stex.sty: sTeX 2.0[*]

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

December 3, 2021

Abstract

TODO

1 Introduction

TODO

Contents

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}
$\mult{a}{b}$
```

```
a b
```

.

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

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

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 · b and a × b
```

.

Not using an explicit option with a semantic macro yields the first declared notation, unless changed[1].

---

[1] 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...
```

Multiplyingagain 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 Pholds for every $x \in A$

.

When using `*[n]`, after reading the provided (nth) 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:

.

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 themself.

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.
2 3

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.

sTeX 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

---

[2]EDNOTE: what about e.g. `\int _x\int _y\int _z f dx dy dz`?
[3]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 As operator precedence should be smaller than Bs 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, STEX 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, TEX only provides very restricted access to the file system, so we are forced to generate namespaces systematically in such a way that they reflect the physical location of the associated files, so that STEX can resolve them accordingly. Largely, users need not concern themselves with namespaces at all, but for completenesses sake, we describe how they are constructed:

- If `\begin{module}{Foo}` occurs in a file `/path/to/file/Foo[.⟨lang⟩].tex` which does not belong to an archive, the namespace is `file://path/to/file`.

- If the same statement occurs in a file `/path/to/file/bar[.⟨lang⟩].tex`, the namespace is `file://path/to/file/bar`.

In other words: outside of 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[1].

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, examplary `\importmodule`:

- `\importmodule{Foo}` outside of an archive refers to module `Foo` in the current namespace. Consequently, `Foo` must have been declared earlier in the same document or, if not, in a file `Foo[.⟨lang⟩].tex` in the same directory.

---

[1]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, is 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

`\sTeX`
`\stex`

both print this STEX logo.

`\stex_debug:n`

`\stex_debug:n {⟨message⟩}`

Logs ⟨*message*⟩, if the package option `debug` is used.

`\stex_kpsewhich:n`

`\stex_kpsewhich:n` executes kpsewhich and stores the return in `\l_stex_kpsewhich_return_str`. This does not require shell escaping.

`\stex_addtosms:n`

Adds the provided code to the `.sms`-file of the document.

3.1.1 SᴄALᴬTᴇX, LᴀTᴇXML and HTML Annotations

`\if@latexml`
`\latexml_if_p:`
`\latexml_if:T`
`\latexml_if:F`
`\latexml_if:TF`

LATEX2e and LATEX3 conditionals for LATEXML.

We have four macros for annotating generated HTML (via LaTeXML or SᴄᴀLATEX) with attributes:

`\stex_annotate:nnn`
`\stex_annotate_invisible:nnn`
`\stex_annotate_invisible:n`
	`\stex_annotate:nnn {⟨property⟩} {⟨resource⟩} {⟨content⟩}`

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.

stex_annotate_env	`\begin{stex_annotate_env}{⟨property⟩}{⟨resource⟩}`
	`⟨content⟩`
	`\end{stex_annotate_env}`
	behaves like `\stex_annotate:nnn {⟨property⟩} {⟨resource⟩} {⟨content⟩}`.

3.1.2 Languages

`\c_stex_languages_prop` `\c_stex_language_abbrevs_prop`

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

`\stex_path_from_string:Nn` `\stex_path_from_string:(NV|cn|cV)`	`\stex_path_from_string:Nn ⟨path-variable⟩ {⟨string⟩}`

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

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

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

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

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

9

| `\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/../..} & \\\hline
\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:` ⋆
`\stex_if_in_module:TF` ⋆

Conditional for whether we are currently in a module

`\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.

\stex_modules_compute_namespace:nN \stex_modules_compute_namespace:nN
{⟨*namespace*⟩} {⟨*path*⟩}

Computes the namespace for file ⟨*path*⟩ in repository with namespace ⟨*namespace*⟩ 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`.

\stex_modules_current_namespace:

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

module   \begin{module}[⟨*options*⟩]{⟨*name*⟩}
             Opens a new module with name ⟨*name*⟩.
             TODO document options.

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

@module   \begin{@module}[⟨*options*⟩]{⟨*name*⟩}
             Core functionality of the `module`-environment 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 `!⟨macro⟩` or `?{⟨symbolname⟩}`. In the first case, it stores the full URI in ⟨*macro*⟩; in the second case, it invokes the symbol ⟨*symbolname*⟩ in the selected module.

Test 6

```
\begin{module}{STEXModuleTest1}
\symdecl{foo}
\end{module}
\begin{module}{STEXModuleTest2}
\importmodule{STEXModuleTest1}
\symdecl{foo}
\end{module}
\begin{module}{STEXModuleTest3}
\importmodule{STEXModuleTest2}
\symdecl{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. STEX determines the module and its containing file by passing its arguments on to `\stex_import_module_-path:nn`.

16

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?Metatheory...
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?Metatheory...
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?aseqfromtovia, http://mathhub.info/sTeX?Meta...
http://mathhub.info/sTeX?Metatheory?module-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}«

.

`\stex_import_module_uri:nn`   `\stex_import_module_uri:nn {⟨archive-ID⟩} {⟨module-path⟩}`

Determines the URI of a module by splitting ⟨*module-path*⟩ into ⟨*path*⟩?⟨*name*⟩. If ⟨*module-path*⟩ does *not* contain a ?-character, we consider it to be the ⟨*name*⟩, and ⟨*path*⟩ to be empty.

If ⟨*archive-ID*⟩ is empty, it is automatically set to the ID of the current archive (if one exists).

1. If ⟨*archive-ID*⟩ is empty:

    (a) If ⟨*path*⟩ is empty, then ⟨*name*⟩ must have been declared earlier in the same file and retrievable from `\g_stex_modules_in_file_seq`, or a file with name ⟨*name*⟩.⟨*lang*⟩.`tex` must exist in the same folder, containing a module ⟨*name*⟩.

    That module should have the same namespace as the current one.

    (b) If ⟨*path*⟩ is not empty, it must point to the relative path of the containing file as well as the namespace.

2. Otherwise:

    (a) If ⟨*path*⟩ is empty, then ⟨*name*⟩ must have been declared earlier in the same file and retrievable from `\g_stex_modules_in_file_seq`, or a file with name ⟨*name*⟩.⟨*lang*⟩.`tex` must exist in the top `source` folder of the archive, containing a module ⟨*name*⟩.

    That module should lie directly in the namespace of the archive.

    (b) If ⟨*path*⟩ 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 `\stex_require_module:nn` on the `source` directory of the archive to find the file.

`\stex_import_require_module:nnnn`   `{⟨ns⟩} {⟨archive-ID⟩} {⟨path⟩} {⟨name⟩}`

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.

`\g_stex_module_files_prop`
`\g_stex_modules_in_file_seq`

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 - `\input`s are considered the same file).

`\stex_activate_module:n`   Activate the module with the provided URI; i.e. executes all macro code of the module's `content`-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 sTEX, 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 sTEX 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\Nat}{x\geq0}`.

`\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.

`\_stex_term_math_arg:nnn`	`\stex_term_arg:nnn`⟨*int*⟩⟨*prec*⟩⟨*body*⟩

Annotates ⟨*body*⟩ as the ⟨*int*⟩th argument of the current `OMA` or `OMBIND`, with (downwards) argument precedence ⟨*prec*⟩.

`\_stex_term_math_assoc_arg:nnnn`	`\stex_term_arg:nnn`⟨*int*⟩⟨*prec*⟩⟨*notation*⟩⟨*body*⟩

Annotates ⟨*body*⟩ as the ⟨*int*⟩th (associative) *sequence* argument (as comma-separated list of terms) of the current `OMA` or `OMBIND`, with (downwards) argument precedence ⟨*prec*⟩ and associative notation ⟨*notation*⟩.

`\infprec` `\neginfprec`	Maximal and minimal notation precedences.

`\dobrackets`	`\dobrackets` {⟨*body*⟩}

Puts ⟨*body*⟩ in parentheses; scaled if in display mode unscaled otherwise. Uses the current sTEX brackets (by default (and)), which can be changed temporarily using `\withbrackets`.

`\withbrackets`	`\withbrackets` ⟨*left*⟩ ⟨*right*⟩ {⟨*body*⟩}

Temporarily (i.e. within ⟨*body*⟩) sets the brackets used by sTEX for automated bracketing (by default (and)) to ⟨*left*⟩ and ⟨*right*⟩.

Note that ⟨*left*⟩ and ⟨*right*⟩ need to be allowed after `\left` and `\right` in display-mode.

Test 14

```
\begin{module}{MathTest1}
\importmodule{Foo}
\notation[foo, prec=500;20x20x20]{bar}{\comp\langle {#1 ^ {#2}}_{#3} \comp\rangle }
$\bar abc$ and $\bar[foo] abc$.

\end{module}
```

> **Module** 3.15[MathTest1]
> ⟨$a^b{}_c$⟩ and ⟨$a^b{}_c$⟩.

.

Test 15

```
\begin{module}{MathTest2}
\importmodule{Foo}
\notation[foo, prec=500;20x20x20]{foobar}{\comp\langle #1 \comp\mid [ #2 ]^{#3} \comp\rangle }{ {#1}_{\com
$\foobar a{b,c,d,e,f}g$ and $\foobar[foo] a{b,c}g$ and $\foobar abc$

\symdecl[args=a]{plus}
\symdecl[args=a]{mult}
\notation[prec=50]{plus}{#1}{#1 \comp+ #2}
\notation[prec=100]{mult}{#1}{#1 \comp\cdot #2}
$\plus{a,\mult{b,c}}$ and $\mult{a,\plus{\frac ab,\frac ac}}$
\[\plus{a,\mult{b,c}}\text{ and }\mult{a,\plus{\frac ab,\frac ac}}\]
$\displaystyle \plus{a,\mult{b,c}}$ and
\withbrackets[]{$\displaystyle
\mult{a,\plus{\frac ab,\frac ac}}$}
\end{module}
```

> **Module** 3.16[MathTest2]
> $\langle a|[b_{:c_{:d_{:e_{:f}}}}]^g\rangle$ and $\langle a|[b_{:c}]^g\rangle$ and $\langle a|[b]^c\rangle$
> $a+(b\cdot c)$ and $a\cdot\frac{a}{b}+\frac{a}{c}$
>
> $$a+(b\cdot c) \text{ and } a\cdot\frac{a}{b}+\frac{a}{c}$$
>
> $a+(b\cdot c)$ and $a\cdot\frac{a}{b}+\frac{a}{c}$

.

`\stex_term_custom:nn`

`\stex_term_custom:nn{`⟨URI⟩`}{`⟨args⟩`}`

Implements custom one-time notation. Invoked by `\stex_invoke_symbol:n` in text mode, or if followed by `*` in math mode, or whenever followed by `!`.

Test 16

```
 \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}
```

> **Module** 3.17[TextTest]
> some aand some band also some chere.
> some a and some b and also some c here.
>
> or just some c
> bar
> or first b, then c, and finally a

.

`\stex_highlight_term:nn`

`\stex_highlight_term:nn{`⟨URI⟩`}{`⟨args⟩`}`

Establishes a context for `\comp`. Stores the URI in a variable so that `\comp` knows which symbol governs the current notation.

`\comp`
`\@comp`
`\@defemph`

`\comp{`⟨args⟩`}`

Marks ⟨args⟩ as a notation component of the current symbol for highlighting, linking, etc.

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`)

\STEXinvisible 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.

\ellipses TODO

3.6   Structural Features

symboldoc \begin{⟨*symboldoc*⟩}{⟨*symbols*⟩} ⟨*text*⟩ \end{⟨*symboldoc*⟩}

Declares ⟨*text*⟩ to be a (natural language, encyclopaedic) description of {⟨*symbols*⟩} (a comma separated list of symbol identifiers).

3.6.1   Structures

structure TODO

Test 17

```
\begin{module}{StructureTest1}
\begin{structure}[name=Magma]{magma}
\symdef{universe}{\comp M}
\symdef[args=2]{op}{#1 \comp\circ #2}
$\isa{\op ab}\universe$
\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 ! {{#1 \comp+ #2 }}
]{mM}
\notation[op = U]{mM/universe}{\comp U}
\notation[op = +]{mM/op}{#1 \comp+ #2}

Test: $\mM{op}ab$

Test2: $\mM{}$
\end{module}
```

Module 3.18[StructureTest1]
$a \circ b : M$
file://home/jazzpirate/work/Software/ext/sTeX/sty/stex-master/stextest?StructureTest1/Magma-feature?universe,file://home/jazzpi 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: $\langle U,+\rangle$

.

4   Implementation

4.1   The S**T**E**X** document class

1 ⟨*cls⟩
2 \RequirePackage{expl3,l3keys2e}

25

```
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,ltxcmds}
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 LaTeXML **and** SCALATEX

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

We add the namespace abbreviation `ns:stex="http://kwarc.info/ns/sTeX"` to SCALATEX:

```
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_ampersand_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:nnn`
`\stex_annotate_invisible:n`
`\stex_annotate_invisible:nnn`

We define four macros for introducing attributes in the HTML output. The definitions depend on the "backend" used (LaTeXML, SCALATEX, pdflatex).

The pdflatex-macros largely do nothing; the SCALATEX-implementations are pretty clear in what they do, the LaTeXML-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 LaTeX3-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
\c__stex_path_up_str

. and .., respectively.

```
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.*)

Canonical\stex_path_if_absolute_p:N
\stex_path_if_absolute:N*TF*

```
278 \prg_new_conditional:Nnn \stex_path_if_absolute:N {p, T, F, TF} {
279   \seq_if_empty:NTF #1 {
280     \prg_return_false:
281   }{
282     \seq_get_left:NN #1 \l_tmpa_tl
283     \str_if_empty:NTF \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

<code>\c_stex_pwd_seq</code>
<code>\c_stex_pwd_str</code>

```
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 sTEX-purposes.

<code>\g__stex_files_stack</code> keeps track of file changes

```
306 \seq_gclear_new:N\g__stex_files_stack
```

(*End definition for* `\g__stex_files_stack`*.*)

<code>\c_stex_mainfile_seq</code>

```
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.*)

<code>\g_stex_currentfile_seq</code> 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 }
```

36

(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_\l_tmpa_str _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 \stex_in_repository:nn {
486   \str_set:Nx \l_tmpa_str { #1 }
487   \cs_set:Npn \l_tmpa_cs ##1 { #2 }
488   \str_if_empty:NTF \l_tmpa_str {
489     \exp_args:Ne \l_tmpa_cs{
490       \prop_item:Nn \l_stex_current_repository_prop { id }
491     }
492   }{
493     \stex_require_repository:n \l_tmpa_str
494     \str_set:Nx \l_tmpa_str { #1 }
495     \exp_args:Nne \use:nn {
```

```
496        \stex_set_current_repository:n \l_tmpa_str
497        \exp_args:Nx \l_tmpa_cs{\l_tmpa_str}
498     }{
499        \stex_set_current_repository:n {
500         \prop_item:Nn \l_stex_current_repository_prop { id }
501        }
502     }
503    }
504 }
505
506 \cs_new_protected:Nn \inputref:nn {
507    \stex_in_repository:nn {#1} {
508       \ifinputref
509       \input{ \c_stex_mathhub_str / ##1 / source / #2 }
510       \else
511         \inputreftrue
512         \input{ \c_stex_mathhub_str / ##1 / source / #2 }
513         \inputreffalse
514       \fi
515    }
516 }
517 \NewDocumentCommand \inputref { O{} m}{
518    \inputref:nn{ #1 }{ #2 }
519 }
```

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

\mhpath

```
520    \def \mhpath #1 #2 {
521       \exp_args:Ne \str_if_eq:nnTF{#1}{}{
522          \c_stex_mathhub_str /
523             \prop_item:Nn \l_stex_current_repository_prop { id }
524             / source / #2
525       }{
526          \c_stex_mathhub_str / #1 / source / #2
527       }
528    }
```

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

\libinput

```
529 \cs_new_protected:Npn \libinput #1 {
530    \prop_get:NnNF \l_stex_current_repository_prop {id} \l_tmpa_str {
531       \msg_set:nnn{stex}{error/norepository}{
532          \c_backslash_str libinput~needs~to~be~called~in~an~archive
533       }
534       \msg_error:nn{stex}{error/norepository}
535    }
536    \bool_set_false:N \l_tmpa_bool
537    \tl_clear:N \l_tmpa_tl
538    \seq_set_eq:NN \l_tmpa_seq \c_stex_mathhub_seq
539    \seq_set_split:NnV \l_tmpb_seq / \l_tmpa_str
540    \seq_pop_right:NN \l_tmpb_seq \l_tmpa_str
541    \seq_pop_left:NNT \l_tmpb_seq \l_tmpb_str {
542       \seq_put_right:No \l_tmpa_seq \l_tmpb_str
```

```
543    \IfFileExists{ \stex_path_to_string:N \l_tmpa_seq
544      / meta-inf / lib / #1.tex}{
545        \bool_set_true:N \l_tmpa_bool
546        \tl_put_right:Nx \l_tmpa_tl {
547          \exp_not:N \input { \stex_path_to_string:N \l_tmpa_seq
548          / meta-inf / lib / #1.tex}
549        }
550      }{}
551    }
552    \IfFileExists{ \stex_path_to_string:N \l_tmpa_seq
553      / \l_tmpa_str / lib / #1.tex
554    }{
555      \bool_set_true:N \l_tmpa_bool
556      \tl_put_right:Nx \l_tmpa_tl {
557        \exp_not:N \input { \stex_path_to_string:N \l_tmpa_seq
558        / \l_tmpa_str / lib / #1.tex}
559      }
560    }{}
561    \bool_if:NF \l_tmpa_bool {
562      \msg_set:nnn{stex}{error/nofile}{
563        \c_backslash_str libinput~no~file~#1.tex~found!
564      }
565      \msg_error:nn{stex}{error/nofile}
566    }
567    \l_tmpa_tl
568 }
```

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

4.5   Module System

```
569 ⟨@@=stex_module⟩
```

<code>\l_stex_current_module_prop</code>

```
570 \prop_new:N \l_stex_current_module_prop
```

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

<code>stex_if_in_module_p:</code>
<code>stex_if_in_module:<u>TF</u></code>

```
571 \prg_new_conditional:Nnn \stex_if_in_module: {p, T, F, TF} {
572    \prop_if_empty:NTF \l_stex_current_module_prop
573      \prg_return_false: \prg_return_true:
574 }
```

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

<code>stex_if_module_exists_p:n</code>
<code>stex_if_module_exists:n<u>TF</u></code>

```
575 \prg_new_conditional:Nnn \stex_if_module_exists:n {p, T, F, TF} {
576    \prop_if_exist:cTF { c_stex_module_#1_prop }
577      \prg_return_true: \prg_return_false:
578 }
```

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

```
579 \cs_new_protected:Nn \stex_add_to_current_module:n {
580   \prop_get:NnN \l_stex_current_module_prop { content } \l_tmpa_tl
581   \tl_put_right:Nn \l_tmpa_tl { #1 }
582   \prop_put:Nno \l_stex_current_module_prop { content } { \l_tmpa_tl }
583 }
584 \cs_new_protected:Npn \STEXexport #1 {
585   #1
586   \stex_add_to_current_module:n { #1 }
587   \stex_smsmode_set_codes:
588 }
```

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

```
589 \cs_new_protected:Nn \stex_add_constant_to_current_module:n {
590   \str_set:Nx \l_tmpa_str { #1 }
591   \prop_get:NnN \l_stex_current_module_prop { constants } \l_tmpa_seq
592   \seq_put_right:No \l_tmpa_seq { \l_tmpa_str }
593   \prop_put:Nno \l_stex_current_module_prop { constants } \l_tmpa_seq
594 }
```

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

```
595 \cs_new_protected:Nn \stex_add_import_to_current_module:n {
596   \str_set:Nx \l_tmpa_str { #1 }
597   \prop_get:NnN \l_stex_current_module_prop { imports } \l_tmpa_seq
598   \seq_put_right:No \l_tmpa_seq { \l_tmpa_str }
599   \prop_put:Nno \l_stex_current_module_prop { imports } \l_tmpa_seq
600 }
```

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

stores its return values in:

```
601 \str_new:N \l_stex_modules_ns_str

602 \cs_new_protected:Nn \stex_modules_compute_namespace:nN {
603   \str_set:Nx \l_tmpa_str { #1 }
604   \seq_set_eq:NN \l_tmpa_seq #2
605   % split off file extension
606   \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
607   \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
608   \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
609   \seq_put_right:No \l_tmpa_seq \l_tmpb_str
610
611   \bool_set_true:N \l_tmpa_bool
612   \bool_while_do:Nn \l_tmpa_bool {
613     \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
614     \exp_args:No \str_case:nnTF { \l_tmpb_str } {
615       {source} { \bool_set_false:N \l_tmpa_bool }
```

```
616    }{}{
617      \seq_if_empty:NT \l_tmpa_seq {
618        \bool_set_false:N \l_tmpa_bool
619      }
620    }
621  }
622
623  \seq_if_empty:NTF \l_tmpa_seq {
624    \str_set_eq:NN \l_stex_modules_ns_str \l_tmpa_str
625  }{
626    \str_set:Nx \l_stex_modules_ns_str {
627      \l_tmpa_str/\stex_path_to_string:N \l_tmpa_seq
628    }
629  }
630 }
```

