  
         ..... 7093, 7104  
 \l\_stex\_add\_module\_notation:nnnnn ..... 122  
 \l\_stex\_all\_modules\_seq .....  
     ..... 122, 2201, 2314,  
     2348, 2378, 2681, 2683, 2704, 4100  
 \l\_stex\_allow\_semantic\_bool .....  
     4980, 5122, 5200, 5201, 5204, 5230,  
     5264, 5434, 5443, 5559, 5570, 5667,  
     5752, 5810, 5821, 5854, 5883, 6044,  
     6125, 6543, 6574, 6591, 6636, 7017  
 \l\_stex\_annotation:nn ..... 334,  
     337, 1278, 1287, 1299, 1771, 1991,  
     1992, 3387, 3823, 3826, 3833, 3838,  
     3840, 4382, 4388, 4389, 4397, 4408,  
     4419, 4420, 4423, 4424, 4428, 4884,  
     4887, 4894, 4898, 4901, 5077, 5080,  
     5087, 5091, 5094, 5125, 5575, 5582,  
     5675, 5784, 5797, 5814, 5839, 5847,  
     5868, 5876, 6070, 6463, 6471, 6476,  
     6684, 6930, 6951, 7028, 7098, 7131,  
     7143, 7240, 7282, 7285, 7327, 7381,  
     7390, 7394, 7447, 7516, 7532, 7641,  
     7646, 8282, 8532, 8876, 8903, 8908,  
     8990, 8993, 9106, 9109, 9217, 9472  
 \l\_stex\_annotation:nnn ..... 141, 142  
 \l\_stex\_annotation\_force\_break:n .....  
     1300, 3386, 3820, 3838, 4881, 4899,  
     5075, 5092, 5582, 5680, 5789, 5802,  
     5819, 5844, 5852, 5873, 5881, 6475,  
     6930, 6952, 7099, 7132, 7642, 9219  
 \l\_stex\_annotation\_invisible:n .....  
     142, 1385, 1407, 2555, 2812, 2856,  
     5227, 5487, 7280, 7639, 7676, 8577  
 \l\_stex\_annotation\_invisible:nn .....  
     ..... 1251, 1446, 1454,  
     1460, 1466, 1932, 1940, 3084, 3308,  
     3349, 3385, 3419, 3460, 3470, 3802,  
     4375, 4862, 5059, 5820, 5853, 5882,  
     6240, 6300, 6828, 7114, 7630, 7677,  
     7745, 7766, 8481, 8884, 9188, 9480  
 \l\_stex\_annotation\_invisible:nnn .. 142  
 \l\_stex\_argnames\_seq ..... 125  
 \l\_stex\_args\_end: ..... 5448,  
     5472, 5475, 5996, 6000, 6050, 7023  
 \l\_stex\_assign\_do:n .....  
     ..... 2877, 3367, 3372, 3600  
 \l\_stex\_assoc\_args\_count .....  
     ..... 125, 3852, 3856, 3898, 3899  
 \l\_stex\_brackets\_dones\_bool .....  
     .. 4648, 5900, 5903, 5904, 5928, 5929  
 \l\_stex\_capitalize:n .....  
     ..... 1289, 5987, 5990, 6033, 7061

```

\stex_check_term:n ..... 124, 125, 2200, 3379,
..... 3698, 3699, 3707, 3918, 4355, 7538
\c_stex_check_terms_bool ..... 33, 3685, 3688, 7541
\stex_close_module: ... 122, 2415,
2415, 2564, 2819, 7622, 7689, 7691
\l_stex_current_archive ..... 1081
\l_stex_current_archive_prop ...
..... 135, 140, 917, 927,
932, 1084, 1095, 1096, 1105, 1194,
1626, 1627, 1720, 1723, 1822, 1823,
1887, 1889, 1892, 1927, 1928, 2000,
2003, 2065, 2079, 2094, 3103, 3106
\l_stex_current_args_tl .....
132, 4570, 4574, 5253, 5447, 5448, 5451
\l_stex_current_arity_str .....
..... 132, 4580, 5142, 5169,
5171, 5252, 5356, 5374, 5375, 5406
\l_stex_current_doc_uri 140, 961,
963, 964, 1538, 1543, 1545, 1549,
1615, 1617, 1657, 1674, 1675, 6909
\l_stex_current_domain_str .....
..... 130, 2826, 2829, 2831,
2839, 2844, 2853, 2862, 2893, 2908,
2920, 2957, 2969, 3437, 3485, 3506,
3526, 3546, 3550, 3623, 3643, 3664
\g_stex_current_file .....
. 138–140, 163, 926, 928, 933, 981,
984, 986, 988, 989, 990, 993, 994,
1009, 1012, 1013, 1630, 1728, 2367,
3113, 3119, 3120, 3168, 3187, 3231,
3257, 7773, 7774, 8470, 8564, 8569
\l_stex_current_language_str 140,
141, 139, 141, 176, 181, 2549, 3218,
3221, 3245, 3248, 7670, 8475, 8574
\l_stex_current_module ..... 123, 126
\l_stex_current_module_str .....
. 122, 144, 2202, 2313, 2330, 2347,
2348, 2363, 2376, 2377, 2378, 2379,
2418, 2419, 2420, 2421, 2422, 2428,
2430, 2434, 2439, 2441, 2548, 2558,
2571, 2579, 2583, 2604, 2610, 2616,
2618, 2628, 2635, 2640, 2649, 2651,
2661, 2685, 2688, 2807, 2818, 2852,
2863, 2960, 2963, 3392, 3426, 3453,
3462, 3472, 3756, 3794, 3803, 3902,
3964, 3973, 3982, 3988, 4527, 4546,
4547, 6128, 6166, 6171, 6172, 6192,
6291, 6313, 6874, 6897, 7089, 7090,
7091, 7092, 7105, 7106, 7660, 7661,
7664, 7669, 7688, 7690, 7703, 7717
\l_stex_current_ns_uri ..... 140,
967, 968, 2385, 2528, 3169, 3174,
..... 3175, 3188, 3193, 3194, 7542, 7623
\l_stex_current_redo_tl .....
..... 5234, 5247, 5258, 5261,
5268, 5402, 6408, 6446, 6481, 6631
\l_stex_current_return_tl .....
132, 4011, 5254, 5323, 5335, 5357, 5407
\stex_current_section_level .....
..... 1274, 1280, 1289, 1315, 7920
\l_stex_current_symbol_str .....
..... 132, 4207, 4237,
4356, 4396, 4407, 4516, 4535, 4598,
4824, 4976, 4978, 4984, 5019, 5109,
5112, 5127, 5155, 5156, 5239, 5251,
5273, 5289, 5321, 5322, 5325, 5331,
5348, 5349, 5352, 5441, 5523, 5550,
5567, 5668, 5669, 5712, 5753, 5754,
5786, 5799, 5816, 5841, 5849, 5870,
5878, 5931, 5936, 6009, 6022, 6035,
6045, 6058, 6066, 6070, 6072, 6127,
6489, 6490, 7018, 7026, 7028, 7030
\l_stex_current_term_tl 4981, 5202,
5256, 5568, 5670, 5671, 5672, 5755,
5783, 5796, 5811, 5822, 5838, 5855,
5867, 5884, 6053, 6482, 6488, 6678
\stex_current_this: . 5248, 6124, 6134
\l_stex_current_this_tl .....
..... 6126, 6130, 6144, 6147
\l_stex_current_type_tl .....
..... 132, 5124, 5255, 6350,
6405, 6412, 6415, 6419, 6734, 6783
\stex_deactivate_macro:Nn .....
..... 143, 344, 344,
2600, 3007, 3008, 3009, 3010, 3011,
3012, 3013, 3294, 3363, 3497, 3498,
3499, 3516, 3537, 3561, 3629, 3650,
3671, 3767, 3996, 4214, 4565, 6179,
6256, 6322, 7042, 7048, 7056, 7064,
7085, 7118, 7136, 7145, 7420, 7493,
7513, 7518, 7525, 7534, 7632, 7656,
7729, 8691, 8749, 8798, 8851, 8900,
8922, 8923, 8924, 8925, 8926, 8927,
8947, 9038, 9062, 9063, 9064, 9065,
9066, 9067, 9087, 9151, 9248, 9494
\stex_debug:n ..... 120, 68, 68, 98, 102, 181, 200, 251,
509, 634, 702, 709, 856, 926, 929,
964, 968, 979, 1020, 1028, 1056,
1061, 1080, 1083, 1116, 1159, 1187,
1198, 1208, 1228, 1244, 1259, 1544,
1609, 1614, 1654, 1658, 1674, 1677,
1683, 1685, 1689, 1692, 1698, 1700,
1703, 1712, 1713, 1739, 1786, 1788,
1790, 1885, 1891, 1896, 1903, 2253,
2255, 2271, 2273, 2277, 2294, 2296,
```

2303, 2305, 2327, 2340, 2353, 2355,  
 2377, 2417, 2495, 2497, 2506, 2610,  
 2635, 2643, 2647, 2682, 2948, 2980,  
 2984, 2987, 3097, 3102, 3104, 3112,  
 3119, 3124, 3128, 3167, 3186, 3203,  
 3209, 3216, 3232, 3243, 3258, 3276,  
 3284, 3286, 3373, 3380, 3417, 3438,  
 3454, 3458, 3468, 3486, 3488, 3570,  
 3573, 3577, 3919, 3921, 3923, 3925,  
 4043, 4060, 4099, 4110, 4112, 4265,  
 4297, 4308, 4358, 4546, 4550, 4626,  
 4635, 4640, 4661, 4810, 4929, 4969,  
 5188, 5193, 5289, 5294, 5299, 5321,  
 5324, 5327, 5348, 5400, 5409, 5433,  
 5441, 5557, 5907, 6190, 6265, 6349,  
 6889, 6892, 6894, 6898, 7078, 7091,  
 7105, 7140, 7314, 7369, 7468, 7665  
`\c_stex_debug_clist` . . . . .  
     . . . . . 28, 38, 69, 72, 86, 90, 92, 96, 100  
`\c_stex_default_metatheory` 2199, 7625  
`\l_stex_default_notation` . . . . .  
     . . . . . 4578, 4583, 4586, 4588, 4589,  
     4606, 5159, 5336, 5341, 5363, 5696  
`\stex_do_default_notation:` . . . . .  
     . . . . . 4568, 5158, 5334, 5695  
`\stex_do_default_notation_op:` . . . . .  
     . . . . . 4568, 4569, 4604, 5360  
`\_stex_do_deprecation:n` . . . . .  
     . . . . . 120, 427, 427, 2333, 3743, 3847  
`\_stex_do_for_list:` . . . . .  
     . . . . . 3017, 6855, 6870, 7238, 7309, 7364  
`\_stex_do_id:` . . . . . 133, 432, 432,  
     1347, 1374, 6934, 6949, 7239, 7318,  
     7378, 7387, 7396, 7476, 8500, 9542  
`\stex_do_up_to_module:n` . . . . .  
     123, 2578, 2578, 2581, 2588, 3345, 6337  
`\l_stex_docheader_sect` . . . . .  
     121, 1272, 1298, 1314, 1322, 1324,  
     1328, 1380, 1392, 1394, 1397, 1400,  
     1406, 1419, 1421, 1425, 1436, 1437,  
     1438, 1439, 1440, 1441, 1443, 1446,  
     1452, 1456, 1462, 1466, 7912, 7919  
`\_stex_eat_exclamation_point:` . . . . .  
     . . . . . 3798, 5325, 5336, 5772, 5774  
`\_stex_end:` . . . . . 416, 424, 2915, 2923  
`\_stex_every_file:` . . . . .  
     . . . . . 995, 998, 1003, 1015, 7777  
`\stex_every_module:n` . . . . .  
     . . . . . 122, 2315, 2318,  
     2602, 3333, 3517, 3538, 3562, 3630,  
     3651, 3672, 3768, 3997, 4215, 4566,  
     6180, 6257, 6324, 6334, 7633, 7657  
`\g_stex_every_module_t1` . . . . .  
     . . . . . 2316, 2319, 2331

