

# **stex.sty: $\text{\TeX}$ 2.0\***

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

2021-11-24

## **Abstract**

TODO

## **1 Introduction**

TODO

---

\*Version v1.9 (last revised 2021/08/01)

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Manual</b>	<b>3</b>
2.1	Modules . . . . .	3
2.2	Semantic Macros and Notations . . . . .	3
2.3	Archives and Imports . . . . .	7
<b>3</b>	<b>Documentation</b>	<b>8</b>
3.1	Utils . . . . .	8
3.2	Files, Paths, URIs . . . . .	9
3.3	MathHub Archives . . . . .	10
3.4	The Module System . . . . .	12
3.5	Symbols and Terms . . . . .	20
3.6	Structural Features . . . . .	25
<b>4</b>	<b>Implementation</b>	<b>25</b>
4.1	The $\text{\LaTeX}$ document class . . . . .	25
4.2	Preliminaries . . . . .	26
4.3	Files, Paths and URIs . . . . .	31
4.4	MathHub Repositories . . . . .	34
4.5	Module System . . . . .	39
4.6	Symbol Declarations . . . . .	56
4.7	Notations . . . . .	62
4.8	Terms . . . . .	72
4.9	Notation Components . . . . .	78
4.10	Structural Features . . . . .	80
4.11	Put these somewhere . . . . .	87
4.12	Metatheory . . . . .	88
4.13	Auxiliary Packages . . . . .	89

## 2 Manual

### 2.1 Modules

`{module}`, `{@module}`

### 2.2 Semantic Macros and Notations

Semantic macros invoke a formally declared symbol.

To declare a symbol (in a module), we use `\symdecl`, which takes as argument the name of the corresponding semantic macro, e.g. `\symdecl{foo}` introduces the macro `\foo`. Additionally, `\symdecl` takes several options, the most important one being its arity. `foo` as declared above yields a *constant* symbol. To introduce an *operator* which takes arguments, we have to specify which arguments it takes.

For example, to introduce binary multiplication, we can do `\symdecl[args=2]{mult}`. We can then supply the semantic macro with arbitrarily many notations, such as `\notation{mult}{#1 #2}`.

#### Example 1

```
\symdecl[args=2]{mult}
\notation{mult}{#1 #2}
 $\mult{a}{b}$ 
```

$(ab)$

.

Since usually, a freshly introduced symbol also comes with a notation from the start, the `\symdef` command combines `\symdecl` and `\notation`. So instead of the above, we could have also written

```
\symdef[args=2]{mult}{#1 #2}
```

Adding more notations like `\notation[cdot]{mult}{#1 \comp{\cdot} #2}` or `\notation[times]{mult}{#1 \comp{\times} #2}` allows us to write  $\mult[cdot]{a}{b}$  and  $\mult[times]{a}{b}$ :

#### Example 2

```
\notation[cdot]{mult}{#1 \comp{\cdot} #2}
\notation[times]{mult}{#1 \comp{\times} #2}
 $\mult[cdot]{a}{b}$  and  $\mult[times]{a}{b}$ 
```

$(a \cdot b)$  and  $(a \times b)$

.

Not using an explicit option with a semantic macro yields the first declared notation, unless changed<sup>1</sup>.

---

<sup>1</sup>EdNOTE: TODO

Outside of math mode, or by using the starred variant `\foo*`, allows to provide a custom notation, where notational (or textual) components can be given explicitly in square brackets.

### Example 3

```
$\mult*{a}[\comp{\ast}]{b}$ is the
\mult[\comp{product of}]{a}[\comp{and}]{b}
```

$a*b$  is the *product of a and b*

In custom mode, prefixing an argument with a star will not print that argument, but still export it to OMDoc:

### Example 4

```
\mult[\comp{Multiplying}]*{$\mult{a}{b}$} again by {$b$} yields...
```

*Multiplying again by b yields...*

The syntax `*[int]` allows switching the order of arguments. For example, given a 2-ary semantic macro `\forevery` with exemplary notation `\forall #1. #2`, we can write

### Example 5

```
\symdecl[ args=2]{forevery}
\forevery*[2]{The proposition $P$}[\comp{holds for every}]*[1]{ $x$ in $A$ }
```

The proposition *P holds for every*  $x \in A$

When using `*[n]`, after reading the provided (*n*th) argument, the “argument counter” automatically continues where we left off, so the `*[1]` in the above example can be omitted.

For a macro with arity  $> 0$ , we can refer to the operator *itself* semantically by suffixing the semantic macro with an exclamation point `!` in either text or math mode. For that reason `\notation` (and thus `\symdef`) take an additional optional argument `op=`, which allows to assign a notation for the operator itself. e.g.

### Example 6

```
\symdef[ args=2, op={+}]{add}{#1 \comp+ #2}
The operator $\add!$ adds two elements, as in $\add ab$.
```

The operator  $+$  adds two elements, as in  $(a+b)$ .

\* is composable with ! for custom notations, as in:

### Example 7

```
\mult![\comp{Multiplication}] (denoted by  $\$ \mult * ! [\comp{cdot}] \$$ ) is defined by...
```

```
Multiplication (denoted by  $\cdot$ ) is defined by...
```

The macro `\comp` as used everywhere above is responsible for highlighting, linking, and tooltips, and should be wrapped around the notation (or text) components that should be treated accordingly. While it is attractive to just wrap a whole notation, this would also wrap around e.g. the arguments themselves, so instead, the user is tasked with marking the notation components themselves.

The precise behaviour of `\comp` is governed by the macro `\@comp`, which takes two arguments: The tex code of the text (unexpanded) to highlight, and the URI of the current symbol. `\@comp` can be safely redefined to customize the behaviour.

The starred variant `\symdecl*{foo}` does not introduce a semantic macro, but still declares a corresponding symbol. `foo` (like any other symbol, for that matter) can then be accessed via `\STEXsymbol{foo}` or (if `foo` was declared in a module `Foo`) via `\STEXModule{Foo}?{foo}`.

both `\STEXsymbol` and `\STEXModule` take any arbitrary ending segment of a full URI to determine which symbol or module is meant. e.g. `\STEXsymbol{Foo?foo}` is also valid, as are e.g. `\STEXModule{path?Foo}?{foo}` or `\STEXsymbol{path?Foo?foo}`

There's also a convient shortcut `\symref{?foo}{some text}` for `\STEXsymbol{?foo}![some text]`

#### 2.2.1 Other Argument Types

So far, we have stated the arity of a semantic macro directly. This works if we only have “normal” (or more precisely: *i*-type) arguments. To make use of other argument types, instead of providing the arity numerically, we can provide it as a sequence of characters representing the argument types – e.g. instead of writing `args=2`, we can equivalently write `args=ii`, indicating that the macro takes two *i*-type arguments.

Besides *i*-type arguments, `STEX` has two other types, which we will discuss now.

The first are *binding* (*b*-type) arguments, representing variables that are *bound* by the operator. This is the case for example in the above `\forevery`-macro: The first argument is not actually an argument that the `forevery` “function” is “applied” to; rather, the first argument is a new variable (e.g. *x*) that is *bound* in the subsequent argument. More accurately, the macro should therefore have been implemented thusly:

```
\symdef[args=bi]{forevery}{\forall #1.\; #2}
```

*b*-type arguments are indistinguishable from *i*-type arguments within `STEX`, but are treated very differently in OMDoc and by MMT. More interesting *within* `STEX` are *a*-type arguments, which represent (associative) arguments of flexible arity, which are provided as comma-separated lists. This allows e.g. better representing the `\mult`-macro above:

### Example 8

```
\symdef[ args=a]{mult}{#1}{#1 \comp\cdot #2}
 $\mult{a,b,c,{d^e},f}$ 
```

$$(a \cdot b \cdot c \cdot d^e \cdot f)$$

As the example above shows, notations get a little more complicated for associative arguments. For every **a**-type argument, the `\notation`-macro takes an additional argument that declares how individual entries in an **a**-type argument list are aggregated. The first notation argument then describes how the aggregated expression is combined into the full representation.

For a more interesting example, consider a flexary operator for ordered sequences in ordered set, that taking arguments `{a,b,c}` and `\mathbb{R}` prints  $a \leq b \leq c \in \mathbb{R}$ . This operator takes two arguments (an **a**-type argument and an **i**-type argument), aggregates the individuals of the associative argument using `\leq`, and combines the result with `\in` and the second argument thusly:

### Example 9

```
\symdef[ args=ai]{numseq}{#1 \comp\in #2}{#1 \comp\leq #2}
 $\numseq{a,b,c}{\mathbb{R}}$ 
```

$$(a \leq b \leq c \in \mathbb{R})$$

Finally, **B**-type arguments combine the functionalities of **a** and **b**, i.e. they represent flexary binding operator arguments.

<sup>2</sup> <sup>3</sup>

### 2.2.2 Precedences

Every notation has an (upwards) *operator precedence* and for each argument *a* (downwards) *argument precedence* used for automated bracketing. For example, a notation for a binary operator `\foo` could be declared like this:

```
\notation[prec=200;500x600]{foo}{#1 \comp{+} #2}
```

assigning an operator precedence of 200, an argument precedence of 500 for the first argument, and an argument precedence of 600 for the second argument.

**TEX** insert brackets thusly: Upon encountering a semantic macro (such as `\foo`), its operator precedence (e.g. 200) is compared to the current downwards precedence (initially `\neginfprec`). If the operator precedence is *larger* than the current downwards precedence, parentheses are inserted around the semantic macro.

Notations for symbols of arity 0 have a default precedence of `\infprec`, i.e. by default, parentheses are never inserted around constants. Notations for symbols with

<sup>2</sup>EDNOTE: what about e.g.  $\int \int \int f dx dy dz$ ?

<sup>3</sup>EDNOTE: “decompose” **a**-type arguments into fixed-arity operators?

arity  $> 0$  have a default operator precedence of 0. If no argument precedences are explicitly provided, then by default they are equal to the operator precedence.

Consequently, if some operator  $A$  should bind stronger than some operator  $B$ , then  $A$ s operator precedence should be smaller than  $B$ s argument precedences.

For example:

### Example 10

```
\notation[prec=100]{plus}{#1 \comp{+} #2}
\notation[prec=50]{times}{#1 \comp{\cdot} #2}
 $\plus{a}{\times{b}{c}}$  and  $\times{a}{\plus{b}{c}}$ 
```

$(a+b \cdot c)$  and  $(a \cdot (b+c))$

## 2.3 Archives and Imports

### 2.3.1 Namespaces

Ideally,  $\text{\texttt{S}\TeX}$  would use arbitrary URIs for modules, with no forced relationships between the *logical* namespace of a module and the *physical* location of the file declaring the module – like MMT does things.

Unfortunately,  $\text{\texttt{T}\TeX}$  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  $\text{\texttt{S}\TeX}$  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  $\text{\texttt{\backslash begin{module}\{Foo\}}}$  occurs in a file  $\text{\texttt{/path/to/file/Foo[.<lang>].tex}}$  which does not belong to an archive, the namespace is  $\text{\texttt{file://path/to/file}}$ .
- If the same statement occurs in a file  $\text{\texttt{/path/to/file/bar[.<lang>].tex}}$ , the namespace is  $\text{\texttt{file://path/to/file/bar}}$ .

In other words: outside of 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<sup>1</sup>.

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**-folder is replaced by the archive’s namespace URI.

### 2.3.2 Paths in Import-Statements

Conversely, here is how namespaces/URIs and file paths are computed in import statements, exemplary  $\text{\texttt{\backslash importmodule}}$ :

- $\text{\texttt{\backslash importmodule}\{Foo\}}$  outside of an archive refers to module **Foo** in the current namespace. Consequently, **Foo** must have been declared earlier in the same document or, if not, in a file  $\text{\texttt{Foo[.<lang>].tex}}$  in the same directory.

<sup>1</sup>which is internally attached to the module name instead, but a user need not worry about that.

- The same statement *within* an archive refers to either the module `Foo` declared earlier in the same document, 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`-folder.
- 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`-folder, 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.
- Finally, `\importmodule{full://uri?Foo}` naturally refers to the module `Foo` in the namespace `full://uri`. Since the file this module is declared in can not be determined directly from the URI, the module must be in memory already, e.g. by being referenced earlier in the same document.

Since this is less compatible with a modular development, using full URIs directly is discouraged.

## 3 Documentation

### 3.1 Utils

---

<code>\sTeX</code>	both print this sTeX logo.
<code>\stex</code>	

---



---

<code>\stex_debug:n</code>	<code>\stex_debug:n {&lt;message&gt;}</code>
----------------------------	----------------------------------------------

---

Logs `<message>`, if the package option `debug` is used.

---

<code>\stex_kpsewhich:n</code>	<code>\stex_kpsewhich:n</code> executes <code>kpsewhich</code> and stores the return in <code>\l_stex_kpsewhich_return_str</code> . This does not require shell escaping.
--------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---



---

<code>\stex_addtosms:n</code>	Adds the provided code to the <code>.sms</code> -file of the document.
-------------------------------	------------------------------------------------------------------------

---

#### 3.1.1 ~~SC~~LaTeX, LaTeXML and HTML Annotations

---

<code>\if@latexml</code>	$\LaTeX$ 2e and $\LaTeX$ 3 conditionals for LaTeXML.
<code>\latexml_if_p:</code>	
<code>\latexml_if:T</code>	
<code>\latexml_if:F</code>	
<code>\latexml_if:TF</code>	

---



We have four macros for annotating generated HTML (via L<sup>A</sup>T<sub>E</sub>XML or S<sub>C</sub>A<sub>L</sub>T<sub>E</sub>X) with attributes:

---

<code>\stex_annotate:nnn</code>	<code>\stex_annotate:nnn {⟨property⟩} {⟨resource⟩} {⟨content⟩}</code>
<code>\stex_annotate_invisible:nnn</code>	
<code>\stex_annotate_invisible:n</code>	

---

Annotates the HTML generated by `⟨content⟩` with

`property="stex:⟨property⟩", resource="⟨resource⟩".`

`\stex_annotate_invisible:n` adds the attributes

`stex:visible="false", style="display:none".`

`\stex_annotate_invisible:nnn` combines the functionality of both.

<code>stex_annotate_env</code>	<code>\begin{stex_annotate_env}{⟨property⟩}{⟨resource⟩}</code> <code>⟨content⟩</code> <code>\end{stex_annotate_env}</code> behaves like <code>\stex_annotate:nnn {⟨property⟩} {⟨resource⟩} {⟨content⟩}</code> .
--------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

### 3.1.2 Languages

---

<code>\c_stex_languages_prop</code>
<code>\c_stex_language_abbrevs_prop</code>

---

Map language abbreviations to their full babel names and vice versa. e.g. `\c_stex_languages_prop{en}` yields `english`, and `\c_stex_language_abbrevs_prop{english}` yields `en`.

## 3.2 Files, Paths, URIs

---

<code>\stex_path_from_string:Nn</code>	<code>\stex_path_from_string:Nn ⟨path-variable⟩ {⟨string⟩}</code>
<code>\stex_path_from_string:(NV cn cV)</code>	

---

turns the `⟨string⟩` into a path by splitting it at `/`-characters and stores the result in `⟨path-variable⟩`. Also applies `\stex_path_canonicalize:N`.

---

<code>\stex_path_to_string:NN</code>	The inverse; turns a path into a string and stores it in the second argument variable, or leaves it in the input stream.
<code>\stex_path_to_string:N</code>	

---



---

<code>\stex_path_canonicalize:N</code>	Canonicalizes the path provided; in particular, resolves <code>.</code> and <code>..</code> path segments.
----------------------------------------	------------------------------------------------------------------------------------------------------------

---



---

<code>\stex_path_if_absolute_p:N *</code>
<code>\stex_path_if_absolute:NTF *</code>

---

Checks whether the path provided is *absolute*, i.e. starts with an empty segment

---

`\c_stex_pwd_seq`  
`\c_stex_pwd_str`  
`\c_stex_mainfile_seq`

---

Store the current working directory as path-sequence and string, respectively, and the (heuristically guessed) full path to the main file, based on the PWD and `\jobname`.

---

`\g_stex_currentfile_seq`

---

The file being currently processed (respecting `\input` etc.)

### Test 1

```
\ExplSyntaxOn
\def\cpath@print#1{
\stex_path_from_string:Nn \l_tmpb_seq { #1 }
\stex_path_to_string:NN \l_tmpb_seq \l_tmpa_str
\str_use:N \l_tmpa_str
}
\ExplSyntaxOff
\begin{center}
\begin{tabular}{|l|l|l|}\hline
path & canonicalized path & expected\\\hline
aaa & \cpath@print{aaa} & aaa \\
../.. / aaa & \cpath@print{../.. / aaa} & & ../.. / aaa \\
aaa/bbb & \cpath@print{aaa/bbb} & & aaa/bbb \\
aaa/. & \cpath@print{aaa/.} & & \\
../.. / aaa/bbb & \cpath@print{../.. / aaa/bbb} & & ../.. / aaa/bbb \\
../aaa / .. / bbb & \cpath@print{../aaa / .. / bbb} & & ../bbb \\
../aaa/bbb & \cpath@print{../aaa/bbb} & & ../aaa/bbb \\
aaa/bbb / .. / ddd & \cpath@print{aaa/bbb / .. / ddd} & & aaa/ddd \\
aaa/bbb / .. / ddd & \cpath@print{aaa/bbb / .. / ddd} & & aaa/bbb/ddd \\
./ & \cpath@print{./} & & \\
aaa/bbb / .. / .. & \cpath@print{aaa/bbb / .. / ..} & & \\
\end{tabular}
\end{center}
```

path	canonicalized path	expected
aaa	aaa	aaa
../.. / aaa	../.. / aaa	../.. / aaa
aaa/bbb	aaa/bbb	aaa/bbb
aaa/. /		
../.. / aaa/bbb	../.. / aaa/bbb	../.. / aaa/bbb
../aaa / .. / bbb	../bbb	../bbb
../aaa/bbb	../aaa/bbb	../aaa/bbb
aaa/bbb / .. / ddd	aaa/ddd	aaa/ddd
aaa/bbb / .. / ddd	aaa/bbb/ddd	aaa/bbb/ddd
./		
aaa/bbb / .. / ..		

## 3.3 MathHub Archives

---

`\mathhub`  
`\c_stex_mathhub_seq`  
`\c_stex_mathhub_str`

---

We determine the path to the local MathHub folder via one of three means, in order of precedence:

1. The `mathhub` package option, or
2. the `\mathhub`-macro, if it has been defined before the `\usepackage{stex}`-statement, or
3. the `MATHHUB` system variable.

In all three cases, `\c_stex_mathhub_seq` and `\c_stex_mathhub_str` are set accordingly.

---

**`\l_stex_current_repository_prop`**

---

Always points to the *current* MathHub repository (if we currently are in one). Has the fields **id**, **ns** (namespace), **narr** (narrative namespace; currently not in use) and **deps** (dependencies; currently not in use).

---

**`\stex_set_current_repository:n`**

---

Sets the current repository to the one with the provided ID. calls `\__stex_mathhub_do_manifest:n`, so works whether this repository's MANIFEST.MF-file has already been read or not.

---

**`\stex_require_repository:n`**

---

Calls `\__stex_mathhub_do_manifest:n` iff the corresponding archive property list does not already exist, and adds a corresponding definition to the `.sms`-file.

---

**`\libinput`**

---

`\libinput{<filename>}`

Inputs `<filename>.tex` from the `lib` folders in the current archive and the `meta-inf`-archive of the current archive group (if existent). Throws an error if no file by that name exists in either folder, includes both if both exist.

## Test 2

```
\ExplSyntaxOn
\stex_require_repository:n { Foo/Bar }
id:-\prop_item:cn {c_stex_mathhub_Foo/Bar_manifest_prop} {id}\ \
narr:-\prop_item:cn {c_stex_mathhub_Foo/Bar_manifest_prop} {narr}\ \
ns:-\prop_item:cn {c_stex_mathhub_Foo/Bar_manifest_prop} {ns}\ \
deps:-\prop_item:cn {c_stex_mathhub_Foo/Bar_manifest_prop} {deps}\ \
\stex_require_repository:n { Bar/Foo }
\ExplSyntaxOff
```

```
id: Foo/Bar
narr:
ns: http://mathhub.info/tests/Foo/Bar
deps:
```

### 3.4 The Module System

---

`\l_stex_current_module_prop`

---

All information of a module is stored as a property list. `\l_stex_current_module_prop` always points to the current module (if existent).

Most importantly, the `content`-field stores all the code to execute on activation; i.e. when this module is being included.

Additionally, it stores:

- The *name* in field `name`,
- the *namespace* in field `ns`,
- this module's *language* in field `lang`,
- if a language module that translates some other modules, the *original* module in field `sig` (for signature),
- the *metatheory* in field `meta`,
- the URIs of all *imported modules* in field `imports`,
- the names of all *declarations* in field `constants`,
- the *file* this module was declared in in field `file`,

---

`\stex_if_in_module_p: *` Conditional for whether we are currently in a module  
`\stex_if_in_module:TF *`

---



---

`\stex_if_module_exists_p:n *`  
`\stex_if_module_exists:nTF *`

---

Conditional for whether a module with the provided URI is already known.

---

`\stex_add_to_current_module:n`  
`\STEXexport`

---

Adds the provided tokens to the `content` field of the current module.

---

`\stex_add_constant_to_current_module:n`

---

Adds the declaration with the provided name to the `constants` field of the current module.

---

`\stex_add_import_to_current_module:n`

---

Adds the module with the provided full URI to the `imports` field of the current module.

---

<code>\stex_modules_compute_namespace:nN</code>	<code>\stex_modules_compute_namespace:nN</code> <code>{\langle namespace \rangle} {\langle path \rangle}</code>
-------------------------------------------------	--------------------------------------------------------------------------------------------------------------------

---

Computes the namespace for file  $\langle path \rangle$  in repository with namespace  $\langle namespace \rangle$  as follows:

If the file is `.../source/sub/file.tex` and the namespace `http://some.namespace/foo`, then the namespace of is `http://some.namespace/foo/sub/file`.

---

<code>\stex_modules_current_namespace:</code>
-----------------------------------------------

---

Computes the current namespace

### Test 3

```

\ExplSyntaxOn
\stex_modules_current_namespace:
Namespace~1:\\ \l_stex_modules_ns_str \\
Faking~a~repository:\\
\stex_set_current_repository:n{Foo/Bar}
\seq_pop_right:NN \g_stex_currentfile_seq \testtemp
\edef\testtempb{\detokenize{source}}
\exp_args:NNo \seq_put_right:Nn \g_stex_currentfile_seq { \testtempb }
\edef\testtempb{\detokenize{test}}
\exp_args:NNo \seq_put_right:Nn \g_stex_currentfile_seq { \testtempb }
\exp_args:NNo \seq_put_right:Nn \g_stex_currentfile_seq { \testtemp }
\stex_modules_current_namespace:
Namespace~2:\\ \l_stex_modules_ns_str
\ExplSyntaxOff

```

```

Namespace 1:
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest
Faking a repository:
Namespace 2:
http://mathhub.info/tests/Foo/Bar/test/stextest

```

#### 3.4.1 The module-environment

<code>module</code>	<code>\begin{module}[\langle options \rangle]{\langle name \rangle}</code> Opens a new module with name $\langle name \rangle$ . TODO document options.
---------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------

---

<code>\stex_modules_heading:</code>	Takes care of the module header, if the <code>showmods</code> package option is true. This macro can be overridden for customization.
-------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------

---

<code>@module</code>	<code>\begin{@module}[\langle options \rangle]{\langle name \rangle}</code> Core functionality of the <code>module-environment</code> without a header.
----------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------

## Test 4

```
\ExplSyntaxOn
\stex_set_current_repository:n {Foo/Bar}
\seq_pop_right:NN \g_stex_currentfile_seq \l_tmpa_tl
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{tests} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{Foo} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{Bar} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{source} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{Foo.tex} }
\begin{@module}{Foo}
Module-path:-
\prop_item:Nn \l_stex_current_module_prop { ns }?
\prop_item:Nn \l_stex_current_module_prop { name }\\
Language:-\prop_item:Nn \l_stex_current_module_prop { lang }\\
Signature:-\prop_item:Nn \l_stex_current_module_prop { sig }\\
Metatheory:-\prop_item:Nn \l_stex_current_module_prop { meta }\\
\end{@module}
\ExplSyntaxOff
```

```
Module path: http://mathhub.info/tests/Foo/Bar?Foo
Language:
Signature:
Metatheory:
```

## Test 5

```
\ExplSyntaxOn
\stex_set_current_repository:n {Foo/Bar}
\stex_debug:n{Test:-\stex_path_to_string:N \g_stex_currentfile_seq }
\seq_pop_right:NN \g_stex_currentfile_seq \l_tmpa_tl
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{tests} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{Foo} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{Bar} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{source} }
\seq_put_right:Nx \g_stex_currentfile_seq { \tl_to_str:n{Foo.tex} }
\stex_debug:n{Test:-\stex_path_to_string:N \g_stex_currentfile_seq }
\begin{module}[title=Foo Bar]{Bar}
Module-path:-
\prop_item:Nn \l_stex_current_module_prop { ns }?
\prop_item:Nn \l_stex_current_module_prop { name }\\
Language:-\prop_item:Nn \l_stex_current_module_prop { lang }\\
Signature:-\prop_item:Nn \l_stex_current_module_prop { sig }\\
Metatheory:-\prop_item:Nn \l_stex_current_module_prop { meta }\\
\end{module}
\ExplSyntaxOff
```

```
Module 3.1[Bar] (FooBar)
Module path: http://mathhub.info/tests/Foo/Bar/Foo?Bar
Language:
Signature:
Metatheory:
```

\l\_stex\_all\_modules\_seq

Stores full URIs for all modules currently in scope.

\STEXModule

\STEXModule {*<fragment>*}

Attempts to find a module whose URI ends with *<fragment>* in the current scope and passes the full URI on to \stex\_invoke\_module:n.

---

`\stex_invoke_module:n`

---

Invoked by `\STEXModule`. Needs to be followed either by `!\langle macro \rangle` or `?{\langle symbolname \rangle}`. In the first case, it stores the full URI in `\langle macro \rangle`; in the second case, it invokes the symbol `\langle symbolname \rangle` in the selected module.

### Test 6

```
\begin{module}{STEXModuleTest1}
\syndeci{foo}
\end{module}
\begin{module}{STEXModuleTest2}
\importmodule{STEXModuleTest1}
\syndeci{foo}
\end{module}
\begin{module}{STEXModuleTest3}
\importmodule{STEXModuleTest2}
\syndeci{foo}
\STEXModule{STEXModuleTest1}!\teststring
\teststring\
\STEXModule{STEXModuleTest2}!\teststring
\teststring\
\STEXModule{STEXModuleTest3}!\teststring
\teststring\
\STEXModule{STEXModuleTest1}?{foo}{\comp{foo1}}\
\STEXModule{STEXModuleTest2}?{foo}{\comp{foo2}}\
\STEXModule{STEXModuleTest3}?{foo}{\comp{foo3}}\
\end{module}
```

Module 3.2[STEXModuleTest1]

Module 3.3[STEXModuleTest2]

Module 3.4[STEXModuleTest3]  
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?STEXModuleTest1  
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?STEXModuleTest2  
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?STEXModuleTest3  
foo1  
foo2  
foo3

### 3.4.2 SMS Mode

“SMS Mode” is used when loading modules from external tex files. It deactivates any output and ignores all  $\TeX$  commands not explicitly allowed via the following lists:

---

`\g_stex_smsmode_allowedmacros_tl`

---

Macros that are executed as is; i.e. with the category code scheme used in SMS mode.

---

`\g_stex_smsmode_allowedmacros_escape_tl`

---

Macros that are executed with the category codes restored.

Importantly, these macros need to call `\stex_smsmode_set_codes:` after reading all arguments. Note, that `\stex_smsmode_set_codes:` takes care of checking whether we are in SMS mode in the first place, so calling this function eagerly is unproblematic.

---

---

`\g_stex_smsmode_allowedenvs_seq`

The names of environments that should be allowed in SMS mode. The corresponding `\begin`-statements are treated like the macros in `\g_stex_smsmode_allowedmacros_escape_tl`, so `\stex_smsmode_set_codes:` should be called at the end of the `\begin`-code. Since `\end`-statements take no arguments anyway, those are called with the SMS mode category code scheme active.

---

---

`\stex_if_smsmode_p: *`  
`\stex_if_smsmode:TF *`

Tests whether SMS mode is currently active.

---

---

`\stex_smsmode_set_codes:`

Sets the current category code scheme to that of the SMS mode, if SMS mode is currently active and if necessary.

This method should be called at the end of every macro or `\begin` environment code that are allowed in SMS mode.

---

---

`\stex_in_smsmode:nn`

`\stex_in_smsmode:nn {<name>} {<code>}`

Executes `<code>` in SMS mode. `<name>` can be arbitrary, but should be distinct, since it allows for nesting `\stex_in_smsmode:nn` without spuriously terminating SMS mode.

