The sTeX3 Package [*]

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

2021-12-11

Abstract

TODO

[*]Version 3.0 (last revised 2021-12-11)

Contents

Part I
Manual

Chapter 1

Stuff

1.1 Modules

`\sTeX`
`\stex`

Both print this sTeX logo.

1.1.1 Semantic Macros and Notations

Semantic macros invoke a formally declared symbol.

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

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

Example 1

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

ab

.

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

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

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

Example 2

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

$a{\cdot}b$ and $a{\times}b$

.

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

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} ]{$b$}
```

$a{\ast}b$ 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 $P$holds for every $x \in A$

[1] EdNote: TODO

.

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:

Example 7

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

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

.

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

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.

EdN:2 EdN:3 [2] [3]

[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?

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 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 $a\cdot(b+c)$

.

1.1.2 Archives and Imports

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.

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.

- 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.

[1]which is internally attached to the module name instead, but a user need not worry about that.

Part II
Documentation

Chapter 2

sTeX-Basics

Both the sTeX package and class offer the following package options:

`debug` (⟨*log-prefix*⟩∗) Logs debugging information with the given prefixes to the terminal, or all if `all` is given.

`showmods` (⟨*boolean*⟩) Shows explicit module information at the document margins.

`lang` (⟨*language*⟩∗) Languages to load with the babel package.

`mathhub` (⟨*directory*⟩) MathHub folder to search for repositories.

`sms` (⟨*boolean*⟩) use *persisted* mode (see ???).

`image` (⟨*boolean*⟩) passed on to tikzinput.

2.1 Macros and Environments

`\sTeX`
`\stex`

Both print this sTeX logo.

`\stex_debug:nn`

`\stex_debug:nn {⟨log-prefix⟩} {⟨message⟩}`

Logs ⟨*message*⟩, if the package option `debug` contains ⟨*log-prefix*⟩.

`\stex_add_to_sms:n`

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

`\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 ScaLaTeX) 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⟩}`.

`\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`.

`\stex_deactivate_macro:Nn`
`\stex_reactivate_macro:N`

`\stex_deactivate_macro:Nn⟨cs⟩{⟨environments⟩}`

Makes the macro ⟨*cs*⟩ throw an error, indicating that it is only allowed in the context of ⟨*environments*⟩.

`\stex_reactivate_macro:N⟨cs⟩` reactivates it again, i.e. this happens ideally in the ⟨*begin*⟩-code of the associated environments.

`\MSC`

`\MSC{⟨msc⟩}`

Designates the *math subject classifier* of the current module / file.

Chapter 3

sTeX-MathHub

Code related to managing and using MathHub repositories, files, paths and related hooks and methods.

3.1 Macros and Environments

`\stex_kpsewhich:n`

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

3.1.1 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

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

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/../..		

.

3.1.2 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.

`\stex_in_repository:nn`

`\stex_in_repository:nn{⟨repository-name⟩}{⟨code⟩}`

Change the current repository to `{`⟨*repository-name*⟩`}` (or not, if `{`⟨*repository-name*⟩`}` is empty), and passes its ID on to `{`⟨*code*⟩`}` as `#1`. Switches back to the previous repository after executing `{`⟨*code*⟩`}`.

`\mhpath` ⋆

`\mhpath{⟨archive-ID⟩}{⟨filename⟩}`

Expands to the full path of file ⟨*filename*⟩ in repository ⟨*archive-ID*⟩. Does not check whether the file or the repository exist.

`\inputref`
`\inputref:nn`

`\inputref[⟨archive-ID⟩]{⟨filename⟩}`

`\input`s the file ⟨*filename*⟩ in repository ⟨*archive-ID*⟩.

`\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:
```

.

Chapter 4

sTeX-References

Code related to links and cross-references

4.1 Macros and Environments

Chapter 5

sTeX-Modules

Code related to Modules

5.1 Macros and Environments

`\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`,

`\l_stex_all_modules_seq` Stores full URIs for all modules currently in scope.

`\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_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/doc/stextest
Faking a repository:
Namespace 2:
http://mathhub.info/tests/Foo/Bar/test/stextest
```

.

5.1.1 The `module`-environment

`module` `\begin{module}[⟨options⟩]{⟨name⟩}`

Opens a new module with name ⟨*name*⟩.
TODO document options.

`\stex_module_setup:nn` `\stex_module_setup:nn{⟨params⟩}{⟨name⟩}`

Sets up a new module with name ⟨*name*⟩ and optional parameters ⟨*params*⟩. In particular, sets `\l_stex_current_module_prop` appropriately.

`\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:nn{modules}{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:nn{modules}{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 5.1.1[Bar] (FooBar)
Module path: http://mathhub.info/tests/Foo/Bar/Foo?Bar
Language:
Signature:
Metatheory:

.

`\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 5.1.2[STEXModuleTest1]

Module 5.1.3[STEXModuleTest2]

Module 5.1.4[STEXModuleTest3]
file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?STEXModuleTest1
file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?STEXModuleTest2
file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?STEXModuleTest3
foo1
foo2
foo3

.

`\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) and adds the module to `\l_stex_all_modules_seq`.

Chapter 6

sTeX-Module Inheritance

Code related to Module Inheritance, in particular *sms mode*.

6.1 Macros and Environments

6.1.1 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
```

.

6.1.2 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`.

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 6.1.1[Foo]
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?Foo?foo}«

Meaning: »macro:->\protect \bar «

Module 6.1.2[Importtest]
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?Foo?foo}«

Module 6.1.3[Importtest2]
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/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 6.1.4[UseTest1]

Module 6.1.5[UseTest2]
    Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?UseTest1?foo}«

Module 6.1.6[UseTest3]
    Meaning: »undefined«
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?UseTest2?bar}«

    All modules: http://mathhub.info/sTeX?Metatheory, file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?UseTest3,
file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?UseTest2
All symbols: http://mathhub.info/sTeX?Metatheory?isa, http://mathhub.info/sTeX?Metatheory?bind, http://mathhub.info/sTeX?Metatheo
http://mathhub.info/sTeX?Metatheory?fromto, http://mathhub.info/sTeX?Metatheory?apply, http://mathhub.info/sTeX?Metatheory?collec
http://mathhub.info/sTeX?Metatheory?seqtype, http://mathhub.info/sTeX?Metatheory?sequence-index, http://mathhub.info/sTeX?Metath
http://mathhub.info/sTeX?Metatheory?aseqfromto, http://mathhub.info/sTeX?Metatheory?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/doc/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 6.1.7[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.

Chapter 7

sTeX-Symbols

Code related to symbol declarations and notations

7.1 Macros and Environments

`\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 `i`s, `a`s and `b`s),
- `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 7.1.1[SymdeclTest]
      Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?SymdeclTest?foo}«
Result: file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?SymdeclTest?foo
Meaning: »macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/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...

`\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 7.1.2[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 7.1.3[SymdefTest]
$a+b+c$

.

Chapter 8

STEX-Terms

Code related to symbolic expressions, typesetting notations, notation components, etc.

8.1 Macros and Environments

`\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}`.

`\_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 8.1.1[MathTest1]
$\langle x20x20a^{b}{}_{c}\rangle$ and $\langle x20x20a^{b}{}_{c}\rangle$.

.

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 8.1.2[MathTest2]
$\langle x20x20a|[b{:}c_{:d_{:e_{:f}}}]^{g}\rangle$ and $\langle x20x20a|[b{:}c]^{g}\rangle$ and $\langle x20x20a|[b]^{c}\rangle$
$a+(b\cdot c)$ and $a\cdot\frac{a}{b}+\frac{a}{c}$

$$a+(b\cdot c) \text{ and } a\cdot\frac{a}{b}+\frac{a}{c}$$

$\displaystyle a+(b\cdot c)$ and $\displaystyle a\cdot\frac{a}{b}+\frac{a}{c}$

.

`\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 8.1.3[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`
`\compemph`
`\compemph@uri`
`\defemph`
`\defemph@uri`
`\symrefemph`
`\symrefemph@uri`

`\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

Chapter 9

sTeX-Structural Features

Code related to structural features

9.1 Macros and Environments

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).

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 9.1.1[StructureTest1]

$a\circ b$:M

file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?StructureTest1/Magma-feature?universe,file://home/jazzpirate/work/S
feature?op

```
»macro:->\stex_invoke_symbol:n {file://home/jazzpirate/work/Software/ext/sTeX/doc/stextest?StructureTest1?Magma}«
```

Test: $a+b$

Test2: $\langle U,+\rangle$

.

Chapter 10

sTeX-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.

10.1 Symbols

Part III
Extensions

Chapter 11

Tikzinput

11.1 Macros and Environments

LocalWords: bibfolder jobname.dtx tikzinput.dtx usetikzlibrary Gin@ewidth Gin@eheight
LocalWords: resizebox ctikzinput mhtikzinput Gin@mhrepos mhpath

Chapter 12

document-structure.sty: Semantic Markup for Open Mathematical Documents in LaTeX

The omdoc package is part of the sTeX collection, a version of TeX/LaTeX that allows to markup TeX/LaTeX documents semantically without leaving the document format, essentially turning TeX/LaTeX into a document format for mathematical knowledge management (MKM).

This package supplies an infrastructure for writing OMDoc documents in LaTeX. This includes a simple structure sharing mechanism for sTeX that allows to to move from a copy-and-paste document development model to a copy-and-reference model, which conserves space and simplifies document management. The augmented structure can be used by MKM systems for added-value services, either directly from the sTeX sources, or after translation.

12.1 Introduction

sTeX is a version of TeX/LaTeX that allows to markup TeX/LaTeX documents semantically without leaving the document format, essentially turning TeX/LaTeX into a document format for mathematical knowledge management (MKM). The package supports direct translation to the OMDoc format [Koh06]

The omdoc package supplies macros and environments that allow to label document fragments and to reference them later in the same document or in other documents. In essence, this enhances the document-as-trees model to documents-as-directed-acyclic-graphs (DAG) model. This structure can be used by MKM systems for added-value services, either directly from the sTeX sources, or after translation. Currently, trans-document referencing provided by this package can only be used in the sTeX collection.

DAG models of documents allow to replace the "Copy and Paste" in the source document with a label-and-reference model where document are shared in the document

EdN:4 source and the formatter does the copying during document formatting/presentation.[4]

12.2 The User Interface

The omdoc package generates two files: omdoc.cls, and omdoc.sty. The OMDoc class is a minimally changed variant of the standard article class that includes the functionality provided by omdoc.sty. The rest of the documentation pertains to the functionality introduced by omdoc.sty.

12.2.1 Package and Class Options

The omdoc class accept the following options:

class=⟨*name*⟩	load ⟨*name*⟩.cls instead of article.cls
topsect=⟨*sect*⟩	The top-level sectioning level; the default for ⟨*sect*⟩ is section
showignores	show the the contents of the ignore environment after all
showmeta	show the metadata; see metakeys.sty
showmods	show modules; see modules.sty
extrefs	allow external references; see sref.sty
defindex	index definienda; see statements.sty
minimal	for testing; do not load any sTeX packages

The omdoc package accepts the same except the first two.

12.2.2 Document Structure

document
\documentkeys
id

The top-level document environment can be given key/value information by the \documentkeys macro in the preamble[2]. This can be used to give metadata about the document. For the moment only the id key is used to give an identifier to the omdoc element resulting from the LaTeXML transformation.

omgroup

The structure of the document is given by the omgroup environment just like in OMDoc. In the LaTeX route, the omgroup environment is flexibly mapped to sectioning commands, inducing the proper sectioning level from the nesting of omgroup environments. Correspondingly, the omgroup environment takes an optional key/value argument for metadata followed by a regular argument for the (section) title of the omgroup. The optional metadata argument has the keys id for an identifier, creators and contributors for the Dublin Core metadata [DCM03]; see [Koh20a] for details of the format. The short allows to give a short title for the generated section. If the title contains semantic macros, they need to be protected by \protect, and we need to give the loadmodules key it needs no value. For instance we would have

id
creators
contributors
short
loadmodules

```
\begin{module}{foo}
\symdef{bar}{B^a_r}
 ...
\begin{omgroup}[id=sec.barderiv,loadmodules]{Introducing $\protect\bar$ Derivations}
```

sTeX automatically computes the sectioning level, from the nesting of omgroup environments. But sometimes, we want to skip levels (e.g. to use a subsection* as an introduction for a chapter). Therefore the omdoc package provides a variant blindomgroup

blindomgroup

[4] EdNote: integrate with latexml's XMRef in the Math mode.

[2] We cannot patch the document environment to accept an optional argument, since other packages we load already do; pity.

that does not produce markup, but increments the sectioning level and logically groups document parts that belong together, but where traditional document markup relies on convention rather than explicit markup. The `blindomgroup` environment is useful e.g. for creating frontmatter at the correct level. Example 1 shows a typical setup for the outer document structure of a book with parts and chapters. We use two levels of `blindomgroup`:

- The outer one groups the introductory parts of the book (which we assume to have a sectioning hierarchy topping at the part level). This `blindomgroup` makes sure that the introductory remarks become a “chapter” instead of a “part”.
- Th inner one groups the frontmatter[3] and makes the preface of the book a section-level construct. Note that here the `display=flow` on the `omgroup` environment prevents numbering as is traditional for prefaces.

```
\begin{document}
\begin{blindomgroup}
\begin{blindomgroup}
\begin{frontmatter}
\maketitle\newpage
\begin{omgroup}[display=flow]{Preface}
... <<preface>> ...
\end{omgroup}
\clearpage\setcounter{tocdepth}{4}\tableofcontents\clearpage
\end{frontmatter}
\end{blindomgroup}
... <<introductory remarks>> ...
\end{blindomgroup}
\begin{omgroup}{Introduction}
... <<intro>> ...
\end{omgroup}
... <<more chapters>> ...
\bibliographystyle{alpha}\bibliography{kwarc}
\end{document}
```

Example 1: A typical Document Structure of a Book

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

`\currentsectionlevel` `\CurrentSectionLevel` The `\currentsectionlevel` macro supplies the name of the current sectioning level, e.g. “chapter”, or “subsection”. `\CurrentSectionLevel` is the capitalized variant. They are useful to write something like “In this `\currentsectionlevel`, we will…” in an `omgroup` environment, where we do not know which sectioning level we will end up.

12.2.3 Ignoring Inputs

`ignore` `showignores` The `ignore` environment can be used for hiding text parts from the document structure. The body of the environment is not PDF or DVI output unless the `showignores` option

[3]We shied away from redefining the `frontmatter` to induce a blindomgroup, but this may be the “right” way to go in the future.

is given to the `omdoc` class or `package`. But in the generated OMDoc result, the body is marked up with a `ignore` element. This is useful in two situations. For

editing One may want to hide unfinished or obsolete parts of a document

narrative/content markup In sTeX we mark up narrative-structured documents. In the generated OMDoc documents we want to be able to cache content objects that are not directly visible. For instance in the `statements` package [Koh20d] we use the `\inlinedef` macro to mark up phrase-level definitions, which verbalize more formal definitions. The latter can be hidden by an ignore and referenced by the `verbalizes` key in `\inlinedef`.

`\prematurestop` `\afterprematurestop` For prematurely stopping the formatting of a document, sTeX provides the `\prematurestop` macro. It can be used everywhere in a document and ignores all input after that – backing out of the omgroup environment as needed. After that – and before the implicit `\end{document}` it calls the internal `\afterprematurestop`, which can be customized to do additional cleanup or e.g. print the bibliography.

`\prematurestop` is useful when one has a driver file, e.g. for a course taught multiple years and wants to generate course notes up to the current point in the lecture. Instead of commenting out the remaining parts, one can just move the `\prematurestop` macro. This is especially useful, if we need the rest of the file for processing, e.g. to generate a theory graph of the whole course with the already-covered parts marked up as an overview over the progress; see `import_graph.py` from the `lmhtools` utilities [LMH].

12.2.4 Structure Sharing

`\STRlabel` `\STRcopy` The `\STRlabel` macro takes two arguments: a label and the content and stores the the content for later use by `\STRcopy[`⟨*URL*⟩`]{`⟨*label*⟩`}`, which expands to the previously stored content. If the `\STRlabel` macro was in a different file, then we can give a URL ⟨*URL*⟩ that lets LaTeXML generate the correct reference.

`\STRsemantics` The `\STRlabel` macro has a variant `\STRsemantics`, where the label argument is optional, and which takes a third argument, which is ignored in LaTeX. This allows to specify the meaning of the content (whatever that may mean) in cases, where the source document is not formatted for presentation, but is transformed into some content markup format.[5]

EdN:5

12.2.5 Global Variables

Text fragments and modules can be made more re-usable by the use of global variables. For instance, the admin section of a course can be made course-independent (and therefore re-usable) by using variables (actually token registers) `courseAcronym` and `courseTitle` instead of the text itself. The variables can then be set in the sTeX preamble of the course

`\setSGvar` `\useSGvar` notes file. `\setSGvar{`⟨*vname*⟩`}{`⟨*text*⟩`}` to set the global variable ⟨*vname*⟩ to ⟨*text*⟩ and `\useSGvar{`⟨*vname*⟩`}` to reference it.

`\ifSGvar` With `\ifSGvar` we can test for the contents of a global variable: the macro call `\ifSGvar{`⟨*vname*⟩`}{`⟨*val*⟩`}{`⟨*ctext*⟩`}` tests the content of the global variable ⟨*vname*⟩, only if (after expansion) it is equal to ⟨*val*⟩, the conditional text ⟨*ctext*⟩ is formatted.

[5] EdNote: document LMID und LMXREf here if we decide to keep them.

12.2.6 Colors

`\blue` `\red` ... `\black`

For convenience, the `omdoc` package defines a couple of color macros for the `color` package: For instance `\blue` abbreviates `\textcolor{blue}`, so that `\blue{`⟨*something*⟩`}` writes ⟨*something*⟩ in blue. The macros `\red` `\green`, `\cyan`, `\magenta`, `\brown`, `\yellow`, `\orange`, `\gray`, and finally `\black` are analogous.

12.3 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the sTEX GitHub repository [sTeX].

1. when option `book` which uses `\pagestyle{headings}` is given and semantic macros are given in the `omgroup` titles, then they sometimes are not defined by the time the heading is formatted. Need to look into how the headings are made.

Chapter 13

Slides and Course Notes

We present a document class from which we can generate both course slides and course notes in a transparent way.

13.1 Introduction

The `mikoslides` document class is derived from `beamer.cls` [Tana], it adds a "notes version" for course notes derived from the `omdoc` class [**Kohlhase:smomdl**] that is more suited to printing than the one supplied by `beamer.cls`.

13.2 The User Interface

The `mikoslides` class takes the notion of a slide frame from Till Tantau's excellent `beamer` class and adapts its notion of frames for use in the sTeXand OMDoc. To support semantic course notes, it extends the notion of mixing frames and explanatory text, but rather than treating the frames as images (or integrating their contents into the flowing text), the `mikoslides` package displays the slides as such in the course notes to give students a visual anchor into the slide presentation in the course (and to distinguish the different writing styles in slides and course notes).

In practice we want to generate two documents from the same source: the slides for presentation in the lecture and the course notes as a narrative document for home study. To achieve this, the `mikoslides` class has two modes: *slides mode* and *notes mode* which are determined by the package option.

13.2.1 Package Options

EdN:6 The `mikoslides` class takes a variety of class options:[6]

`slides` `notes`
- The options `slides` and `notes` switch between slides mode and notes mode (see Section 13.2.2).

`sectocframes`
- If the option `sectocframes` is given, then for the `omgroups`, special frames with the `omgroup` title (and number) are generated.

showmeta

- showmeta. If this is set, then the metadata keys are shown (see [Koh20b] for details and customization options).

frameimages
fiboxed

- If the option frameimages is set, then slide mode also shows the \frameimage-generated frames (see section 13.2.4). If also the fiboxed option is given, the slides are surrounded by a box.

topsect

- topsect=⟨*sect*⟩ can be used to specify the top-level sectioning level; the default for ⟨*sect*⟩ is section.

13.2.2 Notes and Slides

frame
note

Slides are represented with the frame just like in the beamer class, see [Tanb] for details. The mikoslides class adds the note environment for encapsulating the course note fragments.[4]

⚠ Note that it is essential to start and end the notes environment at the start of the line – in particular, there may not be leading blanks – else LaTeX becomes confused and throws error messages that are difficult to decipher.

```
\ifnotes\maketitle\else
\frame[noframenumbering]\maketitle\fi

\begin{note}
  We start this course with ...
\end{note}

\begin{frame}
  \frametitle{The first slide}
  ...
\end{frame}
\begin{note}
  ... and more explanatory text
\end{note}

\begin{frame}
  \frametitle{The second slide}
  ...
\end{frame}
...
```

Example 2: A typical Course Notes File

By interleaving the frame and note environments, we can build course notes as shown in Figure 2.

\ifnotes

Note the use of the \ifnotes conditional, which allows different treatment between notes and slides mode – manually setting \notestrue or \notesfalse is strongly discouraged however.

[6]EdNote: leaving out noproblems for the moment until we decide what to do with it.
[4]MK: it would be very nice, if we did not need this environment, and this should be possible in principle, but not without intensive LaTeX trickery. Hints to the author are welcome.

⚠: We need to give the title frame the `noframenumbering` option so that the frame numbering is kept in sync between the slides and the course notes.

⚠: The `beamer` class recommends not to use the `allowframebreaks` option on frames (even though it is very convenient). This holds even more in the `mikoslides` case: At least in conjunction with `\newpage`, frame numbering behaves funnily (we have tried to fix this, but who knows).

`\inputref*` If we want to transclude a the contents of a file as a note, we can use a new variant `\inputref*` of the `\inputref` macro from [KGA20]: `\inputref*{foo}` is equivalent to `\begin{note}\inputref{foo}\end{note}`.

`nomtext` `nomgroup` `ndefinition` `nexample` `nsproof` `nassertion` There are some environments that tend to occur at the top-level of `note` environments. We make convenience versions of these: e.g. the `nomtext` environment is just an `omtext` inside a `note` environment (but looks nicer in the source, since it avoids one level of source indenting). Similarly, we have the `nomgroup`, `ndefinition`, `nexample`, `nsproof`, and `nassertion` environments.

13.2.3 Header and Footer Lines of the Slides

`\setslidelogo` The default logo provided by the `mikoslides` package is the STEX logo it can be customized using `\setslidelogo{⟨logo name⟩}`.

`\setsource` `\setlicensing` The default footer line of the `mikoslides` package mentions copyright and licensing. In the `beamer` class, `\source` stores the author's name as the copyright holder . By default it is *Michael Kohlhase* in the `mikoslides` package since he is the main user and designer of this package. `\setsource{⟨name⟩}` can change the writer's name. For licensing, we use the Creative Commons Attribuition-ShareAlike license by default to strengthen the public domain. If package `hyperref` is loaded, then we can attach a hyperlink to the license logo. `\setlicensing[⟨url⟩]{⟨logo name⟩}` is used for customization, where ⟨url⟩ is optional.

13.2.4 Frame Images

`\frameimage` Sometimes, we want to integrate slides as images after all – e.g. because we already have a PowerPoint presentation, to which we want to add STEXnotes. In this case we can use `\frameimage[⟨opt⟩]{⟨path⟩}`, where ⟨opt⟩ are the options of `\includegraphics` from the `graphicx` package [CR99] and ⟨path⟩ is the file path (extension can be left off like in `\includegraphics`). We have added the `label` key that allows to give a frame label that can be referenced like a regular `beamer` frame.[7] EdN:7

`\mhframeimage` The `\mhframeimage` macro is a variant of `\frameimage` with repository support. Instead of writing

```
\frameimage{\MathHub{fooMH/bar/source/baz/foobar}}
```

we can simply write (assuming that `\MathHub` is defined as above)

```
\mhframeimage[fooMH/bar]{baz/foobar}
```

Note that the `\mhframeimage` form is more semantic, which allows more advanced document management features in MathHub.

If `baz/foobar` is the "current module", i.e. if we are on the MathHub path ...`MathHub/fooMH/bar`..., then stating the repository in the first optional argument is redundant, so we can just use

[7] EdNote: MK: the hyperref link does not seem to work yet. I wonder why but do not have the time to fix it.

```
\mhframeimage{baz/foobar}
```

13.2.5 Colors and Highlighting

`\textwarning` The `\textwarning` macro generates a warning sign: ⚠

13.2.6 Front Matter, Titles, etc.

13.2.7 Excursions

In course notes, we sometimes want to point to an “excursion” – material that is either presupposed or tangential to the course at the moment – e.g. in an appendix. The typical setup is the following:

```
\excursion{founif}{../ex/founif}{We will cover first-order unification in}
...
\begin{appendix}\printexcursions\end{appendix}
```

`\excursion` `\activateexcursion` The `\excursion{⟨ref⟩}{⟨path⟩}{⟨text⟩}` is syntactic sugar for

```
\begin{nomtext}[title=Excursion]
  \activateexcursion{founif}{../ex/founif}
  We will cover first-order unification in \sref{founif}.
\end{nomtext}
```

`\activateexcursion` `\printexcursions` where `\activateexcursion{⟨path⟩}` augments the `\printexcursions` macro by a call `\inputref{⟨path⟩}`. In this way, the3 `\printexcursions` macro (usually in the appendix) will collect up all excursions that are specified in the main text.

Sometimes, we want to reference – in an excursion – part of another. We can use `\excursionref` `\excursionref{⟨label⟩}` for that.

Finally, we usually want to put the excursions into an `omgroup` environment and add an introduction, therefore we provide the a variant of the `\printexcursions` macro: `\excursiongroup` `\excursiongroup[id=⟨id⟩,intro=⟨path⟩]` is equivalent to

```
\begin{omgroup}[id=<id>]{Excursions}
  \inputref{<path>}
  \printexcursions
\end{omgroup}
```

13.2.8 Miscellaneous

13.3 Limitations

In this section we document known limitations. If you want to help alleviate them, please feel free to contact the package author. Some of them are currently discussed in the sTeXGitHub repository [sTeX].

1. when option `book` which uses `\pagestyle{headings}` is given and semantic macros are given in the `omgroup` titles, then they sometimes are not defined by the time the heading is formatted. Need to look into how the headings are made. This is a problem of the underlying `omdoc` package.