`\l_stex_every_symbol_t1` 5203, 5208,  
     5210, 5211, 5217, 5218, 5221, 5259  
`\stex_execute_in_module:n` . . . . .  
     . . . . . 122, 2332, 2406, 2586,  
     2586, 2590, 2621, 2653, 3302, 3545,  
     3981, 6170, 6233, 6245, 6288, 6305  
`\stex_fatal_error:n` . . . . . 143  
`\stex_fatal_error:nmn` . . . . . 143  
`\l_stex_feature_name_str` . . . . .  
     . . . . . 130, 2859, 2863,  
     2892, 2963, 2968, 2984, 2985, 2987,  
     2988, 3031, 3376, 3392, 3426, 3430  
`\stex_file_in_smsmode:nn` . . . . .  
     . . . . . 136, 2175, 2194, 2223, 2369, 3272  
`\stex_file_resolve:Nn` . . . . . 138,  
     139, 683, 689, 705, 712, 874, 975,  
     984, 986, 989, 1044, 1071, 1073,  
     1629, 1647, 1737, 3120, 3214, 3241  
`\stex_file_set:Nn` . . . . . 138, 139, 677,  
     677, 682, 696, 707, 870, 917, 1012, 1069  
`\stex_file_split_off_ext:NN` . . . . .  
     . . . . . 138, 787, 787, 892, 2367, 3113  
`\stex_file_split_off_lang:NN` . . . . .  
     . . . . . 138, 787,  
     794, 889, 2368, 3114, 8470, 8564, 8569  
`\stex_file_use:N` . . . . . 138,  
     139, 702, 709, 761, 761, 838, 843,  
     848, 902, 908, 923, 926, 979, 990,  
     994, 1049, 1069, 1074, 1079, 1123,  
     1126, 1159, 1160, 1162, 1196, 1630,  
     1631, 1643, 1722, 1728, 1733, 1738,  
     1740, 1888, 1895, 1900, 1912, 1915,  
     1939, 1944, 1955, 2011, 2021, 2061,  
     2065, 2075, 2079, 2089, 2094, 2370,  
     3116, 3119, 3120, 3122, 3143, 3158,  
     3168, 3187, 3215, 3231, 3242, 3257  
`\c_stex_filepath_sep_str` . . . . . 106  
`\stex_filestack_pop:` . . . . .  
     . . . . . 138, 1004, 1004, 1029, 1036, 2220  
`\stex_filestack_push:n` . . . . .  
     . . . . . 138, 982, 982, 1022, 1024, 2204  
`\l_stex_fors_seq` . . . . . 2867,  
     2870, 6856, 6859, 6873, 6890, 6891,  
     6905, 6906, 6964, 6991, 7242, 7243,  
     7293, 7294, 7313, 7368, 7449, 7450  
`\stex_get_env:Nn` . . . . .  
     . . . . . 143, 51, 52, 63, 84, 305, 600,  
     606, 971, 973, 1040, 1042, 1047, 3682  
`\stex_get_in_morphism:n` . . . . .  
     132, 2869, 3020, 3020, 3366, 3410, 3595  
`\_stex_get_mathstructure:n` . . . . .  
     . . . . . 2828, 6207, 6212  
`\stex_get_mathstructure:n` . . . . .  
     . . . . . 6206, 6206, 6224, 6276, 6819

```

\l_stex_get_structure_module_str
..... 2829, 6208, 6213, 6217,
6230, 6232, 6234, 6284, 6286, 6289
\_stex_get_symbol:n . 4015, 4021, 6214
\stex_get_symbol:n ..... 124,
126, 132, 1850, 4014, 4014, 4179,
4454, 4970, 5277, 5966, 5974, 5985,
6858, 7016, 7071, 7124, 7733, 7754
\l_stex_get_symbol_args_tl .....
..... 125, 127,
236, 3069, 3399, 3787, 3909, 3911,
3912, 4049, 4090, 4122, 4133, 4135,
4148, 4574, 4839, 4852, 4934, 5034,
5047, 5282, 5451, 6195, 6882, 7710
\l_stex_get_symbol_arity_int ...
..... 125, 127, 236,
3068, 3398, 3786, 3831, 3855, 3861,
3865, 3869, 3873, 3877, 3881, 3885,
3889, 3893, 3896, 3897, 3898, 3899,
3910, 3969, 4041, 4048, 4089, 4121,
4225, 4260, 4292, 4299, 4319, 4369,
4487, 4809, 4838, 4851, 4892, 4933,
5033, 5046, 5085, 5281, 6881, 7709
\l_stex_get_symbol_def_tl .....
..... 2888, 3070,
3374, 4050, 4091, 4123, 4935, 5283
\l_stex_get_symbol_invoke_cs ...
..... 3073, 3403,
4053, 4094, 4126, 4938, 5286, 6216
\l_stex_get_symbol_macro_str ...
..... 132, 3067, 3397
\l_stex_get_symbol_mod_str .....
..... 1852, 2871, 2881, 3065,
3381, 3385, 3395, 3420, 3429, 3973,
3979, 4022, 4046, 4087, 4119, 4185,
4189, 4204, 4367, 4456, 4460, 4464,
4468, 4475, 4479, 4492, 4496, 4547,
4554, 4931, 5279, 6046, 6860, 7019,
7073, 7126, 7719, 7725, 7746, 7767
\l_stex_get_symbol_name_str 1852,
2871, 2882, 3021, 3025, 3029, 3066,
3373, 3375, 3381, 3385, 3395, 3417,
3420, 3429, 3430, 3974, 3979, 4016,
4023, 4047, 4088, 4120, 4185, 4189,
4203, 4204, 4367, 4456, 4460, 4464,
4468, 4475, 4479, 4492, 4496, 4504,
4509, 4512, 4548, 4554, 4932, 4958,
4960, 4966, 4968, 5280, 5976, 5987,
6009, 6020, 6022, 6033, 6035, 6046,
6047, 6048, 6049, 6215, 6860, 7019,
7020, 7021, 7022, 7053, 7061, 7069,
7073, 7114, 7122, 7126, 7720, 7725,
7732, 7734, 7746, 7753, 7755, 7767
\stex_get_symbol_or_var:n .....
..... 126, 4928, 4965
\l_stex_get_symbol_return_tl ...
..... 3072, 3402, 4052,
4093, 4125, 4811, 4937, 5000, 5285
\l_stex_get_symbol_type_tl ...
... 3071, 3401, 4051, 4092, 4124,
4936, 5284, 6217, 6225, 6278, 6821
\stex_get_var:n ..... 126, 4503,
4928, 4957, 6007, 6018, 6031, 7112
\c_stex_home_file ..... 139, 1038
\stex_if_check_terms: .....
..... 124, 3678, 3689, 3693
\stex_if_check_terms:TF 124, 3677,
3698, 4181, 4506, 4813, 5009, 5012
\stex_if_check_terms_p: ... 124, 3677
\stex_if_do_html: .....
..... 286
\stex_if_do_html:TF .....
..... 141, 283, 295, 1250, 1277,
1286, 1560, 2546, 2566, 2804, 2820,
2850, 2898, 3083, 3307, 3348, 3384,
3418, 3777, 3965, 3977, 4183, 4553,
4807, 4815, 4861, 5008, 6239, 6299,
6827, 6885, 7113, 7310, 7328, 7338,
7355, 7365, 7399, 7406, 7408, 7446,
7667, 7692, 7724, 8224, 8468, 8507,
8510, 8530, 8567, 8592, 8645, 8655,
8667, 8673, 8709, 8718, 8728, 8734,
8758, 8767, 8777, 8783, 8810, 8819,
8830, 8836, 8875, 8913, 8939, 8989,
9051, 9079, 9105, 9206, 9212, 9471
\stex_if_do_html_p: .....
..... 141
\stex_if_file_absolute:N 138, 744, 754
\stex_if_file_absolute:NTF .....
..... 138, 742, 988, 1072
\stex_if_file_absolute_p:N . 138, 742
\stex_if_file_starts_with:NN 138, 764
\stex_if_file_starts_with:NNTF ..
..... 138, 764, 1195
\stex_if_html_backend:TF .....
... 141, 300, 325, 332, 615, 1250,
1294, 1384, 1403, 1450, 1962, 1989,
3677, 5231, 5265, 5780, 5836, 5865,
6043, 6069, 6928, 6938, 7014, 7812,
7852, 7982, 8062, 8110, 8118, 8130,
8140, 8281, 8969, 9089, 9184, 9215
\stex_if_html_backend_p: ... 141, 300
\stex_if_in_module: .....
..... 2570
\stex_if_in_module:TF .....
..... 122, 2322, 2570, 2586, 4485
\stex_if_in_module_p: ... 122, 2570
\stex_if_module_exists:n .....
..... 2574
\stex_if_module_exists:nTF .....
..... 122, 2339, 2352, 2574, 3171, 3174,
3190, 3193, 3278, 6226, 6279, 6822

```

```

\stex_if_module_exists_p:n 122, 2574
\stex_if_smsmode: ..... 2160
\stex_if_smsmode:TF .....
. 136, 433, 1242, 1546, 2158, 2225,
2358, 2557, 2565, 3087, 3230, 3256,
3311, 3352, 3504, 3511, 3524, 3532,
3548, 3556, 3621, 3641, 3662, 3754,
3986, 4192, 4556, 6921, 6937, 6950,
6963, 6990, 7003, 7097, 8589, 8591
\stex_if_smsmode_p: ..... 136, 2158
\stex_ignore_spaces_and_pars: ...
..... 143,
353, 353, 356, 2596, 2597, 8523, 9378
\l_stex_import_archive_str . 131,
2498, 2503, 2835, 3080, 3105, 3109,
3129, 3130, 3131, 3299, 3326, 3342
\stex_import_module_uri:nn .....
..... 131, 2496, 2832,
3078, 3096, 3096, 3297, 3323, 3340
\l_stex_import_name_str .....
131, 2500, 2505, 2837, 3082, 3090,
3099, 3301, 3315, 3328, 3344, 3356
\l_stex_import_ns_str .....
2506,
2509, 2839, 3085, 3089, 3172, 3175,
3178, 3191, 3194, 3197, 3205, 3303,
3306, 3309, 3314, 3346, 3350, 3355
\l_stex_import_path_str .....
..... 131, 2499, 2504,
2836, 3081, 3100, 3111, 3115, 3119,
3120, 3121, 3124, 3300, 3327, 3343
\stex_import_require_module:nnn .
..... 131, 2502, 2834,
3079, 3134, 3134, 3147, 3298, 3341
\stex_import_require_module_-
safe:nnn ..... 3149, 3325
\l_stex_import_uri_str .....
..... 131, 3110, 3131, 3137,
3142, 3152, 3157, 3165, 3167, 3171,
3172, 3178, 3184, 3186, 3190, 3191,
3197, 3216, 3223, 3243, 3278, 3279
\stex_in_archive:nn 135, 1088, 1088,
1910, 1961, 1986, 2044, 2097, 2135
\stex_in_invisible_html_bool ...
..... 3821, 4882, 5778, 5779, 5809
\l_stex_in_meta_bool 2401, 2402, 2407
\l_stex_inparray_bool ..... 5909
\stex_input_with_hooks:n .....
.. 1017, 1912, 1915, 1936, 1953, 1955
\stex_invoke_notation:w .....
..... 5194, 5311, 5312, 5317, 5358
\stex_invoke_outer_field: .....
6200
\stex_invoke_sequence: .....
..... 5038, 5051, 5102, 5102, 5646
\stex_invoke_structure: .....
..... 6168, 6216, 6343
\stex_invoke_symbol ..... 132
\stex_invoke_symbol: .....
..... 125, 126, 132, 3791,
4843, 4856, 5288, 5288, 6883, 7714
\l_stex_invoke_symbol:nnnnnnN ...
..... 123, 132,
2639, 4030, 4044, 4045, 5229, 5229,
5242, 5278, 6749, 6763, 6771, 6776
\l_stex_invoke_symbol:nnnnnnN\obeyedlineuuuuu\l_-
stex_current_symbol_str .....
132
\stex_invoke_text_symbol: 3961, 4009
\l_stex_invoke_variable:nnnnnn . 5263
\l_stex_invoke_variable:nnnnnnN ..
..... 132, 4849, 5044, 5263, 5644
\l_stex_is_sequentialized:n .....
..... 5638, 5665, 5740, 5750
\stex_iterate_break: 123, 2695, 2698
\stex_iterate_break:n . 123, 2695,
2701, 2778, 3459, 3469, 3484, 4118
\stex_iterate_morphisms:nn .....
..... 2761, 2761, 3437, 3453
\stex_iterate_notations:nn .....
..... 126, 2736, 2736, 2957, 6783
\stex_iterate_symbols:n .....
..... 123, 2694, 2694, 4111
\stex_iterate_symbols:nn .....
123, 2711, 2711, 2920, 3487, 6184, 6734
\l_stex_key_answerclass_str .....
..... 8630, 8634, 8648, 8683, 8684
\l_stex_key_archive_str .....
..... 133, 381, 384, 1593, 1611,
1625, 1634, 1635, 1712, 1719, 1733
\l_stex_key_argnames_clist .....
125
\l_stex_key_args_str .....
.....
125, 3710, 3715, 3724, 3760,
3804, 3857, 3862, 3866, 3870, 3874,
3878, 3882, 3886, 3890, 3894, 3913,
4531, 4864, 4995, 4996, 5061, 6840
\l_stex_key_argtypes_clist .....
..... 3729, 3734, 3837, 3839,
3926, 4897, 4900, 5090, 5093, 6844
\l_stex_key_assoc_str 3713, 3718,
3811, 3812, 4868, 4869, 5065, 5066
\l_stex_key_autogradable_bool .....
..... 8411, 8417, 8422, 8476, 8575
\l_stex_key_continues_t1 . 7151, 7160
\l_stex_key_def_t1 125, 3726, 3738,
3759, 3788, 3825, 3826, 3922, 3933,
3937, 3958, 4530, 4840, 4853, 4886,
4887, 5035, 5048, 5079, 5080, 6842
\l_stex_key_deprecate_str .....
..... 407, 409, 428, 429