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

```
631 \cs_new_protected:Nn \stex_modules_current_namespace: {
632    \prop_get:NnNTF \l_stex_current_repository_prop { ns } \l_tmpa_str {
633      \stex_modules_compute_namespace:nN \l_tmpa_str \g_stex_currentfile_seq
634    }{
635      % split off file extension
636      \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
637      \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
638      \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
639      \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
640      \seq_put_right:No \l_tmpa_seq \l_tmpb_str
641      \str_set:Nx \l_stex_modules_ns_str {
642        file:/\stex_path_to_string:N \l_tmpa_seq
643      }
644    }
645 }
```

(*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

```
646 \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`

```
647 \NewDocumentCommand \STEXModule { m } {
648    \exp_args:NNx \str_set:Nn \l_tmpa_str { #1 }
649    \int_set:Nn \l_tmpa_int { \str_count:N \l_tmpa_str }
650    \tl_set:Nn \l_tmpa_tl {
651      \msg_set:nnn{stex}{error/unknownmodule}{
652        No~module~#1~found!
653      }
654      \msg_error:nn{stex}{error/unknownmodule}
655    }
```

41

```
656    \seq_map_inline:Nn \l_stex_all_modules_seq {
657      \str_set:Nn \l_tmpb_str { ##1 }
658      \str_if_eq:eeT { \l_tmpa_str } {
659        \str_range:Nnn \l_tmpb_str { -\l_tmpa_int } { -1 }
660      } {
661        \seq_map_break:n {
662          \tl_set:Nn \l_tmpa_tl {
663            \stex_invoke_module:n { ##1 }
664          }
665        }
666      }
667    }
668    \l_tmpa_tl
669 }
670
671 \cs_new_protected:Nn \stex_invoke_module:n {
672    \stex_debug:n{Invoking~module~#1}
673    \peek_charcode_remove:NTF ! {
674      \__stex_module_invoke_uri:nN { #1 }
675    } {
676      \peek_charcode_remove:NTF ? {
677        \__stex_module_invoke_symbol:nn { #1 }
678      } {
679        \msg_set:nnn{stex}{error/syntax}{
680          Syntax~error:~?~or~!~expected~after~
681          \c_backslash_str STEXModule{#1}
682        }
683        \msg_error:nn{stex}{error/syntax}
684      }
685    }
686 }
687
688 \cs_new_protected:Nn \__stex_module_invoke_uri:nN {
689    \str_set:Nn #2 { #1 }
690 }
691
692 \cs_new_protected:Nn \__stex_module_invoke_symbol:nn {
693    \stex_invoke_symbol:n{#1?#2}
694 }
```

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

module module arguments:

```
695 \keys_define:nn { stex / module } {
696    title         .tl_set_x:N  = \l_stex_module_title_str ,
697    ns            .tl_set_x:N  = \l_stex_module_ns_str ,
698    lang          .tl_set_x:N  = \l_stex_module_lang_str ,
699    sig           .tl_set_x:N  = \l_stex_module_sig_str ,
700    creators      .tl_set_x:N  = \l_stex_module_creators_str ,
701    contributors  .tl_set_x:N  = \l_stex_module_contributors_str ,
702    meta          .tl_set_x:N  = \l_stex_module_meta_str
703 }
704
```

```
705 % module parameters here? In the body?
706
707 \cs_new_protected:Nn \__stex_module_args:n {
708   \str_clear:N \l_stex_module_title_str
709   \str_clear:N \l_stex_module_ns_str
710   \str_clear:N \l_stex_module_lang_str
711   \str_clear:N \l_stex_module_sig_str
712   \str_clear:N \l_stex_module_creators_str
713   \str_clear:N \l_stex_module_contributors_str
714   \str_clear:N \l_stex_module_meta_str
715   \keys_set:nn { stex / module } { #1 }
716   \exp_args:NNo \str_set:Nn \l_stex_module_title_str
717     \l_stex_module_title_str
718   \exp_args:NNo \str_set:Nn \l_stex_module_ns_str
719     \l_stex_module_ns_str
720   \exp_args:NNo \str_set:Nn \l_stex_module_lang_str
721     \l_stex_module_lang_str
722   \exp_args:NNo \str_set:Nn \l_stex_module_sig_str
723     \l_stex_module_sig_str
724   \exp_args:NNo \str_set:Nn \l_stex_module_meta_str
725     \l_stex_module_meta_str
726   \exp_args:NNo \str_set:Nn \l_stex_module_creators_str
727     \l_stex_module_creators_str
728   \exp_args:NNo \str_set:Nn \l_stex_module_contributors_str
729     \l_stex_module_contributors_str
730 }
```

__stex_module_begin_module: implements \begin{module}

```
731 \cs_new_protected:Nn \__stex_module_begin_module: {
732   % Nested module?
733   \stex_if_in_module:TF {
734     % Nested module
735     \prop_get:NnN \l_stex_current_module_prop
736       { ns } \l_stex_module_ns_str
737     \str_set:Nx \l_stex_module_name_str {
738       \prop_item:Nn \l_stex_current_module_prop
739         { name } / \l_stex_module_name_str
740     }
741   }{
742     % not nested:
743     \str_if_empty:NT \l_stex_module_ns_str {
744       \stex_modules_current_namespace:
745       \str_set_eq:NN \l_stex_module_ns_str \l_stex_modules_ns_str
746       \exp_args:NNNo \seq_set_split:Nnn \l_tmpa_seq
747         / {\l_stex_module_ns_str}
748       \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
749       \str_if_eq:NNT \l_tmpa_str \l_stex_module_name_str {
750         \str_set:Nx \l_stex_module_ns_str {
751           \stex_path_to_string:N \l_tmpa_seq
752         }
753       }
754     }
755   }
756
```

```latex
757   % language
758   \str_if_empty:NT \l_stex_module_lang_str {
759     \seq_get_right:NN \g_stex_currentfile_seq \l_tmpa_str
760     \seq_set_split:NnV \l_tmpa_seq . \l_tmpa_str
761     \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str % .tex
762     \seq_pop_left:NN \l_tmpa_seq \l_tmpa_str % <filename>
763     \seq_if_empty:NF \l_tmpa_seq { %remaining element should be language
764       \stex_debug:n {Language~\l_stex_module_lang_str~
765         inferred~from~file~name}
766       \seq_pop_left:NN \l_tmpa_seq \l_stex_module_lang_str
767     }
768   }
769
770   \str_if_empty:NF \l_stex_module_lang_str {
771     \prop_get:NVNTF \c_stex_languages_prop \l_stex_module_lang_str
772       \l_tmpa_str {
773         \ltx@ifpackageloaded{babel}{
774           \exp_args:Nx \selectlanguage { \l_tmpa_str }
775         }{}
776       } {
777         \msg_set:nnn{stex}{error/unknownlanguage}{
778           Unknown~language~\l_tmpa_str
779         }
780         \msg_error:nn{stex}{error/unknownlanguage}
781       }
782   }
783
784   % signature
785   \str_if_empty:NTF \l_stex_module_sig_str {
786     \str_clear:N \l_tmpa_str
787     \seq_clear:N \l_tmpa_seq
788     \tl_clear:N \l_tmpa_tl
789     \exp_args:NNx \prop_set_from_keyval:Nn \l_stex_current_module_prop {
790       name      = \l_stex_module_name_str ,
791       ns        = \l_stex_module_ns_str ,
792       imports   = \exp_not:o { \l_tmpa_seq } ,
793       constants = \exp_not:o { \l_tmpa_seq } ,
794       content   = \exp_not:o { \l_tmpa_tl }  ,
795       file      = \exp_not:o { \g_stex_currentfile_seq } ,
796       lang      = \l_stex_module_lang_str ,
797       sig       = \l_stex_module_sig_str ,
798       meta      = \l_stex_module_meta_str
799     }
800   }{
801     \str_if_empty:NT \l_stex_module_lang_str {
802       \msg_set:nnn{stex}{error/siglanguage}{
803         Module~\l_stex_module_ns_str?\l_stex_module_name_str~
804         declares~signature~\l_stex_module_sig_str,~but~does~not~
805         declare~its~language
806       }
807       \msg_error:nn{stex}{error/siglanguage}
808     }
809
810     \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
```

```
811    \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
812    \seq_set_split:NnV \l_tmpb_seq . \l_tmpa_str
813    \seq_pop_right:NN \l_tmpb_seq \l_tmpa_str % .tex
814    \seq_pop_left:NN \l_tmpb_seq \l_tmpa_str % <filename>
815    \str_set:Nx \l_tmpa_str {
816      \stex_path_to_string:N \l_tmpa_seq /
817      \l_tmpa_str . \l_stex_module_sig_str .tex
818    }
819    \IfFileExists \l_tmpa_str {
820      \exp_args:No \stex_in_smsmode:nn { \l_tmpa_str } {
821        \seq_clear:N \l_stex_all_modules_seq
822        \prop_clear:N \l_stex_current_module_prop
823        \stex_debug:n{Loading~signature~\l_tmpa_str}
824        \input { \l_tmpa_str }
825      }
826    }{
827      \msg_set:nnn{stex}{error/modulemissing}{
828        No~file~for~signature~module~\l_tmpa_str~found
829      }
830      \msg_error:nn{stex}{error/modulemissing}
831    }
832    \stex_activate_module:n {
833      \l_stex_module_ns_str ? \l_stex_module_name_str
834    }
835    \prop_set_eq:Nc \l_stex_current_module_prop {
836      c_stex_module_
837      \l_stex_module_ns_str ?
838      \l_stex_module_name_str
839      _prop
840    }
841  }
842
843  % metatheory
844  \str_if_empty:NT \l_stex_module_meta_str {
845    \str_set:Nx \l_stex_module_meta_str {
846      \c_stex_metatheory_ns_str ? Metatheory
847    }
848  }
849
850
851  \stex_debug:n{
852    New~module:\\
853    Namespace:~\l_stex_module_ns_str\\
854    Name:~\l_stex_module_name_str\\
855    Language:~\l_stex_module_lang_str\\
856    Signature:~\l_stex_module_sig_str\\
857    Metatheory:~\l_stex_module_meta_str\\
858    File:~\stex_path_to_string:N \g_stex_currentfile_seq
859  }
860
861  \seq_put_right:Nx \l_stex_all_modules_seq {
862    \l_stex_module_ns_str ? \l_stex_module_name_str
863  }
864
```

```
865     \seq_gput_right:Nx  \g_stex_modules_in_file_seq
866         { \l_stex_module_ns_str ? \l_stex_module_name_str }
867
868     \stex_if_smsmode:TF {
869       \stex_smsmode_set_codes:
870     } {
871       \begin{stex_annotate_env} {theory} {
872         \l_stex_module_ns_str ? \l_stex_module_name_str
873       }
874
875       \stex_annotate_invisible:nnn{header}{} {
876         \stex_annotate:nnn{language}{ \l_stex_module_lang_str }{}
877         \stex_annotate:nnn{signature}{ \l_stex_module_sig_str }{}
878         \str_if_eq:VnF \l_stex_module_meta_str {NONE} {
879           \stex_annotate:nnn{metatheory}{ \l_stex_module_meta_str }{}
880         }
881       }
882     }
883
884     \str_if_eq:VnF \l_stex_module_meta_str {NONE} {
885       \exp_args:Nx \STEXexport{
886         \stex_activate_module:n {\l_stex_module_meta_str}
887       }
888     }
889     % TODO: Inherit metatheory for nested modules?
890   }
891   \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}

```
892 \iffalse \begin{stex_annotate_env} \fi %^^A make syntax highlighting work again
893 \cs_new_protected:Nn \__stex_module_end_module: {
894   \str_set:Nx \l_tmpa_str {
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   %^^A \prop_new:c { \l_tmpa_str }
901   \prop_gset_eq:cN { \l_tmpa_str } \l_stex_current_module_prop
902   \stex_debug:n{Closing~module~\prop_item:Nn \l_stex_current_module_prop { name }}
903   \stex_if_smsmode:TF {
904     \exp_args:Nx \stex_addtosms:n {
905       \prop_gset_from_keyval:cn {
906         c_stex_module_
907         \prop_item:Nn \l_stex_current_module_prop { ns } ?
908         \prop_item:Nn \l_stex_current_module_prop { name }
909         _prop
910       } {
911         name      = \prop_item:cn { \l_tmpa_str } { name } ,
912         ns        = \prop_item:cn { \l_tmpa_str } { ns } ,
913         imports   = \prop_item:cn { \l_tmpa_str } { imports } ,
914         constants = \prop_item:cn { \l_tmpa_str } { constants } ,
```

```
915        content   = \prop_item:cn { \l_tmpa_str } { content } ,
916        file      = \prop_item:cn { \l_tmpa_str } { file } ,
917        lang      = \prop_item:cn { \l_tmpa_str } { lang } ,
918        sig       = \prop_item:cn { \l_tmpa_str } { sig } ,
919        meta      = \prop_item:cn { \l_tmpa_str } { meta }
920      }
921    }
922  }{
923    \end{stex_annotate_env}
924  }
925 }
```

(*End definition for* __stex_module_end_module:.)

@module The core environment, with no header

```
926 \NewDocumentEnvironment { @module } { O{} m } {
927   \str_set:Nx \l_stex_module_name_str { #2 }
928   \par
929   \__stex_module_args:n { #1 }
930   \__stex_module_begin_module:
931 } {
932   \__stex_module_end_module:
933 }
```

\stex_modules_heading: Code for document headers

```
934 \cs_if_exist:NTF \thesection {
935   \newcounter{module}[section]
936 }{
937   \newcounter{module}
938 }
939
940 \bool_if:NT \c_stex_showmods_bool {
941   \latexml_if:F { \RequirePackage{mdframed} }
942 }
943
944 \cs_new_protected:Nn \stex_modules_heading: {
945   \stepcounter{module}
946   \par
947   \bool_if:NT \c_stex_showmods_bool {
948     \noindent{\textbf{Module} ~
949       \cs_if_exist:NT \thesection {\thesection.}
950       \themodule ~ [\l_stex_module_name_str]
951     }
952     % TODO references
953     % \sref@label@id{Module \thesection.\themodule [\module@name]}%
954     \str_if_empty:NTF \l_stex_module_title_str {
955     }{
956       \quad(\l_stex_module_title_str)\hfill
957     }\par
958   }
959 }
```

(*End definition for* \stex_modules_heading:. *This function is documented on page*)
      Finally:

```
960 \NewDocumentEnvironment { module } { O{} m } {
961   \bool_if:NT \c_stex_showmods_bool {
962     \begin{mdframed}
963   }
964   \begin{@module}[#1]{#2}
965   \stex_modules_heading:
966 }{
967   \end{@module}
968   \bool_if:NT \c_stex_showmods_bool {
969     \end{mdframed}
970   }
971 }
```

4.5.2 SMS Mode

```
972 ⟨@@=stex_smsmode⟩
```

```
973 \tl_new:N \g_stex_smsmode_allowedmacros_tl
974 \tl_new:N \g_stex_smsmode_allowedmacros_escape_tl
975 \seq_new:N \g_stex_smsmode_allowedenvs_seq
976
977 \tl_set:Nn \g_stex_smsmode_allowedmacros_tl {
978   \makeatletter
979   \makeatother
980   \ExplSyntaxOn
981   \ExplSyntaxOff
982 }
983
984 \tl_set:Nn \g_stex_smsmode_allowedmacros_escape_tl {
985   \symdef
986   \importmodule
987   \notation
988   \symdecl
989   \STEXexport
990 }
991
992 \exp_args:NNx \seq_set_from_clist:Nn \g_stex_smsmode_allowedenvs_seq {
993   \tl_to_str:n {
994     module,
995     @module
996   }
997 }
```

(*End definition for* \g_stex_smsmode_allowedmacros_tl, \g_stex_smsmode_allowedmacros_escape_tl, *and* \g_stex_smsmode_allowedenvs_seq. *These variables are documented on page 15.*)

```
 998 \bool_new:N \g__stex_smsmode_bool
 999 \bool_set_false:N \g__stex_smsmode_bool
1000 \prg_new_conditional:Nnn \stex_if_smsmode: { p, T, F, TF } {
1001   \bool_if:NTF \g__stex_smsmode_bool \prg_return_true: \prg_return_false:
1002 }
```

(*End definition for* \stex_if_smsmode:TF. *This function is documented on page 16.*)

`\__stex_smsmode_if_catcodes_p:`
`\__stex_smsmode_if_catcodes:`<u>*TF*</u> Checks whether the SMS mode category code scheme is active.

```
1003 \bool_new:N \g__stex_smsmode_catcode_bool
1004 \bool_set_false:N \g__stex_smsmode_catcode_bool
1005 \prg_new_conditional:Nnn \__stex_smsmode_if_catcodes: { p, T, F, TF } {
1006   \bool_if:NTF \g__stex_smsmode_catcode_bool
1007     \prg_return_true: \prg_return_false:
1008 }
```

(*End definition for* `\__stex_smsmode_if_catcodes:TF.`)

`\stex_smsmode_set_codes:`

```
1009 \cs_new_protected:Nn \stex_smsmode_set_codes: {
1010   \stex_if_smsmode:T {
1011     \__stex_smsmode_if_catcodes:F {
1012       \bool_gset_true:N \g__stex_smsmode_catcode_bool
1013       \exp_after:wN \char_gset_active_eq:NN
1014         \c_backslash_str \__stex_smsmode_cs:
1015       \tex_global:D \char_set_catcode_active:N \\
1016       \tex_global:D \char_set_catcode_other:N $
1017       \tex_global:D \char_set_catcode_other:N ^
1018       \tex_global:D \char_set_catcode_other:N _
1019       \tex_global:D \char_set_catcode_other:N &
1020       \tex_global:D \char_set_catcode_other:N ##
1021     }
1022   }
1023 } \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.

