oops, an object oriented practical scribe's package.[*]

Erwann Rogard[†]

Released 2020/04/06

Abstract

oops is a package for LaTeX (hence "scribe") for generating macro definitions as the need arises in the document, and to organize them along two dimensions: functions and objects, hence "OO". This is done using a minimalist interface built upon xparse[3]. Specifically, `\OopsNew{⟨token list₁⟩}`, where ⟨*token list*$_1$⟩ identifies an object, begins a series of instructions alternating between 'text' and definitions, that themselves optionally expand using predefined or inline rules. For example,

```
\OopsNew{Math}[Let~]{Space=\Omega}*[~denote the sample space]{}
```

expands to: "Let Ω denote the sample space". As a side effect, `$\Space[Math]$` encodes "Ω". `Math`being the default for ⟨*token list*$_1$⟩, `$\Space$` also works. Optionally, the definitions can be written to a file, and restored, which can be useful for typesetting documents sharing the same notational conventions. Altogether, "practical".

Contents

[*] This file describes version v1.2, last revised 2020/04/06.

[†] firstname dot lastname AusTria gmail dot com

Part I
Usage

This part describes .

Convention

1. Loosely, those of [2] and [3], for example as to the meaning of ⟨*token list*⟩ and -NoValue-.
2. If unspecified, the environment in which a function must be declared is `document`.
3. Where ⟨*token list*$_1$⟩ is an optional argument, its default is `Math`.

`\usepackage`

```
\usepackage{oops}
```

Environment Preamble

Requirement `oops.sty` is in the path of the LaTeX engine. See Part III, section 4.

`\OopsClear`

```
\OopsClear[⟨keyval list⟩]
```

Semantics Clears any data created by `\OopsNew{`⟨*token list*$_1$⟩`}`, for all ⟨*token list*$_1$⟩ in ⟨*keyval list*⟩

`\OopsNew`

```
\OopsNew{⟨token list1⟩}
[⟨token list2⟩]
i{⟨code1⟩}
{⟨keyval list1⟩}
*
s{{⟨token list3⟩}{⟨token list4⟩}{⟨token list5⟩}}
o{⟨code2⟩}
[⟨token list6⟩]
```

Requirement ⟨*token list*$_1$⟩ and ⟨*keyval list*$_1$⟩ are mandatory.

⟨*token list*$_1$⟩

Example `Math`, `ModelA`, `ModelB`

Semantics Registers a new object, if applicable

⟨*token list*$_2$⟩

Example `Let~`

Semantics Expands ⟨*token list*$_2$⟩

⟨*code*$_1$⟩

Example `\mathbb{#1}`

Semantics 1. ⟨*val*$_i$ ⟩ ← ⟨*code*$_1$⟩ applied to ⟨*val*$_i$ ⟩

⟨*keyval list*$_1$⟩

Example `Elems={\omega_1, \dots, \omega_n}, Sample=\Omega`

Semantics 2. \⟨*key*$_i$⟩[⟨*token list*$_1$⟩] ← ⟨*val*$_i$ ⟩ defined in 1.

3. If `Write=\BooleanTrue`, writes the definitions made in 2. to file `oops`⟨*digits*⟩`.tex`, where ⟨*digits*⟩=`\pdfdate`

*

Semantics 4. Expands ⟨*code*$_2$⟩ applied to the list created in 1., using {⟨*token list*$_3$⟩}{⟨*token list*$_4$⟩}{⟨*token list*$_5$⟩} as separator.