```

\l\_stex\_key\_due\_tl .....  
     ..... 9517, 9521, 9556, 9557  
 \l\_stex\_key\_feedback\_tl .....  
     ..... 8856, 8859, 8883,  
     8886, 8895, 8896, 8956, 8961, 8983,  
     8984, 9025, 9026, 9099, 9100, 9137,  
     9138, 9462, 9479, 9482, 9489, 9490  
 \l\_stex\_key\_file\_str .....  
     ..... 133, 382, 385, 1594,  
     1610, 1613, 1630, 1645, 1673, 1688,  
     1699, 1712, 1724, 1728, 1734, 1806  
 \l\_stex\_key\_for\_clist 2868, 6839,  
     6847, 6857, 7148, 7157, 7429, 7434  
 \l\_stex\_key\_for\_str ..... 6927  
 \l\_stex\_key\_from\_tl ..... 7149, 7158  
 \l\_stex\_key\_Ftext\_tl .....  
     .. 8959, 8965, 8981, 9023, 9097, 9135  
 \l\_stex\_key\_functions\_tl . 7153, 7161  
 \l\_stex\_key\_given\_tl .....  
     ..... 9516, 9520, 9553, 9554  
 \l\_stex\_key\_hide\_bool 7155, 7164, 7297  
 \l\_stex\_key\_id\_str .....  
     ... 133, 389, 391, 434, 435, 6908,  
     6909, 6976, 6978, 7375, 7384, 7473,  
     8639, 8641, 8647, 8703, 8705, 8711,  
     8752, 8754, 8760, 8803, 8805, 8812,  
     8864, 8866, 8874, 8877, 9368, 9416,  
     9436, 9448, 9450, 9461, 9470, 9473  
 \l\_stex\_key\_intent\_args\_clist ...  
     ..... 4162, 4168, 4334, 4343, 4345  
 \l\_stex\_key\_intent\_str .....  
     ..... 3971, 4161, 4167, 4257, 4337  
 \l\_stex\_key\_macroname\_str .....  
     .. 6838, 6848, 6865, 6867, 6876, 6880  
 \l\_stex\_key\_method\_tl .....  
     .. 7154, 7163, 7284, 7285, 7431, 7435  
 \l\_stex\_key\_mhrepos\_str .....  
     .. 8416, 8424, 8613, 8615, 8623, 9569  
 \l\_stex\_key\_min\_tl .....  
     .. 8414, 8420, 8490, 8587, 8606, 9403  
 \l\_stex\_key\_name\_str .....  
     ..... 125, 3723, 3731,  
     3756, 3757, 3771, 3772, 3785, 3794,  
     3803, 3847, 3902, 3931, 3935, 3945,  
     3946, 3948, 3956, 3964, 3974, 3982,  
     3988, 3989, 4504, 4527, 4528, 4546,  
     4548, 4799, 4800, 4810, 4817, 4820,  
     4835, 4837, 4850, 4863, 4912, 4915,  
     4916, 4920, 4922, 4923, 4992, 4993,  
     5015, 5030, 5032, 5045, 5060, 6837,  
     6846, 6866, 6867, 6871, 6874, 6880,  
     6888, 6897, 6926, 7430, 7437, 7453,  
     7454, 7467, 7468, 7469, 7698, 7699,  
     7708, 7720, 8415, 8421, 8469, 8471,  
     8474, 8563, 8565, 8568, 8570, 8573  
 \l\_stex\_key\_number\_tl .....  
     ..... 9515, 9519, 9535, 9536  
 \l\_stex\_key\_op\_tl .....  
     ... 3970, 4160, 4166, 4218, 4219,  
     4220, 4227, 4228, 4235, 4240, 4371,  
     4405, 4409, 4467, 4472, 4489, 4494,  
     4497, 4919, 4921, 4924, 5002, 5004  
 \l\_stex\_key\_post\_tl .. 5955, 5959,  
     5976, 5987, 6020, 6033, 7053, 7061  
 \l\_stex\_key\_pre\_tl ... 5954, 5958,  
     5976, 5987, 6020, 6033, 7053, 7061  
 \l\_stex\_key\_prec\_str .. 127, 3972,  
     4159, 4165, 4269, 4272, 4275, 4281,  
     4284, 4287, 4290, 4296, 4308, 4309  
 \l\_stex\_key\_proofend\_tl .....  
     ..... 7150, 7159, 7261, 7262  
 \l\_stex\_key\_proot\_tl ..... 5956  
 \l\_stex\_key\_pts\_str .....  
     ... 8855, 8858, 8878, 8879, 8892,  
     8893, 9463, 9474, 9475, 9486, 9487  
 \l\_stex\_key\_pts\_tl 8413, 8419, 8482,  
     8483, 8489, 8584, 8601, 9398, 9421  
 \l\_stex\_key\_reorder\_str 3712, 3716,  
     3814, 3815, 4874, 4875, 5071, 5072  
 \l\_stex\_key\_return\_tl .....  
     ... 3727, 3733, 3790, 3828, 3831,  
     3924, 4811, 4842, 4855, 4889, 4892,  
     5000, 5037, 5050, 5082, 5085, 6843  
 \l\_stex\_key\_role\_str .....  
     ..... 3711, 3719, 3817, 3818,  
     3950, 4871, 4872, 5068, 5069, 6877  
 \l\_stex\_key\_root\_tl ..... 5960  
 \l\_stex\_key\_short\_tl .....  
     .. 1309, 1312, 1339, 1341, 1364, 1366  
 \l\_stex\_key\_sig\_str .....  
     ... 122, 2328, 2370, 2518, 2533, 2550  
 \l\_stex\_key\_style\_clist 134, 401,  
     403, 453, 457, 504, 508, 6911, 6912,  
     7005, 8917, 8918, 9041, 9055, 9056  
 \l\_stex\_key\_T\_bool .....  
     ... 8957, 8962, 8963, 8978, 8991,  
     9006, 9020, 9094, 9107, 9118, 9132  
 \l\_stex\_key\_term\_tl .....  
     .. 7152, 7162, 7281, 7282, 7432, 7436  
 \l\_stex\_key\_testspace\_dim .....  
     ..... 8628, 8631,  
     8633, 8662, 9176, 9178, 9239, 9242  
 \l\_stex\_key\_title\_str ..... 6849  
 \l\_stex\_key\_title\_tl .....  
     ..... 395, 397, 1595, 1773,  
     1774, 2540, 6925, 6930, 8478, 8479,  
     8485, 8578, 8579, 8682, 8683, 8743,

```

8744, 8792, 8793, 8845, 8846, 9368,
9416, 9431, 9432, 9436, 9548, 9549
\l_stex_key_Ttext_t1 . . . . .
. . . 8958, 8964, 8979, 9021, 9095, 9133
\l_stex_key_type_t1 . . . . . 125,
3725, 3737, 3758, 3789, 3822, 3823,
3920, 3932, 3936, 4529, 4841, 4854,
4883, 4884, 5076, 5077, 6841, 6850
\l_stex_key_variant_str . . . . .
. . . . . 127, 4158, 4164, 4171,
4173, 4202, 4255, 4368, 4377, 4409,
4514, 4533, 4822, 4913, 4921, 5017
\l_stex_key_wikidata_str . . . . .
. . . . . 3728, 3735, 3808, 3809
\stex_keys_define:nnnn . . . . .
. . . . . 133, 361, 361, 380,
388, 394, 400, 406, 412, 1338, 1579,
1585, 2517, 3709, 3722, 3930, 4157,
4524, 4790, 5953, 6143, 6836, 7035,
7147, 7428, 8009, 8412, 8612, 8629,
8699, 8854, 8955, 9174, 9514, 9613
\stex_keys_set:nn . . . . .
. . 133, 134, 376, 376, 1345, 1360,
1597, 1798, 1803, 2539, 3746, 3942,
3943, 4178, 4502, 4543, 4797, 4990,
5965, 5973, 5984, 6006, 6017, 6030,
6158, 6918, 6947, 7045, 7051, 7059,
7237, 7308, 7363, 7466, 7662, 7696,
8089, 8460, 8464, 8553, 8622, 8638,
8661, 8702, 8751, 8802, 8863, 8912,
8976, 9017, 9050, 9092, 9129, 9218,
9227, 9447, 9529, 9533, 9568, 9626
\stex_kpsewhich:Nn 143, 43, 43, 54, 64
\c_stex_language_abbrevs_prop . . .
. . . . . 140, 112
\stex_language_from_file: . . .
. . . . . 141, 162, 162, 1000
\c_stex_languages_clist . 29, 185, 188
\c_stex_languages_prop . . .
. . . . . 140, 112, 142, 143, 177, 193, 801
\g_stex_last_feature_str . . .
. . . . . 129, 2818, 6184
\stex_macro_body:N . 142, 539, 541, 559
\stex_macro_definition:N 142, 554, 554
\l_stex_macroname_str . . .
. . . . . 125, 228, 2808,
2809, 3744, 3748, 3750, 3784, 3805,
3806, 3944, 3955, 4544, 4798, 4836,
4848, 4865, 4866, 4991, 5031, 5043,
5062, 5063, 6169, 6876, 7697, 7707
\stex_main_archive: . . . 1194, 1233
\c_stex_main_archive_prop . 135, 1194
\c_stex_main_file . . .
. . . . . 138, 139, 970, 1009, 1074, 7774
\_stex_map_args:N . . . . .
. . . . . 3832, 4132, 4132, 4234, 4330,
4401, 4576, 4893, 5086, 5453, 6198
\_stex_map_notation_args:N . . .
. . . . . 4132, 4145, 4434
\stex_map_uri:Nnnnn 139, 807, 814, 2385
\c_stex_mathhub_file 139, 914, 1038,
1113, 1123, 1126, 1136, 1195, 1643,
1722, 1733, 1888, 1900, 1912, 1915,
1939, 1944, 1955, 2007, 2061, 2065,
2075, 2079, 2089, 2094, 3143, 3158
\c_stex_mathhub_main_manifest_-
prop . . . . . 1201, 1202
\stex_mathml_arg:nn . . . . . 142
\stex_mathml_intent:nn . . . . . 142
\_stex_maybe_brackets:nn . . .
. . . . . 4629, 4646, 5110, 5350, 5361, 5902
\stex_metagroup_do_in:nn . . .
. . . . . 123, 144, 236, 241,
248, 2579, 3391, 3425, 7704, 7705, 7716
\stex_metagroup_new:n . . .
. . . . . 144, 230, 231, 235, 2330, 2379, 2863
\l_stex_metatheory_uri 2199, 2405,
2408, 2492, 2508, 2522, 2524, 2551,
2552, 7623, 7624, 7625, 7672, 7673
\c_stex_module_. . . . . 123, 126
\stex_module_add_code:n . . .
. . . . . 123, 2582, 2582, 2585, 2587, 2597, 7680
\stex_module_add_morphism:nnn . . . 122
\stex_module_add_morphism:nnnn . . .
. . . . . 131, 2603, 2603,
2608, 2967, 3305, 6243, 6303, 7683
\stex_module_add_notation:nnnnm . . .
. . . . . 126, 2646,
2646, 2657, 2959, 2962, 4366, 4486
\stex_module_add_symbol:nnnnnnN . . .
. . . . . 122, 2609
\stex_module_add_symbol:nnnnnnnn . . .
. . . . . 123, 2609, 2981, 2985, 2988,
3783, 3954, 6164, 6196, 6879, 7706
\stex_module_setup:n . . .
. . . . . 122, 2321, 2321, 2544, 2814, 7543, 7663
\stex_module_setup_top_nosig:n . . .
. . . . . 2329, 2338, 2398, 7659
\l_stex_morphism_morphisms_seq . . .
. . . . . 130, 2896, 2906, 3001
\l_stex_morphism_renames_prop . . .
. . . . . 130, 2895, 2905,
2952, 2958, 2971, 2979, 3026, 3428
\l_stex_morphism_symbols_prop . . .
. . . . . 130, 2884,
2894, 2904, 2922, 2926, 2935, 2949,
2954, 3022, 3040, 3045, 3394, 3490
\stex_new_statement:nn . . . . . 6855

```

```

\stex_new_statement:n . . . . .
    . . . . . 6916, 6982, 6988, 7003, 7004
\stex_new_styable_cmd:nnn . . .
    . . . . . 134, 439, 439,
3077, 3290, 3339, 3365, 3409, 3435,
3616, 3636, 3657, 3745, 3941, 4177,
4501, 4542, 4796, 4989, 7235, 7629
\stex_new_styable_env:nnnnnn . .
    . . . . . 134, 439, 474, 2538, 3500,
3522, 3542, 6135, 6221, 6273, 6917,
7332, 7350, 7362, 8454, 8552, 8654,
8717, 8766, 8818, 8911, 9049, 9526
\_stex_next_symbol:n . . . . .
    . . . . . 5206, 5207, 5225, 6554, 6676, 7040
\c_stex_no_frontmatter_bool 35, 1473
\l_stex_notation_ . . . . . 126
\l_stex_notation_add: . . . . .
    . . . . . 3976, 4182, 4354, 4365, 4552, 7723
\l_stex_notation_args_tl . 4146,
4224, 4258, 4329, 4335, 4341, 4550
\l_stex_notation_check: 4181, 4354,
4354, 4506, 4551, 4813, 5012, 7722
\l_stex_notation_do_html:n . 3979,
4185, 4354, 4374, 4554, 4817, 7725
\l_stex_notation_downprec . 5569,
5896, 5897, 5907, 5908, 5928, 5930
\l_stex_notation_macrocode_cs . .
    . . . . . 126–128, 4208, 4253, 4265,
4359, 4370, 4399, 4459, 4488, 4493,
4517, 4536, 4825, 4914, 4917, 5020
\l_stex_notation_make_args: 4209,
4354, 4360, 4433, 4518, 4537, 4826
\stex_notation_parse:n . . . . .
    . . . . . 3975, 4180, 4217, 4217,
4505, 4549, 4812, 5003, 5006, 7721
\stex_notation_parse_and_then:nw
    . . . . . 127, 129
\l_stex_notation_set_default:n . .
    . . . . . 4188, 4453, 4474, 4484, 4509
\stex_notation_top:nw . . . . . 129
\stex_par: . . . . . 9042, 9043, 9045
\stex_persist:n . . . . . .
    . . . . . 122, 142, 612, 616, 617, 619, 622,
625, 626, 632, 633, 1188, 1213, 2427
\c_stex_persist_mode_bool . . . . .
    . . . . . 31, 603, 666, 1232
\l_stex_persist_read_now: . . . . .
    . . . . . 665, 1212, 7776
\c_stex_persist_write_mode_bool . .
    . . . . . 32, 609, 614, 667, 673, 1211, 2416
\c_stex_persist_write_mode_
    bool\obeyedline . . . . . 142
\stex_pseudogroup:nn . . . . . 143,
144, 210, 210, 291, 1092, 2684, 5442
\stex_pseudogroup_restore:N . . .
    . . . . . 143, 144, 210, 213, 1105, 2688
\stex_pseudogroup_with:nn . . .
    . . . . . 143, 220, 220, 815,
937, 2407, 2695, 2713, 2738, 4044,
4725, 4734, 4759, 5928, 5942, 6064
\c_stex_pwd_file . . . . .
    . . . . . 139, 970, 990, 1195, 1196, 1740, 1895
\stex_reactivate_macro:N 143, 344,
350, 2602, 3014, 3015, 3016, 3333,
3335, 3518, 3539, 3563, 3631, 3652,
3673, 3768, 3997, 4215, 4566, 6180,
6258, 6325, 6968, 6969, 6970, 6971,
6972, 6986, 6995, 6996, 6997, 6998,
6999, 7000, 7001, 7319, 7320, 7321,
7322, 7323, 7324, 7325, 7326, 7633,
7657, 7685, 8438, 8439, 8440, 8441,
8442, 8443, 8444, 8828, 8928, 9068
\stex_ref_new_doc_target:n . . .
    . . . . . 121, 133, 435, 1528, 1541, 1565
\l_stex_ref_new_id:n . 1531, 1542, 6977
\stex_ref_new_sym_target:n . . .
    . . . . . 121, 1811, 1835, 1844, 6965, 6992, 7026
\stex_ref_new_sym_target:nn 121, 1842
\stex_ref_new_symbol:n . . . . .
    . . . . . 121, 1811, 2452, 3793, 3963
\l_stex_ref_url_str . 1538, 1559, 1561
\l_stex_renamedecl_do:nn . . . . .
    . . . . . 3411, 3416, 3613
\stex_require_archive:n . . . . .
    . . . . . 131, 135, 1082, 1111, 1111,
1121, 1634, 1899, 3130, 3140, 3155
\stex_resolve_path_pair:Nnn . . .
    . . . . . 135, 1884, 1884, 1906
\l_stex_return_args:nn . . . . .
    . . . . . 3797, 3832, 4893, 5086
\l_stex_return_notation_tl . . .
    . . . . . 5246, 5427, 5428, 6392,
6400, 6562, 6563, 6643, 6669, 6670
\stex_set_current_archive:n . . .
    . . . . . 135, 1081, 1081, 1101, 1207, 3274
\stex_set_current_namespace: . . .
    . . . . . 140, 966, 966, 1001
\stex_set_document_uri: . . . . .
    . . . . . 140, 962, 962, 999
\stex_set_language:n . . . . .
    . . . . . 141, 140, 140, 161, 178, 2531
\stex_set_notation_macro:nnnn . .
    . . . . . 126, 2646, 2663, 2667, 2667, 2679
\stex_sms_allow:N . . . . . 136, 2142,
2142, 2153, 2154, 2155, 2156, 2157
\stex_sms_allow_env:n . . . . .
    . . . . . 137, 2142, 2150, 2569, 3520, 3541,
3565, 6178, 6255, 6321, 6942, 7694