Part IV
Implementation

Chapter 14

sTeX -Basics Implementation

14.1 The sTeXDocument Class

The stex document class is pretty straight-forward: It largely extends the standalone package and loads the stex package, passing all provided options on to the package.

```
⟨*cls⟩

%%%%%%%%%%%%%   basics.dtx   %%%%%%%%%%%%%

\RequirePackage{expl3,l3keys2e}
\ProvidesExplClass{stex}{2021/08/01}{1.9}{bla}
\LoadClass[border=1px,varwidth]{standalone}
\setlength\textwidth{15cm}

\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{stex}}
\ProcessOptions

\RequirePackage{stex}
⟨/cls⟩
```

14.2 Preliminaries

```
⟨*package⟩

%%%%%%%%%%%%%   basics.dtx   %%%%%%%%%%%%%

\RequirePackage{expl3,l3keys2e,ltxcmds}
\ProvidesExplPackage{stex}{2021/08/01}{1.9}{bla}
\RequirePackage{morewrites}
```

Package options:

```
\keys_define:nn { stex } {
  debug     .clist_set:N  = \c_stex_debug_clist ,
  showmods  .bool_set:N   = \c_stex_showmods_bool ,
  lang      .clist_set:N  = \c_stex_languages_clist ,
  mathhub   .tl_set_x:N   = \mathhub ,
  sms       .bool_set:N   = \c_stex_persist_mode_bool ,
  image     .bool_set:N   = \c_tikzinput_image_bool
}
\ProcessKeysOptions { stex }
```

\stex
\sTeX

The sTEX logo:

```
\protected\def\stex{%
  \@ifundefined{texorpdfstring}%
  {\let\texorpdfstring\@firstoftwo}%
  {}%
  \texorpdfstring{\raisebox{-.5ex}S\kern-.5ex\TeX}{sTeX}\xspace%
}
\def\sTeX{\stex}
```

(*End definition for* `\stex` *and* `\sTeX`*. These functions are documented on page 9.*)

Patching expl3, if outdated:

```
⟨@@=keys⟩
\cs_if_exist:cF { \c__keys_props_root_str .str_set:N }{
  \cs_new_protected:cpn { \c__keys_props_root_str .str_set:N } #1
  { \__keys_variable_set:NnnN #1 { str } { } n }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_set:c } #1
  { \__keys_variable_set:cnnN {#1} { str } { } n }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_set_x:N } #1
  { \__keys_variable_set:NnnN #1 { str } { } x }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_set_x:c } #1
  { \__keys_variable_set:cnnN {#1} { str } { } x }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_gset:N } #1
  { \__keys_variable_set:NnnN #1 { str } { g } n }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_gset:c } #1
  { \__keys_variable_set:cnnN {#1} { str } { g } n }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_gset_x:N } #1
  { \__keys_variable_set:NnnN #1 { str } { g } x }
  \cs_new_protected:cpn { \c__keys_props_root_str .str_gset_x:c } #1
  { \__keys_variable_set:cnnN {#1} { str } { g } x }
}
```

14.3 Messages and logging

```
⟨@@=stex_log⟩
```

Warnings and error messages

```
\msg_new:nnn{stex}{error/unknownlanguage}{
  Unknown~language:~#1
}
\msg_new:nnn{stex}{warning/nomathhub}{
  MATHHUB~system~variable~not~found~and~no~
  \detokenize{\mathhub}-value~set!
}
\msg_new:nnn{stex}{error/deactivated-macro}{
  The~\detokenize{#1}~command~is~only~allowed~in~#2!
}
```

\stex_debug:nn A simple macro issuing package messages with subpath.

```
\cs_new_protected:Nn \stex_debug:nn {
  \clist_if_in:NnTF \c_stex_debug_clist { all } {
    \exp_args:Nnnx\msg_set:nnn{stex}{debug / #1}{
      \\Debug~#1:~#2\\
    }
    \msg_none:nn{stex}{debug / #1}
  }{
    \clist_if_in:NnT \c_stex_debug_clist { #1 } {
      \exp_args:Nnnx\msg_set:nnn{stex}{debug / #1}{
        \\Debug~#1:~#2\\
      }
      \msg_none:nn{stex}{debug / #1}
    }
  }
}
```

(*End definition for* `\stex_debug:nn`*. This function is documented on page 9.*)

Redirecting messages:

```
\clist_if_in:NnTF \c_stex_debug_clist {all} {
    \msg_redirect_module:nnn{ stex }{ none }{ term }
}{
  \clist_map_inline:Nn \c_stex_debug_clist {
    \msg_redirect_name:nnn{ stex }{ debug / ##1 }{ term }
  }
}

\stex_debug:nn{log}{debug~mode~on}
```

14.4 Persistence

```
⟨@@=stex_persist⟩
```

`\c__stex_persist_sms_iow` File variable used for the sms-File

```
\iow_new:N \c__stex_persist_sms_iow
\AddToHook{begindocument}{
  \bool_if:NTF \c_stex_persist_mode_bool {
    \ExplSyntaxOn \input{\jobname.sms} \ExplSyntaxOff
  } {
    \iow_open:Nn \c__stex_persist_sms_iow {\jobname.sms}
  }
}
\AddToHook{enddocument}{
  \bool_if:NF \c_stex_persist_mode_bool {
    \iow_close:N \c__stex_persist_sms_iow
  }
}
```

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

`\stex_add_to_sms:n` Adds the provided code to the `.sms`-file of the document.

```
\cs_new_protected:Nn \stex_add_to_sms:n {
  \bool_if:NF \c_stex_persist_mode_bool {
    \iow_now:Nn \c__stex_persist_sms_iow { #1 }
  }
}
```

(*End definition for* `\stex_add_to_sms:n`*. This function is documented on page 9.*)

14.5 HTML Annotations

```
⟨@@=stex_annotate⟩
\RequirePackage{scalatex}
```

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

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

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

Conditionals for LATEXML:

```
\ifcsname if@latexml\endcsname\else
    \expandafter\newif\csname if@latexml\endcsname\@latexmlfalse
\fi

\prg_new_conditional:Nnn \latexml_if: {p, T, F, TF} {
  \if@latexml
    \prg_return_true:
  \else:
    \prg_return_false:
  \fi:
}
```

(*End definition for* `\if@latexml` *and* `\latexml_if:TF`*. These functions are documented on page 9.*)

`\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.

```
\tl_new:N \l__stex_annotate_arg_tl
\tl_const:Nx \c__stex_annotate_emptyarg_tl {
  \scalatex_if:TF {
    \scalatex_direct_HTML:n { \c_ampersand_str lrm; }
  }{~}
}
```

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

`\__stex_annotate_checkempty:n`

```
\cs_new_protected:Nn \__stex_annotate_checkempty:n {
  \tl_set:Nn \l__stex_annotate_arg_tl { #1 }
  \tl_if_empty:NT \l__stex_annotate_arg_tl {
    \tl_set_eq:NN \l__stex_annotate_arg_tl \c__stex_annotate_emptyarg_tl
  }
}
```

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

`\l_stex_html_do_output_bool`
`\stex_if_do_html:`

Whether to (locally) produce HTML output

```
\bool_new:N \l_stex_html_do_output_bool
\bool_set_true:N \l_stex_html_do_output_bool
\prg_new_conditional:Nnn \stex_if_do_html: {p,T,F,TF} {
  \bool_if:nTF \l_stex_html_do_output_bool
    \prg_return_true: \prg_return_false:
}
```

(*End definition for* `\l_stex_html_do_output_bool` *and* `\stex_if_do_html:`. *These functions are documented on page* **??**.)

`\stex_suppress_html:n`

Whether to (locally) produce HTML output

```
\cs_new_protected:Nn \stex_suppress_html:n {
  \exp_args:Nne \use:nn {
    \bool_set_false:N \l_stex_html_do_output_bool
    #1
  }{
    \stex_if_do_html:T {
      \bool_set_true:N \l_stex_html_do_output_bool
    }
  }
}
```

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

`\stex_annotate:nnn`
`\stex_annotate_env`
`\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, SᴄAʟATEX, `pdflatex`).

The `pdflatex`-macros largely do nothing; the SᴄAʟATEX-implementations are pretty clear in what they do, the LaTeXML-implementations resort to perl bindings.

```
\scalatex_if:TF{
  \cs_new_protected:Nn \stex_annotate:nnn {
    \__stex_annotate_checkempty:n { #3 }
    \scalatex_annotate_HTML:nn {
      property="stex:#1" ~
      resource="#2"
    } {
      \tl_use:N \l__stex_annotate_arg_tl
    }
  }
  \cs_new_protected:Nn \stex_annotate_invisible:n {
    \__stex_annotate_checkempty:n { #1 }
    \scalatex_annotate_HTML:nn {
      stex:visible="false" ~
      style:display="none"
    } {
      \tl_use:N \l__stex_annotate_arg_tl
    }
  }
  \cs_new_protected:Nn \stex_annotate_invisible:nnn {
    \__stex_annotate_checkempty:n { #3 }
    \scalatex_annotate_HTML:nn {
      property="stex:#1" ~
      resource="#2" ~
      stex:visible="false" ~
      style:display="none"
    } {
      \tl_use:N \l__stex_annotate_arg_tl
    }
  }
  \NewDocumentEnvironment{stex_annotate_env} { m m } {
    \par
    \scalatex_annotate_HTML_begin:n {
      property="stex:#1" ~
      resource="#2"
    }
  }{
    \scalatex_annotate_HTML_end:
  }
}{
  \latexml_if:TF {
    \cs_new_protected:Nn \stex_annotate:nnn {
      \__stex_annotate_checkempty:n { #3 }
      \mode_if_math:TF {
        \cs:w latexml@annotate@math\cs_end:{#1}{#2}{
          \tl_use:N \l__stex_annotate_arg_tl
        }
      }{
        \cs:w latexml@annotate@text\cs_end:{#1}{#2}{
          \tl_use:N \l__stex_annotate_arg_tl
        }
      }
    }
    \cs_new_protected:Nn \stex_annotate_invisible:n {
      \__stex_annotate_checkempty:n { #1 }
      \mode_if_math:TF {
        \cs:w latexml@invisible@math\cs_end:{
          \tl_use:N \l__stex_annotate_arg_tl
        }
      } {
        \cs:w latexml@invisible@text\cs_end:{
          \tl_use:N \l__stex_annotate_arg_tl
        }
      }
    }
    \cs_new_protected:Nn \stex_annotate_invisible:nnn {
      \__stex_annotate_checkempty:n { #3 }
      \cs:w latexml@annotate@invisible\cs_end:{#1}{#2}{
        \tl_use:N \l__stex_annotate_arg_tl
      }
    }
    \NewDocumentEnvironment{stex_annotate_env} { m m } {
      \par\begin{latexml@annotateenv}{#1}{#2}
    }{
      \end{latexml@annotateenv}
    }
  }{
    \cs_new_protected:Nn \stex_annotate:nnn {#3}
    \cs_new_protected:Nn \stex_annotate_invisible:n {}
    \cs_new_protected:Nn \stex_annotate_invisible:nnn {}
    \NewDocumentEnvironment{stex_annotate_env} { m m } {\par}{}
  }
}
```

(*End definition for* `\stex_annotate:nnn`*,* `\stex_annotate_invisible:n`*, and* `\stex_annotate_invisible:nnn`*. These functions are documented on page 10.*)

14.6 Languages

```
⟨@@=stex_language⟩
```

\c_stex_languages_prop
\c_stex_language_abbrevs_prop

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

```
\prop_const_from_keyval:Nn \c_stex_languages_prop {
  en = english ,
  de = ngerman ,
  ar = arabic ,
  bg = bulgarian ,
  ru = russian ,
  fi = finnish ,
  ro = romanian ,
  tr = turkish ,
  fr = french
}

\prop_const_from_keyval:Nn \c_stex_language_abbrevs_prop {
  english   = en ,
  ngerman   = de ,
  arabic    = ar ,
  bulgarian = bg ,
  russian   = ru ,
  finnish   = fi ,
  romanian  = ro ,
  turkish   = tr ,
  french    = fr
}
% todo: chinese simplified (zhs)
%       chinese traditional (zht)
```

(*End definition for* `\c_stex_languages_prop` *and* `\c_stex_language_abbrevs_prop`*. These variables are documented on page 10.*)

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

```
\clist_if_empty:NF \c_stex_languages_clist {
  \clist_clear:N \l_tmpa_clist
  \clist_map_inline:Nn \c_stex_languages_clist {
    \prop_get:NnNTF \c_stex_languages_prop { #1 } \l_tmpa_str {
      \clist_put_right:No \l_tmpa_clist \l_tmpa_str
    } {
      \msg_error:nnx{stex}{error/unknownlanguage}{\l_tmpa_str}
    }
  }
  \stex_debug:nn{lang} {Languages:~\clist_use:Nn \l_tmpa_clist {,~} }
  \RequirePackage[\clist_use:Nn \l_tmpa_clist ,]{babel}
}
```

14.7 Activating/Deactivating Macros

\stex_deactivate_macro:Nn

```
\cs_new_protected:Nn \stex_deactivate_macro:Nn {
  \exp_after:wN\let\csname \detokenize{#1} - orig\endcsname#1
  \def#1{
    \msg_error:nnnn{stex}{error/deactivated-macro}{#1}{#2}
  }
}
```

(*End definition for* `\stex_deactivate_macro:Nn`*. This function is documented on page 10.*)

\stex_reactivate_macro:N

```
\cs_new_protected:Nn \stex_reactivate_macro:N {
  \exp_after:wN\let\exp_after:wN#1\csname \detokenize{#1} - orig\endcsname
}
```

(*End definition for* `\stex_reactivate_macro:N`*. This function is documented on page 10.*)

```
⟨/package⟩
```

Chapter 15

sTEX -MathHub Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   mathhub.dtx   %%%%%%%%%%%%%

⟨@@=stex_path⟩
```

Warnings and error messages

```
\msg_new:nnn{stex}{error/norepository}{
  No~archive~#1~found~in~#2
}
\msg_new:nnn{stex}{error/notinarchive}{
  Not~currently~in~an~archive,~but~\detokenize{#1}~
  needs~one!
}
\msg_new:nnn{stex}{error/nofile}{
  \detokenize{#1}~could~not~find~file~#2
}
```

15.1 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

```
\cs_new_protected:Nn \stex_path_from_string:Nn {
  \str_set:Nx \l_tmpa_str { #2 }
  \str_if_empty:NTF \l_tmpa_str {
    \seq_clear:N #1
  }{
    \exp_args:NNNo \seq_set_split:Nnn #1 / { \l_tmpa_str }
    \sys_if_platform_windows:T{
      \seq_clear:N \l_tmpa_tl
      \seq_map_inline:Nn #1 {
        \seq_set_split:Nnn \l_tmpb_tl \c_backslash_str { ##1 }
        \seq_concat:NNN \l_tmpa_tl \l_tmpa_tl \l_tmpb_tl
      }
      \seq_set_eq:NN #1 \l_tmpa_tl
    }
    \stex_path_canonicalize:N #1
  }
}
\cs_generate_variant:Nn \stex_path_from_string:Nn
  { NV, cn, cV }
```

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

`\stex_path_to_string:NN`
`\stex_path_to_string:N`

```
\cs_new_protected:Nn \stex_path_to_string:NN {
  \exp_args:NNe \str_set:Nn #2 { \seq_use:Nn #1 / }
}

\cs_new:Nn \stex_path_to_string:N {
  \seq_use:Nn #1 /
}
```

(*End definition for* `\stex_path_to_string:NN` *and* `\stex_path_to_string:N`*. These functions are documented on page 11.*)

`\c__stex_path_dot_str`
`\c__stex_path_up_str`

`.` and `..`, respectively.

```
\str_const:Nn \c__stex_path_dot_str {.}
\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.

```
\cs_new_protected:Nn \stex_path_canonicalize:N {
  \seq_if_empty:NF #1 {
    \seq_clear:N \l_tmpa_seq
    \seq_get_left:NN #1 \l_tmpa_tl
    \str_if_empty:NT \l_tmpa_tl {
      \seq_put_right:Nn \l_tmpa_seq {}
    }
    \seq_map_inline:Nn #1 {
      \str_set:Nn \l_tmpa_tl { ##1 }
      \str_if_eq:NNTF \l_tmpa_tl \c__stex_path_dot_str {} {
        \str_if_eq:NNTF \l_tmpa_tl \c__stex_path_up_str {
          \seq_if_empty:NTF \l_tmpa_seq {
            \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq {
              \c__stex_path_up_str
            }
          }{
            \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
            \str_if_eq:NNTF \l_tmpa_tl \c__stex_path_up_str {
              \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq {
                \c__stex_path_up_str
              }
            }{
              \seq_pop_right:NN \l_tmpa_seq \l_tmpb_tl
            }
          }
        }{
          \str_if_empty:NF \l_tmpa_tl {
            \exp_args:NNo \seq_put_right:Nn \l_tmpa_seq { \l_tmpa_tl }
          }
        }
      }
    }
    \seq_gset_eq:NN #1 \l_tmpa_seq
  }
}
```

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

\stex_path_if_absolute_p:N
\stex_path_if_absolute:N*TF*

```
\prg_new_conditional:Nnn \stex_path_if_absolute:N {p, T, F, TF} {
  \seq_if_empty:NTF #1 {
    \prg_return_false:
  }{
    \seq_get_left:NN #1 \l_tmpa_tl
    \str_if_empty:NTF \l_tmpa_tl {
      \prg_return_true:
    }{
      \prg_return_false:
    }
  }
}
```

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

15.2 PWD and kpsewhich

\stex_kpsewhich:n

```
\str_new:N\l_stex_kpsewhich_return_str
\cs_new_protected:Nn \stex_kpsewhich:n {
  \sys_get_shell:nnN { kpsewhich ~ #1 } { } \l_tmpa_tl
  \exp_args:NNo\str_set:Nn\l_stex_kpsewhich_return_str{\l_tmpa_tl}
  \tl_trim_spaces:N \l_stex_kpsewhich_return_str
}
```

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

We determine the PWD

\c_stex_pwd_seq
\c_stex_pwd_str

```
\sys_if_platform_windows:TF{
  \stex_kpsewhich:n{-expand-var~\c_percent_str CD\c_percent_str}
}{
  \stex_kpsewhich:n{-var-value~PWD}
}

\stex_path_from_string:Nn\c_stex_pwd_seq\l_stex_kpsewhich_return_str
\stex_path_to_string:NN\c_stex_pwd_seq\c_stex_pwd_str
\stex_debug:nn {mathhub} {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 11.*)

15.3 File Hooks and Tracking

```
⟨@@=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.

`\g__stex_files_stack` keeps track of file changes

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

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

`\c_stex_mainfile_seq`
`\c_stex_mainfile_str`

```
\str_set:Nx \c_stex_mainfile_str {\c_stex_pwd_str/\jobname.tex}
\stex_path_from_string:Nn \c_stex_mainfile_seq
  \c_stex_mainfile_str
```

(*End definition for* `\c_stex_mainfile_seq` *and* `\c_stex_mainfile_str`*. These variables are documented on page 11.*)

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

```
\seq_gclear_new:N\g_stex_currentfile_seq
\AddToHook{file/before}{
  \stex_path_from_string:Nn\g_stex_currentfile_seq{\CurrentFilePath}
  \stex_path_if_absolute:NTF\g_stex_currentfile_seq{
    \exp_args:NNe\seq_put_right:Nn\g_stex_currentfile_seq{\CurrentFile}
  }{
    \stex_path_from_string:Nn\g_stex_currentfile_seq{
      \c_stex_pwd_str/\CurrentFilePath/\CurrentFile
    }
  }
  \seq_gset_eq:NN\g_stex_currentfile_seq\g_stex_currentfile_seq
  \exp_args:NNo\seq_gpush:Nn\g__stex_files_stack\g_stex_currentfile_seq
}
\AddToHook{file/after}{
  \seq_if_empty:NF\g__stex_files_stack{
    \seq_gpop:NN\g__stex_files_stack\l_tmpa_seq
  }
  \seq_if_empty:NTF\g__stex_files_stack{
    \seq_gset_eq:NN\g_stex_currentfile_seq\c_stex_mainfile_seq
  }{
    \seq_get:NN\g__stex_files_stack\l_tmpa_seq
    \seq_gset_eq:NN\g_stex_currentfile_seq\l_tmpa_seq
  }
}
```

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

15.4 MathHub Repositories

```
⟨@@=stex_mathhub⟩
```

\mathhub
\c_stex_mathhub_seq
\c_stex_mathhub_str

```
\str_if_empty:NTF\mathhub{
  \stex_kpsewhich:n{-var-value~MATHHUB}
  \str_set_eq:NN\c_stex_mathhub_str\l_stex_kpsewhich_return_str

  \str_if_empty:NTF\c_stex_mathhub_str{
    \msg_warning:nn{stex}{warning/nomathhub}
  }{
    \stex_debug:nn{mathhub} {MathHub:~\str_use:N\c_stex_mathhub_str}
    \exp_args:NNo \stex_path_from_string:Nn\c_stex_mathhub_seq\c_stex_mathhub_str
  }
}{
  \stex_path_from_string:Nn \c_stex_mathhub_seq \mathhub
  \stex_path_if_absolute:NF \c_stex_mathhub_seq {
    \exp_args:NNx \stex_path_from_string:Nn \c_stex_mathhub_seq {
      \c_stex_pwd_str/\mathhub
    }
  }
  \stex_path_to_string:NN\c_stex_mathhub_seq\c_stex_mathhub_str
  \stex_debug:nn{mathhub} {MathHub:~\str_use:N\c_stex_mathhub_str}
}
```

(*End definition for* \mathhub*,* \c_stex_mathhub_seq*, and* \c_stex_mathhub_str*. These variables are documented on page 12.*)

\__stex_mathhub_do_manifest:n

```
\cs_new_protected:Nn \__stex_mathhub_do_manifest:n {
  \str_set:Nx \l_tmpa_str { #1 }
  \prop_if_exist:cF {c_stex_mathhub_#1_manifest_prop} {
    \prop_new:c { c_stex_mathhub_#1_manifest_prop }
    \seq_set_split:NnV \l_tmpa_seq / \l_tmpa_str
    \seq_concat:NNN \l_tmpa_seq \c_stex_mathhub_seq \l_tmpa_seq
    \__stex_mathhub_find_manifest:N \l_tmpa_seq
    \seq_if_empty:NTF \l__stex_mathhub_manifest_file_seq {
      \msg_error:nnnn{stex}{error/norepository}{#1}{
        \stex_path_to_string:N \c_stex_mathhub_str
      }
    } {
      \exp_args:No \__stex_mathhub_parse_manifest:n { \l_tmpa_str }
    }
  }
}
```

(*End definition for* \__stex_mathhub_do_manifest:n*.*)

\l__stex_mathhub_manifest_file_seq

```
\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`:

```
\cs_new_protected:Nn \__stex_mathhub_find_manifest:N {
  \seq_set_eq:NN\l_tmpa_seq #1
  \bool_set_true:N\l_tmpa_bool
  \bool_while_do:Nn \l_tmpa_bool {
    \seq_if_empty:NTF \l_tmpa_seq {
      \bool_set_false:N\l_tmpa_bool
    }{
      \file_if_exist:nTF{
        \stex_path_to_string:N\l_tmpa_seq/MANIFEST.MF
      }{
        \seq_put_right:Nn\l_tmpa_seq{MANIFEST.MF}
        \bool_set_false:N\l_tmpa_bool
      }{
        \file_if_exist:nTF{
          \stex_path_to_string:N\l_tmpa_seq/META-INF/MANIFEST.MF
        }{
          \seq_put_right:Nn\l_tmpa_seq{META-INF}
          \seq_put_right:Nn\l_tmpa_seq{MANIFEST.MF}
          \bool_set_false:N\l_tmpa_bool
        }{
          \file_if_exist:nTF{
            \stex_path_to_string:N\l_tmpa_seq/meta-inf/MANIFEST.MF
          }{
            \seq_put_right:Nn\l_tmpa_seq{meta-inf}
            \seq_put_right:Nn\l_tmpa_seq{MANIFEST.MF}
            \bool_set_false:N\l_tmpa_bool
          }{
            \seq_pop_right:NN\l_tmpa_seq\l_tmpa_tl
          }
        }
      }
    }
  }
  \seq_set_eq:NN\l__stex_mathhub_manifest_file_seq\l_tmpa_seq
}
```

(*End definition for* `\__stex_mathhub_find_manifest:N`.)

\c__stex_mathhub_manifest_ior

File variable used for `MANIFEST`-files

```
\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:

```
\cs_new_protected:Nn \__stex_mathhub_parse_manifest:n {
  \seq_set_eq:NN \l_tmpa_seq \l__stex_mathhub_manifest_file_seq
  \ior_open:Nn \c__stex_mathhub_manifest_ior {\stex_path_to_string:N \l_tmpa_seq}
  \ior_map_inline:Nn \c__stex_mathhub_manifest_ior {
    \str_set:Nn \l_tmpa_str {##1}
    \exp_args:NNoo \seq_set_split:Nnn
        \l_tmpb_seq \c_colon_str \l_tmpa_str
    \seq_pop_left:NNTF \l_tmpb_seq \l_tmpa_tl {
      \exp_args:NNe \str_set:Nn \l_tmpb_tl {
        \exp_args:NNo \seq_use:Nn \l_tmpb_seq \c_colon_str
      }
      \exp_args:No \str_case:nnTF \l_tmpa_tl {
        {id} {
          \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
            { id } \l_tmpb_tl
        }
        {narration-base} {
          \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
            { narr } \l_tmpb_tl
        }
        {url-base} {
          \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
            { docurl } \l_tmpb_tl
        }
        {source-base} {
          \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
            { ns } \l_tmpb_tl
        }
        {ns} {
          \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
            { ns } \l_tmpb_tl
        }
        {dependencies} {
          \prop_gput:cno { c_stex_mathhub_#1_manifest_prop }
            { deps } \l_tmpb_tl
        }
      }{}{}
    }{}
  }
  \ior_close:N \c__stex_mathhub_manifest_ior
}
```

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

\stex_set_current_repository:n

```
\cs_new_protected:Nn \stex_set_current_repository:n {
  \stex_require_repository:n { #1 }
  \prop_set_eq:Nc \l_stex_current_repository_prop {
    c_stex_mathhub_#1_manifest_prop
  }
}
```

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

\stex_require_repository:n

```
\cs_new_protected:Nn \stex_require_repository:n {
  \prop_if_exist:cF { c_stex_mathhub_#1_manifest_prop } {
    \stex_debug:nn{mathhub}{Opening~archive:~#1}
    \__stex_mathhub_do_manifest:n { #1 }
    \exp_args:Nx \stex_add_to_sms:n {
      \prop_const_from_keyval:cn { c_stex_mathhub_#1_manifest_prop } {
        id   = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } {  id  } ,
        ns   = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } {  ns  } ,
        narr = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } { narr } ,
        deps = \prop_item:cn { c_stex_mathhub_#1_manifest_prop } { deps }
      }
    }
  }
}
```

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

\l_stex_current_repository_prop

Current MathHub repository

```
\prop_new:N \l_stex_current_repository_prop

\__stex_mathhub_find_manifest:N \c_stex_pwd_seq
\seq_if_empty:NTF \l__stex_mathhub_manifest_file_seq {
  \stex_debug:nn{mathhub}{Not~currently~in~a~MathHub~repository}
} {
  \__stex_mathhub_parse_manifest:n { main }
  \prop_get:NnN \c_stex_mathhub_main_manifest_prop {id}
    \l_tmpa_str
  \prop_set_eq:cN { c_stex_mathhub_\l_tmpa_str _manifest_prop }
    \c_stex_mathhub_main_manifest_prop
  \exp_args:Nx \stex_set_current_repository:n { \l_tmpa_str }
  \stex_debug:nn{mathhub}{Current~repository:~
    \prop_item:Nn \l_stex_current_repository_prop {id}
  }
}
```