### Test 7

```
\immediate\openout\testfile=./tests/sometest.tex
\immediate\write\testfile{\detokenize{\this is \a test}^~J}
\immediate\write\testfile{\detokenize{this \is a \test}}
\immediate\closeout\testfile
\ExplSyntaxOn
\stex_in_smsmode:nn { foo } {
  \input{tests/sometest.tex}
}
\ExplSyntaxOff
```

### 3.4.3 Imports and Inheritance

---

---

`\importmodule`

`\importmodule[<archive-ID>]{<module-path>}`

Imports a module by reading it from a file and “activating” it. `\TeX` determines the module and its containing file by passing its arguments on to `\stex_import_module_path:nn`.



## Test 8

```
\begin{module}{Foo}
\symdecl[name=foo, args=3]{bar}
\symdecl[ args=bai]{foobar}
Meaning:-\present\bar\
\end{module}
Meaning:-\present\bar\
\begin{module}{Importtest}
\importmodule{Foo}
Meaning:-\present\bar\
\end{module}
\begin{module}{Importtest2}
\importmodule{Importtest}
Meaning:-\present\bar\
\end{module}
```

Module 3.5[Foo]

Meaning: >macro:->\stex\_invoke\_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?Foo?foo}<

Meaning: >macro:->\protect \bar <

Module 3.6[Importtest]

Meaning: >macro:->\stex\_invoke\_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?Foo?foo}<

Module 3.7[Importtest2]

Meaning: >macro:->\stex\_invoke\_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?Foo?foo}<

---

**\usemodule**    \importmodule[<archive-ID>]{<module-path>}

---

Like \importmodule, but does not export its contents; i.e. including the current module will not activate the used module

## Test 9

```
\begin{module}{UseTest1}
\symdecl{foo}
\end{module}
\begin{module}{UseTest2}
\usemodule{UseTest1}
\symdecl{bar}
Meaning:-\present\foo\
\end{module}
\begin{module}{UseTest3}
\importmodule{UseTest2}
Meaning:-\present\foo\
Meaning:-\present\bar\

All modules: \ExplSyntaxOn
\seq_use:Nn \l_stex_all_modules_seq {,-} \
All-symbols:-
\seq_use:Nn \l_stex_all_symbols_seq {,-}
\ExplSyntaxOff
\end{module}
```

Module 3.8[UseTest1]

Module 3.9[UseTest2]
Meaning: »macro:->\stex\_invoke\_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?UseTest1?foo}<

Module 3.10[UseTest3]
Meaning: »undefined<
Meaning: »macro:->\stex\_invoke\_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?UseTest2?bar}<
All modules: file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?UseTest3, http://mathhub.info/sTeX?Metath
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?UseTest2
All symbols: http://mathhub.info/sTeX?Metatheory?isa, http://mathhub.info/sTeX?Metatheory?bind, http://mathhub.info/sTeX?Metatheo
http://mathhub.info/sTeX?Metatheory?fromto, http://mathhub.info/sTeX?Metatheory?apply, http://mathhub.info/sTeX?Metatheory?collec
http://mathhub.info/sTeX?Metatheory?seqtype, http://mathhub.info/sTeX?Metatheory?sequence-index, http://mathhub.info/sTeX?Metath
http://mathhub.info/sTeX?Metatheory?aseqfromto, http://mathhub.info/sTeX?Metatheory?letin, http://mathhub.info/sTeX?Metatheory?m
type, http://mathhub.info/sTeX?Metatheory?mathematical-structure, file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-
master/stextest?UseTest2?bar

## Test 10

Circular dependencies:

```

\begin{module}{CircDep1}
\importmodule[Foo/Bar]{circular1?Circular1}
\importmodule[Bar/Foo]{circular2?Circular2}
\present\fooA\
\present\fooB
\end{module}

```

Circular dependencies:

Module 3.11[CircDep1]
»macro:->\stex\_invoke\_symbol:n {http://mathhub.info/tests/Foo/Bar/circular1?Circular1?fooA}<
»macro:->\stex\_invoke\_symbol:n {http://mathhub.info/tests/Bar/Foo/circular2?Circular2?fooB}<

<hr/> <hr/> <code>\stex_import_module_uri:nn</code>	<code>\stex_import_module_uri:nn {⟨archive-ID⟩} {⟨module-path⟩}</code>  Determines the URI of a module by splitting <code>⟨module-path⟩</code> into <code>⟨path⟩?⟨name⟩</code> . If <code>⟨module-path⟩</code> does <i>not</i> contain a <code>?</code> -character, we consider it to be the <code>⟨name⟩</code> , and <code>⟨path⟩</code> to be empty. If <code>⟨archive-ID⟩</code> is empty, it is automatically set to the ID of the current archive (if one exists).
	<ol style="list-style-type: none"> <li>1. If <code>⟨archive-ID⟩</code> is empty: <ol style="list-style-type: none"> <li>(a) If <code>⟨path⟩</code> is empty, then <code>⟨name⟩</code> must have been declared earlier in the same file and retrievable from <code>\g_stex_modules_in_file_seq</code>, or a file with name <code>⟨name⟩.⟨lang⟩.tex</code> must exist in the same folder, containing a module <code>⟨name⟩</code>. That module should have the same namespace as the current one.</li> <li>(b) If <code>⟨path⟩</code> is not empty, it must point to the relative path of the containing file as well as the namespace.</li> </ol> </li> <li>2. Otherwise: <ol style="list-style-type: none"> <li>(a) If <code>⟨path⟩</code> is empty, then <code>⟨name⟩</code> must have been declared earlier in the same file and retrievable from <code>\g_stex_modules_in_file_seq</code>, or a file with name <code>⟨name⟩.⟨lang⟩.tex</code> must exist in the top <code>source</code> folder of the archive, containing a module <code>⟨name⟩</code>. That module should lie directly in the namespace of the archive.</li> <li>(b) If <code>⟨path⟩</code> is not empty, it must point to the path of the containing file as well as the namespace, relative to the namespace of the archive.  If a module by that namespace exists, it is returned. Otherwise, we call <code>\stex_require_module:nn</code> on the <code>source</code> directory of the archive to find the file.</li> </ol> </li> </ol>

---

---

<code>\stex_import_require_module:nnnn</code>	<code>{⟨ns⟩} {⟨archive-ID⟩} {⟨path⟩} {⟨name⟩}</code>
-----------------------------------------------	------------------------------------------------------

Checks whether a module with URI `⟨ns⟩?⟨name⟩` already exists. If not, it looks for a plausible file that declares a module with that URI.

Finally, activates that module by executing its `content`-field.

---

---

<code>\g_stex_module_files_prop</code>	
<code>\g_stex_modules_in_file_seq</code>	

A property list mapping file paths to the lists of all modules declared therein. `\g_stex_modules_in_file_seq` always points to the current file(-stream - `\inputs` are considered the same file).

---

---

<code>\stex_activate_module:n</code>	Activate the module with the provided URI; i.e. executes all macro code of the module's <code>content</code> -field (does nothing if the module is already activated in the current context)
--------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

### 3.5 Symbols and Terms

---

`\symdecl`    `\symdecl[⟨args⟩]{⟨macroname⟩}`

---

Declares a new symbol with semantic macro `\macroname`. Optional arguments are:

- **name**: An (OMDOC) name. By default equal to `⟨macroname⟩`.
- **type**: An (ideally semantic) term. Not used by  $\S\mathrm{TEX}$ , but passed on to MMT for semantic services.
- **local**: A boolean (by default false). If set, this declaration will not be added to the module content, i.e. importing the current module will not make this declaration available.
- **args**: Specifies the “signature” of the semantic macro. Can be either an integer  $0 \leq n \leq 9$ , or a (more precise) sequence of the following characters:
  - i a “normal” argument, e.g. `\symdecl[args=ii]{plus}` allows for `\plus{2}{2}`.
  - a an *associative* argument; i.e. a sequence of arbitrarily many arguments provided as a comma-separated list, e.g. `\symdecl[args=a]{plus}` allows for `\plus{2,2,2}`.
  - b a *variable* argument. Is treated by  $\S\mathrm{TEX}$  like an i-argument, but an application is turned into an `OMBind` in OMDOC, binding the provided variable in the subsequent arguments of the operator; e.g. `\symdecl[args=bi]{forall}` allows for `\forall{x\in\mathrm{Nat}}{x\geq 0}`.

---

`\stex_symdecl_do:n`

---

Implements the core functionality of `\symdecl`, and is called by `\symdecl` and `\symdef`.

Ultimately stores the symbol `⟨URI⟩` in the property list `\g_stex_symdecl_⟨URI⟩_prop` with fields:

- **name** (string),
- **module** (string),
- **notations** (sequence of strings; initially empty),
- **local** (boolean),
- **type** (token list),
- **args** (string of is, as and bs),
- **arity** (integer string),
- **assocs** (integer string; number of associative arguments),

## Test 11

```
\begin{module}{SymdeclTest}
\symdecl[name=foo, args=3]{bar}
\symdecl[name=foobar, args=iab]{bari}
\symdecl[def=\bar* abc]{bardef}
\ExplSyntaxOn
Meaning:-\present\bar\
\stex_get_symbol:n { bar }
Result:-\l_stex_get_symbol_uri_str\
Meaning:-\present\bardef\
\ExplSyntaxOff
\end{module}
```

```
Module 3.12[SymdeclTest]
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?SymdeclTest?foo}<
Result: file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?SymdeclTest?foo
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?SymdeclTest?bardef}<
```

\l\_stex\_all\_symbols\_seq

Stores full URIs for all modules currently in scope.

\stex\_get\_symbol:n

Computes the full URI of a symbol from a macro argument, e.g. the macro name, the macro itself, the full URI...

\STEXsymbol

Uses `\stex_get_symbol:n` to find the symbol denoted by the first argument and passes the result on to `\stex_invoke_symbol:n`

\symref

`\symref{⟨symbol⟩}{⟨text⟩}`  
shortcut for `\STEXsymbol{⟨symbol⟩}! [⟨text⟩]`

\stex\_invoke\_symbol:n

Executes a semantic macro. Outside of math mode or if followed by `*`, it continues to `\stex_term_custom:nn`. In math mode, it uses the default or optionally provided notation of the associated symbol.

If followed by `!`, it will invoke the symbol *itself* rather than its application (and continue to `\stex_term_custom:nn`), i.e. it allows to refer to `\plus!`[addition] as an operation, rather than `\plus`[addition of]{some}{terms}.

\notation

`\notation[⟨args⟩]{⟨symbol⟩}{⟨notations+⟩}`  
Introduces a new notation for `⟨symbol⟩`, see `\stex_notation_do:nn`

---

`\stex_notation_do:nn`

---

`\stex_notation_do:nn{<URI>}{<notations+>}`

Implements the core functionality of `\notation`, and is called by `\notation` and `\symdef`.

Ultimately stores the notation in the property list `\g_stex_notation_<URI>#<variant>#<lang>_prop` with fields:

- symbol (URI string),
- language (string),
- variant (string),
- opprec (integer string),
- argprecs (sequence of integer strings)

### Test 12

```
\begin{module}{NotationTest}
\importmodule{Foo}
\notation{foo, prec=500;20x20x20}{bar}{\comp\langle {#1} ^ {#2} _ {#3} \comp\rangle }
\notation{foo, prec=500;20x20x20}{foobar}{\comp\langle #1 \comp\mid [ #2 ] ^ {#3} \comp\rangle }{ {#1}_ {\com
\end{module}
```

Module 3.13[NotationTest]

---

`\symdef`

---

`\symdef[<args>]{<symbol>}{<notations+>}`

Combines `\symdecl` and `\notation` by introducing a new symbol and assigning a new notation for it.

### Test 13

```
\begin{module}{SymdefTest}
\symdef[ args=a, prec=50]{plus}{ #1 }{#1 \comp+ #2}
$\plus{a,b,c}$
\end{module}
```

Module 3.14[SymdefTest]  
(a+b+c)

---

`\_stex_term_math_oms:nnnn`  
`\_stex_term_math_oma:nnnn`  
`\_stex_term_math_omb:nnnn`

---

`<URI><fragment><precedence><body>`

Annotates `<body>` as an OMDOC-term (OMID, OMA or OMBIND, respectively) with head symbol `<URI>`, generated by the specific notation `<fragment>` with (upwards) operator precedence `<precedence>`. Inserts parentheses according to the current downwards precedence and operator precedence.



<div> <div> Module 3.16[MathTest2] </div> <div> <math display="block">\langle a   [b, c, d, e, f]^g \rangle</math> <math display="block">(a + (b \cdot c)) \text{ and } (a \cdot \frac{a}{b} + \frac{a}{c})</math> <math display="block">(a + (b \cdot c)) \text{ and } [a \cdot \frac{a}{b} + \frac{a}{c}]</math> </div> </div>	<div> <div> <math display="block">\langle a   [b, c]^g \rangle</math> <math display="block">(a + (b \cdot c)) \text{ and } (a \cdot \frac{a}{b} + \frac{a}{c})</math> </div> <div> <math display="block">\langle a   [b]^c \rangle</math> </div> </div>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<u><code>\stex_term_custom:nn</code></u>	<code>\stex_term_custom:nn{&lt;URI&gt;}{&lt;args&gt;}</code>
	Implements custom one-time notation. Invoked by <code>\stex_invoke_symbol:n</code> in text mode, or if followed by <code>*</code> in math mode, or whenever followed by <code>!</code> .

### Test 16

<pre> \begin{module}{TextTest} \importmodule{Foo}  \bar[some ]a[ and some ]b[ and also some ]c[ here].  \$\bar*[\text{some }]a[\text{ and some }]b[\text{ and also some }]c[\text{ here}]\$.  \$\bar![\mathtt{bar}]\$  \bar*{a}*{b}[or just some ]c  \bar![bar]  \bar[or first ]*[2]{b}[ , then ]*[3]{c}[ , and finally ]a  \end{module} </pre>	
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

<div> <div> Module 3.17[TextTest] </div> <div> some a and some b and also some c here. </div> <div> or just some c </div> <div> bar </div> <div> or first b, then c, and finally a </div> </div>	
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

<u><code>\stex_highlight_term:nn</code></u>	<code>\stex_highlight_term:nn{&lt;URI&gt;}{&lt;args&gt;}</code>
	Establishes a context for <code>\comp</code> . Stores the URI in a variable so that <code>\comp</code> knows which symbol governs the current notation.

<u><code>\comp</code></u>	<code>\comp{&lt;args&gt;}</code>
<u><code>\@comp</code></u>	Marks <code>&lt;args&gt;</code> as a notation component of the current symbol for highlighting, linking, etc.
<u><code>\@defemph</code></u>	

The precise behavior is governed by `\@comp`, which takes as additional argument the URI of the current symbol. By default, `\@comp` adds the URI as a PDF tooltip and colors the highlighted part in blue.

`\@defemph` behaves like `\@comp`, and can be similarly redefined, but marks an expression as *definiendum* (used by `\definiendum`)



---

<code>\STEXinvisible</code>	Exports its argument as OMDOC (invisible), but does not produce PDF output. Useful e.g. for semantic macros that take arguments that are not part of the symbolic notation.
-----------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

---



---

<code>\ellipses</code>	TODO
------------------------	------

---

### 3.6 Structural Features

<code>symboldoc</code>	$\begin{array}{l} \backslash\text{begin}\{\langle\text{symboldoc}\rangle\}\{\langle\text{symbols}\rangle\} \langle\text{text}\rangle \backslash\text{end}\{\langle\text{symboldoc}\rangle\} \\ \text{Declares } \langle\text{text}\rangle \text{ to be a (natural language, encyclopaedic) description of } \{\langle\text{symbols}\rangle\} \\ \text{(a comma separated list of symbol identifiers).} \end{array}$
------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

#### 3.6.1 Structures

<code>structure</code>	TODO
------------------------	------

#### Test 17

```

\begin{module}{StructureTest1}
\begin{structure}[name=Magma]{magma}
\symdef{universe}{\comp M}
\symdef[ args=2]{op}{#1 \comp\circ #2}

$$\S\isa{\op{ab}}{\universe}$$

\end{structure}

\ExplSyntaxOn
\prop_get:NnN \g_stex_last_feature__prop {fields} \l_tmpa_seq
\seq_use:Nn \l_tmpa_seq {,}
\ExplSyntaxOff

\present\magma

\instantiate{magma}[
universe ! {\comp U},
op ! {\comp+ #2}
]{mM}
\notation[op = U]{mM/universe}{\comp U}
\notation[op = +]{mM/op}{#1 \comp+ #2}

Test: 
$$\S\mM{\op{ab}}$$

Test2: 
$$\S\mM{\}$$

\end{module}

```

```

Module 3.18[StructureTest1]
(aob:(M))
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?StructureTest1/Magma-feature?universe,file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?StructureTest1/Magma-feature?op
>macro->stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?StructureTest1?Magma}
Test: (a+b)
Test2: ((U,+))

```

## 4 Implementation

### 4.1 The $\text{\TeX}$ document class

```

1 \*cls
2 \RequirePackage{expl3,13keys2e}

```

```

3 \ProvidesExplClass{stex}{2021/08/01}{1.9}{bla}
4 \LoadClass[border=1px,varwidth]{standalone}
5 \setlength\textwidth{15cm}
6 %\g@addto@macro{\@parboxrestore}{\setlength\parskip{\baselineskip}}
7
8 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{stex}}
9 \ProcessOptions
10
11 \RequirePackage{stex}
12 \</cls>

```

## 4.2 Preliminaries

```

13 <*package>
14 \RequirePackage{expl3,l3keys2e}
15 \ProvidesExplPackage{stex}{2021/08/01}{1.9}{bla}

Package options:
16 \keys_define:nn { stex } {
17   debug      .bool_set:N = \c_stex_debug_bool ,
18   showmods   .bool_set:N = \c_stex_showmods_bool ,
19   lang        .clist_set:N = \c_stex_languages_clist ,
20   mathhub     .tl_set_x:N = \mathhub ,
21   sms         .bool_set:N = \c_stex_persist_mode_bool ,
22   image       .bool_set:N = \c_tikzinput_image_bool
23 }
24 \ProcessKeysOptions { stex }

```

**\sTeX** The sTeX logo:

```

25 \protected\def\sTeX{%
26   \ifundefined{texorpdfstring}%
27   {\let\texorpdfstring\@firstoftwo}%
28   }%
29   \texorpdfstring{\raisebox{-.5ex}{S}\kern-.5ex\TeX}{sTeX}\xspace%
30 }
31 \def\sTeX{\sTeX}

```

(End definition for \sTeX. This function is documented on page 8.)

Messages

```

32 \msg_new:nnn{stex}{debug}{}
33 \msg_new:nnn{stex}{warning/nomathhub}{
34   MATHHUB~system~variable~not~found~and~no~
35   \detokenize{\mathhub}-value~set!
36 }
37 \msg_new:nnn{stex}{error/norepository}{}

```

**\stex\_debug:n** Debug mode

```

38 \cs_new_protected:Nn \stex_debug:n {
39   \bool_if:nT{\c_stex_debug_bool}{
40     \exp_args:Nnnx\msg_set:nnn{stex}{debug}{\Debug:~#1\\}
41     \msg_term:nn{stex}{debug} % should be \msg_note:nn
42   }
43 }
44
45 \stex_debug:n{Debug~mode~on}

```

(End definition for `\stex_debug:n`. This function is documented on page 8.)

```
\c__stex_sms_iow File variable used for the sms-File
46 \iow_new:N \c__stex_sms_iow
47 \AddToHook{begindocument}{
48   \bool_if:NTF \c_stex_persist_mode_bool {
49     \ExplSyntaxOn \input{\jobname.sms} \ExplSyntaxOff
50   } {
51     \iow_open:Nn \c__stex_sms_iow {\jobname.sms}
52   }
53 }
54 \AddToHook{enddocument}{
55   \bool_if:NF \c_stex_persist_mode_bool {
56     \iow_close:N \c__stex_sms_iow
57   }
58 }
```

(End definition for `\c__stex_sms_iow`.)

```
\stex_addtosms:n
59 \cs_new_protected:Nn \stex_addtosms:n {
60   \bool_if:NF \c_stex_persist_mode_bool {
61     \iow_now:Nn \c__stex_sms_iow { #1 }
62   }
63 }
```

(End definition for `\stex_addtosms:n`. This function is documented on page 8.)

#### 4.2.1 L<sup>A</sup>T<sub>E</sub>X<sub>M</sub>L and S<sup>C</sup>A<sub>L</sub>T<sub>E</sub>X

```
64 \RequirePackage{scalatex}
```

We add the namespace abbreviation `ns:stex="http://kwarc.info/ns/sTeX"` to S<sup>C</sup>A<sub>L</sub>T<sub>E</sub>X:

```
65 \scalatex_add_Namespace:nn{stex}{http://kwarc.info/ns/sTeX}
```

```
\if@latexml Conditionals for LATEXML:
\latexml_if_p:
\latexml_if:TF
66 \ifcsname if@latexml\endcsname\else
67   \expandafter\newif\csname if@latexml\endcsname\@latexmlfalse
68 \fi
69
70 \prg_new_conditional:Nnn \latexml_if: {p, T, F, TF} {
71   \if@latexml
72     \prg_return_true:
73   \else:
74     \prg_return_false:
75   \fi:
76 }
```

(End definition for `\if@latexml` and `\latexml_if:TF`. These functions are documented on page 8.)

## 4.2.2 HTML Annotations

77 `<@=stex_annotate>`

`\l__stex_annotate_arg_tl`  
`\c__stex_annotate_emptyarg_tl`

Used by annotation macros to ensure that the HTML output to annotate is not empty.

```
78 \tl_new:N \l__stex_annotate_arg_tl
79 \tl_const:Nx \c__stex_annotate_emptyarg_tl {
80   \scalatex_if:TF {
81     \scalatex_direct_HTML:n { \c_ampsand_str lrm; }
82   }{-}
83 }
```

(End definition for `\l__stex_annotate_arg_tl` and `\c__stex_annotate_emptyarg_tl`.)

`\__stex_annotate_checkempty:n`

```
84 \cs_new_protected:Nn \__stex_annotate_checkempty:n {
85   \tl_set:Nn \l__stex_annotate_arg_tl { #1 }
86   \tl_if_empty:NT \l__stex_annotate_arg_tl {
87     \tl_set_eq:NN \l__stex_annotate_arg_tl \c__stex_annotate_emptyarg_tl
88   }
89 }
```

(End definition for `\__stex_annotate_checkempty:n`.)

`\stex_annotate:nnw`

We define four macros for introducing attributes in the HTML output. The definitions depend on the “backend” used (L<sup>A</sup>T<sub>E</sub>X<sub>M</sub>L, S<sup>C</sup>A<sub>L</sub>T<sub>E</sub>X, p<sub>d</sub>f<sub>l</sub>at<sub>e</sub>x).

The p<sub>d</sub>f<sub>l</sub>at<sub>e</sub>x-macros largely do nothing; the S<sup>C</sup>A<sub>L</sub>T<sub>E</sub>X-implementations are pretty clear in what they do, the L<sup>A</sup>T<sub>E</sub>X<sub>M</sub>L-implementations resort to perl bindings.

```
90 \scalatex_if:TF{
91   \cs_new_protected:Nn \stex_annotate:nnn {
92     \__stex_annotate_checkempty:n { #3 }
93     \scalatex_annotate_HTML:nn {
94       property="stex:#1" ~
95       resource="#2"
96     } {
97       \tl_use:N \l__stex_annotate_arg_tl
98     }
99   }
100   \cs_new_protected:Nn \stex_annotate_invisible:n {
101     \__stex_annotate_checkempty:n { #1 }
102     \scalatex_annotate_HTML:nn {
103       stex:visible="false" ~
104       style:display="none"
105     } {
106       \tl_use:N \l__stex_annotate_arg_tl
107     }
108   }
109   \cs_new_protected:Nn \stex_annotate_invisible:nnn {
110     \__stex_annotate_checkempty:n { #3 }
111     \scalatex_annotate_HTML:nn {
112       property="stex:#1" ~
113       resource="#2" ~
114       stex:visible="false" ~
115       style:display="none"
```

```

116   } {
117     \tl_use:N \l__stex_annotate_arg_tl
118   }
119 }
120 \NewDocumentEnvironment{stex_annotate_env} { m m } {
121   \par
122   \scalatex_annotate_HTML_begin:n {
123     property="stex:#1" ~
124     resource="#2"
125   }
126 }{
127   \scalatex_annotate_HTML_end:
128 }
129 }{
130   \latexml_if:TF {
131     \cs_new_protected:Nn \stex_annotate:nnn {
132       \__stex_annotate_checkempty:n { #3 }
133       \mode_if_math:TF {
134         \cs:w latexml@annotate@math\cs_end:{#1}{#2}{
135           \tl_use:N \l__stex_annotate_arg_tl
136         }
137       }{
138         \cs:w latexml@annotate@text\cs_end:{#1}{#2}{
139           \tl_use:N \l__stex_annotate_arg_tl
140         }
141       }
142     }
143     \cs_new_protected:Nn \stex_annotate_invisible:n {
144       \__stex_annotate_checkempty:n { #1 }
145       \mode_if_math:TF {
146         \cs:w latexml@invisible@math\cs_end:{
147           \tl_use:N \l__stex_annotate_arg_tl
148         }
149       } {
150         \cs:w latexml@invisible@text\cs_end:{
151           \tl_use:N \l__stex_annotate_arg_tl
152         }
153       }
154     }
155     \cs_new_protected:Nn \stex_annotate_invisible:nnn {
156       \__stex_annotate_checkempty:n { #3 }
157       \cs:w latexml@annotate@invisible\cs_end:{#1}{#2}{
158         \tl_use:N \l__stex_annotate_arg_tl
159       }
160     }
161     \NewDocumentEnvironment{stex_annotate_env} { m m } {
162       \par\begin{latexml@annotateenv}{#1}{#2}
163     }{
164       \end{latexml@annotateenv}
165     }
166   }{
167     \cs_new_protected:Nn \stex_annotate:nnn {#3}
168     \cs_new_protected:Nn \stex_annotate_invisible:n {}
169     \cs_new_protected:Nn \stex_annotate_invisible:nnn {}

```

```

170 \NewDocumentEnvironment{stex_annotate_env} { m m } {\par}{\}
171 }
172 }

```

(End definition for `\stex_annotate:nnn`, `\stex_annotate_invisible:n`, and `\stex_annotate_invisible:nnn`. These functions are documented on page 9.)

### 4.2.3 Languages

```

173 <@@=stex_language>

```

`\c_stex_languages_prop`

`\c_stex_language_abbrevs_prop`

We store language abbreviations in two (mutually inverse) property lists:

```

174 \prop_const_from_keyval:Nn \c_stex_languages_prop {
175   en = english ,
176   de = ngerman ,
177   ar = arabic ,
178   bg = bulgarian ,
179   ru = russian ,
180   fi = finnish ,
181   ro = romanian ,
182   tr = turkish ,
183   fr = french
184 }
185
186 \prop_const_from_keyval:Nn \c_stex_language_abbrevs_prop {
187   english   = en ,
188   ngerman   = de ,
189   arabic    = ar ,
190   bulgarian = bg ,
191   russian   = ru ,
192   finnish   = fi ,
193   romanian  = ro ,
194   turkish   = tr ,
195   french    = fr
196 }
197 % todo: chinese simplified (zhs)
198 %       chinese traditional (zht)

```

(End definition for `\c_stex_languages_prop` and `\c_stex_language_abbrevs_prop`. These variables are documented on page 9.)

we use the `lang-package` option to load the corresponding babel languages:

```

199 \clist_if_empty:NF \c_stex_languages_clist {
200   \clist_clear:N \l_tmpa_clist
201   \clist_map_inline:Nn \c_stex_languages_clist {
202     \prop_get:NnNTF \c_stex_languages_prop { #1 } \l_tmpa_str {
203       \clist_put_right:No \l_tmpa_clist \l_tmpa_str
204     } {
205       \msg_set:nnn{stex}{error/unknownlanguage}{
206         Unknown~language~\l_tmpa_str
207       }
208       \msg_error:nn{stex}{error/unknownlanguage}
209     }
210   }
211   \stex_debug:n {Languages:~\clist_use:Nn \l_tmpa_clist {,~} }
212   \RequirePackage[\clist_use:Nn \l_tmpa_clist ,]{babel}
213 }

```

## 4.3 Files, Paths and URIs

214 `<@@=stex_path>`

### 4.3.1 Generic Path Handling

We treat paths as L<sup>A</sup>T<sub>E</sub>X3-sequences (of the individual path segments, i.e. separated by a /-character) unix-style; i.e. a path is absolute if the sequence starts with an empty entry.

```

\stex_path_from_string:Nn
\stex_path_from_string:NV
\stex_path_from_string:cn
\stex_path_from_string:cV
215 \cs_new_protected:Nn \stex_path_from_string:Nn {
216   \str_set:Nx \l_tmpa_str { #2 }
217   \str_if_empty:NTF \l_tmpa_str {
218     \seq_clear:N #1
219   }{
220     \exp_args:NNNo \seq_set_split:Nnn #1 / { \l_tmpa_str }
221     \sys_if_platform_windows:T{
222       \seq_clear:N \l_tmpa_tl
223       \seq_map_inline:Nn #1 {
224         \seq_set_split:Nnn \l_tmpb_tl \c_backslash_str { ##1 }
225         \seq_concat:NNN \l_tmpa_tl \l_tmpa_tl \l_tmpb_tl
226       }
227       \seq_set_eq:NN #1 \l_tmpa_tl
228     }
229     \stex_path_canonicalize:N #1
230   }
231 }
232 \cs_generate_variant:Nn \stex_path_from_string:Nn
233 { NV, cn, cV }

```

(End definition for `\stex_path_from_string:Nn`. This function is documented on page 9.)

```

\stex_path_to_string:NN
\stex_path_to_string:N
234 \cs_new_protected:Nn \stex_path_to_string:NN {
235   \exp_args:NNe \str_set:Nn #2 { \seq_use:Nn #1 / }
236 }
237
238 \cs_new:Nn \stex_path_to_string:N {
239   \seq_use:Nn #1 /
240 }

```

(End definition for `\stex_path_to_string:NN` and `\stex_path_to_string:N`. These functions are documented on page 9.)

```

\c__stex_path_dot_str . and .., respectively.
\c__stex_path_up_str
241 \str_const:Nn \c__stex_path_dot_str {.}
242 \str_const:Nn \c__stex_path_up_str {...}

```

(End definition for `\c__stex_path_dot_str` and `\c__stex_path_up_str`.)