```
1024 \cs_new_protected:Nn \__stex_smsmode_unset_codes: {
1025   \__stex_smsmode_if_catcodes:T {
1026     \bool_gset_false:N \g__stex_smsmode_catcode_bool
1027     \exp_after:wN \tex_global:D \exp_after:wN
1028       \char_set_catcode_escape:N \c_backslash_str
1029     \tex_global:D \char_set_catcode_math_toggle:N $
1030     \tex_global:D \char_set_catcode_math_superscript:N ^
1031     \tex_global:D \char_set_catcode_math_subscript:N _
1032     \tex_global:D \char_set_catcode_alignment:N &
1033     \tex_global:D \char_set_catcode_parameter:N ##
1034   }
1035 } \iffalse $ \fi % to make syntax highlighting work again
```

(*End definition for* `\__stex_smsmode_unset_codes:.`)

`\stex_in_smsmode:nn`

```
1036 \cs_new_protected:Nn \stex_in_smsmode:nn {
1037   \vbox_set:Nn \l_tmpa_box {
1038     \bool_set_eq:cN { l__stex_smsmode_#1_bool } \g__stex_smsmode_bool
1039     \bool_gset_true:N \g__stex_smsmode_bool
1040     \stex_smsmode_set_codes:
1041     #2
1042     \bool_gset_eq:Nc \g__stex_smsmode_bool { l__stex_smsmode_#1_bool }
1043     \stex_if_smsmode:F {
```

```
1044                \__stex_smsmode_unset_codes:
1045            }
1046        }
1047        \box_clear:N \l_tmpa_box
1048 }
```

(*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:

```
1049 \cs_new_protected:Nn \__stex_smsmode_cs: {
1050    \str_clear:N \l_tmpa_str
1051    \peek_analysis_map_inline:n {
1052        % #1: token (one expansion)
1053        % #2: charcode
1054        % #3 catcode
1055        \token_if_eq_charcode:NNTF ##3 B {
1056            % token is a letter
1057            \exp_args:NNo \str_put_right:Nn \l_tmpa_str { ##1 }
1058        } {
1059            \str_if_empty:NTF \l_tmpa_str {
1060                % we don't allow (or need) single non-letter CSs
1061                % for now
1062                \peek_analysis_map_break:
1063            }{
1064                \str_if_eq:onTF \l_tmpa_str { begin } {
1065                    \peek_analysis_map_break:n {
1066                        \exp_after:wN \__stex_smsmode_checkbegin:n ##1
1067                    }
1068                } {
1069                    \str_if_eq:onTF \l_tmpa_str { end } {
1070                        \peek_analysis_map_break:n {
1071                            \exp_after:wN \__stex_smsmode_checkend:n ##1
1072                        }
1073                    } {
1074                        \tl_set:Nn \l_tmpa_tl { \use:c{\l_tmpa_str} }
1075                        \exp_args:NNNo \exp_args:NNo \tl_if_in:NnTF
1076                            \g_stex_smsmode_allowedmacros_tl
1077                            { \use:c{\l_tmpa_str} } {
1078                            \stex_debug:n{Executing~1:~\l_tmpa_str}
1079                            \peek_analysis_map_break:n {
1080                                \exp_after:wN \l_tmpa_tl ##1
1081                            }
1082                        } {
1083                            \exp_args:NNNo \exp_args:NNo \tl_if_in:NnTF
1084                                \g_stex_smsmode_allowedmacros_escape_tl
1085                                { \use:c{\l_tmpa_str} } {
1086                                \__stex_smsmode_unset_codes:
1087                                \stex_debug:n{Executing~2:~\l_tmpa_str}
1088                                % TODO \__stex_smsmode_rescan_cs:
1089 %                               \exp_after:wN \exp_after:wN \exp_after:wN
1090 %                               \token_if_eq_charcode:NNTF \exp_after:wN \c_backslash_str ##1 {
1091 %                                   \peek_analysis_map_break:n {
1092 %                                       \__stex_smsmode_unset_codes:
```

```
1093 %                        \__stex_smsmode_rescan_cs:
1094 %                      }
1095 %                    } {
1096                      \peek_analysis_map_break:n {
1097                        \exp_after:wN \l_tmpa_tl ##1
1098                      }
1099 %                    }
1100                  } {
1101                    \peek_analysis_map_break:n { ##1 }
1102                  }
1103                }
1104              }
1105            }
1106          }
1107        }
1108    }
1109 }
```

(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:

```
1110 \cs_new_protected:Nn \__stex_smsmode_rescan_cs: {
1111    \str_clear:N \l_tmpb_str
1112    \peek_analysis_map_inline:n {
1113      \token_if_eq_charcode:NNTF ##3 B {
1114        % token is a letter
1115        \exp_args:NNo \str_put_right:Nn \l_tmpb_str { ##1 }
1116      } {
1117        \peek_analysis_map_break:n {
1118          \exp_after:wN \use:c \exp_after:wN {
1119            \exp_after:wN \l_tmpa_str\exp_after:wN
1120          } \use:c { \l_tmpb_str \exp_after:wN } ##1
1121        }
1122      }
1123    }
1124 }
```

(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.

```
1125 \cs_new_protected:Nn \__stex_smsmode_checkbegin:n {
1126    \str_set:Nn \l_tmpa_str { #1 }
1127    \seq_if_in:NoT \g_stex_smsmode_allowedenvs_seq \l_tmpa_str {
1128      \__stex_smsmode_unset_codes:
1129      \begin{#1}
1130    }
1131 }
```

(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.

```
1132 \cs_new_protected:Nn \__stex_smsmode_checkend:n {
1133    \str_set:Nn \l_tmpa_str { #1 }
1134    \seq_if_in:NoT \g_stex_smsmode_allowedenvs_seq \l_tmpa_str {
```

51

```
1135        \end{#1}
1136    }
1137 }
```

(*End definition for* __stex_smsmode_checkend:n.)

4.5.3 Inheritance

```
1138 ⟨@@=stex_importmodule⟩
```

\stex_import_module_uri:nn

```
1139 \cs_new_protected:Nn \stex_import_module_uri:nn {
1140    \str_set:Nx \l__stex_importmodule_archive_str { #1 }
1141    \str_set:Nn \l__stex_importmodule_path_str { #2 }
1142    \str_if_empty:NT \l__stex_importmodule_archive_str {
1143       \prop_if_empty:NF \l_stex_current_repository_prop {
1144          \prop_get:NnN \l_stex_current_repository_prop { id } \l__stex_importmodule_archive_str
1145       }
1146    }
1147
1148    \exp_args:NNNo \seq_set_split:Nnn \l_tmpb_seq ? { \l__stex_importmodule_path_str }
1149    \seq_pop_right:NN \l_tmpb_seq \l__stex_importmodule_name_str
1150    \str_set:Nx \l__stex_importmodule_path_str { \seq_use:Nn \l_tmpb_seq ? }
1151
1152    \str_if_empty:NTF \l__stex_importmodule_archive_str {
1153       \stex_modules_current_namespace:
1154       \str_if_empty:NF \l__stex_importmodule_path_str {
1155          \str_set:Nx \l_stex_module_ns_str {
1156             \l_stex_module_ns_str / \l__stex_importmodule_path_str
1157          }
1158       }
1159    }{
1160       \stex_require_repository:n \l__stex_importmodule_archive_str
1161       \prop_get:cnN { c_stex_mathhub_\l__stex_importmodule_archive_str _manifest_prop } { ns }
1162          \l_stex_module_ns_str
1163       \str_if_empty:NF \l__stex_importmodule_path_str {
1164          \str_set:Nx \l_stex_module_ns_str {
1165             \l_stex_module_ns_str / \l__stex_importmodule_path_str
1166          }
1167       }
1168    }
1169 }
```

(*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.

```
1170 \str_new:N \l__stex_importmodule_name_str
1171 \str_new:N \l__stex_importmodule_archive_str
1172 \str_new:N \l__stex_importmodule_path_str
1173 \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⟩}`

```
1174 \cs_new_protected:Nn \stex_import_require_module:nnnn {
1175   \exp_args:Nx \stex_if_module_exists:nF { #1 ? #4 } {
1176     % \stex_debug:n{Arguments: #1, #2, #3, #4}
1177
1178     % archive
1179     \str_set:Nx \l_tmpa_str { #2 }
1180     \str_if_empty:NTF \l_tmpa_str {
1181       \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
1182     } {
1183       \stex_path_from_string:Nn \l_tmpb_seq { \l_tmpa_str }
1184       \seq_concat:NNN \l_tmpa_seq \c_stex_mathhub_seq \l_tmpb_seq
1185       \seq_put_right:Nn \l_tmpa_seq { source }
1186     }
1187
1188     % path
1189     \str_set:Nx \l_tmpb_str { #3 }
1190     \str_if_empty:NTF \l_tmpb_str {
1191       \str_set:Nx \l_tmpa_str { \stex_path_to_string:N \l_tmpa_seq / #4 }
1192
1193       \ltx@ifpackageloaded{babel} {
1194         \exp_args:NNx \prop_get:NnNF \c_stex_language_abbrevs_prop
1195           { \languagename } \l_tmpb_str {
1196             \msg_set:nnn{stex}{error/unknownlanguage}{
1197               Unknown~language~\languagename
1198             }
1199             \msg_error:nn{stex}{error/unknownlanguage}
1200           }
1201       } {
1202         \str_clear:N \l_tmpb_str
1203       }
1204
1205       \stex_debug:n{Checking~\l_tmpa_str.\l_tmpb_str.tex}
1206       \IfFileExists{ \l_tmpa_str.\l_tmpb_str.tex }{
1207         \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.\l_tmpb_str.tex }
1208       }{
1209         \stex_debug:n{Checking~\l_tmpa_str.tex}
1210         \IfFileExists{ \l_tmpa_str.tex }{
1211           \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.tex }
1212         }{
1213           % try english as default
1214           \stex_debug:n{Checking~\l_tmpa_str.en.tex}
1215           \IfFileExists{ \l_tmpa_str.en.tex }{
1216             \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.en.tex }
1217           }{
1218             \msg_set:nnn{stex}{error/modulemissing}{
1219               No~file~for~module~#1?#4~found
1220             }
1221             \msg_error:nn{stex}{error/modulemissing}
1222           }
1223         }
1224       }
1225
1226     } {
```

```
1227        \seq_set_split:NnV \l_tmpb_seq / \l_tmpb_str
1228        \seq_concat:NNN \l_tmpa_seq \l_tmpa_seq \l_tmpb_seq
1229
1230        \ltx@ifpackageloaded{babel} {
1231          \exp_args:NNx \prop_get:NnNF \c_stex_language_abbrevs_prop
1232              { \languagename } \l_tmpb_str {
1233                  \msg_set:nnn{stex}{error/unknownlanguage}{
1234                    Unknown~language~\languagename
1235                  }
1236                  \msg_error:nn{stex}{error/unknownlanguage}
1237              }
1238        } {
1239          \str_clear:N \l_tmpb_str
1240        }
1241
1242        \stex_path_to_string:NN \l_tmpa_seq \l_tmpa_str
1243
1244        \stex_debug:n{Checking~\l_tmpa_str/#4.\l_tmpb_str.tex}
1245        \IfFileExists{ \l_tmpa_str/#4.\l_tmpb_str.tex }{
1246          \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.\l_tmpb_str.tex }
1247        }{
1248          \stex_debug:n{Checking~\l_tmpa_str/#4.tex}
1249          \IfFileExists{ \l_tmpa_str/#4.tex }{
1250            \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.tex }
1251          }{
1252            % try english as default
1253            \stex_debug:n{Checking~\l_tmpa_str/#4.en.tex}
1254            \IfFileExists{ \l_tmpa_str/#4.en.tex }{
1255              \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.en.tex }
1256            }{
1257              \stex_debug:n{Checking~\l_tmpa_str.\l_tmpb_str.tex}
1258              \IfFileExists{ \l_tmpa_str.\l_tmpb_str.tex }{
1259                \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.\l_tmpb_str.tex }
1260              }{
1261                \stex_debug:n{Checking~\l_tmpa_str.tex}
1262                \IfFileExists{ \l_tmpa_str.tex }{
1263                  \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.tex }
1264                }{
1265                  % try english as default
1266                  \stex_debug:n{Checking~\l_tmpa_str.en.tex}
1267                  \IfFileExists{ \l_tmpa_str.en.tex }{
1268                    \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.en.tex }
1269                  }{
1270                    \msg_set:nnn{stex}{error/modulemissing}{
1271                      No~file~for~module~#1?#4~found
1272                    }
1273                    \msg_error:nn{stex}{error/modulemissing}
1274                  }
1275                }
1276              }
1277            }
1278          }
1279        }
1280      }
```

```
1281
1282      \seq_set_eq:NN \l_tmpa_seq \g_stex_modules_in_file_seq
1283      \seq_clear:N \g_stex_modules_in_file_seq
1284 %     \exp_args:Nnx \use:nn {
1285        \exp_args:No \stex_in_smsmode:nn { \g__stex_importmodule_file_str } {
1286          \seq_clear:N \l_stex_all_modules_seq
1287          \prop_clear:N \l_stex_current_module_prop
1288          \str_set:Nx \l_tmpb_str { #2 }
1289          \str_if_empty:NF \l_tmpb_str {
1290            \stex_set_current_repository:n { #2 }
1291          }
1292          \stex_debug:n{Loading~\g__stex_importmodule_file_str}
1293          \input { \g__stex_importmodule_file_str }
1294        }
1295 %     }{
1296
1297 %     }
1298      \prop_gput:Noo \g_stex_module_files_prop
1299      \g__stex_importmodule_file_str \g_stex_modules_in_file_seq
1300      \seq_set_eq:NN \g_stex_modules_in_file_seq \l_tmpa_seq
1301
1302      \stex_if_module_exists:nF { #1 ? #4 } {
1303        \msg_set:nnn{stex}{error/modulemissing}{
1304          Module~#1?#4~not~found~in~file~\g__stex_importmodule_file_str
1305        }
1306        \msg_error:nn{stex}{error/modulemissing}
1307      }
1308    }
1309    \stex_activate_module:n { #1 ? #4 }
1310 }
```

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

```
1311 \cs_new_protected:Nn \stex_activate_module:n {
1312    \stex_debug:n{Activating~module~#1}
1313    \exp_args:NNx \seq_if_in:NnF \l_stex_all_modules_seq { #1 } {
1314      \seq_put_right:Nx \l_stex_all_modules_seq { #1 }
1315      \prop_item:cn { c_stex_module_#1_prop } { content }
1316    }
1317 }
```

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

```
1318 \NewDocumentCommand \importmodule { O{} m } {
1319    \stex_import_module_uri:nn { #1 } { #2 }
1320    \stex_debug:n{Importing~module:~
1321      \l_stex_module_ns_str ? \l__stex_importmodule_name_str
1322    }
1323    \stex_if_smsmode:F {
1324      \stex_import_require_module:nnnn
1325      { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
1326      { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
1327      \stex_annotate_invisible:nnn
```

```
1328            {import} {\l_stex_module_ns_str ? \l__stex_importmodule_name_str} {}
1329        }
1330        \exp_args:Nx \stex_add_to_current_module:n {
1331            \stex_import_require_module:nnnn
1332            { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
1333            { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
1334        }
1335        \exp_args:Nx \stex_add_import_to_current_module:n {
1336            \l_stex_module_ns_str ? \l__stex_importmodule_name_str
1337        }
1338        \stex_smsmode_set_codes:
1339    }
```

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

The red margin label:

`\usemodule`

```
1340    \NewDocumentCommand \usemodule { O{} m } {
1341        \stex_if_smsmode:F {
1342            \stex_import_module_uri:nn { #1 } { #2 }
1343            \stex_import_require_module:nnnn
1344            { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
1345            { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
1346            \stex_annotate_invisible:nnn
1347                {usemodule} {\l_stex_module_ns_str ? \l__stex_importmodule_name_str} {}
1348        }
1349        \stex_smsmode_set_codes:
1350    }
```

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

`\g_stex_modules_in_file_seq`
`\g_stex_module_files_prop`

```
1351    \seq_new:N \g_stex_modules_in_file_seq
1352    \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

```
1353    ⟨@@=stex_symdecl⟩
```

`\l_stex_all_symbols_seq`   Stores all available symbols

```
1354    \seq_new:N \l_stex_all_symbols_seq
```

(*End definition for* `\l_stex_all_symbols_seq`*. This variable is documented on page* *21.*)

`\STEXsymbol`

```
1355    \NewDocumentCommand \STEXsymbol { m } {
1356        \stex_get_symbol:n { #1 }
1357        \exp_args:No
1358        \stex_invoke_symbol:n { \l_stex_get_symbol_uri_str }
1359    }
```

56

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

symdecl arguments:

```
1360 \keys_define:nn { stex / symdecl } {
1361   name       .tl_set_x:N  = \l_stex_symdecl_name_str ,
1362   local      .bool_set:N  = \l_stex_symdecl_local_bool ,
1363   args       .tl_set_x:N  = \l_stex_symdecl_args_str ,
1364   type       .tl_set:N    = \l_stex_symdecl_type_tl ,
1365   align      .tl_set:N    = \l_stex_symdecl_align_str , % TODO(?)
1366   gfc        .tl_set:N    = \l_stex_symdecl_gfc_str , % TODO(?)
1367   specializes .tl_set:N   = \l_stex_symdecl_specializes_str , % TODO(?)
1368   def        .tl_set:N    = \l_stex_symdecl_definiens_tl
1369 }
1370
1371 \bool_new:N \l_stex_symdecl_make_macro_bool
1372
1373 \cs_new_protected:Nn \__stex_symdecl_args:n {
1374   \str_clear:N \l_stex_symdecl_name_str
1375   \str_clear:N \l_stex_symdecl_args_str
1376   \bool_set_false:N \l_stex_symdecl_local_bool
1377   \tl_clear:N \l_stex_symdecl_type_tl
1378   \tl_clear:N \l_stex_symdecl_definiens_tl
1379
1380   \keys_set:nn { stex /symdecl } { #1 }
1381
1382   \exp_args:NNo \str_set:Nn \l_stex_symdecl_name_str
1383     \l_stex_symdecl_name_str
1384   \exp_args:NNo \str_set:Nn \l_stex_symdecl_args_str
1385     \l_stex_symdecl_args_str
1386 }
```

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

```
1387
1388 \NewDocumentCommand \symdecl { s O{} m } {
1389   \__stex_symdecl_args:n { #2 }
1390   \IfBooleanTF #1 {
1391     \bool_set_false:N \l_stex_symdecl_make_macro_bool
1392   } {
1393     \bool_set_true:N \l_stex_symdecl_make_macro_bool
1394   }
1395   \stex_symdecl_do:n { #3 }
1396   \stex_smsmode_set_codes:
1397 }
```

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

`\stex_symdecl_do:n`

```
1398 \cs_new_protected:Nn \stex_symdecl_do:n {
1399   \stex_if_in_module:F {
1400     % TODO throw error? some default namespace?
1401   }
1402
1403   \str_if_empty:NT \l_stex_symdecl_name_str {
```

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

58

```
1458        }
1459     }
1460     \bool_if:NTF \l_tmpa_bool {
1461       % possibly numeric
1462       \str_if_empty:NTF \l_stex_symdecl_args_str {
1463         \prop_put:Nnn \l_tmpa_prop { args } {}
1464         \prop_put:Nnn \l_tmpa_prop { arity } { 0 }
1465       }{
1466         \int_set:Nn \l_tmpa_int { \l_stex_symdecl_args_str }
1467         \prop_put:Nnx \l_tmpa_prop { arity } { \int_use:N \l_tmpa_int }
1468         \str_clear:N \l_tmpa_str
1469         \int_step_inline:nn \l_tmpa_int {
1470           \str_put_right:Nn \l_tmpa_str i
1471         }
1472         \prop_put:Nnx \l_tmpa_prop { args } { \l_tmpa_str }
1473       }
1474     } {
1475       \prop_put:Nnx \l_tmpa_prop { args } { \l_stex_symdecl_args_str }
1476       \prop_put:Nnx \l_tmpa_prop { arity }
1477         { \str_count:N \l_stex_symdecl_args_str }
1478     }
1479     \prop_put:Nnx \l_tmpa_prop { assocs } { \int_use:N \l_tmpb_int }


1482     % semantic macro

1484     \bool_if:NT \l_stex_symdecl_make_macro_bool {
1485       \tl_set:cx { #1 } { \stex_invoke_symbol:n {
1486         \prop_item:Nn \l_tmpa_prop { module } ?
1487           \prop_item:Nn \l_tmpa_prop { name }
1488       } }

1490       \bool_if:NF \l_stex_symdecl_local_bool {
1491         \exp_args:Nx \stex_add_to_current_module:n {
1492           \tl_set:cx { #1 } { \stex_invoke_symbol:n {
1493             \prop_item:Nn \l_tmpa_prop { module } ?
1494               \prop_item:Nn \l_tmpa_prop { name }
1495           } }
1496         }
1497       }
1498     }

1500     % add to all symbols

1502     \bool_if:NF \l_stex_symdecl_local_bool {
1503       \exp_args:Nx \stex_add_to_current_module:n {
1504         \seq_put_right:Nn \exp_not:N \l_stex_all_symbols_seq {
1505           \prop_item:Nn \l_tmpa_prop { module } ?
1506           \prop_item:Nn \l_tmpa_prop { name }
1507         }
1508       }
1509     }

1511     \stex_debug:n{New~symbol:~
```

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

```
1566        } {
1567          \stex_annotate_invisible:nnn{type}{}{$\l_stex_symdecl_type_tl$}
1568          \stex_annotate_invisible:nnn{args}{}{
1569            \prop_item:Nn \l_tmpa_prop { args }
1570          }
1571          \stex_annotate_invisible:nnn{macroname}{}{#1}
1572          \tl_if_empty:NF \l_stex_symdecl_definiens_tl {
1573            \stex_annotate_invisible:nnn{definiens}{}
1574              {$\l_stex_symdecl_definiens_tl$}
1575          }
1576        }
1577      }
1578 }
```

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

```
1579 \str_new:N \l_stex_get_symbol_uri_str
1580
1581 \cs_new_protected:Nn \stex_get_symbol:n {
1582   \tl_if_head_eq_catcode:nNTF { #1 } \relax {
1583     \__stex_symdecl_get_symbol_from_cs:n { #1 }
1584   }{
1585     % argument is a string
1586     % is it a command name?
1587     \cs_if_exist:cTF { #1 }{
1588       \cs_set_eq:Nc \l_tmpa_tl { #1 }
1589       \str_set:Nx \l_tmpa_str { \cs_argument_spec:N \l_tmpa_tl }
1590       \str_if_empty:NTF \l_tmpa_str {
1591         \exp_args:Nx \cs_if_eq:NNTF {
1592           \tl_head:N \l_tmpa_tl
1593         } \stex_invoke_symbol:n {
1594           \exp_args:No \__stex_symdecl_get_symbol_from_cs:n { \use:c { #1 } }
1595         }{
1596           \__stex_symdecl_get_symbol_from_string:n { #1 }
1597         }
1598       } {
1599         \__stex_symdecl_get_symbol_from_string:n { #1 }
1600       }
1601     }{
1602       % argument is not a command name
1603       \__stex_symdecl_get_symbol_from_string:n { #1 }
1604       % \l_stex_all_symbols_seq
1605     }
1606   }
1607 }
1608
1609 \cs_new_protected:Nn \__stex_symdecl_get_symbol_from_string:n {
1610   \str_set:Nn \l_tmpa_str { #1 }
1611   \bool_set_false:N \l_tmpa_bool
1612   \stex_if_in_module:T {
1613     \prop_get:NnN \l_stex_current_module_prop
1614     { constants } \l_tmpa_seq
1615     \exp_args:NNo \seq_if_in:NnT \l_tmpa_seq { \l_tmpa_str } {
```