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

\stex_in_repository:nn

Executes the code in the second argument in the context of the repository whose ID is provided as the first argument.

```
\cs_new_protected:Nn \stex_in_repository:nn {
  \str_set:Nx \l_tmpa_str { #1 }
  \cs_set:Npn \l_tmpa_cs ##1 { #2 }
  \str_if_empty:NTF \l_tmpa_str {
    \exp_args:Ne \l_tmpa_cs{
      \prop_item:Nn \l_stex_current_repository_prop { id }
    }
  }{
    \stex_require_repository:n \l_tmpa_str
    \str_set:Nx \l_tmpa_str { #1 }
    \exp_args:Nne \use:nn {
      \stex_set_current_repository:n \l_tmpa_str
      \exp_args:Nx \l_tmpa_cs{\l_tmpa_str}
    }{
       \stex_set_current_repository:n {
        \prop_item:Nn \l_stex_current_repository_prop { id }
       }
    }
  }
}
```

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

\inputref
\inputref:nn

```
\newif \ifinputref \inputreffalse

\cs_new_protected:Nn \inputref:nn {
  \stex_in_repository:nn {#1} {
    \ifinputref
      \input{ \c_stex_mathhub_str / ##1 / source / #2 }
    \else
      \inputreftrue
      \input{ \c_stex_mathhub_str / ##1 / source / #2 }
      \inputreffalse
    \fi
  }
}
\NewDocumentCommand \inputref { O{} m}{
  \inputref:nn{ #1 }{ #2 }
}
```

(*End definition for* `\inputref` *and* `\inputref:nn`*. These functions are documented on page 13.*)

\mhpath

```
  \def \mhpath #1 #2 {
    \exp_args:Ne \str_if_eq:nnTF{#1}{}{
      \c_stex_mathhub_str /
        \prop_item:Nn \l_stex_current_repository_prop { id }
        / source / #2
    }{
      \c_stex_mathhub_str / #1 / source / #2
    }
  }
```

(*End definition for* `\mhpath`*. This function is documented on page 13.*)

\libinput

```
\cs_new_protected:Npn \libinput #1 {
  \prop_get:NnNF \l_stex_current_repository_prop {id} \l_tmpa_str {
    \msg_error:nnn{stex}{error/notinarchive}\libinput
  }
  \bool_set_false:N \l_tmpa_bool
  \tl_clear:N \l_tmpa_tl
  \seq_set_eq:NN \l_tmpa_seq \c_stex_mathhub_seq
  \seq_set_split:NnV \l_tmpb_seq / \l_tmpa_str
  \seq_pop_right:NN \l_tmpb_seq \l_tmpa_str
  \seq_pop_left:NNT \l_tmpb_seq \l_tmpb_str {
    \seq_put_right:No \l_tmpa_seq \l_tmpb_str
    \IfFileExists{ \stex_path_to_string:N \l_tmpa_seq
      / meta-inf / lib / #1.tex}{
        \bool_set_true:N \l_tmpa_bool
        \tl_put_right:Nx \l_tmpa_tl {
          \exp_not:N \input { \stex_path_to_string:N \l_tmpa_seq
          / meta-inf / lib / #1.tex}
        }
      }{}
  }
  \IfFileExists{ \stex_path_to_string:N \l_tmpa_seq
    / \l_tmpa_str / lib / #1.tex
  }{
    \bool_set_true:N \l_tmpa_bool
    \tl_put_right:Nx \l_tmpa_tl {
      \exp_not:N \input { \stex_path_to_string:N \l_tmpa_seq
      / \l_tmpa_str / lib / #1.tex}
    }
  }{}
  \bool_if:NF \l_tmpa_bool {
    \msg_error:nnnn{stex}{error/nofile}\libinput{#1.tex}
  }
  \l_tmpa_tl
}
```

(*End definition for* `\libinput`*. This function is documented on page 13.*)

```
⟨/package⟩
```

Chapter 16

sTeX -References Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   references.dtx   %%%%%%%%%%%%%

%\RequirePackage{hyperref}
%\RequirePackage{cleveref}
⟨@@=stex_refs⟩
```

Warnings and error messages

```

\iow_new:N \c__stex_refs_refs_iow
\AddToHook{begindocument}{
  \iow_open:Nn \c__stex_refs_refs_iow {\jobname.sref}
}
\AddToHook{enddocument}{
  \iow_close:N \c__stex_refs_refs_iow
}

\str_set:Nn \g__stex_refs_title_tl {Unnamed~Document}

\NewDocumentCommand \STEXreftitle { m } {
  \tl_gset:Nx \g__stex_refs_title_tl { #1 }
}
```

16.1 Document URIs and URLs

```
\seq_new:N \g__stex_refs_all_refs_seq

\str_new:N \l_stex_current_docns_str

\cs_new_protected:Nn \stex_get_document_uri: {
  \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
  \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
  \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
  \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
  \seq_put_right:No \l_tmpa_seq \l_tmpb_str

  \str_clear:N \l_tmpa_str
  \prop_get:NnNF \l_stex_current_repository_prop { narr } \l_tmpa_str {
    \prop_get:NnNF \l_stex_current_repository_prop { ns } \l_tmpa_str {}
  }

  \str_if_empty:NTF \l_tmpa_str {
    \str_set:Nx \l_stex_current_docns_str {
      file:/\stex_path_to_string:N \l_tmpa_seq
    }
  }{
    \bool_set_true:N \l_tmpa_bool
    \bool_while_do:Nn \l_tmpa_bool {
      \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
      \exp_args:No \str_case:nnTF { \l_tmpb_str } {
        {source} { \bool_set_false:N \l_tmpa_bool }
      }{}{
        \seq_if_empty:NT \l_tmpa_seq {
          \bool_set_false:N \l_tmpa_bool
        }
      }
    }

    \seq_if_empty:NTF \l_tmpa_seq {
      \str_set_eq:NN \l_stex_current_docns_str \l_tmpa_str
    }{
      \str_set:Nx \l_stex_current_docns_str {
        \l_tmpa_str/\stex_path_to_string:N \l_tmpa_seq
      }
    }
  }
}
\str_new:N \l_stex_current_docurl_str
\cs_new_protected:Nn \stex_get_document_url: {
  \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
  \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
  \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
  \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
  \seq_put_right:No \l_tmpa_seq \l_tmpb_str

  \str_clear:N \l_tmpa_str
  \prop_get:NnNF \l_stex_current_repository_prop { docurl } \l_tmpa_str {
    \prop_get:NnNF \l_stex_current_repository_prop { narr } \l_tmpa_str {
      \prop_get:NnNF \l_stex_current_repository_prop { ns } \l_tmpa_str {}
    }
  }

  \str_if_empty:NTF \l_tmpa_str {
    \str_set:Nx \l_stex_current_docurl_str {
      file:/\stex_path_to_string:N \l_tmpa_seq
    }
  }{
    \bool_set_true:N \l_tmpa_bool
    \bool_while_do:Nn \l_tmpa_bool {
      \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
      \exp_args:No \str_case:nnTF { \l_tmpb_str } {
        {source} { \bool_set_false:N \l_tmpa_bool }
      }{}{
        \seq_if_empty:NT \l_tmpa_seq {
          \bool_set_false:N \l_tmpa_bool
        }
      }
    }

    \seq_if_empty:NTF \l_tmpa_seq {
      \str_set_eq:NN \l_stex_current_docurl_str \l_tmpa_str
    }{
      \str_set:Nx \l_stex_current_docurl_str {
        \l_tmpa_str/\stex_path_to_string:N \l_tmpa_seq
      }
    }
  }
}
```

16.2 Setting Reference Targets

```
\str_const:Nn \c__stex_refs_url_str{URL}
\str_const:Nn \c__stex_refs_ref_str{REF}
\cs_new_protected:Nn \stex_ref_new_doc_target:n {
  \stex_get_document_uri:
  \str_set:Nx \l_tmpa_str { #1 }
  \str_if_empty:NT \l_tmpa_str {
    \int_zero:N \l_tmpa_int
    \bool_set_true:N \l_tmpa_bool
    \bool_while_do:Nn \l_tmpa_bool {
      \cs_if_exist:cTF {
        sref_\l_stex_current_docns_str\c_hash_str REF_\int_use:N \l_tmpa_int _type
      }{
        \int_incr:N \l_tmpa_int
      }{
        \str_set:Nx \l_tmpa_str { REF_\int_use:N \l_tmpa_int }
        \bool_set_false:N \l_tmpa_bool
      }
    }
  }
  \str_set:Nx \l_tmpa_str {
    \l_stex_current_docns_str\c_hash_str\l_tmpa_str
  }
  \seq_gput_right:No \g__stex_refs_all_refs_seq \l_tmpa_str
  \stex_if_smsmode:TF {
    \stex_get_document_url:
    \str_gset_eq:cN {sref_url_\l_tmpa_str _str}\l_stex_current_docurl_str
    \str_gset_eq:cN {sref_\l_tmpa_str _type}\c__stex_refs_url_str
  }{
    \iow_now:Nx \c__stex_refs_refs_iow { \l_tmpa_str~=~\expandafter{\@currentlabel~in~\exp_a
    \exp_after:wN\label\exp_after:wN{sref_\l_tmpa_str}
    \str_gset:cn {sref_\l_tmpa_str _type}\c__stex_refs_ref_str
  }
}
\cs_new_protected:Nn \stex_ref_new_sym_target:n {
  \str_gset_eq:cN {sref_sym_#1_uri} \l_stex_current_docns_str
}
```

16.3 Using References

```
\keys_define:nn { stex / sref } {
  linktext      .tl_set:N  = \l__stex_refs_linktext_tl ,
  fallback      .tl_set:N  = \l__stex_refs_fallback_tl ,
  pre           .tl_set:N  = \l__stex_refs_pre_tl ,
  post          .tl_set:N  = \l__stex_refs_post_tl ,
  indoc         .str_set_x:N  = \l__stex_refs_repo_str ,
}

\cs_new_protected:Nn \__stex_refs_args:n {
  \tl_clear:N \l__stex_refs_linktext_tl
  \tl_clear:N \l__stex_refs_fallback_tl
  \tl_clear:N \l__stex_refs_pre_tl
  \tl_clear:N \l__stex_refs_post_tl
  \str_clear:N \l__stex_refs_repo_str
  \keys_set:nn { stex / sref } { #1 }
}

⟨/package⟩
```

Chapter 17

sTeX -Modules Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   modules.dtx   %%%%%%%%%%%%%

⟨@@=stex_modules⟩
```

Warnings and error messages

```
\msg_new:nnn{stex}{error/unknownmodule}{
  No~module~#1~found
}
\msg_new:nnn{stex}{error/syntax}{
  Syntax~error:~#1
}
\msg_new:nnn{stex}{error/siglanguage}{
  Module~#1~declares~signature~#2,~but~does~not~
  declare~its~language
}
```

\l_stex_current_module_prop

The current module:

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

(*End definition for* \l_stex_current_module_prop. *This variable is documented on page 15.*)

\l_stex_all_modules_seq

Stores all available modules

```
\seq_new:N \l_stex_all_modules_seq
```

(*End definition for* \l_stex_all_modules_seq. *This variable is documented on page 15.*)

\g_stex_modules_in_file_seq
\g_stex_module_files_prop

All modules sorted by containing file; used e.g. in \importmodule

```
\seq_new:N \g_stex_modules_in_file_seq
\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 16.*)

\stex_if_in_module_p:
\stex_if_in_module:*TF*

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

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

\stex_if_module_exists_p:n
\stex_if_module_exists:n*TF*

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

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

\stex_add_to_current_module:n
\STEXexport

Only allowed within modules:

```
\cs_new_protected:Nn \stex_add_to_current_module:n {
  \prop_get:NnN \l_stex_current_module_prop { content } \l_tmpa_tl
  \tl_put_right:Nn \l_tmpa_tl { #1 }
  \prop_put:Nno \l_stex_current_module_prop { content } { \l_tmpa_tl }
}
\cs_new_protected:Npn \STEXexport #1 {
  #1
  \stex_add_to_current_module:n { #1 }
  \stex_smsmode_set_codes:
}
\stex_deactivate_macro:Nn \STEXexport {module~environments}
```

(*End definition for* `\stex_add_to_current_module:n` *and* `\STEXexport`*. These functions are documented on page 16.*)

\stex_add_constant_to_current_module:n

```
\cs_new_protected:Nn \stex_add_constant_to_current_module:n {
  \str_set:Nx \l_tmpa_str { #1 }
  \prop_get:NnN \l_stex_current_module_prop { constants } \l_tmpa_seq
  \seq_put_right:No \l_tmpa_seq { \l_tmpa_str }
  \prop_put:Nno \l_stex_current_module_prop { constants } \l_tmpa_seq
}
```

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

\stex_add_import_to_current_module:n

```
\cs_new_protected:Nn \stex_add_import_to_current_module:n {
  \str_set:Nx \l_tmpa_str { #1 }
  \prop_get:NnN \l_stex_current_module_prop { imports } \l_tmpa_seq
  \seq_put_right:No \l_tmpa_seq { \l_tmpa_str }
  \prop_put:Nno \l_stex_current_module_prop { imports } \l_tmpa_seq
}
```

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

\stex_modules_compute_namespace:nN Computer the appropriate namespace from the top-level namespace of a repository (#1) and a file path (#2).

```
\cs_new_protected:Nn \stex_modules_compute_namespace:nN {
  \str_set:Nx \l_tmpa_str { #1 }
  \seq_set_eq:NN \l_tmpa_seq #2
  % split off file extension
  \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
  \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
  \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
  \seq_put_right:No \l_tmpa_seq \l_tmpb_str

  \bool_set_true:N \l_tmpa_bool
  \bool_while_do:Nn \l_tmpa_bool {
    \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
    \exp_args:No \str_case:nnTF { \l_tmpb_str } {
      {source} { \bool_set_false:N \l_tmpa_bool }
    }{}{
      \seq_if_empty:NT \l_tmpa_seq {
        \bool_set_false:N \l_tmpa_bool
      }
    }
  }

  \seq_if_empty:NTF \l_tmpa_seq {
    \str_set_eq:NN \l_stex_modules_ns_str \l_tmpa_str
  }{
    \str_set:Nx \l_stex_modules_ns_str {
      \l_tmpa_str/\stex_path_to_string:N \l_tmpa_seq
    }
  }
}
```

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

Stores its return values in:

`\l_stex_modules_ns_str`

```
\str_new:N \l_stex_modules_ns_str
```

(*End definition for* `\l_stex_modules_ns_str`. *This variable is documented on page ??.*)

\stex_modules_current_namespace: Computes the current namespace based on the current MathHub repository (if existent) and the current file.

```
\cs_new_protected:Nn \stex_modules_current_namespace: {
  \prop_get:NnNTF \l_stex_current_repository_prop { ns } \l_tmpa_str {
    \stex_modules_compute_namespace:nN \l_tmpa_str \g_stex_currentfile_seq
  }{
    % split off file extension
    \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
    \seq_pop_right:NN \l_tmpa_seq \l_tmpb_str
    \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq . \l_tmpb_str
    \seq_get_left:NN \l_tmpb_seq \l_tmpb_str
    \seq_put_right:No \l_tmpa_seq \l_tmpb_str
    \str_set:Nx \l_stex_modules_ns_str {
      file:/\stex_path_to_string:N \l_tmpa_seq
    }
  }
}
```

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

17.1 The module environment

`module` arguments:

```
\keys_define:nn { stex / module } {
  title          .str_set_x:N  = \l_stex_module_title_str ,
  ns             .str_set_x:N  = \l_stex_module_ns_str ,
  lang           .str_set_x:N  = \l_stex_module_lang_str ,
  sig            .str_set_x:N  = \l_stex_module_sig_str ,
  creators       .str_set_x:N  = \l_stex_module_creators_str ,
  contributors   .str_set_x:N  = \l_stex_module_contributors_str ,
  meta           .str_set_x:N  = \l_stex_module_meta_str
}

\cs_new_protected:Nn \__stex_modules_args:n {
  \str_clear:N \l_stex_module_title_str
  \str_clear:N \l_stex_module_ns_str
  \str_clear:N \l_stex_module_lang_str
  \str_clear:N \l_stex_module_sig_str
  \str_clear:N \l_stex_module_creators_str
  \str_clear:N \l_stex_module_contributors_str
  \str_clear:N \l_stex_module_meta_str
  \keys_set:nn { stex / module } { #1 }
}

% module parameters here? In the body?

```

`\stex_module_setup:nn` Sets up a new module property list:

```
\cs_new_protected:Nn \stex_module_setup:nn {
  \str_set:Nx \l_stex_module_name_str { #2 }
  \__stex_modules_args:n { #1 }
```

First, we set up the name and namespace of the module.
Are we in a nested module?

```
  \stex_if_in_module:TF {
    % Nested module
    \prop_get:NnN \l_stex_current_module_prop
      { ns } \l_stex_module_ns_str
    \str_set:Nx \l_stex_module_name_str {
      \prop_item:Nn \l_stex_current_module_prop
        { name } / \l_stex_module_name_str
    }
  }{
    % not nested:
    \str_if_empty:NT \l_stex_module_ns_str {
      \stex_modules_current_namespace:
      \str_set_eq:NN \l_stex_module_ns_str \l_stex_modules_ns_str
      \exp_args:NNNo \seq_set_split:Nnn \l_tmpa_seq
         / {\l_stex_module_ns_str}
      \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
      \str_if_eq:NNT \l_tmpa_str \l_stex_module_name_str {
        \str_set:Nx \l_stex_module_ns_str {
          \stex_path_to_string:N \l_tmpa_seq
        }
      }
    }
  }
```

Next, we determine the language of the module:

```
  \str_if_empty:NT \l_stex_module_lang_str {
    \seq_get_right:NN \g_stex_currentfile_seq \l_tmpa_str
    \seq_set_split:NnV \l_tmpa_seq . \l_tmpa_str
    \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str % .tex
    \seq_pop_left:NN \l_tmpa_seq \l_tmpa_str % <filename>
    \seq_if_empty:NF \l_tmpa_seq { %remaining element should be language
      \stex_debug:nn{modules} {Language~\l_stex_module_lang_str~
        inferred~from~file~name}
      \seq_pop_left:NN \l_tmpa_seq \l_stex_module_lang_str
    }
  }

  \str_if_empty:NF \l_stex_module_lang_str {
    \prop_get:NVNTF \c_stex_languages_prop \l_stex_module_lang_str
      \l_tmpa_str {
        \ltx@ifpackageloaded{babel}{
          \exp_args:Nx \selectlanguage { \l_tmpa_str }
        }{}
      } {
        \msg_error:nnn{stex}{error/unknownlanguage}{\l_tmpa_str}
      }
  }
```

We check if we need to extend a signature module, and set `\l_stex_current_module_prop` accordingly:

```
  \str_if_empty:NTF \l_stex_module_sig_str {
    \str_clear:N \l_tmpa_str
    \seq_clear:N \l_tmpa_seq
    \tl_clear:N \l_tmpa_tl
    \exp_args:NNx \prop_set_from_keyval:Nn \l_stex_current_module_prop {
      name      = \l_stex_module_name_str ,
      ns        = \l_stex_module_ns_str ,
      imports   = \exp_not:o { \l_tmpa_seq } ,
      constants = \exp_not:o { \l_tmpa_seq } ,
      content   = \exp_not:o { \l_tmpa_tl }  ,
      file      = \exp_not:o { \g_stex_currentfile_seq } ,
      lang      = \l_stex_module_lang_str ,
      sig       = \l_stex_module_sig_str ,
      meta      = \l_stex_module_meta_str
    }
  }{
    \str_if_empty:NT \l_stex_module_lang_str {
      \msg_error:nnnn{stex}{error/siglanguage}{
        \l_stex_module_ns_str?\l_stex_module_name_str
      }{\l_stex_module_sig_str}
    }

    \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
    \seq_pop_right:NN \l_tmpa_seq \l_tmpa_str
    \seq_set_split:NnV \l_tmpb_seq . \l_tmpa_str
    \seq_pop_right:NN \l_tmpb_seq \l_tmpa_str % .tex
    \seq_pop_left:NN \l_tmpb_seq \l_tmpa_str % <filename>
    \str_set:Nx \l_tmpa_str {
      \stex_path_to_string:N \l_tmpa_seq /
      \l_tmpa_str . \l_stex_module_sig_str .tex
    }
    \IfFileExists \l_tmpa_str {
      \exp_args:No \stex_in_smsmode:nn { \l_tmpa_str } {
        \seq_clear:N \l_stex_all_modules_seq
        \prop_clear:N \l_stex_current_module_prop
        \stex_debug:nn{modules}{Loading~signature~\l_tmpa_str}
        \input { \l_tmpa_str }
      }
    }{
      \msg_error:nnn{stex}{error/unknownmodule}{for~signature~\l_tmpa_str}
    }
    \stex_activate_module:n {
      \l_stex_module_ns_str ? \l_stex_module_name_str
    }
    \prop_set_eq:Nc \l_stex_current_module_prop {
      c_stex_module_
      \l_stex_module_ns_str ?
      \l_stex_module_name_str
      _prop
    }
  }
```

We load the metatheory:

```
  \str_if_empty:NT \l_stex_module_meta_str {
    \str_set:Nx \l_stex_module_meta_str {
      \c_stex_metatheory_ns_str ? Metatheory
    }
  }
  \str_if_eq:VnF \l_stex_module_meta_str {NONE} {
    \exp_args:Nx \stex_add_to_current_module:n {
      \stex_activate_module:n {\l_stex_module_meta_str}
    }
    \stex_activate_module:n {\l_stex_module_meta_str}
  }
}
```

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

`module` The `module` environment.

`\__stex_modules_begin_module:nn` implements `\begin{module}`

```
\cs_new_protected:Nn \__stex_modules_begin_module:nn {
  \stex_reactivate_macro:N \STEXexport
  \stex_reactivate_macro:N \importmodule
  \stex_reactivate_macro:N \symdecl
  \stex_reactivate_macro:N \notation
  \stex_reactivate_macro:N \symdef
  \stex_module_setup:nn{#1}{#2}

  \stex_debug:nn{modules}{
    New~module:\\
    Namespace:~\l_stex_module_ns_str\\
    Name:~\l_stex_module_name_str\\
    Language:~\l_stex_module_lang_str\\
    Signature:~\l_stex_module_sig_str\\
    Metatheory:~\l_stex_module_meta_str\\
    File:~\stex_path_to_string:N \g_stex_currentfile_seq
  }

  \seq_put_right:Nx \l_stex_all_modules_seq {
    \l_stex_module_ns_str ? \l_stex_module_name_str
  }

  \seq_gput_right:Nx  \g_stex_modules_in_file_seq
      { \l_stex_module_ns_str ? \l_stex_module_name_str }

  \stex_if_smsmode:TF {
    \stex_smsmode_set_codes:
  } {
    \begin{stex_annotate_env} {theory} {
      \l_stex_module_ns_str ? \l_stex_module_name_str
    }

    \stex_annotate_invisible:nnn{header}{} {
      \stex_annotate:nnn{language}{ \l_stex_module_lang_str }{}
      \stex_annotate:nnn{signature}{ \l_stex_module_sig_str }{}
      \str_if_eq:VnF \l_stex_module_meta_str {NONE} {
        \stex_annotate:nnn{metatheory}{ \l_stex_module_meta_str }{}
      }
    }
  }
  % TODO: Inherit metatheory for nested modules?
}
\iffalse \end{stex_annotate_env} \fi %^^A make syntax highlighting work again
```

(*End definition for* `\__stex_modules_begin_module:nn`.)

`\__stex_modules_end_module:` implements `\end{module}`

```
\cs_new_protected:Nn \__stex_modules_end_module: {
  \str_set:Nx \l_tmpa_str {
    c_stex_module_
    \prop_item:Nn \l_stex_current_module_prop { ns } ?
    \prop_item:Nn \l_stex_current_module_prop { name }
    _prop
  }
  %^^A \prop_new:c { \l_tmpa_str }
  \prop_gset_eq:cN { \l_tmpa_str } \l_stex_current_module_prop
  \stex_debug:nn{modules}{Closing~module~\prop_item:Nn \l_stex_current_module_prop { name }}
}
```

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

`@module` The core environment, with no header

```
\iffalse \begin{stex_annotate_env} \fi %^^A make syntax highlighting work again
\NewDocumentEnvironment { @module } { O{} m } {
  \par
  \__stex_modules_begin_module:nn{#1}{#2}
} {
  \__stex_modules_end_module:
  \stex_if_smsmode:TF {
    \exp_args:Nx \stex_add_to_sms:n {
      \prop_gset_from_keyval:cn {
        c_stex_module_
        \prop_item:Nn \l_stex_current_module_prop { ns } ?
        \prop_item:Nn \l_stex_current_module_prop { name }
        _prop
      } {
        name      = \prop_item:cn { \l_tmpa_str } { name } ,
        ns        = \prop_item:cn { \l_tmpa_str } { ns } ,
        imports   = \prop_item:cn { \l_tmpa_str } { imports } ,
        constants = \prop_item:cn { \l_tmpa_str } { constants } ,
        content   = \prop_item:cn { \l_tmpa_str } { content } ,
        file      = \prop_item:cn { \l_tmpa_str } { file } ,
        lang      = \prop_item:cn { \l_tmpa_str } { lang } ,
        sig       = \prop_item:cn { \l_tmpa_str } { sig } ,
        meta      = \prop_item:cn { \l_tmpa_str } { meta }
      }
    }
  }{
    \end{stex_annotate_env}
  }
}
```

`\stex_modules_heading:` Code for document headers

```
\cs_if_exist:NTF \thesection {
  \newcounter{module}[section]
}{
  \newcounter{module}
}

\bool_if:NT \c_stex_showmods_bool {
  \latexml_if:F { \RequirePackage{mdframed} }
}

\cs_new_protected:Nn \stex_modules_heading: {
  \stepcounter{module}
  \par
  \bool_if:NT \c_stex_showmods_bool {
    \noindent{\textbf{Module} ~
```

```
      \cs_if_exist:NT \thesection {\thesection.}
      \themodule ~ [\l_stex_module_name_str]
    }
    \str_if_empty:NTF \l_stex_module_title_str {
    }{
      \quad(\l_stex_module_title_str)\hfill
    }\par
  }
  \edef\@currentlabel{Module~\thesection.\themodule~[\l_stex_module_name_str]}
  % TODO
  \stex_ref_new_doc_target:n \l_stex_module_name_str
}
```

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

Finally:

```
\NewDocumentEnvironment { module } { O{} m } {
  \bool_if:NT \c_stex_showmods_bool {
    \begin{mdframed}
  }
  \begin{@module}[#1]{#2}
  \stex_modules_heading:
}{
  \end{@module}
  \bool_if:NT \c_stex_showmods_bool {
    \end{mdframed}
  }
}
```

17.2 Invoking modules

`\STEXModule`
`\stex_invoke_module:n`