`\stex_path_canonicalize:N` Canonicalizes the path provided; in particular, resolves . and .. path segments.

```

243 \cs_new_protected:Nn \stex_path_canonicalize:N {
244   \seq_if_empty:NF #1 {
245     \seq_clear:N \l_tmpa_seq
246     \seq_get_left:NN #1 \l_tmpa_tl
247     \str_if_empty:NT \l_tmpa_tl {

```

```

248     \seq_put_right:Nn \l_tmpa_seq {}
249 }
250 \seq_map_inline:Nn #1 {
251   \str_set:Nn \l_tmpa_tl { ##1 }
252   \str_if_eq:NNTF \l_tmpa_tl \c__stex_path_dot_str {} {
253     \str_if_eq:NNTF \l_tmpa_tl \c__stex_path_up_str {
254       \seq_if_empty:NTF \l_tmpa_seq {
255         \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq {
256           \c__stex_path_up_str
257         }
258       }{
259         \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
260         \str_if_eq:NNTF \l_tmpa_tl \c__stex_path_up_str {
261           \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq {
262             \c__stex_path_up_str
263           }
264         }{
265           \seq_pop_right:NN \l_tmpa_seq \l_tmpb_tl
266         }
267       }
268     }{
269       \str_if_empty:NF \l_tmpa_tl {
270         \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq { \l_tmpa_tl }
271       }
272     }
273   }
274 }
275 \seq_gset_eq:NN #1 \l_tmpa_seq
276 }
277 }

```

(End definition for `\stex_path_canonicalize:N`. This function is documented on page 9.)

`\stex_path_if_absolute_p:N`  
`\stex_path_if_absolute:NTF`

```

278 \prg_new_conditional:Nnn \stex_path_if_absolute:N {p, T, F, TF} {
279   \seq_if_empty:NNTF #1 {
280     \prg_return_false:
281   }{
282     \seq_get_left:NN #1 \l_tmpa_tl
283     \str_if_empty:NNTF \l_tmpa_tl {
284       \prg_return_true:
285     }{
286       \prg_return_false:
287     }
288   }
289 }

```

(End definition for `\stex_path_if_absolute:NTF`. This function is documented on page 9.)

### 4.3.2 PWD and kpsewhich

`\stex_kpsewhich:n`

```

290 \str_new:N\l_stex_kpsewhich_return_str
291 \cs_new_protected:Nn \stex_kpsewhich:n {

```



```

292 \sys_get_shell:nnN { kpsewhich ~ #1 } { } \l_tmpa_tl
293 \exp_args:NNo\str_set:Nn\l_stex_kpsewhich_return_str{\l_tmpa_tl}
294 \tl_trim_spaces:N \l_stex_kpsewhich_return_str
295 }

```

(End definition for `\stex_kpsewhich:n`. This function is documented on page 8.)

We determine the PWD

`\c_stex_pwd_seq`  
`\c_stex_pwd_str`

```

296 \sys_if_platform_windows:TF{
297   \stex_kpsewhich:n{-expand-var~\c_percent_str CD\c_percent_str}
298 }{
299   \stex_kpsewhich:n{-var-value~PWD}
300 }
301
302 \stex_path_from_string:Nn\c_stex_pwd_seq\l_stex_kpsewhich_return_str
303 \stex_path_to_string:NN\c_stex_pwd_seq\c_stex_pwd_str
304 \stex_debug:n {PWD:~\str_use:N\c_stex_pwd_str}

```

(End definition for `\c_stex_pwd_seq` and `\c_stex_pwd_str`. These variables are documented on page 10.)

### 4.3.3 File Hooks and Tracking

```

305 <@=stex_files>

```

We introduce hooks for file inputs that keep track of the absolute paths of files used. This will be useful to keep track of modules, their archives, namespaces etc.

Note that the absolute paths are only accurate in `\input`-statements for paths relative to the PWD, so they shouldn't be relied upon in any other setting than for  $\TeX$ -purposes.

`\g__stex_files_stack` keeps track of file changes

```

306 \seq_gclear_new:N\g__stex_files_stack

```

(End definition for `\g__stex_files_stack`.)

`\c_stex_mainfile_seq`

```

307 \stex_path_from_string:Nn \c_stex_mainfile_seq {
308   \c_stex_pwd_str/\jobname.tex
309 }

```

(End definition for `\c_stex_mainfile_seq`. This variable is documented on page 10.)

`\g_stex_currentfile_seq` Hooks for file inputs that push/pop `\g__stex_files_stack` to update `\c_stex_mainfile_seq`.

```

310 \seq_gclear_new:N\g_stex_currentfile_seq
311 \AddToHook{file/before}{
312   \stex_path_from_string:Nn\g_stex_currentfile_seq{\CurrentFilePath}
313   \stex_path_if_absolute:NTF\g_stex_currentfile_seq{
314     \exp_args:NNe\seq_put_right:Nn\g_stex_currentfile_seq{\CurrentFile}
315   }{
316     \stex_path_from_string:Nn\g_stex_currentfile_seq{
317       \c_stex_pwd_str/\CurrentFilePath/\CurrentFile
318     }
319   }

```

```

320 \seq_gset_eq:NN\g_stex_currentfile_seq\g_stex_currentfile_seq
321 \exp_args:NNo\seq_gpush:Nn\g__stex_files_stack\g_stex_currentfile_seq
322 }
323 \AddToHook{file/after}{
324   \seq_if_empty:NF\g__stex_files_stack{
325     \seq_gpop:NN\g__stex_files_stack\l_tmpa_seq
326   }
327   \seq_if_empty:NTF\g__stex_files_stack{
328     \seq_gset_eq:NN\g_stex_currentfile_seq\c_stex_mainfile_seq
329   }{
330     \seq_get:NN\g__stex_files_stack\l_tmpa_seq
331     \seq_gset_eq:NN\g_stex_currentfile_seq\l_tmpa_seq
332   }
333 }

```

(End definition for `\g_stex_currentfile_seq`. This variable is documented on page 10.)

## 4.4 MathHub Repositories

```

334 <@@=stex_mathhub>
\mathhub
\c_stex_mathhub_seq
\c_stex_mathhub_str
335 \str_if_empty:NTF\mathhub{
336   \stex_kpsewhich:n{-var-value~MATHHUB}
337   \str_set_eq:NN\c_stex_mathhub_str\l_stex_kpsewhich_return_str
338 }
339 \str_if_empty:NTF\c_stex_mathhub_str{
340   \msg_warning:nn{stex}{warning/nomathhub}
341 }{
342   \stex_debug:n {MathHub:~\str_use:N\c_stex_mathhub_str}
343   \exp_args:NNo \stex_path_from_string:Nn\c_stex_mathhub_seq\c_stex_mathhub_str
344 }
345 }{
346   \stex_path_from_string:Nn \c_stex_mathhub_seq \mathhub
347   \stex_path_if_absolute:NF \c_stex_mathhub_seq {
348     \exp_args:NNx \stex_path_from_string:Nn \c_stex_mathhub_seq {
349       \c_stex_pwd_str/\mathhub
350     }
351   }
352   \stex_path_to_string:NN\c_stex_mathhub_seq\c_stex_mathhub_str
353   \stex_debug:n {MathHub:~\str_use:N\c_stex_mathhub_str}
354 }

```

(End definition for `\mathhub`, `\c_stex_mathhub_seq`, and `\c_stex_mathhub_str`. These variables are documented on page 10.)

```

\__stex_mathhub_do_manifest:n
355 \cs_new_protected:Nn \__stex_mathhub_do_manifest:n {
356   \str_set:Nx \l_tmpa_str { #1 }
357   \prop_if_exist:cF {c_stex_mathhub_#1_manifest_prop} {
358     \prop_new:c { c_stex_mathhub_#1_manifest_prop }
359     \seq_set_split:NnV \l_tmpa_seq / \l_tmpa_str
360     \seq_concat:NNN \l_tmpa_seq \c_stex_mathhub_seq \l_tmpa_seq
361     \__stex_mathhub_find_manifest:N \l_tmpa_seq
362     \seq_if_empty:NTF \l__stex_mathhub_manifest_file_seq {

```

```

363     \msg_set:nnn{stex}{error/norepository}{
364       No~archive~#1~found~in~
365       \stex_path_to_string:N \c_stex_mathhub_str
366     }
367     \msg_error:nn{stex}{error/norepository}
368   } {
369     \exp_args:No \__stex_mathhub_parse_manifest:n { \l_tmpa_str }
370   }
371 }
372 }

```

(End definition for \\_\_stex\_mathhub\_do\_manifest:n.)

\l\_stex\_mathhub\_manifest\_file\_seq

```

373 \str_new:N\l__stex_mathhub_manifest_file_seq

```

(End definition for \l\_stex\_mathhub\_manifest\_file\_seq.)

\\_\_stex\_mathhub\_find\_manifest:N

Attempts to find the MANIFEST.MF in some file path and stores its path in \l\_\_stex\_mathhub\_manifest\_file\_seq:

```

374 \cs_new_protected:Nn \__stex_mathhub_find_manifest:N {
375   \seq_set_eq:NN\l_tmpa_seq #1
376   \bool_set_true:N\l_tmpa_bool
377   \bool_while_do:Nn \l_tmpa_bool {
378     \seq_if_empty:NTF \l_tmpa_seq {
379       \bool_set_false:N\l_tmpa_bool
380     }{
381       \file_if_exist:nTF{
382         \stex_path_to_string:N\l_tmpa_seq/MANIFEST.MF
383       }{
384         \seq_put_right:Nn\l_tmpa_seq{MANIFEST.MF}
385         \bool_set_false:N\l_tmpa_bool
386       }{
387         \file_if_exist:nTF{
388           \stex_path_to_string:N\l_tmpa_seq/META-INF/MANIFEST.MF
389         }{
390           \seq_put_right:Nn\l_tmpa_seq{META-INF}
391           \seq_put_right:Nn\l_tmpa_seq{MANIFEST.MF}
392           \bool_set_false:N\l_tmpa_bool
393         }{
394           \file_if_exist:nTF{
395             \stex_path_to_string:N\l_tmpa_seq/meta-inf/MANIFEST.MF
396           }{
397             \seq_put_right:Nn\l_tmpa_seq{meta-inf}
398             \seq_put_right:Nn\l_tmpa_seq{MANIFEST.MF}
399             \bool_set_false:N\l_tmpa_bool
400           }{
401             \seq_pop_right:NN\l_tmpa_seq\l_tmpa_tl
402           }
403         }
404       }
405     }
406   }
407   \seq_set_eq:NN\l__stex_mathhub_manifest_file_seq\l_tmpa_seq
408 }

```

(End definition for `\_stex_mathhub_find_manifest:N`.)

`\c_stex_mathhub_manifest_ior` File variable used for MANIFEST-files

409 `\ior_new:N \c__stex_mathhub_manifest_ior`

(End definition for `\c_stex_mathhub_manifest_ior`.)

`\_stex_mathhub_parse_manifest:n` Stores the entries in manifest file in the corresponding property list:

```

410 \cs_new_protected:Nn \_stex_mathhub_parse_manifest:n {
411   \seq_set_eq:NN \l_tmpa_seq \l__stex_mathhub_manifest_file_seq
412   \ior_open:Nn \c__stex_mathhub_manifest_ior {\stex_path_to_string:N \l_tmpa_seq}
413   \ior_map_inline:Nn \c__stex_mathhub_manifest_ior {
414     \str_set:Nn \l_tmpa_str {#1}
415     \exp_args:NNoo \seq_set_split:Nnn
416       \l_tmpb_seq \c_colon_str \l_tmpa_str
417     \seq_pop_left:NNTF \l_tmpb_seq \l_tmpa_tl {
418       \exp_args:NNe \str_set:Nn \l_tmpb_tl {
419         \exp_args:NNo \seq_use:Nn \l_tmpb_seq \c_colon_str
420       }
421       \exp_args:No \str_case:nnTF \l_tmpa_tl {
422         {id} {
423           \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
424             { id } \l_tmpb_tl
425         }
426         {narration-base} {
427           \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
428             { narr } \l_tmpb_tl
429         }
430         {source-base} {
431           \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
432             { ns } \l_tmpb_tl
433         }
434         {ns} {
435           \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
436             { ns } \l_tmpb_tl
437         }
438         {dependencies} {
439           \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
440             { deps } \l_tmpb_tl
441         }
442       }{}{}
443     }{}
444   }
445   \ior_close:N \c__stex_mathhub_manifest_ior
446 }

```

(End definition for `\_stex_mathhub_parse_manifest:n`.)

`\stex_set_current_repository:n`

```

447 \cs_new_protected:Nn \stex_set_current_repository:n {
448   \stex_require_repository:n { #1 }
449   \prop_set_eq:Nc \l_stex_current_repository_prop {
450     c_stex_mathhub_#1_manifest_prop
451   }
452 }

```

(End definition for `\stex_set_current_repository:n`. This function is documented on page 11.)

`\stex_require_repository:n`

```

453 \cs_new_protected:Nn \stex_require_repository:n {
454   \prop_if_exist:cF { c_stex_mathhub_#1_manifest_prop } {
455     \stex_debug:n{Opening~archive:~#1}
456     \__stex_mathhub_do_manifest:n { #1 }
457     \exp_args:Nx \stex_addtosms:n {
458       \prop_const_from_keyval:cn { c_stex_mathhub_#1_manifest_prop } {
459         id = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } { id },
460         ns = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } { ns },
461         narr = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } { narr },
462         deps = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } { deps }
463       }
464     }
465   }
466 }

```

(End definition for `\stex_require_repository:n`. This function is documented on page 11.)

`\l_stex_current_repository_prop`

Current MathHub repository

```

467 \prop_new:N \l_stex_current_repository_prop
468
469 \__stex_mathhub_find_manifest:N \c_stex_pwd_seq
470 \seq_if_empty:NTF \l__stex_mathhub_manifest_file_seq {
471   \stex_debug:n{Not~currently~in~a~MathHub~repository}
472 } {
473   \__stex_mathhub_parse_manifest:n { main }
474   \prop_get:NnN \c_stex_mathhub_main_manifest_prop {id}
475   \l_tmpa_str
476   \prop_set_eq:cN { c_stex_mathhub_#1_manifest_prop }
477   \c_stex_mathhub_main_manifest_prop
478   \exp_args:Nx \stex_set_current_repository:n { \l_tmpa_str }
479   \stex_debug:n{Current~repository:~
480     \prop_item:Nn \l_stex_current_repository_prop {id}
481   }
482 }

```

(End definition for `\l_stex_current_repository_prop`. This variable is documented on page 11.)

`\inputref`

```

483 \newif \ifinputref \inputreffalse
484
485 \cs_new_protected:Nn \inputref:nn {
486   \str_set:Nx \l_tmpa_str { #1 }
487   \str_set:Nx \l_tmpb_str { #2 }
488   \str_if_empty:NT \l_tmpa_str {
489     \prop_if_empty:NF \l_stex_current_repository_prop {
490       \prop_get:NnN \l_stex_current_repository_prop { id } \l_tmpa_str
491     }
492   }
493   \str_if_empty:NF \l_tmpa_str {
494     \stex_require_repository:n \l_tmpa_str
495   }

```

```

496 \str_set:Nx \l_tmpa_str { \c_stex_mathhub_str / \l_tmpa_str / source / \l_tmpb_str }
497 \ifinputref
498   \input{ \l_tmpa_str }
499 \else
500   \inputreftrue
501   \input{ \l_tmpa_str }
502   \inputreffalse
503 \fi
504 }
505 \NewDocumentCommand \inputref { 0{ } m }{
506   \inputref:nn{ #1 }{ #2 }
507 }

```

(End definition for `\inputref`. This function is documented on page ??.)

`\mhp`

```

508 \def \mhp #1 #2 {
509   \str_if_eq:nnTF{#1}{#2}{
510     \c_stex_mathhub_str /
511     \prop_item:Nn \l_stex_current_repository_prop { id }
512     / source / #2
513   }{
514     \c_stex_mathhub_str / #1 / source / #2
515   }
516 }

```

(End definition for `\mhp`. This function is documented on page ??.)

`\libinput`

```

517 \cs_new_protected:Npn \libinput #1 {
518   \prop_get:NnNF \l_stex_current_repository_prop {id} \l_tmpa_str {
519     \msg_set:nnn{stex}{error/norepository}{
520       \c_backslash_str libinput~needs~to~be~called~in~an~archive
521     }
522     \msg_error:nn{stex}{error/norepository}
523   }
524   \bool_set_false:N \l_tmpa_bool
525   \tl_clear:N \l_tmpa_tl
526   \seq_set_eq:NN \l_tmpa_seq \c_stex_mathhub_seq
527   \seq_set_split:NnV \l_tmpb_seq / \l_tmpa_str
528   \seq_pop_right:NN \l_tmpb_seq \l_tmpa_str
529   \seq_pop_left:NNT \l_tmpb_seq \l_tmpb_str {
530     \seq_put_right:No \l_tmpa_seq \l_tmpb_str
531     \IfFileExists{ \stex_path_to_string:N \l_tmpa_seq
532       / meta-inf / lib / #1.tex}{
533       \bool_set_true:N \l_tmpa_bool
534       \tl_put_right:Nx \l_tmpa_tl {
535         \exp_not:N \input { \stex_path_to_string:N \l_tmpa_seq
536           / meta-inf / lib / #1.tex}
537       }
538     }{}
539   }
540   \IfFileExists{ \stex_path_to_string:N \l_tmpa_seq
541     / \l_tmpa_str / lib / #1.tex
542   }{

```

```

543 \bool_set_true:N \l_tmpa_bool
544 \tl_put_right:Nx \l_tmpa_tl {
545   \exp_not:N \input { \stex_path_to_string:N \l_tmpa_seq
546     / \l_tmpa_str / lib / #1.tex}
547 }
548 }{}
549 \bool_if:NF \l_tmpa_bool {
550   \msg_set:nnn{stex}{error/nofile}{
551     \c_backslash_str libinput~no~file~#1.tex~found!
552   }
553   \msg_error:nn{stex}{error/nofile}
554 }
555 \scalatexBREAK
556 \l_tmpa_tl
557 }

```

(End definition for `\libinput`. This function is documented on page 11.)

## 4.5 Module System

```

558 <@@=stex_module>

```

`\l_stex_current_module_prop`

```

559 \prop_new:N \l_stex_current_module_prop

```

(End definition for `\l_stex_current_module_prop`. This variable is documented on page 12.)

`stex_if_in_module_p:`

`stex_if_in_module:TF`

```

560 \prg_new_conditional:Nnn \stex_if_in_module: {p, T, F, TF} {
561   \prop_if_empty:NTF \l_stex_current_module_prop
562   \prg_return_false: \prg_return_true:
563 }

```

(End definition for `stex_if_in_module:TF`. This function is documented on page 12.)

`stex_if_module_exists_p:n`

`stex_if_module_exists:nTF`

```

564 \prg_new_conditional:Nnn \stex_if_module_exists:n {p, T, F, TF} {
565   \prop_if_exist:cTF { c_stex_module_#1_prop }
566   \prg_return_true: \prg_return_false:
567 }

```

(End definition for `stex_if_module_exists:nTF`. This function is documented on page 12.)

`\stex_add_to_current_module:n`

`\STEXexport`

```

568 \cs_new_protected:Nn \stex_add_to_current_module:n {
569   \prop_get:NnN \l_stex_current_module_prop { content } \l_tmpa_tl
570   \tl_put_right:Nn \l_tmpa_tl { #1 }
571   \prop_put:Nno \l_stex_current_module_prop { content } { \l_tmpa_tl }
572 }
573 \NewDocumentCommand \STEXexport { m }{
574   \stex_smsmode_set_codes:
575   \stex_add_to_current_module:n { #1 }
576   #1
577 }

```

(End definition for `\stex_add_to_current_module:n` and `\STEXexport`. These functions are documented on page 12.)

`\stex_add_constant_to_current_module:n`

```
578 \cs_new_protected:Nn \stex_add_constant_to_current_module:n {
579   \str_set:Nx \l_tmpa_str { #1 }
580   \prop_get:NnN \l_stex_current_module_prop { constants } \l_tmpa_seq
581   \seq_put_right:No \l_tmpa_seq { \l_tmpa_str }
582   \prop_put:Nno \l_stex_current_module_prop { constants } \l_tmpa_seq
583 }
```

(End definition for `\stex_add_constant_to_current_module:n`. This function is documented on page 12.)

`\stex_add_import_to_current_module:n`

```
584 \cs_new_protected:Nn \stex_add_import_to_current_module:n {
585   \str_set:Nx \l_tmpa_str { #1 }
586   \prop_get:NnN \l_stex_current_module_prop { imports } \l_tmpa_seq
587   \seq_put_right:No \l_tmpa_seq { \l_tmpa_str }
588   \prop_put:Nno \l_stex_current_module_prop { imports } \l_tmpa_seq
589 }
```

(End definition for `\stex_add_import_to_current_module:n`. This function is documented on page 12.)

`\stex_modules_compute_namespace:nN` stores its return values in:

`\l_stex_modules_ns_str`

```
590 \str_new:N \l_stex_modules_ns_str

591 \cs_new_protected:Nn \stex_modules_compute_namespace:nN {
592   \str_set:Nx \l_tmpa_str { #1 }
593   \seq_set_eq:NN \l_tmpa_seq #2
594   % split off file extension
595   \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
596   \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
597   \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
598   \seq_put_right:No \l_tmpa_seq \l_tmpb_str
599
600   \bool_set_true:N \l_tmpa_bool
601   \bool_while_do:Nn \l_tmpa_bool {
602     \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
603     \exp_args:No \str_case:nnTF { \l_tmpb_str } {
604       {source} { \bool_set_false:N \l_tmpa_bool }
605     }{}{
606       \seq_if_empty:NT \l_tmpa_seq {
607         \bool_set_false:N \l_tmpa_bool
608       }
609     }
610   }
611
612   \seq_if_empty:NTF \l_tmpa_seq {
613     \str_set_eq:NN \l_stex_modules_ns_str \l_tmpa_str
614   }{
615     \str_set:Nx \l_stex_modules_ns_str {
616       \l_tmpa_str/\stex_path_to_string:N \l_tmpa_seq
```



```

617     }
618   }
619 }

```

(End definition for `\stex_modules_compute_namespace:nN` and `\l_stex_modules_ns_str`. These functions are documented on page 13.)

`\stex_modules_current_namespace:`

```

620 \cs_new_protected:Nn \stex_modules_current_namespace: {
621   \prop_get:NnNTF \l_stex_current_repository_prop { ns } \l_tmpa_str {
622     \stex_modules_compute_namespace:nN \l_tmpa_str \g_stex_currentfile_seq
623   }{
624     % split off file extension
625     \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
626     \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
627     \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
628     \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
629     \seq_put_right:No \l_tmpa_seq \l_tmpb_str
630     \str_set:Nx \l_stex_modules_ns_str {
631       file:/\stex_path_to_string:N \l_tmpa_seq
632     }
633   }
634 }

```

(End definition for `\stex_modules_current_namespace:.` This function is documented on page 13.)

#### 4.5.1 The module environment

`\l_stex_all_modules_seq` Stores all available modules

```

635 \seq_new:N \l_stex_all_modules_seq

```

(End definition for `\l_stex_all_modules_seq`. This variable is documented on page 14.)

`\STEXModule`

`\stex_invoke_module:n`

```

636 \NewDocumentCommand \STEXModule { m } {
637   \exp_args:NNx \str_set:Nn \l_tmpa_str { #1 }
638   \int_set:Nn \l_tmpa_int { \str_count:N \l_tmpa_str }
639   \tl_set:Nn \l_tmpa_tl {
640     \msg_set:nnn{stex}{error/unknownmodule}{
641       No~module~#1~found!
642     }
643     \msg_error:nn{stex}{error/unknownmodule}
644   }
645   \seq_map_inline:Nn \l_stex_all_modules_seq {
646     \str_set:Nn \l_tmpb_str { ##1 }
647     \str_if_eq:eeT { \l_tmpa_str } {
648       \str_range:Nnn \l_tmpb_str { -\l_tmpa_int } { -1 }
649     } {
650       \seq_map_break:n {
651         \tl_set:Nn \l_tmpa_tl {
652           \stex_invoke_module:n { ##1 }
653         }
654       }
655     }
656   }

```

```

657 \l_tmpa_tl
658 }
659
660 \cs_new_protected:Nn \stex_invoke_module:n {
661   \stex_debug:n{Invoking~module~#1}
662   \peek_charcode_remove:NTF ! {
663     \__stex_module_invoke_uri:nN { #1 }
664   } {
665     \peek_charcode_remove:NTF ? {
666       \__stex_module_invoke_symbol:nn { #1 }
667     } {
668       \msg_set:nnn{stex}{error/syntax}{
669         Syntax~error:~?~or~!~expected~after~
670         \c_backslash_str STEXModule{#1}
671       }
672       \msg_error:nn{stex}{error/syntax}
673     }
674   }
675 }
676
677 \cs_new_protected:Nn \__stex_module_invoke_uri:nN {
678   \str_set:Nn #2 { #1 }
679 }
680
681 \cs_new_protected:Nn \__stex_module_invoke_symbol:nn {
682   \stex_invoke_symbol:n{#1?#2}
683 }

```

(End definition for `\STEXModule` and `\stex_invoke_module:n`. These functions are documented on page 14.)

module module arguments:

```

684 \keys_define:nn { stex / module } {
685   title      .tl_set_x:N = \l_stex_module_title_str ,
686   ns         .tl_set_x:N = \l_stex_module_ns_str ,
687   lang       .tl_set_x:N = \l_stex_module_lang_str ,
688   sig        .tl_set_x:N = \l_stex_module_sig_str ,
689   creators   .tl_set_x:N = \l_stex_module_creators_str ,
690   contributors .tl_set_x:N = \l_stex_module_contributors_str ,
691   meta       .tl_set_x:N = \l_stex_module_meta_str
692 }
693
694 % module parameters here? In the body?
695
696 \cs_new_protected:Nn \__stex_module_args:n {
697   \str_clear:N \l_stex_module_title_str
698   \str_clear:N \l_stex_module_ns_str
699   \str_clear:N \l_stex_module_lang_str
700   \str_clear:N \l_stex_module_sig_str
701   \str_clear:N \l_stex_module_creators_str
702   \str_clear:N \l_stex_module_contributors_str
703   \str_clear:N \l_stex_module_meta_str
704   \keys_set:nn { stex / module } { #1 }
705   \exp_args:NNo \str_set:Nn \l_stex_module_title_str

```

```

706     \l_stex_module_title_str
707 \exp_args:NNo \str_set:Nn \l_stex_module_ns_str
708     \l_stex_module_ns_str
709 \exp_args:NNo \str_set:Nn \l_stex_module_lang_str
710     \l_stex_module_lang_str
711 \exp_args:NNo \str_set:Nn \l_stex_module_sig_str
712     \l_stex_module_sig_str
713 \exp_args:NNo \str_set:Nn \l_stex_module_meta_str
714     \l_stex_module_meta_str
715 \exp_args:NNo \str_set:Nn \l_stex_module_creators_str
716     \l_stex_module_creators_str
717 \exp_args:NNo \str_set:Nn \l_stex_module_contributors_str
718     \l_stex_module_contributors_str
719 }