```
1616        \bool_set_true:N \l_tmpa_bool
1617        \str_set:Nx \l_stex_get_symbol_uri_str {
1618          \prop_item:Nn \l_stex_current_module_prop { ns } ?
1619          \prop_item:Nn \l_stex_current_module_prop { name } ? #1
1620        }
1621      }
1622    }
1623    \bool_if:NF \l_tmpa_bool {
1624      \tl_set:Nn \l_tmpa_tl {
1625        \msg_set:nnn{stex}{error/unknownsymbol}{
1626          No~symbol~#1~found!
1627        }
1628        \msg_error:nn{stex}{error/unknownsymbol}
1629      }
1630      \str_set:Nn \l_tmpa_str { #1 }
1631      \int_set:Nn \l_tmpa_int { \str_count:N \l_tmpa_str }
1632      \seq_map_inline:Nn \l_stex_all_symbols_seq {
1633        \str_set:Nn \l_tmpb_str { ##1 }
1634        \str_if_eq:eeT { \l_tmpa_str } {
1635          \str_range:Nnn \l_tmpb_str { -\l_tmpa_int } { -1 }
1636        } {
1637          \seq_map_break:n {
1638            \tl_set:Nn \l_tmpa_tl {
1639              \str_set:Nn \l_stex_get_symbol_uri_str {
1640                ##1
1641              }
1642            }
1643          }
1644        }
1645      }
1646      \l_tmpa_tl
1647    }
1648 }
1649
1650 \cs_new_protected:Nn \__stex_symdecl_get_symbol_from_cs:n {
1651    \exp_args:NNx \tl_set:Nn \l_tmpa_tl
1652      { \tl_tail:N \l_tmpa_tl }
1653    \tl_if_single:NTF \l_tmpa_tl {
1654      \exp_args:No \tl_if_head_is_group:nTF \l_tmpa_tl {
1655        \exp_after:wN \str_set:Nn \exp_after:wN
1656          \l_stex_get_symbol_uri_str \l_tmpa_tl
1657      }{
1658        % TODO
1659        % tail is not a single group
1660      }
1661    }{
1662      % TODO
1663      % tail is not a single group
1664    }
1665 }
```

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

4.7 Notations

⟨@@=stex_notation⟩

`notation` arguments:

```
1667 \keys_define:nn { stex / notation } {
1668   lang    .tl_set_x:N = \l__stex_notation_lang_str ,
1669   variant .tl_set_x:N = \l__stex_notation_variant_str ,
1670   prec    .tl_set_x:N = \l__stex_notation_prec_str ,
1671   op      .tl_set:N   = \l__stex_notation_op_tl ,
1672   unknown .code:n     = \str_set:Nx
1673       \l__stex_notation_variant_str \l_keys_key_str
1674 }
1675
1676 \cs_new_protected:Nn \__stex_notation_args:n {
1677   \str_clear:N \l__stex_notation_lang_str
1678   \str_clear:N \l__stex_notation_variant_str
1679   \str_clear:N \l__stex_notation_prec_str
1680   \tl_clear:N \l__stex_notation_op_tl
1681
1682   \keys_set:nn { stex / notation } { #1 }
1683
1684   \str_set:Nx \l__stex_notation_lang_str \l__stex_notation_lang_str
1685   \str_set:Nx \l__stex_notation_variant_str \l__stex_notation_variant_str
1686   \str_set:Nx \l__stex_notation_prec_str \l__stex_notation_prec_str
1687 }
```

\notation

```
1688 \NewDocumentCommand \notation { O{} m } {
1689   \__stex_notation_args:n { #1 }
1690   \tl_clear:N \l_stex_symdecl_definiens_tl
1691   \stex_get_symbol:n { #2 }
1692   \stex_notation_do:nn { \l_stex_get_symbol_uri_str }
1693 }
```

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

\stex_notation_do:nn

```
1694 \cs_new_protected:Nn \stex_notation_do:nn {
1695   \prop_set_eq:Nc \l_tmpa_prop {
1696     g_stex_symdecl_ #1 _prop
1697   }
1698
1699   \prop_clear:N \l_tmpb_prop
1700   \prop_put:Nno \l_tmpb_prop { symbol } { #1 }
1701   \prop_put:Nno \l_tmpb_prop { language } \l__stex_notation_lang_str
1702   \prop_put:Nno \l_tmpb_prop { variant } \l__stex_notation_variant_str
1703
1704   % precedences
1705   \seq_clear:N \l_tmpb_seq
1706   \exp_args:NNno
1707   \str_if_empty:NTF \l__stex_notation_prec_str {
1708     \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1709     \int_compare:nNnTF \l_tmpa_str = 0 {
1710       \exp_args:NNnx
```

63

```
1711        \prop_put:Nno \l_tmpb_prop { opprec }
1712          { \neginfprec }
1713      }{
1714        \prop_put:Nnn \l_tmpb_prop { opprec } { 0 }
1715      }
1716    } {
1717      \str_if_eq:onTF \l__stex_notation_prec_str {nobrackets}{
1718        \exp_args:NNnx
1719        \prop_put:Nno \l_tmpb_prop { opprec }
1720          { \neginfprec }
1721        \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1722        \int_step_inline:nn { \l_tmpa_str } {
1723          \exp_args:NNx
1724          \seq_put_right:Nn \l_tmpb_seq { \infprec }
1725        }
1726      }{
1727        \seq_set_split:NnV \l_tmpa_seq ; \l__stex_notation_prec_str
1728        \seq_pop_left:NNTF \l_tmpa_seq \l_tmpa_str {
1729          \prop_put:Nno \l_tmpb_prop { opprec } \l_tmpa_str
1730          \seq_pop_left:NNT \l_tmpa_seq \l_tmpa_str {
1731            \exp_args:NNNo \exp_args:NNno \seq_set_split:Nnn
1732              \l_tmpa_seq {\tl_to_str:n{x} } { \l_tmpa_str }
1733            \seq_map_inline:Nn \l_tmpa_seq {
1734              \seq_put_right:Nn \l_tmpb_seq { ##1 }
1735            }
1736          }
1737          \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1738        }{
1739          \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
1740          \int_compare:nNnTF \l_tmpa_str = 0 {
1741            \exp_args:NNnx
1742            \prop_put:Nno \l_tmpb_prop { opprec }
1743              { \infprec }
1744          }{
1745            \prop_put:Nnn \l_tmpb_prop { opprec } { 0 }
1746          }
1747        }
1748      }
1749    }
1750
1751    \seq_set_eq:NN \l_tmpa_seq \l_tmpb_seq
1752    \int_step_inline:nn { \l_tmpa_str } {
1753      \seq_pop_left:NNF \l_tmpa_seq \l_tmpb_str {
1754        \exp_args:NNx
1755        \seq_put_right:Nn \l_tmpb_seq {
1756          \prop_item:Nn \l_tmpb_prop { opprec }
1757        }
1758      }
1759    }
1760
1761    \prop_put:Nno \l_tmpb_prop { argprecs } \l_tmpb_seq
1762    \tl_clear:N \l_tmpa_tl
1763
1764    \int_compare:nNnTF \l_tmpa_str = 0 {
```

```
1765      \exp_args:NNe
1766      \cs_set:Npn \l__stex_notation_macrocode_cs {
1767        \_stex_term_math_oms:nnnn { #1 }
1768          { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1769          { \prop_item:Nn \l_tmpb_prop { opprec } }
1770          { \exp_not:n { #2 } }
1771      }
1772      \__stex_notation_final:
1773    }{
1774      \prop_get:NnN \l_tmpa_prop { args } \l_tmpb_str
1775      \str_if_in:NnTF \l_tmpb_str b {
1776        \exp_args:Nne \use:nn
1777          {
1778          \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
1779          \cs_set:Npn \l_tmpa_str } { { {
1780            \_stex_term_math_omb:nnnn { #1 }
1781              { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1782              { \prop_item:Nn \l_tmpb_prop { opprec } }
1783              { \exp_not:n { #2 } }
1784        }}
1785      }{
1786        \str_if_in:NnTF \l_tmpb_str B {
1787          \exp_args:Nne \use:nn
1788            {
1789            \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
1790            \cs_set:Npn \l_tmpa_str } { { {
1791              \_stex_term_math_omb:nnnn { #1 }
1792                { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1793                { \prop_item:Nn \l_tmpb_prop { opprec } }
1794                { \exp_not:n { #2 } }
1795          } }
1796        }{
1797          \exp_args:Nne \use:nn
1798            {
1799            \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
1800            \cs_set:Npn \l_tmpa_str } { { {
1801              \_stex_term_math_oma:nnnn { #1 }
1802                { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
1803                { \prop_item:Nn \l_tmpb_prop { opprec } }
1804                { \exp_not:n { #2 } }
1805          } }
1806        }
1807      }
1808
1809      \int_zero:N \l_tmpa_int
1810      \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
1811      \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
1812      \__stex_notation_arguments:
1813    }
1814  }
```

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

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

```
1815 \cs_new_protected:Nn \__stex_notation_arguments: {
1816   \int_incr:N \l_tmpa_int
1817   \str_if_empty:NTF \l_tmpa_str {
1818     \__stex_notation_final:
1819   }{
1820     \str_set:Nx \l_tmpb_str { \str_head:N \l_tmpa_str }
1821     \str_set:Nx \l_tmpa_str { \str_tail:N \l_tmpa_str }
1822     \str_if_eq:VnTF \l_tmpb_str a {
1823       \__stex_notation_argument_assoc:n
1824     }{
1825       \str_if_eq:VnTF \l_tmpb_str B {
1826         \__stex_notation_argument_assoc:n
1827       }{
1828         \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
1829         \tl_put_right:Nx \l_tmpa_tl {
1830           { \_stex_term_math_arg:nnn
1831             { \int_use:N \l_tmpa_int }
1832             { \l_tmpb_str }
1833             { ####\int_use:N \l_tmpa_int }
1834           }
1835         }
1836         \__stex_notation_arguments:
1837       }
1838     }
1839   }
1840 }
```

(End definition for __stex_notation_arguments:.)

__stex_notation_argument_assoc:n

```
1841 \cs_new_protected:Nn \__stex_notation_argument_assoc:n {
1842   \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
1843   \cs_set:Npn \l_tmpa_cs ##1 ##2 { #1 }
1844   \tl_put_right:Nx \l_tmpa_tl {
1845     { \_stex_term_math_assoc_arg:nnnn
1846       { \int_use:N \l_tmpa_int }
1847       { \l_tmpb_str }
1848       \exp_args:No \exp_not:n
1849       {\exp_after:wN { \l_tmpa_cs {####1} {####2} } }
1850       { ####\int_use:N \l_tmpa_int }
1851     }
1852   }
1853   \__stex_notation_arguments:
1854 }
```

(End definition for __stex_notation_argument_assoc:n.)

__stex_notation_final: Called after processing all notation arguments

```
1855 \cs_new_protected:Nn \__stex_notation_final: {
1856   \prop_get:NnN \l_tmpa_prop { arity } \l_tmpb_str
1857   \prop_get:NnN \l_tmpb_prop { symbol } \l_tmpa_str
1858   \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
1859   \exp_args:Nne \use:nn
1860   {
```

```
1861    \cs_generate_from_arg_count:cNnn {
1862        stex_notation_ \l_tmpa_str \c_hash_str
1863        \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1864        _cs
1865      }
1866      \cs_gset:Npn \l_tmpb_str } { {
1867        \exp_after:wN \exp_after:wN \exp_after:wN
1868        \exp_not:n \exp_after:wN \exp_after:wN \exp_after:wN
1869        { \exp_after:wN \l__stex_notation_macrocode_cs \l_tmpa_tl }
1870    } }
1871
1872    \tl_if_empty:NF \l__stex_notation_op_tl {
1873      \cs_gset:cpx {
1874        stex_op_notation_ \l_tmpa_str \c_hash_str
1875        \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1876        _cs
1877      } {
1878        \_stex_term_oms:nnn {
1879          \l_tmpa_str \c_hash_str \l__stex_notation_variant_str \c_hash_str
1880          \l__stex_notation_lang_str
1881        }{
1882          \l_tmpa_str
1883        }{ \comp{ \exp_args:No \exp_not:n { \l__stex_notation_op_tl } } } }
1884      }
1885    }
1886
1887
1888
1889    \stex_debug:n{
1890      Notation~\l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1891      ~for~\prop_item:Nn \l_tmpb_prop { symbol }^^J
1892      Operator~precedence:~
1893        \prop_item:Nn \l_tmpb_prop { opprec }^^J
1894      Argument~precedences:~
1895        \seq_use:Nn \l_tmpa_seq {,~}^^J
1896      Notation: \cs_meaning:c {
1897        stex_notation_ \l_tmpa_str \c_hash_str
1898        \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1899        _cs
1900      }
1901    }
1902
1903    \prop_gset_eq:cN {
1904      g_stex_notation_ \l_tmpa_str \c_hash_str \l__stex_notation_variant_str
1905        \c_hash_str \l__stex_notation_lang_str _prop
1906    } \l_tmpb_prop
1907
1908    \exp_args:Nx
1909    \stex_add_to_current_module:n {
1910      \prop_get:cnN {
1911        g_stex_symdecl_
1912          \prop_item:Nn \l_tmpb_prop { symbol }
1913        _prop
1914      } { notations } \exp_not:N \l_tmpa_seq
```

```
1915    \seq_put_right:Nn \exp_not:N \l_tmpa_seq {
1916      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1917    }
1918    \prop_put:cno {
1919      g_stex_symdecl_
1920        \prop_item:Nn \l_tmpb_prop { symbol }
1921      _prop
1922    } { notations } \exp_not:N \l_tmpa_seq
1923  }
1924
1925  \stex_if_smsmode:TF {
1926    \stex_smsmode_set_codes:
1927    \exp_args:Nx \stex_addtosms:n {
1928      \prop_gset_from_keyval:cn {
1929        g_stex_notation_ \l_tmpa_str \c_hash_str \l__stex_notation_variant_str
1930          \c_hash_str \l__stex_notation_lang_str _prop
1931      } {
1932        symbol    = \prop_item:Nn \l_tmpb_prop { symbol }     ,
1933        language  = \prop_item:Nn \l_tmpb_prop { language }   ,
1934        variant   = \prop_item:Nn \l_tmpb_prop { variant }    ,
1935        opprec    = \prop_item:Nn \l_tmpb_prop { opprec }     ,
1936        argprecs  = \prop_item:Nn \l_tmpb_prop { argprecs }   ,
1937      }
1938    }
1939  }{
1940    \prop_get:NnN \l_tmpa_prop { notations } \l_tmpa_seq
1941    \seq_put_right:Nx \l_tmpa_seq {
1942      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
1943    }
1944    \prop_put:Nno \l_tmpa_prop { notations } \l_tmpa_seq
1945    \prop_set_eq:cN {
1946      g_stex_symdecl_ \l_tmpa_str _prop
1947    } \l_tmpa_prop
1948
1949    % HTML annotations
1950    \stex_annotate_invisible:nnn { notation }
1951      { \prop_item:Nn \l_tmpb_prop { symbol } } {
1952        \stex_annotate_invisible:nnn { notationfragment }
1953          { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }{}
1954        \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
1955        \stex_annotate_invisible:nnn { precedence }
1956          { \prop_item:Nn \l_tmpb_prop { opprec };
1957            \seq_use:Nn \l_tmpa_seq { x }
1958          }{}
1959
1960        \int_zero:N \l_tmpa_int
1961        \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
1962        \tl_clear:N \l_tmpa_tl
1963        \int_step_inline:nn { \prop_item:Nn \l_tmpa_prop { arity } }{
1964          \int_incr:N \l_tmpa_int
1965          \str_set:Nx \l_tmpb_str { \str_head:N \l_tmpa_str }
1966          \str_set:Nx \l_tmpa_str { \str_tail:N \l_tmpa_str }
1967          \str_if_eq:VnTF \l_tmpb_str a {
1968            \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
```

```
1969              \c_hash_str \c_hash_str \int_use:N \l_tmpa_int a ,
1970              \c_hash_str \c_hash_str \int_use:N \l_tmpa_int b
1971            } }
1972          }{
1973            \str_if_eq:VnTF \l_tmpb_str B {
1974              \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
1975                \c_hash_str \c_hash_str \int_use:N \l_tmpa_int a ,
1976                \c_hash_str \c_hash_str \int_use:N \l_tmpa_int b
1977              } }
1978            }{
1979              \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
1980                \c_hash_str \c_hash_str \int_use:N \l_tmpa_int
1981              } }
1982            }
1983          }
1984        }
1985        \stex_annotate_invisible:nnn { notationcomp }{}{
1986          $ \exp_args:Nno \use:nn { \use:c {
1987            stex_notation_ \prop_item:Nn \l_tmpb_prop { symbol }
1988            \c_hash_str \l__stex_notation_variant_str
1989            \c_hash_str \l__stex_notation_lang_str _cs
1990          } } { \l_tmpa_tl } $
1991        }
1992      }
1993    }
1994 }
```

(End definition for `\__stex_notation_final:`*.)*

```
1995 \keys_define:nn { stex / symdef } {
1996   name   .tl_set_x:N  = \l_stex_symdecl_name_str ,
1997   local .bool_set:N  = \l_stex_symdecl_local_bool ,
1998   args   .tl_set_x:N  = \l_stex_symdecl_args_str ,
1999   type  .tl_set:N    = \l_stex_symdecl_type_tl ,
2000   def    .tl_set:N    = \l_stex_symdecl_definiens_tl ,
2001   op     .tl_set:N    = \l__stex_notation_op_tl ,
2002   lang    .tl_set_x:N = \l__stex_notation_lang_str ,
2003   variant .tl_set_x:N = \l__stex_notation_variant_str ,
2004   prec    .tl_set_x:N = \l__stex_notation_prec_str ,
2005   unknown .code:n     = \str_set:Nx
2006       \l__stex_notation_variant_str \l_keys_key_str
2007 }
2008
2009 \cs_new_protected:Nn \__stex_notation_symdef_args:n {
2010   \str_clear:N \l_stex_symdecl_name_str
2011   \str_clear:N \l_stex_symdecl_args_str
2012   \bool_set_false:N \l_stex_symdecl_local_bool
2013   \tl_clear:N \l_stex_symdecl_type_tl
2014   \tl_clear:N \l_stex_symdecl_definiens_tl
2015   \str_clear:N \l__stex_notation_lang_str
2016   \str_clear:N \l__stex_notation_variant_str
2017   \str_clear:N \l__stex_notation_prec_str
2018   \tl_clear:N \l__stex_notation_op_tl
```

```
2019
2020    \keys_set:nn { stex /symdef } { #1 }

2022    \exp_args:NNo \str_set:Nn \l_stex_symdecl_name_str
2023      \l_stex_symdecl_name_str
2024    \exp_args:NNo \str_set:Nn \l_stex_symdecl_args_str
2025      \l_stex_symdecl_args_str
2026    \exp_args:NNo \str_set:Nn \l__stex_notation_lang_str
2027      \l__stex_notation_lang_str
2028    \exp_args:NNo \str_set:Nn \l__stex_notation_variant_str
2029      \l__stex_notation_variant_str
2030    \exp_args:NNo \str_set:Nn \l__stex_notation_prec_str
2031      \l__stex_notation_prec_str
2032 }

2034 \NewDocumentCommand \symdef { O{} m } {
2035   \__stex_notation_symdef_args:n { #1 }
2036   \bool_set_true:N \l_stex_symdecl_make_macro_bool
2037   \stex_symdecl_do:n { #2 }
2038   \exp_args:Nx \stex_notation_do:nn {
2039     \prop_item:Nn \l_tmpa_prop { module } ?
2040     \prop_item:Nn \l_tmpa_prop { name }
2041   }
2042 }
```

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

\stex_invoke_symbol:n   Invokes a semantic macro

```
2043 %\cs_new_protected:Nn \stex_invoke_symbol:n {
2044 %   \peek_charcode_remove:NTF ! {
2045 %     \stex_term_custom:nn { #1 } { }
2046 %   } {
2047 %     \if_mode_math:
2048 %       \exp_after:wN \__stex_notation_invoke_math:n
2049 %     \else:
2050 %       \exp_after:wN \__stex_notation_invoke_text:n
2051 %     \fi: { #1 }
2052 %   }
2053 %}