```

```

\stex_sms_allow_escape:N ..... 7395, 7404, 8499, 8506, 8589, 8591,
                                8651, 8668, 8672, 8714, 8729, 8733,
                                8763, 8778, 8782, 8815, 8831, 8835,
                                8936, 8938, 9076, 9078, 9541, 9544
\stex_suppress_html:n ..... 141, 290, 290, 2175, 4248
\_stex_symdecl_check_terms: .... 125, 3775, 3917, 3917, 3953, 4804
\stex_symdecl_do: 125, 3774, 3846, 3846, 3952, 4803, 4998, 6878, 7701
\_stex_symdecl_html: ..... 3778, 3801, 3966, 6885
\stex_symdecl_top:n ..... 125, 3752, 3770, 3770, 4545
\_stex_symdef_styledefs: . 4526, 4558
\_stex_term_arg:nnn ..... 5560, 5564, 5571, 5581
\_stex_term_arg:nnnn ..... 4449, 4690, 4698,
                                5564, 5565, 5595, 5629, 5666, 5751
\_stex_term_arg_aB:nnnnn ..... 4703, 4711, 4718, 5585, 5593
\_stex_term_do_aB_clist: .. 4725,
                                4726, 4735, 4739, 4760, 4765, 5590,
                                5602, 5631, 5681, 5710, 5758, 5761
\_stex_term_oma:nnn ..... 4642, 5452, 5836, 5837, 5863
\_stex_term_oma_or_omb:nnn ..... 4642, 4647, 4695, 4708
\_stex_term_omb:nnn ..... 4695,
                                4708, 5480, 5481, 5865, 5866, 5892
\_stex_term_oms:nnn ..... 4409, 5233, 5235, 5772,
                                5782, 5831, 5835, 5967, 5977, 5988
\_stex_term_oms_or_omv:nnn ..... 4011, 4630, 5111, 5233,
                                5235, 5267, 5269, 5351, 5362, 5435,
                                5772, 5835, 6409, 6447, 6485, 6541
\_stex_term_omv:nnn ..... 4982, 5267, 5269,
                                5772, 5795, 5832, 6011, 6024, 6037
\_stex_titlefragment: 1354, 1370, 7924
\stex_undefined: ..... 41
\stex_uri_add_module:NNn ..... 140, 935, 935, 960, 1543, 7623
\stex_uri_from_current_file:Nn ..... 140, 925, 925, 963
\stex_uri_from_current_file_- nolang:Nn ..... 140, 925, 931, 967
\stex_uri_from_repo_file ..... 140
\stex_uri_from_repo_file:NNNn ..... 139, 140, 888, 891, 927, 1648
\stex_uri_from_repo_file_- nolang:NNNn ..... 140, 888, 888, 932

```

```

\stex_uri_resolve:Nn ..... 139,
    873, 873, 876, 2508, 2524, 2528, 7542
\stex_uri_set:Nn 139, 823, 869, 872, 923
\stex_uri_use:N ..... 139, 856,
    877, 881, 929, 964, 968, 1538, 1544,
    1545, 1549, 1615, 1617, 1650, 1657,
    1674, 1675, 2408, 2552, 3169, 3174,
    3175, 3188, 3193, 3194, 6909, 7673
\stex_vardecl_notation_macro: ...
    ... 126, 4507, 4814, 4910, 4910, 5013
\stex_variable:nnnnnnnN . 4789, 4930
\l_stex_variables_prop .....
    ..... 126, 4787, 4835, 4942, 5030
stex internal commands:
\stex_annotation_env_str ... 305, 306
\stex_aux_apply_patch:n .. 445, 452
\stex_aux_apply_patch_begin:n .
    ..... 480, 503
\stex_aux_apply_patch_end:n ...
    ..... 485, 518
\stex_aux_args:n . 543, 547, 550, 553
\stex_aux_end: ..... 539, 540, 543
\stex_aux_params:n 557, 588, 595, 598
\stex_aux_patch:nnn ..... 441, 466
\stex_aux_patch:nnnn ..... 476, 493
\stex_aux_prefix:n .. 555, 563, 577
\stex_aux_prefix_long:n .....
    ..... 569, 573, 579, 586
\stex_aux_split_at_bracket:w ...
    ..... 416, 424
\stex_aux_start: ..... 539, 542
\l_stex_aux_tl .....
    ..... 366, 367, 368, 370, 373, 374
\stex_debug:nn ..... 70, 73, 78
\l_stex_debug_cl ..... 86, 88, 91
\stex_debug_env_str .... 84, 85, 88
\stex_doc_check_topsect: .....
    ..... 1451, 1457, 1463, 1469
\stex_doc_do_section:n .....
    ..... 1321, 1325, 1329, 1346, 1373
\stex_doc_maketitle: ... 1264, 1269
\stex_doc_orig_backmatter .....
    ..... 1485, 1488, 1506
\stex_doc_orig_frontmatter ...
    ..... 1475, 1478, 1495
\stex_doc_set_title:n .. 1241, 1257
\stex_doc_skip_fragment:n .....
    ..... 1414, 1420,
    1424, 1428, 1429, 1430, 1431, 1432
\stex_doc_skip_section: .....
    ..... 1382, 1404, 1410
\stex_doc_skip_section_i: .....
    ..... 1391, 1394, 1397, 1405, 1410
\stex_doc_title_html: .....
    ..... 1246, 1249, 1261
\g_stex_doc_title_tl .....
    .. 1236, 1238, 1245, 1251, 1256, 1259
\l_stex_doc_titlefragment_bool .
    ..... 1352, 1362, 1372, 1377
\stex_expr_aB_arg:nnnn 5600, 5607
\stex_expr_aB_simple_arg:nnnn
    ..... 5618, 5627
\stex_expr_add_prop_arg:nnw ...
    ..... 5448, 5472, 5475
\stex_expr_arg:n ..... 5449, 5485
\l_stex_expr_arg_counter_int ...
    ..... 5439, 5446, 5461, 5497, 5498
\stex_expr_arg_do:nnn .....
    ..... 5499, 5505, 5521, 5556, 5563
\stex_expr_arg_inner:nnn .....
    ..... 5488, 5491, 5495, 5501
\stex_expr_assoc_make_seq:nnn .
    ..... 5662, 5691, 5770
\stex_expr_assoc_seq:nnnnnnn ...
    ..... 5611, 5659
\stex_expr_check:n ..... 5529
\stex_expr_check:nTF ... 5498, 5504
\stex_expr_check_b:nn .. 5453, 5478
\l_stex_expr_count_int .....
    .. 5592, 5598, 5608, 5629, 5666, 5751
\l_stex_expr_cs ... 5693, 5696, 5716
\l_stex_expr_customs_prop .....
    ..... 5444, 5456, 5457,
    5458, 5473, 5509, 5515, 5530, 5533
\l_stex_expr_customs_seq .....
    .. 5445, 5462, 5463, 5464, 5474, 5531
\stex_expr_do_aB_clist: .....
    ..... 5585, 5590, 5631, 5710
\stex_expr_do_ab_next:nnn .....
    ..... 5452, 5454, 5480, 5481
\stex_expr_do_headterm:nn .....
    ..... 5760, 5792, 5805, 5808, 5833
\stex_expr_do_ref:nNn ... 5967,
    5975, 5986, 6008, 6019, 6032, 6042
\stex_expr_do_seqmap:nnnnnnn ...
    ..... 5616, 5723
\stex_expr_end: ..... 5613, 5623
\stex_expr_gobble:nnnnnnn ...
    ..... 5613, 5623
\l_stex_expr_iarg_tl 5703, 5705, 5716
\stex_expr_invoke_custom:n ...
    ..... 5295, 5309, 5440
\stex_expr_invoke_math: 5290, 5298
\stex_expr_invoke_op_custom:n .
    ..... 5295, 5302, 5432
\stex_expr_invoke_op_notation:w ...
    ..... 5304, 5305, 5319, 5347

```

```

\__stex_expr_invoke_return: .....
..... 5388, 5391
\__stex_expr_invoke_return_-
  maybe:n ..... 5329, 5339, 5369
\__stex_expr_invoke_return_next:
..... 5376, 5385
\__stex_expr_invoke_text: 5290, 5293
\__stex_expr_is_seqmap:n .... 5652
\__stex_expr_is_seqmap:nTF ... 5615
\__stex_expr_is_varseq:n .... 5641
\__stex_expr_is_varseq:nTF 5609, 5728
\l__stex_expr_left_bracket_str ...
..... 5898, 5924, 5932, 5942, 5943
\l__stex_expr_old_seq .....
..... 5735, 5738, 5742, 5747
\l__stex_expr_reset_tl .....
..... 5206, 5209, 5213, 5214, 5220
\__stex_expr_ret_cs .....
..... 5373, 5378, 5405, 5410, 5415
\__stex_expr_return_arg:n 5375, 5381
\l__stex_expr_return_args_tl ...
.. 5370, 5382, 5387, 5396, 5412, 5417
\__stex_expr_return_notation:n ...
..... 5393, 5426
\l__stex_expr_return_this_tl ...
..... 5371, 5387,
5392, 5396, 5400, 5401, 5411, 5419
\l__stex_expr_right_bracket_str ...
..... 5899, 5925, 5937, 5942, 5944
\__stex_expr_setup:nnnnnn ...
..... 5232, 5244, 5266
\__stex_expr_varseq_in_map:nnnnnnnn ...
..... 5730, 5769
\__stex_features_add_definiens:nn ...
..... 2875, 3018
\__stex_features_break: ... 2876, 2880
\__stex_features_check_break:nnnnnnnn ...
..... 3039, 3044, 3053, 3056, 3063
\__stex_features_clean:nnw 2915, 2923
\__stex_features_do_decls: 2907, 2919
\__stex_features_do_elaboration: ...
..... 2897, 2947
\__stex_features_do_for_list: ...
..... 2866, 3017
\__stex_features_do_morph:nnnn ...
..... 2995, 2999
\__stex_features_do_morphisms:n ...
..... 2908, 2993, 3003
\__stex_features_elab_check: ...
..... 2955, 2978
\l__stex_features_feature_str ...
..... 2858, 2970
\__stex_features_get_check:nnnn ...
..... 3023, 3051
\l__stex_features_implicit_bool .
..... 2824, 2843, 2846, 2983
\__stex_features_reactivate: ...
..... 2861, 3006
\__stex_features_rename:nn 2971, 2974
\__stex_features_rename_all: .. 2911
\__stex_features_renamed_-
  check:nnnn ..... 3027, 3037
\__stex_features_set_definiens_-
  macros: ..... 2876, 2880
\__stex_features_set_definiens_-
  macros_i:nnnnnnn ..... 2883, 2887
\__stex_features_setup: ... 2860, 2903
\__stex_features_split_qm:w ...
..... 2946, 2964
\l__stex_features_tmp ...
..... 2958, 2960, 2979, 2980, 2981
\__stex_features_total_check: ...
..... 2936, 2940
\__stex_groups_do: .... 261, 268, 277
\__stex_groups_do_in:nn 243, 250, 271
\__stex_groups_exists:n .... 236
\__stex_groups_exists:nTF .... 242
\l__stex_groups_ids_seq 230, 233, 269
\__stex_groups_tmp .... 252, 259, 265
\l__stex_importmodule_archive_-
  str 3136, 3141, 3151, 3156, 3273, 3274
\__stex_importmodule_check_-
  file:nn ..... 3217,
3218, 3219, 3220, 3221, 3222, 3244,
3245, 3246, 3247, 3248, 3249, 3283
\__stex_importmodule_get_from_-
  file:nnn ..... 3177, 3213
\__stex_importmodule_get_from_-
  file_safe:nnn ..... 3196, 3240
\__stex_importmodule_get_-
  module:nnn ..... 3138, 3144, 3202
\__stex_importmodule_get_module_-
  safe:nnn ..... 3153, 3159, 3208
\__stex_importmodule_get_module_-
  uri:nnn ..... 3163, 3204
\__stex_importmodule_get_module_-
  uri_safe:nnn ..... 3182, 3210
\__stex_importmodule_import_-
  module:nn ..... 3291, 3296, 3336
\__stex_importmodule_import_-
  module_presms:nn ..... 3322, 3336
\__stex_importmodule_load_file:n ...
..... 3234, 3237, 3261, 3266, 3271
\l__stex_importmodule_path_seq ...
..... 3113, 3114, 3116
\__stex_importmodule_seq ...
..... 3098, 3099, 3100

```

```

\l__stex_importmodule_seq . . .
    .. 3120, 3122, 3214, 3215, 3241, 3242
\l__stex_importmodule_str . . .
    .. 3143, 3144, 3158,
    3159, 3215, 3231, 3242, 3257, 3260,
    3265, 3272, 3276, 3284, 3285, 3287
\l__stex_inputs_bibresource:n . .
    .. 1972, 1984, 1986
\l__stex_inputs_gin_repo_str . .
    .. 2058, 2088, 2092, 2097
\l__stex_inputs_id_seq . . .
    .. 2008, 2010, 2017
\l__stex_inputs_id_str . . .
    .. 2003, 2008
\l__stex_inputs_inputref:nn 1960, 1970
\l__stex_inputs_inputref_html:nn .
    .. 1925, 1963
\l__stex_inputs_inputref_pdf:nn .
    .. 1949, 1964
\l__stex_inputs_libinput:n . .
    .. 2029, 2042, 2044
\l__stex_inputs_libinput_files_-
    seq . . .
        .. 1974,
        1977, 2006, 2013, 2014, 2023, 2024,
        2031, 2034, 2049, 2050, 2121, 2122
\l__stex_inputs_mhinput:nn 1909, 1922
\l__stex_inputs_path_seq . .
    .. 2007, 2011, 2018, 2021
\l__stex_inputs_path_str . .
    .. 2011, 2012, 2013, 2014,
    2017, 2018, 2021, 2022, 2023, 2024
\l__stex_inputs_tmp_str . .
    .. 2050, 2052
\l__stex_inputs_up_archive:nn . .
    .. 1973, 1999, 2030, 2048, 2120
\l__stex_inputs_usetikzlibrary:n .
    .. 2119, 2133, 2135
\l__stex_inputs_usetikzlibrary_-
    i:nn . . .
        .. 2103, 2122
\l__stex_iterate_continue_bool . .
    .. 2763, 2766, 2779, 2799
\l__stex_iterate_it_decl_check:nnnn .
    .. 2724, 2731
\l__stex_iterate_it_decl_i:n . .
    .. 2716, 2720, 2733
\l__stex_iterate_it_not_check:nnnn .
    .. 2752, 2756
\l__stex_iterate_it_not_i:n . .
    .. 2741, 2745, 2758
\l__stex_iterate_iterate_morphism:nn .
    .. 2769, 2773, 2782, 2785
\l__stex_iterate_mods_seq . .
    .. 2712, 2721, 2722,
    2737, 2746, 2747, 2762, 2787, 2788
\l__stex_iterate_morphism_cs:nnnn .
    .. 2764, 2790
\l__stex_iterate_not_cs:nnnnn . .
    .. 2738, 2739, 2749
\l__stex_iterate_sym_cs:nnnnnnnnN .
    .. 2695, 2696, 2706, 2713, 2714, 2727
\l__stex_iterate_todo_tl . .
    .. 2768, 2772, 2786, 2799
\l__stex_lang_lang_str . 143, 146, 153
\l__stex_lang_str .. 175, 176, 177, 178
\l__stex_lang_turkish_bool . .
    .. 186, 191, 201
\l__stex_mathhub_bool . . .
    .. 1134,
    1135, 1137, 1140, 1150, 1152, 1161
\l__stex_mathhub_check_manifest: . .
    .. 1139, 1148
\l__stex_mathhub_check_manifest:n .
    .. 1149, 1151, 1153, 1158
\l__stex_mathhub_cs . . .
    .. 1089, 1090,
    1093, 1096, 1098, 1102, 1106, 1107
\l__stex_mathhub_do_manifest:n . .
    .. 1117, 1122
\l__stex_mathhub_find_manifest:n .
    .. 1123, 1131, 1146, 1196
\l__stex_mathhub_key 1173, 1174, 1176
\c__stex_mathhub_manifest_ior . .
    .. 1166, 1168, 1170, 1185
\l__stex_mathhub_manifest_str . .
    .. 1124, 1132, 1162, 1168, 1197
\l__stex_mathhub_parse_manifest:n .
    .. 1128, 1167, 1200
\l__stex_mathhub_prop . .
    .. 1169, 1177, 1178,
    1179, 1180, 1181, 1186, 1187, 1190
\l__stex_mathhub_seq . .
    .. 1133, 1136, 1141,
    1159, 1160, 1162, 1172, 1173, 1175
\l__stex_mathhub_str . .
    .. 1040, 1042, 1044, 1047, 1048, 1052,
    1053, 1058, 1064, 1067, 1071, 1074,
    1204, 1205, 1207, 1214, 1218, 1221
\l__stex_mathhub_t1 . .
    .. 1141
\l__stex_mathhub_val . .
    .. 1175, 1177, 1178, 1179, 1180, 1181
\l__stex_module_setup_end: 2376, 2397
\l__stex_module_setup_get_uri_-
    str:n . .
        .. 2326, 2383
\l__stex_module_setup_load_meta: . .
    .. 2335, 2404
\l__stex_module_setup_load_sig: . .
    .. 2356, 2366
\l__stex_module_setup_ns_str . .
    .. 2327, 2329, 2384, 2386, 2393
\l__stex_module_setup_seg . .
    .. 2389, 2390, 2391