```
\NewDocumentCommand \STEXModule { m } {
  \exp_args:NNx \str_set:Nn \l_tmpa_str { #1 }
  \int_set:Nn \l_tmpa_int { \str_count:N \l_tmpa_str }
  \tl_set:Nn \l_tmpa_tl {
    \msg_error:nnn{stex}{error/unknownmodule}{#1}
  }
  \seq_map_inline:Nn \l_stex_all_modules_seq {
    \str_set:Nn \l_tmpb_str { ##1 }
    \str_if_eq:eeT { \l_tmpa_str } {
      \str_range:Nnn \l_tmpb_str { -\l_tmpa_int } { -1 }
    } {
      \seq_map_break:n {
        \tl_set:Nn \l_tmpa_tl {
          \stex_invoke_module:n { ##1 }
        }
      }
    }
  }
  \l_tmpa_tl
}

\cs_new_protected:Nn \stex_invoke_module:n {
  \stex_debug:nn{modules}{Invoking~module~#1}
  \peek_charcode_remove:NTF ! {
    \__stex_modules_invoke_uri:nN { #1 }
  } {
    \peek_charcode_remove:NTF ? {
      \__stex_modules_invoke_symbol:nn { #1 }
    } {
      \msg_error:nnn{stex}{error/syntax}{
        ?~or~!~expected~after~
        \c_backslash_str STEXModule{#1}
      }
    }
  }
}

\cs_new_protected:Nn \__stex_modules_invoke_uri:nN {
  \str_set:Nn #2 { #1 }
}

\cs_new_protected:Nn \__stex_modules_invoke_symbol:nn {
  \stex_invoke_symbol:n{#1?#2}
}
```

(*End definition for* `\STEXModule` *and* `\stex_invoke_module:n`*. These functions are documented on page 18.*)

`\stex_activate_module:n`

```
\cs_new_protected:Nn \stex_activate_module:n {
  \stex_debug:nn{modules}{Activating~module~#1}
  \exp_args:NNx \seq_if_in:NnF \l_stex_all_modules_seq { #1 } {
    \seq_put_right:Nx \l_stex_all_modules_seq { #1 }
    \prop_item:cn { c_stex_module_#1_prop } { content }
  }
}
```

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

```
⟨/package⟩
```

Chapter 18

STEX -Module Inheritance Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   inheritance.dtx   %%%%%%%%%%%%%

```

18.1 SMS Mode

```
⟨@@=stex_smsmode⟩
```

\g_stex_smsmode_allowedmacros_tl
\g_stex_smsmode_allowedmacros_escape_tl
\g_stex_smsmode_allowedenvs_seq

```
\tl_new:N \g_stex_smsmode_allowedmacros_tl
\tl_new:N \g_stex_smsmode_allowedmacros_escape_tl
\seq_new:N \g_stex_smsmode_allowedenvs_seq

\tl_set:Nn \g_stex_smsmode_allowedmacros_tl {
  \makeatletter
  \makeatother
  \ExplSyntaxOn
  \ExplSyntaxOff
}

\tl_set:Nn \g_stex_smsmode_allowedmacros_escape_tl {
  \symdef
  \importmodule
  \notation
  \symdecl
  \STEXexport
}

\exp_args:NNx \seq_set_from_clist:Nn \g_stex_smsmode_allowedenvs_seq {
  \tl_to_str:n {
    module,
    @module
  }
}
```

(*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 20.*)

`\stex_if_smsmode_p:`
`\stex_if_smsmode:TF`

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

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

`\__stex_smsmode_if_catcodes_p:`
`__stex_smsmode_if_catcodes:TF`

Checks whether the SMS mode category code scheme is active.

```
\bool_new:N \g__stex_smsmode_catcode_bool
\bool_set_false:N \g__stex_smsmode_catcode_bool
\prg_new_conditional:Nnn \__stex_smsmode_if_catcodes: { p, T, F, TF } {
  \bool_if:NTF \g__stex_smsmode_catcode_bool
    \prg_return_true: \prg_return_false:
}
```

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

`\stex_smsmode_set_codes:`

```
\cs_new_protected:Nn \stex_smsmode_set_codes: {
  \stex_if_smsmode:T {
    \__stex_smsmode_if_catcodes:F {
      \bool_gset_true:N \g__stex_smsmode_catcode_bool
      \exp_after:wN \char_gset_active_eq:NN
        \c_backslash_str \__stex_smsmode_cs:
      \tex_global:D \char_set_catcode_active:N \\
      \tex_global:D \char_set_catcode_other:N $
      \tex_global:D \char_set_catcode_other:N ^
      \tex_global:D \char_set_catcode_other:N _
      \tex_global:D \char_set_catcode_other:N &
      \tex_global:D \char_set_catcode_other:N ##
    }
  }
} \iffalse $ \fi % to make syntax highlighting work again
```

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

`\__stex_smsmode_unset_codes:`

Sets category code scheme back from the one used in SMS mode.

```
\cs_new_protected:Nn \__stex_smsmode_unset_codes: {
  \__stex_smsmode_if_catcodes:T {
    \bool_gset_false:N \g__stex_smsmode_catcode_bool
    \exp_after:wN \tex_global:D \exp_after:wN
      \char_set_catcode_escape:N \c_backslash_str
    \tex_global:D \char_set_catcode_math_toggle:N $
    \tex_global:D \char_set_catcode_math_superscript:N ^
    \tex_global:D \char_set_catcode_math_subscript:N _
    \tex_global:D \char_set_catcode_alignment:N &
    \tex_global:D \char_set_catcode_parameter:N ##
  }
} \iffalse $ \fi % to make syntax highlighting work again
```

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

`\stex_in_smsmode:nn`

```
\cs_new_protected:Nn \stex_in_smsmode:nn {
  \vbox_set:Nn \l_tmpa_box {
    \bool_set_eq:cN { l__stex_smsmode_#1_bool } \g__stex_smsmode_bool
    \bool_gset_true:N \g__stex_smsmode_bool
    \stex_smsmode_set_codes:
    #2
    \bool_gset_eq:Nc \g__stex_smsmode_bool { l__stex_smsmode_#1_bool }
    \stex_if_smsmode:F {
      \__stex_smsmode_unset_codes:
    }
  }
  \box_clear:N \l_tmpa_box
}
```

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

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

```
\cs_new_protected:Nn \__stex_smsmode_cs: {
  \str_clear:N \l_tmpa_str
  \peek_analysis_map_inline:n {
    % #1: token (one expansion)
    % #2: charcode
    % #3 catcode
    \token_if_eq_charcode:NNTF ##3 B {
      % token is a letter
      \exp_args:NNo \str_put_right:Nn \l_tmpa_str { ##1 }
    } {
      \str_if_empty:NTF \l_tmpa_str {
        % we don't allow (or need) single non-letter CSs
        % for now
        \peek_analysis_map_break:
      }{
        \str_if_eq:onTF \l_tmpa_str { begin } {
          \peek_analysis_map_break:n {
            \exp_after:wN \__stex_smsmode_checkbegin:n ##1
          }
        } {
          \str_if_eq:onTF \l_tmpa_str { end } {
            \peek_analysis_map_break:n {
              \exp_after:wN \__stex_smsmode_checkend:n ##1
            }
          } {
          \tl_set:Nn \l_tmpa_tl { \use:c{\l_tmpa_str} }
          \exp_args:NNNo \exp_args:NNo \tl_if_in:NnTF
            \g_stex_smsmode_allowedmacros_tl
              { \use:c{\l_tmpa_str} } {
              \stex_debug:nn{modules}{Executing~1:~\l_tmpa_str}
              \peek_analysis_map_break:n {
                \exp_after:wN \l_tmpa_tl ##1
              }
            } {
              \exp_args:NNNo \exp_args:NNo \tl_if_in:NnTF
              \g_stex_smsmode_allowedmacros_escape_tl
                { \use:c{\l_tmpa_str} } {
                \__stex_smsmode_unset_codes:
                \stex_debug:nn{modules}{Executing~2:~\l_tmpa_str}
                % TODO \__stex_smsmode_rescan_cs:
%                 \exp_after:wN \exp_after:wN \exp_after:wN
%                 \token_if_eq_charcode:NNTF \exp_after:wN \c_backslash_str ##1 {
%                   \peek_analysis_map_break:n {
%                     \__stex_smsmode_unset_codes:
%                     \__stex_smsmode_rescan_cs:
%                   }
%                 } {
                  \peek_analysis_map_break:n {
                    \exp_after:wN \l_tmpa_tl ##1
                  }
%                 }
              } {
                \peek_analysis_map_break:n { ##1 }
              }
            }
          }
        }
      }
    }
  }
}
```

(*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:

```
\cs_new_protected:Nn \__stex_smsmode_rescan_cs: {
  \str_clear:N \l_tmpb_str
  \peek_analysis_map_inline:n {
    \token_if_eq_charcode:NNTF ##3 B {
      % token is a letter
      \exp_args:NNo \str_put_right:Nn \l_tmpb_str { ##1 }
    } {
      \peek_analysis_map_break:n {
        \exp_after:wN \use:c \exp_after:wN {
          \exp_after:wN \l_tmpa_str\exp_after:wN
        } \use:c { \l_tmpb_str \exp_after:wN } ##1
      }
    }
  }
}
```

(*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.

```
\cs_new_protected:Nn \__stex_smsmode_checkbegin:n {
  \str_set:Nn \l_tmpa_str { #1 }
  \seq_if_in:NoT \g_stex_smsmode_allowedenvs_seq \l_tmpa_str {
    \__stex_smsmode_unset_codes:
    \begin{#1}
  }
}
```

(*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.

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

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

18.2 Inheritance

```
⟨@@=stex_importmodule⟩
```

`\stex_import_module_uri:nn`

```
\cs_new_protected:Nn \stex_import_module_uri:nn {
  \str_set:Nx \l__stex_importmodule_archive_str { #1 }
  \str_set:Nn \l__stex_importmodule_path_str { #2 }
  \str_if_empty:NT \l__stex_importmodule_archive_str {
    \prop_if_empty:NF \l_stex_current_repository_prop {
      \prop_get:NnN \l_stex_current_repository_prop { id } \l__stex_importmodule_archive_str
    }
  }

  \exp_args:NNNo \seq_set_split:Nnn \l_tmpb_seq ? { \l__stex_importmodule_path_str }
  \seq_pop_right:NN \l_tmpb_seq \l__stex_importmodule_name_str
  \str_set:Nx \l__stex_importmodule_path_str { \seq_use:Nn \l_tmpb_seq ? }

  \str_if_empty:NTF \l__stex_importmodule_archive_str {
    \stex_modules_current_namespace:
    \str_if_empty:NF \l__stex_importmodule_path_str {
      \str_set:Nx \l_stex_module_ns_str {
        \l_stex_module_ns_str / \l__stex_importmodule_path_str
      }
    }
  }{
    \stex_require_repository:n \l__stex_importmodule_archive_str
    \prop_get:cnN { c_stex_mathhub_\l__stex_importmodule_archive_str _manifest_prop } { ns }
      \l_stex_module_ns_str
    \str_if_empty:NF \l__stex_importmodule_path_str {
      \str_set:Nx \l_stex_module_ns_str {
        \l_stex_module_ns_str / \l__stex_importmodule_path_str
      }
    }
  }
}
```

(*End definition for* `\stex_import_module_uri:nn`*. This function is documented on page 23.*)

\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`.

```
\str_new:N \l__stex_importmodule_name_str
\str_new:N \l__stex_importmodule_archive_str
\str_new:N \l__stex_importmodule_path_str
\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*⟩}

```
\cs_new_protected:Nn \stex_import_require_module:nnnn {
  \exp_args:Nx \stex_if_module_exists:nF { #1 ? #4 } {

    % archive
    \str_set:Nx \l_tmpa_str { #2 }
    \str_if_empty:NTF \l_tmpa_str {
      \seq_set_eq:NN \l_tmpa_seq \g_stex_currentfile_seq
    } {
      \stex_path_from_string:Nn \l_tmpb_seq { \l_tmpa_str }
      \seq_concat:NNN \l_tmpa_seq \c_stex_mathhub_seq \l_tmpb_seq
      \seq_put_right:Nn \l_tmpa_seq { source }
    }

    % path
    \str_set:Nx \l_tmpb_str { #3 }
    \str_if_empty:NTF \l_tmpb_str {
      \str_set:Nx \l_tmpa_str { \stex_path_to_string:N \l_tmpa_seq / #4 }

      \ltx@ifpackageloaded{babel} {
        \exp_args:NNx \prop_get:NnNF \c_stex_language_abbrevs_prop
            { \languagename } \l_tmpb_str {
              \msg_error:nnn{stex}{error/unknownlanguage}{\languagename}
            }
      } {
        \str_clear:N \l_tmpb_str
      }

      \stex_debug:nn{modules}{Checking~\l_tmpa_str.\l_tmpb_str.tex}
      \IfFileExists{ \l_tmpa_str.\l_tmpb_str.tex }{
        \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.\l_tmpb_str.tex }
      }{
        \stex_debug:nn{modules}{Checking~\l_tmpa_str.tex}
        \IfFileExists{ \l_tmpa_str.tex }{
          \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.tex }
        }{
          % try english as default
          \stex_debug:nn{modules}{Checking~\l_tmpa_str.en.tex}
          \IfFileExists{ \l_tmpa_str.en.tex }{
            \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.en.tex }
          }{
            \msg_error:nnn{stex}{error/unknownmodule}{#1?#4}
          }
        }
      }

    } {
      \seq_set_split:NnV \l_tmpb_seq / \l_tmpb_str
      \seq_concat:NNN \l_tmpa_seq \l_tmpa_seq \l_tmpb_seq

      \ltx@ifpackageloaded{babel} {
        \exp_args:NNx \prop_get:NnNF \c_stex_language_abbrevs_prop
            { \languagename } \l_tmpb_str {
              \msg_error:nnn{stex}{error/unknownlanguage}{\languagename}
            }
      } {
        \str_clear:N \l_tmpb_str
      }

      \stex_path_to_string:NN \l_tmpa_seq \l_tmpa_str

      \stex_debug:nn{modules}{Checking~\l_tmpa_str/#4.\l_tmpb_str.tex}
      \IfFileExists{ \l_tmpa_str/#4.\l_tmpb_str.tex }{
        \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.\l_tmpb_str.tex }
      }{
        \stex_debug:nn{modules}{Checking~\l_tmpa_str/#4.tex}
        \IfFileExists{ \l_tmpa_str/#4.tex }{
          \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.tex }
        }{
          % try english as default
          \stex_debug:nn{modules}{Checking~\l_tmpa_str/#4.en.tex}
          \IfFileExists{ \l_tmpa_str/#4.en.tex }{
            \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str/#4.en.tex }
          }{
            \stex_debug:nn{modules}{Checking~\l_tmpa_str.\l_tmpb_str.tex}
            \IfFileExists{ \l_tmpa_str.\l_tmpb_str.tex }{
              \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.\l_tmpb_str.tex }
            }{
              \stex_debug:nn{modules}{Checking~\l_tmpa_str.tex}
              \IfFileExists{ \l_tmpa_str.tex }{
                \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.tex }
              }{
                % try english as default
                \stex_debug:nn{modules}{Checking~\l_tmpa_str.en.tex}
                \IfFileExists{ \l_tmpa_str.en.tex }{
                  \str_gset:Nx \g__stex_importmodule_file_str { \l_tmpa_str.en.tex }
                }{
                  \msg_error:nnn{stex}{error/unknownmodule}{#1?#4}
                }
              }
            }
          }
        }
      }
    }

    \seq_set_eq:NN \l_tmpa_seq \g_stex_modules_in_file_seq
    \seq_clear:N \g_stex_modules_in_file_seq
```

```
%    \exp_args:Nnx \use:nn {
      \exp_args:No \stex_in_smsmode:nn { \g__stex_importmodule_file_str } {
        \seq_clear:N \l_stex_all_modules_seq
        \prop_clear:N \l_stex_current_module_prop
        \str_set:Nx \l_tmpb_str { #2 }
        \str_if_empty:NF \l_tmpb_str {
          \stex_set_current_repository:n { #2 }
        }
        \stex_debug:nn{modules}{Loading~\g__stex_importmodule_file_str}
        \input { \g__stex_importmodule_file_str }
      }
%    }{

%    }
    \prop_gput:Noo \g_stex_module_files_prop
    \g__stex_importmodule_file_str \g_stex_modules_in_file_seq
    \seq_set_eq:NN \g_stex_modules_in_file_seq \l_tmpa_seq

    \stex_if_module_exists:nF { #1 ? #4 } {
      \msg_error:nnn{stex}{error/unknownmodule}{
        #1?#4~(in~file~\g__stex_importmodule_file_str)
      }
    }
  }
  \stex_activate_module:n { #1 ? #4 }
}
```

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

`\importmodule`

```
\NewDocumentCommand \importmodule { O{} m } {
  \stex_import_module_uri:nn { #1 } { #2 }
  \stex_debug:nn{modules}{Importing~module:~
    \l_stex_module_ns_str ? \l__stex_importmodule_name_str
  }
  \stex_if_smsmode:F {
    \stex_import_require_module:nnnn
    { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
    { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
    \stex_annotate_invisible:nnn
      {import} {\l_stex_module_ns_str ? \l__stex_importmodule_name_str} {}
  }
  \exp_args:Nx \stex_add_to_current_module:n {
    \stex_import_require_module:nnnn
    { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
    { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
  }
  \exp_args:Nx \stex_add_import_to_current_module:n {
    \l_stex_module_ns_str ? \l__stex_importmodule_name_str
  }
  \stex_smsmode_set_codes:
}
\stex_deactivate_macro:Nn \importmodule {module~environments}
```

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

\usemodule

```
\NewDocumentCommand \usemodule { O{} m } {
  \stex_if_smsmode:F {
    \stex_import_module_uri:nn { #1 } { #2 }
    \stex_import_require_module:nnnn
    { \l_stex_module_ns_str } { \l__stex_importmodule_archive_str }
    { \l__stex_importmodule_path_str } { \l__stex_importmodule_name_str }
    \stex_annotate_invisible:nnn
      {usemodule} {\l_stex_module_ns_str ? \l__stex_importmodule_name_str} {}
  }
  \stex_smsmode_set_codes:
}
```

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

```
⟨/package⟩
```

Chapter 19

sTeX -Symbols Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   symbols.dtx   %%%%%%%%%%%%%

```

Warnings and error messages

19.1 Symbol Declarations

```
⟨@@=stex_symdecl⟩
```

\l_stex_all_symbols_seq Stores all available symbols

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

(*End definition for* \l_stex_all_symbols_seq*. This variable is documented on page 25.*)

\STEXsymbol

```
\NewDocumentCommand \STEXsymbol { m } {
  \stex_get_symbol:n { #1 }
  \exp_args:No
  \stex_invoke_symbol:n { \l_stex_get_symbol_uri_str }
}
```

(*End definition for* \STEXsymbol*. This function is documented on page 27.*)

symdecl arguments:

```
\keys_define:nn { stex / symdecl } {
  name        .str_set_x:N  = \l_stex_symdecl_name_str ,
  local       .bool_set:N  = \l_stex_symdecl_local_bool ,
  args        .str_set_x:N  = \l_stex_symdecl_args_str ,
  type        .tl_set:N    = \l_stex_symdecl_type_tl ,
  align       .str_set:N    = \l_stex_symdecl_align_str , % TODO(?)
  gfc         .str_set:N    = \l_stex_symdecl_gfc_str , % TODO(?)
  specializes .str_set:N    = \l_stex_symdecl_specializes_str , % TODO(?)
  def         .tl_set:N    = \l_stex_symdecl_definiens_tl
}
```

```

\bool_new:N \l_stex_symdecl_make_macro_bool

\cs_new_protected:Nn \__stex_symdecl_args:n {
  \str_clear:N \l_stex_symdecl_name_str
  \str_clear:N \l_stex_symdecl_args_str
  \bool_set_false:N \l_stex_symdecl_local_bool
  \tl_clear:N \l_stex_symdecl_type_tl
  \tl_clear:N \l_stex_symdecl_definiens_tl

  \keys_set:nn { stex / symdecl } { #1 }
}
```

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

```

\NewDocumentCommand \symdecl { s O{} m } {
  \__stex_symdecl_args:n { #2 }
  \IfBooleanTF #1 {
    \bool_set_false:N \l_stex_symdecl_make_macro_bool
  } {
    \bool_set_true:N \l_stex_symdecl_make_macro_bool
  }
  \stex_symdecl_do:n { #3 }
  \stex_smsmode_set_codes:
}
\stex_deactivate_macro:Nn \symdecl {module~environments}
```

(*End definition for* \symdecl*. This function is documented on page 24.*)

\stex_symdecl_do:n

```
\cs_new_protected:Nn \stex_symdecl_do:n {
  \stex_if_in_module:F {
    % TODO throw error? some default namespace?
  }

  \str_if_empty:NT \l_stex_symdecl_name_str {
    \str_set:Nx \l_stex_symdecl_name_str { #1 }
  }

  \prop_if_exist:cT { g_stex_symdecl_
    \prop_item:Nn \l_stex_current_module_prop {ns} ?
    \prop_item:Nn \l_stex_current_module_prop {name} ?
      \l_stex_symdecl_name_str
    _prop
  }{
    % TODO throw error (beware of circular dependencies)
  }

  \prop_clear:N \l_tmpa_prop
  \prop_put:Nnx \l_tmpa_prop { module } {
    \prop_item:Nn \l_stex_current_module_prop {ns} ?
    \prop_item:Nn \l_stex_current_module_prop {name}
  }
```

```
\seq_clear:N \l_tmpa_seq
\prop_put:Nno \l_tmpa_prop { notations } \l_tmpa_seq
\prop_put:Nno \l_tmpa_prop { name } \l_stex_symdecl_name_str
\prop_put:Nno \l_tmpa_prop { local } \l_stex_symdecl_local_bool
\prop_put:Nno \l_tmpa_prop { type } \l_stex_symdecl_type_tl

\exp_args:No \stex_add_constant_to_current_module:n {
  \l_stex_symdecl_name_str
}

% arity/args
\int_zero:N \l_tmpb_int

\bool_set_true:N \l_tmpa_bool
\str_map_inline:Nn \l_stex_symdecl_args_str {
  \token_case_meaning:NnF ##1 {
    0 {} 1 {} 2 {} 3 {} 4 {} 5 {} 6 {} 7 {} 8 {} 9 {}
    {\tl_to_str:n i} { \bool_set_false:N \l_tmpa_bool }
    {\tl_to_str:n b} { \bool_set_false:N \l_tmpa_bool }
    {\tl_to_str:n a} {
      \bool_set_false:N \l_tmpa_bool
      \int_incr:N \l_tmpb_int
    }
    {\tl_to_str:n B} {
      \bool_set_false:N \l_tmpa_bool
      \int_incr:N \l_tmpb_int
    }
  }{
    \msg_set:nnn{stex}{error/wrongargs}{
      args~value~in~symbol~declaration~for~
      \prop_item:Nn \l_stex_current_module_prop {ns} ?
      \prop_item:Nn \l_stex_current_module_prop {name} ?
      \l_stex_symdecl_name_str ~
      needs~to~be~
      i,~a,~b~or~B,~but~##1~given
    }
    \msg_error:nn{stex}{error/wrongargs}
  }
}
\bool_if:NTF \l_tmpa_bool {
  % possibly numeric
  \str_if_empty:NTF \l_stex_symdecl_args_str {
    \prop_put:Nnn \l_tmpa_prop { args } {}
    \prop_put:Nnn \l_tmpa_prop { arity } { 0 }
  }{
    \int_set:Nn \l_tmpa_int { \l_stex_symdecl_args_str }
    \prop_put:Nnx \l_tmpa_prop { arity } { \int_use:N \l_tmpa_int }
    \str_clear:N \l_tmpa_str
    \int_step_inline:nn \l_tmpa_int {
      \str_put_right:Nn \l_tmpa_str i
    }
    \prop_put:Nnx \l_tmpa_prop { args } { \l_tmpa_str }
  }
} {
  \prop_put:Nnx \l_tmpa_prop { args } { \l_stex_symdecl_args_str }
  \prop_put:Nnx \l_tmpa_prop { arity }
    { \str_count:N \l_stex_symdecl_args_str }
}
\prop_put:Nnx \l_tmpa_prop { assocs } { \int_use:N \l_tmpb_int }


% semantic macro

\bool_if:NT \l_stex_symdecl_make_macro_bool {
  \tl_set:cx { #1 } { \stex_invoke_symbol:n {
    \prop_item:Nn \l_tmpa_prop { module } ?
      \prop_item:Nn \l_tmpa_prop { name }
  } }

  \bool_if:NF \l_stex_symdecl_local_bool {
    \exp_args:Nx \stex_add_to_current_module:n {
      \tl_set:cx { #1 } { \stex_invoke_symbol:n {
        \prop_item:Nn \l_tmpa_prop { module } ?
          \prop_item:Nn \l_tmpa_prop { name }
      } }
    }
  }
}

% add to all symbols

\bool_if:NF \l_stex_symdecl_local_bool {
  \exp_args:Nx \stex_add_to_current_module:n {
    \seq_put_right:Nn \exp_not:N \l_stex_all_symbols_seq {
      \prop_item:Nn \l_tmpa_prop { module } ?
      \prop_item:Nn \l_tmpa_prop { name }
    }
  }
}

\stex_debug:nn{symbols}{New~symbol:~
  \prop_item:Nn \l_tmpa_prop { module } ?
    \prop_item:Nn \l_tmpa_prop { name }^^J
  Type:~\exp_not:o { \l_stex_symdecl_type_tl }^^J
  Args:~\prop_item:Nn \l_tmpa_prop { args }
}

% circular dependencies require this:

\prop_if_exist:cF {
  g_stex_symdecl_
  \prop_item:Nn \l_tmpa_prop { module } ?
  \prop_item:Nn \l_tmpa_prop { name }
  _prop
} {
  \prop_gset_eq:cN {
    g_stex_symdecl_
    \prop_item:Nn \l_tmpa_prop { module } ?
```

```
      \prop_item:Nn \l_tmpa_prop { name }
      _prop
    } \l_tmpa_prop
  }

  \stex_if_smsmode:TF {
    \bool_if:NF \l_stex_symdecl_local_bool {
      \exp_args:Nx \stex_add_to_sms:n {
        \prop_gset_from_keyval:cn {
          g_stex_symdecl_
          \prop_item:Nn \l_tmpa_prop { module } ?
          \prop_item:Nn \l_tmpa_prop { name }
          _prop
        } {
          name      = \prop_item:Nn \l_tmpa_prop { name }       ,
          module    = \prop_item:Nn \l_tmpa_prop { module }     ,
          notations = \prop_item:Nn \l_tmpa_prop { notations }  ,
          local     = \prop_item:Nn \l_tmpa_prop { local }      ,
          type      = \prop_item:Nn \l_tmpa_prop { type }       ,
          args      = \prop_item:Nn \l_tmpa_prop { args }       ,
          arity     = \prop_item:Nn \l_tmpa_prop { arity }      ,
          assocs    = \prop_item:Nn \l_tmpa_prop { assocs }
        }
        \seq_put_right:Nn \exp_not:N \l_stex_all_symbols_seq {
          \prop_item:Nn \l_tmpa_prop { module } ?
          \prop_item:Nn \l_tmpa_prop { name }
        }
      }
    }
  }{
    \exp_args:NNx \seq_put_right:Nn \l_stex_all_symbols_seq {
      \prop_item:Nn \l_tmpa_prop { module } ?
      \prop_item:Nn \l_tmpa_prop { name }
    }
    \stex_if_do_html:T {
      \stex_annotate_invisible:nnn {symdecl} {
        \prop_item:Nn \l_tmpa_prop { module } ?
        \prop_item:Nn \l_tmpa_prop { name }
      } {
        \stex_annotate_invisible:nnn{type}{}{$\l_stex_symdecl_type_tl$}
        \stex_annotate_invisible:nnn{args}{}{
          \prop_item:Nn \l_tmpa_prop { args }
        }
        \stex_annotate_invisible:nnn{macroname}{}{#1}
        \tl_if_empty:NF \l_stex_symdecl_definiens_tl {
          \stex_annotate_invisible:nnn{definiens}{}
           {$\l_stex_symdecl_definiens_tl$}
        }
      }
    }
  }
}
```