⟨*token list*$_3$⟩

Example `{~\&~}`

⟨*token list*$_4$⟩

Example `{,~}`

⟨*token list*$_5$⟩

Example `{~\&~}`

⟨*code*$_2$⟩

Example `\text{#1}`

⟨*token list*$_6$⟩

Semantics `\OopsNew{`⟨*token list*$_1$⟩`}[`⟨*token list*$_6$⟩`]`

`\OopsOption` `\OopsOption{`⟨*kvl0*⟩`}`

Semantics Set default options for `\OopsNew`

`Inner`

Semantics Default for ⟨*code*$_1$⟩

Syntax Use `####1` as the argument to be replaced

`Name`

Semantics Default for ⟨*token list*$_1$⟩

`Outer`

Semantics Default for ⟨*code*$_2$⟩

Syntax Use `####1` as the argument to be replaced

`Separ`

Semantics Default for `{`⟨*token list*$_3$⟩`}{`⟨*token list*$_4$⟩`}{`⟨*token list*$_5$⟩`}`

Syntax That of 'separators' in [2, Section 8 of l3seq]

`Write`

Syntax ⟨*boolean*⟩

`\OopsRead` `\OopsRead[`⟨*path*⟩`]`

Semantics

1. Reads the definitions in ⟨*path*⟩
2. Writes to `oops.log`: 'read from ⟨*path*⟩'

Part II
Listing

Warning: To reproduce the listings in a LaTeX document, use the same formatting instructions as those of the documentation portion of `oops.dtx` (such as `\documentclass`, `\usepackage`, and `\newtcblisting`), and remove any `^^A`. Any deviation from the original may require tinkering.[1]

Listing 1.

```
%      \OopsOption{
%      Inner={\char`{####1\char`}},
%      ^^A% spaces betw. inner and outer brackets matter!->
%      Separ={{\ \char`@\ }{\%\ }{\ \char`@\ }},
%      Outer={\char`^####1\$}}
%      \OopsNew{Test}{ X =x, Y = y, Z = z }*
%      \tab \X[Test]\Y[Test]\Z[Test]\\
%      \OopsNew{Test}i{(#1)}{ X = x, Y = y, Z = z }*
%      \tab \X[Test]\Y[Test]\Z[Test]\\
%      \OopsNew{Test}{ X = x, Y = y, Z = z }*s{{\ \&\ }{,\ }{\ \&\ }}
%      \tab \X[Test]\Y[Test]\Z[Test]\\
%      \OopsOption{ Write = \BooleanTrue }
%      \OopsNew{Test}{ X = x, Y = y, Z = z }*o{\char`[#1\char`]}
%      \tab \X[Test]\Y[Test]\Z[Test]\\
%      \OopsClear[Test]
%      \OopsOption{ Write = \BooleanFalse }
%
```

```
^{x}% {y} @ {z}$                    {x}{y}{z}
^(x)% (y) @ (z)$                    (x)(y)(z)
^{x}, {y} & {z}$                    {x}{y}{z}
[{x}% {y} @ {z}]                    {x}{y}{z}
```

Listing 2.

```
%      \OopsRead \tab \X[Test]\Y[Test]\Z[Test]
%      \OopsClear[Test]
%
```

```
                                    {x}{y}{z}
```

Listing 3.

```
%      \OopsNew{Math}[We call~]{Elems={\omega_1, \dots, \omega_n}}*
%      [~the elementary events, and ]{Space=\Omega}
%      [\begin{equation*}\Space=(\Elems)\end{equation*}~the sample space.]
```

[1] For instance, in testing v1.1, I realized `\usepackage[T1]{fontenc}` was needed, to work with `\documentclass{article}` in place of `\documentclass[full]{l3doc}`, hence added it to the documentation portion of `oops.dtx`

```
%     {}
%     \OopsClear
%
```

We call $\omega_1, \ldots, \omega_n$ the elementary events, and

$$\Omega = (\omega_1, \ldots, \omega_n)$$

the sample space.

Listing 4.

```
%     \OopsOption{ Write = \BooleanTrue }
%     \OopsNew{Math}[Let ]
%     {Space=\Omega, SigmaField=\mathcal{F}, Measure=\mathcal{P}}
%     *s{{,}{,}{,}}o{\ensuremath{\{#1\}}}
%     [~denote the probability space, where $\SigmaField\subset
    2^{\Space}$.]
%     {}
%     \OopsClear
%     \OopsOption{ Write = \BooleanFalse }
%
```

Let $\{\Omega, \mathcal{F}, \mathcal{P}\}$ denote the probability space, where $\mathcal{F} \subset 2^{\Omega}$.