```

`\_stex_module_begin_module:` implements `\begin{module}`

```

720 \cs_new_protected:Nn \_stex_module_begin_module: {
721   % Nested module?
722   \stex_if_in_module:TF {
723     % Nested module
724     \prop_get:NnN \l_stex_current_module_prop
725       { ns } \l_stex_module_ns_str
726     \str_set:Nx \l_stex_module_name_str {
727       \prop_item:Nn \l_stex_current_module_prop
728         { name } / \l_stex_module_name_str
729     }
730   }{
731     % not nested:
732     \str_if_empty:NT \l_stex_module_ns_str {
733       \stex_modules_current_namespace:
734       \str_set_eq:NN \l_stex_module_ns_str \l_stex_modules_ns_str
735       \exp_args:NNNo \seq_set_split:Nnn \l_tmpa_seq
736         / {\l_stex_module_ns_str}
737       \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
738       \str_if_eq:NNT \l_tmpa_str \l_stex_module_name_str {
739         \str_set:Nx \l_stex_module_ns_str {
740           \stex_path_to_string:N \l_tmpa_seq
741         }
742       }
743     }
744   }
745
746   % language
747   \str_if_empty:NT \l_stex_module_lang_str {
748     \seq_get_right:NN \g_stex_currentfile_seq \l_tmpa_str
749     \seq_set_split:NnV \l_tmpa_seq . \l_tmpa_str
750     \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str % .tex
751     \seq_pop_left:NN \l_tmpa_seq \l_tmpa_str % <filename>
752     \seq_if_empty:NF \l_tmpa_seq { %remaining element should be language
753       \stex_debug:n {Language~\l_stex_module_lang_str~
754         inferred~from~file~name}
755       \seq_pop_left:NN \l_tmpa_seq \l_stex_module_lang_str
756     }
757   }

```

```

758
759 \str_if_empty:NF \l_stex_module_lang_str {
760   \prop_get:NVNTF \c_stex_languages_prop \l_stex_module_lang_str
761   \l_tmpa_str {
762     \ltx@ifpackageloaded{babel}{
763       \exp_args:Nx \selectlanguage { \l_tmpa_str }
764     }{}
765   } {
766     \msg_set:nnn{stex}{error/unknownlanguage}{
767       Unknown~language~\l_tmpa_str
768     }
769     \msg_error:nn{stex}{error/unknownlanguage}
770   }
771 }
772
773 % signature
774 \str_if_empty:NTF \l_stex_module_sig_str {
775   \str_clear:N \l_tmpa_str
776   \seq_clear:N \l_tmpa_seq
777   \tl_clear:N \l_tmpa_tl
778   \exp_args:NNx \prop_set_from_keyval:Nn \l_stex_current_module_prop {
779     name      = \l_stex_module_name_str ,
780     ns        = \l_stex_module_ns_str ,
781     imports   = \exp_not:o { \l_tmpa_seq } ,
782     constants = \exp_not:o { \l_tmpa_seq } ,
783     content   = \exp_not:o { \l_tmpa_tl } ,
784     file      = \exp_not:o { \g_stex_currentfile_seq } ,
785     lang      = \l_stex_module_lang_str ,
786     sig       = \l_stex_module_sig_str ,
787     meta      = \l_stex_module_meta_str
788   }
789 }{
790   \str_if_empty:NT \l_stex_module_lang_str {
791     \msg_set:nnn{stex}{error/siglanguage}{
792       Module~\l_stex_module_ns_str?\l_stex_module_name_str~
793       declares~signature~\l_stex_module_sig_str,~but~does~not~
794       declare~its~language
795     }
796     \msg_error:nn{stex}{error/siglanguage}
797   }
798
799   \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
800   \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
801   \seq_set_split:NnV \l_tmpb_seq . \l_tmpa_str
802   \seq_pop_right:NN \l_tmpb_seq \l_tmpa_str % .tex
803   \seq_pop_left:NN \l_tmpb_seq \l_tmpa_str % <filename>
804   \str_set:Nx \l_tmpa_str {
805     \stex_path_to_string:N \l_tmpa_seq /
806     \l_tmpa_str . \l_stex_module_sig_str .tex
807   }
808   \IfFileExists \l_tmpa_str {
809     \exp_args:No \stex_in_smsmode:nn { \l_tmpa_str } {
810       \seq_clear:N \l_stex_all_modules_seq
811       \prop_clear:N \l_stex_current_module_prop

```

```

812     \stex_debug:n{Loading~signature~\l_tmpa_str}
813     \input { \l_tmpa_str }
814   }
815   ){
816     \msg_set:nnn{stex}{error/modulemissing}{
817       No~file~for~signature~module~\l_tmpa_str~found
818     }
819     \msg_error:nn{stex}{error/modulemissing}
820   }
821   \stex_activate_module:n {
822     \l_stex_module_ns_str ? \l_stex_module_name_str
823   }
824   \prop_set_eq:Nc \l_stex_current_module_prop {
825     c_stex_module_
826     \l_stex_module_ns_str ?
827     \l_stex_module_name_str
828     _prop
829   }
830 }
831
832 % metatheory
833 \str_if_empty:NT \l_stex_module_meta_str {
834   \str_set:Nx \l_stex_module_meta_str {
835     \c_stex_metatheory_ns_str ? Metatheory
836   }
837 }
838
839
840 \stex_debug:n{
841   New~module:\\
842   Namespace:~\l_stex_module_ns_str\\
843   Name:~\l_stex_module_name_str\\
844   Language:~\l_stex_module_lang_str\\
845   Signature:~\l_stex_module_sig_str\\
846   Metatheory:~\l_stex_module_meta_str\\
847   File:~\stex_path_to_string:N \g_stex_currentfile_seq
848 }
849
850 \seq_put_right:Nx \l_stex_all_modules_seq {
851   \l_stex_module_ns_str ? \l_stex_module_name_str
852 }
853
854 \seq_gput_right:Nx \g_stex_modules_in_file_seq
855   { \l_stex_module_ns_str ? \l_stex_module_name_str }
856
857 \stex_if_smsmode:TF {
858   \stex_smsmode_set_codes:
859 } {
860   \begin{stex_annotate_env} {theory} {
861     \l_stex_module_ns_str ? \l_stex_module_name_str
862   }
863
864   \stex_annotate_invisible:nnn{header}{} {
865     \stex_annotate:nnn{language}{ \l_stex_module_lang_str }{}

```

```

866     \stex_annotate:nnn{signature}{ \l_stex_module_sig_str }{}
867     \str_if_eq:VnF \l_stex_module_meta_str {NONE} {
868         \stex_annotate:nnn{metatheory}{ \l_stex_module_meta_str }{}
869     }
870 }
871 }
872
873 \str_if_eq:VnF \l_stex_module_meta_str {NONE} {
874     \exp_args:Nx \STEXexport{
875         \stex_activate_module:n { \l_stex_module_meta_str }
876     }
877 }
878 % TODO: Inherit metatheory for nested modules?
879 }
880 \iffalse \end{stex_annotate_env} \fi % make syntax highlighting work again

```

(End definition for \\_stex\_module\_begin\_module:.)

\\_stex\_module\_end\_module: implements \end{module}

```

881 \iffalse \begin{stex_annotate_env} \fi %^^A make syntax highlighting work again
882 \cs_new_protected:Nn \_stex_module_end_module: {
883     \str_set:Nx \l_tmpa_str {
884         c_stex_module_
885         \prop_item:Nn \l_stex_current_module_prop { ns } ?
886         \prop_item:Nn \l_stex_current_module_prop { name }
887         _prop
888     }
889     %^^A \prop_new:c { \l_tmpa_str }
890     \prop_gset_eq:cn { \l_tmpa_str } \l_stex_current_module_prop
891     \stex_debug:n{Closing module~\prop_item:Nn \l_stex_current_module_prop { name }}
892     \stex_if_smsmode:TF {
893         \exp_args:Nx \stex_addtosms:n {
894             \prop_gset_from_keyval:cn {
895                 c_stex_module_
896                 \prop_item:Nn \l_stex_current_module_prop { ns } ?
897                 \prop_item:Nn \l_stex_current_module_prop { name }
898                 _prop
899             } {
900                 name      = \prop_item:cn { \l_tmpa_str } { name } ,
901                 ns        = \prop_item:cn { \l_tmpa_str } { ns } ,
902                 imports   = \prop_item:cn { \l_tmpa_str } { imports } ,
903                 constants = \prop_item:cn { \l_tmpa_str } { constants } ,
904                 content   = \prop_item:cn { \l_tmpa_str } { content } ,
905                 file      = \prop_item:cn { \l_tmpa_str } { file } ,
906                 lang      = \prop_item:cn { \l_tmpa_str } { lang } ,
907                 sig       = \prop_item:cn { \l_tmpa_str } { sig } ,
908                 meta      = \prop_item:cn { \l_tmpa_str } { meta }
909             }
910         }
911     }{
912         \end{stex_annotate_env}
913     }
914 }

```

(End definition for \\_stex\_module\_end\_module:.)

`@module` The core environment, with no header

```

915 \NewDocumentEnvironment { @module } { 0{} m } {
916   \str_set:Nx \l_stex_module_name_str { #2 }
917   \par
918   \__stex_module_args:n { #1 }
919   \__stex_module_begin_module:
920 } {
921   \__stex_module_end_module:
922 }

```

`\stex_modules_heading:` Code for document headers

```

923 \cs_if_exist:NTF \thesection {
924   \newcounter{module}[section]
925 }{
926   \newcounter{module}
927 }
928
929 \bool_if:NT \c_stex_showmods_bool {
930   \latexml_if:F { \RequirePackage{mdframed} }
931 }
932
933 \cs_new_protected:Nn \stex_modules_heading: {
934   \stepcounter{module}
935   \par
936   \bool_if:NT \c_stex_showmods_bool {
937     \noindent{\textbf{Module} ~
938       \cs_if_exist:NT \thesection {\thesection.}
939       \themodule ~ [\l_stex_module_name_str]
940     }
941     % TODO references
942     % \sref@label@id{Module \thesection.\themodule [\module@name]]%
943     \str_if_empty:NTF \l_stex_module_title_str {
944       }{
945         \quad(\l_stex_module_title_str)\hfill
946       }\par
947     }
948   }

```

(End definition for `\stex_modules_heading:`. This function is documented on page 13.)

Finally:

```

949 \NewDocumentEnvironment { module } { 0{} m } {
950   \bool_if:NT \c_stex_showmods_bool {
951     \begin{mdframed}
952   }
953   \begin{@module}[#1]{#2}
954   \stex_modules_heading:
955 }{
956   \end{@module}
957   \bool_if:NT \c_stex_showmods_bool {
958     \end{mdframed}
959   }
960 }

```

## 4.5.2 SMS Mode

961  $\langle @@=\text{stex\_smsmode} \rangle$

$\backslash\text{g\_stex\_smsmode\_allowedmacros\_tl}$   
 $\backslash\text{g\_stex\_smsmode\_allowedmacros\_escape\_tl}$   
 $\backslash\text{g\_stex\_smsmode\_allowedenvs\_seq}$

```

962 \tl_new:N \g_stex_smsmode_allowedmacros_tl
963 \tl_new:N \g_stex_smsmode_allowedmacros_escape_tl
964 \seq_new:N \g_stex_smsmode_allowedenvs_seq
965
966 \tl_set:Nn \g_stex_smsmode_allowedmacros_tl {
967   \makeatletter
968   \makeatother
969   \ExplSyntaxOn
970   \ExplSyntaxOff
971 }
972
973 \tl_set:Nn \g_stex_smsmode_allowedmacros_escape_tl {
974   \symdef
975   \importmodule
976   \notation
977   \symdecl
978   \STEXexport
979 }
980
981 \exp_args:NNx \seq_set_from_clist:Nn \g_stex_smsmode_allowedenvs_seq {
982   \tl_to_str:n {
983     module,
984     @module
985   }
986 }
```

(End definition for  $\backslash\text{g\_stex\_smsmode\_allowedmacros\_tl}$ ,  $\backslash\text{g\_stex\_smsmode\_allowedmacros\_escape\_tl}$ , and  $\backslash\text{g\_stex\_smsmode\_allowedenvs\_seq}$ . These variables are documented on page 15.)

$\backslash\text{stex\_if\_smsmode\_p}$ :  
 $\backslash\text{stex\_if\_smsmode}$ :  $\underline{TF}$

```

987 \bool_new:N \g__stex_smsmode_bool
988 \bool_set_false:N \g__stex_smsmode_bool
989 \prg_new_conditional:Nnn \stex_if_smsmode: { p, T, F, TF } {
990   \bool_if:NTF \g__stex_smsmode_bool \prg_return_true: \prg_return_false:
991 }
```

(End definition for  $\backslash\text{stex\_if\_smsmode}$ : $\underline{TF}$ . This function is documented on page 16.)

$\backslash\text{stex\_smsmode\_if\_catcodes\_p}$ :  
 $\backslash\text{stex\_smsmode\_if\_catcodes}$ :  $\underline{TF}$

Checks whether the SMS mode category code scheme is active.

```

992 \bool_new:N \g__stex_smsmode_catcode_bool
993 \bool_set_false:N \g__stex_smsmode_catcode_bool
994 \prg_new_conditional:Nnn \stex_smsmode_if_catcodes: { p, T, F, TF } {
995   \bool_if:NTF \g__stex_smsmode_catcode_bool
996     \prg_return_true: \prg_return_false:
997 }
```

(End definition for  $\backslash\text{stex\_smsmode\_if\_catcodes}$ : $\underline{TF}$ .)



`\stex_smsmode_set_codes:`

```

998 \cs_new_protected:Nn \stex_smsmode_set_codes: {
999   \stex_if_smsmode:T {
1000     \__stex_smsmode_if_catcodes:F {
1001       \bool_gset_true:N \g__stex_smsmode_catcode_bool
1002       \exp_after:wN \char_gset_active_eq:NN
1003       \c_backslash_str \__stex_smsmode_cs:
1004       \tex_global:D \char_set_catcode_active:N \
1005       \tex_global:D \char_set_catcode_other:N $
1006       \tex_global:D \char_set_catcode_other:N ^
1007       \tex_global:D \char_set_catcode_other:N _
1008       \tex_global:D \char_set_catcode_other:N &
1009       \tex_global:D \char_set_catcode_other:N ##
1010     }
1011   }
1012 } \iffalse $ \fi % to make syntax highlighting work again

```

(End definition for `\stex_smsmode_set_codes:`. This function is documented on page 16.)

`\__stex_smsmode_unset_codes:` Sets category code scheme back from the one used in SMS mode.

```

1013 \cs_new_protected:Nn \__stex_smsmode_unset_codes: {
1014   \__stex_smsmode_if_catcodes:T {
1015     \bool_gset_false:N \g__stex_smsmode_catcode_bool
1016     \exp_after:wN \tex_global:D \exp_after:wN
1017     \char_set_catcode_escape:N \c_backslash_str
1018     \tex_global:D \char_set_catcode_math_toggle:N $
1019     \tex_global:D \char_set_catcode_math_superscript:N ^
1020     \tex_global:D \char_set_catcode_math_subscript:N _
1021     \tex_global:D \char_set_catcode_alignment:N &
1022     \tex_global:D \char_set_catcode_parameter:N ##
1023   }
1024 } \iffalse $ \fi % to make syntax highlighting work again

```

(End definition for `\__stex_smsmode_unset_codes:`.)

`\stex_in_smsmode:nn`

```

1025 \cs_new_protected:Nn \stex_in_smsmode:nn {
1026   \vbox_set:Nn \l_tmpa_box {
1027     \bool_set_eq:cN { l__stex_smsmode_#1_bool } \g__stex_smsmode_bool
1028     \bool_gset_true:N \g__stex_smsmode_bool
1029     \stex_smsmode_set_codes:
1030     #2
1031     \bool_gset_eq:Nc \g__stex_smsmode_bool { l__stex_smsmode_#1_bool }
1032     \stex_if_smsmode:F {
1033       \__stex_smsmode_unset_codes:
1034     }
1035   }
1036   \box_clear:N \l_tmpa_box
1037 }

```

(End definition for `\stex_in_smsmode:nn`. This function is documented on page 16.)

`\__stex_smsmode_cs:` is executed on encountering `\` in smsmode. It checks whether the corresponding command is allowed and executes or ignores it accordingly:

```

1038 \cs_new_protected:Nn \__stex_smsmode_cs: {
1039   \str_clear:N \l_tmpa_str
1040   \peek_analysis_map_inline:n {
1041     % #1: token (one expansion)
1042     % #2: charcode
1043     % #3 catcode
1044     \token_if_eq_charcode:NNTF ##3 B {
1045       % token is a letter
1046       \exp_args:NNo \str_put_right:Nn \l_tmpa_str { ##1 }
1047     } {
1048       \str_if_empty:NNTF \l_tmpa_str {
1049         % we don't allow (or need) single non-letter CSs
1050         % for now
1051         \peek_analysis_map_break:
1052       } {
1053         \str_if_eq:ontf \l_tmpa_str { begin } {
1054           \peek_analysis_map_break:n {
1055             \exp_after:wN \__stex_smsmode_checkbegin:n ##1
1056           }
1057         } {
1058           \str_if_eq:ontf \l_tmpa_str { end } {
1059             \peek_analysis_map_break:n {
1060               \exp_after:wN \__stex_smsmode_checkend:n ##1
1061             }
1062           } {
1063             \tl_set:Nn \l_tmpa_tl { \use:c{\l_tmpa_str} }
1064             \exp_args:NNNo \exp_args:NNo \tl_if_in:NnTF
1065               \g_stex_smsmode_allowedmacros_tl
1066               { \use:c{\l_tmpa_str} } {
1067               \stex_debug:n{Executing~1:~\l_tmpa_str}
1068               \peek_analysis_map_break:n {
1069                 \exp_after:wN \l_tmpa_tl ##1
1070               }
1071             } {
1072               \exp_args:NNNo \exp_args:NNo \tl_if_in:NnTF
1073               \g_stex_smsmode_allowedmacros_escape_tl
1074               { \use:c{\l_tmpa_str} } {
1075               \stex_debug:n{Executing~2:~\l_tmpa_str}
1076               % TODO \__stex_smsmode_rescan_cs:
1077               \exp_after:wN \exp_after:wN \exp_after:wN
1078               \token_if_eq_charcode:NNTF \exp_after:wN \c_backslash_str ##1 {
1079               \peek_analysis_map_break:n {
1080                 \__stex_smsmode_unset_codes:
1081                 \__stex_smsmode_rescan_cs:
1082               }
1083             } {
1084               \peek_analysis_map_break:n {
1085                 \__stex_smsmode_unset_codes:
1086                 \exp_after:wN \l_tmpa_tl ##1
1087               }
1088             }
1089           } {
1090             \peek_analysis_map_break:n { ##1 }
1091           }

```

```

1092     }
1093   }
1094 }
1095 }
1096 }
1097 }
1098 }

```

(End definition for `\_stex_smsmode_cs:`.)

`\_stex_smsmode_rescan_cs:` If the last token gobbled by `\stex_smsmode_cs:` happened to be a `\`, we need to rescan the cs name and reinsert it into the input stream:

```

1099 \cs_new_protected:Nn \_stex_smsmode_rescan_cs: {
1100   \str_clear:N \l_tmpb_str
1101   \peek_analysis_map_inline:n {
1102     \token_if_eq_charcode:NNTF ##3 B {
1103       % token is a letter
1104       \exp_args:NNo \str_put_right:Nn \l_tmpb_str { ##1 }
1105     } {
1106       \peek_analysis_map_break:n {
1107         \exp_after:wN \use:c \exp_after:wN {
1108           \exp_after:wN \l_tmpa_str\exp_after:wN
1109         } \use:c { \l_tmpb_str \exp_after:wN } ##1
1110       }
1111     }
1112   }
1113 }

```

(End definition for `\_stex_smsmode_rescan_cs:`.)

`\_stex_smsmode_checkbegin:n` called on `\begin`; checks whether the environment being opened is allowed in SMS mode.

```

1114 \cs_new_protected:Nn \_stex_smsmode_checkbegin:n {
1115   \str_set:Nn \l_tmpa_str { #1 }
1116   \seq_if_in:NoT \g_stex_smsmode_allowedenvs_seq \l_tmpa_str {
1117     \_stex_smsmode_unset_codes:
1118     \begin{#1}
1119   }
1120 }

```

(End definition for `\_stex_smsmode_checkbegin:n`.)

`\_stex_smsmode_checkend:n` called on `\end`; checks whether the environment being opened is allowed in SMS mode.

```

1121 \cs_new_protected:Nn \_stex_smsmode_checkend:n {
1122   \str_set:Nn \l_tmpa_str { #1 }
1123   \seq_if_in:NoT \g_stex_smsmode_allowedenvs_seq \l_tmpa_str {
1124     \end{#1}
1125   }
1126 }

```

(End definition for `\_stex_smsmode_checkend:n`.)

### 4.5.3 Inheritance

1127 <@@=stex\_importmodule>

\stex\_import\_module\_uri:nn

```

1128 \cs_new_protected:Nn \stex_import_module_uri:nn {
1129   \str_set:Nx \l__stex_importmodule_archive_str { #1 }
1130   \str_set:Nx \l__stex_importmodule_path_str { #2 }
1131   \str_if_empty:NT \l__stex_importmodule_archive_str {
1132     \prop_if_empty:NF \l_stex_current_repository_prop {
1133       \prop_get:NnN \l_stex_current_repository_prop { id } \l__stex_importmodule_archive_str
1134     }
1135   }
1136
1137   \exp_args:NNo \seq_set_split:Nnn \l_tmpb_seq ? { \l__stex_importmodule_path_str }
1138   \seq_pop_right:NN \l_tmpb_seq \l__stex_importmodule_name_str
1139   \str_set:Nx \l__stex_importmodule_path_str { \seq_use:Nn \l_tmpb_seq ? }
1140
1141   \str_if_empty:NTF \l__stex_importmodule_archive_str {
1142     \stex_modules_current_namespace:
1143     \str_if_empty:NF \l__stex_importmodule_path_str {
1144       \str_set:Nx \l_stex_module_ns_str {
1145         \l_stex_module_ns_str / \l__stex_importmodule_path_str
1146       }
1147     }
1148   }{
1149     \stex_require_repository:n \l__stex_importmodule_archive_str
1150     \prop_get:cnN { c_stex_mathhub\l__stex_importmodule_archive_str _manifest_prop } { ns }
1151     \l_stex_module_ns_str
1152     \str_if_empty:NF \l__stex_importmodule_path_str {
1153       \str_set:Nx \l_stex_module_ns_str {
1154         \l_stex_module_ns_str / \l__stex_importmodule_path_str
1155       }
1156     }
1157   }
1158 }

```

(End definition for \stex\_import\_module\_uri:nn. This function is documented on page 19.)

\l\_stex\_importmodule\_name\_str  
\l\_stex\_importmodule\_archive\_str  
\l\_stex\_importmodule\_path\_str  
\l\_stex\_importmodule\_file\_str

Store the return values of \stex\_import\_module\_uri:nn.

```

1159 \str_new:N \l__stex_importmodule_name_str
1160 \str_new:N \l__stex_importmodule_archive_str
1161 \str_new:N \l__stex_importmodule_path_str
1162 \str_new:N \g__stex_importmodule_file_str

```

(End definition for \l\_\_stex\_importmodule\_name\_str and others.)

\stex\_import\_require\_module:nnnn

{<ns>} {<archive-ID>} {<path>} {<name>}

```

1163 \cs_new_protected:Nn \stex_import_require_module:nnnn {
1164   \exp_args:Nx \stex_if_module_exists:nF { #1 ? #4 } {
1165     % \stex_debug:n{Arguments: #1, #2, #3, #4}
1166
1167     % archive
1168     \str_set:Nx \l_tmpa_str { #2 }
1169     \str_if_empty:NTF \l_tmpa_str {

```

```

1170 \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
1171 } {
1172 \stex_path_from_string:Nn \l_tmpb_seq { \l_tmpa_str }
1173 \seq_concat:NNN \l_tmpa_seq \c_stex_mathhub_seq \l_tmpb_seq
1174 \seq_put_right:Nn \l_tmpa_seq { source }
1175 }
1176
1177 % path
1178 \str_set:Nx \l_tmpb_str { #3 }
1179 \str_if_empty:NTF \l_tmpb_str {
1180 \str_set:Nx \l_tmpa_str { \stex_path_to_string:N \l_tmpa_seq / #4 }
1181
1182 \ltx@ifpackageloaded{babel} {
1183 \exp_args:NNx \prop_get:NnNF \c_stex_language_abbrevs_prop
1184 { \language } \l_tmpb_str {
1185 \msg_set:nnn{stex}{error/unknownlanguage}{
1186 Unknown-language-\language
1187 }
1188 \msg_error:nn{stex}{error/unknownlanguage}
1189 }
1190 } {
1191 \str_clear:N \l_tmpb_str
1192 }
1193
1194 \stex_debug:n{Checking-\l_tmpa_str.\l_tmpb_str.tex}
1195 \IfFileExists{ \l_tmpa_str.\l_tmpb_str.tex }{
1196 \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.\l_tmpb_str.tex }
1197 }{
1198 \stex_debug:n{Checking-\l_tmpa_str.tex}
1199 \IfFileExists{ \l_tmpa_str.tex }{
1200 \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.tex }
1201 }{
1202 % try english as default
1203 \stex_debug:n{Checking-\l_tmpa_str.en.tex}
1204 \IfFileExists{ \l_tmpa_str.en.tex }{
1205 \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.en.tex }
1206 }{
1207 \msg_set:nnn{stex}{error/modulemissing}{
1208 No-file-for-module-#1?#4-found
1209 }
1210 \msg_error:nn{stex}{error/modulemissing}
1211 }
1212 }
1213 }
1214
1215 } {
1216 \seq_set_split:NnV \l_tmpb_seq / \l_tmpb_str
1217 \seq_concat:NNN \l_tmpa_seq \l_tmpa_seq \l_tmpb_seq
1218
1219 \ltx@ifpackageloaded{babel} {
1220 \exp_args:NNx \prop_get:NnNF \c_stex_language_abbrevs_prop
1221 { \language } \l_tmpb_str {
1222 \msg_set:nnn{stex}{error/unknownlanguage}{
1223 Unknown-language-\language

```

```

1224     }
1225     \msg_error:nn{stex}{error/unknownlanguage}
1226   }
1227 } {
1228   \str_clear:N \l_tmpb_str
1229 }
1230
1231 \stex_path_to_string:NN \l_tmpa_seq \l_tmpa_str
1232
1233 \stex_debug:n{Checking~\l_tmpa_str/#4.\l_tmpb_str.tex}
1234 \IfFileExists{ \l_tmpa_str/#4.\l_tmpb_str.tex }{
1235   \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.\l_tmpb_str.tex }
1236 }{
1237   \stex_debug:n{Checking~\l_tmpa_str/#4.tex}
1238   \IfFileExists{ \l_tmpa_str/#4.tex }{
1239     \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.tex }
1240   }{
1241     % try english as default
1242     \stex_debug:n{Checking~\l_tmpa_str/#4.en.tex}
1243     \IfFileExists{ \l_tmpa_str/#4.en.tex }{
1244       \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.en.tex }
1245     }{
1246       \stex_debug:n{Checking~\l_tmpa_str.\l_tmpb_str.tex}
1247       \IfFileExists{ \l_tmpa_str.\l_tmpb_str.tex }{
1248         \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.\l_tmpb_str.tex }
1249       }{
1250         \stex_debug:n{Checking~\l_tmpa_str.tex}
1251         \IfFileExists{ \l_tmpa_str.tex }{
1252           \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.tex }
1253         }{
1254           % try english as default
1255           \stex_debug:n{Checking~\l_tmpa_str.en.tex}
1256           \IfFileExists{ \l_tmpa_str.en.tex }{
1257             \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.en.tex }
1258           }{
1259             \msg_set:nnn{stex}{error/modulemissing}{
1260               No~file~for~module~#1?#4~found
1261             }
1262             \msg_error:nn{stex}{error/modulemissing}
1263           }
1264         }
1265       }
1266     }
1267   }
1268 }
1269 }
1270
1271 \seq_set_eq:NN \l_tmpa_seq \g_stex_modules_in_file_seq
1272 \seq_clear:N \g_stex_modules_in_file_seq
1273 % \exp_args:Nnx \use:nn {
1274   \exp_args:No \stex_in_smsmode:nn { \g__stex_importmodule_file_str } {
1275     \seq_clear:N \l_stex_all_modules_seq
1276     \prop_clear:N \l_stex_current_module_prop
1277     \str_set:Nx \l_tmpb_str { #2 }

```

```

1278     \str_if_empty:NF \l_tmpb_str {
1279         \stex_set_current_repository:n { #2 }
1280     }
1281     \stex_debug:n{Loading~\g__stex_importmodule_file_str}
1282     \input { \g__stex_importmodule_file_str }
1283 }
1284 % }{
1285
1286 % }
1287 \prop_gput:Noo \g_stex_module_files_prop
1288 \g__stex_importmodule_file_str \g_stex_modules_in_file_seq
1289 \seq_set_eq:NN \g_stex_modules_in_file_seq \l_tmpa_seq
1290
1291 \stex_if_module_exists:nF { #1 ? #4 } {
1292     \msg_set:nnn{stex}{error/modulemissing}{
1293         Module~#1?#4~not~found~in~file~\g__stex_importmodule_file_str
1294     }
1295     \msg_error:nn{stex}{error/modulemissing}
1296 }
1297 }
1298 \stex_activate_module:n { #1 ? #4 }
1299 }

```

(End definition for `\stex_import_require_module:nnnn`. This function is documented on page 19.)

`\stex_activate_module:n`

```

1300 \cs_new_protected:Nn \stex_activate_module:n {
1301     \stex_debug:n{Activating~module~#1}
1302     \exp_args:NNx \seq_if_in:NnF \l_stex_all_modules_seq { #1 } {
1303         \seq_put_right:Nx \l_stex_all_modules_seq { #1 }
1304         \prop_item:cn { c_stex_module_#1_prop } { content }
1305     }
1306 }

```

(End definition for `\stex_activate_module:n`. This function is documented on page 19.)