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

\stex_get_symbol:n

```
\str_new:N \l_stex_get_symbol_uri_str

\cs_new_protected:Nn \stex_get_symbol:n {
  \tl_if_head_eq_catcode:nNTF { #1 } \relax {
    \__stex_symdecl_get_symbol_from_cs:n { #1 }
  }{
    % argument is a string
    % is it a command name?
    \cs_if_exist:cTF { #1 }{
      \cs_set_eq:Nc \l_tmpa_tl { #1 }
      \str_set:Nx \l_tmpa_str { \cs_argument_spec:N \l_tmpa_tl }
      \str_if_empty:NTF \l_tmpa_str {
        \exp_args:Nx \cs_if_eq:NNTF {
          \tl_head:N \l_tmpa_tl
        } \stex_invoke_symbol:n {
          \exp_args:No \__stex_symdecl_get_symbol_from_cs:n { \use:c { #1 } }
        }{
          \__stex_symdecl_get_symbol_from_string:n { #1 }
        }
      } {
        \__stex_symdecl_get_symbol_from_string:n { #1 }
      }
    }{
      % argument is not a command name
      \__stex_symdecl_get_symbol_from_string:n { #1 }
      % \l_stex_all_symbols_seq
    }
  }
}

\cs_new_protected:Nn \__stex_symdecl_get_symbol_from_string:n {
  \str_set:Nn \l_tmpa_str { #1 }
  \bool_set_false:N \l_tmpa_bool
  \stex_if_in_module:T {
    \prop_get:NnN \l_stex_current_module_prop
    { constants } \l_tmpa_seq
    \exp_args:NNo \seq_if_in:NnT \l_tmpa_seq { \l_tmpa_str } {
      \bool_set_true:N \l_tmpa_bool
      \str_set:Nx \l_stex_get_symbol_uri_str {
        \prop_item:Nn \l_stex_current_module_prop { ns } ?
        \prop_item:Nn \l_stex_current_module_prop { name } ? #1
      }
    }
  }
  \bool_if:NF \l_tmpa_bool {
    \tl_set:Nn \l_tmpa_tl {
      \msg_set:nnn{stex}{error/unknownsymbol}{
        No~symbol~#1~found!
      }
      \msg_error:nn{stex}{error/unknownsymbol}
    }
    \str_set:Nn \l_tmpa_str { #1 }
    \int_set:Nn \l_tmpa_int { \str_count:N \l_tmpa_str }
    \seq_map_inline:Nn \l_stex_all_symbols_seq {
      \str_set:Nn \l_tmpb_str { ##1 }
      \str_if_eq:eeT { \l_tmpa_str } {
        \str_range:Nnn \l_tmpb_str { -\l_tmpa_int } { -1 }
      } {
        \seq_map_break:n {
          \tl_set:Nn \l_tmpa_tl {
            \str_set:Nn \l_stex_get_symbol_uri_str {
              ##1
            }
          }
        }
      }
    }
    \l_tmpa_tl
  }
}

\cs_new_protected:Nn \__stex_symdecl_get_symbol_from_cs:n {
  \exp_args:NNx \tl_set:Nn \l_tmpa_tl
    { \tl_tail:N \l_tmpa_tl }
  \tl_if_single:NTF \l_tmpa_tl {
    \exp_args:No \tl_if_head_is_group:nTF \l_tmpa_tl {
      \exp_after:wN \str_set:Nn \exp_after:wN
        \l_stex_get_symbol_uri_str \l_tmpa_tl
    }{
      % TODO
      % tail is not a single group
    }
  }{
    % TODO
    % tail is not a single group
  }
}
```

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

19.2 Notations

```
⟨@@=stex_notation⟩
```

`notation` arguments:

```
\keys_define:nn { stex / notation } {
  lang    .tl_set_x:N = \l__stex_notation_lang_str ,
  variant .tl_set_x:N = \l__stex_notation_variant_str ,
  prec    .tl_set_x:N = \l__stex_notation_prec_str ,
  op      .tl_set:N   = \l__stex_notation_op_tl ,
  unknown .code:n     = \str_set:Nx
      \l__stex_notation_variant_str \l_keys_key_str
}

\cs_new_protected:Nn \__stex_notation_args:n {
  \str_clear:N \l__stex_notation_lang_str
  \str_clear:N \l__stex_notation_variant_str
  \str_clear:N \l__stex_notation_prec_str
  \tl_clear:N \l__stex_notation_op_tl

  \keys_set:nn { stex / notation } { #1 }
}
```

\notation

```
\NewDocumentCommand \notation { O{} m } {
  \__stex_notation_args:n { #1 }
  \tl_clear:N \l_stex_symdecl_definiens_tl
  \stex_get_symbol:n { #2 }
  \stex_notation_do:nn { \l_stex_get_symbol_uri_str }
}
\stex_deactivate_macro:Nn \notation {module~environments}
```

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

\stex_notation_do:nn

```
\cs_new_protected:Nn \stex_notation_do:nn {
  \prop_set_eq:Nc \l_tmpa_prop {
    g_stex_symdecl_ #1 _prop
  }

  \prop_clear:N \l_tmpb_prop
  \prop_put:Nno \l_tmpb_prop { symbol } { #1 }
  \prop_put:Nno \l_tmpb_prop { language } \l__stex_notation_lang_str
  \prop_put:Nno \l_tmpb_prop { variant } \l__stex_notation_variant_str

  % precedences
  \seq_clear:N \l_tmpb_seq
  \exp_args:NNno
  \str_if_empty:NTF \l__stex_notation_prec_str {
    \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
    \int_compare:nNnTF \l_tmpa_str = 0 {
      \exp_args:NNnx
      \prop_put:Nno \l_tmpb_prop { opprec }
        { \neginfprec }
    }{
      \prop_put:Nnn \l_tmpb_prop { opprec } { 0 }
    }
  } {
    \str_if_eq:onTF \l__stex_notation_prec_str {nobrackets}{
      \exp_args:NNnx
      \prop_put:Nno \l_tmpb_prop { opprec }
        { \neginfprec }
      \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
      \int_step_inline:nn { \l_tmpa_str } {
        \exp_args:NNx
        \seq_put_right:Nn \l_tmpb_seq { \infprec }
      }
    }{
      \seq_set_split:NnV \l_tmpa_seq ; \l__stex_notation_prec_str
      \seq_pop_left:NNTF \l_tmpa_seq \l_tmpa_str {
        \prop_put:Nno \l_tmpb_prop { opprec } \l_tmpa_str
        \seq_pop_left:NNT \l_tmpa_seq \l_tmpa_str {
          \exp_args:NNNo \exp_args:NNno \seq_set_split:Nnn
            \l_tmpa_seq {\tl_to_str:n{x} } { \l_tmpa_str }
          \seq_map_inline:Nn \l_tmpa_seq {
            \seq_put_right:Nn \l_tmpb_seq { ##1 }
          }
        }
        \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
      }{
        \prop_get:NnN \l_tmpa_prop { arity } \l_tmpa_str
        \int_compare:nNnTF \l_tmpa_str = 0 {
          \exp_args:NNnx
          \prop_put:Nno \l_tmpb_prop { opprec }
            { \infprec }
        }{
          \prop_put:Nnn \l_tmpb_prop { opprec } { 0 }
        }
      }
    }
  }

  \seq_set_eq:NN \l_tmpa_seq \l_tmpb_seq
  \int_step_inline:nn { \l_tmpa_str } {
    \seq_pop_left:NNF \l_tmpa_seq \l_tmpb_str {
      \exp_args:NNx
      \seq_put_right:Nn \l_tmpb_seq {
        \prop_item:Nn \l_tmpb_prop { opprec }
      }
    }
  }

  \prop_put:Nno \l_tmpb_prop { argprecs } \l_tmpb_seq
  \tl_clear:N \l_tmpa_tl

  \int_compare:nNnTF \l_tmpa_str = 0 {
    \exp_args:NNe
    \cs_set:Npn \l__stex_notation_macrocode_cs {
      \_stex_term_math_oms:nnnn { #1 }
        { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
        { \prop_item:Nn \l_tmpb_prop { opprec } }
        { \exp_not:n { #2 } }
    }
    \__stex_notation_final:
  }{
    \prop_get:NnN \l_tmpa_prop { args } \l_tmpb_str
    \str_if_in:NnTF \l_tmpb_str b {
      \exp_args:Nne \use:nn
      {
      \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
      \cs_set:Npn \l_tmpa_str } { {
        \_stex_term_math_omb:nnnn { #1 }
          { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
          { \prop_item:Nn \l_tmpb_prop { opprec } }
          { \exp_not:n { #2 } }
      }}
    }{
      \str_if_in:NnTF \l_tmpb_str B {
        \exp_args:Nne \use:nn
        {
        \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
        \cs_set:Npn \l_tmpa_str } { {
          \_stex_term_math_omb:nnnn { #1 }
            { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
            { \prop_item:Nn \l_tmpb_prop { opprec } }
            { \exp_not:n { #2 } }
        } }
      }{
        \exp_args:Nne \use:nn
        {
        \cs_generate_from_arg_count:NNnn \l__stex_notation_macrocode_cs
        \cs_set:Npn \l_tmpa_str } { {
          \_stex_term_math_oma:nnnn { #1 }
            { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }
            { \prop_item:Nn \l_tmpb_prop { opprec } }
            { \exp_not:n { #2 } }
        } }
      }
    }

    \int_zero:N \l_tmpa_int
    \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
    \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
    \__stex_notation_arguments:
  }
}
```

(*End definition for* `\stex_notation_do:nn`. *This function is documented on page 26.*)

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

```
\cs_new_protected:Nn \__stex_notation_arguments: {
  \int_incr:N \l_tmpa_int
  \str_if_empty:NTF \l_tmpa_str {
    \__stex_notation_final:
  }{
    \str_set:Nx \l_tmpb_str { \str_head:N \l_tmpa_str }
    \str_set:Nx \l_tmpa_str { \str_tail:N \l_tmpa_str }
    \str_if_eq:VnTF \l_tmpb_str a {
      \__stex_notation_argument_assoc:n
    }{
      \str_if_eq:VnTF \l_tmpb_str B {
        \__stex_notation_argument_assoc:n
      }{
        \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
        \tl_put_right:Nx \l_tmpa_tl {
          { \_stex_term_math_arg:nnn
            { \int_use:N \l_tmpa_int }
            { \l_tmpb_str }
            { ####\int_use:N \l_tmpa_int }
          }
        }
        \__stex_notation_arguments:
      }
    }
  }
}
```

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

__stex_notation_argument_assoc:n

```
\cs_new_protected:Nn \__stex_notation_argument_assoc:n {
  \seq_pop_left:NN \l_tmpa_seq \l_tmpb_str
  \cs_set:Npn \l_tmpa_cs ##1 ##2 { #1 }
  \tl_put_right:Nx \l_tmpa_tl {
    { \_stex_term_math_assoc_arg:nnnn
      { \int_use:N \l_tmpa_int }
      { \l_tmpb_str }
      \exp_args:No \exp_not:n
      {\exp_after:wN { \l_tmpa_cs {####1} {####2} } }
      { ####\int_use:N \l_tmpa_int }
    }
  }
  \__stex_notation_arguments:
}
```

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

__stex_notation_final: Called after processing all notation arguments

```
\cs_new_protected:Nn \__stex_notation_final: {
  \prop_get:NnN \l_tmpa_prop { arity } \l_tmpb_str
  \prop_get:NnN \l_tmpb_prop { symbol } \l_tmpa_str
  \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
  \exp_args:Nne \use:nn
  {
  \cs_generate_from_arg_count:cNnn {
      stex_notation_ \l_tmpa_str \c_hash_str
      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
      _cs
    }
    \cs_gset:Npn \l_tmpb_str } { {
      \exp_after:wN \exp_after:wN \exp_after:wN
      \exp_not:n \exp_after:wN \exp_after:wN \exp_after:wN
      { \exp_after:wN \l__stex_notation_macrocode_cs \l_tmpa_tl }
  } }

  \tl_if_empty:NF \l__stex_notation_op_tl {
    \cs_gset:cpx {
      stex_op_notation_ \l_tmpa_str \c_hash_str
      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
      _cs
    } {
      \_stex_term_oms:nnn {
        \l_tmpa_str \c_hash_str \l__stex_notation_variant_str \c_hash_str
        \l__stex_notation_lang_str
      }{
        \l_tmpa_str
      }{ \comp{ \exp_args:No \exp_not:n { \l__stex_notation_op_tl } } }
    }
  }



  \stex_debug:nn{symbols}{
    Notation~\l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
    ~for~\prop_item:Nn \l_tmpb_prop { symbol }^^J
    Operator~precedence:~
      \prop_item:Nn \l_tmpb_prop { opprec }^^J
    Argument~precedences:~
      \seq_use:Nn \l_tmpa_seq {,~}^^J
    Notation: \cs_meaning:c {
      stex_notation_ \l_tmpa_str \c_hash_str
      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
      _cs
    }
  }

  \prop_gset_eq:cN {
    g_stex_notation_ \l_tmpa_str \c_hash_str \l__stex_notation_variant_str
      \c_hash_str \l__stex_notation_lang_str _prop
  } \l_tmpb_prop

  \exp_args:Nx
  \stex_add_to_current_module:n {
    \prop_get:cnN {
      g_stex_symdecl_
        \prop_item:Nn \l_tmpb_prop { symbol }
      _prop
    } { notations } \exp_not:N \l_tmpa_seq
    \seq_put_right:Nn \exp_not:N \l_tmpa_seq {
      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
    }
    \prop_put:cno {
      g_stex_symdecl_
        \prop_item:Nn \l_tmpb_prop { symbol }
      _prop
    } { notations } \exp_not:N \l_tmpa_seq
  }

  \stex_if_smsmode:TF {
    \stex_smsmode_set_codes:
    \exp_args:Nx \stex_add_to_sms:n {
      \prop_gset_from_keyval:cn {
        g_stex_notation_ \l_tmpa_str \c_hash_str \l__stex_notation_variant_str
          \c_hash_str \l__stex_notation_lang_str _prop
      } {
        symbol    = \prop_item:Nn \l_tmpb_prop { symbol }     ,
        language  = \prop_item:Nn \l_tmpb_prop { language }   ,
        variant   = \prop_item:Nn \l_tmpb_prop { variant }    ,
        opprec    = \prop_item:Nn \l_tmpb_prop { opprec }     ,
        argprecs  = \prop_item:Nn \l_tmpb_prop { argprecs }   ,
      }
    }
  }{
    \prop_get:NnN \l_tmpa_prop { notations } \l_tmpa_seq
    \seq_put_right:Nx \l_tmpa_seq {
      \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str
    }
    \prop_put:Nno \l_tmpa_prop { notations } \l_tmpa_seq
    \prop_set_eq:cN {
      g_stex_symdecl_ \l_tmpa_str _prop
    } \l_tmpa_prop

    % HTML annotations
    \stex_if_do_html:T {
      \stex_annotate_invisible:nnn { notation }
      { \prop_item:Nn \l_tmpb_prop { symbol } } {
        \stex_annotate_invisible:nnn { notationfragment }
          { \l__stex_notation_variant_str \c_hash_str \l__stex_notation_lang_str }{}
        \prop_get:NnN \l_tmpb_prop { argprecs } \l_tmpa_seq
        \stex_annotate_invisible:nnn { precedence }
          { \prop_item:Nn \l_tmpb_prop { opprec };
            \seq_use:Nn \l_tmpa_seq { x }
          }{}

        \int_zero:N \l_tmpa_int
        \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
        \tl_clear:N \l_tmpa_tl
        \int_step_inline:nn { \prop_item:Nn \l_tmpa_prop { arity } }{
          \int_incr:N \l_tmpa_int
          \str_set:Nx \l_tmpb_str { \str_head:N \l_tmpa_str }
          \str_set:Nx \l_tmpa_str { \str_tail:N \l_tmpa_str }
          \str_if_eq:VnTF \l_tmpb_str a {
            \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
              \c_hash_str \c_hash_str \int_use:N \l_tmpa_int a ,
              \c_hash_str \c_hash_str \int_use:N \l_tmpa_int b
            } }
          }{
            \str_if_eq:VnTF \l_tmpb_str B {
              \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
                \c_hash_str \c_hash_str \int_use:N \l_tmpa_int a ,
                \c_hash_str \c_hash_str \int_use:N \l_tmpa_int b
              } }
            }{
              \tl_set:Nx \l_tmpa_tl { \l_tmpa_tl {
                \c_hash_str \c_hash_str \int_use:N \l_tmpa_int
              } }
            }
          }
        }
        \stex_annotate_invisible:nnn { notationcomp }{}{
          $ \exp_args:Nno \use:nn { \use:c {
            stex_notation_ \prop_item:Nn \l_tmpb_prop { symbol }
```

```
            \c_hash_str \l__stex_notation_variant_str
            \c_hash_str \l__stex_notation_lang_str _cs
          } } { \l_tmpa_tl } $
        }
      }
    }
  }
}
```

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

\symdef

```
\keys_define:nn { stex / symdef } {
  name    .str_set_x:N = \l_stex_symdecl_name_str ,
  local   .bool_set:N  = \l_stex_symdecl_local_bool ,
  args    .str_set_x:N = \l_stex_symdecl_args_str ,
  type    .tl_set:N    = \l_stex_symdecl_type_tl ,
  def     .tl_set:N    = \l_stex_symdecl_definiens_tl ,
  op      .tl_set:N    = \l__stex_notation_op_tl ,
  lang    .str_set_x:N = \l__stex_notation_lang_str ,
  variant .str_set_x:N = \l__stex_notation_variant_str ,
  prec    .str_set_x:N = \l__stex_notation_prec_str ,
  unknown .code:n      = \str_set:Nx
      \l__stex_notation_variant_str \l_keys_key_str
}

\cs_new_protected:Nn \__stex_notation_symdef_args:n {
  \str_clear:N \l_stex_symdecl_name_str
  \str_clear:N \l_stex_symdecl_args_str
  \bool_set_false:N \l_stex_symdecl_local_bool
  \tl_clear:N \l_stex_symdecl_type_tl
  \tl_clear:N \l_stex_symdecl_definiens_tl
  \str_clear:N \l__stex_notation_lang_str
  \str_clear:N \l__stex_notation_variant_str
  \str_clear:N \l__stex_notation_prec_str
  \tl_clear:N \l__stex_notation_op_tl

  \keys_set:nn { stex / symdef } { #1 }
}

\NewDocumentCommand \symdef { O{} m } {
  \__stex_notation_symdef_args:n { #1 }
  \bool_set_true:N \l_stex_symdecl_make_macro_bool
  \stex_symdecl_do:n { #2 }
  \exp_args:Nx \stex_notation_do:nn {
    \prop_item:Nn \l_tmpa_prop { module } ?
    \prop_item:Nn \l_tmpa_prop { name }
  }
}
\stex_deactivate_macro:Nn \symdef {module~environments}
```

(*End definition for* `\symdef`. *This function is documented on page 26.*)

```
⟨/package⟩
```

Chapter 20

sTEX -Terms Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   terms.dtx   %%%%%%%%%%%%%

⟨@@=stex_terms⟩
```

Warnings and error messages

```
\msg_new:nnn{stex}{error/nonotation}{
  Symbol~#1~invoked,~but~has~no~notation#2!
}
\msg_new:nnn{stex}{error/notationarg}{
  Error~in~parsing~notation~#1
}

```

20.1 Symbol Invokations

Arguments:

```
\keys_define:nn { stex / terms } {
  lang    .tl_set_x:N = \l__stex_terms_lang_str ,
  variant .tl_set_x:N = \l__stex_terms_variant_str ,
  unknown .code:n     = \str_set:Nx
      \l__stex_terms_variant_str \l_keys_key_str
}

\cs_new_protected:Nn \__stex_terms_args:n {
  \str_clear:N \l__stex_terms_lang_str
  \str_clear:N \l__stex_terms_variant_str
  \str_clear:N \l__stex_terms_prec_str
  \tl_clear:N \l__stex_terms_op_tl

  \keys_set:nn { stex / terms } { #1 }
}
```

\stex_invoke_symbol:n Invokes a semantic macro

```
\cs_new_protected:Nn \stex_invoke_symbol:n {
  \if_mode_math:
    \exp_after:wN \__stex_terms_invoke_math:n
  \else:
    \exp_after:wN \__stex_terms_invoke_text:n
  \fi: { #1 }
}
```

(*End definition for* `\stex_invoke_symbol:n`. *This function is documented on page 27.*)

`\__stex_terms_invoke_math:n`

```
\cs_new_protected:Nn \__stex_terms_invoke_math:n {
  \peek_charcode_remove:NTF ! {
    \peek_charcode:NTF [ {
      \__stex_terms_invoke_op:nw { #1 }
    }{
      \__stex_terms_invoke_op:nw { #1 } []
    }
  }{
    \peek_charcode_remove:NTF * {
      \__stex_terms_invoke_text:n { #1 }
    }{
      \peek_charcode:NTF [ {
        \__stex_terms_invoke_math:nw { #1 }
      }{
        \__stex_terms_invoke_math:nw { #1 } []
      }
    }
  }
}
```

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

`\__stex_terms_invoke_op:nw`

```
\cs_new_protected:Npn \__stex_terms_invoke_op:nw  #1 [#2] {
  \__stex_terms_args:n { #2 }
  \cs_if_exist:cTF {
    stex_op_notation_ #1 \c_hash_str
    \l__stex_terms_variant_str \c_hash_str \l__stex_terms_lang_str _cs
  }{
    \csname stex_op_notation_ #1 \c_hash_str
      \l__stex_terms_variant_str \c_hash_str \l__stex_terms_lang_str _cs
    \endcsname
  }{
    % TODO throw error
  }
}
```

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

`\__stex_terms_invoke_math:nw`

```
\cs_new_protected:Npn \__stex_terms_invoke_math:nw  #1 [#2] {
  \__stex_terms_args:n { #2 }
  \prop_set_eq:Nc \l_tmpa_prop {
    g_stex_symdecl_ #1 _prop
  }
  \prop_get:NnN \l_tmpa_prop { notations } \l_tmpa_seq
  \seq_if_empty:NTF \l_tmpa_seq {
    \msg_error:nnnn{stex}{error/nonotation}{#1}{s}
  } {
    \seq_if_in:NxTF \l_tmpa_seq
      { \l__stex_terms_variant_str \c_hash_str \l__stex_terms_lang_str }{
      \use:c{
        stex_notation_ #1 \c_hash_str
        \l__stex_terms_variant_str \c_hash_str \l__stex_terms_lang_str
        _cs
      }
    }{
      \str_if_empty:NTF \l__stex_terms_variant_str {
        \str_if_empty:NTF \l__stex_terms_lang_str {
          \seq_get_left:NN \l_tmpa_seq \l_tmpa_str
          \use:c{
            stex_notation_ #1 \c_hash_str \l_tmpa_str
            _cs
          }
        }{
          \msg_error:nn{stex}{error/nonotation}{#1}{
            ~\l__stex_terms_variant_str \c_hash_str \l__stex_terms_lang_str
          }
        }
      }{
        \msg_error:nn{stex}{error/nonotation}{#1}{
          ~\l__stex_terms_variant_str \c_hash_str \l__stex_terms_lang_str
        }
      }
    }
  }
}
```

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

`\__stex_terms_invoke_text:n`

```
\cs_new_protected:Nn \__stex_terms_invoke_text:n {
  \peek_charcode_remove:NTF ! {
    \stex_term_custom:nn { #1 } { }
  }{
    \prop_set_eq:Nc \l_tmpa_prop {
      g_stex_symdecl_ #1 _prop
    }
    \prop_get:NnN \l_tmpa_prop { args } \l_tmpa_str
    \exp_args:Nnx \stex_term_custom:nn { #1 } { \l_tmpa_str }
  }
}
```

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

20.2 Terms

Precedences:

\infprec
\neginfprec
\l__stex_terms_downprec

```
\tl_const:Nx \infprec {\int_use:N \c_max_int}
\tl_const:Nx \neginfprec {-\int_use:N \c_max_int}
\int_new:N \l__stex_terms_downprec
\int_set_eq:NN \l__stex_terms_downprec \infprec
```

(*End definition for* `\infprec`*,* `\neginfprec`*, and* `\l__stex_terms_downprec`*. These variables are documented on page 28.*)

Bracketing:

\l__stex_terms_left_bracket_str
\l__stex_terms_right_bracket_str

```
\tl_set:Nn \l__stex_terms_left_bracket_str (
\tl_set:Nn \l__stex_terms_right_bracket_str )
```

(*End definition for* `\l__stex_terms_left_bracket_str` *and* `\l__stex_terms_right_bracket_str`*.*)

__stex_terms_maybe_brackets:nn

Compares precedences and insert brackets accordingly

```
\cs_new_protected:Nn \__stex_terms_maybe_brackets:nn {
  \int_compare:nNnTF { #1 } > \l__stex_terms_downprec {
    \bool_if:NTF \l_stex_inparray_bool { #2 }{
      \dobrackets { #2 }
    }
  }{ #2 }
}
```

(*End definition for* `\__stex_terms_maybe_brackets:nn`*.*)

\dobrackets

```
%\RequirePackage{scalerel}
\cs_new_protected:Npn \dobrackets #1 {
  %\ThisStyle{\if D\m@switch
  %     \exp_args:Nnx \use:nn
  %     { \exp_after:wN \left\l__stex_terms_left_bracket_str #1 }
  %     { \exp_not:N\right\l__stex_terms_right_bracket_str }
  %   \else
      \exp_args:Nnx \use:nn
      { \l__stex_terms_left_bracket_str #1 }
      { \l__stex_terms_right_bracket_str }
  %\fi}
}
```

(*End definition for* `\dobrackets`*. This function is documented on page 28.*)

\withbrackets