```

```

\l__stex_module_setup_seq . . . .
    ..... 2388, 2389, 2391, 2393
\__stex_module_setup_setup_-
    nested:n ..... 2322, 2374
\__stex_module_setup_setup_top:n
    ..... 2322, 2325
\__stex_module_setup_setup_top_-
    sig:n ..... 2329, 2351
\l__stex_module_setup_sigfile ...
    ..... 2367, 2368, 2370
\__stex_module_setup_split_-
    module:n ..... 2376, 2397
\__stex_modules_activate_-
    i:nnnnnnnn ..... 2630, 2634
\__stex_modules_activate_not:nn .
    ..... 2654, 2660
\__stex_modules_activate_sym:n ..
    ..... 2622, 2627
\__stex_modules_export:n . 2593, 2595
\__stex_modules_persist_module: .
    ..... 2416, 2426, 7537
\__stex_modules_persist_nots_-
    i:nn ..... 2442, 2463
\l__stex_modules_restore_mod_str
    ..... 2456, 2484
\__stex_modules_restore_module:nnnn
    ..... 2428, 2447
\__stex_modules_restore_nots:n ..
    ..... 2460, 2469
\__stex_modules_restore_nots_i:n
    ..... 2470, 2473, 2490
\__stex_modules_restore_nots_-
    ii:nnnn ..... 2477, 2481
\__stex_modules_set_metatheory:nn
    ..... 2493, 2513
\__stex_modules_tl ..... 2449, 2450
\l__stex_modules_tl ..... 2482, 2487
\l__stex_morphisms_ass_tl . . .
    ..... 3569, 3585, 3589, 3599, 3600
\__stex_morphisms_do_morph_-
    assign:nnn ..... 3442, 3445, 3483
\__stex_morphisms_do_parsed_-
    assign: ..... 3591, 3594
\__stex_morphisms_do_parsed_-
    newname: ..... 3597, 3604
\__stex_morphisms_do_parsed_-
    newname:w ..... 3606, 3608, 3612
\__stex_morphisms_end: ... 3597, 3612
\l__stex_morphisms_morphism_dom_-
    str ... 3436, 3449, 3461, 3471, 3485
\l__stex_morphisms_name_str . . .
    ..... 3567, 3576,
    3577, 3578, 3582, 3583, 3584, 3595
\l__stex_morphisms_newname_str . . .
    ..... 3568, 3587, 3588, 3596, 3597
\l__stex_morphisms_next_tl . . .
    ..... 3574, 3579, 3581
\__stex_morphisms_parse_assign:n
    ..... 3566, 3619, 3639, 3660
\l__stex_morphisms_seq . . .
    ..... 3571, 3572, 3574, 3576,
    3579, 3581, 3583, 3585, 3587, 3589
\__stex_notations_add:nnnnn . . .
    ..... 4676, 4681, 4685
\__stex_notations_add_last:nnnnn
    ..... 4669, 4680
\__stex_notations_add_missing_-
    args:nn ..... 4234, 4349
\__stex_notations_add_next:nnnnnn
    ..... 4671, 4675
\l__stex_notations_after_tl . . .
    ..... 4655, 4682
\__stex_notations_args_end: . . .
    ..... 4135, 4138, 4141,
    4148, 4151, 4154, 4664, 4667, 4677
\l__stex_notations_args_tl . . .
    ..... 4571, 4575, 4588
\__stex_notations_augment_arg:nn
    ..... 4576, 4593
\__stex_notations_check_aB_-
    arg:Nn ..... 4716, 4724, 4733, 4758
\l__stex_notations_clist_count_-
    int ..... 4756, 4764, 4770
\l__stex_notations_code_tl . . .
    ... 4612, 4627, 4635, 4643, 4657,
    4658, 4661, 4689, 4697, 4702, 4710
\__stex_notations_complex:nnnnnn
    ..... 4621, 4639
\__stex_notations_const_precs: . . .
    ..... 4226, 4268
\l__stex_notations_cs . . .
    ..... 4628, 4633, 4644, 4653
\__stex_notations_do_argname:nn .
    ..... 4330, 4333
\__stex_notations_do_argnames: . . .
    ..... 4304, 4328
\__stex_notations_fun_precs: . . .
    ..... 4231, 4280
\__stex_notations_make_arg:nnnn .
    ..... 4434, 4438
\__stex_notations_make_arg_-
    html:nn ..... 4401, 4416
\__stex_notations_make_name: . . .
    ..... 4573, 4597, 4605
\__stex_notations_map_args_i:w . . .
    ..... 4134, 4138, 4141

```

```

\__stex_notations_map_args_ii:w . . . . . 4147, 4151, 4154
\__stex_notations_map_cs: . . . . . 4387, 4389, 4736, 4738, 4746, 4761, 4763, 4774
\l__stex_notations_missing_str . . . . . 4232, 4350
\l__stex_notations_missing_t1 . . . . . 4233, 4262, 4351
\l__stex_notations_name_str . . . . . 4599, 4600, 4601, 4606
\l__stex_notations_opprec_t1 4256, 4270, 4273, 4275, 4282, 4285, 4287, 4291, 4298, 4311, 4317, 4323, 4378
\__stex_notations_parse_notation_- args:nnnnw . . . . . 4664, 4667, 4677
\__stex_notations_parse_precs: . . . . . 4302, 4307
\l__stex_notations_pre_t1 . . . . . 4642, 4657, 4694, 4707
\l__stex_notations_precs_seq . . . . . 4223, 4293, 4300, 4321, 4323, 4336, 4342, 4379
\__stex_notations_process_- notation:nnnnnn . . . . . 4611, 4617
\l__stex_notations_seq . . . . . 4309, 4310, 4312, 4313, 4320
\__stex_notations_simple:nnnnn . . . . . 4619, 4625
\l__stex_notations_str . . . . . 4310, 4311, 4312, 4314, 4320, 4321
\__stex_notations_styledefs: . . . . . 4194, 4201
\__stex_path_: . . . . . 684, 694
\l__stex_path_a_seq . . . . . 765, 771, 777
\l__stex_path_a_t1 . . . . . 777, 779
\__stex_path_auth:n . . . . . 139, 807, 810, 811, 812, 815, 816, 861, 877, 882, 936, 938
\l__stex_path_auth_str . . . . . 829, 837, 842, 847
\l__stex_path_b_seq . . . . . 766, 772, 778, 784
\l__stex_path_b_t1 . . . . . 778, 779
\__stex_path_canonicalize:N . . . . . 697, 708, 714
\__stex_path_colonslash . . . . . 823, 828, 833
\l__stex_path_do_hooks_pre_t1 . . . . . 1018, 1019, 1031, 1034
\__stex_path_dodots:n . . . . . 719, 723, 729
\l__stex_path_file . . . . . 896, 902, 908, 913
\__stex_path_from_repo_file:NNNNn . . . . . 889, 892, 895
\l__stex_path_maybe.win_str . . . . . 746, 747, 748
\l__stex_path_mod . . . . . 840, 843, 848
\__stex_path_module:n . . . . . 139, 811, 815, 818, 863, 877, 938, 940, 942
\l__stex_path_name . . . . . 845, 848
\__stex_path_name:n . . . . . 139, 812, 815, 819, 864, 877, 938, 947, 949
\l__stex_path_path . . . . . 834, 835, 838, 843, 848, 917, 918
\__stex_path_path:n . . . . . 139, 810, 815, 817, 862, 877, 938
\__stex_path_relativize:N . . . . . 905, 912
\l__stex_path_return_t1 . . . . . 767, 773, 780, 783, 785
\l__stex_path_seq . . . . . 717, 720, 725, 733, 734, 737, 790, 791, 792, 797, 798, 800, 802, 805, 913, 915, 919, 921, 923
\g__stex_path_stack . . . . . 980, 994, 1005, 1006, 1008, 1011
\l__stex_path_str . . . . . 685, 686, 687, 690, 692, 693, 694, 696, 699, 706, 707, 716, 718, 722, 734, 789, 790, 791, 796, 797, 798, 800, 801, 802, 971, 973, 975, 1006, 1011, 1012
\l__stex_path_t1 . . . . . 915, 919, 921
\l__stex_path_uri . . . . . 898, 899, 900, 905, 923
\__stex_path_uri_set:NnN . . . . . 824, 870, 874
\__stex_path_uri_set:Nnnnn . . . . . 837, 842, 847, 859, 867, 901, 907
\__stex_path_uri_use:w . . . . . 877, 883
\l__stex_path_win_drive . . . . . 686, 691, 698, 700
\__stex_path_win_take:w . . . . . 684, 694
\__stex_persist_env_str . . . . . 600, 601, 602, 606, 607, 608
\__stex_persist_load_file:n . . . . . 629, 661, 670
\__stex_persist_read_and_write: . . . . . 646, 668
\c__stex_persist_sms_iow . . . . . 612, 620, 642, 643, 658
\__stex_persist_write_only: . . . . . 641, 655, 662, 673
\__stex_proof_add_counter: . . . . . 7210, 7391
\__stex_proof_begin_proof:nn . . . . . 7305, 7333, 7351
\l__stex_proof_counter_intarray . . . . . 7168, 7173, 7176, 7185, 7188, 7197, 7205, 7206, 7214, 7219, 7226, 7232, 7306, 7307
\__stex_proof_end_list: . . . . . 7275, 7346, 7407, 7417, 7483, 7506
\__stex_proof_html: . . . . . 7279, 7301, 7457

```

```

\__stex_proof_html_env:n .....
    ..... 7290, 7311, 7366
\l__stex_proof_in_spfblock_bool .
    ... 7304, 7334, 7345, 7352, 7380,
    7383, 7402, 7405, 7407, 7412, 7415,
    7423, 7465, 7479, 7501, 7521, 7527
\__stex_proof_inblock_restore: ..
    ..... 7409, 7422, 7523
\__stex_proof_inc_counter: .....
    ..... 7193, 7405, 7496, 7498
\l__stex_proof_inc_counter_bool 7167
\__stex_proof_insert_number: ...
    ..... 7169, 7389, 7496, 7498
\__stex_proof_make_step_macro:Nnnnn
    ..... 7463, 7496, 7497, 7498
\__stex_proof_number_as_string:N
    ..... 7180, 7376, 7385, 7474
\__stex_proof_proof_box_tl 7150, 7254
\__stex_proof_remove_counter: ...
    ..... 7222, 7416
\__stex_proof_start_list:n .....
    ..... 7268, 7334, 7389, 7483, 7506
\__stex_proof_step_html:nn 7445,
    7481, 7483, 7489, 7504, 7506, 7510
\__stex_refs_add_doc_ref:nn .....
    ..... 1545, 1567, 1578
\l__stex_refs_default_archive_-
    str ..... 1587, 1593, 1602
\l__stex_refs_default_file_str ..
    ..... 1588, 1594, 1603
\l__stex_refs_default_title_tl ..
    ..... 1589, 1595, 1604
\__stex_refs_do_autoref:n .....
    ..... 1663, 1675, 1686,
    1690, 1701, 1715, 1741, 1752, 1760
\__stex_refs_do_internal_link:nn
    ..... 1855, 1867, 1873
\__stex_refs_do_return:nnnn .....
    ..... 1769, 1783, 1791
\__stex_refs_do_sref:nn .. 1671, 1800
\__stex_refs_do_sref_in:n .....
    ..... 1679, 1694, 1705, 1711, 1809
\__stex_refs_do_url_link:nn .....
    ..... 1861, 1875
\l__stex_refs_file .....
    .. 1629, 1631, 1647, 1649, 1737, 1738
\l__stex_refs_file_str .....
    1721, 1727, 1732, 1737, 1738, 1739,
    1740, 1745, 1749, 1751, 1759, 1771
\g__stex_refs_files_seq .....
    ..... 1528, 1568, 1573, 1620, 1682
\__stex_refs_find_uri:n .....
    ..... 1607, 1799, 1804
\__stex_refs_find_uri_in_-
    file:nnn ..... 1616, 1621, 1653
\__stex_refs_find_uri_in_prop_-
    file:N ..... 1627, 1636, 1641
\l__stex_refs_id_str .....
    ..... 1747, 1782, 1788, 1789
\c__stex_refs_iow .....
    ..... 1519, 1520, 1521, 1547
\__stex_refs_new_symbol:n 1813, 1821
\l__stex_refs_prop ..... 1635, 1636
\__stex_refs_restore_target:nnnn
    ..... 1748, 1780
\l__stex_refs_return_tl .....
    ..... 1746, 1750, 1756, 1770
\__stex_refs_set_keys_b:n .....
    ..... 1591, 1678, 1693, 1704, 1805
\l__stex_refs_str 1534, 1536, 1538,
    1543, 1545, 1550, 1823, 1825, 6978
\__stex_refs_sym_aux:nn .....
    ..... 1852, 1865, 1870, 1881
\l__stex_refs_unnamed_counter_-
    int ..... 1529, 1533, 1534
\l__stex_refs_uri .....
    ..... 1543, 1544, 1648, 1650
\l__stex_refs_uri_str .....
    ... 1608, 1619, 1631, 1642, 1647,
    1650, 1657, 1672, 1682, 1683, 1684,
    1685, 1686, 1689, 1690, 1692, 1698,
    1700, 1701, 1703, 1713, 1714, 1715,
    1742, 1753, 1761, 1781, 1786, 1787
\__stex_seqs_add: ..... 5010, 5028
\l__stex_seqs_args_tl ... 5086, 5088
\__stex_seqs_check_terms: 5009, 5056
\l__stex_seqs_clist .....
    ..... 5123, 5130, 5137, 5141
\l__stex_seqs_cs .....
    ..... 5084, 5088, 5144, 5148,
    5156, 5159, 5168, 5175, 5188, 5189
\__stex_seqs_do_all:w ... 5182, 5192
\__stex_seqs_do_first: ... 5105, 5166
\__stex_seqs_do_first_arg:n .....
    ..... 5164, 5171
\__stex_seqs_do_first_next: .....
    ..... 5173, 5178
\__stex_seqs_do_one:w ... 5180, 5186
\__stex_seqs_do_op:w ... 5104, 5108
\__stex_seqs_doop_arg:n .. 5124, 5135
\__stex_seqs_doop_range:w 5117, 5121
\l__stex_seqs_first_args_tl .....
    ..... 5170, 5188, 5189, 5193, 5194
\__stex_seqs_get_index_notation:n
    ..... 5116, 5154, 5187
\__stex_seqs_html: ..... 5008, 5058
\__stex_seqs_macro: ..... 5011, 5042