`\importmodule`

```

1307 \NewDocumentCommand \importmodule { 0{} m } {
1308     \stex_import_module_uri:nn { #1 } { #2 }
1309     \stex_debug:n{Importing~module:~
1310         \l_stex_module_ns_str ? \l__stex_importmodule_name_str
1311     }
1312     \stex_if_smsmode:F {
1313         \stex_import_require_module:nnnn
1314         { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
1315         { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
1316         \stex_annotate_invisible:nnn
1317         {import} { \l_stex_module_ns_str ? \l__stex_importmodule_name_str } {}
1318     }
1319     \exp_args:Nx \stex_add_to_current_module:n {
1320         \stex_import_require_module:nnnn
1321         { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
1322         { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
1323     }
1324     \exp_args:Nx \stex_add_import_to_current_module:n {

```

```

1325 \l_stex_module_ns_str ? \l__stex_importmodule_name_str
1326 }
1327 \stex_smsmode_set_codes:
1328 }

```

(End definition for \importmodule. This function is documented on page 16.)

#### \usemodule

```

1329 \NewDocumentCommand \usemodule { 0{} m } {
1330 \stex_if_smsmode:F {
1331 \stex_import_module_uri:nn { #1 } { #2 }
1332 \stex_import_require_module:nnnn
1333 { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
1334 { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
1335 \stex_annotate_invisible:nnn
1336 {usemodule} {\l_stex_module_ns_str ? \l__stex_importmodule_name_str} {}
1337 }
1338 \stex_smsmode_set_codes:
1339 }

```

(End definition for \usemodule. This function is documented on page 17.)

#### \g\_stex\_modules\_in\_file\_seq \g\_stex\_module\_files\_prop

```

1340 \seq_new:N \g_stex_modules_in_file_seq
1341 \prop_new:N \g_stex_module_files_prop

```

(End definition for \g\_stex\_modules\_in\_file\_seq and \g\_stex\_module\_files\_prop. These variables are documented on page 19.)

## 4.6 Symbol Declarations

```

1342 <@@=stex_symdecl>

```

#### \l\_stex\_all\_symbols\_seq Stores all available symbols

```

1343 \seq_new:N \l_stex_all_symbols_seq

```

(End definition for \l\_stex\_all\_symbols\_seq. This variable is documented on page 21.)

#### \STEXsymbol

```

1344 \NewDocumentCommand \STEXsymbol { m } {
1345 \stex_get_symbol:n { #1 }
1346 \exp_args:No
1347 \stex_invoke_symbol:n { \l_stex_get_symbol_uri_str }
1348 }

```

(End definition for \STEXsymbol. This function is documented on page 21.)

symdecl arguments:

```

1349 \keys_define:nn { stex / symdecl } {
1350 name .tl_set:x:N = \l_stex_symdecl_name_str ,
1351 local .bool_set:N = \l_stex_symdecl_local_bool ,
1352 args .tl_set:x:N = \l_stex_symdecl_args_str ,
1353 type .tl_set:N = \l_stex_symdecl_type_tl ,
1354 align .tl_set:N = \l_stex_symdecl_align_str , % TODO(?)
1355 gfc .tl_set:N = \l_stex_symdecl_gfc_str , % TODO(?)
1356 specializes .tl_set:N = \l_stex_symdecl_specializes_str , % TODO(?)

```



```

1357   def          .tl_set:N      = \l_stex_symdecl_definiens_tl
1358 }
1359
1360 \bool_new:N \l_stex_symdecl_make_macro_bool
1361
1362 \cs_new_protected:Nn \__stex_symdecl_args:n {
1363   \str_clear:N \l_stex_symdecl_name_str
1364   \str_clear:N \l_stex_symdecl_args_str
1365   \bool_set_false:N \l_stex_symdecl_local_bool
1366   \tl_clear:N \l_stex_symdecl_type_tl
1367   \tl_clear:N \l_stex_symdecl_definiens_tl
1368
1369   \keys_set:nn { stex /symdecl } { #1 }
1370
1371   \exp_args:NNo \str_set:Nn \l_stex_symdecl_name_str
1372     \l_stex_symdecl_name_str
1373   \exp_args:NNo \str_set:Nn \l_stex_symdecl_args_str
1374     \l_stex_symdecl_args_str
1375 }

```

**\symdecl** Parses the optional arguments and passes them on to `\stex_symdecl_do:` (so that `\symdef` can do the same)

```

1376
1377 \NewDocumentCommand \symdecl { s O{} m } {
1378   \__stex_symdecl_args:n { #2 }
1379   \IfBooleanTF #1 {
1380     \bool_set_false:N \l_stex_symdecl_make_macro_bool
1381   } {
1382     \bool_set_true:N \l_stex_symdecl_make_macro_bool
1383   }
1384   \stex_symdecl_do:n { #3 }
1385   \stex_smsmode_set_codes:
1386 }

```

(End definition for `\symdecl`. This function is documented on page 20.)

**\stex\_symdecl\_do:n**

```

1387 \cs_new_protected:Nn \stex_symdecl_do:n {
1388   \stex_if_in_module:F {
1389     % TODO throw error? some default namespace?
1390   }
1391
1392   \str_if_empty:NT \l_stex_symdecl_name_str {
1393     \str_set:Nx \l_stex_symdecl_name_str { #1 }
1394   }
1395
1396   \prop_if_exist:cT { g_stex_symdecl_
1397     \prop_item:Nn \l_stex_current_module_prop {ns} ?
1398     \prop_item:Nn \l_stex_current_module_prop {name} ?
1399     \l_stex_symdecl_name_str
1400     _prop
1401   }{
1402     % TODO throw error (beware of circular dependencies)
1403   }

```

```

1404
1405 \prop_clear:N \l_tmpa_prop
1406 \prop_put:Nnx \l_tmpa_prop { module } {
1407   \prop_item:Nn \l_stex_current_module_prop {ns} ?
1408   \prop_item:Nn \l_stex_current_module_prop {name}
1409 }
1410 \seq_clear:N \l_tmpa_seq
1411 \prop_put:Nno \l_tmpa_prop { notations } \l_tmpa_seq
1412 \prop_put:Nno \l_tmpa_prop { name } \l_stex_symdecl_name_str
1413 \prop_put:Nno \l_tmpa_prop { local } \l_stex_symdecl_local_bool
1414 \prop_put:Nno \l_tmpa_prop { type } \l_stex_symdecl_type_tl
1415
1416 \exp_args:No \stex_add_constant_to_current_module:n {
1417   \l_stex_symdecl_name_str
1418 }
1419
1420 % arity/args
1421 \int_zero:N \l_tmpb_int
1422
1423 \bool_set_true:N \l_tmpa_bool
1424 \str_map_inline:Nn \l_stex_symdecl_args_str {
1425   \token_case_meaning:NnF ##1 {
1426     0 {} 1 {} 2 {} 3 {} 4 {} 5 {} 6 {} 7 {} 8 {} 9 {}
1427     {\tl_to_str:n i} { \bool_set_false:N \l_tmpa_bool }
1428     {\tl_to_str:n b} { \bool_set_false:N \l_tmpa_bool }
1429     {\tl_to_str:n a} {
1430       \bool_set_false:N \l_tmpa_bool
1431       \int_incr:N \l_tmpb_int
1432     }
1433     {\tl_to_str:n B} {
1434       \bool_set_false:N \l_tmpa_bool
1435       \int_incr:N \l_tmpb_int
1436     }
1437   }{
1438     \msg_set:nnn{stex}{error/wrongargs}{
1439       args~value~in~symbol~declaration~for~
1440       \prop_item:Nn \l_stex_current_module_prop {ns} ?
1441       \prop_item:Nn \l_stex_current_module_prop {name} ?
1442       \l_stex_symdecl_name_str ~
1443       needs~to~be~
1444       i,~a,~b~or~B,~but~##1~given
1445     }
1446     \msg_error:nn{stex}{error/wrongargs}
1447   }
1448 }
1449 \bool_if:NTF \l_tmpa_bool {
1450   % possibly numeric
1451   \str_if_empty:NTF \l_stex_symdecl_args_str {
1452     \prop_put:Nnn \l_tmpa_prop { args } {}
1453     \prop_put:Nnn \l_tmpa_prop { arity } { 0 }
1454   }{
1455     \int_set:Nn \l_tmpa_int { \l_stex_symdecl_args_str }
1456     \prop_put:Nnx \l_tmpa_prop { arity } { \int_use:N \l_tmpa_int }
1457     \str_clear:N \l_tmpa_str

```

```

1458     \int_step_inline:nn \l_tmpa_int {
1459       \str_put_right:Nn \l_tmpa_str i
1460     }
1461     \prop_put:Nnx \l_tmpa_prop { args } { \l_tmpa_str }
1462   }
1463 } {
1464   \prop_put:Nnx \l_tmpa_prop { args } { \l_stex_symdecl_args_str }
1465   \prop_put:Nnx \l_tmpa_prop { arity }
1466     { \str_count:N \l_stex_symdecl_args_str }
1467 }
1468 \prop_put:Nnx \l_tmpa_prop { assoc } { \int_use:N \l_tmpb_int }
1469
1470
1471 % semantic macro
1472
1473 \bool_if:NT \l_stex_symdecl_make_macro_bool {
1474   \tl_set:cx { #1 } { \stex_invoke_symbol:n {
1475     \prop_item:Nn \l_tmpa_prop { module } ?
1476     \prop_item:Nn \l_tmpa_prop { name }
1477   } }
1478
1479   \bool_if:NF \l_stex_symdecl_local_bool {
1480     \exp_args:Nx \stex_add_to_current_module:n {
1481       \tl_set:cx { #1 } { \stex_invoke_symbol:n {
1482         \prop_item:Nn \l_tmpa_prop { module } ?
1483         \prop_item:Nn \l_tmpa_prop { name }
1484       } }
1485     }
1486   }
1487 }
1488
1489 % add to all symbols
1490
1491 \bool_if:NF \l_stex_symdecl_local_bool {
1492   \exp_args:Nx \stex_add_to_current_module:n {
1493     \seq_put_right:Nn \exp_not:N \l_stex_all_symbols_seq {
1494       \prop_item:Nn \l_tmpa_prop { module } ?
1495       \prop_item:Nn \l_tmpa_prop { name }
1496     }
1497   }
1498 }
1499
1500 \stex_debug:n{New~symbol:~
1501   \prop_item:Nn \l_tmpa_prop { module } ?
1502   \prop_item:Nn \l_tmpa_prop { name }^^J
1503   Type:~\exp_not:o { \l_stex_symdecl_type_tl }^^J
1504   Args:~\prop_item:Nn \l_tmpa_prop { args }
1505 }
1506
1507 % circular dependencies require this:
1508
1509 \prop_if_exist:cF {
1510   g_stex_symdecl_
1511   \prop_item:Nn \l_tmpa_prop { module } ?

```

```

1512     \prop_item:Nn \l_tmpa_prop { name }
1513     _prop
1514   } {
1515     \prop_gset_eq:cN {
1516       g_stex_symdecl_
1517       \prop_item:Nn \l_tmpa_prop { module } ?
1518       \prop_item:Nn \l_tmpa_prop { name }
1519       _prop
1520     } \l_tmpa_prop
1521   }
1522
1523   \stex_if_smsmode:TF {
1524     \bool_if:NF \l_stex_symdecl_local_bool {
1525       \exp_args:Nx \stex_addtosms:n {
1526         \prop_gset_from_keyval:cn {
1527           g_stex_symdecl_
1528           \prop_item:Nn \l_tmpa_prop { module } ?
1529           \prop_item:Nn \l_tmpa_prop { name }
1530           _prop
1531         } {
1532           name      = \prop_item:Nn \l_tmpa_prop { name }      ,
1533           module    = \prop_item:Nn \l_tmpa_prop { module }    ,
1534           notations = \prop_item:Nn \l_tmpa_prop { notations } ,
1535           local     = \prop_item:Nn \l_tmpa_prop { local }      ,
1536           type      = \prop_item:Nn \l_tmpa_prop { type }       ,
1537           args      = \prop_item:Nn \l_tmpa_prop { args }       ,
1538           arity     = \prop_item:Nn \l_tmpa_prop { arity }      ,
1539           assocs    = \prop_item:Nn \l_tmpa_prop { assocs }
1540         }
1541         \seq_put_right:Nn \exp_not:N \l_stex_all_symbols_seq {
1542           \prop_item:Nn \l_tmpa_prop { module } ?
1543           \prop_item:Nn \l_tmpa_prop { name }
1544         }
1545       }
1546     }
1547   }{
1548     \exp_args:NNx \seq_put_right:Nn \l_stex_all_symbols_seq {
1549       \prop_item:Nn \l_tmpa_prop { module } ?
1550       \prop_item:Nn \l_tmpa_prop { name }
1551     }
1552     \stex_annotate_invisible:nnn {symdecl} {
1553       \prop_item:Nn \l_tmpa_prop { module } ?
1554       \prop_item:Nn \l_tmpa_prop { name }
1555     } {
1556       \stex_annotate_invisible:nnn{type}{}{\l_stex_symdecl_type_tl$}
1557       \stex_annotate_invisible:nnn{args}{}{
1558         \prop_item:Nn \l_tmpa_prop { args }
1559       }
1560       \stex_annotate_invisible:nnn{macroname}{}{#1}
1561       \tl_if_empty:NF \l_stex_symdecl_definiens_tl {
1562         \stex_annotate_invisible:nnn{definiens}{}{
1563           {\l_stex_symdecl_definiens_tl$}
1564         }
1565       }
1566     }
  
```

```

1566 }
1567 }

```

(End definition for `\stex_symdecl_do:n`. This function is documented on page 20.)

`\stex_get_symbol:n`

```

1568 \str_new:N \l_stex_get_symbol_uri_str
1569
1570 \cs_new_protected:Nn \stex_get_symbol:n {
1571   \tl_if_head_eq_catcode:nNTF { #1 } \relax {
1572     \__stex_symdecl_get_symbol_from_cs:n { #1 }
1573   }{
1574     % argument is a string
1575     % is it a command name?
1576     \cs_if_exist:cTF { #1 }{
1577       \cs_set_eq:Nc \l_tmpa_tl { #1 }
1578       \str_set:Nx \l_tmpa_str { \cs_argument_spec:N \l_tmpa_tl }
1579       \str_if_empty:NNTF \l_tmpa_str {
1580         \exp_args:Nx \cs_if_eq:NNTF {
1581           \tl_head:N \l_tmpa_tl
1582         } \stex_invoke_symbol:n {
1583           \exp_args:No \__stex_symdecl_get_symbol_from_cs:n { \use:c { #1 } }
1584         }{
1585           \__stex_symdecl_get_symbol_from_string:n { #1 }
1586         }
1587       } {
1588         \__stex_symdecl_get_symbol_from_string:n { #1 }
1589       }
1590     }{
1591       % argument is not a command name
1592       \__stex_symdecl_get_symbol_from_string:n { #1 }
1593       % \l_stex_all_symbols_seq
1594     }
1595   }
1596 }
1597
1598 \cs_new_protected:Nn \__stex_symdecl_get_symbol_from_string:n {
1599   \prop_get:NnN \l_stex_current_module_prop
1600   { constants } \l_tmpa_seq
1601   \seq_if_in:NnTF \l_tmpa_seq { #1 } {
1602     \str_set:Nx \l_stex_get_symbol_uri_str {
1603       \prop_item:Nn \l_stex_current_module_prop { ns } ?
1604       \prop_item:Nn \l_stex_current_module_prop { name } ? #1
1605     }
1606   } {
1607     \tl_set:Nn \l_tmpa_tl {
1608       \msg_set:nnn{stex}{error/unknownsymbol}{
1609         No~symbol~#1~found!
1610       }
1611       \msg_error:nn{stex}{error/unknownsymbol}
1612     }
1613     \str_set:Nn \l_tmpa_str { #1 }
1614     \int_set:Nn \l_tmpa_int { \str_count:N \l_tmpa_str }
1615     \seq_map_inline:Nn \l_stex_all_symbols_seq {

```

```

1616     \str_set:Nn \l_tmpb_str { ##1 }
1617     \str_if_eq:eeT { \l_tmpa_str } {
1618       \str_range:Nnn \l_tmpb_str { -\l_tmpa_int } { -1 }
1619     } {
1620       \seq_map_break:n {
1621         \tl_set:Nn \l_tmpa_tl {
1622           \str_set:Nn \l_stex_get_symbol_uri_str {
1623             ##1
1624           }
1625         }
1626       }
1627     }
1628   }
1629   \l_tmpa_tl
1630 }
1631 }
1632
1633 \cs_new_protected:Nn \__stex_symdecl_get_symbol_from_cs:n {
1634   \exp_args:NNx \tl_set:Nn \l_tmpa_tl
1635     { \tl_tail:N \l_tmpa_tl }
1636   \tl_if_single:NTF \l_tmpa_tl {
1637     \exp_args:No \tl_if_head_is_group:nTF \l_tmpa_tl {
1638       \exp_after:wN \str_set:Nn \exp_after:wN
1639         \l_stex_get_symbol_uri_str \l_tmpa_tl
1640     }{
1641       % TODO
1642       % tail is not a single group
1643     }
1644   }{
1645     % TODO
1646     % tail is not a single group
1647   }
1648 }

```

(End definition for `\stex_get_symbol:n`. This function is documented on page 21.)

## 4.7 Notations

```

1649 <@@=stex_notation>
1650 notation arguments:
1651 \keys_define:nn { stex / notation } {
1652   lang .tl_set_x:N = \l__stex_notation_lang_str ,
1653   variant .tl_set_x:N = \l__stex_notation_variant_str ,
1654   prec .tl_set_x:N = \l__stex_notation_prec_str ,
1655   op .tl_set:N = \l__stex_notation_op_tl ,
1656   unknown .code:n = \str_set:Nx
1657     \l__stex_notation_variant_str \l_keys_key_str
1658 }
1659 \cs_new_protected:Nn \__stex_notation_args:n {
1660   \str_clear:N \l__stex_notation_lang_str
1661   \str_clear:N \l__stex_notation_variant_str
1662   \str_clear:N \l__stex_notation_prec_str
1663   \tl_clear:N \l__stex_notation_op_tl

```

```

1664
1665 \keys_set:nn { stex / notation } { #1 }
1666
1667 \str_set:Nx \l__stex_notation_lang_str \l__stex_notation_lang_str
1668 \str_set:Nx \l__stex_notation_variant_str \l__stex_notation_variant_str
1669 \str_set:Nx \l__stex_notation_prec_str \l__stex_notation_prec_str
1670 }

```

**\notation**

```

1671 \NewDocumentCommand \notation { 0{ } m } {
1672   \__stex_notation_args:n { #1 }
1673   \tl_clear:N \l_stex_symdecl_definiens_tl
1674   \stex_get_symbol:n { #2 }
1675   \stex_notation_do:nn { \l_stex_get_symbol_uri_str }
1676 }

```

(End definition for \notation. This function is documented on page 21.)

**\stex\_notation\_do:nn**

```

1677 \cs_new_protected:Nn \stex_notation_do:nn {
1678   \prop_set_eq:Nc \l_tmpa_prop {
1679     g_stex_symdecl_ #1 _prop
1680   }
1681
1682   \prop_clear:N \l_tmpb_prop
1683   \prop_put:Nno \l_tmpb_prop { symbol } { #1 }
1684   \prop_put:Nno \l_tmpb_prop { language } \l__stex_notation_lang_str
1685   \prop_put:Nno \l_tmpb_prop { variant } \l__stex_notation_variant_str
1686
1687   % precedences
1688   \seq_clear:N \l_tmpb_seq
1689   \exp_args:NNno
1690   \str_if_empty:NTF \l__stex_notation_prec_str {
1691     \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1692     \int_compare:nNnTF \l_tmpa_str = 0 {
1693       \exp_args:NNnx
1694       \prop_put:Nno \l_tmpb_prop { opprec }
1695       { \infprec }
1696     }{
1697       \prop_put:Nnn \l_tmpb_prop { opprec } { 0 }
1698     }
1699   } {
1700     \str_if_eq:onTF \l__stex_notation_prec_str {nobrackets}{
1701       \exp_args:NNnx
1702       \prop_put:Nno \l_tmpb_prop { opprec }
1703       { \infprec }
1704       \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1705       \int_step_inline:nn { \l_tmpa_str } {
1706         \exp_args:NNx
1707         \seq_put_right:Nn \l_tmpb_seq { \neginfprec }
1708       }
1709     }{
1710       \seq_set_split:NnV \l_tmpa_seq ; \l__stex_notation_prec_str
1711       \seq_pop_left:NNTF \l_tmpa_seq \l_tmpa_str {
1712         \prop_put:Nno \l_tmpb_prop { opprec } \l_tmpa_str

```

```

1713     \seq_pop_left:NNT \l_tmpa_seq \l_tmpa_str {
1714       \exp_args:NNNo \exp_args:NNno \seq_set_split:Nnn
1715         \l_tmpa_seq {\tl_to_str:n{x}} } { \l_tmpa_str }
1716       \seq_map_inline:Nn \l_tmpa_seq {
1717         \seq_put_right:Nn \l_tmpb_seq { ##1 }
1718       }
1719     }
1720   \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1721 }{
1722   \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1723   \int_compare:nNnTF \l_tmpa_str = 0 {
1724     \exp_args:NNnx
1725     \prop_put:Nno \l_tmpb_prop { opprec }
1726     { \infprec }
1727   }{
1728     \prop_put:Nnn \l_tmpb_prop { opprec } { 0 }
1729   }
1730 }
1731 }
1732 }
1733
1734 \seq_set_eq:NN \l_tmpa_seq \l_tmpb_seq
1735 \int_step_inline:nn { \l_tmpa_str } {
1736   \seq_pop_left:NNT \l_tmpa_seq \l_tmpb_str {
1737     \exp_args:NNx
1738     \seq_put_right:Nn \l_tmpb_seq {
1739       \prop_item:Nn \l_tmpb_prop { opprec }
1740     }
1741   }
1742 }
1743
1744 \prop_put:Nno \l_tmpb_prop { argprecs } \l_tmpb_seq
1745 \tl_clear:N \l_tmpa_tl
1746
1747 \int_compare:nNnTF \l_tmpa_str = 0 {
1748   \exp_args:NNe
1749   \cs_set:Npn \l__stex_notation_macrocode_cs {
1750     \_stex_term_math_oms:nnnn { #1 }
1751     { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1752     { \prop_item:Nn \l_tmpb_prop { opprec } }
1753     { \exp_not:n { #2 } }
1754   }
1755   \__stex_notation_final:
1756 }{
1757   \prop_get:NnN \l_tmpa_prop { args } \l_tmpb_str
1758   \str_if_in:NnTF \l_tmpb_str b {
1759     \exp_args:Nne \use:nn
1760     {
1761       \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
1762       \cs_set:Npn \l_tmpa_str { {
1763         \_stex_term_math_omb:nnnn { #1 }
1764         { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1765         { \prop_item:Nn \l_tmpb_prop { opprec } }
1766         { \exp_not:n { #2 } }

```



```

1767     }}
1768   }{
1769     \str_if_in:NnTF \l_tmpb_str B {
1770       \exp_args:Nne \use:nn
1771       {
1772         \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
1773         \cs_set:Npn \l_tmpa_str } { {
1774           \stex_term_math_omb:nnnn { #1 }
1775           { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1776           { \prop_item:Nn \l_tmpb_prop { opprec } }
1777           { \exp_not:n { #2 } } }
1778         } }
1779   }{
1780     \exp_args:Nne \use:nn
1781     {
1782       \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
1783       \cs_set:Npn \l_tmpa_str } { {
1784         \stex_term_math_oma:nnnn { #1 }
1785         { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1786         { \prop_item:Nn \l_tmpb_prop { opprec } }
1787         { \exp_not:n { #2 } } }
1788       } }
1789   }
1790 }
1791
1792 \int_zero:N \l_tmpa_int
1793 \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
1794 \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
1795 \__stex_notation_arguments:
1796 }
1797 }

```

(End definition for `\stex_notation_do:nn`. This function is documented on page 22.)

`\__stex_notation_arguments:` Takes care of annotating the arguments in a notation macro

```

1798 \cs_new_protected:Nn \__stex_notation_arguments: {
1799   \int_incr:N \l_tmpa_int
1800   \str_if_empty:NnTF \l_tmpa_str {
1801     \__stex_notation_final:
1802   }{
1803     \str_set:Nx \l_tmpb_str { \str_head:N \l_tmpa_str }
1804     \str_set:Nx \l_tmpa_str { \str_tail:N \l_tmpa_str }
1805     \str_if_eq:NnTF \l_tmpb_str a {
1806       \__stex_notation_argument_assoc:n
1807     }{
1808       \str_if_eq:NnTF \l_tmpb_str B {
1809         \__stex_notation_argument_assoc:n
1810       }{
1811         \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
1812         \tl_put_right:Nx \l_tmpa_tl {
1813           { \stex_term_math_arg:nnn
1814             { \int_use:N \l_tmpa_int }
1815             { \l_tmpb_str }
1816             { ####\int_use:N \l_tmpa_int }

```

```

1817     }
1818   }
1819   \__stex_notation_arguments:
1820 }
1821 }
1822 }
1823 }

```

(End definition for \\_\_stex\_notation\_arguments:.)

\\_stex\_notation\_argument\_assoc:n

```

1824 \cs_new_protected:Nn \__stex_notation_argument_assoc:n {
1825   \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
1826   \cs_set:Npn \l_tmpa_cs ##1 ##2 { #1 }
1827   \tl_put_right:Nx \l_tmpa_tl {
1828     { \stex_term_math_assoc_arg:nnnn
1829       { \int_use:N \l_tmpa_int }
1830       { \l_tmpb_str }
1831       \exp_args:No \exp_not:n
1832       {\exp_after:wN { \l_tmpa_cs {####1} {####2} } }
1833       { ####\int_use:N \l_tmpa_int }
1834     }
1835   }
1836   \__stex_notation_arguments:
1837 }

```

(End definition for \\_stex\_notation\_argument\_assoc:n.)