```
\cs_new_protected:Npn \withbrackets #1 #2 #3 {
  \exp_args:Nnx \use:nn
  {
    \tl_set:Nx \l__stex_terms_left_bracket_str { #1 }
    \tl_set:Nx \l__stex_terms_right_bracket_str { #2 }
    #3
  }
  {
    \tl_set:Nn \exp_not:N \l__stex_terms_left_bracket_str
      {\l__stex_terms_left_bracket_str}
    \tl_set:Nn \exp_not:N \l__stex_terms_right_bracket_str
```

```
      {\l__stex_terms_right_bracket_str}
  }
}
```

(*End definition for* `\withbrackets`*. This function is documented on page 28.*)

\STEXinvisible

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

(*End definition for* `\STEXinvisible`*. This function is documented on page 29.*)

OMDoc terms:

_stex_term_math_oms:nnnn

```
\cs_new_protected:Nn \_stex_term_oms:nnn {
  \stex_annotate:nnn{ OMID }{ #2 }{
    \stex_highlight_term:nn { #1 } { #3 }
  }
}

\cs_new_protected:Nn \_stex_term_math_oms:nnnn {
  \__stex_terms_maybe_brackets:nn { #3 }{
    \_stex_term_oms:nnn { #1 } { #1\c_hash_str#2 } { #4 }
  }
}
```

(*End definition for* `\_stex_term_math_oms:nnnn`*. This function is documented on page 27.*)

_stex_term_math_oma:nnnn

```
\cs_new_protected:Nn \_stex_term_oma:nnn {
  \stex_annotate:nnn{ OMA }{ #2 }{
    \stex_highlight_term:nn { #1 } { #3 }
  }
}

\cs_new_protected:Nn \_stex_term_math_oma:nnnn {
  \__stex_terms_maybe_brackets:nn { #3 }{
    \_stex_term_oma:nnn { #1 } { #1\c_hash_str#2 } { #4 }
  }
}
```

(*End definition for* `\_stex_term_math_oma:nnnn`*. This function is documented on page 27.*)

_stex_term_math_omb:nnnn

```
\cs_new_protected:Nn \_stex_term_ombind:nnn {
  \stex_annotate:nnn{ OMBIND }{ #2 }{
    \stex_highlight_term:nn { #1 } { #3 }
  }
}

\cs_new_protected:Nn \_stex_term_math_omb:nnnn {
  \__stex_terms_maybe_brackets:nn { #3 }{
    \_stex_term_ombind:nnn { #1 } { #1\c_hash_str#2 } { #4 }
  }
}
```

(*End definition for* `\_stex_term_math_omb:nnnn`*. This function is documented on page 27.*)

`\_stex_term_math_arg:nnn`

```
\cs_new_protected:Nn \_stex_term_arg:nn {
  \stex_unhighlight_term:n {
    \stex_annotate:nnn{ arg }{ #1 }{ #2 }
  }
}
\cs_new_protected:Nn \_stex_term_math_arg:nnn {
  \exp_args:Nnx \use:nn
    { \int_set:Nn \l__stex_terms_downprec { #2 }
        \_stex_term_arg:nn { #1 }{ #3 }
    }
    { \int_set:Nn \exp_not:N \l__stex_terms_downprec { \int_use:N \l__stex_terms_downprec }
}
```

(*End definition for* `\_stex_term_math_arg:nnn`*. This function is documented on page 27.*)

`\_stex_term_math_assoc_arg:nnnn`

```
\cs_new_protected:Nn \_stex_term_math_assoc_arg:nnnn {
  \seq_set_split:Nnn \l_tmpa_seq , { #4 }
  \int_compare:nNnTF { \seq_count:N \l_tmpa_seq } < 2 {
    \tl_set:Nn \l_tmpa_tl { #4 }
  }{
    \cs_set:Npn \l_tmpa_cs ##1 ##2 { #3 }
    \seq_reverse:N \l_tmpa_seq
    \seq_pop_left:NN \l_tmpa_seq \l_tmpb_tl
    \tl_set:No \l_tmpa_tl { \l_tmpb_tl }

    \seq_map_inline:Nn \l_tmpa_seq {
      \exp_args:NNo \tl_set:No \l_tmpa_tl {
        \exp_args:Nno
        \l_tmpa_cs { ##1 } \l_tmpa_tl
      }
    }

  }
  \exp_args:Nnno
  \_stex_term_math_arg:nnn{#1}{#2}\l_tmpa_tl
}
```

(*End definition for* `\_stex_term_math_assoc_arg:nnnn`*. This function is documented on page 27.*)

`\stex_term_custom:nn`

```
\cs_new_protected:Nn \stex_term_custom:nn {
  \str_set:Nn \l__stex_terms_custom_uri { #1 }
  \str_set:Nn \l_tmpa_str { #2 }
  \tl_clear:N \l_tmpa_tl
  \int_zero:N \l_tmpa_int
  \int_set:Nn \l_tmpb_int { \str_count:N \l_tmpa_str }
  \__stex_terms_custom_loop:
}
```

(*End definition for* `\stex_term_custom:nn`*. This function is documented on page 29.*)

__stex_terms_custom_loop:

```
\cs_new_protected:Nn \__stex_terms_custom_loop: {
  \bool_set_false:N \l_tmpa_bool
  \bool_while_do:nn {
    \str_if_eq_p:ee X {
      \str_item:Nn \l_tmpa_str { \l_tmpa_int + 1 }
    }
  }{
    \int_incr:N \l_tmpa_int
  }

  \peek_charcode:NTF [ {
    % notation/text component
    \__stex_terms_custom_component:w
  } {
    \int_compare:nNnTF \l_tmpa_int = \l_tmpb_int {
      % all arguments read => finish
      \__stex_terms_custom_final:
    } {
      % arguments missing
      \peek_charcode_remove:NTF * {
        % invisible, specific argument position or both
        \peek_charcode:NTF [ {
          % visible specific argument position
          \__stex_terms_custom_arg:wn
        } {
          % invisible
          \peek_charcode_remove:NTF * {
            % invisible specific argument position
            \__stex_terms_custom_arg_inv:wn
          } {
            % invisible next argument
            \__stex_terms_custom_arg_inv:wn [ \l_tmpa_int + 1 ]
          }
        }
      } {
        % next normal argument
        \__stex_terms_custom_arg:wn [ \l_tmpa_int + 1 ]
      }
    }
  }
}
```

(*End definition for* __stex_terms_custom_loop:.)

__stex_terms_custom_arg_inv:wn

```
\cs_new_protected:Npn \__stex_terms_custom_arg_inv:wn [ #1 ] #2 {
  \bool_set_true:N \l_tmpa_bool
  \__stex_terms_custom_arg:wn [ #1 ] { #2 }
}
```

(*End definition for* __stex_terms_custom_arg_inv:wn.)

__stex_terms_custom_arg:wn

```
\cs_new_protected:Npn \__stex_terms_custom_arg:wn [ #1 ] #2 {
  \str_set:Nx \l_tmpb_str {
    \str_item:Nn \l_tmpa_str { #1 }
  }
  \str_case:VnTF \l_tmpb_str {
    { X } {
      \msg_error:nnn{stex}{error/notationarg}{\l__stex_terms_custom_uri}
    }
    { i } { \__stex_terms_custom_set_X:n { #1 } }
    { b } { \__stex_terms_custom_set_X:n { #1 } }
    { a } { \__stex_terms_custom_set_X:n { #1 } } % TODO ?
    { B } { \__stex_terms_custom_set_X:n { #1 } } % TODO ?
  }{}{
    \msg_error:nnn{stex}{error/notationarg}{\l__stex_terms_custom_uri}
  }

  \bool_if:nTF \l_tmpa_bool {
    \tl_put_right:Nx \l_tmpa_tl {
      \stex_annotate_invisible:n {
        \_stex_term_arg:nn { \int_eval:n { #1 } }
          \exp_not:n { { #2 } }
      }
    }
  } {
    \tl_put_right:Nx \l_tmpa_tl {
      \_stex_term_arg:nn { \int_eval:n { #1 } }
        \exp_not:n { { #2 } }
    }
  }

  \__stex_terms_custom_loop:
}
```

(*End definition for* `\__stex_terms_custom_arg:wn`.)

__stex_terms_custom_set_X:n

```
\cs_new_protected:Nn \__stex_terms_custom_set_X:n {
  \str_set:Nx \l_tmpa_str {
    \str_range:Nnn \l_tmpa_str 1 { #1 - 1 }
    X
    \str_range:Nnn \l_tmpa_str { #1 + 1 } { -1 }
  }
}
```

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

__stex_terms_custom_component:

```
\cs_new_protected:Npn \__stex_terms_custom_component:w [ #1 ] {
  \tl_put_right:Nn \l_tmpa_tl { \comp{ #1 } }
  \__stex_terms_custom_loop:
}
```

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

__stex_terms_custom_final:

```
\cs_new_protected:Nn \__stex_terms_custom_final: {
  \int_compare:nNnTF \l_tmpb_int = 0 {
    \exp_args:Nnno \_stex_term_oms:nnn
  }{
    \str_if_in:NnTF \l_tmpa_str {b} {
      \exp_args:Nnno \_stex_term_ombind:nnn
    } {
      \exp_args:Nnno \_stex_term_oma:nnn
    }
  }
  { \l__stex_terms_custom_uri } { \l__stex_terms_custom_uri } { \l_tmpa_tl }
}
```

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

\symref
\symname

```
\NewDocumentCommand \symref { m m }{
  \let\compemph_uri_prev:\compemph@uri
  \let\compemph@uri\symrefemph@uri
  \STEXsymbol{#1}![#2]
  \let\compemph@uri\compemph_uri_prev:
}

\keys_define:nn { stex / symname } {
  post    .str_set_x:N   = \l_stex_symname_post_str
}

\cs_new_protected:Nn \stex_symname_args:n {
  \str_clear:N \l_stex_symname_post_str
  \keys_set:nn { stex / symname } { #1 }
}

\NewDocumentCommand \symname { O{} m }{
  \stex_symname_args:n { #1 }
  \stex_get_symbol:n { #2 }
  \str_set:Nx \l_tmpa_str {
    \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
  }
  \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}

  \let\compemph_uri_prev:\compemph@uri
  \let\compemph@uri\symrefemph@uri
  \exp_args:NNx \use:nn
  \stex_invoke_symbol:n { { \l_stex_get_symbol_uri_str }![
    \l_tmpa_str \l_stex_symname_post_str
  ] }
  \let\compemph@uri\compemph_uri_prev:
}
```

(*End definition for* `\symref` *and* `\symname`*. These functions are documented on page 27.*)

20.3 Notation Components

```
⟨@@=stex_notationcomps⟩
```

\stex_highlight_term:nn

```

\str_new:N \l__stex_notationcomps_highlight_uri_str
\cs_new_protected:Nn \stex_highlight_term:nn {
  \exp_args:Nnx
  \use:nn {
    \str_set:Nx \l__stex_notationcomps_highlight_uri_str { #1 }
    #2
  } {
    \str_set:Nx \exp_not:N \l__stex_notationcomps_highlight_uri_str
      { \l__stex_notationcomps_highlight_uri_str }
  }
}

\cs_new_protected:Nn \stex_unhighlight_term:n {
%  \latexml_if:TF {
%    #1
%  } {
%    \scalatex_if:TF {
%      #1
%    } {
      #1 %\iffalse{{\fi}} #1 {{\iffalse}}\fi
%    }
%  }
}
```

(*End definition for* `\stex_highlight_term:nn`. *This function is documented on page 29.*)

\comp
\compemph@uri
\compemph
\defemph
\defemph@uri
\symrefemph
\symrefemph@uri

```
\cs_new_protected:Npn \comp #1 {
  \str_if_empty:NF \l__stex_notationcomps_highlight_uri_str {
    \scalatex_if:TF {
      \stex_annotate:nnn { comp }{ \l__stex_notationcomps_highlight_uri_str }{ #1 }
    }{
      \exp_args:Nnx \compemph@uri { #1 } { \l__stex_notationcomps_highlight_uri_str }
    }
  }
}

\cs_new_protected:Npn \compemph@uri #1 #2 {
    \compemph{ #1 }
}


\cs_new_protected:Npn \compemph #1 {
    \textcolor{blue}{#1}
}

\cs_new_protected:Npn \defemph@uri #1 #2 {
    \defemph{#1}
}
```

```
\cs_new_protected:Npn \defemph #1 {
    \textbf{#1}
}

\cs_new_protected:Npn \symrefemph@uri #1 #2 {
    \symrefemph{#1}
}

\cs_new_protected:Npn \symrefemph #1 {
    \textbf{#1}
}
```

(*End definition for* `\comp` *and others. These functions are documented on page 29.*)

`\ellipses`

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

(*End definition for* `\ellipses`*. This function is documented on page 29.*)

`\parray`
`\prmatrix`
`\parrayline`
`\parraylineh`
`\parraycell`

```
\bool_new:N \l_stex_inparray_bool
\bool_set_false:N \l_stex_inparray_bool
\NewDocumentCommand \parray { m m } {
  \begingroup
  \bool_set_true:N \l_stex_inparray_bool
  \begin{array}{#1}
    #2
  \end{array}
  \endgroup
}

\NewDocumentCommand \prmatrix { m } {
  \begingroup
  \bool_set_true:N \l_stex_inparray_bool
  \begin{matrix}
    #1
  \end{matrix}
  \endgroup
}

\def \parrayline #1 #2 {
  #1 #2 \bool_if:NT \l_stex_inparray_bool {\\}
}

\def \parraylineh #1 #2 {
  #1 #2 \bool_if:NT \l_stex_inparray_bool {\\\hline}
}

\def \parraycell #1 {
  #1 \bool_if:NT \l_stex_inparray_bool {&}
}
```

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

```
⟨/package⟩
```

Chapter 21

sTeX -Structural Features Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   features.dtx   %%%%%%%%%%%%%

⟨@@=stex_features⟩
```

Warnings and error messages

symboldoc

```
\NewDocumentEnvironment{symboldoc}{ m }{
  \seq_set_split:Nnn \l_tmpa_seq , { #1 }
  \seq_clear:N \l_tmpb_seq
  \seq_map_inline:Nn \l_tmpa_seq {
    \stex_get_symbol:n { ##1 }
    \exp_args:NNo \seq_put_right:Nn \l_tmpb_seq {
      \l_stex_get_symbol_uri_str
    }
  }
  \par
  \exp_args:Nnnx
  \begin{stex_annotate_env}{symboldoc}{\seq_use:Nn \l_tmpb_seq {,}}
}{
  \end{stex_annotate_env}
}
```

STEXdefinition

```

\NewDocumentCommand \definiendum { O{} m m} {
  \stex_get_symbol:n { #2 }
  \stex_ref_new_sym_target:n \l_stex_get_symbol_uri_str
  \scalatex_if:TF {
    \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } { #3 }
  } {
    \exp_args:Nnx \defemph@uri { #3 } { \l_stex_get_symbol_uri_str }
  }
}
\stex_deactivate_macro:Nn \definiendum {definition~environments}
\NewDocumentCommand \definame { O{} m } {
  % TODO: root
  \stex_get_symbol:n { #2 }
  \stex_ref_new_sym_target:n \l_stex_get_symbol_uri_str
  \str_set:Nx \l_tmpa_str {
    \prop_item:cn { g_stex_symdecl_ \l_stex_get_symbol_uri_str _prop } { name }
  }
  \exp_args:NNno \str_replace_all:Nnn \l_tmpa_str {-} {~}
  \scalatex_if:TF {
    \stex_annotate:nnn { definiendum } { \l_stex_get_symbol_uri_str } {
      \l_tmpa_str
      }
  } {
    \defemph@uri {
      \l_tmpa_str
    } { \l_stex_get_symbol_uri_str }
  }
}
\stex_deactivate_macro:Nn \definame {definition~environments}

\cs_new_protected:Nn \__stex_features_defi_begin:n {
  \stex_reactivate_macro:N \definiendum
  \stex_reactivate_macro:N \definame
  \seq_set_split:Nnn \l_tmpa_seq , { #1 }
  \seq_clear:N \l_tmpb_seq
  \seq_map_inline:Nn \l_tmpa_seq {
    \stex_get_symbol:n { ##1 }
    \exp_args:NNo \seq_put_right:Nn \l_tmpb_seq {
      \l_stex_get_symbol_uri_str
    }
  }
  \exp_args:Nnnx
  \begin{stex_annotate_env}{definition}{\seq_use:Nn \l_tmpb_seq {,}}
}

\cs_new_protected:Nn \__stex_features_defi_end: {
  \end{stex_annotate_env}
}

\NewDocumentEnvironment{STEXdefinition}{ m }{
  \__stex_features_defi_begin:n { #1 }
}{
  \__stex_features_defi_end:
}
```

\setSTEXdefinition

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

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

`structural@feature`

```

\NewDocumentEnvironment{structural@feature}{ m m m }{
  \stex_if_in_module:F {
    \msg_set:nnn{stex}{error/nomodule}{
      Structural~Feature~has~to~occur~in~a~module:\\
      Feature~#2~of~type~#1\\
      In~File:~\stex_path_to_string:N \g_stex_currentfile_seq
    }
    \msg_error:nn{stex}{error/nomodule}
  }

  \str_set:Nx \l_stex_module_name_str {
    \prop_item:Nn \l_stex_current_module_prop
      { name } / #2 - feature
  }

  \str_set:Nx \l_stex_module_ns_str {
    \prop_item:Nn \l_stex_current_module_prop
      { ns }
  }


  \str_clear:N \l_tmpa_str
  \seq_clear:N \l_tmpa_seq
  \tl_clear:N \l_tmpa_tl
  \exp_args:NNx \prop_set_from_keyval:Nn \l_stex_current_module_prop {
    origname  = #2,
    name      = \l_stex_module_name_str ,
    ns        = \l_stex_module_ns_str ,
    imports   = \exp_not:o { \l_tmpa_seq } ,
    constants = \exp_not:o { \l_tmpa_seq } ,
    content   = \exp_not:o { \l_tmpa_tl }  ,
    file      = \exp_not:o { \g_stex_currentfile_seq } ,
    lang      = \l_stex_module_lang_str ,
    sig       = \l_tmpa_str ,
    meta      = \l_tmpa_str ,
    feature   = #1 ,
  }

  \stex_if_smsmode:TF {
    \stex_smsmode_set_codes:
  } {
    \begin{stex_annotate_env}{ feature:#1 }{}
      \stex_annotate_invisible:nnn{header}{}{ #3 }
  }
}{
  \str_set:Nx \l_tmpa_str {
    c_stex_feature_
    \prop_item:Nn \l_stex_current_module_prop { ns } ?
    \prop_item:Nn \l_stex_current_module_prop { name }
    _prop
  }
  \prop_gset_eq:cN { \l_tmpa_str } \l_stex_current_module_prop
  \prop_gset_eq:NN \g_stex_last_feature_prop \l_stex_current_module_prop
  \stex_if_smsmode:TF {
    \exp_args:Nx \stex_add_to_sms:n {
      \prop_gset_from_keyval:cn {
        c_stex_feature_
        \prop_item:Nn \l_stex_current_module_prop { ns } ?
        \prop_item:Nn \l_stex_current_module_prop { name }
        _prop
      } {
        origname  = #2,
        name      = \prop_item:cn { \l_tmpa_str } { name } ,
        ns        = \prop_item:cn { \l_tmpa_str } { ns } ,
        imports   = \prop_item:cn { \l_tmpa_str } { imports } ,
        constants = \prop_item:cn { \l_tmpa_str } { constants } ,
        content   = \prop_item:cn { \l_tmpa_str } { content } ,
        file      = \prop_item:cn { \l_tmpa_str } { file } ,
        lang      = \prop_item:cn { \l_tmpa_str } { lang } ,
        sig       = \prop_item:cn { \l_tmpa_str } { sig } ,
        meta      = \prop_item:cn { \l_tmpa_str } { meta } ,
        feature   = \prop_item:cn { \l_tmpa_str } { feature }
      }
    }
  } {
      \end{stex_annotate_env}
  }
}

```

structure

```

\prop_new:N \l_stex_all_structures_prop

\keys_define:nn { stex / features / structure } {
  name         .str_set_x:N  = \l__stex_features_structure_name_str ,
}

\cs_new_protected:Nn \__stex_features_structure_args:n {
  \str_clear:N \l__stex_features_structure_name_str
  \keys_set:nn { stex / features / structure } { #1 }
}

%\stex_new_feature:nnnn { structure } { O{} m } {
%  \__stex_features_structure_args:n { ##1 }
%  \str_if_empty:NT \l__stex_features_structure_name_str {
%    \str_set:Nx \l__stex_features_structure_name_str { ##2 }
%  }
%} {
%
%}

```

```
\NewDocumentEnvironment{structure}{ O{} m }{
  \__stex_features_structure_args:n { #1 }
  \str_if_empty:NT \l__stex_features_structure_name_str {
    \str_set:Nx \l__stex_features_structure_name_str { #2 }
  }
  \exp_args:Nnnx
  \begin{structural@feature}{ structure }
    { \l__stex_features_structure_name_str }{}
    \seq_clear:N \l_tmpa_seq
    \prop_put:Nno \l_stex_current_module_prop { fields } \l_tmpa_seq

}{
    \prop_get:NnN \l_stex_current_module_prop { constants } \l_tmpa_seq
    \prop_get:NnN \l_stex_current_module_prop { fields } \l_tmpb_seq
    \str_set:Nx \l_tmpa_str {
      \prop_item:Nn \l_stex_current_module_prop { ns } ?
      \prop_item:Nn \l_stex_current_module_prop { name }
    }
    \seq_map_inline:Nn \l_tmpa_seq {
      \exp_args:NNx \seq_put_right:Nn \l_tmpb_seq { \l_tmpa_str ? ##1 }
    }
    \prop_put:Nno \l_stex_current_module_prop { fields } { \l_tmpb_seq }
    \exp_args:Nnx
    \AddToHookNext { env / structure / after }{
      \symdecl[type = \exp_not:N\collection,def={\STEXsymbol{module-type}{
        \_stex_term_math_oms:nnnn { \l_tmpa_str }{}{0}{}
      }}, name = \prop_item:Nn \l_stex_current_module_prop { origname }]{ #2 }
      \STEXexport {
        \prop_put:Nno \exp_not:N \l_stex_all_structures_prop
          {\prop_item:Nn \l_stex_current_module_prop { origname }}
          {\l_tmpa_str}
          \prop_put:Nno \exp_not:N \l_stex_all_structures_prop
            {#2}{\l_tmpa_str}
%        \seq_put_right:Nn \exp_not:N \l_stex_all_structures_seq {
%          \prop_item:Nn \l_stex_current_module_prop { origname },
%          \l_tmpa_str
%        }
%        \seq_put_right:Nn \exp_not:N \l_stex_all_structures_seq {
%          #2,\l_tmpa_str
%        }
%        \tl_set:cx { #2 } {
%          \stex_invoke_structure:n { \l_tmpa_str }
      }
    }

  \end{structural@feature}
  % \g_stex_last_feature_prop
}
```

\instantiate

```
\seq_new:N \l__stex_features_structure_field_seq
\str_new:N \l__stex_features_structure_field_str
\str_new:N \l__stex_features_structure_def_tl
\prop_new:N \l__stex_features_structure_prop
```

```
\NewDocumentCommand \instantiate { m O{} m }{
  \stex_smsmode_set_codes:
  \prop_get:NnN \l_stex_all_structures_prop {#1} \l_tmpa_str
  \prop_set_eq:Nc \l__stex_features_structure_prop {
    c_stex_feature_\l_tmpa_str _prop
  }
  \seq_set_from_clist:Nn \l__stex_features_structure_field_seq { #2 }
  \seq_map_inline:Nn \l__stex_features_structure_field_seq {
    \seq_set_split:Nnn \l_tmpa_seq{=}{ ##1 }
    \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > 1 {
      \seq_get_left:NN \l_tmpa_seq \l_tmpa_tl
      \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq
        {!} \l_tmpa_tl
      \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} > 1 {
        \str_set:Nx \l__stex_features_structure_field_str {\seq_item:Nn \l_tmpb_seq 1}
        \seq_get_right:NN \l_tmpb_seq \l_tmpb_tl
        \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
      }{
        \str_set:Nx \l__stex_features_structure_field_str \l_tmpa_tl
        \seq_get_right:NN \l_tmpa_seq \l_tmpa_tl
        \exp_args:NNno \seq_set_split:Nnn \l_tmpb_seq{!}
          \l_tmpa_tl
        \int_compare:nNnTF {\seq_count:N \l_tmpb_seq} > 1 {
          \seq_get_left:NN \l_tmpb_seq \l_tmpa_tl
          \seq_get_right:NN \l_tmpb_seq \l_tmpb_tl
        }{
          \tl_clear:N \l_tmpb_tl
        }
      }
    }{
      \seq_set_split:Nnn \l_tmpa_seq{!}{ ##1 }
      \int_compare:nNnTF {\seq_count:N \l_tmpa_seq} > 1 {
        \str_set:Nx \l__stex_features_structure_field_str {\seq_item:Nn \l_tmpa_seq 1}
        \seq_get_right:NN \l_tmpa_seq \l_tmpb_tl
        \tl_clear:N \l_tmpa_tl
      }{
        % TODO throw error
      }
    }
    % \l_tmpa_str: name
    % \l_tmpa_tl: definiens
    % \l_tmpb_tl: notation
    \tl_if_empty:NT \l__stex_features_structure_field_str {
      % TODO throw error
    }
    \str_clear:N \l_tmpb_str

    \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
    \seq_map_inline:Nn \l_tmpa_seq {
      \seq_set_split:Nnn \l_tmpb_seq ? { ####1 }
      \seq_get_right:NN \l_tmpb_seq \l_tmpb_str
      \str_if_eq:NNT \l__stex_features_structure_field_str \l_tmpb_str {
        \seq_map_break:n {
          \str_set:Nn \l_tmpb_str { ####1 }
        }
      }
    }
    \prop_get:cnN { g_stex_symdecl_ \l_tmpb_str _prop } {args}
      \l_tmpb_str

    \tl_if_empty:NTF \l_tmpb_tl {
      \tl_if_empty:NF \l_tmpa_tl {
        \exp_args:Nx \use:n {
          \symdecl[args=\l_tmpb_str,def={\exp_args:No\exp_not:n{\l_tmpa_tl}}]{#3/\l__stex_fe
        }
      }
    }{
      \tl_if_empty:NTF \l_tmpa_tl {
        \exp_args:Nx \use:n {
          \symdef[args=\l_tmpb_str]{#3/\l__stex_features_structure_field_str}\exp_after:wN\e
        }

      }{
        \exp_args:Nx \use:n {
          \symdef[args=\l_tmpb_str,def={\exp_args:No\exp_not:n{\l_tmpa_tl}}]{#3/\l__stex_fea
          \exp_after:wN\exp_not:n\exp_after:wN{\l_tmpb_tl}
        }
      }
    }
%    \par \prop_item:Nn \l_stex_current_module_prop {ns} ?
%    \prop_item:Nn \l_stex_current_module_prop {name} ?
%    #3/\l__stex_features_structure_field_str
%    \par
%    \expandafter\present\csname
%      g_stex_symdecl_
%      \prop_item:Nn \l_stex_current_module_prop {ns} ?
%      \prop_item:Nn \l_stex_current_module_prop {name} ?
%      #3/\l__stex_features_structure_field_str
%      _prop
%    \endcsname
  }

  \tl_clear:N \l__stex_features_structure_def_tl

  \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
  \seq_map_inline:Nn \l_tmpa_seq {
    \seq_set_split:Nnn \l_tmpb_seq ? { ##1 }
    \seq_get_right:NN \l_tmpb_seq \l_tmpa_str
    \exp_args:Nx \use:n {
      \tl_put_right:Nn \exp_not:N \l__stex_features_structure_def_tl {

      }
    }

    \prop_if_exist:cF {
      g_stex_symdecl_
      \prop_item:Nn \l_stex_current_module_prop {ns} ?
      \prop_item:Nn \l_stex_current_module_prop {name} ?
      #3/\l_tmpa_str
      _prop
    }{
      \prop_get:cnN { g_stex_symdecl_ ##1 _prop } {args}
        \l_tmpb_str
      \exp_args:Nx \use:n {
        \symdecl[args=\l_tmpb_str]{#3/\l_tmpa_str}
      }
    }
  }

  \symdecl*[type={\STEXsymbol{module-type}{
    \_stex_term_math_oms:nnnn {
      \prop_item:Nn \l__stex_features_structure_prop {ns} ?
      \prop_item:Nn \l__stex_features_structure_prop {name}
      }{}{0}{}
  }}]{#3}

  % TODO: -> sms file

  \tl_set:cx{ #3 }{
    \stex_invoke_structure:nnn {
      \prop_item:Nn \l_stex_current_module_prop {ns} ?
      \prop_item:Nn \l_stex_current_module_prop {name} ? #3
    } {
      \prop_item:Nn \l__stex_features_structure_prop {ns} ?
      \prop_item:Nn \l__stex_features_structure_prop {name}
    }
  }

}
```

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

`\stex_invoke_structure:nnn`

```
% #1: URI of the instance
% #2: URI of the instantiated module
\cs_new_protected:Nn \stex_invoke_structure:nnn {
  \tl_if_empty:nTF{ #3 }{
    \prop_set_eq:Nc \l__stex_features_structure_prop {
      c_stex_feature_ #2 _prop
    }
    \tl_clear:N \l_tmpa_tl
    \prop_get:NnN \l__stex_features_structure_prop { fields } \l_tmpa_seq
    \seq_map_inline:Nn \l_tmpa_seq {
      \seq_set_split:Nnn \l_tmpb_seq ? { ##1 }
      \seq_get_right:NN \l_tmpb_seq \l_tmpa_str
      \cs_if_exist:cT {
        stex_notation_ #1/\l_tmpa_str \c_hash_str\c_hash_str _cs
      }{
        \tl_if_empty:NF \l_tmpa_tl {
          \tl_put_right:Nn \l_tmpa_tl {,}
        }
        \tl_put_right:Nx \l_tmpa_tl {
          \stex_invoke_symbol:n {#1/\l_tmpa_str}!
        }
      }
    }
    \exp_args:No \mathstruct \l_tmpa_tl
  }{
    \stex_invoke_symbol:n{#1/#3}
  }
}
```

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

```
⟨/package⟩
```

Some auxiliary code, and clean up to be executed at the end of the package.