```

```

\__stex_seqs_make_args: .. 5021, 5055
\__stex_seqs_range_clist .....
..... 5001, 5036, 5049
\g__stex_smsmode_allowed_escape_-
tl ..... 2140, 2147, 2276
\g__stex_smsmode_allowed_import_-
env_seq 2164, 2170, 2210, 2259, 2262
\g__stex_smsmode_allowed_import_-
tl ..... 2163, 2166, 2207, 2254
\g__stex_smsmode_allowed_tl .....
..... 2139, 2143, 2272
\g__stex_smsmode_allowedenvs_seq
..... 2141, 2151, 2281, 2284
\g__stex_smsmode_bool .....
..... 2158, 2159, 2161, 2177
\__stex_smsmode_check_begin:Nn ..
..... 2259, 2281, 2293
\__stex_smsmode_check_cs:NNn ...
..... 2232, 2240
\__stex_smsmode_check_end:Nn ...
..... 2262, 2284, 2302
\__stex_smsmode_do:w .....
..... 2226, 2238, 2240, 2242,
2264, 2274, 2286, 2299, 2306, 2309
\__stex_smsmode_do_aux:N . 2240, 2246
\__stex_smsmode_do_aux_curr:N ...
..... 2206, 2217, 2248
\__stex_smsmode_do_aux_imports:N
..... 2206, 2252
\__stex_smsmode_do_aux_normal:N .
..... 2217, 2270
\__stex_smsmode_importmodules_-
seq ..... 2195
\__stex_smsmode_in_smsmode:n ...
..... 2175, 2205, 2215, 2216
\__stex_smsmode_sigmodules_seq 2196
\__stex_smsmode_smsmode_do: ...
..... 2188, 2224
\__stex_smsmode_start_smsmode:n .
..... 2186, 2213, 2218
\__stex_statements_do_defref:nn .
..... 7013, 7046, 7052, 7060
\__stex_statements_force_id: ...
..... 6872, 6975, 6983, 7006
\__stex_statements_html_keyvals:nn
..... 6902, 6923, 6951
\__stex_statements_setup:nn ...
.. 6864, 6984, 6989, 7003, 7007, 7010
\__stex_statements_setup_def: ...
..... 6962, 6985, 7008
\__stex_statements_uri_str ...
6887, 6891, 6892, 6897, 6898, 7072,
7075, 7078, 7080, 7125, 7128, 7131
\__stex_structures_assigned_seq
..... 6550, 6661, 6700, 6716
\__stex_structures_begin:nn .....
..... 6136, 6157, 6237, 6296
\__stex_structures_check_-
def:nnnnnnn .. 6315, 6328
\__stex_structures_clist .....
..... 6524, 6545, 6551, 6553
\__stex_structures_comp_cs .....
..... 6593, 6598
\__stex_structures_cs .....
..... 6526, 6532, 6539
\__stex_structures_current_type:
..... 6480, 6611, 6682
\__stex_structures_current_-
type_tl .... 6407, 6443, 6483, 6488
\__stex_structures_do_assign:nn .
..... 6427, 6431
\__stex_structures_do_assign_-
list:n ..... 6403, 6424
\__stex_structures_do_decl:nnnnnnnn
..... 6738, 6758
\__stex_structures_do_decl_-
nomacro:nnnnnnnn .. 6736, 6744
\__stex_structures_do_exernals:
..... 6140, 6182, 6252, 6318
\__stex_structures_end: .....
..... 6371, 6375, 6391, 6396
\__stex_structures_exstruct_-
name_str ..... 6291, 6296, 6340
\__stex_structures_extend_-
structure:nn ..... 6264, 6291
\__stex_structures_extend_-
structure_i:Nnnnnnnnn . 6261, 6269
\__stex_structures_external_-
decl:nnnn ..... 6185, 6189
\__stex_structures_extmod_str ..
..... 6262, 6266
\__stex_structures_extname_-
count ..... 6334, 6338, 6340
\__stex_structures_field_name_-
str ..... 6648, 6652, 6653, 6684
\__stex_structures_fields_clist
..... 6404,
6425, 6432, 6450, 6453, 6707, 6708
\__stex_structures_get_field_-
name:n ..... 6647, 6659
\__stex_structures_imports_seq .
..... 6222, 6227, 6238,
6275, 6280, 6298, 6820, 6823, 6826
\__stex_structures_invocation_-
type:n ..... 6355, 6402
\__stex_structures_invoke_-
field:n ..... 6625, 6657

```

```

\__stex_structures_invoke_maybe_-
    field:nn ..... 6614, 6618
\__stex_structures_invoke_this:n
    ..... 6364, 6606
\__stex_structures_invoke_top:n .
    ..... 6345,
    6348, 6372, 6376, 6379, 6382, 6384
\__stex_structures_make_mod:n ...
    ..... 6449, 6461
\__stex_structures_make_oml:n ...
    ..... 6453, 6467
\__stex_structures_make_oml:nn ..
    ..... 6468, 6470
\__stex_structures_make_prop: ...
    ..... 6492, 6619, 6697
\__stex_structures_make_prop_-
    assign: ..... 6493, 6622, 6706
\__stex_structures_make_prop_-
    assign:nn ..... 6709, 6714
\__stex_structures_make_prop_-
    assign_replace:nnnn ... 6717, 6723
\__stex_structures_make_type:n ..
    ..... 6412, 6417, 6419, 6435
\__stex_structures_maybe_-
    notation:w ..... 6359, 6361, 6487
\__stex_structures_merge:nw ...
    ..... 6353, 6369
\__stex_structures_more_-
    nextsymbol_tl ... 6660, 6662, 6687
\__stex_structures_name_str 6128,
    6145, 6150, 6152, 6159, 6160, 6165,
    6166, 6171, 6172, 6175, 6191, 6197
\__stex_structures_new_extstruct_-
    name: ..... 6274, 6336
\__stex_structures_present: ...
    ..... 6498, 6623
\__stex_structures_present:nn ...
    ..... 6502, 6508, 6513, 6520, 6523
\__stex_structures_present_-
    entry:nn ..... 6528, 6534, 6549
\__stex_structures_present_i:w ..
    ..... 6494, 6500, 6506
\__stex_structures_present_ii:nw
    ..... 6511, 6519
\__stex_structures_prop .....
    6538, 6650, 6658, 6690, 6698, 6715,
    6718, 6724, 6745, 6747, 6759, 6761
\__stex_structures_prop_do_-
    decls: ..... 6702, 6733
\__stex_structures_prop_do_-
    notations: ..... 6703, 6782
\__stex_structures_redo_tl ...
    .. 6677, 6701, 6727, 6774, 6785, 6800
\__stex_structures_replace_-
    this_t1 ..... 6183
\__stex_structures_seq .....
    ..... 6699, 6746, 6760, 6784
\__stex_structures_set_comp_t1 .
    ..... 6344, 6393, 6397, 6555, 6672
\__stex_structures_set_custom_-
    comp:n ..... 6398, 6587
\__stex_structures_set_customcomp:
    ..... 6371, 6396
\__stex_structures_set_this:n ...
    ..... 6620, 6629
\__stex_structures_set_thiscomp:
    ..... 6344, 6581
\__stex_structures_set_thisnotation:
    ..... 6375, 6391
\__stex_structures_shift_-
    argls:nn ..... 6198, 6203
\__stex_structures_this_t1 .....
    6356, 6556, 6557, 6560, 6575, 6632,
    6635, 6642, 6663, 6664, 6667, 6681
\__stex_symdecl_add_decl: 3776, 3782
\__stex_symdecl_args_t1 . 3832, 3834
\__stex_symdecl_cs .....
    .. 3830, 3834, 4025, 4027, 4029, 4055
\__stex_symdecl_do_args: . 3849, 3908
\__stex_symdecl_env_str .....
    ..... 3682, 3683, 3684
\__stex_symdecl_get_from_one_-
    string:n ..... 4064, 4109
\__stex_symdecl_get_symbol_from_-
    cs: ..... 4031, 4042
\__stex_symdecl_get_symbol_from_-
    modules:nn ..... 4066, 4098
\__stex_symdecl_get_symbol_from_-
    string:n ... 4032, 4034, 4037, 4059
\__stex_symdecl_name ... 4062, 4068
\__stex_symdecl_parse_arity: ...
    ..... 3848, 3854
\__stex_symdecl_seq .....
    ..... 4061, 4062, 4063, 4067
\__stex_symdecl_set_textsymdecl_-
    macro:nnn ..... 3982, 4000
\__stex_symdecl_sym_from_str_-
    i:nnnn ..... 4072, 4103
\__stex_symdecl_sym_i_finish:nnnnnnn
    ..... 4078, 4085
\__stex_symdecl_sym_i_gobble:nnnnn
    ..... 4080, 4083
\__stex_vars_add: ..... 4805, 4833
\__stex_vars_args_t1 ... 4893, 4895
\__stex_vars_bind_bool .....
    ..... 4788, 4791, 4793, 4877

```

```

\__stex_vars_check_var:nnnnnnnnN ..... 4943, 4947
\l__stex_vars_cs ..... 4891, 4895
\__stex_vars_get_var:n ..... 4941, 4959, 4967
\__stex_vars_html: ..... 4807, 4860
\__stex_vars_macro: ..... 4806, 4847
\__stex_vars_set_vars:nnnnnnN ..... 4928, 4949, 4952
\sTeX/ComputerScience/Software ..... 15
stex_annotation_env (env.) ..... 142
\stexcommentfont ..... 7265, 7335, 7402, 7424, 7487, 7508
\stexdoctitle ..... 1237, 1296, 1308, 1369, 2542, 8102, 8273, 8479, 8579
\STEXexport ..... 49, 86, 87, 121, 2591
\stexhtmlfalse ..... 329
\stexhtmltrue ..... 326
\STEXInternalAssocArgMarkerI ..... 128
\STEXInternalAssocArgMarkerII ..... 128
\STEXInternalNotation .. 4254, 4588, 4608
\STEXInternalSetSrefSymURL ..... 1524, 1825, 1828, 1831
\STEXInternalSrefRestoreTarget ..... 1522, 1548, 1748
\STEXInternalSymbolAfterInvocationTL ..... 128, 132
\STEXInternalTermMathArgiii ..... 128
\STEXInternalTermMathAssocArgiiii ..... 128
\STEXInternalTermMathOMAiii ..... 128
\STEXInternalTermMathOMBiii ..... 128
\STEXInternalTermMathOMSOorOMViii .. 128
\STEXinvisible .. 81, 4351, 5226, 7315, 7370
\STEXRestoreNotsEnd ..... 2443, 2467, 2474
\stexstyle ..... 134
\stexstyleassertion ..... 104
\stexstyleassign ..... 104
\stexstyleassignMorphism ..... 104
\stexstylecopymod ..... 104
\stexstylecopymodule ..... 104
\stexstyledefinition ..... 104
\stexstyleexample ..... 104
\stexstyleextstructure ..... 104
\stexstylegnote ..... 9429
\stexstyleimportmodule ..... 104, 134
\stexstyleinterpretmod ..... 104
\stexstyleinterpretmodule ..... 104
\stexstylemathstructure ..... 104
\stexstylemcb ..... 8943
\stexstyleMMTinclude ..... 104
\stexstylemodule ..... 104, 135
\stexstylenotation ..... 104
\stexstyleparagraph ..... 104
\stexstyleproblem ..... 9346
\stexstyleproof ..... 104
\stexstylerealization ..... 104
\stexstylerealize ..... 104
\stexstylerenamedecl ..... 104
\stexstylerequiremodule ..... 104
\stexstylescb ..... 9083
\stexstylepsfsketch ..... 104
\stexstylesubproblem ..... 9388
\stexstylesubproof ..... 104
\stexstylesymdecl ..... 104
\stexstylesymdef ..... 104
\stexstyletextsymdecl ..... 104
\stexstyleusemodule ..... 104
\stexstylevardef ..... 104
\stexstylevarnotation ..... 104
\stexstylevarseq ..... 104
\stopsolutions ..... 111, 8692
str commands:
\c_backslash_str ..... 107, 692
\c_colon_str ..... 823, 878, 1172, 2386
\c_dollar_str ..... 4343
\c_hash_str ..... 590, 2648, 4350
\c_left_brace_str ..... 9367, 9415, 9435, 9460
\c_percent_str ..... 54, 55, 1040
\c_right_brace_str ..... 9384, 9425, 9441, 9464
\str_case:Nn ..... 1176
\str_case:nn ..... 4687, 5479
\str_case:nnTF .. 1435, 3858, 4417, 4439, 5507, 5535, 7737, 7758, 7876
\str_clear:N ..... 56, 381, 382, 389, 407, 691, 898, 1058, 1132, 1608, 1926, 2092, 2202, 2384, 2518, 2826, 3021, 3109, 3136, 3151, 3436, 3567, 3568, 3710, 3711, 3712, 3713, 3723, 3724, 3728, 3748, 3931, 3971, 3972, 4022, 4023, 4158, 4159, 4160, 4161, 4931, 4958, 4966, 5567, 6145, 6213, 6837, 6838, 6840, 6887, 7069, 7122, 7181, 7430, 7661, 7732, 7753, 8010, 8312, 8313, 8415, 8416, 8613, 8630, 8855
\str_const:Nn ..... 109, 7795
\str_count:n ..... 531, 536
\str_gput_right:Nn ..... 9309
\str_gset:Nn ..... 1525, 1838, 1846
\str_gset_eq:NN ..... 1053, 2892, 2893
\str_if_empty:NTF ..... 85, 428, 434, 601, 607, 698, 718, 905, 1046, 1048, 1067, 1124, 1197, 1613, 1619, 1625, 1672, 1673, 1688, 1699, 1719, 1742, 1753, 1761, 1781, 1806, 1872, 2328, 2571, 2808, 2831, 3025, 3029, 3111, 3273,