2055 \cs_new_protected:Nn \stex_invoke_symbol:n {
2056   \if_mode_math:
2057     \exp_after:wN \__stex_notation_invoke_math:n
2058   \else:
2059     \exp_after:wN \__stex_notation_invoke_text:n
2060   \fi: { #1 }
2061 }
```

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

__stex_notation_invoke_math:n

```
2062 \cs_new_protected:Nn \__stex_notation_invoke_math:n {
2063   \peek_charcode_remove:NTF ! {
2064     \peek_charcode:NTF [ {
```

```
2065        \__stex_notation_invoke_op:nw { #1 }
2066      }{
2067        \__stex_notation_invoke_op:nw { #1 } []
2068      }
2069    }{
2070      \peek_charcode_remove:NTF * {
2071        \__stex_notation_invoke_text:n { #1 }
2072      }{
2073        \peek_charcode:NTF [ {
2074          \__stex_notation_invoke_math:nw { #1 }
2075        }{
2076          \__stex_notation_invoke_math:nw { #1 } []
2077        }
2078      }
2079    }
2080 }
```

(End definition for __stex_notation_invoke_math:n.)

__stex_notation_invoke_op:nw

```
2081 \cs_new_protected:Npn \__stex_notation_invoke_op:nw  #1 [#2] {
2082    \__stex_notation_args:n { #2 }
2083    \cs_if_exist:cTF {
2084      stex_op_notation_ #1 \c_hash_str
2085      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str _cs
2086    }{
2087      \csname stex_op_notation_ #1 \c_hash_str
2088        \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str _cs
2089      \endcsname
2090    }{
2091      % TODO throw error
2092    }
2093 }
```

(End definition for __stex_notation_invoke_op:nw.)

__stex_notation_invoke_math:nw

```
2094 \cs_new_protected:Npn \__stex_notation_invoke_math:nw  #1 [#2] {
2095    \__stex_notation_args:n { #2 }
2096    \prop_set_eq:Nc \l_tmpa_prop {
2097      g_stex_symdecl_ #1 _prop
2098    }
2099    \prop_get:NnN \l_tmpa_prop { notations } \l_tmpa_seq
2100    \seq_if_empty:NTF \l_tmpa_seq {
2101      \msg_set:nnn{stex}{error/nonotations}{
2102        Symbol~#1~used,~but~has~no~notations!
2103      }
2104      \msg_error:nn{stex}{error/nonotations}
2105    } {
2106      \seq_if_in:NxTF \l_tmpa_seq
2107        { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }{
2108        \use:c{
2109          stex_notation_ #1 \c_hash_str
2110          \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
2111          _cs
```

71

```
2112           }
2113        }{
2114          \str_if_empty:NTF \l__stex_notation_variant_str {
2115            \str_if_empty:NTF \l__stex_notation_lang_str {
2116              \seq_get_left:NN \l_tmpa_seq \l_tmpa_str
2117              \use:c{
2118                stex_notation_ #1 \c_hash_str \l_tmpa_str
2119                _cs
2120              }
2121            }{
2122              \msg_set:nnn{stex}{error/wrongnotation}{
2123                Symbol~#1~has~no~notation~
2124                \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
2125              }
2126              \msg_error:nn{stex}{error/wrongnotation}
2127            }
2128          }{
2129            \msg_set:nnn{stex}{error/wrongnotation}{
2130              Symbol~#1~has~no~notation~
2131              \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
2132            }
2133            \msg_error:nn{stex}{error/wrongnotation}
2134          }
2135        }
2136      }
2137  }
```

(*End definition for* `\__stex_notation_invoke_math:nw`.)

`\__stex_notation_invoke_text:n`

```
2138  \cs_new_protected:Nn \__stex_notation_invoke_text:n {
2139    \peek_charcode_remove:NTF ! {
2140      \stex_term_custom:nn { #1 } { }
2141    }{
2142      \prop_set_eq:Nc \l_tmpa_prop {
2143        g_stex_symdecl_ #1 _prop
2144      }
2145      \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
2146      \exp_args:Nnx \stex_term_custom:nn { #1 } { \l_tmpa_str }
2147    }
2148  }
```

(*End definition for* `\__stex_notation_invoke_text:n`.)

4.8   Terms

```
2149  ⟨@@=stex_term⟩
```

Precedences:

`\infprec`
`\neginfprec`
`\l__stex_term_downprec`

```
2150  \tl_const:Nx \infprec {\int_use:N \c_max_int}
2151  \tl_const:Nx \neginfprec {-\int_use:N \c_max_int}
2152  \int_new:N \l__stex_term_downprec
2153  \int_set_eq:NN \l__stex_term_downprec \infprec
```

(*End definition for* \infprec*,* \neginfprec*, and* \l__stex_term_downprec*. These variables are documented on page* 23*.*)

Bracketing:

```
2154 \tl_set:Nn \l__stex_term_left_bracket_str (
2155 \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

```
2156 \cs_new_protected:Nn \__stex_term_maybe_brackets:nn {
2157   \int_compare:nNnTF { #1 } > \l__stex_term_downprec {
2158     \bool_if:NTF \l_stex_inparray_bool { #2 }{
2159       \dobrackets { #2 }
2160     }
2161   }{ #2 }
2162 }
```

(*End definition for* __stex_term_maybe_brackets:nn*.*)

\dobrackets

```
2163 %\RequirePackage{scalerel}
2164 \cs_new_protected:Npn \dobrackets #1 {
2165   %\ThisStyle{\if D\m@switch
2166   %   \exp_args:Nnx \use:nn
2167   %   { \exp_after:wN \left\l__stex_term_left_bracket_str #1 }
2168   %   { \exp_not:N\right\l__stex_term_right_bracket_str }
2169   % \else
2170     \exp_args:Nnx \use:nn
2171     { \l__stex_term_left_bracket_str #1 }
2172     { \l__stex_term_right_bracket_str }
2173   %\fi}
2174 }
```

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

\withbrackets

```
2175 \cs_new_protected:Npn \withbrackets #1 #2 #3 {
2176   \exp_args:Nnx \use:nn
2177   {
2178     \tl_set:Nx \l__stex_term_left_bracket_str { #1 }
2179     \tl_set:Nx \l__stex_term_right_bracket_str { #2 }
2180     #3
2181   }
2182   {
2183     \tl_set:Nn \exp_not:N \l__stex_term_left_bracket_str
2184       {\l__stex_term_left_bracket_str}
2185     \tl_set:Nn \exp_not:N \l__stex_term_right_bracket_str
2186       {\l__stex_term_right_bracket_str}
2187   }
2188 }
```

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

```
2189 \cs_new_protected:Npn \STEXinvisible #1 {
2190   \stex_annotate_invisible:n { #1 }
2191 }
```

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

OMDoc terms:

```
2192 \cs_new_protected:Nn \_stex_term_oms:nnn {
2193   \stex_annotate:nnn{ OMID }{ #2 }{
2194     \stex_highlight_term:nn { #1 } { #3 }
2195   }
2196 }
2197
2198 \cs_new_protected:Nn \_stex_term_math_oms:nnnn {
2199   \__stex_term_maybe_brackets:nn { #3 }{
2200     \_stex_term_oms:nnn { #1 } { #1\c_hash_str#2 } { #4 }
2201   }
2202 }
```

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

```
2203 \cs_new_protected:Nn \_stex_term_oma:nnn {
2204   \stex_annotate:nnn{ OMA }{ #2 }{
2205     \stex_highlight_term:nn { #1 } { #3 }
2206   }
2207 }
2208
2209 \cs_new_protected:Nn \_stex_term_math_oma:nnnn {
2210   \__stex_term_maybe_brackets:nn { #3 }{
2211     \_stex_term_oma:nnn { #1 } { #1\c_hash_str#2 } { #4 }
2212   }
2213 }
```

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

```
2214 \cs_new_protected:Nn \_stex_term_ombind:nnn {
2215   \stex_annotate:nnn{ OMBIND }{ #2 }{
2216     \stex_highlight_term:nn { #1 } { #3 }
2217   }
2218 }
2219
2220 \cs_new_protected:Nn \_stex_term_math_omb:nnnn {
2221   \__stex_term_maybe_brackets:nn { #3 }{
2222     \_stex_term_ombind:nnn { #1 } { #1\c_hash_str#2 } { #4 }
2223   }
2224 }
```

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

_stex_term_math_arg:nnn

```
2225 \cs_new_protected:Nn \_stex_term_arg:nn {
2226   \stex_unhighlight_term:n {
2227     \stex_annotate:nnn{ arg }{ #1 }{ #2 }
2228   }
2229 }
2230 \cs_new_protected:Nn \_stex_term_math_arg:nnn {
2231   \exp_args:Nnx \use:nn
2232     { \int_set:Nn \l__stex_term_downprec { #2 }
2233       \_stex_term_arg:nn { #1 }{ #3 }
2234     }
2235     { \int_set:Nn \exp_not:N \l__stex_term_downprec { \int_use:N \l__stex_term_downprec } }
2236 }
```

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

_stex_term_math_assoc_arg:nnnn

```
2237 \cs_new_protected:Nn \_stex_term_math_assoc_arg:nnnn {
2238   \seq_set_split:Nnn \l_tmpa_seq , { #4 }
2239   \int_compare:nNnTF { \seq_count:N \l_tmpa_seq } < 2 {
2240     \tl_set:Nn \l_tmpa_tl { #4 }
2241   }{
2242     \cs_set:Npn \l_tmpa_cs ##1 ##2 { #3 }
2243     \seq_reverse:N \l_tmpa_seq
2244     \seq_pop_left:NN \l_tmpa_seq \l_tmpb_tl
2245     \tl_set:No \l_tmpa_tl { \l_tmpb_tl }
2246
2247     \seq_map_inline:Nn \l_tmpa_seq {
2248       \exp_args:NNo \tl_set:No \l_tmpa_tl {
2249         \exp_args:Nno
2250         \l_tmpa_cs { ##1 } \l_tmpa_tl
2251       }
2252     }
2253
2254   }
2255   \exp_args:Nnno
2256   \_stex_term_math_arg:nnn{#1}{#2}\l_tmpa_tl
2257 }
```

(End definition for _stex_term_math_assoc_arg:nnnn. *This function is documented on page 23.)*

\stex_term_custom:nn

```
2258 \cs_new_protected:Nn \stex_term_custom:nn {
2259   \str_set:Nn \l__stex_term_custom_uri { #1 }
2260   \str_set:Nn \l_tmpa_str { #2 }
2261   \tl_clear:N \l_tmpa_tl
2262   \int_zero:N \l_tmpa_int
2263   \int_set:Nn \l_tmpb_int { \str_count:N \l_tmpa_str }
2264   \__stex_term_custom_loop:
2265 }
```

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

__stex_term_custom_loop:

```
2266 \cs_new_protected:Nn \__stex_term_custom_loop: {
2267   \bool_set_false:N \l_tmpa_bool
2268   \bool_while_do:nn {
2269     \str_if_eq_p:ee X {
2270       \str_item:Nn \l_tmpa_str { \l_tmpa_int + 1 }
2271     }
2272   }{
2273     \int_incr:N \l_tmpa_int
2274   }
2275
2276   \peek_charcode:NTF [ {
2277     % notation/text component
2278     \__stex_term_custom_component:w
2279   } {
2280     \int_compare:nNnTF \l_tmpa_int = \l_tmpb_int {
2281       % all arguments read => finish
2282       \__stex_term_custom_final:
2283     } {
2284       % arguments missing
2285       \peek_charcode_remove:NTF * {
2286         % invisible, specific argument position or both
2287         \peek_charcode:NTF [ {
2288           % visible specific argument position
2289           \__stex_term_custom_arg:wn
2290         } {
2291           % invisible
2292           \peek_charcode_remove:NTF * {
2293             % invisible specific argument position
2294             \__stex_term_custom_arg_inv:wn
2295           } {
2296             % invisible next argument
2297             \__stex_term_custom_arg_inv:wn [ \l_tmpa_int + 1 ]
2298           }
2299         }
2300       } {
2301         % next normal argument
2302         \__stex_term_custom_arg:wn [ \l_tmpa_int + 1 ]
2303       }
2304     }
2305   }
2306 }
```

(End definition for __stex_term_custom_loop:*.)*

__stex_term_custom_arg_inv:wn

```
2307 \cs_new_protected:Npn \__stex_term_custom_arg_inv:wn [ #1 ] #2 {
2308   \bool_set_true:N \l_tmpa_bool
2309   \__stex_term_custom_arg:wn [ #1 ] { #2 }
2310 }
```

(End definition for __stex_term_custom_arg_inv:wn*.)*

__stex_term_custom_arg:wn

```
2311 \cs_new_protected:Npn \__stex_term_custom_arg:wn [ #1 ] #2 {
2312   \str_set:Nx \l_tmpb_str {
2313     \str_item:Nn \l_tmpa_str { #1 }
2314   }
2315   \str_case:VnTF \l_tmpb_str {
2316     { X } { } % TODO throw error ?
2317     { i } { \__stex_term_custom_set_X:n { #1 } }
2318     { b } { \__stex_term_custom_set_X:n { #1 } }
2319     { a } { \__stex_term_custom_set_X:n { #1 } } % TODO ?
2320     { B } { \__stex_term_custom_set_X:n { #1 } } % TODO ?
2321   }{}{
2322     % TODO throw error
2323   }
2324
2325   \bool_if:nTF \l_tmpa_bool {
2326     \tl_put_right:Nx \l_tmpa_tl {
2327       \stex_annotate_invisible:n {
2328         \_stex_term_arg:nn { \int_eval:n { #1 } }
2329           \exp_not:n { { #2 } }
2330       }
2331     }
2332   } {
2333     \tl_put_right:Nx \l_tmpa_tl {
2334       \_stex_term_arg:nn { \int_eval:n { #1 } }
2335         \exp_not:n { { #2 } }
2336     }
2337   }
2338
2339   \__stex_term_custom_loop:
2340 }
```

(*End definition for* __stex_term_custom_arg:wn.)

```
2341 \cs_new_protected:Nn \__stex_term_custom_set_X:n {
2342   \str_set:Nx \l_tmpa_str {
2343     \str_range:Nnn \l_tmpa_str 1 { #1 - 1 }
2344     X
2345     \str_range:Nnn \l_tmpa_str { #1 + 1 } { -1 }
2346   }
2347 }
```

(*End definition for* __stex_term_custom_set_X:n.)

```
2348 \cs_new_protected:Npn \__stex_term_custom_component:w [ #1 ] {
2349   \tl_put_right:Nn \l_tmpa_tl { \comp{ #1 } }
2350   \__stex_term_custom_loop:
2351 }
```

(*End definition for* __stex_term_custom_component:.)

```
2352 \cs_new_protected:Nn \__stex_term_custom_final: {
2353   \int_compare:nNnTF \l_tmpb_int = 0 {
```

```
2354        \exp_args:Nnno \_stex_term_oms:nnn
2355      }{
2356        \str_if_in:NnTF \l_tmpa_str {b} {
2357          \exp_args:Nnno \_stex_term_ombind:nnn
2358        } {
2359          \exp_args:Nnno \_stex_term_oma:nnn
2360        }
2361      }
2362      { \l__stex_term_custom_uri } { \l__stex_term_custom_uri } { \l_tmpa_tl }
2363    }
```

(*End definition for* `\__stex_term_custom_final:.`)

\symname

```
2364    \NewDocumentCommand \symref { m m }{
2365      \STEXsymbol{#1}![#2]
2366    }
2367
2368    \keys_define:nn { stex / symname } {
2369      post    .tl_set_x:N  = \l_stex_symname_post_str
2370    }
2371
2372    \cs_new_protected:Nn \stex_symname_args:n {
2373      \str_clear:N \l_stex_symname_post_str
2374      \keys_set:nn { stex / symname } { #1 }
2375      \exp_args:NNo \str_set:Nn \l_stex_symname_post_str
2376        \l_stex_symname_post_str
2377    }
2378
2379    \NewDocumentCommand \symname { O{} m }{
2380      \stex_symname_args:n { #1 }
2381      \stex_get_symbol:n { #2 }
2382      \str_set:Nx \l_tmpa_str {
2383        \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
2384      }
2385      \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
2386      \exp_args:NNx \use:nn
2387      \stex_invoke_symbol:n { { \l_stex_get_symbol_uri_str }![
2388        \l_tmpa_str \l_stex_symname_post_str
2389      ] }
2390    }
```

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

4.9 Notation Components

```
2391    ⟨@@=stex_notationcomps⟩
```

```
2392    \latexml_if:F {
2393      \scalatex_if:F{
2394      % \RequirePackage{pdfcomment}
2395      }
2396    }
2397
```

78

```
2398  \str_new:N \l__stex_notationcomps_highlight_uri_str
2399  \cs_new_protected:Nn \stex_highlight_term:nn {
2400    \exp_args:Nnx
2401    \use:nn {
2402      \str_set:Nx \l__stex_notationcomps_highlight_uri_str { #1 }
2403      #2
2404    } {
2405      \str_set:Nx \exp_not:N \l__stex_notationcomps_highlight_uri_str
2406        { \l__stex_notationcomps_highlight_uri_str }
2407    }
2408  }
2409
2410  \cs_new_protected:Nn \stex_unhighlight_term:n {
2411  %  \latexml_if:TF {
2412  %     #1
2413  %  } {
2414  %     \scalatex_if:TF {
2415  %        #1
2416  %     } {
2417         #1 %\iffalse{{\fi}} #1 {{\iffalse}}\fi
2418  %     }
2419  %  }
2420  }
```

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

```
2421  \cs_new_protected:Npn \comp #1 {
2422    \str_if_empty:NF \l__stex_notationcomps_highlight_uri_str {
2423      \scalatex_if:TF {
2424        \stex_annotate:nnn { comp }{ \l__stex_notationcomps_highlight_uri_str }{ #1 }
2425      }{
2426        \exp_args:Nnx \@comp { #1 } { \l__stex_notationcomps_highlight_uri_str }
2427      }
2428    }
2429  }
2430
2431  \cs_new_protected:Npn \@comp #1 #2 {
2432  % \pdftooltip {
2433      \textcolor{blue}{#1}
2434  % } { #2 }
2435  }
2436
2437  \cs_new_protected:Npn \@defemph #1 #2 {
2438  % \pdftooltip {
2439      \textbf{\textcolor{magenta}{#1}}
2440  % } { #2 }
2441  }
```

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

```
2442  \NewDocumentCommand \ellipses {} { \ldots }
```

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

79

<div style="text-align: right;">\parray</div>
<div style="text-align: right;">\prmatrix</div>
<div style="text-align: right;">\parrayline</div>
<div style="text-align: right;">\parraycell</div>

```
2443 \bool_new:N \l_stex_inparray_bool
2444 \bool_set_false:N \l_stex_inparray_bool
2445 \NewDocumentCommand \parray { m m } {
2446   \begingroup
2447   \bool_set_true:N \l_stex_inparray_bool
2448   \begin{array}{#1}
2449     #2
2450   \end{array}
2451   \endgroup
2452 }
2453
2454 \NewDocumentCommand \prmatrix { m } {
2455   \begingroup
2456   \bool_set_true:N \l_stex_inparray_bool
2457   \begin{matrix}
2458     #1
2459   \end{matrix}
2460   \endgroup
2461 }
2462
2463 \def \parrayline #1 #2 {
2464   #1 #2 \bool_if:NT \l_stex_inparray_bool {\\}
2465 }
2466
2467 \def \parraycell #1 {
2468   #1 \bool_if:NT \l_stex_inparray_bool {&}
2469 }
```

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

4.10 Structural Features

```
2470 ⟨@@=stex_features⟩
```

symboldoc

```
2471 \NewDocumentEnvironment{symboldoc}{ m }{
2472   \seq_set_split:Nnn \l_tmpa_seq , { #1 }
2473   \seq_clear:N \l_tmpb_seq
2474   \seq_map_inline:Nn \l_tmpa_seq {
2475     \stex_get_symbol:n { ##1 }
2476     \exp_args:NNo \seq_put_right:Nn \l_tmpb_seq {
2477       \l_stex_get_symbol_uri_str
2478     }
2479   }
2480   \par
2481   \exp_args:Nnnx
2482   \begin{stex_annotate_env}{symboldoc}{\seq_use:Nn \l_tmpb_seq {,}}
2483 }{
2484   \end{stex_annotate_env}
2485 }
```