Chapter 22

sTeX -Others Implementation

```
⟨*package⟩

%%%%%%%%%%%%%   others.dtx   %%%%%%%%%%%%%

⟨@@=stex_others⟩
```

Warnings and error messages

```
  % None
```

\MSC Math subject classifier

```
\NewDocumentCommand \MSC {m} {
  % TODO
}
```

(*End definition for* `\MSC`*. This function is documented on page 10.*)

Patching tikzinput, if loaded

```
\@ifpackageloaded{tikzinput}{
  \RequirePackage{stex-tikzinput}
}{}
⟨/package⟩
```

Chapter 23

sTeX -Metatheory Implementation

```
⟨*package⟩
⟨@@=stex_modules⟩

%%%%%%%%%%%%%   metatheory.dtx   %%%%%%%%%%%%%

\str_const:Nn \c_stex_metatheory_ns_str {http://mathhub.info/sTeX}
\begingroup
\stex_module_setup:nn{
  ns=\c_stex_metatheory_ns_str,
  meta=NONE
}{Metatheory}
\stex_reactivate_macro:N \symdecl
\stex_reactivate_macro:N \notation
\stex_reactivate_macro:N \symdef
\ExplSyntaxOff
\csname stex_suppress_html:n\endcsname{
  % is-a (a:A, a \in A, a is an A, etc.)
  \symdecl[args=ai]{isa}
  \notation[typed]{isa}{#1 \comp{:} #2}{#1 \comp, #2}
  \notation[in]{isa}{#1 \comp\in #2}{#1 \comp, #2}
  \notation[pred]{isa}{#2\comp(#1 \comp)}{#1 \comp, #2}

  % bind (\forall, \Pi, \lambda etc.)
  \symdecl[args=Bi]{bind}
  \notation[forall]{bind}{\comp\forall #1.\;#2}{#1 \comp, #2}
  \notation[Pi]{bind}{\comp\prod_{#1}#2}{#1 \comp, #2}
  \notation[depfun]{bind}{\comp( #1 \comp{)\;\to\;} #2}{#1 \comp, #2}

  % dummy variable
  \symdecl{dummyvar}
  \notation[underscore]{dummyvar}{\comp\_}
  \notation[dot]{dummyvar}{\comp\cdot}
  \notation[dash]{dummyvar}{\comp{{\rm --}}}

  %fromto (function space, Hom-set, implication etc.)
  \symdecl[args=ai]{fromto}
  \notation[xarrow]{fromto}{#1 \comp\to #2}{#1 \comp\times #2}
  \notation[arrow]{fromto}{#1 \comp\to #2}{#1 \comp\to #2}

  % mapto (lambda etc.)
  %\symdecl[args=Bi]{mapto}
  %\notation[mapsto]{mapto}{#1 \comp\mapsto #2}{#1 \comp, #2}
  %\notation[lambda]{mapto}{\comp\lambda #1 \comp.\; #2}{#1 \comp, #2}
  %\notation[lambdau]{mapto}{\comp\lambda_{#1} \comp.\; #2}{#1 \comp, #2}

  % function/operator application
  \symdecl[args=ia]{apply}
  \notation[prec=0;0x\neginfprec,parens]{apply}{#1 \comp( #2 \comp)}{#1 \comp, #2}
  \notation[prec=0;0x\neginfprec,lambda]{apply}{#1 \; #2 }{#1 \; #2}

  % ‘‘type’’ of all collections (sets,classes,types,kinds)
  \symdecl{collection}
  \notation[U]{collection}{\comp{\mathcal{U}}}
  \notation[set]{collection}{\comp{\textsf{Set}}}

  % sequences
  \symdecl[args=1]{seqtype}
  \notation[kleene]{seqtype}{#1^{\comp\ast}}

  \symdef[args=2,li]{sequence-index}{#1_{#2}}
  \notation[ui]{sequence-index}{#1^{#2}}

  %\symdef[args=3,li]{sequence-from-to}{#1_{#2}\comp{,\ellipses,}#1_{#3}}
  %\notation[ui]{sequence-from-to}{#1^{#2}\comp{,\ellipses,}#1^{#3}}
  % ^ superceded by \aseqfromto and \livar/\uivar

  \symdef[args=a,prec=nobrackets]{aseqdots}{#1\comp{,\ellipses}}{#1\comp,#2}
  \symdef[args=ai,prec=nobrackets]{aseqfromto}{#1\comp{,\ellipses\comp,}#2 }{#1\comp,#2}
  \symdef[args=aii,prec=nobrackets]{aseqfromtovia}{#1\comp{,\ellipses\comp,}#2\comp{,\ellips

  % letin (‘‘let’’, local definitions, variable substitution)
  \symdecl[args=bii]{letin}
  \notation[let]{letin}{\comp{{\rm let}}\;#1\comp{=}#2\;\comp{{\rm in}}\;#3}
  \notation[subst]{letin}{#3 \comp[ #1 \comp/ #2 \comp]}
  \notation[frac]{letin}{#3 \comp[ \frac{#2}{#1} \comp]}

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

}
  \ExplSyntaxOn
  \stex_add_to_current_module:n{
    \let\nappa\apply
    \def\nappli#1#2#3#4{\apply{#1}{\naseqli{#2}{#3}{#4}}}
    \def\livar{\csname sequence-index\endcsname[li]}
    \def\uivar{\csname sequence-index\endcsname[ui]}
    \def\naseqli#1#2#3{\aseqfromto{\livar{#1}{#2}}{\livar{#1}{#3}}}
    \def\nasequi#1#2#3{\aseqfromto{\uivar{#1}{#2}}{\uivar{#1}{#3}}}
  }
\__stex_modules_end_module:
\endgroup
```

⟨/package⟩

Chapter 24

Tikzinput Implementation

```
⟨*package⟩

%%%%%%%%%%%%   tikzinput.dtx   %%%%%%%%%%%%

\ProvidesExplPackage{tikzinput}{2021/08/31}{1.9}{bla}
\RequirePackage{l3keys2e}

\keys_define:nn { tikzinput } {
  image   .bool_set:N   = \c_tikzinput_image_bool,
  image   .default:n    = false ,
}

\ProcessKeysOptions { tikzinput }

\bool_if:NTF \c_tikzinput_image_bool {
  \RequirePackage{graphicx}

  \providecommand\usetikzlibrary[]{}
  \newcommand\tikzinput[2][]{\includegraphics[#1]{#2}}
}{
  \RequirePackage{tikz}
  \RequirePackage{standalone}

  \newcommand \tikzinput [2] [] {
    \setkeys{Gin}{#1}
    \ifx \Gin@ewidth \Gin@exclamation
      \ifx \Gin@eheight \Gin@exclamation
        \input { #2 }
      \else
        \resizebox{!}{ \Gin@eheight }{
          \input { #2 }
        }
      \fi
    \else
      \ifx \Gin@eheight \Gin@exclamation
        \resizebox{ \Gin@ewidth }{!}{
          \input { #2 }
        }
      \else
        \resizebox{ \Gin@ewidth }{ \Gin@eheight }{
          \input { #2 }
        }
      \fi
    \fi
  }
}

\newcommand \ctikzinput [2] [] {
  \begin{center}
    \tikzinput [#1] {#2}
  \end{center}
}

\@ifpackageloaded{stex}{
  \RequirePackage{stex-tikzinput}
}{}

⟨/package⟩
⟨*stex⟩
\ProvidesExplPackage{stex-tikzinput}{2021/08/31}{1.9}{bla}
\RequirePackage{stex}
\RequirePackage{tikzinput}

\newcommand\mhtikzinput[2][]{%
  \def\Gin@mhrepos{}\setkeys{Gin}{#1}%
  \stex_in_repository:nn\Gin@mhrepos{
    \tikzinput[#1]{\mhpath{##1}{#2}}
  }
}
\newcommand\cmhtikzinput[2][]{\begin{center}\mhtikzinput[#1]{#2}\end{center}}
⟨/stex⟩
```

LocalWords: bibfolder jobname.dtx tikzinput.dtx usetikzlibrary Gin@ewidth Gin@eheight
LocalWords: resizebox ctikzinput mhtikzinput Gin@mhrepos mhpath

Chapter 25

document-structure.sty Implementation

25.1 The OMDoc Class

The functionality is spread over the omdoc class and package. The class provides the document environment and the omdoc element corresponds to it, whereas the package provides the concrete functionality.

25.2 Class Options

To initialize the omdoc class, we declare and process the necessary options using the kvoptions package for key/value options handling. For omdoc.cls this is quite simple. We have options report and book, which set the \omdoc@cls@class macro and pass on the macro to omdoc.sty for further processing.

\omdoc@cls@class

```
⟨*cls⟩
\RequirePackage{etoolbox}
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=omdoc@cls,prefix=omdoc@cls@}
\DeclareStringOption[article]{class}
\AddToKeyvalOption*{class}{\PassOptionsToPackage{class=\omdoc@cls@class}{omdoc}}
```

the following options are deprecated.

```
\DeclareVoidOption{report}{\def\omdoc@cls@class{report}%
\ClassWarning{omdoc}{the option 'report' is deprecated, use 'class=report', instead}}
\DeclareVoidOption{book}{\def\omdoc@cls@class{book}%
\ClassWarning{omdoc}{the option 'part' is deprecated, use 'class=book', instead}}
\DeclareVoidOption{bookpart}{\def\omdoc@cls@class{book}%
\PassOptionsToPackage{topsect=chapter}{omdoc}%
\ClassWarning{omdoc}{the option 'bookpart' is deprecated, use 'class=book,topsect=chapter',
\DeclareBoolOption{minimal}
```

the rest of the options are only passed on to omdoc.sty and the class selected by the first options. We need to load the etoolbox package early for \@xappto.

```
\def\@omdoc@cls@docopt{}
\DeclareDefaultOption{%
\ifx\@omdoc@cls@docopt\@empty%
\xdef\@omdoc@cls@docopt{\CurrentOption}%
\else\xappto\@omdoc@cls@docopt{,\CurrentOption}%
\fi}%
\PassOptionsToPackage{\CurrentOption}{omdoc}
\ProcessKeyvalOptions{omdoc@cls}
```

We load article.cls, and the desired packages. For the LaTeXML bindings, we make sure the right packages are loaded.

```
\LoadClass[\@omdoc@cls@docopt]{\omdoc@cls@class}
```

25.3 Beefing up the document environment

Now, – unless the option minimal is defined – we include the stex package

```
\ifomdoc@cls@minimal\else%
\RequirePackage{omdoc}
```

And define the environments we need. The top-level one is the document environment, which we redefined so that we can provide keyval arguments.

document For the moment we do not use them on the LaTeX level, but the document identifier is picked up by LaTeXML.[8] EdN:8

```
\srefaddidkey{document}
\newcommand\documentkeys[1]{\metasetkeys{document}{#1}}
\let\orig@document=\document
\renewcommand{\document}[1][]{\metasetkeys{document}{#1}\orig@document}
```

Finally, we end the test for the minimal option.

```
\fi% \ifomdoc@cls@minimal
⟨/cls⟩
```

25.4 Implementation: OMDoc Package

25.5 Package Options

We declare some switches which will modify the behavior according to the package options. Generally, an option xxx will just set the appropriate switches to true (otherwise they stay false).

```
⟨*package⟩
\RequirePackage{kvoptions}
\SetupKeyvalOptions{family=omdoc@sty,prefix=omdoc@sty@}
\DeclareStringOption[article]{class}
\DeclareBoolOption{showignores}
\DeclareStringOption[section]{topsect}
\newcount\section@level
\DeclareDefaultOption{\PassOptionsToPackage{\CurrentOption}{sref}}
\ProcessKeyvalOptions{omdoc@sty}
```

[8]EdNote: faking documentkeys for now. @HANG, please implement

Then we need to set up the packages by requiring the `sref` package to be loaded.

```
\RequirePackage{stex-metakeys}
%\RequirePackage{sref}
\RequirePackage{xspace}
\RequirePackage{comment}
%\RequirePackage{pathsuris}
\@ifpackageloaded{babel}{}{\RequirePackage[base]{babel}}

\def\srefaddidkey#1{\addmetakey{#1}{id}}

```

We set up triggers for the other languages, currently only German.

```
\ExplSyntaxOn
\@ifpackageloaded{babel}{
    \clist_set:Nx \l_tmpa_clist {\bbl@loaded}
    \clist_if_in:NnT \l_tmpa_clist {ngerman}{
      \input{omdoc-ngerman.ldf}
    }
}{}
\ExplSyntaxOff
%\AfterBabelLanguage{ngerman}{\input{omdoc-ngerman.ldf}}
```

`\section@level` Finally, we set the `\section@level` macro that governs sectioning. The default is two (corresponding to the `article` class), then we set the defaults for the standard classes `book` and `report` and then we take care of the levels passed in via the `topsect` option.

```
\section@level=2
\ifdefstring{\omdoc@sty@class}{book}{\section@level=0}{}
\ifdefstring{\omdoc@sty@class}{report}{\section@level=0}{}
\ifdefstring{\omdoc@sty@topsect}{part}{\section@level=0}{}
\ifdefstring{\omdoc@sty@topsect}{chapter}{\section@level=1}{}
```

25.6 Document Structure

The structure of the document is given by the `omgroup` environment just like in OMDoc. The hierarchy is adjusted automatically according to the LaTeX class in effect.

`\currentsectionlevel` For the `\currentsectionlevel` and `\Currentsectionlevel` macros we use an internal macro `\current@section@level` that only contains the keyword (no markup). We initialize it with "document" as a default. In the generated OMDoc, we only generate a text element of class `omdoc_currentsectionlevel`, wich will be instantiated by CSS later.[9] EdN:9

```
\def\current@section@level{document}%
\newcommand\currentsectionlevel{\lowercase\expandafter{\current@section@level}\xspace}%
\newcommand\Currentsectionlevel{\expandafter\MakeUppercase\current@section@level\xspace}%
```

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

`\skipomgroup`

```
\newcommand\skipomgroup{%
  \ifcase\section@level%
```

[9]EdNote: MK: we may have to experiment with the more powerful uppercasing macro from `mfirstuc.sty` once we internationalize.

```
  \or\stepcounter{chapter}%
  \or\stepcounter{section}%
  \or\stepcounter{subsection}%
  \or\stepcounter{subsubsection}%
  \or\stepcounter{paragraph}%
  \or\stepcounter{subparagraph}%
  \fi}% \ifcase
```

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

`blindomgroup`

```
\newcommand\at@begin@blindomgroup[1]{}
\newenvironment{blindomgroup}
{\advance\section@level by 1\at@begin@blindomgroup\setion@level}
{\advance\section@level by -1}
```

`\omgroup@nonum` convenience macro: `\omgroup@nonum{`⟨*level*⟩`}{`⟨*title*⟩`}` makes an unnumbered sectioning with title ⟨*title*⟩ at level ⟨*level*⟩.

```
\newcommand\omgroup@nonum[2]{%
\ifx\hyper@anchor\@undefined\else\phantomsection\fi%
\addcontentsline{toc}{#1}{#2}\@nameuse{#1}*{#2}}
```

(*End definition for* `\omgroup@nonum`*. This function is documented on page* **??**.)

`\omgroup@num` convenience macro: `\omgroup@nonum{`⟨*level*⟩`}{`⟨*title*⟩`}` makes numbered sectioning with title ⟨*title*⟩ at level ⟨*level*⟩. We have to check the `short` key was given in the `omgroup` environment and – if it is use it. But how to do that depends on whether the `rdfmeta` package has been loaded. In the end we call `\sref@label@id` to enable crossreferencing.

```
\newcommand\omgroup@num[2]{%
\ifx\omgroup@short\@empty% no short title
\@nameuse{#1}{#2}%
\else% we have a short title
\@ifundefined{rdfmeta@sectioning}%
  {\@nameuse{#1}[\omgroup@short]{#2}}%
  {\@nameuse{rdfmeta@#1@old}[\omgroup@short]{#2}}%
\fi%
%\sref@label@id@arg{\omdoc@sect@name~\@nameuse{the#1}}\omgroup@id
}
```

(*End definition for* `\omgroup@num`*. This function is documented on page* **??**.)

`omgroup`

```
\def\@true{true}
\def\@false{false}
\srefaddidkey{omgroup}
\addmetakey{omgroup}{date}
\addmetakey{omgroup}{creators}
\addmetakey{omgroup}{contributors}
\addmetakey{omgroup}{srccite}
\addmetakey{omgroup}{type}
\addmetakey*{omgroup}{short}
\addmetakey*{omgroup}{display}
\addmetakey*{omgroup}{intro}% ignored
\addmetakey[false]{omgroup}{loadmodules}[true]
```

\at@begin@omgroup

we define a switch for numbering lines and a hook for the beginning of groups: The \at@begin@omgroup macro allows customization. It is run at the beginning of the omgroup, i.e. after the section heading.

```
\newif\if@mainmatter\@mainmattertrue
\newcommand\at@begin@omgroup[3][]{}
```

Then we define a helper macro that takes care of the sectioning magic. It comes with its own key/value interface for customization.

```
\addmetakey{omdoc@sect}{name}
\addmetakey[false]{omdoc@sect}{clear}[true]
\addmetakey{omdoc@sect}{ref}
\addmetakey[false]{omdoc@sect}{num}[true]
\newcommand\omdoc@sectioning[3][]{\metasetkeys{omdoc@sect}{#1}%
\ifx\omdoc@sect@clear\@true\cleardoublepage\fi%
\if@mainmatter% numbering not overridden by frontmatter, etc.
\ifx\omdoc@sect@num\@true\omgroup@num{#2}{#3}\else\omgroup@nonum{#2}{#3}\fi%
\def\current@section@level{\omdoc@sect@name}%
\else\omgroup@nonum{#2}{#3}%
\fi}% if@mainmatter
```

and another one, if redefines the \addtocontentsline macro of LaTeX to import the respective macros. It takes as an argument a list of module names.

```
\newcommand\omgroup@redefine@addtocontents[1]{%
%\edef\@@import{#1}%
%\@for\@I:=\@@import\do{%
%\edef\@path{\csname module@\@I  @path\endcsname}%
%\@ifundefined{tf@toc}\relax%
%     {\protected@write\tf@toc{}{\string\@requiremodules{\@path}}}}
%\ifx\hyper@anchor\@undefined% hyperref.sty loaded?
%\def\addcontentsline##1##2##3{%
%\addtocontents{##1}{\protect\contentsline{##2}{\string\withusedmodules{#1}{##3}}{\thepage}}
%\else% hyperref.sty not loaded
%\def\addcontentsline##1##2##3{%
%\addtocontents{##1}{\protect\contentsline{##2}{\string\withusedmodules{#1}{##3}}{\thepage}{
%\fi
}% hypreref.sty loaded?
```

now the omgroup environment itself. This takes care of the table of contents via the helper macro above and then selects the appropriate sectioning command from article.cls. It also registeres the current level of omgroups in the \omgroup@level counter.

```
\newcount\omgroup@level
\newenvironment{omgroup}[2][]% keys, title
{\metasetkeys{omgroup}{#1}%\sref@target%
\advance\omgroup@level by 1\relax%
```

If the loadmodules key is set on \begin{omgroup}, we redefine the \addcontetsline macro that determines how the sectioning commands below construct the entries for the table of contents.

```
\ifx\omgroup@loadmodules\@true%
\omgroup@redefine@addtocontents{\@ifundefined{module@id}\used@modules%
{\@ifundefined{module@\module@id @path}{\used@modules}\module@id}}\fi%
```

now we only need to construct the right sectioning depending on the value of \section@level.

```
\advance\section@level by 1\relax%
\ifcase\section@level%
```

```
\or\omdoc@sectioning[name=\omdoc@part@kw,clear,num]{part}{#2}%
\or\omdoc@sectioning[name=\omdoc@chapter@kw,clear,num]{chapter}{#2}%
\or\omdoc@sectioning[name=\omdoc@section@kw,num]{section}{#2}%
\or\omdoc@sectioning[name=\omdoc@subsection@kw,num]{subsection}{#2}%
\or\omdoc@sectioning[name=\omdoc@subsubsection@kw,num]{subsubsection}{#2}%
\or\omdoc@sectioning[name=\omdoc@paragraph@kw,ref=this \omdoc@paragraph@kw]{paragraph}{#2}%
\or\omdoc@sectioning[name=\omdoc@subparagraph@kw,ref=this \omdoc@subparagraph@kw]{paragraph}
\fi% \ifcase
\at@begin@omgroup[#1]\section@level{#2}
\csname stex_ref_new_doc_target:n\endcsname\omgroup@id%
}% for customization
{\advance\section@level by -1\advance\omgroup@level by -1}
```

and finally, we localize the sections

```
\newcommand\omdoc@part@kw{Part}
\newcommand\omdoc@chapter@kw{Chapter}
\newcommand\omdoc@section@kw{Section}
\newcommand\omdoc@subsection@kw{Subsection}
\newcommand\omdoc@subsubsection@kw{Subsubsection}
\newcommand\omdoc@paragraph@kw{paragraph}
\newcommand\omdoc@subparagraph@kw{subparagraph}
```

25.7 Front and Backmatter

Index markup is provided by the `omtext` package [Koh20c], so in the `omdoc` package we only need to supply the corresponding `\printindex` command, if it is not already defined

`\printindex`

```
\providecommand\printindex{\IfFileExists{\jobname.ind}{\input{\jobname.ind}}{}}
```

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

some classes (e.g. `book.cls`) already have `\frontmatter`, `\mainmatter`, and `\backmatter` macros. As we want to define `frontmatter` and `backmatter` environments, we save their behavior (possibly defining it) in `orig@*matter` macros and make them undefined (so that we can define the environments).

```
\ifcsdef{frontmatter}% to redefine if necessary
  {\cslet{orig@frontmatter}{\frontmatter}\cslet{frontmatter}{\relax}}
  {\cslet{orig@frontmatter}{\clearpage\@mainmatterfalse\pagenumbering{roman}}}
\ifcsdef{backmatter}% to redefine if necessary
  {\cslet{orig@backmatter}{\backmatter}\cslet{backmatter}{\relax}}
  {\cslet{orig@backmatter}{\clearpage\@mainmatterfalse\pagenumbering{roman}}}
```

Using these, we can now define the `frontmatter` and `backmatter` environments

`frontmatter` we use the `\orig@frontmatter` macro defined above and `\mainmatter` if it exists, otherwise we define it.

```
\newenvironment{frontmatter}
{\orig@frontmatter}
{\ifcsdef{mainmatter}{\mainmatter}{\clearpage\@mainmattertrue\pagenumbering{arabic}}}
```

`backmatter` As backmatter is at the end of the document, we do nothing for `\endbackmatter`.

```
\newenvironment{backmatter}
{\orig@backmatter}
{\ifcsdef{mainmatter}{\mainmatter}{\clearpage\@mainmattertrue\pagenumbering{arabic}}}
```

finally, we make sure that page numbering is arabic and we have main matter as the default

```
\@mainmattertrue\pagenumbering{arabic}
```

25.8 Ignoring Inputs

ignore

```
\ifomdoc@sty@showignores
\addmetakey{ignore}{type}
\addmetakey{ignore}{comment}
\newenvironment{ignore}[1][]
{\metasetkeys{ignore}{#1}\textless\ignore@type\textgreater\bgroup\itshape}
{\egroup\textless/\ignore@type\textgreater}
\renewenvironment{ignore}{}{}\else\excludecomment{ignore}\fi
```

\prematurestop We initialize `\afterprematurestop`, and provide `\prematurestop@endomgroup` which looks up `\omgroup@level` and recursively ends enough `{omgroup}`s.

```
\newcommand\afterprematurestop{}
\def\prematurestop@endomgroup{\ifnum\omgroup@level=0\else%
\end{omgroup}\advance\omgroup@level by -1\prematurestop@endomgroup\fi}
\providecommand\prematurestop{%
\message{Stopping sTeX processing prematurely}
\prematurestop@endomgroup\afterprematurestop
\end{document}}
```

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

25.9 Structure Sharing

EdN:10 [10]

```
\providecommand{\lxDocumentID}[1]{}%
\def\LXMID#1#2{\expandafter\gdef\csname xmarg#1\endcsname{#2}\csname xmarg#1\endcsname}
\def\LXMRef#1{\csname xmarg#1\endcsname}
```

\STRlabel The main macro, it it used to attach a label to some text expansion. Later on, using the `\STRcopy` macro, the author can use this label to get the expansion originally assigned.

```
\long\def\STRlabel#1#2{\STRlabeldef{#1}{#2}{#2}}
```

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

\STRcopy The `\STRcopy` macro is used to call the expansion of a given label. In case the label is not defined it will issue a warning.[11] EdN:11

```
\newcommand\STRcopy[2][]{\expandafter\ifx\csname STR@#2\endcsname\relax
\message{STR warning: reference #2 undefined!}
\else\csname STR@#2\endcsname\fi}
```

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

[10] EdNote: The following is simply copied over from the `latexml` package, which we eliminated, we should integrate better.