\\_\_stex\_notation\_final: Called after processing all notation arguments

```

1838 \cs_new_protected:Nn \__stex_notation_final: {
1839   \prop_get:NnN \l_tmpa_prop { arity } \l_tmpb_str
1840   \prop_get:NnN \l_tmpb_prop { symbol } \l_tmpa_str
1841   \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
1842   \exp_args:Nne \use:nn
1843   {
1844     \cs_generate_from_arg_count:cNnn {
1845       stex_notation_ \l_tmpa_str \c_hash_str
1846       \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1847       _cs
1848     }
1849     \cs_gset:Npn \l_tmpb_str { { {
1850       \exp_after:wN \exp_after:wN \exp_after:wN
1851       \exp_not:n \exp_after:wN \exp_after:wN \exp_after:wN
1852       { \exp_after:wN \l__stex_notation_macrocode_cs \l_tmpa_tl }
1853     } } }
1854
1855     \tl_if_empty:NF \l__stex_notation_op_tl {
1856       \cs_gset:cpx {
1857         stex_op_notation_ \l_tmpa_str \c_hash_str
1858         \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1859         _cs
1860       } {
1861         \stex_term_oms:nnn {
1862           \l_tmpa_str \c_hash_str \l__stex_notation_variant_str \c_hash_str

```

```

1863     \l__stex_notation_lang_str
1864   }{
1865     \l_tmpa_str
1866   }{ \comp{ \exp_args:No \exp_not:n { \l__stex_notation_op_tl } } }
1867 }
1868 }
1869
1870
1871
1872 \stex_debug:n{
1873   Notation~\l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1874   ~for~\prop_item:Nn \l_tmpb_prop { symbol }^^J
1875   Operator~precedence:~
1876   \prop_item:Nn \l_tmpb_prop { opprec }^^J
1877   Argument~precedences:~
1878   \seq_use:Nn \l_tmpa_seq {,~}^^J
1879   Notation: \cs_meaning:c {
1880     stex_notation_ \l_tmpa_str \c_hash_str
1881     \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1882     _cs
1883   }
1884 }
1885
1886 \prop_gset_eq:cN {
1887   g_stex_notation_ \l_tmpa_str \c_hash_str \l__stex_notation_variant_str
1888   \c_hash_str \l__stex_notation_lang_str _prop
1889 } \l_tmpb_prop
1890
1891 \exp_args:Nx
1892 \stex_add_to_current_module:n {
1893   \prop_get:cnN {
1894     g_stex_symdecl_
1895     \prop_item:Nn \l_tmpb_prop { symbol }
1896     _prop
1897   } { notations } \exp_not:N \l_tmpa_seq
1898   \seq_put_right:Nn \exp_not:N \l_tmpa_seq {
1899     \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1900   }
1901   \prop_put:cno {
1902     g_stex_symdecl_
1903     \prop_item:Nn \l_tmpb_prop { symbol }
1904     _prop
1905   } { notations } \exp_not:N \l_tmpa_seq
1906 }
1907
1908 \stex_if_smsmode:TF {
1909   \stex_smsmode_set_codes:
1910   \exp_args:Nx \stex_addtosms:n {
1911     \prop_gset_from_keyval:cn {
1912       g_stex_notation_ \l_tmpa_str \c_hash_str \l__stex_notation_variant_str
1913       \c_hash_str \l__stex_notation_lang_str _prop
1914     } {
1915       symbol      = \prop_item:Nn \l_tmpb_prop { symbol }      ,
1916       language    = \prop_item:Nn \l_tmpb_prop { language }    ,

```

```

1917         variant    = \prop_item:Nn \l_tmpb_prop { variant }      ,
1918         opprec      = \prop_item:Nn \l_tmpb_prop { opprec }      ,
1919         argprec     = \prop_item:Nn \l_tmpb_prop { argprec }      ,
1920     }
1921 }
1922 }{
1923   \prop_get:NnN \l_tmpa_prop { notations } \l_tmpa_seq
1924   \seq_put_right:Nx \l_tmpa_seq {
1925     \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1926   }
1927   \prop_put:Nno \l_tmpa_prop { notations } \l_tmpa_seq
1928   \prop_set_eq:cN {
1929     g_stex_symdecl_ \l_tmpa_str _prop
1930   } \l_tmpa_prop
1931
1932   % HTML annotations
1933   \stex_annotate_invisible:nnn { notation }
1934   { \prop_item:Nn \l_tmpb_prop { symbol } } {
1935     \stex_annotate_invisible:nnn { notationfragment }
1936     { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }{}
1937     \prop_get:NnN \l_tmpb_prop { argprec } \l_tmpa_seq
1938     \stex_annotate_invisible:nnn { precedence }
1939     { \prop_item:Nn \l_tmpb_prop { opprec } ;
1940       \seq_use:Nn \l_tmpa_seq { x }
1941     }{}
1942
1943     \int_zero:N \l_tmpa_int
1944     \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
1945     \tl_clear:N \l_tmpa_tl
1946     \int_step_inline:nn { \prop_item:Nn \l_tmpa_prop { arity } }{
1947       \int_incr:N \l_tmpa_int
1948       \str_set:Nx \l_tmpb_str { \str_head:N \l_tmpa_str }
1949       \str_set:Nx \l_tmpa_str { \str_tail:N \l_tmpa_str }
1950       \str_if_eq:VnTF \l_tmpb_str a {
1951         \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
1952           \c_hash_str \c_hash_str \int_use:N \l_tmpa_int a ,
1953           \c_hash_str \c_hash_str \int_use:N \l_tmpa_int b
1954         } }
1955       }{
1956         \str_if_eq:VnTF \l_tmpb_str B {
1957           \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
1958             \c_hash_str \c_hash_str \int_use:N \l_tmpa_int a ,
1959             \c_hash_str \c_hash_str \int_use:N \l_tmpa_int b
1960           } }
1961         }{
1962           \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
1963             \c_hash_str \c_hash_str \int_use:N \l_tmpa_int
1964           } }
1965         }
1966       }
1967     }
1968     \stex_annotate_invisible:nnn { notationcomp }{}{
1969       $ \exp_args:Nno \use:nn { \use:c {
1970         stex_notation_ \prop_item:Nn \l_tmpb_prop { symbol }

```

```

1971         \c_hash_str \l__stex_notation_variant_str
1972         \c_hash_str \l__stex_notation_lang_str _cs
1973     } } { \l_tmpa_tl } $
1974 }
1975 }
1976 }
1977 }

```

(End definition for \\_stex\_notation\_final:.)

**\symdef**

```

1978 \keys_define:nn { stex / symdef } {
1979   name .tl_set_x:N = \l_stex_symdecl_name_str ,
1980   local .bool_set:N = \l_stex_symdecl_local_bool ,
1981   args .tl_set_x:N = \l_stex_symdecl_args_str ,
1982   type .tl_set:N = \l_stex_symdecl_type_tl ,
1983   def .tl_set:N = \l_stex_symdecl_definiens_tl ,
1984   op .tl_set:N = \l__stex_notation_op_tl ,
1985   lang .tl_set_x:N = \l__stex_notation_lang_str ,
1986   variant .tl_set_x:N = \l__stex_notation_variant_str ,
1987   prec .tl_set_x:N = \l__stex_notation_prec_str ,
1988   unknown .code:n = \str_set:Nx
1989     \l__stex_notation_variant_str \l_keys_key_str
1990 }
1991
1992 \cs_new_protected:Nn \_stex_notation_symdef_args:n {
1993   \str_clear:N \l_stex_symdecl_name_str
1994   \str_clear:N \l_stex_symdecl_args_str
1995   \bool_set_false:N \l_stex_symdecl_local_bool
1996   \tl_clear:N \l_stex_symdecl_type_tl
1997   \tl_clear:N \l_stex_symdecl_definiens_tl
1998   \str_clear:N \l__stex_notation_lang_str
1999   \str_clear:N \l__stex_notation_variant_str
2000   \str_clear:N \l__stex_notation_prec_str
2001   \tl_clear:N \l__stex_notation_op_tl
2002
2003   \keys_set:nn { stex / symdef } { #1 }
2004
2005   \exp_args:NNo \str_set:Nn \l_stex_symdecl_name_str
2006     \l_stex_symdecl_name_str
2007   \exp_args:NNo \str_set:Nn \l_stex_symdecl_args_str
2008     \l_stex_symdecl_args_str
2009   \exp_args:NNo \str_set:Nn \l__stex_notation_lang_str
2010     \l__stex_notation_lang_str
2011   \exp_args:NNo \str_set:Nn \l__stex_notation_variant_str
2012     \l__stex_notation_variant_str
2013   \exp_args:NNo \str_set:Nn \l__stex_notation_prec_str
2014     \l__stex_notation_prec_str
2015 }
2016
2017 \NewDocumentCommand \symdef { 0{} m } {
2018   \_stex_notation_symdef_args:n { #1 }
2019   \bool_set_true:N \l_stex_symdecl_make_macro_bool
2020   \stex_symdecl_do:n { #2 }

```

```

2021 \exp_args:Nx \stex_notation_do:nn {
2022   \prop_item:Nn \l_tmpa_prop { module } ?
2023   \prop_item:Nn \l_tmpa_prop { name }
2024 }
2025 }

```

(End definition for \symdef. This function is documented on page 22.)

**\stex\_invoke\_symbol:n** Invokes a semantic macro

```

2026 %\cs_new_protected:Nn \stex_invoke_symbol:n {
2027 % \peek_charcode_remove:NTF ! {
2028 %   \stex_term_custom:nn { #1 } { }
2029 % } {
2030 %   \if_mode_math:
2031 %     \exp_after:wN \__stex_notation_invoke_math:n
2032 %   \else:
2033 %     \exp_after:wN \__stex_notation_invoke_text:n
2034 %   \fi: { #1 }
2035 % }
2036 %}
2037
2038 \cs_new_protected:Nn \stex_invoke_symbol:n {
2039   \if_mode_math:
2040     \exp_after:wN \__stex_notation_invoke_math:n
2041   \else:
2042     \exp_after:wN \__stex_notation_invoke_text:n
2043   \fi: { #1 }
2044 }

```

(End definition for \stex\_invoke\_symbol:n. This function is documented on page 21.)

**\\_\_stex\_notation\_invoke\_math:n**

```

2045 \cs_new_protected:Nn \__stex_notation_invoke_math:n {
2046   \peek_charcode_remove:NTF ! {
2047     \peek_charcode:NTF [ {
2048       \__stex_notation_invoke_op:nw { #1 }
2049     }{
2050       \__stex_notation_invoke_op:nw { #1 } []
2051     }
2052   }{
2053     \peek_charcode_remove:NTF * {
2054       \__stex_notation_invoke_text:n { #1 }
2055     }{
2056       \peek_charcode:NTF [ {
2057         \__stex_notation_invoke_math:nw { #1 }
2058       }{
2059         \__stex_notation_invoke_math:nw { #1 } []
2060       }
2061     }
2062   }
2063 }

```

(End definition for \\_\_stex\_notation\_invoke\_math:n.)

\\_stex\_notation\_invoke\_op:nw

```

2064 \cs_new_protected:Npn \__stex_notation_invoke_op:nw #1 [#2] {
2065   \__stex_notation_args:n { #2 }
2066   \cs_if_exist:cTF {
2067     stex_op_notation_ #1 \c_hash_str
2068     \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str _cs
2069   }{
2070     \csname stex_op_notation_ #1 \c_hash_str
2071       \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str _cs
2072     \endcsname
2073   }{
2074     % TODO throw error
2075   }
2076 }

```

(End definition for \\_stex\_notation\_invoke\_op:nw.)

\\_stex\_notation\_invoke\_math:nw

```

2077 \cs_new_protected:Npn \__stex_notation_invoke_math:nw #1 [#2] {
2078   \__stex_notation_args:n { #2 }
2079   \prop_set_eq:Nc \l_tmpa_prop {
2080     g_stex_symdecl_ #1 _prop
2081   }
2082   \prop_get:NnN \l_tmpa_prop { notations } \l_tmpa_seq
2083   \seq_if_empty:NTF \l_tmpa_seq {
2084     \msg_set:nnn{stex}{error/nonotations}{
2085       Symbol~#1~used,~but~has~no~notations!
2086     }
2087     \msg_error:nn{stex}{error/nonotations}
2088   } {
2089     \seq_if_in:NxTF \l_tmpa_seq
2090     { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }{
2091       \use:c{
2092         stex_notation_ #1 \c_hash_str
2093         \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
2094         _cs
2095       }
2096     }{
2097       \str_if_empty:NTF \l__stex_notation_variant_str {
2098         \str_if_empty:NTF \l__stex_notation_lang_str {
2099           \seq_get_left:NN \l_tmpa_seq \l_tmpa_str
2100           \use:c{
2101             stex_notation_ #1 \c_hash_str \l_tmpa_str
2102             _cs
2103           }
2104         }{
2105           \msg_set:nnn{stex}{error/wrongnotation}{
2106             Symbol~#1~has~no~notation~
2107             \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
2108           }
2109           \msg_error:nn{stex}{error/wrongnotation}
2110         }
2111       }{
2112         \msg_set:nnn{stex}{error/wrongnotation}{

```

```

2113         Symbol~#1~has~no~notation~
2114         \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
2115     }
2116     \msg_error:nn{stex}{error/wrongnotation}
2117 }
2118 }
2119 }
2120 }

```

(End definition for \\_\_stex\_notation\_invoke\_math:nw.)

\\_stex\_notation\_invoke\_text:n

```

2121 \cs_new_protected:Nn \__stex_notation_invoke_text:n {
2122     \peek_charcode_remove:NTF ! {
2123         \stex_term_custom:nn { #1 } { }
2124     }{
2125         \prop_set_eq:Nc \l_tmpa_prop {
2126             g_stex_symdecl_ #1 _prop
2127         }
2128         \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
2129         \exp_args:Nnx \stex_term_custom:nn { #1 } { \l_tmpa_str }
2130     }
2131 }

```

(End definition for \\_\_stex\_notation\_invoke\_text:n.)

## 4.8 Terms

2132 <@@=stex\_term>

Precedences:

```

\infprec
\neginfprec
\l__stex_term_downprec
2133 \tl_const:Nx \infprec {\int_use:N \c_max_int}
2134 \tl_const:Nx \neginfprec {-\int_use:N \c_max_int}
2135 \int_new:N \l__stex_term_downprec
2136 \int_set_eq:NN \l__stex_term_downprec \neginfprec

```

(End definition for \infprec, \neginfprec, and \l\_\_stex\_term\_downprec. These variables are documented on page 23.)

Bracketing:

```

\l__stex_term_left_bracket_str
\l__stex_term_right_bracket_str
2137 \tl_set:Nn \l__stex_term_left_bracket_str (
2138 \tl_set:Nn \l__stex_term_right_bracket_str )

```

(End definition for \l\_\_stex\_term\_left\_bracket\_str and \l\_\_stex\_term\_right\_bracket\_str.)

\\_stex\_term\_maybe\_brackets:nn Compares precedences and insert brackets accordingly

```

2139 \cs_new_protected:Nn \__stex_term_maybe_brackets:nn {
2140     \int_compare:nNnTF { #1 } > \l__stex_term_downprec {
2141         \bool_if:NTF \l_stex_inarray_bool { #2 }{
2142             \dobrackets { #2 }
2143         }
2144     }{ #2 }
2145 }

```



(End definition for `\_stex_term_maybe_brackets:nn`.)

`\dobrackets`

```

2146 %\RequirePackage{scalerel}
2147 \cs_new_protected:Npn \dobrackets #1 {
2148   %\ThisStyle{\if D\m@switch
2149   %   \exp_args:Nnx \use:nn
2150   %   { \exp_after:wN \left\l__stex_term_left_bracket_str #1 }
2151   %   { \exp_not:N\right\l__stex_term_right_bracket_str }
2152   %   \else
2153   %   \exp_args:Nnx \use:nn
2154   %   { \l__stex_term_left_bracket_str #1 }
2155   %   { \l__stex_term_right_bracket_str }
2156   %\fi}
2157 }
```

(End definition for `\dobrackets`. This function is documented on page 23.)

`\withbrackets`

```

2158 \cs_new_protected:Npn \withbrackets #1 #2 #3 {
2159   \exp_args:Nnx \use:nn
2160   {
2161     \tl_set:Nx \l__stex_term_left_bracket_str { #1 }
2162     \tl_set:Nx \l__stex_term_right_bracket_str { #2 }
2163     #3
2164   }
2165   {
2166     \tl_set:Nn \exp_not:N \l__stex_term_left_bracket_str
2167     { \l__stex_term_left_bracket_str }
2168     \tl_set:Nn \exp_not:N \l__stex_term_right_bracket_str
2169     { \l__stex_term_right_bracket_str }
2170   }
2171 }
```

(End definition for `\withbrackets`. This function is documented on page 23.)

`\STEXinvisible`

```

2172 \cs_new_protected:Npn \STEXinvisible #1 {
2173   \stex_annotate_invisible:n { #1 }
2174 }
```

(End definition for `\STEXinvisible`. This function is documented on page 25.)

OMDOC terms:

`\_stex_term_math_oms:nnnn`

```

2175 \cs_new_protected:Nn \_stex_term_oms:nnn {
2176   \stex_annotate:nnn{ OMID }{ #2 }{
2177     \stex_highlight_term:nn { #1 } { #3 }
2178   }
2179 }
2180
2181 \cs_new_protected:Nn \_stex_term_math_oms:nnnn {
2182   \_stex_term_maybe_brackets:nn { #3 }{
2183     \_stex_term_oms:nnn { #1 } { #1\c_hash_str#2 } { #4 }
2184   }
2185 }
```

(End definition for `\_stex_term_math_oms:nnnn`. This function is documented on page 22.)

`\_stex_term_math_oma:nnnn`

```

2186 \cs_new_protected:Nn \_stex_term_oma:nnn {
2187   \stex_annotate:nnn{ OMA }{ #2 }{
2188     \stex_highlight_term:nn { #1 } { #3 }
2189   }
2190 }
2191
2192 \cs_new_protected:Nn \_stex_term_math_oma:nnnn {
2193   \_stex_term_maybe_brackets:nn { #3 }{
2194     \_stex_term_oma:nnn { #1 } { #1\c_hash_str#2 } { #4 }
2195   }
2196 }

```

(End definition for `\_stex_term_math_oma:nnnn`. This function is documented on page 22.)

`\_stex_term_math_omb:nnnn`

```

2197 \cs_new_protected:Nn \_stex_term_ombind:nnn {
2198   \stex_annotate:nnn{ OMBIND }{ #2 }{
2199     \stex_highlight_term:nn { #1 } { #3 }
2200   }
2201 }
2202
2203 \cs_new_protected:Nn \_stex_term_math_omb:nnnn {
2204   \_stex_term_maybe_brackets:nn { #3 }{
2205     \_stex_term_ombind:nnn { #1 } { #1\c_hash_str#2 } { #4 }
2206   }
2207 }

```

(End definition for `\_stex_term_math_omb:nnnn`. This function is documented on page 22.)

`\_stex_term_math_arg:nnn`

```

2208 \cs_new_protected:Nn \_stex_term_arg:nn {
2209   \stex_unhighlight_term:n {
2210     \stex_annotate:nnn{ arg }{ #1 }{ #2 }
2211   }
2212 }
2213 \cs_new_protected:Nn \_stex_term_math_arg:nnn {
2214   \exp_args:Nnx \use:nn
2215   { \int_set:Nn \l__stex_term_downprec { #2 }
2216     \_stex_term_arg:nn { #1 }{ #3 }
2217   }
2218   { \int_set:Nn \exp_not:N \l__stex_term_downprec { \int_use:N \l__stex_term_downprec } }
2219 }

```

(End definition for `\_stex_term_math_arg:nnn`. This function is documented on page 23.)

`\_stex_term_math_assoc_arg:nnnn`

```

2220 \cs_new_protected:Nn \_stex_term_math_assoc_arg:nnnn {
2221   \seq_set_split:Nnn \l_tmpa_seq , { #4 }
2222   \int_compare:nNnTF { \seq_count:N \l_tmpa_seq } < 2 {
2223     \tl_set:Nn \l_tmpa_tl { #4 }
2224   }{
2225     \cs_set:Npn \l_tmpa_cs ##1 ##2 { #3 }

```

```

2226 \seq_reverse:N \l_tmpa_seq
2227 \seq_pop_left:NN \l_tmpa_seq \l_tmpb_tl
2228 \tl_set:No \l_tmpa_tl { \l_tmpb_tl }
2229
2230 \seq_map_inline:Nn \l_tmpa_seq {
2231   \exp_args:NNo \tl_set:No \l_tmpa_tl {
2232     \exp_args:Nno
2233     \l_tmpa_cs { ##1 } \l_tmpa_tl
2234   }
2235 }
2236
2237 }
2238 \exp_args:Nnno
2239 \stex_term_math_arg:nnn{#1}{#2}\l_tmpa_tl
2240 }

```

(End definition for `\stex_term_math_assoc_arg:nnnn`. This function is documented on page 23.)

`\stex_term_custom:nn`

```

2241 \cs_new_protected:Nn \stex_term_custom:nn {
2242   \str_set:Nn \l__stex_term_custom_uri { #1 }
2243   \str_set:Nn \l_tmpa_str { #2 }
2244   \tl_clear:N \l_tmpa_tl
2245   \int_zero:N \l_tmpa_int
2246   \int_set:Nn \l_tmpb_int { \str_count:N \l_tmpa_str }
2247   \__stex_term_custom_loop:
2248 }

```

(End definition for `\stex_term_custom:nn`. This function is documented on page 24.)

`\__stex_term_custom_loop:`

```

2249 \cs_new_protected:Nn \__stex_term_custom_loop: {
2250   \bool_set_false:N \l_tmpa_bool
2251   \bool_while_do:nn {
2252     \str_if_eq_p:ee X {
2253       \str_item:Nn \l_tmpa_str { \l_tmpa_int + 1 }
2254     }
2255   }{
2256     \int_incr:N \l_tmpa_int
2257   }
2258
2259   \peek_charcode:NTF [ {
2260     % notation/text component
2261     \__stex_term_custom_component:w
2262   } {
2263     \int_compare:nNnTF \l_tmpa_int = \l_tmpb_int {
2264       % all arguments read => finish
2265       \__stex_term_custom_final:
2266     } {
2267       % arguments missing
2268       \peek_charcode_remove:NTF * {
2269         % invisible, specific argument position or both
2270         \peek_charcode:NTF [ {
2271           % visible specific argument position
2272           \__stex_term_custom_arg:wn

```

```

2273     } {
2274         % invisible
2275         \peek_charcode_remove:NTF * {
2276             % invisible specific argument position
2277             \__stex_term_custom_arg_inv:wn
2278         } {
2279             % invisible next argument
2280             \__stex_term_custom_arg_inv:wn [ \l_tmpa_int + 1 ]
2281         }
2282     }
2283 } {
2284     % next normal argument
2285     \__stex_term_custom_arg:wn [ \l_tmpa_int + 1 ]
2286 }
2287 }
2288 }
2289 }

```

(End definition for \\_\_stex\_term\_custom\_loop:.)

\\_\_stex\_term\_custom\_arg\_inv:wn

```

2290 \cs_new_protected:Npn \__stex_term_custom_arg_inv:wn [ #1 ] #2 {
2291     \bool_set_true:N \l_tmpa_bool
2292     \__stex_term_custom_arg:wn [ #1 ] { #2 }
2293 }

```

(End definition for \\_\_stex\_term\_custom\_arg\_inv:wn.)

\\_\_stex\_term\_custom\_arg:wn

```

2294 \cs_new_protected:Npn \__stex_term_custom_arg:wn [ #1 ] #2 {
2295     \str_set:Nx \l_tmpb_str {
2296         \str_item:Nn \l_tmpa_str { #1 }
2297     }
2298     \str_case:VnTF \l_tmpb_str {
2299         { X } { } % TODO throw error ?
2300         { i } { \__stex_term_custom_set_X:n { #1 } }
2301         { b } { \__stex_term_custom_set_X:n { #1 } }
2302         { a } { \__stex_term_custom_set_X:n { #1 } } % TODO ?
2303         { B } { \__stex_term_custom_set_X:n { #1 } } % TODO ?
2304     }{}{
2305         % TODO throw error
2306     }
2307
2308     \bool_if:nTF \l_tmpa_bool {
2309         \tl_put_right:Nx \l_tmpa_tl {
2310             \stex_annotate_invisible:n {
2311                 \stex_term_arg:nn { \int_eval:n { #1 } }
2312                 \exp_not:n { { #2 } }
2313             }
2314         }
2315     } {
2316         \tl_put_right:Nx \l_tmpa_tl {
2317             \stex_term_arg:nn { \int_eval:n { #1 } }
2318             \exp_not:n { { #2 } }
2319         }

```

```

2320 }
2321
2322 \__stex_term_custom_loop:
2323 }

```

(End definition for \\_\_stex\_term\_custom\_arg:wn.)

\\_\_stex\_term\_custom\_set\_X:n

```

2324 \cs_new_protected:Nn \__stex_term_custom_set_X:n {
2325   \str_set:Nx \l_tmpa_str {
2326     \str_range:Nnn \l_tmpa_str 1 { #1 - 1 }
2327     X
2328     \str_range:Nnn \l_tmpa_str { #1 + 1 } { -1 }
2329   }
2330 }

```

(End definition for \\_\_stex\_term\_custom\_set\_X:n.)

\\_stex\_term\_custom\_component:

```

2331 \cs_new_protected:Npn \_stex_term_custom_component:w [ #1 ] {
2332   \tl_put_right:Nn \l_tmpa_tl { \comp{ #1 } }
2333   \__stex_term_custom_loop:
2334 }

```

(End definition for \\_stex\_term\_custom\_component:.)

\\_\_stex\_term\_custom\_final:

```

2335 \cs_new_protected:Nn \__stex_term_custom_final: {
2336   \int_compare:nNnTF \l_tmpb_int = 0 {
2337     \exp_args:Nnno \_stex_term_oms:nnn
2338   }{
2339     \str_if_in:NnTF \l_tmpa_str {b} {
2340       \exp_args:Nnno \_stex_term_ombind:nnn
2341     } {
2342       \exp_args:Nnno \_stex_term_oma:nnn
2343     }
2344   }
2345   { \l__stex_term_custom_uri } { \l__stex_term_custom_uri } { \l_tmpa_tl }
2346 }

```

(End definition for \\_\_stex\_term\_custom\_final:.)

**\symref**

\symname

```

2347 \NewDocumentCommand \symref { m m }{
2348   \STEXsymbol{#1}![#2]
2349 }
2350
2351 \keys_define:nn { stex / symname } {
2352   post      .tl_set_x:N    = \l_stex_symname_post_str
2353 }
2354
2355 \cs_new_protected:Nn \stex_symname_args:n {
2356   \str_clear:N \l_stex_symname_post_str
2357   \keys_set:nn { stex / symname } { #1 }
2358   \exp_args:NNo \str_set:Nn \l_stex_symname_post_str

```

```

2359     \l_stex_symname_post_str
2360 }
2361
2362 \NewDocumentCommand \symname { 0{} m }{
2363   \stex_symname_args:n { #1 }
2364   \stex_get_symbol:n { #2 }
2365   \str_set:Nx \l_tmpa_str {
2366     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
2367   }
2368   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
2369   \exp_args:NNx \use:nn
2370   \stex_invoke_symbol:n { { \l_stex_get_symbol_uri_str }![
2371     \l_tmpa_str \l_stex_symname_post_str
2372   ] }
2373 }

```

(End definition for `\symref` and `\symname`. These functions are documented on page 21.)

## 4.9 Notation Components

```

2374 <@@=stex_notationcomps>

```

`\stex_highlight_term:nn`

```

2375 \latexml_if:F {
2376   \scalatex_if:F{
2377     \RequirePackage{pdfcomment}
2378   }
2379 }
2380
2381 \str_new:N \l__stex_notationcomps_highlight_uri_str
2382 \cs_new_protected:Nn \stex_highlight_term:nn {
2383   \exp_args:Nnx
2384   \use:nn {
2385     \str_set:Nx \l__stex_notationcomps_highlight_uri_str { #1 }
2386     #2
2387   } {
2388     \str_set:Nx \exp_not:N \l__stex_notationcomps_highlight_uri_str
2389       { \l__stex_notationcomps_highlight_uri_str }
2390   }
2391 }
2392
2393 \cs_new_protected:Nn \stex_unhighlight_term:n {
2394   % \latexml_if:TF {
2395   %   #1
2396   % } {
2397   %   \scalatex_if:TF {
2398   %     #1
2399   %   } {
2400     #1 %\iffalse{{\fi}} #1 {{\iffalse}}\fi
2401   % }
2402   % }
2403 }

```

(End definition for `\stex_highlight_term:nn`. This function is documented on page 24.)

```

\comp
\@comp
\@defemph
2404 \cs_new_protected:Npn \comp #1 {
2405   \str_if_empty:NF \l__stex_notationcomps_highlight_uri_str {
2406     \scalatex_if:TF {
2407       \stex_annotate:nnn { comp } { \l__stex_notationcomps_highlight_uri_str } { #1 }
2408     } {
2409       \exp_args:Nnx \@comp { #1 } { \l__stex_notationcomps_highlight_uri_str }
2410     }
2411   }
2412 }
2413
2414 \cs_new_protected:Npn \@comp #1 #2 {
2415   \pdftooltip {
2416     \textcolor{blue}{#1}
2417   } { #2 }
2418 }
2419
2420 \cs_new_protected:Npn \@defemph #1 #2 {
2421   \pdftooltip {
2422     \textbf{\textcolor{magenta}{#1}}
2423   } { #2 }
2424 }

```

(End definition for `\comp`, `\@comp`, and `\@defemph`. These functions are documented on page 24.)

`\ellipses`

```

2425 \NewDocumentCommand \ellipses {} { \ldots }

```

(End definition for `\ellipses`. This function is documented on page 25.)

```

\parray
\prmatrix
\parrayline
\parraycell
2426 \bool_new:N \l_stex_inparray_bool
2427 \bool_set_false:N \l_stex_inparray_bool
2428 \NewDocumentCommand \parray { m m } {
2429   \begingroup
2430   \bool_set_true:N \l_stex_inparray_bool
2431   \begin{array}{#1}
2432     #2
2433   \end{array}
2434   \endgroup
2435 }
2436
2437 \NewDocumentCommand \prmatrix { m } {
2438   \begingroup
2439   \bool_set_true:N \l_stex_inparray_bool
2440   \begin{matrix}
2441     #1
2442   \end{matrix}
2443   \endgroup
2444 }
2445
2446 \def \parrayline #1 #2 {
2447   #1 #2 \bool_if:NT \l_stex_inparray_bool {\}
2448 }

```

```

2449
2450 \def \parraycell #1 {
2451   #1 \bool_if:NT \l_stex_inarray_bool {&}
2452 }

```

(End definition for \parray and others. These functions are documented on page ??.)