```

3449, 3582, 3596, 3683, 3771, 3805,  
 3808, 3811, 3814, 3817, 3945, 4016,  
 4170, 4269, 4281, 4290, 4337, 4799,  
 4865, 4868, 4871, 4874, 4960, 4968,  
 4992, 4995, 5062, 5065, 5068, 5071,  
 6066, 6149, 6159, 6208, 6215, 6652,  
 6865, 6866, 6871, 6888, 6908, 6976,  
 7075, 7128, 7375, 7384, 7453, 7467,  
 7473, 7698, 7734, 7755, 7866, 7930,  
 8093, 8469, 8563, 8568, 8639, 8683,  
 8703, 8752, 8803, 8845, 8864, 8878,  
 8892, 9431, 9448, 9474, 9486, 9489  
`\str_if_empty:nTF` . . . . . 467, 494,  
 678, 730, 825, 1532, 1886, 2521, 3422  
`\str_if_eq:NNTF` . . . . . 176, 779  
`\str_if_eq:nnTF` . . . . .  
 . . . . . 55, 150, 166, 167, 168, 190,  
 306, 530, 535, 565, 581, 590, 602,  
 608, 693, 731, 732, 748, 1656, 1740,  
 1782, 1787, 1789, 1843, 2308, 2390,  
 2629, 2662, 2767, 3038, 3043, 3052,  
 3055, 3231, 3257, 3441, 3457, 3684,  
 4272, 4284, 4296, 4948, 4951, 5142,  
 7092, 7934, 7937, 8002, 8161, 8949  
`\str_if_eq_p:nn` 4074, 4075, 4114, 4115  
`\str_if_exist:NTF` . . . . . 237, 1869  
`\str_if_in:NnTF` . . . . . 4350, 6047, 7020  
`\str_item:Nn` . . . . . 693, 748, 3913  
`\str_lowercase:n` . . . . . 8949  
`\str_map_break:` . . . . . 3859  
`\str_map_break:n` . 3860, 3864, 3868,  
 3872, 3876, 3880, 3884, 3888, 3892  
`\str_map_function:nn` . . . . . 9319  
`\str_map_inline:Nn` . . . . . 3857  
`\str_new:N` . . . . .  
 139, 1587, 1588, 2058, 2313, 3744, 9301  
`\str_put_right:Nn` . . . . . 7188  
`\str_range:Nnn` . . . . . 2052  
`\str_range:nnn` . . . . . 531, 536, 570  
`\str_replace_all:Nnn` . . . . . 692  
`\str_set:Nn` . . . . . 48, 59,  
 141, 189, 232, 685, 686, 687, 690,  
 706, 823, 1079, 1162, 1174, 1175,  
 1534, 1536, 1538, 1602, 1603, 1631,  
 1642, 1650, 1657, 1721, 1727, 1732,  
 1738, 1747, 1888, 1895, 1900, 2011,  
 2021, 2050, 2088, 2347, 2363, 2386,  
 2456, 2685, 2839, 2844, 2847, 2858,  
 2881, 2882, 3065, 3066, 3067, 3100,  
 3105, 3110, 3115, 3121, 3129, 3131,  
 3137, 3141, 3142, 3143, 3152, 3156,  
 3157, 3158, 3165, 3172, 3175, 3178,  
 3184, 3191, 3194, 3197, 3215, 3242,  
 3287, 3485, 3578, 3584, 3588, 3718,  
 3750, 3772, 3862, 3866, 3870, 3874,  
 3878, 3882, 3886, 3890, 3894, 3944,  
 3946, 3948, 3950, 4046, 4047, 4087,  
 4088, 4119, 4120, 4171, 4203, 4232,  
 4237, 4356, 4396, 4407, 4544, 4798,  
 4800, 4932, 4976, 4978, 4991, 4993,  
 4996, 5251, 5252, 5668, 5712, 5753,  
 6045, 6048, 6127, 6150, 6160, 6169,  
 6266, 6334, 6338, 6340, 6648, 6653,  
 6877, 6891, 6897, 7018, 7021, 7072,  
 7125, 7377, 7386, 7475, 7697, 7699,  
 8641, 8705, 8754, 8805, 8866, 9450  
`\str_set_eq:NN` . . . . . 107,  
 1064, 1593, 1594, 2558, 2829, 2859,  
 3973, 3974, 4173, 4202, 4504, 4514,  
 4533, 4547, 4548, 4822, 5017, 6009,  
 6022, 6035, 6152, 6217, 6867, 6876,  
 6926, 6978, 7361, 7442, 7660, 7664,  
 7688, 7690, 7703, 7717, 7719, 7720  
`\str_uppercase:n` . . . . . 8002  
`\l_tmpa_str` . . . . . 163, 164, 165,  
 166, 167, 168, 169, 173, 189, 193,  
 194, 195, 197, 1926, 1927, 1928,  
 1933, 1941, 2844, 2847, 2852, 2859  
`\string` . . . . . 9295, 9336  
`\subparagraph` . . . . . 27, 82  
`subproblem` (env.) . . . . . 8550  
`subproof` (env.) . . . . . 7361  
`\subproof` . . . . . 7319, 7420  
`\subproofautorefname` . . . . . 7361  
`\subsection` . . . . . 26, 27, 82  
`\subsubsection` . . . . . 27, 82  
`\svar` . . . . . 95, 3798, 4973  
`\symbol` . . . . . 89  
`\symdecl` . . . . . 23,  
 24, 31, 32, 39, 40, 47, 87–89, 99,  
 104, 125, 130, 136, 143, 144, 3007,  
 3744, 7561, 7563, 7592, 7600, 7603  
`\symdef` . . . . . 32, 33, 35, 40, 41, 88,  
 95, 125, 130, 136, 3009, 4524, 7545,  
 7548, 7554, 7556, 7558, 7560, 7570,  
 7571, 7572, 7573, 7579, 7587, 7598,  
 7607, 7608, 7611, 7614, 7616, 7618  
`\Symname` . . . . . 89, 100, 5951  
`\symname` 19, 20, 24, 62, 89, 90, 100, 102, 5951  
`\symref` . . . . . 19,  
 20, 24, 48, 88, 89, 93, 100, 102, 5951  
`\symrefemph` . . . . . 102, 103, 6063  
`\symuse` . 48, 89, 5197, 5276, 5756, 6444,  
 6451, 6462, 6544, 6609, 6679, 6680  
`sys commands:`  
`\sys_get_shell:nnN` . . . . . 45  
`\sys_if_platform_windows:TF` . . . . .  
 . . . . . 51, 106, 683, 743, 970, 1039

## T

|                                                                               |                                                   |                             |                                                                      |
|-------------------------------------------------------------------------------|---------------------------------------------------|-----------------------------|----------------------------------------------------------------------|
| \target                                                                       | 135, 138–140                                      | \Gin@ewidth                 | 8195, 8196, 9674, 9684, 9688                                         |
| \test                                                                         | 73, 115                                           | \Gin@exclamation            | 9674, 9675, 9683                                                     |
| test                                                                          | 110                                               | \Gin@mrepos                 | .. 2061, 2064, 2068, 2089, 2093, 2098                                |
| \testbigspace                                                                 | 9255                                              | \hwexam@bonuspts            | 9636                                                                 |
| \testemptypage                                                                | 73, 115                                           | \hwexam@checktime           | 9630                                                                 |
| \testemptypage                                                                | 9249, 9506                                        | \hwexam@duration            | 9606, 9609, 9615, 9620                                               |
| testheading (env.)                                                            | 74, 116                                           | \hwexam@kw@due              | 9557                                                                 |
| \testheading                                                                  | 9603                                              | \hwexam@kw@forgrading       | 9598                                                                 |
| \testmedspace                                                                 | 9254                                              | \hwexam@kw@given            | 9554                                                                 |
| \testnewpage                                                                  | 73, 115                                           | \hwexam@kw@grade            | 9599                                                                 |
| \testnewpage                                                                  | 9256                                              | \hwexam@kw@probs            | 9599                                                                 |
| \testsmsmallspace                                                             | 73, 73, 73, 115, 115, 115                         | \hwexam@kw@pts              | 9600                                                                 |
| \testsmsmallspace                                                             | 9253                                              | \hwexam@kw@reached          | 9601                                                                 |
| \testspace                                                                    | 73, 115                                           | \hwexam@kw@sum              | 9599                                                                 |
| \testspace                                                                    | 8662, 9252                                        | \hwexam@kw@testemptypage    | 9250, 9510                                                           |
| \TeX                                                                          | 23, 24                                            | \hwexam@min                 | 9607, 9614, 9619, 9630                                               |
| \tex                                                                          | 23, 24                                            | \hwexam@minutes@kw          | 9607                                                                 |
| T <sub>E</sub> X and L <sup>A</sup> T <sub>E</sub> X 2 <sub>E</sub> commands: |                                                   | \hwexam@reqpts              | 9616, 9621, 9633, 9637                                               |
| \@                                                                            | 2107, 2110, 2114                                  | \hwexam@tools               | 9617, 9622                                                           |
| \@addtoreset                                                                  | 8431                                              | \hwexam@totalmin            | 9629, 9630                                                           |
| \@arabic                                                                      | 8284                                              | \hwexam@totalpts            | 9628, 9637                                                           |
| \@author                                                                      | 8263                                              | \if@bonuspoints             | 9632                                                                 |
| \@auxout                                                                      | 1818, 7782, 8544                                  | \if@crustex                 | 310                                                                  |
| \@bonuspointsfalse                                                            | 9634                                              | \itemize@inner              | 8042, 8048, 8057                                                     |
| \@bonuspointstrue                                                             | 9639                                              | \itemize@label              | 8046, 8049, 8052                                                     |
| \@currentHref                                                                 | 7377, 7386, 7475                                  | \itemize@level              | 8040, 8045, 8048, 8057                                               |
| \@currentcounter                                                              | 1551                                              | \itemize@outer              | 8041, 8045                                                           |
| \@currentlabel                                                                | 1552, 7376,<br>7377, 7385, 7386, 7474, 7475, 8092 | \lst@mrepos                 | 2075, 2078, 2082                                                     |
| \@currentlabelname                                                            | 1554, 1555                                        | \ltx@ifpackageloaded        | .....<br>145, 149, 2059, 2073,<br>2086, 7255, 8394, 8971, 9000, 9276 |
| \@currenvir                                                                   | 7976, 7977                                        | \m@switch                   | 5922                                                                 |
| \@dblarg                                                                      | 8256, 8265                                        | \mdf@patchamsthdm           | 8066                                                                 |
| \@gobble                                                                      | 356                                               | \metakeys@show@keys         | 8043                                                                 |
| \@ifnextchar                                                                  | 355                                               | \notesslides@slidelabel     | 8069, 8084                                                           |
| \@ifstar                                                                      | 8204                                              | \ns@author                  | 8256, 8257                                                           |
| \@mainmatterfalse                                                             | 1480, 1490                                        | \ns@title                   | 8265, 8266                                                           |
| \@mainmattertrue                                                              | 1501, 1512                                        | \pgf@temp                   | 2104, 2105, 2106                                                     |
| \@notprerr                                                                    | 144, 1445                                         | \pgfkeys@spdef              | 2104                                                                 |
| \@onlypreamble                                                                | 144, 1445                                         | \pgfutil@empty              | 2106                                                                 |
| \@rusttexfalse                                                                | 302                                               | \pgfutil@InputIfExists      | 2113                                                                 |
| \@title                                                                       | 1266, 1267, 8272                                  | \prematurestop@endsfragment | .....<br>7975, 7978, 7984                                            |
| \activate@excursion                                                           | 8296, 8301                                        | \problem@kw@*               | 8391                                                                 |
| \bb@loaded                                                                    | 8395, 9277                                        | \problem@kw@correct         | 8979, 9095                                                           |
| \beamer@shortauthor                                                           | 8259, 8261, 8276                                  | \problem@kw@feedback        | 8896, 9490                                                           |
| \beamer@shorttitle                                                            | 8268, 8270, 8279                                  | \problem@kw@grading         | 8845, 9431                                                           |
| \c@chapter                                                                    | 7957                                              | \problem@kw@hint            | 8743                                                                 |
| \c@framenumber                                                                | 8284                                              | \problem@kw@minutes         | .....<br>8518, 8606, 8908, 9356, 9403                                |
| \c@part                                                                       | 7969                                              | \problem@kw@note            | 8792                                                                 |
| \compemph@uri                                                                 | 103, 132, 4393, 6055, 6063                        | \problem@kw@points          | .....<br>8893, 9351, 9398, 9487                                      |
| \correction@table                                                             | 9582                                              |                             |                                                                      |
| \defemph@uri                                                                  | 103, 6063                                         |                             |                                                                      |
| \define@key                                                                   | 2060, 2074, 2087                                  |                             |                                                                      |
| \Gin@eheight                                                                  | .. 9675, 9678, 9683, 9688                         |                             |                                                                      |

```

\problem@kw@pts .... 8513, 8601, 8903 \thisstyle ..... 134, 454, 457,
\problem@kw@solution ..... 8682 458, 459, 505, 508, 509, 510, 511,
\problem@kw@wrong ..... 8981, 9097 519, 522, 523, 3091, 3316, 3357,
\problem@restore ..... 8545, 8549, 9576 3761, 3990, 4513, 4532, 4821, 5016
\stex@backend ..... 141, 300 \thistitle 103, 134, 2540, 2541, 2542,
\symrefemph@uri ..... 102, 103, 5967, 5977, 5988, 6063 6925, 8485, 8531, 8532, 8536, 8537
\varempheuri ..... 103, 132, 6008, 6021, 6034, 6063 \thistype ..... 3758, 4529
\textname ..... 24 \thisvarname ..... 4512, 4516, 4820, 4824, 5015, 5019
\text ..... 20 \throwaway ..... 136
\textbf ..... 19, 6105, \tikzinput ..... 118, 2098, 9667, 9672, 9698
\textcolor ..... 9229 \tikzinput commands:
\textsymdecl .. 23, 24, 88, 130, 3008, 3929 \c_tikzinput_image_bool 34, 9656, 9663
\textwarning ..... 108 \tiny ..... 8070, 8084
\textwidth ..... 8193, 8215, 9595 \title ..... 73, 116
\the 255, 256, 258, 260, 262, 2107, 2108, 2109 \title ..... 1363, 8265
\theassignment ..... 9540, 9547 \tl commands:
\thechapter ... 1327, 1396, 1424, 1459, \tl_clear:N ..... 395, 454, 505,
7883, 7888, 7892, 7896, 7900, 7904 767, 1339, 1746, 2197, 2316, 2522,
\theframenumber ..... 8092 2786, 3091, 3316, 3357, 3569, 3725,
\thepage ..... 9508 3726, 3727, 3761, 3909, 3932, 3933,
\theparagraph ..... 7900, 7904 3970, 3990, 4224, 4233, 4329, 4472,
\thechapter ..... 1323, 1393, 1420, 1453, 7878 4513, 4532, 4571, 4766, 4821, 4981,
\theplainproblem ..... 8435, 8436 5004, 5016, 5246, 5256, 5357, 5370,
\thesection ..... 7888, 7892, 7896, 7900, 7904 5568, 5954, 5955, 5956, 6053, 6144,
\thesproblem ..... 8432, 8436, 8535, 6632, 6660, 6701, 6841, 6842, 6843,
8545, 9347, 9369, 9394, 9417, 9540 7148, 7149, 7151, 7152, 7153, 7154,
\thesubparagraph ..... 7904 7431, 7432, 8311, 8856, 8956, 8958,
\thesubsection ... 7892, 7896, 7900, 7904 8959, 9175, 9515, 9516, 9517, 9587,
\thesubsubsection ..... 7896, 7900, 7904 9588, 9589, 9614, 9615, 9616, 9617
\this ..... 57–59, 96, 97, 5248, \tl_const:Nn ..... 5894, 5895
6124, 6560, 6637, 6638, 6642, 6667 \tl_count:N ..... 5512, 5518
\thisarchive ..... 3313, 3354 \tl_count:n ..... 5642, 5653
\thisargs ..... 3760, 4531 \tl_gclear:N ..... 2342
\thiscopyname ..... 3505, 3525, 3549, 3622, 3642, 3663 \tl_gput_left:Nn ..... 367, 369, 370
\thisdeclname ... 3757, 3989, 4203, 4528 \tl_gput_right:Nn ..... 256, 2143,
\thisdecluri ..... 3756, 3988, 4204, 4207, 4527, 4535 2147, 2166, 2319, 2583, 3324, 8297
\thisdefiniens ..... 3759, 4530 \tl_gset:Nn ..... 258, 362, 363,
\thisfor ..... 6927 1019, 1238, 1245, 2167, 2171, 4240,
\thismodulename 134, 2559, 3090, 3315, 3356 5209, 5213, 8584, 8587, 9579, 9580
\thismoduleuri ..... 134, 2558, 3089, 3314, 3355 \tl_gset_eq:NN ..... 46, 2818
\thisname ..... 3506, 3526, 3550, 3623, 3643, 3664 \tl_head:N .. 882, 936, 4029, 6582, 6588
\thisnotation ..... 4205, 4515, 4534, 4823, 5018 \tl_head:n ..... 549, 590, 593, 4717, 5643, 5654
\thisnotationvariant ..... 4202, 4514, 4533, 4822, 5017 \tl_if_empty:NTF ..... 519, 783, 1021, 1256, 1266, 1309,
\ThisStyle ..... 5922 1364, 1773, 2405, 2541, 2551, 3374,