[11] EdNote: MK: we need to do something about the ref!

\STRsemantics if we have a presentation form and a semantic form, then we can use

```
\newcommand\STRsemantics[3][]{#2\def\@test{#1}\ifx\@test\@empty\STRlabeldef{#1}{#2}\fi}
```

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

\STRlabeldef This is the macro that does the actual labeling. Is it called inside `\STRlabel`

```
\def\STRlabeldef#1{\expandafter\gdef\csname STR@#1\endcsname}
```

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

25.10 Global Variables

\setSGvar set a global variable

```
\newcommand\setSGvar[1]{\@namedef{sTeX@Gvar@#1}}
```

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

\useSGvar use a global variable

```
\newrobustcmd\useSGvar[1]{%
  \@ifundefined{sTeX@Gvar@#1}
  {\PackageError{omdoc}
    {The sTeX Global variable #1 is undefined}
    {set it with \protect\setSGvar}}
\@nameuse{sTeX@Gvar@#1}}
```

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

\ifSGvar execute something conditionally based on the state of the global variable.

```
\newrobustcmd\ifSGvar[3]{\def\@test{#2}%
  \@ifundefined{sTeX@Gvar@#1}
  {\PackageError{omdoc}
    {The sTeX Global variable #1 is undefined}
    {set it with \protect\setSGvar}}
  {\expandafter\ifx\csname sTeX@Gvar@#1\endcsname\@test #3\fi}}
```

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

25.11 Colors

blue, red, green, magenta We will use the following abbreviations for colors from `color.sty`

```
\def\black#1{\textcolor{black}{#1}}
\def\gray#1{\textcolor{gray}{#1}}
\def\blue#1{\textcolor{blue}{#1}}
\def\red#1{\textcolor{red}{#1}}
\def\green#1{\textcolor{green}{#1}}
\def\cyan#1{\textcolor{cyan}{#1}}
\def\magenta#1{\textcolor{magenta}{#1}}
\def\brown#1{\textcolor{brown}{#1}}
\def\yellow#1{\textcolor{yellow}{#1}}
\def\orange#1{\textcolor{orange}{#1}}
⟨/package⟩
```

Chapter 26

MiKoSlides – Implementation

26.1 Class and Package Options

We define some Package Options and switches for the `mikoslides` class and activate them by passing them on to `beamer.cls` and `omdoc.cls` and the `mikoslides` package. We pass the `nontheorem` option to the `statements` package when we are not in notes mode, since the `beamer` package has its own (overlay-aware) theorem environments.

```
⟨*cls⟩
\RequirePackage{kvoptions}
\RequirePackage{stex-metakeys}
\RequirePackage{etoolbox}
\SetupKeyvalOptions{family=mks@cls,prefix=mks@cls@}
\DeclareStringOption[article]{class}
\AddToKeyvalOption*{class}{\PassOptionsToClass{class=\mks@cls@class}{omdoc}
  \ifdefstring{\mks@cls@class}{book}{\PassOptionsToPackage{defaulttopsect=part}{mikoslides}}
  \ifdefstring{\mks@cls@class}{report}{\PassOptionsToPackage{defaulttopsect=part}{mikoslides
\DeclareBoolOption{notes}
\DeclareComplementaryOption{slides}{notes}
\DeclareDefaultOption{%
  \PassOptionsToClass{\CurrentOption}{omdoc}
  \PassOptionsToClass{\CurrentOption}{beamer}
  \PassOptionsToPackage{\CurrentOption}{mikoslides}}
\ProcessKeyvalOptions{mks@cls}
⟨/cls⟩
```

now we do the same for the `mikoslides` package.

```
⟨*package⟩
%\RequirePackage{stex-base}
\RequirePackage{kvoptions}
\RequirePackage{stex-metakeys}
\SetupKeyvalOptions{family=mks@sty,prefix=mks@sty@}
\DeclareStringOption{topsect}
\DeclareStringOption{defaulttopsect}
\newif\ifnotes\notestrue
\DeclareBoolOption{notes}
\AddToKeyvalOption*{notes}{\notestrue}%\PassOptionsToPackage{notes}{statements}}
\DeclareComplementaryOption{slides}{notes}
\AddToKeyvalOption*{slides}{\notesfalse}%\PassOptionsToPackage{nontheorem}{statements}}
```

```
\DeclareBoolOption{sectocframes}
\DeclareBoolOption{frameimages}
\DeclareBoolOption{fiboxed}
\DeclareBoolOption{noproblems}
%\DeclareDefaultOption{
  %\PassOptionsToPackage{\CurrentOption}{stex}
  %\PassOptionsToPackage{\CurrentOption}{smglom}
  %\PassOptionsToPackage{\CurrentOption}{tikzinput}}
\ProcessKeyvalOptions{mks@sty}
```

we give ourselves a macro `\@@topsect` that needs only be evaluated once, so that the `\ifdefstring` conditionals work below.

```
\ifx\mks@sty@topsect\@empty\edef\@@topsect{\mks@sty@defaulttopsect}
\else\edef\@@topsect{\mks@sty@topsect}\fi
⟨/package⟩
```

Depending on the options, we either load the `article`-based `omdoc` or the `beamer` class (and set some counters).

```
⟨*cls⟩
\ifmks@cls@notes
  \LoadClass{omdoc}
\else
  \LoadClass[10pt,notheorems,xcolor={dvipsnames,svgnames}]{beamer}
  \newcounter{Item}
  \newcounter{paragraph}
  \newcounter{subparagraph}
  \newcounter{Hfootnote}
\fi
```

now it only remains to load the `mikoslides` package that does all the rest.

```
\RequirePackage{mikoslides}
⟨/cls⟩
```

In `notes` mode, we also have to make the `beamer`-specific things available to `article` via the `beamerarticle` package. We use options to avoid loading theorem-like environments, since we want to use our own from the sTEX packages. The first batch of packages we want are loaded on `mikoslides.sty`. These are the general ones, we will load the sTEX-specific ones after we have done some work (e.g. defined the counters `m*`). Only the `stex-logo` package is already needed now for the default theme.

```
⟨*package⟩
%\RequirePackage{stex}
\RequirePackage{stex-compatibility}
\ifmks@sty@notes
\RequirePackage{a4wide}
\RequirePackage{marginnote}
\RequirePackage[dvipsnames,svgnames]{xcolor}
\RequirePackage{mdframed}
\RequirePackage[noxcolor,noamsthm]{beamerarticle}
\fi
\RequirePackage{etoolbox}
\RequirePackage{amssymb}
\RequirePackage{amsmath}
\RequirePackage{comment}
\RequirePackage{textcomp}
\RequirePackage{url}
```

```
\RequirePackage{graphicx}
\RequirePackage{pgf}
%\RequirePackage{omtext}
\ifmks@sty@notes
\RequirePackage[bookmarks,bookmarksopen,bookmarksnumbered,breaklinks,hidelinks]{hyperref}
\fi
```

finally, we require the `metakeys` package from sTEX, so that we can use the `\addmetakey` mechanism.

```
%\RequirePackage{metakeys}
```

26.2 Notes and Slides

For the lecture notes cases, we also provide the `\usetheme` macro that would otherwise come from the the `beamer` class. While the latter loads `beamertheme`⟨*theme*⟩.sty, the notes version loads `beamernotestheme`⟨*theme*⟩`.sty`.[12] EdN:12

```
\ifmks@sty@notes
\renewcommand\usetheme[2][]{\usepackage[#1]{beamernotestheme#2}}
\fi
```

We define the sizes of slides in the notes. Somehow, we cannot get by with the same here.

```
\newcounter{slide}
\newlength{\slidewidth}\setlength{\slidewidth}{13.5cm}
\newlength{\slideheight}\setlength{\slideheight}{9cm}
```

`note` The `note` environment is used to leave out text in the `slides` mode. It does not have a counterpart in OMDoc. So for course notes, we define the `note` environment to be a no-operation otherwise we declare the `note` environment as a comment via the `comment` package.

```
\ifmks@sty@notes%
  \renewenvironment{note}{\ignorespaces}{}%
\else%
  \excludecomment{note}%
\fi%
```

We first set up the slide boxes in `article` mode. We set up sizes and provide a box register for the frames and a counter for the slides.

```
\ifmks@sty@notes
  \newlength{\slideframewidth}
  \setlength{\slideframewidth}{1.5pt}
```

`frame` We first define the keys.

```
  \addmetakey{frame}{label}
  \addmetakey[yes]{frame}{allowframebreaks}
  \addmetakey{frame}{allowdisplaybreaks}
  \addmetakey[yes]{frame}{fragile}
  \addmetakey[yes]{frame}{shrink}
  \addmetakey[yes]{frame}{squeeze}
  \addmetakey[yes]{frame}{t}
```

[12]EdNote: MK: This is not ideal, but I am not sure that I want to be able to provide the full theme functionality there.

We define the environment, read them, and construct the slide number and label.

```
  \renewenvironment{frame}[1][]{%
    \metasetkeys{frame}{#1}%
    \sffamily%
    \stepcounter{slide}%
    \def\@currentlabel{\theslide}%
    \ifx\frame@label\@empty\else\label{\frame@label}\fi%
```

We redefine the itemize environment so that it looks more like the one in beamer.

```
    \def\itemize@level{outer}%
    \def\itemize@outer{outer}%
    \def\itemize@inner{inner}%
    \renewcommand\newpage{\addtocounter{framenumber}{1}}%
    \renewcommand\metakeys@show@keys[2]{\marginnote{{\scriptsize ##2}}}%
    \renewenvironment{itemize}{%
      \ifx\itemize@level\itemize@outer%
        \def\itemize@label{$\rhd$}%
      \fi%
      \ifx\itemize@level\itemize@inner%
        \def\itemize@label{$\scriptstyle\rhd$}%
      \fi%
      \begin{list}%
      {\itemize@label}%
      {\setlength{\labelsep}{.3em}%
       \setlength{\labelwidth}{.5em}%
       \setlength{\leftmargin}{1.5em}%
      }%
      \edef\itemize@level{\itemize@inner}%
    }{%
      \end{list}%
    }%
```

We create the box with the mdframed environment from the equinymous package.

```
    \begin{mdframed}[linewidth=\slideframewidth,skipabove=1ex,skipbelow=1ex,userdefinedwidth
  }{%
    \medskip\miko@slidelabel\end{mdframed}%
  }%
```

Now, we need to redefine the frametitle (we are still in course notes mode).

\frametitle

```
  \renewcommand{\frametitle}[1]{{\Large\bf\sf\color{blue}{#1}}\medskip}%
\fi %ifmks@sty@notes
```

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

EdN:13 \pause [13]

```
\ifmks@sty@notes\newcommand\pause{}\fi
```

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

nomtext

```
\ifmks@sty@notes\newenvironment{nomtext}[1][]{\begin{omtext}[#1]}{\end{omtext}}%
\else\excludecomment{nomtext}\fi%
```

[13]EdNote: MK: fake it in notes mode for now

nomgroup

```
\ifmks@sty@notes\newenvironment{nomgroup}[2][]{\begin{omgroup}[#1]{#2}}{\end{omgroup}}%
\else\excludecomment{nomgroup}\fi%
```

ndefinition

```
\ifmks@sty@notes\newenvironment{ndefinition}[1][]{\begin{definition}[#1]}{\end{definition}}%
\else\excludecomment{ndefinition}\fi%
```

nassertion

```
\ifmks@sty@notes\newenvironment{nassertion}[1][]{\begin{assertion}[#1]}{\end{assertion}}%
\else\excludecomment{nassertion}\fi%
```

nsproof

```
\ifmks@sty@notes\newenvironment{nsproof}[2][]{\begin{sproof}[#1]{#2}}{\end{sproof}}%
\else\excludecomment{nsproof}\fi%
```

nexample

```
\ifmks@sty@notes\newenvironment{nexample}[1][]{\begin{example}[#1]}{\end{example}}%
\else\excludecomment{nexample}\fi%
```

`\inputref@*skip` We customize the hooks for in `\inputref`.

```
\def\inputref@preskip{\smallskip}
\def\inputref@postskip{\medskip}
```

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

`\inputref*`

```
\let\orig@inputref\inputref
\def\inputref{\@ifstar\ninputref\orig@inputref}
\newcommand\ninputref[2][]{\ifmks@sty@notes\orig@inputref[#1]{#2}\fi}
```

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

26.3 Header and Footer Lines

Now, we set up the infrastructure for the footer line of the slides, we use boxes for the logos, so that they are only loaded once, that considerably speeds up processing.

`\setslidelogo` The default logo is the sTEX logo. Customization can be done by `\setslidelogo{`⟨*logo name*⟩`}`.

```
\newlength{\slidelogoheight}
\ifmks@sty@notes%
  \setlength{\slidelogoheight}{.4cm}%
\else%
  \setlength{\slidelogoheight}{1cm}%
\fi%
\newsavebox{\slidelogo}%
\sbox{\slidelogo}{\sTeX}%
\newrobustcmd{\setslidelogo}[1]{%
  \sbox{\slidelogo}{\includegraphics[height=\slidelogoheight]{#1}}%
}%
```

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

`\setsource` `\source` stores the writer's name. By default it is *Michael Kohlhase* since he is the main user and designer of this package. `\setsource{`⟨*name*⟩`}` can change the writer's name.

```
\def\source{Michael Kohlhase}% customize locally
\newrobustcmd{\setsource}[1]{\def\source{#1}}%
```

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

`\setlicensing` Now, we set up the copyright and licensing. By default we use the Creative Commons Attribuition-ShareAlike license to strengthen the public domain. If package `hyperref` is loaded, then we can attach a hyperlink to the license logo. `\setlicensing[`⟨*url*⟩`]{`⟨*logo name*⟩`}` is used for customization, where ⟨*url*⟩ is optional.

```
\def\copyrightnotice{\footnotesize\copyright:\hspace{.3ex}{\source}}%
\newsavebox{\cclogo}%
\sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{cc_somerights}}%
\newif\ifcchref\cchreffalse%
\AtBeginDocument{%
  \@ifpackageloaded{hyperref}{\cchreftrue}{\cchreffalse}
}%
\def\licensing{%
  \ifcchref%
    \href{http://creativecommons.org/licenses/by-sa/2.5/}{\usebox{\cclogo}}%
  \else%
    {\usebox{\cclogo}}%
  \fi%
}%
\newrobustcmd{\setlicensing}[2][]{%
  \def\@url{#1}%
  \sbox{\cclogo}{\includegraphics[height=\slidelogoheight]{#2}}%
  \ifx\@url\@empty%
    \def\licensing{{\usebox{\cclogo}}}%
  \else%
    \def\licensing{%
      \ifcchref%
      \href{#1}{\usebox{\cclogo}}%
      \else%
      {\usebox{\cclogo}}%
      \fi%
    }%
  \fi%
}%
```

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

EdN:14 `\slidelabel` Now, we set up the slide label for the `article` mode.[14]

```
\newrobustcmd\miko@slidelabel{%
  \vbox to \slidelogoheight{%
    \vss\hbox to \slidewidth%
    {\licensing\hfill\copyrightnotice\hfill\arabic{slide}\hfill\usebox{\slidelogo}}%
  }%
}%
% \subsection{Frame Images}\label{sec:impl:frameimage}
%
```

[14] EdNote: see that we can use the themes for the slides some day. This is all fake.

```
% \begin{macro}{\frameimage}
%   We have to make sure that the width is overwritten, for that we check the
%   |\Gin@ewidth| macro from the |graphicx| package. We also add the |label| key.
%    \begin{macrocode}
\define@key{Gin}{label}{\def\@currentlabel{\arabic{slide}}\label{#1}}
\newrobustcmd\frameimage[2][]{%
  \stepcounter{slide}%
  \ifmks@sty@frameimages%
    \def\Gin@ewidth{}\setkeys{Gin}{#1}%
    \ifmks@sty@notes\else\vfill\fi%
    \begin{center}
      \ifmks@sty@fiboxed%
        \fbox{\ifx\Gin@ewidth\@empty\includegraphics[width=\slidewidth,#1]{#2}\else\mygraphi
      \else
        \ifx\Gin@ewidth\@empty\includegraphics[width=\slidewidth,#1]{#2}\else\mygraphics[#1]
      \fi% ifmks@sty@fiboxed
     \end{center}
    \par\strut\hfill{\footnotesize Slide \arabic{slide}}%
    \ifmks@sty@notes\else\vfill\fi%
  \fi} % ifmks@sty@frameimages
```

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

`\mhframeimage` Use the current value of `\mh@currentrepos` or the value of the `mhrepos` key if it is given in `\frameimage`.

```
\def\Gin@mhrepos{}
\define@key{Gin}{mhrepos}{\def\Gin@mhrepos{#1}}
\newcommand\mhframeimage[2][]{%
  \setkeys{Gin}{#1}%
  \edef\mh@@repos{\mh@currentrepos}%
  \ifx\Gin@mhrepos\@empty%
    \edef\temp@path{\MathHub{\mh@currentrepos/source/#2}}%
  \else%
    \edef\temp@path{\MathHub{\Gin@mhrepos/source/#2}}%
  \fi%
  \if@iswindows@\path@to@windows\temp@path\fi%
  \frameimage[#1]{\temp@path}%
}%
```

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

26.4 Colors and Highlighting

We first specify sans serif fonts as the default.

```
\sffamily
```

Now, we set up an infrastructure for highlighting phrases in slides. Note that we use content-oriented macros for highlighting rather than directly using color markup. The first thing to to is to adapt the green so that it is dark enough for most beamers

```
\AtBeginDocument{%
\definecolor{green}{rgb}{0,.5,0}%
\definecolor{purple}{cmyk}{.3,1,0,.17}%
}%
```

We customize the `\defemph`, `\notemph`, and `\stDMemph` macros with colors for the use in the `statements` package. Furthermore we customize the `\@@lec` macro for the appearance of line end comments in `\lec`.

```
% \def\STpresent#1{\textcolor{blue}{#1}}
\def\defemph#1{{\textcolor{magenta}{#1}}}
\def\symrefemph#1{{\textcolor{cyan}{#1}}}
\def\compemph#1{{\textcolor{magenta}{#1}}}
\def\titleemph#1{{\textcolor{blue}{#1}}}
\def\@@lec#1{(\textcolor{green}{#1})}
```

I like to use the dangerous bend symbol for warnings, so we provide it here.

`\textwarning` as the macro can be used quite often we put it into a box register, so that it is only loaded once.

```
\pgfdeclareimage[width=.8em]{miko@small@dbend}{dangerous-bend}
\def\smalltextwarning{%
  \pgfuseimage{miko@small@dbend}%
  \xspace%
}%
\pgfdeclareimage[width=1.2em]{miko@dbend}{dangerous-bend}
\newrobustcmd\textwarning{%
  \raisebox{-.05cm}{\pgfuseimage{miko@dbend}}%
  \xspace%
}%
\pgfdeclareimage[width=2.5em]{miko@big@dbend}{dangerous-bend}%
\newrobustcmd\bigtextwarning{%
  \raisebox{-.05cm}{\pgfuseimage{miko@big@dbend}}%
  \xspace%
}%
```

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

```
\newrobustcmd\putgraphicsat[3]{%
  \begin{picture}(0,0)\put(#1){\includegraphics[#2]{#3}}\end{picture}%
}%
\newrobustcmd\putat[2]{%
  \begin{picture}(0,0)\put(#1){#2}\end{picture}%
}%
```

26.5 Sectioning

If the `sectocframes` option is set, then we make section frames. We first define counters for `part` and `chapter`, which `beamer.cls` does not have and we make the `section` counter which it does dependent on `chapter`.

```
\ifmks@sty@sectocframes%
\ifdefstring\@@topsect{part}{%
  \newcounter{chapter}\counterwithin*{section}{chapter}}
{\ifdefstring\@@topsect{chapter}{\newcounter{chapter}\counterwithin*{section}{chapter}}{}}
\fi% ifsectocframes
```

Now that we have defined the counters, we can load the sTeX-specific packages (in particular `statements` that needs these counters).

```
\RequirePackage{tikzinput}
```

\section@level Finally, we set the \section@level counter that governs sectioning according to the class options. We also introduce the sectioning counters accordingly.

\section@level

```
\section@level=2
\def\part@prefix{}
\ifdefstring{\@@topsect}{part}
{\section@level=0%
  \def\thesection{\arabic{chapter}.\arabic{section}}%
  \def\part@prefix{\arabic{chapter}.}}{}
\ifdefstring{\@@topsect}{chapter}
{\section@level=1%
  \def\thesection{\arabic{chapter}.\arabic{section}}%
  \def\part@prefix{\arabic{chapter}.}}{}
\ifmks@sty@notes\else% only in slides
```

(*End definition for* \section@level. *This function is documented on page* **??**.)

The new counters are used in the omgroup environment that choses the LaTeX sectioning macros according to \section@level.

omgroup

```
\renewenvironment{omgroup}[2][]{%
  \metasetkeys{omgroup}{#1}%
  \advance\section@level by 1%
  \advance\omgroup@level by 1%
  \ifmks@sty@sectocframes%
  \stepcounter{slide}
  \begin{frame}[noframenumbering]%
  \vfill\Large\centering%
  \red{%
    \ifcase\section@level\or
    \stepcounter{part}
    \def\@@label{\omdoc@part@kw~\Roman{part}}
    \def\currentsectionlevel{\omdoc@part@kw}
    \or%
    \stepcounter{chapter}
    \def\@@label{\omdoc@chapter@kw~\arabic{chapter}}
    \def\currentsectionlevel{\omdoc@chapter@kw}
    \or
    \stepcounter{section}
    \def\@@label{\part@prefix\arabic{section}}
    \def\currentsectionlevel{\omdoc@section@kw}
    \or
    \stepcounter{subsection}
    \def\@@label{\part@prefix\arabic{section}.\arabic{subsection}}
    \def\currentsectionlevel{\omdoc@subsection@kw}
    \or
    \stepcounter{subsubsection}
    \def\@@label{\part@prefix\arabic{section}.\arabic{subsection}.\arabic{subsubsection}}
    \def\currentsectionlevel{\omdoc@subsubsection@kw}
    \or
    \stepcounter{mparagraph}
    \def\@@label{\part@prefix\arabic{section}.\arabic{msubsection}.\arabic{subsubsection}.\a
    \def\currentsectionlevel{\omdoc@paragraph@kw}
    \fi% end ifcase
    \@@label\sref@label@id\@@label
    \quad #2%
  }%
  \vfill%
  \end{frame}%
  \fi %ifmks@sty@sectocframes
  \csname stex_ref_new_doc_target:n\endcsname\omgroup@id%
}
{\advance\section@level by -1}%
\fi% ifmks@sty@notes
```

We set up a beamer template for theorems like ams style, but without a block environment.

```
\def\inserttheorembodyfont{\normalfont}
\ifmks@sty@notes\else% only in slides
\defbeamertemplate{theorem begin}{miko}
{\inserttheoremheadfont\inserttheoremname\inserttheoremnumber
  \ifx\inserttheoremaddition\@empty\else\ (\inserttheoremaddition)\fi%
  \inserttheorempunctuation\inserttheorembodyfont\xspace}
\defbeamertemplate{theorem end}{miko}{}
```

and we set it as the default one.

```
\setbeamertemplate{theorems}[miko]
```

The following fixes an error I do not understand, this has something to do with beamer compatibility, which has similar definitions but only up to 1.

```
\expandafter\def\csname Parent2\endcsname{}
\fi% ifmks@sty@notes
\ifmks@sty@notes%
  \renewenvironment{columns}[1][]{%
    \par\noindent%
    \begin{minipage}%
    \slidewidth\centering\leavevmode%
  }{%
    \end{minipage}\par\noindent%
  }%
  \newsavebox\columnbox%
  \renewenvironment<>{column}[2][]{%
    \begin{lrbox}{\columnbox}\begin{minipage}{#2}%
  }{%
    \end{minipage}\end{lrbox}\usebox\columnbox%
  }%
\fi% ifmks@sty@notes
```

```
\ifmks@sty@noproblems%
  \newenvironment{problems}{}{}%
\else%
  \excludecomment{problems}%
\fi%
```

26.6 Excursions

`\excursion` The excursion macros are very simple, we define a new internal macro `\excursionref` and use it in `\excursion`, which is just an `\inputref` that checks if the new macro is defined before formatting the file in the argument.

```
\gdef\printexcursions{}
\newcommand\excursionref[2]{% label, text
\ifnotes\begin{omtext}[title=Excursion]#2 \sref[fallback=the appendix]{#1}.\end{omtext}\fi}
\newcommand\activate@excursion[2][]{\gappto\printexcursions{\inputref[#1]{#2}}}
\newcommand\excursion[4][]{% repos, label, path, text
  \ifnotes\activate@excursion[#1]{#3}\excursionref{#2}{#4}\fi}%
```

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

`\excursiongroup`

```
\srefaddidkey{excursiongroup}%
\addmetakey{excursiongroup}{intro}%
\addmetakey{excursiongroup}{mhrepos}%
\newcommand\excursiongroup[1][]{\metasetkeys{excursiongroup}{#1}%
  \ifdefempty\printexcursions{}% only if there are excursions
  {\begin{omgroup}[#1]{Excursions}%
      \ifdefempty\excursiongroup@intro{}{\inputref[\excursiongroup@mhrepos]{\excursiongroup@
      \printexcursions%
    \end{omgroup}}}
⟨/package⟩
```

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

26.7 Miscellaneous