STEXdefinition

```
2486
```

```
2487  \NewDocumentCommand \stex_definiendum:w { O{} m m} {
2488    \stex_get_symbol:n { #2 }
2489    \scalatex_if:TF {
2490      \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } { #3 }
2491    } {
2492      \exp_args:Nnx \@defemph { #3 } { \l_stex_get_symbol_uri_str }
2493    }
2494  }
2495  \NewDocumentCommand \stex_definame:w { O{} m } {
2496    % TODO: root
2497    \stex_get_symbol:n { #2 }
2498    \str_set:Nx \l_tmpa_str {
2499      \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
2500    }
2501    \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
2502    \scalatex_if:TF {
2503      \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
2504        \l_tmpa_str
2505      }
2506    } {
2507      \@defemph {
2508        \l_tmpa_str
2509      } { \l_stex_get_symbol_uri_str }
2510    }
2511  }
2512
2513  \cs_new_protected:Nn \__stex_features_defi_begin:n {
2514    \let\definiendum\stex_definiendum:w
2515    \let\definame\stex_definame:w
2516    \seq_set_split:Nnn \l_tmpa_seq , { #1 }
2517    \seq_clear:N \l_tmpb_seq
2518    \seq_map_inline:Nn \l_tmpa_seq {
2519      \stex_get_symbol:n { ##1 }
2520      \exp_args:NNo \seq_put_right:Nn \l_tmpb_seq {
2521        \l_stex_get_symbol_uri_str
2522      }
2523    }
2524    \exp_args:Nnnx
2525    \begin{stex_annotate_env}{definition}{\seq_use:Nn \l_tmpb_seq {,}}
2526  }
2527
2528  \cs_new_protected:Nn \__stex_features_defi_end: {
2529    \end{stex_annotate_env}
2530  }
2531
2532  \NewDocumentEnvironment{STEXdefinition}{ m }{
2533    \__stex_features_defi_begin:n { #1 }
2534  }{
2535    \__stex_features_defi_end:
2536  }
```

\setSTEXdefinition

```
2537  \cs_new_protected:Npn \setSTEXdefinition #1 {
2538    \AddToHook{env/#1/before}[stex]{\__stex_features_defi_begin:n{}}
```

```
2539      \AddToHook{env/#1/after}[stex]{\__stex_features_defi_end:}
2540 }
```

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

structural@feature

```
2541
2542 \NewDocumentEnvironment{structural@feature}{ m m m }{
2543    \stex_if_in_module:F {
2544       \msg_set:nnn{stex}{error/nomodule}{
2545          Structural~Feature~has~to~occur~in~a~module:\\
2546          Feature~#2~of~type~#1\\
2547          In~File:~\stex_path_to_string:N \g_stex_currentfile_seq
2548       }
2549       \msg_error:nn{stex}{error/nomodule}
2550    }
2551
2552    \str_set:Nx \l_stex_module_name_str {
2553       \prop_item:Nn \l_stex_current_module_prop
2554          { name } / #2 - feature
2555    }
2556
2557    \str_set:Nx \l_stex_module_ns_str {
2558       \prop_item:Nn \l_stex_current_module_prop
2559          { ns }
2560    }
2561
2562
2563    \str_clear:N \l_tmpa_str
2564    \seq_clear:N \l_tmpa_seq
2565    \tl_clear:N \l_tmpa_tl
2566    \exp_args:NNx \prop_set_from_keyval:Nn \l_stex_current_module_prop {
2567       origname  = #2,
2568       name      = \l_stex_module_name_str ,
2569       ns        = \l_stex_module_ns_str ,
2570       imports   = \exp_not:o { \l_tmpa_seq } ,
2571       constants = \exp_not:o { \l_tmpa_seq } ,
2572       content   = \exp_not:o { \l_tmpa_tl }   ,
2573       file      = \exp_not:o { \g_stex_currentfile_seq } ,
2574       lang      = \l_stex_module_lang_str ,
2575       sig       = \l_tmpa_str ,
2576       meta      = \l_tmpa_str ,
2577       feature   = #1 ,
2578    }
2579
2580    \stex_if_smsmode:TF {
2581       \stex_smsmode_set_codes:
2582    } {
2583       \begin{stex_annotate_env}{ feature:#1 }{}
2584          \stex_annotate_invisible:nnn{header}{}{ #3 }
2585    }
2586 }{
2587    \str_set:Nx \l_tmpa_str {
2588       c_stex_feature_
```

```
2589      \prop_item:Nn \l_stex_current_module_prop { ns } ?
2590      \prop_item:Nn \l_stex_current_module_prop { name }
2591      _prop
2592    }
2593    \prop_gset_eq:cN { \l_tmpa_str } \l_stex_current_module_prop
2594    \prop_gset_eq:NN \g_stex_last_feature_prop \l_stex_current_module_prop
2595    \stex_if_smsmode:TF {
2596      \exp_args:Nx \stex_addtosms:n {
2597        \prop_gset_from_keyval:cn {
2598          c_stex_feature_
2599          \prop_item:Nn \l_stex_current_module_prop { ns } ?
2600          \prop_item:Nn \l_stex_current_module_prop { name }
2601          _prop
2602        } {
2603          origname  = #2,
2604          name      = \prop_item:cn { \l_tmpa_str } { name } ,
2605          ns        = \prop_item:cn { \l_tmpa_str } { ns } ,
2606          imports   = \prop_item:cn { \l_tmpa_str } { imports } ,
2607          constants = \prop_item:cn { \l_tmpa_str } { constants } ,
2608          content   = \prop_item:cn { \l_tmpa_str } { content } ,
2609          file      = \prop_item:cn { \l_tmpa_str } { file } ,
2610          lang      = \prop_item:cn { \l_tmpa_str } { lang } ,
2611          sig       = \prop_item:cn { \l_tmpa_str } { sig } ,
2612          meta      = \prop_item:cn { \l_tmpa_str } { meta } ,
2613          feature   = \prop_item:cn { \l_tmpa_str } { feature }
2614        }
2615      }
2616    } {
2617        \end{stex_annotate_env}
2618    }
2619  }
2620
```

structure

```
2621
2622  \prop_new:N \l_stex_all_structures_prop
2623
2624  \keys_define:nn { stex / features / structure } {
2625    name          .tl_set_x:N  = \l__stex_features_structure_name_str ,
2626  }
2627
2628  \cs_new_protected:Nn \__stex_features_structure_args:n {
2629    \str_clear:N \l__stex_features_structure_name_str
2630    \keys_set:nn { stex / features / structure } { #1 }
2631    \exp_args:NNo \str_set:Nn \l__stex_features_structure_name_str
2632      \l__stex_features_structure_name_str
2633  }
2634
2635  %\stex_new_feature:nnnn { structure } { O{} m } {
2636  %  \__stex_features_structure_args:n { ##1 }
2637  %  \str_if_empty:NT \l__stex_features_structure_name_str {
2638  %    \str_set:Nx \l__stex_features_structure_name_str { ##2 }
2639  %  }
2640  %} {
```

```
2641 %
2642 %}
2643
2644 \NewDocumentEnvironment{structure}{ O{} m }{
2645   \__stex_features_structure_args:n { #1 }
2646   \str_if_empty:NT \l__stex_features_structure_name_str {
2647     \str_set:Nx \l__stex_features_structure_name_str { #2 }
2648   }
2649   \exp_args:Nnnx
2650   \begin{structural@feature}{ structure }
2651     { \l__stex_features_structure_name_str }{}
2652     \seq_clear:N \l_tmpa_seq
2653     \prop_put:Nno \l_stex_current_module_prop { fields } \l_tmpa_seq
2654
2655 }{
2656     \prop_get:NnN \l_stex_current_module_prop { constants } \l_tmpa_seq
2657     \prop_get:NnN \l_stex_current_module_prop { fields } \l_tmpb_seq
2658     \str_set:Nx \l_tmpa_str {
2659       \prop_item:Nn \l_stex_current_module_prop { ns } ?
2660       \prop_item:Nn \l_stex_current_module_prop { name }
2661     }
2662     \seq_map_inline:Nn \l_tmpa_seq {
2663       \exp_args:NNx \seq_put_right:Nn \l_tmpb_seq { \l_tmpa_str ? ##1 }
2664     }
2665     \prop_put:Nno \l_stex_current_module_prop { fields } { \l_tmpb_seq }
2666     \exp_args:Nnx
2667     \AddToHookNext { env / structure / after }{
2668       \symdecl[type = \exp_not:N\collection,def={\STEXsymbol{module-type}{
2669         \_stex_term_math_oms:nnnn { \l_tmpa_str }{}{0}{}
2670       }}, name = \prop_item:Nn \l_stex_current_module_prop { origname }]{ #2 }
2671       \STEXexport {
2672         \prop_put:Nno \exp_not:N \l_stex_all_structures_prop
2673           {\prop_item:Nn \l_stex_current_module_prop { origname }}
2674           {\l_tmpa_str}
2675         \prop_put:Nno \exp_not:N \l_stex_all_structures_prop
2676           {#2}{\l_tmpa_str}
2677 %        \seq_put_right:Nn \exp_not:N \l_stex_all_structures_seq {
2678 %          \prop_item:Nn \l_stex_current_module_prop { origname },
2679 %          \l_tmpa_str
2680 %        }
2681 %        \seq_put_right:Nn \exp_not:N \l_stex_all_structures_seq {
2682 %          #2,\l_tmpa_str
2683 %        }
2684 %        \tl_set:cx { #2 } {
2685 %          \stex_invoke_structure:n { \l_tmpa_str }
2686       }
2687     }
2688
2689   \end{structural@feature}
2690   % \g_stex_last_feature_prop
2691 }
```

\instantiate

```
2692 \seq_new:N \l__stex_features_structure_field_seq
```

```
2693  \str_new:N \l__stex_features_structure_field_str
2694  \str_new:N \l__stex_features_structure_def_tl
2695  \prop_new:N \l__stex_features_structure_prop
2696  \NewDocumentCommand \instantiate { m O{} m }{
2697    \stex_smsmode_set_codes:
2698    \prop_get:NnN \l_stex_all_structures_prop {#1} \l_tmpa_str
2699    \prop_set_eq:Nc \l__stex_features_structure_prop {
2700      c_stex_feature_\l_tmpa_str _prop
2701    }
2702    \seq_set_from_clist:Nn \l__stex_features_structure_field_seq { #2 }
2703    \seq_map_inline:Nn \l__stex_features_structure_field_seq {
2704      \seq_set_split:Nnn \l_tmpa_seq{=}{ ##1 }
2705      \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > 1 {
2706        \seq_get_left:NN \l_tmpa_seq \l_tmpa_tl
2707        \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq
2708          {!} \l_tmpa_tl
2709        \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} > 1 {
2710          \str_set:Nx \l__stex_features_structure_field_str {\seq_item:Nn \l_tmpb_seq 1}
2711          \seq_get_right:NN \l_tmpb_seq \l_tmpb_tl
2712          \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
2713        }{
2714          \str_set:Nx \l__stex_features_structure_field_str \l_tmpa_tl
2715          \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
2716          \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq{!}
2717            \l_tmpa_tl
2718          \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} > 1 {
2719            \seq_get_left:NN \l_tmpb_seq \l_tmpa_tl
2720            \seq_get_right:NN \l_tmpb_seq \l_tmpb_tl
2721          }{
2722            \tl_clear:N \l_tmpb_tl
2723          }
2724        }
2725      }{
2726        \seq_set_split:Nnn \l_tmpa_seq{!}{ ##1 }
2727        \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > 1 {
2728          \str_set:Nx \l__stex_features_structure_field_str {\seq_item:Nn \l_tmpa_seq 1}
2729          \seq_get_right:NN \l_tmpa_seq \l_tmpb_tl
2730          \tl_clear:N \l_tmpa_tl
2731        }{
2732          % TODO throw error
2733        }
2734      }
2735      % \l_tmpa_str: name
2736      % \l_tmpa_tl: definiens
2737      % \l_tmpb_tl: notation
2738      \tl_if_empty:NT \l__stex_features_structure_field_str {
2739        % TODO throw error
2740      }
2741      \str_clear:N \l_tmpb_str
2742
2743      \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
2744      \seq_map_inline:Nn \l_tmpa_seq {
2745        \seq_set_split:Nnn \l_tmpb_seq ? { ####1 }
2746        \seq_get_right:NN \l_tmpb_seq \l_tmpb_str
```

```
2747        \str_if_eq:NNT \l__stex_features_structure_field_str \l_tmpb_str {
2748          \seq_map_break:n {
2749            \str_set:Nn \l_tmpb_str { ####1 }
2750          }
2751        }
2752      }
2753      \prop_get:cnN { g_stex_symdecl_ \l_tmpb_str _prop } {args}
2754        \l_tmpb_str
2755
2756      \tl_if_empty:NTF \l_tmpb_tl {
2757        \tl_if_empty:NF \l_tmpa_tl {
2758          \exp_args:Nx \use:n {
2759            \symdecl[args=\l_tmpb_str,def={\exp_args:No\exp_not:n{\l_tmpa_tl}}]{#3/\l__stex_fe
2760          }
2761        }
2762      }{
2763        \tl_if_empty:NTF \l_tmpa_tl {
2764          \exp_args:Nx \use:n {
2765            \symdef[args=\l_tmpb_str]{#3/\l__stex_features_structure_field_str}\exp_after:wN\e
2766          }
2767
2768        }{
2769          \exp_args:Nx \use:n {
2770            \symdef[args=\l_tmpb_str,def={\exp_args:No\exp_not:n{\l_tmpa_tl}}]{#3/\l__stex_fea
2771            \exp_after:wN\exp_not:n\exp_after:wN{\l_tmpb_tl}
2772          }
2773        }
2774      }
2775 %    \par \prop_item:Nn \l_stex_current_module_prop {ns} ?
2776 %    \prop_item:Nn \l_stex_current_module_prop {name} ?
2777 %    #3/\l__stex_features_structure_field_str
2778 %    \par
2779 %    \expandafter\present\csname
2780 %      g_stex_symdecl_
2781 %      \prop_item:Nn \l_stex_current_module_prop {ns} ?
2782 %      \prop_item:Nn \l_stex_current_module_prop {name} ?
2783 %      #3/\l__stex_features_structure_field_str
2784 %      _prop
2785 %    \endcsname
2786    }
2787
2788    \tl_clear:N \l__stex_features_structure_def_tl
2789
2790    \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
2791    \seq_map_inline:Nn \l_tmpa_seq {
2792      \seq_set_split:Nnn \l_tmpb_seq ? { ##1 }
2793      \seq_get_right:NN \l_tmpb_seq \l_tmpa_str
2794      \exp_args:Nx \use:n {
2795        \tl_put_right:Nn \exp_not:N \l__stex_features_structure_def_tl {
2796
2797        }
2798      }
2799
2800    \prop_if_exist:cF {
```

86

```
2801        g_stex_symdecl_
2802        \prop_item:Nn \l_stex_current_module_prop {ns} ?
2803        \prop_item:Nn \l_stex_current_module_prop {name} ?
2804        #3/\l_tmpa_str
2805        _prop
2806      }{
2807      \prop_get:cnN { g_stex_symdecl_ ##1 _prop } {args}
2808        \l_tmpb_str
2809      \exp_args:Nx \use:n {
2810        \symdecl[args=\l_tmpb_str]{#3/\l_tmpa_str}
2811      }
2812    }
2813    }
2814
2815    \symdecl*[type={\STEXsymbol{module-type}{
2816      \_stex_term_math_oms:nnnn {
2817        \prop_item:Nn \l__stex_features_structure_prop {ns} ?
2818        \prop_item:Nn \l__stex_features_structure_prop {name}
2819      }{}{0}{}
2820  }}]{#3}
2821
2822    % TODO: -> sms file
2823
2824    \tl_set:cx{ #3 }{
2825      \stex_invoke_structure:nnn {
2826        \prop_item:Nn \l_stex_current_module_prop {ns} ?
2827        \prop_item:Nn \l_stex_current_module_prop {name} ? #3
2828      } {
2829        \prop_item:Nn \l__stex_features_structure_prop {ns} ?
2830        \prop_item:Nn \l__stex_features_structure_prop {name}
2831      }
2832    }
2833
2834 }
```

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

\stex_invoke_structure:nnn

```
2835 % #1: URI of the instance
2836 % #2: URI of the instantiated module
2837 \cs_new_protected:Nn \stex_invoke_structure:nnn {
2838   \tl_if_empty:nTF{ #3 }{
2839     \prop_set_eq:Nc \l__stex_features_structure_prop {
2840       c_stex_feature_ #2 _prop
2841     }
2842   \tl_clear:N \l_tmpa_tl
2843   \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
2844   \seq_map_inline:Nn \l_tmpa_seq {
2845     \seq_set_split:Nnn \l_tmpb_seq ? { ##1 }
2846     \seq_get_right:NN \l_tmpb_seq \l_tmpa_str
2847     \cs_if_exist:cT {
2848       stex_notation_ #1/\l_tmpa_str \c_hash_str\c_hash_str _cs
2849     }{
2850       \tl_if_empty:NF \l_tmpa_tl {
```

```
2851            \tl_put_right:Nn \l_tmpa_tl {,}
2852          }
2853          \tl_put_right:Nx \l_tmpa_tl {
2854            \stex_invoke_symbol:n {#1/\l_tmpa_str}!
2855          }
2856        }
2857      }
2858      \exp_args:No \mathstruct \l_tmpa_tl
2859    }{
2860      \stex_invoke_symbol:n{#1/#3}
2861    }
2862 }
```

(*End definition for* \stex_invoke_structure:nnn. *This function is documented on page* **??**.)

4.11 Put these somewhere

```
2863 \NewDocumentCommand \MSC {m} {
2864    % TODO
2865 }
```

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

```
2866 \@ifpackageloaded{tikzinput}{
2867    \RequirePackage{stex-tikzinput}
2868 }{}
2869
2870 \AddToHook{begindocument}{
2871    \input{stex-metatheory}
2872 }
2873 ⟨/package⟩
```

4.12 Metatheory

The default meta theory for an sTeX 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 (nth-order) logic, or a Π in dependent type theories.

```
2874 ⟨*metatheory⟩
2875 \ExplSyntaxOn
2876 \str_const:Nn \c_stex_metatheory_ns_str {http://mathhub.info/sTeX}
2877 \begin{@module}[ns=\c_stex_metatheory_ns_str,meta=NONE]{Metatheory}
2878    \ExplSyntaxOff
2879
2880    % is-a (a:A, a \in A, a is an A, etc.)
2881    \symdecl[args=ai]{isa}
2882    \notation[typed]{isa}{#1 \comp: #2}{#1 \comp, #2}
2883    \notation[in]{isa}{#1 \comp\in #2}{#1 \comp, #2}
```