```

8531, 8536, 8578, 8682, 8743, 8792,  
 8883, 8895, 8979, 8981, 8983, 9021,  
 9023, 9025, 9033, 9095, 9097, 9099,  
 9133, 9135, 9137, 9145, 9230, 9479,  
 9535, 9548, 9553, 9556, 9606, 9633  
`\tl_if_empty:nTF` . . . . . 364, 548,  
 564, 580, 589, 747, 756, 878, 879,  
 941, 948, 1094, 1592, 1750, 1772,  
 1930, 1952, 1983, 2041, 2132, 2239,  
 2605, 2620, 2672, 2732, 2757, 2827,  
 2842, 2941, 3000, 3101, 3135, 3150,  
 3164, 3183, 3214, 3241, 3430, 4027,  
 4140, 4153, 4618, 4668, 4975, 5475,  
 5496, 5572, 5594, 5997, 6198, 6329,  
 6378, 6381, 6426, 6436, 6525, 6527,  
 6621, 6630, 6725, 6735, 6799, 6944,  
 7070, 7123, 7139, 7635, 8258, 8267  
`\tl_if_empty_p:N` . . . . . 773  
`\tl_if_eq:NNTF` . . . . . 1136, 6216, 6582, 6588, 9192  
`\tl_if_eq:NnTF` . . . . . 8344, 8491,  
 8494, 8512, 8517, 9350, 9355, 9372  
`\tl_if_eq:nnTF` . . . . . .. 2474, 4742, 5136, 5654, 5699, 5737  
`\tl_if_exist:NTF` . . . . . 214, 255,  
 270, 309, 458, 510, 522, 1282, 1291,  
 1554, 1714, 2575, 6290, 7909, 9333  
`\tl_if_in:NnTF` . . . . . 2254, 2272, 2276  
`\tl_item:nn` . . . . . 5645  
`\tl_map_inline:Nn` . . . . . 2207  
`\tl_map_inline:nn` . . . . . 221, 225  
`\tl_new:N` 961, 1031, 1236, 1589, 2139,  
 2140, 2163, 2174, 2492, 5202, 5206  
`\tl_put_left:Nn` . . . . . .. 4657, 4658, 5401, 8459, 9528  
`\tl_put_right:Nn` . . . . . 1212, 2393, 2768, 2772, 3911, 3912,  
 4335, 4341, 4351, 4578, 4583, 4586,  
 4689, 4697, 4702, 4710, 4773, 5234,  
 5258, 5268, 5382, 6631, 6727, 6774,  
 6785, 6800, 9198, 9591, 9592, 9593  
`\tl_range:nmm` . . . . . 566, 582  
`\tl_set:Nn` . . . . . 215, 252, 449,  
 468, 470, 489, 490, 495, 496, 498,  
 499, 746, 780, 860, 954, 1018, 1274,  
 1315, 1478, 1488, 1604, 1770, 2061,  
 2064, 2559, 2668, 2670, 2673, 2888,  
 3069, 3070, 3071, 3072, 3073, 3313,  
 3354, 3505, 3525, 3549, 3579, 3585,  
 3589, 3622, 3642, 3663, 3756, 3832,  
 3988, 4049, 4050, 4051, 4052, 4053,  
 4090, 4091, 4092, 4093, 4094, 4122,  
 4123, 4124, 4125, 4126, 4204, 4219,  
 4228, 4253, 4270, 4273, 4282, 4285,  
 4291, 4298, 4317, 4527, 4575, 4588,  
 4606, 4627, 4642, 4643, 4655, 4694,  
 4707, 4726, 4739, 4765, 4848, 4893,  
 4934, 4935, 4936, 4937, 4938, 5043,  
 5086, 5170, 5203, 5208, 5210, 5217,  
 5247, 5253, 5254, 5255, 5371, 5392,  
 5512, 5518, 5531, 5537, 5540, 5543,  
 5585, 5671, 5703, 5705, 5755, 5898,  
 5899, 5943, 5944, 6183, 6195, 6267,  
 6344, 6356, 6392, 6393, 6397, 6400,  
 6407, 6443, 6482, 6556, 6562, 6572,  
 6635, 6637, 6662, 6663, 6669, 6678,  
 6726, 6728, 6770, 6775, 6787, 6790,  
 6794, 6797, 6802, 6805, 6809, 6812,  
 7254, 7260, 7878, 7879, 7883, 7884,  
 7888, 7889, 7892, 7893, 7896, 7897,  
 7900, 7901, 7904, 7905, 7920, 8413,  
 8414, 8529, 8560, 8561, 8621, 8914,  
 8970, 8977, 8999, 9004, 9019, 9060,  
 9090, 9093, 9115, 9116, 9131, 9186,  
 9187, 9567, 9573, 9574, 9630, 9636  
`\tl_set_eq:NN` . . . . . 345, 366, 368,  
 373, 1595, 2540, 3089, 3090, 3314,  
 3315, 3355, 3356, 3506, 3526, 3550,  
 3623, 3643, 3664, 3757, 3758, 3759,  
 3760, 3989, 4207, 4275, 4287, 4311,  
 4459, 4467, 4512, 4528, 4529, 4530,  
 4531, 4574, 4811, 4820, 4911, 4916,  
 4920, 5000, 5015, 5451, 5590, 5631,  
 5710, 6488, 6560, 6642, 6667, 6925,  
 7150, 8485, 8489, 8490, 9628, 9629  
`\tl_set_rescan:Nnn` . . . . . 143, 194  
`\tl_tail:n` . . . . . .. 549, 550, 595, 2239, 5616, 6452  
`\tl_to_str:n` . . . . . .. 69, 72, 93, 96, 112, 125, 222,  
 226, 227, 251, 345, 567, 583, 679,  
 702, 709, 808, 826, 828, 831, 850,  
 856, 907, 977, 1069, 1172, 1244,  
 1259, 1525, 1812, 1837, 1838, 1866,  
 1867, 1873, 1922, 1970, 2130, 2151,  
 2167, 2170, 2171, 2208, 2211, 2253,  
 2255, 2271, 2273, 2277, 2295, 2304,  
 2448, 2450, 2451, 2454, 2455, 2485,  
 2486, 2544, 2612, 2613, 2614, 2636,  
 2869, 3023, 3098, 3373, 3382, 3571,  
 4314, 4626, 4640, 4719, 4764, 5557,  
 6193, 6226, 6279, 6350, 6370, 6374,  
 6533, 6661, 6716, 6746, 6760, 6784,  
 6822, 6858, 7779, 8395, 9277, 9319  
`\tl_trim_spaces:N` . . . . . 49  
`\l_tmpa_tl` .. 45, 46, 48, 4345, 4766,  
 4773, 4780, 5197, 5499, 5505, 5510,  
 5512, 5516, 5518, 5521, 5530, 5532,

|                                                                                                                  |                                                                                 |
|------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|
| 5537, 5540, 5543, 5757, 6440, 8977,<br>8993, 9019, 9033, 9093, 9109, 9131,<br>9145, 9186, 9192, 9587, 9591, 9599 | \use_i:nn ..... 6649, 9592                                                      |
| \l_tmpb_tl .....<br>... 5499, 5505, 5507, 5521, 5531,<br>5535, 9187, 9192, 9588, 9592, 9600                      | \use_ii:nn ..... 1862, 2960, 2975, 6689                                         |
| tmpc commands:                                                                                                   | \use_none:nnnnnn ..... 3058                                                     |
| \l_tmpc_tl ..... 9589, 9593, 9601                                                                                | \usebox ..... 8147                                                              |
| \to ..... 7580, 7583, 7593, 7594                                                                                 | \usemodule ..... 15, 18, 22, 23, 26,<br>37, 49, 93, 94, 96, 213, 418, 425, 3077 |
| \today ..... 8286                                                                                                | \usepackage ..... 7, 21, 2051, 7990                                             |
| \TODO ..... 5055, 5056, 6427, 8342                                                                               | \useSGvar ..... 109, 8335                                                       |
| todo internal commands:                                                                                          | \usestructure ..... 96, 6818                                                    |
| \l__todo_mmt_module_str 7660, 7665,<br>7678, 7681, 7684, 7688, 7703, 7719                                        | \usetikzlibrary ..... 7990                                                      |
| \l__todo_newlabel:n ..... 7778, 7785                                                                             | \usebox ..... 80, 2129, 9666                                                    |
| \l__todo_old_metagroup_cd 7704, 7716                                                                             |                                                                                 |
| \l__todo_old_newlabel: ... 7779, 7784                                                                            |                                                                                 |
| \l__todo_stex_module_str .....<br>... 7664, 7665, 7690, 7717                                                     |                                                                                 |
| token commands:                                                                                                  |                                                                                 |
| \c_math_subscript_token .....<br>... 49, 87, 4441, 4442, 4445,<br>4446, 4450, 6572, 7585, 7596, 7598             | <b>V</b>                                                                        |
| \topsep ..... 7270,<br>8595, 8871, 8931, 9071, 9390, 9467                                                        | \varbind ..... 47,<br>52, 95, 100, 6972, 6995, 7110, 7118                       |
| \trueFfalseT ..... 9169                                                                                          | \vardef ..... 35,                                                               |
| \trueTfalseF ..... 9164                                                                                          | 47, 52, 95, 125, 126, 4787, 7141, 7469                                          |
| \type ..... 73, 116                                                                                              | \varempth ..... 103, 6063                                                       |
|                                                                                                                  | \Varname ..... 5951                                                             |
|                                                                                                                  | \varname ..... 5951                                                             |
|                                                                                                                  | \varnotation ..... 95, 4501                                                     |
|                                                                                                                  | \varref ..... 5951                                                              |
|                                                                                                                  | \varseq ..... 43, 95, 4989                                                      |
|                                                                                                                  | \vbox ..... 136, 7256, 8064, 8153, 8168,<br>8663, 8724, 8773, 8825, 8972, 9001  |
|                                                                                                                  | vbox commands:                                                                  |
|                                                                                                                  | \vbox_set:Nn ..... 2176                                                         |
|                                                                                                                  | \vdash ..... 7608                                                               |
|                                                                                                                  | \vfill ..... 7914, 9250, 9510                                                   |
|                                                                                                                  | \vM ..... 54                                                                    |
|                                                                                                                  | \vmS ..... 55                                                                   |
|                                                                                                                  | \vn ..... 43                                                                    |
|                                                                                                                  | \vpop ..... 45, 47                                                              |
|                                                                                                                  | \phantom ..... 9242                                                             |
|                                                                                                                  | \vrule ..... 7256, 8215, 8972, 9001                                             |
|                                                                                                                  | \vskip ..... 7256, 8214, 8216, 8972, 9001                                       |
|                                                                                                                  | \vspace ..... 9252                                                              |
|                                                                                                                  |                                                                                 |
|                                                                                                                  | <b>W</b>                                                                        |
|                                                                                                                  | \wff ..... 20                                                                   |
|                                                                                                                  | \withbrackets ..... 92, 5941, 5949                                              |
|                                                                                                                  |                                                                                 |
|                                                                                                                  | <b>X</b>                                                                        |
|                                                                                                                  | \xdef ..... 9295, 9347, 9394, 9508                                              |
|                                                                                                                  | \xspace . 1282, 1291, 4004, 4005, 4011, 4012                                    |
|                                                                                                                  |                                                                                 |
|                                                                                                                  | <b>Y</b>                                                                        |
|                                                                                                                  | \yesFnoT ..... 9159                                                             |
|                                                                                                                  | \yesTnoF ..... 9154                                                             |
|                                                                                                                  | \yield ..... 7324, 7515, 7518                                                   |