## 4.10 Structural Features

```

2453 <@@=stex_features>

```

symboldoc

```

2454 \NewDocumentEnvironment{symboldoc}{m}{
2455   \seq_set_split:Nnn \l_tmpa_seq , { #1 }
2456   \seq_clear:N \l_tmpb_seq
2457   \seq_map_inline:Nn \l_tmpa_seq {
2458     \stex_get_symbol:n { ##1 }
2459     \exp_args:NNo \seq_put_right:Nn \l_tmpb_seq {
2460       \l_stex_get_symbol_uri_str
2461     }
2462   }
2463   \par
2464   \exp_args:Nnnx
2465   \begin{stex_annotate_env}{symboldoc}{\seq_use:Nn \l_tmpb_seq {,}}
2466 }{
2467   \end{stex_annotate_env}
2468 }

```

STEXdefinition

```

2469
2470 \NewDocumentCommand \__stex_features_definiendum:w { 0{} m m } {
2471   \stex_get_symbol:n { #2 }
2472   \scalatex_if:TF {
2473     \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } { ##3 }
2474   } {
2475     \exp_args:Nnx \@defemph { ##3 } { \l_stex_get_symbol_uri_str }
2476   }
2477 }
2478 \NewDocumentCommand \__stex_features_definame:w { 0{} m } {
2479   % TODO: root
2480   \stex_get_symbol:n { #2 }
2481   \str_set:Nx \l_tmpa_str {
2482     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
2483   }
2484   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
2485   \scalatex_if:TF {
2486     \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
2487       \l_tmpa_str
2488     }
2489   } {
2490     \@defemph {
2491       \l_tmpa_str
2492     } { \l_stex_get_symbol_uri_str }
2493   }

```



```

2494 }
2495
2496 \cs_new_protected:Nn \__stex_features_defi_begin:n {
2497   \let\definiendum\__stex_features_definiendum:w
2498   \let\definame\__stex_features_definame:w
2499   \seq_set_split:Nnn \l_tmpa_seq , { #1 }
2500   \seq_clear:N \l_tmpb_seq
2501   \seq_map_inline:Nn \l_tmpa_seq {
2502     \stex_get_symbol:n { ##1 }
2503     \exp_args:NNo \seq_put_right:Nn \l_tmpb_seq {
2504       \l_stex_get_symbol_uri_str
2505     }
2506   }
2507   \exp_args:Nnnx
2508   \begin{stex_annotate_env}{definition}{\seq_use:Nn \l_tmpb_seq {,}}
2509 }
2510
2511 \cs_new_protected:Nn \__stex_features_defi_end: {
2512   \end{stex_annotate_env}
2513 }
2514
2515 \NewDocumentEnvironment{STEXdefinition}{ m }{
2516   \__stex_features_defi_begin:n { #1 }
2517 }{
2518   \__stex_features_defi_end:
2519 }

```

\setSTEXdefinition

```

2520 \cs_new_protected:Npn \setSTEXdefinition #1 {
2521   \AddToHook{env/#1/before}[stex]{\__stex_features_defi_begin:n{}}
2522   \AddToHook{env/#1/after}[stex]{\__stex_features_defi_end:}
2523 }

```

(End definition for \setSTEXdefinition. This function is documented on page ??.)

structural@feature

```

2524
2525 \NewDocumentEnvironment{structural@feature}{ m m m }{
2526   \stex_if_in_module:F {
2527     \msg_set:nnn{stex}{error/nomodule}{
2528       Structural~Feature~has~to~occur~in~a~module:\\
2529       Feature~#2~of~type~#1\\
2530       In~File:~\stex_path_to_string:N \g_stex_currentfile_seq
2531     }
2532     \msg_error:nn{stex}{error/nomodule}
2533   }
2534
2535   \str_set:Nx \l_stex_module_name_str {
2536     \prop_item:Nn \l_stex_current_module_prop
2537       { name } / #2 - feature
2538   }
2539
2540
2541   \str_clear:N \l_tmpa_str

```

```

2542 \seq_clear:N \l_tmpa_seq
2543 \tl_clear:N \l_tmpa_tl
2544 \exp_args:NNx \prop_set_from_keyval:Nn \l_stex_current_module_prop {
2545     origname = #2,
2546     name      = \l_stex_module_name_str ,
2547     ns        = \l_stex_module_ns_str ,
2548     imports   = \exp_not:o { \l_tmpa_seq } ,
2549     constants = \exp_not:o { \l_tmpa_seq } ,
2550     content   = \exp_not:o { \l_tmpa_tl } ,
2551     file      = \exp_not:o { \g_stex_currentfile_seq } ,
2552     lang      = \l_stex_module_lang_str ,
2553     sig       = \l_tmpa_str ,
2554     meta      = \l_tmpa_str ,
2555     feature   = #1 ,
2556 }
2557
2558 \stex_if_smsmode:TF {
2559     \stex_smsmode_set_codes:
2560 } {
2561     \begin{stex_annotate_env}{ feature:#1 }{}
2562     \stex_annotate_invisible:nnn{header}{}{ #3 }
2563 }
2564 }{
2565     \str_set:Nx \l_tmpa_str {
2566         c_stex_feature_
2567         \prop_item:Nn \l_stex_current_module_prop { ns } ?
2568         \prop_item:Nn \l_stex_current_module_prop { name }
2569         _prop
2570     }
2571     \prop_gset_eq:cn { \l_tmpa_str } \l_stex_current_module_prop
2572     \prop_gset_eq:NN \g_stex_last_feature_prop \l_stex_current_module_prop
2573     \stex_if_smsmode:TF {
2574         \exp_args:Nx \stex_addtosms:n {
2575             \prop_gset_from_keyval:cn {
2576                 c_stex_feature_
2577                 \prop_item:Nn \l_stex_current_module_prop { ns } ?
2578                 \prop_item:Nn \l_stex_current_module_prop { name }
2579                 _prop
2580             } {
2581                 origname = #2,
2582                 name      = \prop_item:cn { \l_tmpa_str } { name } ,
2583                 ns        = \prop_item:cn { \l_tmpa_str } { ns } ,
2584                 imports   = \prop_item:cn { \l_tmpa_str } { imports } ,
2585                 constants = \prop_item:cn { \l_tmpa_str } { constants } ,
2586                 content   = \prop_item:cn { \l_tmpa_str } { content } ,
2587                 file      = \prop_item:cn { \l_tmpa_str } { file } ,
2588                 lang      = \prop_item:cn { \l_tmpa_str } { lang } ,
2589                 sig       = \prop_item:cn { \l_tmpa_str } { sig } ,
2590                 meta      = \prop_item:cn { \l_tmpa_str } { meta } ,
2591                 feature   = \prop_item:cn { \l_tmpa_str } { feature }
2592             }
2593         }
2594     } {
2595         \end{stex_annotate_env}

```

```

2596 }
2597 }
2598

```

structure

```

2599
2600 \prop_new:N \l_stex_all_structures_prop
2601
2602 \keys_define:nn { stex / features / structure } {
2603   name .tl_set_x:N = \l__stex_features_structure_name_str ,
2604 }
2605
2606 \cs_new_protected:Nn \__stex_features_structure_args:n {
2607   \str_clear:N \l__stex_features_structure_name_str
2608   \keys_set:nn { stex / features / structure } { #1 }
2609   \exp_args:NNo \str_set:Nn \l__stex_features_structure_name_str
2610     \l__stex_features_structure_name_str
2611 }
2612
2613 %\stex_new_feature:nnnn { structure } { 0{} m } {
2614 %   \__stex_features_structure_args:n { ##1 }
2615 %   \str_if_empty:NT \l__stex_features_structure_name_str {
2616 %     \str_set:Nx \l__stex_features_structure_name_str { ##2 }
2617 %   }
2618 %} {
2619 %
2620 %}
2621
2622 \NewDocumentEnvironment{structure}{ 0{} m }{
2623   \__stex_features_structure_args:n { #1 }
2624   \str_if_empty:NT \l__stex_features_structure_name_str {
2625     \str_set:Nx \l__stex_features_structure_name_str { #2 }
2626   }
2627   \exp_args:Nnnx
2628   \begin{structural@feature}{ structure }
2629     { \l__stex_features_structure_name_str }{}
2630     \seq_clear:N \l_tmpa_seq
2631     \prop_put:Nno \l_stex_current_module_prop { fields } \l_tmpa_seq
2632
2633   }{
2634     \prop_get:NnN \l_stex_current_module_prop { constants } \l_tmpa_seq
2635     \prop_get:NnN \l_stex_current_module_prop { fields } \l_tmpb_seq
2636     \str_set:Nx \l_tmpa_str {
2637       \prop_item:Nn \l_stex_current_module_prop { ns } ?
2638       \prop_item:Nn \l_stex_current_module_prop { name }
2639     }
2640     \seq_map_inline:Nn \l_tmpa_seq {
2641       \exp_args:NNx \seq_put_right:Nn \l_tmpb_seq { \l_tmpa_str ? ##1 }
2642     }
2643     \prop_put:Nno \l_stex_current_module_prop { fields } { \l_tmpb_seq }
2644     \exp_args:Nnx
2645     \AddToHookNext { env / structure / after }{
2646       \symdecl[type = \exp_not:N\collection,def={\STEXsymbol{module-type}}{
2647         \_stex_term_math_oms:nnnn { \l_tmpa_str }{}{}{}{}

```

```

2648     }}, name = \prop_item:Nn \l_stex_current_module_prop { origname }]{ #2 }
2649     \STEXexport {
2650         \prop_put:Nno \exp_not:N \l_stex_all_structures_prop
2651             {\prop_item:Nn \l_stex_current_module_prop { origname }}
2652             {\l_tmpa_str}
2653         \prop_put:Nno \exp_not:N \l_stex_all_structures_prop
2654             {#2}{\l_tmpa_str}
2655     %     \seq_put_right:Nn \exp_not:N \l_stex_all_structures_seq {
2656     %         \prop_item:Nn \l_stex_current_module_prop { origname },
2657     %         \l_tmpa_str
2658     %     }
2659     %     \seq_put_right:Nn \exp_not:N \l_stex_all_structures_seq {
2660     %         #2,\l_tmpa_str
2661     %     }
2662     %     \tl_set:cx { #2 } {
2663     %         \stex_invoke_structure:n { \l_tmpa_str }
2664     %     }
2665     }
2666
2667     \end{structural@feature}
2668     % \g_stex_last_feature_prop
2669 }

```

\instantiate

```

2670 \seq_new:N \l__stex_features_structure_field_seq
2671 \str_new:N \l__stex_features_structure_field_str
2672 \str_new:N \l__stex_features_structure_def_tl
2673 \prop_new:N \l__stex_features_structure_prop
2674 \NewDocumentCommand \instantiate { m O{} m }{
2675     \stex_smsmode_set_codes:
2676     \prop_get:Nn \l_stex_all_structures_prop {#1} \l_tmpa_str
2677     \prop_set_eq:Nc \l__stex_features_structure_prop {
2678         c_stex_feature_\l_tmpa_str _prop
2679     }
2680     \seq_set_from_clist:Nn \l__stex_features_structure_field_seq { #2 }
2681     \seq_map_inline:Nn \l__stex_features_structure_field_seq {
2682         \seq_set_split:Nnn \l_tmpa_seq={}{ ##1 }
2683         \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > 1 {
2684             \seq_get_left:NN \l_tmpa_seq \l_tmpa_tl
2685             \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq
2686             {!} \l_tmpa_tl
2687             \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} > 1 {
2688                 \str_set:Nx \l__stex_features_structure_field_str {\seq_item:Nn \l_tmpb_seq 1}
2689                 \seq_get_right:NN \l_tmpb_seq \l_tmpb_tl
2690                 \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
2691             }{
2692                 \str_set:Nx \l__stex_features_structure_field_str \l_tmpa_tl
2693                 \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
2694                 \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq{!}
2695                 \l_tmpa_tl
2696                 \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} > 1 {
2697                     \seq_get_left:NN \l_tmpb_seq \l_tmpa_tl
2698                     \seq_get_right:NN \l_tmpb_seq \l_tmpb_tl
2699                 }{

```

```

2700         \tl_clear:N \l_tmpb_tl
2701     }
2702 }
2703 }{
2704     \seq_set_split:Nnn \l_tmpa_seq{!}{ ##1 }
2705     \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > 1 {
2706         \str_set:Nx \l__stex_features_structure_field_str {\seq_item:Nn \l_tmpa_seq 1}
2707         \seq_get_right:NN \l_tmpa_seq \l_tmpb_tl
2708         \tl_clear:N \l_tmpa_tl
2709     }{
2710         % TODO throw error
2711     }
2712 }
2713 % \l_tmpa_str: name
2714 % \l_tmpa_tl: definiens
2715 % \l_tmpb_tl: notation
2716 \tl_if_empty:NT \l__stex_features_structure_field_str {
2717     % TODO throw error
2718 }
2719 \str_clear:N \l_tmpb_str
2720
2721 \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
2722 \seq_map_inline:Nn \l_tmpa_seq {
2723     \seq_set_split:Nnn \l_tmpb_seq ? { ####1 }
2724     \seq_get_right:NN \l_tmpb_seq \l_tmpb_str
2725     \str_if_eq:NNT \l__stex_features_structure_field_str \l_tmpb_str {
2726         \seq_map_break:n {
2727             \str_set:Nn \l_tmpb_str { ####1 }
2728         }
2729     }
2730 }
2731 \prop_get:cnN { g_stex_symdecl_ \l_tmpb_str _prop } {args}
2732 \l_tmpb_str
2733
2734 \tl_if_empty:NTF \l_tmpb_tl {
2735     \tl_if_empty:NF \l_tmpa_tl {
2736         \exp_args:Nx \use:n {
2737             \symdecl[args=\l_tmpb_str,def={\exp_args:No\exp_not:n{\l_tmpa_tl}}]{#3/\l__stex_fe
2738         }
2739     }
2740 }{
2741     \tl_if_empty:NTF \l_tmpa_tl {
2742         \exp_args:Nx \use:n {
2743             \symdef[args=\l_tmpb_str]{#3/\l__stex_features_structure_field_str}\exp_after:wN\
2744         }
2745     }
2746 }{
2747     \exp_args:Nx \use:n {
2748         \symdef[args=\l_tmpb_str,def={\exp_args:No\exp_not:n{\l_tmpa_tl}}]{#3/\l__stex_fea
2749         \exp_after:wN\exp_not:n\exp_after:wN{\l_tmpb_tl}
2750     }
2751 }
2752 }
2753 % \par \prop_item:Nn \l_stex_current_module_prop {ns} ?

```

```

2754 % \prop_item:Nn \l_stex_current_module_prop {name} ?
2755 % #3/\l__stex_features_structure_field_str
2756 % \par
2757 % \expandafter\present\csname
2758 % g_stex_symdecl_
2759 % \prop_item:Nn \l_stex_current_module_prop {ns} ?
2760 % \prop_item:Nn \l_stex_current_module_prop {name} ?
2761 % #3/\l__stex_features_structure_field_str
2762 % _prop
2763 % \endcsname
2764 }
2765
2766 \tl_clear:N \l__stex_features_structure_def_tl
2767
2768 \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
2769 \seq_map_inline:Nn \l_tmpa_seq {
2770   \seq_set_split:Nnn \l_tmpb_seq ? { ##1 }
2771   \seq_get_right:NN \l_tmpb_seq \l_tmpa_str
2772   \exp_args:Nx \use:n {
2773     \tl_put_right:Nn \exp_not:N \l__stex_features_structure_def_tl {
2774
2775     }
2776   }
2777
2778   \prop_if_exist:cF {
2779     g_stex_symdecl_
2780     \prop_item:Nn \l_stex_current_module_prop {ns} ?
2781     \prop_item:Nn \l_stex_current_module_prop {name} ?
2782     #3/\l_tmpa_str
2783     _prop
2784   }{
2785     \prop_get:cnN { g_stex_symdecl_ ##1 _prop } {args}
2786     \l_tmpb_str
2787     \exp_args:Nx \use:n {
2788       \symdecl[args=\l_tmpb_str]{#3/\l_tmpa_str}
2789     }
2790   }
2791 }
2792
2793 \symdecl*[type={\STEXsymbol{module-type}}{
2794   \_stex_term_math_oms:nnnn {
2795     \prop_item:Nn \l__stex_features_structure_prop {ns} ?
2796     \prop_item:Nn \l__stex_features_structure_prop {name}
2797     }{}{0}{}
2798   }}{#3}
2799
2800 % TODO: -> sms file
2801
2802 \tl_set:cx{ #3 }{
2803   \stex_invoke_structure:nnn {
2804     \prop_item:Nn \l_stex_current_module_prop {ns} ?
2805     \prop_item:Nn \l_stex_current_module_prop {name} ? #3
2806   } {
2807     \prop_item:Nn \l__stex_features_structure_prop {ns} ?

```

```

2808     \prop_item:Nn \l__stex_features_structure_prop {name}
2809   }
2810 }
2811
2812 }

```

(End definition for \instantiate. This function is documented on page ??.)

\stex\_invoke\_structure:nnn

```

2813 % #1: URI of the instance
2814 % #2: URI of the instantiated module
2815 \cs_new_protected:Nn \stex_invoke_structure:nnn {
2816   \tl_if_empty:nTF{ #3 }{
2817     \prop_set_eq:Nc \l__stex_features_structure_prop {
2818       c_stex_feature_ #2 _prop
2819     }
2820     \tl_clear:N \l_tmpa_tl
2821     \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
2822     \seq_map_inline:Nn \l_tmpa_seq {
2823       \seq_set_split:Nnn \l_tmpb_seq ? { ##1 }
2824       \seq_get_right:NN \l_tmpb_seq \l_tmpa_str
2825       \cs_if_exist:cT {
2826         stex_notation_ #1/\l_tmpa_str \c_hash_str\c_hash_str_cs
2827       }{
2828         \tl_if_empty:NF \l_tmpa_tl {
2829           \tl_put_right:Nn \l_tmpa_tl {,}
2830         }
2831         \tl_put_right:Nx \l_tmpa_tl {
2832           \stex_invoke_symbol:n {#1/\l_tmpa_str}!
2833         }
2834       }
2835     }
2836     \scalatexBREAK
2837     \exp_args:No \mathstrut \l_tmpa_tl
2838   }{
2839     \stex_invoke_symbol:n{#1/#3}
2840   }
2841 }

```

(End definition for \stex\_invoke\_structure:nnn. This function is documented on page ??.)

## 4.11 Put these somewhere

\MSC

```

2842 \NewDocumentCommand \MSC {m} {
2843   % TODO
2844 }

```

(End definition for \MSC. This function is documented on page ??.)

```

2845 \@ifpackageloaded{tikzinput}{
2846   \RequirePackage{stex-tikzinput}
2847 }{}
2848
2849 \AddToHook{begindocument}{

```

```

2850 \input{stex-metattheory}
2851 }
2852 </package>

```

## 4.12 Metatheory

The default meta theory for an  $\text{\LaTeX}$  module. Contains symbols so ubiquitous, that it is virtually impossible to describe any flexiformal content without them, or that are required to annotate even the most primitive symbols with meaningful (foundation-independent) “type”-annotations, or required for basic structuring principles (theorems, definitions).

Foundations should ideally instantiate these symbols with their formal counterparts, e.g. `isa` corresponds to a typing operation in typed setting, or the  $\in$ -operator in set-theoretic contexts; `bind` corresponds to a universal quantifier in ( $n$ th-order) logic, or a  $\Pi$  in dependent type theories.

```

2853 (*metatheory)
2854 \ExplSyntaxOn
2855 \str_const:Nn \c_stex_metatheory_ns_str {http://mathhub.info/sTeX}
2856 \begin{@module}[ns=\c_stex_metatheory_ns_str,meta=NONE]{Metatheory}
2857 \ExplSyntaxOff
2858
2859 % is-a (a:A, a \in A, a is an A, etc.)
2860 \symdecl[args=ai]{isa}
2861 \notation[typed]{isa}{#1 \comp: #2}{#1 \comp, #2}
2862 \notation[in]{isa}{#1 \comp\in #2}{#1 \comp, #2}
2863 \notation[pred]{isa}{#2\comp(#1 \comp)}{#1 \comp, #2}
2864
2865 % bind (\forall, \Pi, \lambda etc.)
2866 \symdecl[args=Bi]{bind}
2867 \notation[forall]{bind}{\comp\forall #1.;#2}{#1 \comp, #2}
2868 \notation[\Pi]{bind}{\comp\prod_{#1}#2}{#1 \comp, #2}
2869 \notation[deffun]{bind}{\comp( #1 \comp{} )\;\to\;}{#1 \comp, #2}
2870
2871 % dummy variable
2872 \symdecl{dummyvar}
2873 \notation[underscore]{dummyvar}{\comp\_}
2874 \notation[dot]{dummyvar}{\comp\cdot}
2875 \notation[dot]{dummyvar}{\comp\cdot}
2876 \notation[dash]{dummyvar}{\comp{\rm --}}
2877
2878 %fromto (function space, Hom-set, implication etc.)
2879 \symdecl[args=ai]{fromto}
2880 \notation[xarrow]{fromto}{#1 \comp\to #2}{#1 \comp\times #2}
2881 \notation[arrow]{fromto}{#1 \comp\to #2}{#1 \comp\to #2}
2882
2883 % mapto (lambda etc.)
2884 \symdecl[args=Bi]{mapto}
2885 \notation[mapsto]{mapto}{#1 \comp\mapsto #2}{#1 \comp, #2}
2886 \notation[lambda]{mapto}{\comp\lambda #1 \comp.; #2}{#1 \comp, #2}
2887 \notation[lambdau]{mapto}{\comp\lambda_{#1} \comp.; #2}{#1 \comp, #2}
2888
2889 % function/operator application
2890 \symdecl[args=ia]{apply}

```



```

2891 \notation[prec=0;0x\neginfprec,parens]{apply}{#1 \comp( #2 \comp)}{#1 \comp, #2}
2892 \notation[prec=0;0x\neginfprec,lambda]{apply}{#1 \; #2 }{#1 \; #2}
2893
2894 % ‘‘type’’ of all collections (sets, classes, types, kinds)
2895 \symdecl{collection}
2896 \notation[U]{collection}{\comp{\mathcal{U}}}
2897 \notation[set]{collection}{\comp{\textsf{Set}}}
2898
2899 % sequences
2900 \symdecl[args=1]{seqtype}
2901 \notation[kleene]{seqtype}{#1^{\comp\ast}}
2902
2903 \symdef[args=2,li]{sequence-index}{#1_{#2}}
2904 \notation[ui]{sequence-index}{#1^{#2}}
2905
2906 %\symdef[args=3,li]{sequence-from-to}{#1_{#2}\comp{,\ellipses,}#1_{#3}}
2907 %\notation[ui]{sequence-from-to}{#1^{#2}\comp{,\ellipses,}#1^{#3}}
2908 % ^ superceded by \aseqfromto and \livar/\uivar
2909
2910 \symdef[args=a,prec=nobrackets]{aseqdots}{#1\comp{,\ellipses}}{#1\comp,#2}
2911 \symdef[args=ai,prec=nobrackets]{aseqfromto}{#1\comp{,\ellipses\comp,}#2 }{#1\comp,#2}
2912
2913 % letin (‘‘let’’, local definitions, variable substitution)
2914 \symdecl[args=bii]{letin}
2915 \notation[let]{letin}{\comp{\rm let}}{#1\comp{=}#2\; \comp{\rm in}}{#3}
2916 \notation[subst]{letin}{#3 \comp[ #1 \comp/ #2 \comp]}
2917 \notation[frac]{letin}{#3 \comp[ \frac{#2}{#1} \comp]}
2918
2919 % structures
2920 \symdecl*[args=1]{module-type}
2921 \notation{module-type}{\mathtt{MOD} #1}
2922 \symdecl[name=mathematical-structure,args=a]{mathstruct} % TODO
2923 \notation[angle,prec=nobrackets]{mathstruct}{\comp\angle #1 \comp\rangle}{#1 \comp, #2}
2924
2925 \STEXexport{
2926   \let\nappa\apply
2927   \def\nappli#1#2#3#4{\apply{#1}{\naseqli{#2}{#3}{#4}}}
2928   \def\livar{\csname sequence-index\endcsname[li]}
2929   \def\uivar{\csname sequence-index\endcsname[ui]}
2930   \def\naseqli#1#2#3{\aseqfromto{\livar{#1}{#2}}{\livar{#1}{#3}}}
2931   \def\nasequi#1#2#3{\aseqfromto{\uivar{#1}{#2}}{\uivar{#1}{#3}}}
2932 }
2933
2934 \end{@module}
2935 \ExplSyntaxOff
2936 \metatheory

```

## 4.13 Auxiliary Packages

### 4.13.1 tikzinput

```

2937 \tikzinput
2938 \tikzinput
2939 \ProvidesExplPackage{tikzinput}{2021/08/31}{1.9}{bla}

```

```

2940 \RequirePackage{l3keys2e}
2941
2942 \keys_define:nn { tikzinput } {
2943   image .bool_set:N = \c_tikzinput_image_bool
2944 }
2945
2946 \ProcessKeysOptions { tikzinput }
2947
2948 \bool_if:NTF \c_tikzinput_image_bool {
2949   \RequirePackage{graphicx}
2950
2951   \providecommand\usetikzlibrary[]{}
2952   \newcommand\tikzinput [2] []{\includegraphics[#1]{#2}}
2953 }{
2954   \RequirePackage{tikz}
2955   \RequirePackage{standalone}
2956
2957   \newcommand \tikzinput [2] [] {
2958     \setkeys{Gin}{#1}
2959     \ifx \Gin@width \Gin@exclamation
2960       \ifx \Gin@height \Gin@exclamation
2961         \input { #2 }
2962       \else
2963         \resizebox{!}{ \Gin@height }{
2964           \input { #2 }
2965         }
2966       \fi
2967     \else
2968       \ifx \Gin@height \Gin@exclamation
2969         \resizebox{ \Gin@width }{!}{
2970           \input { #2 }
2971         }
2972       \else
2973         \resizebox{ \Gin@width }{ \Gin@height }{
2974           \input { #2 }
2975         }
2976       \fi
2977     \fi
2978   }
2979 }
2980
2981 \newcommand \ctikzinput [2] [] {
2982   \begin{center}
2983     \tikzinput [#1] {#2}
2984   \end{center}
2985 }
2986
2987 \@ifpackageloaded{stex}{
2988   \RequirePackage{stex-tikzinput}
2989 }{}
2990 </tikzinput>
2991 <*stex-tikzinput>
2992 \ProvidesExplPackage{stex-tikzinput}{2021/08/31}{1.9}{bla}
2993 \RequirePackage{stex}

```

```

2994 \RequirePackage{tikzinput}
2995
2996 % TODO
2997
2998 \stex-tikzinput

```

#### 4.13.2 sTeX1 Compatibility

```

2999 \smglom
3000 \RequirePackage{expl3,l3keys2e}
3001 \ProvidesExplClass{smglom}{2021/08/01}{1.9}{sTeX1 compatibility}
3002 \LoadClass[border=1px,varwidth]{standalone}
3003 \setlength\textwidth{15cm}
3004 %\g@addto@macro{\@parboxrestore}{\setlength\parskip{\baselineskip}}
3005 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{stex}}
3006 \ProcessOptions
3007
3008 \RequirePackage{stex-compatibility}
3009 \smglom
3010
3011 \compat
3012 \stex_deprec
3013 \ProvidesExplPackage{stex-compatibility}{2021/08/01}{1.9}{bla}
3014 \RequirePackage[lang={de,en,ro,tr,fr}]{stex}
3015
3016 \NewDocumentEnvironment { mhmodnl } { 0{} m m } {
3017   \msg_set:nnn{stex}{warning/deprecated}{
3018     \
3019     Environment~mhmodnl-is~deprected! \
3020     Please~update~module~#2~in~file~
3021     \stex_path_to_string:N \g_stex_currentfile_seq!
3022     \
3023   }
3024   \msg_warning:nn{stex}{warning/deprecated}
3025
3026   \begin{module}[#1,lang=#3]{#2}
3027     \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
3028     \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
3029     \seq_set_split:NnV \l_tmpb_seq . \l_tmpa_str
3030     \seq_pop_left:NN \l_tmpb_seq \l_tmpa_str
3031     \input { \stex_path_to_string:N \l_tmpa_seq / \l_tmpa_str.tex }
3032   } {
3033     \end{module}
3034   }
3035
3036 \NewDocumentEnvironment { modsig } { 0{} m } {
3037   \stex_if_in_module:TF {
3038     \prop_get:NnN \l_stex_current_module_prop {name} \l_tmpa_str
3039     \str_set:Nn \l_tmpb_str { #2 }
3040     \str_if_eq:NNTF \l_tmpa_str \l_tmpb_str {
3041       \prop_set_eq:NN \l_modsig_old_module_prop \l_stex_current_module_prop
3042       \begin{@module}{modsig-#2}
3043         % \prop_set_eq:NN \l_stex_current_module_prop \l_modsig_old_module_prop
3044       } {
3045         \begin{@module}{#2}

```

```

3046     }
3047   } {
3048     \begin{@module}{#2}
3049   }
3050 }{
3051   \end{@module}
3052   \AddToHookNext { env / modsig / after }{
3053     \stex_if_in_module:T {
3054       \prop_get:NnN \l_stex_current_module_prop {name} \l_tmpa_str
3055       \str_set:Nn \l_tmpb_str { #2 }
3056       \str_if_eq:NNT \l_tmpa_str \l_tmpb_str {
3057         % \xdef \g_stex_module_after_group_tl {
3058           \stex_if_smsmode:TF {
3059             \exp_args:Nx
3060             \stex_add_to_current_module:n {
3061               \stex_debug:n{Activating~signature~of~#2}
3062               \exp_not:N \prop_item:cn { c_stex_module_
3063               \prop_item:Nn \l_stex_current_module_prop {ns} ?
3064               \prop_item:Nn \l_stex_current_module_prop {name}
3065               / modsig-#2_prop } { content }
3066             }
3067           }
3068         {
3069           \gdef \g_stex_modsig_after_group_tl {
3070             \stex_activate_module:n {
3071               \prop_item:Nn \l_stex_current_module_prop {ns} ?
3072               \prop_item:Nn \l_stex_current_module_prop {name}
3073               / modsig-#2
3074             }
3075           }
3076           \exp_args:Nx
3077           \stex_add_to_current_module:n {
3078             \stex_activate_module:n {
3079               \prop_item:Nn \l_stex_current_module_prop {ns} ?
3080               \prop_item:Nn \l_stex_current_module_prop {name}
3081               / modsig-#2
3082             }
3083           }
3084         }
3085         \aftergroup \g_stex_modsig_after_group_tl
3086       }
3087     }
3088   }
3089 }
3090 }
3091
3092 \cs_new_protected:Npn \gimport {
3093   \peek_charcode_remove:NTF * {
3094     \gimport_do:
3095   } {
3096     \gimport_do:
3097   }
3098 }
3099