88

```
2884    \notation[pred]{isa}{#2\comp(#1 \comp)}{#1 \comp, #2}
2885
2886    % bind (\forall, \Pi, \lambda etc.)
2887    \symdecl[args=Bi]{bind}
2888    \notation[forall]{bind}{\comp\forall #1.\;#2}{#1 \comp, #2}
2889    \notation[Pi]{bind}{\comp\prod_{#1}#2}{#1 \comp, #2}
2890    \notation[depfun]{bind}{\comp( #1 \comp()\;\to\;} #2}{#1 \comp, #2}
2891
2892    % dummy variable
2893    \symdecl{dummyvar}
2894    \notation[underscore]{dummyvar}{\comp\_}
2895    \notation[dot]{dummyvar}{\comp\cdot}
2896    \notation[dot]{dummyvar}{\comp\cdot}
2897    \notation[dash]{dummyvar}{\comp{{\rm --}}}}
2898
2899    %fromto (function space, Hom-set, implication etc.)
2900    \symdecl[args=ai]{fromto}
2901    \notation[xarrow]{fromto}{#1 \comp\to #2}{#1 \comp\times #2}
2902    \notation[arrow]{fromto}{#1 \comp\to #2}{#1 \comp\to #2}
2903
2904    % mapto (lambda etc.)
2905    %\symdecl[args=Bi]{mapto}
2906    %\notation[mapsto]{mapto}{#1 \comp\mapsto #2}{#1 \comp, #2}
2907    %\notation[lambda]{mapto}{\comp\lambda #1 \comp.\; #2}{#1 \comp, #2}
2908    %\notation[lambdau]{mapto}{\comp\lambda_{#1} \comp.\; #2}{#1 \comp, #2}
2909
2910    % function/operator application
2911    \symdecl[args=ia]{apply}
2912    \notation[prec=0;0x\neginfprec,parens]{apply}{#1 \comp( #2 \comp)}{#1 \comp, #2}
2913    \notation[prec=0;0x\neginfprec,lambda]{apply}{#1 \; #2 }{#1 \; #2}
2914
2915    % ``type'' of all collections (sets,classes,types,kinds)
2916    \symdecl{collection}
2917    \notation[U]{collection}{\comp{\mathcal{U}}}
2918    \notation[set]{collection}{\comp{\textsf{Set}}}
2919
2920    % sequences
2921    \symdecl[args=1]{seqtype}
2922    \notation[kleene]{seqtype}{#1^{\comp\ast}}
2923
2924    \symdef[args=2,li]{sequence-index}{#1_{#2}}
2925    \notation[ui]{sequence-index}{#1^{#2}}
2926
2927    %\symdef[args=3,li]{sequence-from-to}{#1_{#2}\comp{,\ellipses,}#1_{#3}}
2928    %\notation[ui]{sequence-from-to}{#1^{#2}\comp{,\ellipses,}#1^{#3}}
2929    % ^ superceded by \aseqfromto and \livar/\uivar
2930
2931    \symdef[args=a,prec=nobrackets]{aseqdots}{#1\comp{,\ellipses}}{#1\comp,#2}
2932    \symdef[args=ai,prec=nobrackets]{aseqfromto}{#1\comp{,\ellipses\comp,}#2 }{#1\comp,#2}
2933    \symdef[args=aii,prec=nobrackets]{aseqfromtovia}{#1\comp{,\ellipses\comp,}#2\comp{,\ellips
2934
2935    % letin (``let'', local definitions, variable substitution)
2936    \symdecl[args=bii]{letin}
2937    \notation[let]{letin}{\comp{{\rm let}}\;#1\comp{=}#2\;\comp{{\rm in}}\;#3}
```

```
2938    \notation[subst]{letin}{#3 \comp[ #1 \comp/ #2 \comp]}
2939    \notation[frac]{letin}{#3 \comp[ \frac{#2}{#1} \comp]}

2940
2941    % structures
2942    \symdecl*[args=1]{module-type}
2943    \notation{module-type}{\mathtt{MOD} #1}
2944    \symdecl[name=mathematical-structure,args=a]{mathstruct} % TODO
2945    \notation[angle,prec=nobrackets]{mathstruct}{\comp\langle #1 \comp\rangle}{#1 \comp, #2}

2946
2947    \STEXexport{
2948      \let\nappa\apply
2949      \def\nappli#1#2#3#4{\apply{#1}{\naseqli{#2}{#3}{#4}}}
2950      \def\livar{\csname sequence-index\endcsname[li]}
2951      \def\uivar{\csname sequence-index\endcsname[ui]}
2952      \def\naseqli#1#2#3{\aseqfromto{\livar{#1}{#2}}{\livar{#1}{#3}}}
2953      \def\nasequi#1#2#3{\aseqfromto{\uivar{#1}{#2}}{\uivar{#1}{#3}}}
2954    }

2955
2956  \end{@module}
2957  \ExplSyntaxOff
2958  ⟨/metatheory⟩
```

4.13 Auxiliary Packages

4.13.1 tikzinput

```
2959  ⟨*tikzinput⟩
2960  ⟨@@=tikzinput⟩
2961  \ProvidesExplPackage{tikzinput}{2021/08/31}{1.9}{bla}
2962  \RequirePackage{l3keys2e}

2963
2964  \keys_define:nn { tikzinput } {
2965    image    .bool_set:N  = \c_tikzinput_image_bool,
2966    image    .default:n   = false ,
2967  }

2968
2969  \ProcessKeysOptions { tikzinput }

2970
2971  \bool_if:NTF \c_tikzinput_image_bool {
2972    \RequirePackage{graphicx}

2973
2974    \providecommand\usetikzlibrary[]{}
2975    \newcommand\tikzinput[2][]{\includegraphics[#1]{#2}}
2976  }{
2977    \RequirePackage{tikz}
2978    \RequirePackage{standalone}

2979
2980    \newcommand \tikzinput [2] [] {
2981      \setkeys{Gin}{#1}
2982      \ifx \Gin@ewidth \Gin@exclamation
2983        \ifx \Gin@eheight \Gin@exclamation
2984          \input { #2 }
2985        \else
2986          \resizebox{!}{ \Gin@eheight }{
```

```
2987            \input { #2 }
2988          }
2989        \fi
2990      \else
2991        \ifx \Gin@eheight \Gin@exclamation
2992          \resizebox{ \Gin@ewidth }{!}{
2993            \input { #2 }
2994          }
2995        \else
2996          \resizebox{ \Gin@ewidth }{ \Gin@eheight }{
2997            \input { #2 }
2998          }
2999        \fi
3000      \fi
3001    }
3002 }
3003
3004 \newcommand \ctikzinput [2] [] {
3005    \begin{center}
3006      \tikzinput [#1] {#2}
3007    \end{center}
3008 }
3009
3010 \@ifpackageloaded{stex}{
3011    \RequirePackage{stex-tikzinput}
3012 }{}
3013 ⟨/tikzinput⟩
3014 ⟨*stex-tikzinput⟩
3015 \ProvidesExplPackage{stex-tikzinput}{2021/08/31}{1.9}{bla}
3016 \RequirePackage{stex}
3017 \RequirePackage{tikzinput}
3018
3019 % TODO
3020
3021 ⟨/stex-tikzinput⟩
```