```

```

3100 \NewDocumentCommand \gimport_do: { 0{ } m } {
3101   \msg_set:nnn{stex}{warning/deprecated}{
3102     \\
3103     \c_backslash_str gimport~is~deprecated! \\
3104     Please~use~\c_backslash_str importmodule[#1]{#2}~instead!~(in~file~
3105     \stex_path_to_string:N \g_stex_currentfile_seq)
3106     \\ \\
3107   }
3108   \msg_warning:nn{stex}{warning/deprecated}
3109   \importmodule[#1]{#2}
3110 }
3111
3112 \cs_new_protected:Npn \guse {
3113   \peek_charcode_remove:NTF * {
3114     \guse_do:
3115   } {
3116     \guse_do:
3117   }
3118 }
3119
3120 \NewDocumentCommand \guse_do: { 0{ } m } {
3121   \msg_set:nnn{stex}{warning/deprecated}{
3122     \\
3123     \c_backslash_str guse~is~deprecated! \\
3124     Please~use~\c_backslash_str usemodule[#1]{#2}~instead!~(in~file~
3125     \stex_path_to_string:N \g_stex_currentfile_seq)
3126     \\ \\
3127   }
3128   \msg_warning:nn{stex}{warning/deprecated}
3129   \usemodule[#1]{#2}
3130 }
3131
3132 \cs_new:Nn \stex_capitalize:n { \uppercase{#1} }
3133
3134 \cs_new_protected:Npn \symi {
3135   \peek_charcode_remove:NTF * {
3136     \symi_do:
3137   } {
3138     \symi_do:
3139   }
3140 }
3141
3142 \NewDocumentCommand \symi_do: { 0{ } m } {
3143   \msg_set:nnn{stex}{warning/deprecated}{
3144     \\
3145     \c_backslash_str symi~is~deprecated! \\
3146     Please~use~\c_backslash_str symdecl[#1]{#2}~instead!~(in~file~
3147     \stex_path_to_string:N \g_stex_currentfile_seq)
3148     \\ \\
3149   }
3150   \msg_warning:nn{stex}{warning/deprecated}
3151   \symdecl*[#1]{#2}
3152 }
3153

```

```

3154 \cs_new_protected:Npn \symii {
3155   \peek_charcode_remove:NTF * {
3156     \symii_do:
3157   } {
3158     \symii_do:
3159   }
3160 }
3161
3162 \NewDocumentCommand \symii_do: { 0{} m m } {
3163   \msg_set:nnn{stex}{warning/deprecated}{
3164     \\\
3165     \c_backslash_str symii~is~deprecated! \\\
3166     Please~use~\c_backslash_str symdecl[#1]{#2-#3}~instead!~(in~file~
3167     \stex_path_to_string:N \g_stex_currentfile_seq)
3168     \\\ \\\
3169   }
3170   \msg_warning:nn{stex}{warning/deprecated}
3171   \symdecl*{#1}{#2-#3}
3172 }
3173
3174 \cs_new_protected:Npn \symiii {
3175   \peek_charcode_remove:NTF * {
3176     \symiii_do:
3177   } {
3178     \symiii_do:
3179   }
3180 }
3181
3182 \NewDocumentCommand \symiii_do: { 0{} m m m } {
3183   \msg_set:nnn{stex}{warning/deprecated}{
3184     \\\
3185     \c_backslash_str symiii~is~deprecated! \\\
3186     Please~use~\c_backslash_str symdecl[#1]{#2-#3-#4}~instead!~(in~file~
3187     \stex_path_to_string:N \g_stex_currentfile_seq)
3188     \\\ \\\
3189   }
3190   \msg_warning:nn{stex}{warning/deprecated}
3191   \symdecl*{#1}{#2-#3-#4}
3192 }
3193
3194 \keys_define:nn { stex / deprec / defi } {
3195   name .tl_set_x:N = \l_tmpa_str
3196 }
3197
3198 \cs_new_protected:Npn \defi {
3199   \peek_charcode_remove:NTF * {
3200     \defi_do:
3201   } {
3202     \defi_do:
3203   }
3204 }
3205
3206 \NewDocumentCommand \defi_do: { 0{} m } {
3207   \str_clear:N \l_tmpa_str

```

```

3208 \keys_set:nn { stex / deprec / defi } { #1 }
3209
3210 \str_if_empty:NTF \l_tmpa_str {
3211   \msg_set:nnn{stex}{warning/deprecated}{
3212     \\\
3213     \c_backslash_str defi-is~deprecated! \\\
3214     Please~use~\c_backslash_str STEXsymbol{#2}![#2]~instead!~(in~file~
3215     \stex_path_to_string:N \g_stex_currentfile_seq)
3216     \\\ \\\
3217   }
3218   \msg_warning:nn{stex}{warning/deprecated}
3219   \STEXsymbol { #2 }![ \comp{#2} ]
3220 } {
3221   \msg_set:nnn{stex}{warning/deprecated}{
3222     \\\
3223     \c_backslash_str defi-is~deprecated! \\\
3224     Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2 ]~instead!~(in~file~
3225     \stex_path_to_string:N \g_stex_currentfile_seq)
3226     \\\ \\\
3227   }
3228   \msg_warning:nn{stex}{warning/deprecated}
3229   \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2} ]
3230 }
3231 }
3232
3233
3234 \cs_new_protected:Npn \Defi {
3235   \peek_charcode_remove:NTF * {
3236     \Defi_do:
3237   } {
3238     \Defi_do:
3239   }
3240 }
3241
3242 \NewDocumentCommand \Defi_do: { 0{ } m } {
3243   \str_clear:N \l_tmpa_str
3244   \keys_set:nn { stex / deprec / defi } { #1 }
3245
3246   \str_if_empty:NTF \l_tmpa_str {
3247     \msg_set:nnn{stex}{warning/deprecated}{
3248       \\\
3249       \c_backslash_str Defi-is~deprecated! \\\
3250       Please~use~\c_backslash_str STEXsymbol{#2}![\exp_after:wN \stex_capitalize:n #2]~inste
3251       \stex_path_to_string:N \g_stex_currentfile_seq)
3252       \\\ \\\
3253     }
3254     \msg_warning:nn{stex}{warning/deprecated}
3255     \STEXsymbol { #2 }![ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3256   } {
3257     \msg_set:nnn{stex}{warning/deprecated}{
3258       \\\
3259       \c_backslash_str Defi-is~deprecated! \\\
3260       Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ \exp_after:wN \stex_capitalize
3261       \stex_path_to_string:N \g_stex_currentfile_seq)

```

```

3262     \\\ \\\
3263   }
3264   \msg_warning:nn{stex}{warning/deprecated}
3265   \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3266 }
3267 }
3268
3269 \cs_new_protected:Npn \adeffi {
3270   \peek_charcode_remove:NTF * {
3271     \adeffi_do:
3272   } {
3273     \adeffi_do:
3274   }
3275 }
3276
3277 \NewDocumentCommand \adeffi_do: { 0{} m m } {
3278   \str_clear:N \l_tmpa_str
3279   \keys_set:nn { stex / deprec / defi } { #1 }
3280
3281   \str_if_empty:NTF \l_tmpa_str {
3282     \msg_set:nnn{stex}{warning/deprecated}{
3283       \\\
3284       \c_backslash_str adefi-is-deprecated! \\\
3285       Please~use~\c_backslash_str STEXsymbol{#3}![#2]~instead!~(in~file~
3286       \stex_path_to_string:N \g_stex_currentfile_seq)
3287       \\\ \\\
3288     }
3289     \msg_warning:nn{stex}{warning/deprecated}
3290     \STEXsymbol { #3 }![ \comp{#2} ]
3291   } {
3292     \msg_set:nnn{stex}{warning/deprecated}{
3293       \\\
3294       \c_backslash_str adefi-is-deprecated! \\\
3295       Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2 ]~instead!~(in~file~
3296       \stex_path_to_string:N \g_stex_currentfile_seq)
3297       \\\ \\\
3298     }
3299     \msg_warning:nn{stex}{warning/deprecated}
3300     \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2} ]
3301   }
3302 }
3303
3304 \cs_new_protected:Npn \defis {
3305   \peek_charcode_remove:NTF * {
3306     \defis_do:
3307   } {
3308     \defis_do:
3309   }
3310 }
3311
3312 \NewDocumentCommand \defis_do: { 0{} m } {
3313   \str_clear:N \l_tmpa_str
3314   \keys_set:nn { stex / deprec / defi } { #1 }
3315

```



```

3316 \str_if_empty:NTF \l_tmpa_str {
3317   \msg_set:nnn{stex}{warning/deprecated}{
3318     \
3319     \c_backslash_str defis-is-deprecated! \
3320     Please~use~\c_backslash_str STExsymbol{#2}![#2s]~instead!~(in~file~
3321     \stex_path_to_string:N \g_stex_currentfile_seq)
3322     \
3323   }
3324   \msg_warning:nn{stex}{warning/deprecated}
3325   \STExsymbol { #2 }![ \comp{#2s} ]
3326 } {
3327   \msg_set:nnn{stex}{warning/deprecated}{
3328     \
3329     \c_backslash_str defis-is-deprecated! \
3330     Please~use~\c_backslash_str STExsymbol { \l_tmpa_str }[ #2s ]~instead!~(in~file~
3331     \stex_path_to_string:N \g_stex_currentfile_seq)
3332     \
3333   }
3334   \msg_warning:nn{stex}{warning/deprecated}
3335   \exp_args:No \STExsymbol { \l_tmpa_str }![ \comp{#2s} ]
3336 }
3337 }
3338
3339 \cs_new_protected:Npn \defii {
3340   \peek_charcode_remove:NTF * {
3341     \defii_do:
3342   } {
3343     \defii_do:
3344   }
3345 }
3346
3347 \NewDocumentCommand \defii_do: { 0{} m m } {
3348   \str_clear:N \l_tmpa_str
3349   \keys_set:nn { stex / deprec / defi } { #1 }
3350   \str_if_empty:NTF \l_tmpa_str {
3351     \msg_set:nnn{stex}{warning/deprecated}{
3352       \
3353       \c_backslash_str defii-is-deprecated! \
3354       Please~use~\c_backslash_str STExsymbol{#2~#3}![#2~#3]~instead!~(in~file~
3355       \stex_path_to_string:N \g_stex_currentfile_seq)
3356       \
3357     }
3358     \msg_warning:nn{stex}{warning/deprecated}
3359     \STExsymbol { #2~#3 }![ \comp{#2~#3} ]
3360   } {
3361     \msg_set:nnn{stex}{warning/deprecated}{
3362       \
3363       \c_backslash_str defii-is-deprecated! \
3364       Please~use~\c_backslash_str STExsymbol { \l_tmpa_str }[ #2~#3 ]~instead!~(in~file~
3365       \stex_path_to_string:N \g_stex_currentfile_seq)
3366       \
3367     }
3368     \msg_warning:nn{stex}{warning/deprecated}
3369     \exp_args:No \STExsymbol { \l_tmpa_str }![ \comp{#2~#3} ]

```

```

3370 }
3371 }
3372
3373
3374 \cs_new_protected:Npn \defiis {
3375   \peek_charcode_remove:NTF * {
3376     \defiis_do:
3377   } {
3378     \defiis_do:
3379   }
3380 }
3381
3382 \NewDocumentCommand \defiis_do: { 0{} m m } {
3383   \str_clear:N \l_tmpa_str
3384   \keys_set:nn { stex / deprec / defi } { #1 }
3385   \str_if_empty:NTF \l_tmpa_str {
3386     \msg_set:nnn{stex}{warning/deprecated}{
3387       \\\
3388       \c_backslash_str defiis-is-deprecated! \\\
3389       Please~use~\c_backslash_str STEXsymbol{#2~#3}![#2~#3s]~instead!~(in~file~
3390       \stex_path_to_string:N \g_stex_currentfile_seq)
3391       \\\ \\\
3392     }
3393     \msg_warning:nn{stex}{warning/deprecated}
3394     \STEXsymbol { #2~#3 }![ \comp{#2~#3s} ]
3395   } {
3396     \msg_set:nnn{stex}{warning/deprecated}{
3397       \\\
3398       \c_backslash_str defiis-is-deprecated! \\\
3399       Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2~#3s ]~instead!~(in~file~
3400       \stex_path_to_string:N \g_stex_currentfile_seq)
3401       \\\ \\\
3402     }
3403     \msg_warning:nn{stex}{warning/deprecated}
3404     \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2~#3s} ]
3405   }
3406 }
3407
3408
3409 \cs_new_protected:Npn \defiii {
3410   \peek_charcode_remove:NTF * {
3411     \defiii_do:
3412   } {
3413     \defiii_do:
3414   }
3415 }
3416
3417 \NewDocumentCommand \defiii_do: { 0{} m m m } {
3418   \str_clear:N \l_tmpa_str
3419   \keys_set:nn { stex / deprec / defi } { #1 }
3420   \str_if_empty:NTF \l_tmpa_str {
3421     \msg_set:nnn{stex}{warning/deprecated}{
3422       \\\
3423       \c_backslash_str defiii-is-deprecated! \\\

```

```

3424     Please~use~\c_backslash_str STExsymbol{#2~#3~#4}![#2~#3~#4]~instead!~(in~file~
3425     \stex_path_to_string:N \g_stex_currentfile_seq)
3426     \\\ \\\
3427   }
3428   \msg_warning:nn{stex}{warning/deprecated}
3429   \STExsymbol { #2~#3~#4 }![ \comp{#2~#3~#4} ]
3430 } {
3431   \msg_set:nnn{stex}{warning/deprecated}{
3432     \\\
3433     \c_backslash_str defiii~is-deprecated! \\\
3434     Please~use~\c_backslash_str STExsymbol { \l_tmpa_str }[ #2~#3~#4 ]~instead!~(in~file~
3435     \stex_path_to_string:N \g_stex_currentfile_seq)
3436     \\\ \\\
3437   }
3438   \msg_warning:nn{stex}{warning/deprecated}
3439   \exp_args:No \STExsymbol { \l_tmpa_str }![ \comp{#2~#3~#4} ]
3440 }
3441 }
3442
3443 %\RequirePackage[hyperref]{ntheorem}
3444 %\theoremstyle{plain}
3445 %\RequirePackage{amsthm}
3446
3447 \NewDocumentEnvironment {definition} { 0{} } {
3448   \begin{STExdefinition}{}
3449 }{
3450   \end{STExdefinition}
3451 }
3452 \keys_define:nn { stex / omtex } {
3453   title .tl_set_x:N = \l_stex_omtext_title_str
3454 }
3455 \cs_new_protected:Nn \stex_omtext_args:n {
3456   \str_clear:N \l_stex_omtext_title_str
3457   \keys_set:nn { stex / omtex }{ #1 }
3458   \exp_args:NNo \str_set:Nn \l_stex_omtext_title_str
3459     \l_stex_omtext_title_str
3460 }
3461 \NewDocumentEnvironment {omtext} { 0{} } {
3462   \stex_omtext_args:n { #1 }
3463   \paragraph{\l_stex_omtext_title_str}
3464 }{
3465 }
3466 }
3467 \NewDocumentEnvironment {assertion} { 0{} } {
3468 }{
3469 }
3470 }
3471 }
3472
3473 \NewDocumentCommand \inlinedef { m } {
3474   \begingroup
3475   \let\definiendum\_stex_deprec_definiendum:w
3476   \let\definame\_stex_deprec_definame:w
3477   #1

```

```

3478 \endgroup
3479 }
3480
3481 \NewDocumentCommand \inlineass { m } { #1 }
3482
3483 \NewDocumentCommand \trefi { 0{} m } {
3484   \str_set:Nn \l_tmpa_str { #1 }
3485   \str_if_empty:NTF \l_tmpa_str {
3486     \msg_set:nnn{stex}{warning/deprecated}{
3487       \\\
3488       \c_backslash_str trefi-is-deprecated! \\\
3489       Please~use~\c_backslash_str STExsymbol{#2}! [#2]~instead!~(in~file~
3490       \stex_path_to_string:N \g_stex_currentfile_seq)
3491       \\\
3492     }
3493     \msg_warning:nn{stex}{warning/deprecated}
3494     \STExsymbol { #2 }! [ \comp{#2} ]
3495   } {
3496     \msg_set:nnn{stex}{warning/deprecated}{
3497       \\\
3498       \c_backslash_str trefi-is-deprecated! \\\
3499       Please~use~\c_backslash_str STExsymbol { #1?#2 } [ #2 ]~instead!~(in~file~
3500       \stex_path_to_string:N \g_stex_currentfile_seq)
3501       \\\
3502     }
3503     \msg_warning:nn{stex}{warning/deprecated}
3504     \STExsymbol { #1 }! [ \comp{#2} ]
3505   }
3506 }
3507
3508
3509 \NewDocumentCommand \Trefi { 0{} m } {
3510   \str_set:Nn \l_tmpa_str { #1 }
3511   \str_if_empty:NTF \l_tmpa_str {
3512     \msg_set:nnn{stex}{warning/deprecated}{
3513       \\\
3514       \c_backslash_str Trefi-is-deprecated! \\\
3515       Please~use~\c_backslash_str STExsymbol{#2}! [\exp_after:wN \stex_capitalize:n #2]~inste
3516       \stex_path_to_string:N \g_stex_currentfile_seq)
3517       \\\
3518     }
3519     \msg_warning:nn{stex}{warning/deprecated}
3520     \STExsymbol { #2 }! [ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3521   } {
3522     \msg_set:nnn{stex}{warning/deprecated}{
3523       \\\
3524       \c_backslash_str Trefi-is-deprecated! \\\
3525       Please~use~\c_backslash_str STExsymbol { #1 } [ \exp_after:wN \stex_capitalize:n #2 ]~i
3526       \stex_path_to_string:N \g_stex_currentfile_seq)
3527       \\\
3528     }
3529     \msg_warning:nn{stex}{warning/deprecated}
3530     \STExsymbol { #1 }! [ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3531   }

```

```

3532 }
3533
3534 \NewDocumentCommand \trefis { O{} m } {
3535   \str_set:Nn \l_tmpa_str { #1 }
3536   \str_if_empty:NTF \l_tmpa_str {
3537     \msg_set:nnn{stex}{warning/deprecated}{
3538       \\\
3539       \c_backslash_str trefi-is-deprecated! \\\
3540       Please~use~\c_backslash_str STEXsymbol{#2}![#2s]~instead!~(in~file~
3541       \stex_path_to_string:N \g_stex_currentfile_seq)
3542       \\\ \\\
3543     }
3544     \msg_warning:nn{stex}{warning/deprecated}
3545     \STEXsymbol { #2 }![ \comp{#2s} ]
3546   } {
3547     \msg_set:nnn{stex}{warning/deprecated}{
3548       \\\
3549       \c_backslash_str trefi-is-deprecated! \\\
3550       Please~use~\c_backslash_str STEXsymbol { #1 }[ #2s ]~instead!~(in~file~
3551       \stex_path_to_string:N \g_stex_currentfile_seq)
3552       \\\ \\\
3553     }
3554     \msg_warning:nn{stex}{warning/deprecated}
3555     \STEXsymbol { #1 }![ \comp{#2s} ]
3556   }
3557 }
3558
3559
3560 \NewDocumentCommand \Trefis { O{} m } {
3561   \str_set:Nn \l_tmpa_str { #1 }
3562   \str_if_empty:NTF \l_tmpa_str {
3563     \msg_set:nnn{stex}{warning/deprecated}{
3564       \\\
3565       \c_backslash_str Trefis-is-deprecated! \\\
3566       Please~use~\c_backslash_str STEXsymbol{#2}![\exp_after:wN \stex_capitalize:n #2s]~inst
3567       \stex_path_to_string:N \g_stex_currentfile_seq)
3568       \\\ \\\
3569     }
3570     \msg_warning:nn{stex}{warning/deprecated}
3571     \STEXsymbol { #2 }![ \comp{\exp_after:wN \stex_capitalize:n #2s} ]
3572   } {
3573     \msg_set:nnn{stex}{warning/deprecated}{
3574       \\\
3575       \c_backslash_str Trefis-is-deprecated! \\\
3576       Please~use~\c_backslash_str STEXsymbol { #1 }[ \exp_after:wN \stex_capitalize:n #2s ]~
3577       \stex_path_to_string:N \g_stex_currentfile_seq)
3578       \\\ \\\
3579     }
3580     \msg_warning:nn{stex}{warning/deprecated}
3581     \STEXsymbol { #1 }![ \comp{\exp_after:wN \stex_capitalize:n #2s} ]
3582   }
3583 }
3584
3585 \NewDocumentCommand \trefii { O{} m m } {

```

```

3586 \str_set:Nn \l_tmpa_str { #1 }
3587 \str_if_empty:NTF \l_tmpa_str {
3588   \msg_set:nnn{stex}{warning/deprecated}{
3589     \\\
3590     \c_backslash_str trefii~is-deprecated! \\\
3591     Please~use~\c_backslash_str STEXsymbol{#2~#3}![#2~#3]~instead!~(in~file~
3592     \stex_path_to_string:N \g_stex_currentfile_seq)
3593     \\\ \\\
3594   }
3595   \msg_warning:nn{stex}{warning/deprecated}
3596   \STEXsymbol { #2~#3 }![ \comp{#2~#3} ]
3597 } {
3598   \msg_set:nnn{stex}{warning/deprecated}{
3599     \\\
3600     \c_backslash_str trefii~is-deprecated! \\\
3601     Please~use~\c_backslash_str STEXsymbol { #1 }[ #2~#3 ]~instead!~(in~file~
3602     \stex_path_to_string:N \g_stex_currentfile_seq)
3603     \\\ \\\
3604   }
3605   \msg_warning:nn{stex}{warning/deprecated}
3606   \STEXsymbol { #1 }![ \comp{#2~#3} ]
3607 }
3608 }
3609
3610 \NewDocumentCommand \trefiii { 0{ } m m m } {
3611   \str_set:Nn \l_tmpa_str { #1 }
3612   \str_if_empty:NTF \l_tmpa_str {
3613     \msg_set:nnn{stex}{warning/deprecated}{
3614       \\\
3615       \c_backslash_str trefiii~is-deprecated! \\\
3616       Please~use~\c_backslash_str STEXsymbol{#2~#3~#4}![#2~#3~#4]~instead!~(in~file~
3617       \stex_path_to_string:N \g_stex_currentfile_seq)
3618       \\\ \\\
3619     }
3620     \msg_warning:nn{stex}{warning/deprecated}
3621     \STEXsymbol { #2~#3~#4 }![ \comp{#2~#3~#4} ]
3622   } {
3623     \msg_set:nnn{stex}{warning/deprecated}{
3624       \\\
3625       \c_backslash_str trefiii~is-deprecated! \\\
3626       Please~use~\c_backslash_str STEXsymbol { #1 }[ #2~#3~#4 ]~instead!~(in~file~
3627       \stex_path_to_string:N \g_stex_currentfile_seq)
3628       \\\ \\\
3629     }
3630     \msg_warning:nn{stex}{warning/deprecated}
3631     \STEXsymbol { #1 }![ \comp{#2~#3~#4} ]
3632   }
3633 }
3634
3635
3636 \NewDocumentCommand \trefiis { 0{ } m m } {
3637   \str_set:Nn \l_tmpa_str { #1 }
3638   \str_if_empty:NTF \l_tmpa_str {
3639     \msg_set:nnn{stex}{warning/deprecated}{

```

```

3640     \
3641     \c_backslash_str trefiis~is-deprecated! \
3642     Please~use~\c_backslash_str STEXsymbol{#2~#3}![#2~#3s]~instead!~(in~file~
3643     \stex_path_to_string:N \g_stex_currentfile_seq)
3644     \
3645   }
3646   \msg_warning:nn{stex}{warning/deprecated}
3647   \STEXsymbol { #2~#3 }![ \comp{#2~#3s} ]
3648 } {
3649   \msg_set:nnn{stex}{warning/deprecated}{
3650     \
3651     \c_backslash_str trefiis~is-deprecated! \
3652     Please~use~\c_backslash_str STEXsymbol { #1 }[ #2~#3s ]~instead!~(in~file~
3653     \stex_path_to_string:N \g_stex_currentfile_seq)
3654     \
3655   }
3656   \msg_warning:nn{stex}{warning/deprecated}
3657   \STEXsymbol { #1 }![ \comp{#2~#3s} ]
3658 }
3659 }
3660
3661 \NewDocumentCommand \symvariant { O{} m O{0} m m } {
3662   \msg_set:nnn{stex}{warning/deprecated}{
3663     \
3664     \c_backslash_str symvariant~is-deprecated! \
3665     Please~use~\c_backslash_str notation[#4]{ #2 }~instead!~(in~file~
3666     \stex_path_to_string:N \g_stex_currentfile_seq)
3667     \
3668   }
3669   \msg_warning:nn{stex}{warning/deprecated}
3670
3671   \notation[variant=#4]{#2}{#5}
3672 }
3673
3674 \NewDocumentCommand \mixfixi { O{} m m m } {
3675   \msg_set:nnn{stex}{warning/deprecated}{
3676     \c_backslash_str mixfixi~is-fatally-deprecated!\
3677     Symbol:~\l__stex_term_highlight_uri_str\
3678     Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3679   }
3680   \msg_error:nn{stex}{warning/deprecated}
3681 }
3682
3683
3684 \NewDocumentCommand \infix {} {
3685   \msg_set:nnn{stex}{warning/deprecated}{
3686     \c_backslash_str infix~is-fatally-deprecated!\
3687     Symbol:~\l__stex_term_highlight_uri_str\
3688     Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3689   }
3690   \msg_error:nn{stex}{warning/deprecated}
3691 }
3692
3693 \let\iprec\infix

```

```

3694
3695 \NewDocumentCommand \inlineex { m } {
3696   \msg_set:nnn{stex}{warning/deprecated}{
3697     \c_backslash_str inlineex~is~deprecated!\\
3698     No~replacement~exists~yet.\\
3699     Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3700   }
3701   \msg_warning:nn{stex}{warning/deprecated}
3702   #1
3703 }
3704
3705
3706 \NewDocumentCommand \term { m } {
3707   \msg_set:nnn{stex}{warning/deprecated}{
3708     \c_backslash_str term~is~deprecated!\\
3709     No~replacement~exists~yet.\\
3710     Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3711   }
3712   \msg_warning:nn{stex}{warning/deprecated}
3713   #1
3714 }
3715
3716
3717 \NewDocumentCommand \Definame { 0{} m } {
3718   \stex_get_symbol:n { #2 }
3719   \str_set:Nx \l_tmpa_str {
3720     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
3721   }
3722   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
3723   \scalatex_if:TF {
3724     \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
3725       \l_tmpa_str
3726     }
3727   } {
3728     \@defemph {
3729       \exp_after:wN \stex_capitalize:n \l_tmpa_str
3730     } { \l_stex_get_symbol_uri_str }
3731   }
3732 }
3733
3734 \NewDocumentCommand \Definiendum { 0{} m m } {
3735   \stex_get_symbol:n { #2 }
3736   \str_set:Nx \l_tmpa_str {
3737     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
3738   }
3739   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
3740   \scalatex_if:TF {
3741     \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
3742       \l_tmpa_str
3743     }
3744   } {
3745     \@defemph {
3746       \exp_after:wN \stex_capitalize:n \l_tmpa_str
3747     } { \l_stex_get_symbol_uri_str }

```



```

3748 }
3749 }
3750
3751 \NewDocumentCommand \Symname { 0{} m }{
3752   \stex_symname_args:n { #1 }
3753   \stex_get_symbol:n { #2 }
3754   \str_set:Nx \l_tmpa_str {
3755     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
3756   }
3757   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
3758   \exp_args:NNx \use:nn
3759   \stex_invoke_symbol:n { { \l_stex_get_symbol_uri_str }![
3760     \exp_after:wN \stex_capitalize:n \l_tmpa_str
3761     \l_stex_symname_post_str
3762   ] }
3763 }
3764
3765
3766 \seq_gput_right:Nx \g_stex_smsmode_allowedenvs_seq { \tl_to_str:n { mhmodnl } }
3767 \seq_gput_right:Nx \g_stex_smsmode_allowedenvs_seq { \tl_to_str:n { modsig } }
3768 \tl_gput_right:Nn \g_stex_smsmode_allowedmacros_escape_tl {\gimport\syml\symii\symiii\symiv\
3769
3770 % omtex:
3771 \cs_new_protected:Npn \lec #1 {
3772   \strut\hfil\strut\hfill(#1)
3773 }
3774 \cs_new_protected:Npn \nlex #1 {
3775   \textcolor{green}{\s1 #1}}
3776 }
3777
3778
3779 </compat>

```