4.13.2 sTEX1 Compatibility

```
3022 ⟨*smglom⟩
3023 \RequirePackage{expl3,l3keys2e}
3024 \ProvidesExplClass{smglom}{2021/08/01}{1.9}{sTeX1 compatibility}
3025 \LoadClass[border=1px,varwidth]{standalone}
3026 \setlength\textwidth{15cm}
3027 %\g@addto@macro{\@parboxrestore}{\setlength\parskip{\baselineskip}}
3028 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{stex}}
3029 \ProcessOptions
3030
3031 \RequirePackage{stex-compatibility}
3032 ⟨/smglom⟩
3033
3034 ⟨*compat⟩
3035 ⟨@@=stex_deprec⟩
3036 \ProvidesExplPackage{stex-compatibility}{2021/08/01}{1.9}{bla}
3037 %\RequirePackage[lang={de,en,ro,tr,fr}]{stex}
3038 \RequirePackage[lang=en]{stex}
```

```
3039
3040 \NewDocumentEnvironment { mhmodnl } { O{} m m } {
3041   \msg_set:nnn{stex}{warning/deprecated}{
3042     \\
3043     Environment~mhmodnl~is~deprected! \\
3044     Please~update~module~#2~in~file~
3045     \stex_path_to_string:N \g_stex_currentfile_seq!
3046     \\ \\
3047   }
3048   \msg_warning:nn{stex}{warning/deprecated}
3049
3050   \begin{module}[#1,lang=#3]{#2}
3051     \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
3052     \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
3053     \seq_set_split:NnV \l_tmpb_seq . \l_tmpa_str
3054     \seq_pop_left:NN \l_tmpb_seq \l_tmpa_str
3055     \input { \stex_path_to_string:N \l_tmpa_seq / \l_tmpa_str.tex }
3056 } {
3057   \end{module}
3058 }
3059
3060 \NewDocumentEnvironment { modsig } { O{} m } {
3061   \stex_if_in_module:TF {
3062     \prop_get:NnN \l_stex_current_module_prop {name} \l_tmpa_str
3063     \str_set:Nn \l_tmpb_str { #2 }
3064     \str_if_eq:NNTF \l_tmpa_str \l_tmpb_str {
3065       \prop_set_eq:NN \l_modsig_old_module_prop \l_stex_current_module_prop
3066       \begin{@module}{modsig-#2}
3067       % \prop_set_eq:NN \l_stex_current_module_prop \l_modsig_old_module_prop
3068     } {
3069       \begin{@module}{#2}
3070     }
3071   } {
3072     \begin{@module}{#2}
3073   }
3074 }{
3075   \end{@module}
3076   \AddToHookNext { env / modsig / after }{
3077     \stex_if_in_module:T {
3078       \prop_get:NnN \l_stex_current_module_prop {name} \l_tmpa_str
3079       \str_set:Nn \l_tmpb_str { #2 }
3080       \str_if_eq:NNT \l_tmpa_str \l_tmpb_str {
3081 %       \xdef \g_stex_module_after_group_tl {
3082         \stex_if_smsmode:TF {
3083           \exp_args:Nx
3084           \stex_add_to_current_module:n {
3085             \stex_debug:n{Activating~signature~of~#2}
3086             \exp_not:N \prop_item:cn { c_stex_module_
3087             \prop_item:Nn \l_stex_current_module_prop {ns} ?
3088             \prop_item:Nn \l_stex_current_module_prop {name}
3089             / modsig-#2_prop } { content }
3090           }
3091         }
3092         {
```

```
          \gdef \g_stex_modsig_after_group_tl  {
            \stex_activate_module:n {
              \prop_item:Nn \l_stex_current_module_prop {ns} ?
              \prop_item:Nn \l_stex_current_module_prop {name}
              / modsig-#2
            }

            \exp_args:Nx
            \stex_add_to_current_module:n {
              \stex_activate_module:n {
                \prop_item:Nn \l_stex_current_module_prop {ns} ?
                \prop_item:Nn \l_stex_current_module_prop {name}
                / modsig-#2
              }
            }
          }
          \aftergroup \g_stex_modsig_after_group_tl
        }
      }
    }
  }
}

\cs_new_protected:Npn \gimport {
  \peek_charcode_remove:NTF * {
    \gimport_do:
  } {
    \gimport_do:
  }
}

\NewDocumentCommand \gimport_do: { O{} m } {
  \msg_set:nnn{stex}{warning/deprecated}{
    \\
    \c_backslash_str gimport~is~deprecated! \\
    Please~use~\c_backslash_str importmodule[#1]{#2}~instead!~(in~file~
    \stex_path_to_string:N \g_stex_currentfile_seq)
    \\ \\
  }
  \msg_warning:nn{stex}{warning/deprecated}
  \importmodule[#1]{#2}
}

\cs_new_protected:Npn \guse {
  \peek_charcode_remove:NTF * {
    \guse_do:
  } {
    \guse_do:
  }
}

\NewDocumentCommand \guse_do: { O{} m } {
  \msg_set:nnn{stex}{warning/deprecated}{
    \\
```

```
3147      \c_backslash_str guse~is~deprecated! \\
3148      Please~use~\c_backslash_str usemodule[#1]{#2}~instead!~(in~file~
3149      \stex_path_to_string:N \g_stex_currentfile_seq)
3150      \\ \\
3151    }
3152    \msg_warning:nn{stex}{warning/deprecated}
3153    \usemodule[#1]{#2}
3154 }

3156 \cs_new:Nn \stex_capitalize:n { \uppercase{#1} }

3158 \cs_new_protected:Npn \symi {
3159    \peek_charcode_remove:NTF * {
3160      \symi_do:
3161    } {
3162      \symi_do:
3163    }
3164 }

3166 \NewDocumentCommand \symi_do: { O{} m } {
3167    \msg_set:nnn{stex}{warning/deprecated}{
3168      \\
3169      \c_backslash_str symi~is~deprecated! \\
3170      Please~use~\c_backslash_str symdecl[#1]{#2}~instead!~(in~file~
3171      \stex_path_to_string:N \g_stex_currentfile_seq)
3172      \\ \\
3173    }
3174    \msg_warning:nn{stex}{warning/deprecated}
3175    \symdecl*[#1]{#2}
3176 }

3178 \cs_new_protected:Npn \symii {
3179    \peek_charcode_remove:NTF * {
3180      \symii_do:
3181    } {
3182      \symii_do:
3183    }
3184 }

3186 \NewDocumentCommand \symii_do: { O{} m m } {
3187    \msg_set:nnn{stex}{warning/deprecated}{
3188      \\
3189      \c_backslash_str symii~is~deprecated! \\
3190      Please~use~\c_backslash_str symdecl[#1]{#2-#3}~instead!~(in~file~
3191      \stex_path_to_string:N \g_stex_currentfile_seq)
3192      \\ \\
3193    }
3194    \msg_warning:nn{stex}{warning/deprecated}
3195    \symdecl*[#1]{#2-#3}
3196 }

3198 \cs_new_protected:Npn \symiii {
3199    \peek_charcode_remove:NTF * {
3200      \symiii_do:
```

```
3201    } {
3202      \symiii_do:
3203    }
3204 }
3205
3206 \NewDocumentCommand \symiii_do: { O{} m m m } {
3207    \msg_set:nnn{stex}{warning/deprecated}{
3208      \\
3209      \c_backslash_str symiii~is~deprecated! \\
3210      Please~use~\c_backslash_str symdecl[#1]{#2-#3-#4}~instead!~(in~file~
3211      \stex_path_to_string:N \g_stex_currentfile_seq)
3212      \\ \\
3213    }
3214    \msg_warning:nn{stex}{warning/deprecated}
3215    \symdecl*[#1]{#2-#3-#4}
3216 }
3217
3218 \keys_define:nn { stex / deprec / defi } {
3219    name    .tl_set_x:N = \l_tmpa_str
3220 }
3221
3222 \cs_new_protected:Npn \defi {
3223    \peek_charcode_remove:NTF * {
3224      \defi_do:
3225    } {
3226      \defi_do:
3227    }
3228 }
3229
3230 \NewDocumentCommand \defi_do: { O{} m } {
3231    \str_clear:N \l_tmpa_str
3232    \keys_set:nn { stex / deprec / defi } { #1 }
3233
3234    \str_if_empty:NTF \l_tmpa_str {
3235      \msg_set:nnn{stex}{warning/deprecated}{
3236        \\
3237        \c_backslash_str defi~is~deprecated! \\
3238        Please~use~\c_backslash_str STEXsymbol{#2}![#2]~instead!~(in~file~
3239        \stex_path_to_string:N \g_stex_currentfile_seq)
3240        \\ \\
3241      }
3242      \msg_warning:nn{stex}{warning/deprecated}
3243      \STEXsymbol { #2 }![ \comp{#2} ]
3244    } {
3245      \msg_set:nnn{stex}{warning/deprecated}{
3246        \\
3247        \c_backslash_str defi~is~deprecated! \\
3248        Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2 ]~instead!~(in~file~
3249        \stex_path_to_string:N \g_stex_currentfile_seq)
3250        \\ \\
3251      }
3252      \msg_warning:nn{stex}{warning/deprecated}
3253      \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2} ]
3254    }
```

```
3255 }
3256
3257
3258 \cs_new_protected:Npn \Defi {
3259   \peek_charcode_remove:NTF * {
3260     \Defi_do:
3261   } {
3262     \Defi_do:
3263   }
3264 }
3265
3266 \NewDocumentCommand \Defi_do: { O{} m } {
3267   \str_clear:N \l_tmpa_str
3268   \keys_set:nn { stex / deprec / defi } { #1 }
3269
3270   \str_if_empty:NTF \l_tmpa_str {
3271     \msg_set:nnn{stex}{warning/deprecated}{
3272       \\
3273       \c_backslash_str Defi~is~deprecated! \\
3274       Please~use~\c_backslash_str STEXsymbol{#2}![\exp_after:wN \stex_capitalize:n #2]~inste
3275       \stex_path_to_string:N \g_stex_currentfile_seq)
3276       \\ \\
3277     }
3278     \msg_warning:nn{stex}{warning/deprecated}
3279     \STEXsymbol { #2 }![ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3280   } {
3281     \msg_set:nnn{stex}{warning/deprecated}{
3282       \\
3283       \c_backslash_str Defi~is~deprecated! \\
3284       Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ \exp_after:wN \stex_capitalize
3285       \stex_path_to_string:N \g_stex_currentfile_seq)
3286       \\ \\
3287     }
3288     \msg_warning:nn{stex}{warning/deprecated}
3289     \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3290   }
3291 }
3292
3293 \cs_new_protected:Npn \adefi {
3294   \peek_charcode_remove:NTF * {
3295     \adefi_do:
3296   } {
3297     \adefi_do:
3298   }
3299 }
3300
3301 \NewDocumentCommand \adefi_do: { O{} m m } {
3302   \str_clear:N \l_tmpa_str
3303   \keys_set:nn { stex / deprec / defi } { #1 }
3304
3305   \str_if_empty:NTF \l_tmpa_str {
3306     \msg_set:nnn{stex}{warning/deprecated}{
3307       \\
3308       \c_backslash_str adefi~is~deprecated! \\
```

```
3309          Please~use~\c_backslash_str STEXsymbol{#3}![#2]~instead!~(in~file~
3310          \stex_path_to_string:N \g_stex_currentfile_seq)
3311          \\ \\
3312       }
3313       \msg_warning:nn{stex}{warning/deprecated}
3314       \STEXsymbol { #3 }![ \comp{#2} ]
3315    } {
3316       \msg_set:nnn{stex}{warning/deprecated}{
3317          \\
3318          \c_backslash_str adefi~is~deprecated! \\
3319          Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2 ]~instead!~(in~file~
3320          \stex_path_to_string:N \g_stex_currentfile_seq)
3321          \\ \\
3322       }
3323       \msg_warning:nn{stex}{warning/deprecated}
3324       \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2} ]
3325    }
3326 }
3327
3328 \cs_new_protected:Npn \defis {
3329    \peek_charcode_remove:NTF * {
3330       \defis_do:
3331    } {
3332       \defis_do:
3333    }
3334 }
3335
3336 \NewDocumentCommand \defis_do: { O{} m } {
3337    \str_clear:N \l_tmpa_str
3338    \keys_set:nn { stex / deprec / defi } { #1 }
3339
3340    \str_if_empty:NTF \l_tmpa_str {
3341       \msg_set:nnn{stex}{warning/deprecated}{
3342          \\
3343          \c_backslash_str defis~is~deprecated! \\
3344          Please~use~\c_backslash_str STEXsymbol{#2}![#2s]~instead!~(in~file~
3345          \stex_path_to_string:N \g_stex_currentfile_seq)
3346          \\ \\
3347       }
3348       \msg_warning:nn{stex}{warning/deprecated}
3349       \STEXsymbol { #2 }![ \comp{#2s} ]
3350    } {
3351       \msg_set:nnn{stex}{warning/deprecated}{
3352          \\
3353          \c_backslash_str defis~is~deprecated! \\
3354          Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2s ]~instead!~(in~file~
3355          \stex_path_to_string:N \g_stex_currentfile_seq)
3356          \\ \\
3357       }
3358       \msg_warning:nn{stex}{warning/deprecated}
3359       \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2s} ]
3360    }
3361 }
3362
```

```
3363 \cs_new_protected:Npn \defii {
3364   \peek_charcode_remove:NTF * {
3365     \defii_do:
3366   } {
3367     \defii_do:
3368   }
3369 }
3370
3371 \NewDocumentCommand \defii_do: { O{} m m } {
3372   \str_clear:N \l_tmpa_str
3373   \keys_set:nn { stex / deprec / defi } { #1 }
3374   \str_if_empty:NTF \l_tmpa_str {
3375     \msg_set:nnn{stex}{warning/deprecated}{
3376       \\
3377       \c_backslash_str defii~is~deprecated! \\
3378       Please~use~\c_backslash_str STEXsymbol{#2-#3}![#2~#3]~instead!~(in~file~
3379       \stex_path_to_string:N \g_stex_currentfile_seq)
3380       \\ \\
3381     }
3382     \msg_warning:nn{stex}{warning/deprecated}
3383     \STEXsymbol { #2-#3 }![ \comp{#2~#3} ]
3384   } {
3385     \msg_set:nnn{stex}{warning/deprecated}{
3386       \\
3387       \c_backslash_str defii~is~deprecated! \\
3388       Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2~#3 ]~instead!~(in~file~
3389       \stex_path_to_string:N \g_stex_currentfile_seq)
3390       \\ \\
3391     }
3392     \msg_warning:nn{stex}{warning/deprecated}
3393     \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2~#3} ]
3394   }
3395 }
3396
3397
3398 \cs_new_protected:Npn \defiis {
3399   \peek_charcode_remove:NTF * {
3400     \defiis_do:
3401   } {
3402     \defiis_do:
3403   }
3404 }
3405
3406 \NewDocumentCommand \defiis_do: { O{} m m } {
3407   \str_clear:N \l_tmpa_str
3408   \keys_set:nn { stex / deprec / defi } { #1 }
3409   \str_if_empty:NTF \l_tmpa_str {
3410     \msg_set:nnn{stex}{warning/deprecated}{
3411       \\
3412       \c_backslash_str defiis~is~deprecated! \\
3413       Please~use~\c_backslash_str STEXsymbol{#2-#3}![#2~#3s]~instead!~(in~file~
3414       \stex_path_to_string:N \g_stex_currentfile_seq)
3415       \\ \\
3416     }
```

```
3417    \msg_warning:nn{stex}{warning/deprecated}
3418    \STEXsymbol { #2-#3 }![ \comp{#2~#3s} ]
3419  } {
3420    \msg_set:nnn{stex}{warning/deprecated}{
3421      \\
3422      \c_backslash_str defiis~is~deprecated! \\
3423      Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2~#3s ]~instead!~(in~file~
3424      \stex_path_to_string:N \g_stex_currentfile_seq)
3425      \\ \\
3426    }
3427    \msg_warning:nn{stex}{warning/deprecated}
3428    \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2~#3s} ]
3429  }
3430 }
3431
3432
3433 \cs_new_protected:Npn \defiii {
3434   \peek_charcode_remove:NTF * {
3435     \defiii_do:
3436   } {
3437     \defiii_do:
3438   }
3439 }
3440
3441 \NewDocumentCommand \defiii_do: { O{} m m m } {
3442   \str_clear:N \l_tmpa_str
3443   \keys_set:nn { stex / deprec / defi } { #1 }
3444   \str_if_empty:NTF \l_tmpa_str {
3445     \msg_set:nnn{stex}{warning/deprecated}{
3446       \\
3447       \c_backslash_str defiii~is~deprecated! \\
3448       Please~use~\c_backslash_str STEXsymbol{#2-#3-#4}![#2~#3~#4]~instead!~(in~file~
3449       \stex_path_to_string:N \g_stex_currentfile_seq)
3450       \\ \\
3451     }
3452     \msg_warning:nn{stex}{warning/deprecated}
3453     \STEXsymbol { #2-#3-#4 }![ \comp{#2~#3~#4} ]
3454   } {
3455     \msg_set:nnn{stex}{warning/deprecated}{
3456       \\
3457       \c_backslash_str defiii~is~deprecated! \\
3458       Please~use~\c_backslash_str STEXsymbol { \l_tmpa_str }[ #2~#3~#4 ]~instead!~(in~file~
3459       \stex_path_to_string:N \g_stex_currentfile_seq)
3460       \\ \\
3461     }
3462     \msg_warning:nn{stex}{warning/deprecated}
3463     \exp_args:No \STEXsymbol { \l_tmpa_str }![ \comp{#2~#3~#4} ]
3464   }
3465 }
3466
3467 %\RequirePackage[hyperref]{ntheorem}
3468 %\theoremstyle{plain}
3469 %\RequirePackage{amsthm}
3470
```

```
3471 \NewDocumentEnvironment {definition} { O{} } {
3472   \begin{STEXdefinition}{}
3473 }{
3474   \end{STEXdefinition}
3475 }
3476 \keys_define:nn { stex / omtext} {
3477   id    .tl_set_x:N  = \l_stex_omtext_id_str ,
3478   title   .tl_set_x:N  = \l_stex_omtext_title_str ,
3479   type    .tl_set_x:N  = \l_stex_omtext_type_tl ,
3480   for     .tl_set_x:N  = \l_stex_omtext_for_tl ,
3481   from    .tl_set_x:N  = \l_stex_omtext_from_tl ,
3482   start   .tl_set_x:N  = \l_stex_omtext_start_str ,
3483 }
3484 \cs_new_protected:Nn \stex_omtext_args:n {
3485   \str_clear:N \l_stex_omtext_title_str
3486   \str_clear:N \l_stex_omtext_start_str
3487   \keys_set:nn { stex / omtext }{ #1 }
3488   \exp_args:NNo \str_set:Nn \l_stex_omtext_title_str
3489     \l_stex_omtext_title_str
3490   \exp_args:NNo \str_set:Nn \l_stex_omtext_start_str
3491     \l_stex_omtext_start_str
3492 }
3493 \NewDocumentEnvironment {omtext} { O{} } {
3494   \stex_omtext_args:n { #1 }
3495   \textbf{\str_if_empty:NTF \l_stex_omtext_start_str {
3496     \l_stex_omtext_title_str
3497   }{
3498     \l_stex_omtext_start_str :
3499   }}
3500 }{
3501
3502 }
3503 \NewDocumentEnvironment {assertion} { O{} } {
3504
3505 }{
3506
3507 }
3508
3509 \NewDocumentCommand \inlinedef { m } {
3510   \begingroup
3511   \let\definiendum\stex_definiendum:w
3512   \let\definame\stex_definame:w
3513   #1
3514   \endgroup
3515 }
3516
3517 \NewDocumentCommand \inlineass { m } { #1 }
3518
3519 \NewDocumentCommand \trefi { O{} m } {
3520   \str_set:Nn \l_tmpa_str { #1 }
3521   \str_if_empty:NTF \l_tmpa_str {
3522     \msg_set:nnn{stex}{warning/deprecated}{
3523       \\
3524       \c_backslash_str trefi~is~deprecated! \\
```

```
3525        Please~use~\c_backslash_str STEXsymbol{#2}![#2]~instead!~(in~file~
3526        \stex_path_to_string:N \g_stex_currentfile_seq)
3527        \\ \\
3528      }
3529      \msg_warning:nn{stex}{warning/deprecated}
3530      \STEXsymbol { #2 }![ \comp{#2} ]
3531    } {
3532      \msg_set:nnn{stex}{warning/deprecated}{
3533        \\
3534        \c_backslash_str trefi~is~deprecated! \\
3535        Please~use~\c_backslash_str STEXsymbol { #1?#2 }[ #2 ]~instead!~(in~file~
3536        \stex_path_to_string:N \g_stex_currentfile_seq)
3537        \\ \\
3538      }
3539      \msg_warning:nn{stex}{warning/deprecated}
3540      \STEXsymbol { #1 }![ \comp{#2} ]
3541    }
3542 }
3543
3544
3545 \NewDocumentCommand \Trefi { O{} m } {
3546    \str_set:Nn \l_tmpa_str { #1 }
3547    \str_if_empty:NTF \l_tmpa_str {
3548      \msg_set:nnn{stex}{warning/deprecated}{
3549        \\
3550        \c_backslash_str Trefi~is~deprecated! \\
3551        Please~use~\c_backslash_str STEXsymbol{#2}![\exp_after:wN \stex_capitalize:n #2]~inste
3552        \stex_path_to_string:N \g_stex_currentfile_seq)
3553        \\ \\
3554      }
3555      \msg_warning:nn{stex}{warning/deprecated}
3556      \STEXsymbol { #2 }![ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3557    } {
3558      \msg_set:nnn{stex}{warning/deprecated}{
3559        \\
3560        \c_backslash_str Trefi~is~deprecated! \\
3561        Please~use~\c_backslash_str STEXsymbol { #1 }[ \exp_after:wN \stex_capitalize:n #2 ]~i
3562        \stex_path_to_string:N \g_stex_currentfile_seq)
3563        \\ \\
3564      }
3565      \msg_warning:nn{stex}{warning/deprecated}
3566      \STEXsymbol { #1 }![ \comp{\exp_after:wN \stex_capitalize:n #2} ]
3567    }
3568 }
3569
3570 \NewDocumentCommand \trefis { O{} m } {
3571    \str_set:Nn \l_tmpa_str { #1 }
3572    \str_if_empty:NTF \l_tmpa_str {
3573      \msg_set:nnn{stex}{warning/deprecated}{
3574        \\
3575        \c_backslash_str trefi~is~deprecated! \\
3576        Please~use~\c_backslash_str STEXsymbol{#2}![#2s]~instead!~(in~file~
3577        \stex_path_to_string:N \g_stex_currentfile_seq)
3578        \\ \\
```

```
3579        }
3580      \msg_warning:nn{stex}{warning/deprecated}
3581      \STEXsymbol { #2 }![ \comp{#2s} ]
3582    } {
3583      \msg_set:nnn{stex}{warning/deprecated}{
3584        \\
3585        \c_backslash_str trefi~is~deprecated! \\
3586        Please~use~\c_backslash_str STEXsymbol { #1 }[ #2s ]~instead!~(in~file~
3587        \stex_path_to_string:N \g_stex_currentfile_seq)
3588        \\ \\
3589      }
3590      \msg_warning:nn{stex}{warning/deprecated}
3591      \STEXsymbol { #1 }![ \comp{#2s} ]
3592    }
3593  }
3594
3595
3596  \NewDocumentCommand \Trefis { O{} m } {
3597    \str_set:Nn \l_tmpa_str { #1 }
3598    \str_if_empty:NTF \l_tmpa_str {
3599      \msg_set:nnn{stex}{warning/deprecated}{
3600        \\
3601        \c_backslash_str Trefis~is~deprecated! \\
3602        Please~use~\c_backslash_str STEXsymbol{#2}![\exp_after:wN \stex_capitalize:n #2s]~inst
3603        \stex_path_to_string:N \g_stex_currentfile_seq)
3604        \\ \\
3605      }
3606      \msg_warning:nn{stex}{warning/deprecated}
3607      \STEXsymbol { #2 }![ \comp{\exp_after:wN \stex_capitalize:n #2s} ]
3608    } {
3609      \msg_set:nnn{stex}{warning/deprecated}{
3610        \\
3611        \c_backslash_str Trefis~is~deprecated! \\
3612        Please~use~\c_backslash_str STEXsymbol { #1 }[ \exp_after:wN \stex_capitalize:n #2s ]~
3613        \stex_path_to_string:N \g_stex_currentfile_seq)
3614        \\ \\
3615      }
3616      \msg_warning:nn{stex}{warning/deprecated}
3617      \STEXsymbol { #1 }![ \comp{\exp_after:wN \stex_capitalize:n #2s} ]
3618    }
3619  }
3620
3621  \NewDocumentCommand \trefii { O{} m m } {
3622    \str_set:Nn \l_tmpa_str { #1 }
3623    \str_if_empty:NTF \l_tmpa_str {
3624      \msg_set:nnn{stex}{warning/deprecated}{
3625        \\
3626        \c_backslash_str trefii~is~deprecated! \\
3627        Please~use~\c_backslash_str STEXsymbol{#2-#3}![#2~#3]~instead!~(in~file~
3628        \stex_path_to_string:N \g_stex_currentfile_seq)
3629        \\ \\
3630      }
3631      \msg_warning:nn{stex}{warning/deprecated}
3632      \STEXsymbol { #2-#3 }![ \comp{#2~#3} ]
```

```
3633    } {
3634      \msg_set:nnn{stex}{warning/deprecated}{
3635        \\
3636        \c_backslash_str trefii~is~deprecated! \\
3637        Please~use~\c_backslash_str STEXsymbol { #1 }[ #2~#3 ]~instead!~(in~file~
3638        \stex_path_to_string:N \g_stex_currentfile_seq)
3639        \\ \\
3640      }
3641      \msg_warning:nn{stex}{warning/deprecated}
3642      \STEXsymbol { #1 }![ \comp{#2~#3} ]
3643    }
3644  }
3645
3646  \NewDocumentCommand \trefiii { O{} m m m } {
3647    \str_set:Nn \l_tmpa_str { #1 }
3648    \str_if_empty:NTF \l_tmpa_str {
3649      \msg_set:nnn{stex}{warning/deprecated}{
3650        \\
3651        \c_backslash_str trefiii~is~deprecated! \\
3652        Please~use~\c_backslash_str STEXsymbol{#2-#3-#4}![#2~#3~#4]~instead!~(in~file~
3653        \stex_path_to_string:N \g_stex_currentfile_seq)
3654        \\ \\
3655      }
3656      \msg_warning:nn{stex}{warning/deprecated}
3657      \STEXsymbol { #2-#3-#4 }![ \comp{#2~#3~#4} ]
3658    } {
3659      \msg_set:nnn{stex}{warning/deprecated}{
3660        \\
3661        \c_backslash_str trefiii~is~deprecated! \\
3662        Please~use~\c_backslash_str STEXsymbol { #1 }[ #2~#3~#4 ]~instead!~(in~file~
3663        \stex_path_to_string:N \g_stex_currentfile_seq)
3664        \\ \\
3665      }
3666      \msg_warning:nn{stex}{warning/deprecated}
3667      \STEXsymbol { #1 }![ \comp{#2~#3~#4} ]
3668    }
3669  }
3670
3671
3672  \NewDocumentCommand \trefiis { O{} m m } {
3673    \str_set:Nn \l_tmpa_str { #1 }
3674    \str_if_empty:NTF \l_tmpa_str {
3675      \msg_set:nnn{stex}{warning/deprecated}{
3676        \\
3677        \c_backslash_str trefiis~is~deprecated! \\
3678        Please~use~\c_backslash_str STEXsymbol{#2-#3}![#2~#3s]~instead!~(in~file~
3679        \stex_path_to_string:N \g_stex_currentfile_seq)
3680        \\ \\
3681      }
3682      \msg_warning:nn{stex}{warning/deprecated}
3683      \STEXsymbol { #2-#3 }![ \comp{#2~#3s} ]
3684    } {
3685      \msg_set:nnn{stex}{warning/deprecated}{
3686        \\
```

```
3687        \c_backslash_str trefiis~is~deprecated! \\
3688        Please~use~\c_backslash_str STEXsymbol { #1 }[ #2~#3s ]~instead!~(in~file~
3689        \stex_path_to_string:N \g_stex_currentfile_seq)
3690        \\ \\
3691      }
3692      \msg_warning:nn{stex}{warning/deprecated}
3693      \STEXsymbol { #1 }![ \comp{#2~#3s} ]
3694    }
3695  }
3696
3697  \NewDocumentCommand \symvariant { O{} m O{O} m m} {
3698    \msg_set:nnn{stex}{warning/deprecated}{
3699      \\
3700      \c_backslash_str symvariant~is~deprecated! \\
3701      Please~use~\c_backslash_str notation[#4]{ #2 }~instead!~(in~file~
3702      \stex_path_to_string:N \g_stex_currentfile_seq)
3703      \\ \\
3704    }
3705    \msg_warning:nn{stex}{warning/deprecated}
3706
3707    \notation[variant=#4]{#2}{#5}
3708  }
3709
3710  \NewDocumentCommand \mixfixi { O{} m m m} {
3711    \msg_set:nnn{stex}{warning/deprecated}{
3712      \c_backslash_str mixfixi~is~fatally~deprecated!\\
3713      Symbol:~\l__stex_term_highlight_uri_str\\
3714      Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3715    }
3716    \msg_error:nn{stex}{warning/deprecated}
3717  }
3718
3719
3720  \NewDocumentCommand \infix {} {
3721    \msg_set:nnn{stex}{warning/deprecated}{
3722      \c_backslash_str infix~is~fatally~deprecated!\\
3723      Symbol:~\l__stex_term_highlight_uri_str\\
3724      Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3725    }
3726    \msg_error:nn{stex}{warning/deprecated}
3727  }
3728
3729  \let\iprec\infprec
3730
3731  \NewDocumentCommand \inlineex { m } {
3732    \msg_set:nnn{stex}{warning/deprecated}{
3733      \c_backslash_str inlineex~is~deprecated!\\
3734      No~replacement~exists~yet.\\
3735      Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3736    }
3737    \msg_warning:nn{stex}{warning/deprecated}
3738    #1
3739  }
3740
```

```
3741
3742 \NewDocumentCommand \term { m } {
3743   \msg_set:nnn{stex}{warning/deprecated}{
3744     \c_backslash_str term~is~deprecated!\\
3745     No~replacement~exists~yet.\\
3746     Current~file:~\stex_path_to_string:N \g_stex_currentfile_seq
3747   }
3748   \msg_warning:nn{stex}{warning/deprecated}
3749   #1
3750 }
3751
3752
3753 \NewDocumentCommand \Definame { O{} m } {
3754   \stex_get_symbol:n { #2 }
3755   \str_set:Nx \l_tmpa_str {
3756     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
3757   }
3758   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
3759   \scalatex_if:TF {
3760     \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
3761       \l_tmpa_str
3762     }
3763   } {
3764     \@defemph {
3765       \exp_after:wN \stex_capitalize:n \l_tmpa_str
3766     } { \l_stex_get_symbol_uri_str }
3767   }
3768 }
3769
3770 \NewDocumentCommand \Definiendum { O{} m m } {
3771   \stex_get_symbol:n { #2 }
3772   \str_set:Nx \l_tmpa_str {
3773     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
3774   }
3775   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
3776   \scalatex_if:TF {
3777     \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
3778       \l_tmpa_str
3779     }
3780   } {
3781     \@defemph {
3782       \exp_after:wN \stex_capitalize:n \l_tmpa_str
3783     } { \l_stex_get_symbol_uri_str }
3784   }
3785 }
3786
3787 \NewDocumentCommand \Symname { O{} m }{
3788   \stex_symname_args:n { #1 }
3789   \stex_get_symbol:n { #2 }
3790   \str_set:Nx \l_tmpa_str {
3791     \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
3792   }
3793   \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
3794   \exp_args:NNx \use:nn
```

```
3795     \stex_invoke_symbol:n { { \l_stex_get_symbol_uri_str }![
3796       \exp_after:wN \stex_capitalize:n \l_tmpa_str
3797         \l_stex_symname_post_str
3798     ] } }
3799 }
3800

3801

3802 \seq_gput_right:Nx \g_stex_smsmode_allowedenvs_seq { \tl_to_str:n { mhmodnl } }
3803 \seq_gput_right:Nx \g_stex_smsmode_allowedenvs_seq { \tl_to_str:n { modsig } }
3804 \tl_gput_right:Nn \g_stex_smsmode_allowedmacros_escape_tl {\gimport\symi\symii\symiii\symiv\
3805

3806 % omtext:
3807 \cs_new_protected:Npn \lec #1 {
3808   \strut\hfil\strut\null\hfill(#1)
3809 }
3810 \cs_new_protected:Npn \nlex #1 {
3811   \textcolor{green}{{\sl #1}}
3812 }
3813

3814 \newcommand\hateq{\ensuremath{\widehat=}\xspace}
3815 \newcommand\hatequiv{\ensuremath{\widehat\equiv}\xspace}
3816 \@ifundefined{ergo}%
3817 {\newcommand\ergo{\ensuremath{\leadsto}\xspace}}%
3818 {\renewcommand\ergo{\ensuremath{\leadsto}\xspace}}%
3819 \newcommand{\reflect@squig}[2]{\reflectbox{$\m@th#1\rightsquigarrow$}}%
3820 \newcommand\ogre{\ensuremath{\mathrel{\mathpalette\reflect@squig\relax}}\xspace}%
3821 \newcommand\notergo{\ensuremath{\not\leadsto}}
3822 \newcommand\notogre{\ensuremath{\not\mathrel{\mathpalette\reflect@squig\relax}}\xspace}%
3823

3824 % mathhub convenience macros
3825

3826 \define@key{Gin}{mhrepos}{\def\Gin@mhrepos{#1}}
3827 \newcommand\mhgraphics[2][]{%
3828   \def\Gin@mhrepos{}\setkeys{Gin}{#1}%
3829   \includegraphics[#1]{\mhpath\Gin@mhrepos{#2}}}
3830 \newcommand\cmhgraphics[2][]{\begin{center}\mhgraphics[#1]{#2}\end{center}}
3831

3832 \newcommand\mhtikzinput[2][]{%
3833   \def\Gin@mhrepos{}\setkeys{Gin}{#1}%
3834   \stex_in_repository:nn\Gin@mhrepos{
3835     \tikzinput[#1]{\mhpath{##1}{#2}}
3836   }
3837 }
3838 \newcommand\cmhtikzinput[2][]{\begin{center}\mhgraphics[#1]{#2}\end{center}}
3839

3840 \newcommand\lstinputmhlisting[2][]{%
3841   \def\lst@mhrepos{}\setkeys{lst}{#1}%
3842   \lstinputlisting[#1]{\mhpath\lst@mhrepos{#2}}}
3843 \newcommand\clstinputmhlisting[2][]{\begin{center}\lstinputmhlisting[#1]{#2}\end{center}}
3844

3845 %%%%%%%%%%% CHEATING --> implement us %%%%%%%%%%%%%%%%%%
3846 \newcommand\assdef[2][]{#2}
3847 \newcommand\impdec[1]{#1}
3848 \newenvironment{inlineAssertion}{}{}
```

```
3849  \newenvironment{sproof}[2][]{}{}
3850  \newcommand\spfsketch[2][]{#2}
3851  \newenvironment{spfstep}[1][]{}{}
3852  \newenvironment{spfcases}[2][]{#2}{}
3853  \newenvironment{spfcase}[2][]{#2}{}
3854  \newcommand\gstructure[3][]{\importmodule[#1]{#3}}
3855  \newcommand\fassign[3]{}
3856  \newcommand\vassign[2]{}
3857  \newcommand\tassign[2]{}
3858  \newenvironment{gviewsig}[4][]{}{}
3859  \newenvironment{gviewnl}[5][]{}{}
3860  \newenvironment{mhview}[5][]{}{}
3861  \newenvironment{axiom}[1][]{}{}
3862  \newenvironment{example}[1][]{}{}
3863  \newenvironment{sblockquote}[1][]{}{}
3864  \newcommand\hypernym[3][1]{}
3865  \newcommand\withcite[2]{}
3866  \newcommand\sref[2][]{}
3867  ⟨/compat⟩
```