Listing 5.

```
%     \OopsRead \tab $\Omega$ $\SigmaField$ $\Measure$
%     \OopsClear
%
```

Ω $\mathcal{F}$ $\mathcal{P}$

Listing 6.

```
%     \OopsOption{ Write = \BooleanTrue }
%     \newtheorem{theorem}{Theorem}
%     \OopsNew{Math}i{\mathbb{#1}}
%     { N = { N } , R = { R }, Grad = { \operatorname{grad} } }
%     [\begin{theorem}
%       [Mittelwertsatz f\"ur $n$ Variable]Es~sei~]
%       { OffMenge = {D}, Ci = {C^{1}}, Strecke = {[x_0,x]} }
%       [$n\in\N$,~$\OffMenge\subseteq\N^n$ eine offene Menge und
    $f\in\Ci(\OffMenge,\R)$.
%       Dann gibt es auf jeder Strecke $\Strecke\subset\OffMenge$ einen
    Punkt $\xi\in\Strecke$,~]
%       { yD = { f(x)-f(x_0) }, xD = { x-x_0 }, Steig = { \frac{\yD}{\xD}
    } }
%       [so dass gilt
%       \begin{equation*}
%         \Steig = \Grad f(\xi)^{\top}
%       \end{equation*}
%     \end{theorem}]
%     {}
%     \OopsClear
%     \OopsOption{ Write = \BooleanFalse }
%
```

Theorem 1 (Mittelwertsatz für n Variable) *Es sei $n \in N$, $D \subseteq N^n$ eine offene Menge und $f \in C^1(D, R)$. Dann gibt es auf jeder Strecke $[x_0, x] \subset D$ einen Punkt $\xi \in [x_0, x]$, so dass gilt*

$$\frac{f(x) - f(x_0)}{x - x_0} = \mathrm{grad} f(\xi)^{\top}$$

Listing 7.

```
%     \OopsRead \tab $\N$ $\R$ $\OffMenge$ $\Ci$ $\Strecke$
%
```

N R D C^1 $[x_0, x]$

Part III
Other

1 Acknowledgment

This work has benefited from Q&A's from the LaTeXcommunity, see here: `https://tex.stackexchange.com/users/112708/erwann?tab=questions`. Specific references are made in Part IV. Listing 3 and Listing 4 are from [1]. Listing 6 is from `tcolbox`[4, 17.3].

2 Bug

1. **Input:** `Inner={\{####1\}`

   Symptom: `\OopsRead` fails

   Workaround: ``Inner={\char`{####1\char`}}``

   See: Listing 1

2. **Input:** Inside ⟨*keyval list*$_1$⟩, `{\[a,b\]}`

   Workaround: `{[a,b]}` or ``{\char`[a, b\char`]}``

   See: Listing 6

3. **Input:** Inside ⟨*token list*$_2$⟩, `{[a, b)}`

   Workaround: `{{[}a, b{)}}`

4. **Input:** Inside ⟨*token list*$_2$⟩, `\cal F`

   Workaround: `\mathcal{F}`

3 Install

Compiling `oops.dtx` (under Unix, `$tex oops.dtx`) will generate `oops.sty` and `oops.pdf`

4 Support

This package is available from `https://www.ctan.org/pkg/oops` and `https://github.com/rogard/oops`.

5 Unit testing

It's not possible to check the expansion of a certain class of macros against predefined values[5]. Instead, one can check that Part II, as generated in section 3 on one's own machine, agrees with `bench.pdf` available at `https://github.com/rogard/oops`,

References

[1] A.N. Shiryaev *Probability* Springer, 1995

[2] The LaTeX3 Project Team *The LaTeX3 interfaces* http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/interface3.pdf

[3] The LaTeX3 Project Team *The xparse package* http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf

[4] Thomas F. Sturm *The tcolorbox package* http://www.texdoc.net/texmf-dist/doc/latex/tcolorbox/tcolorbox.pdf

[5] https://tex.stackexchange.com/a/534100/112708

Change History

v1.0

General: Initial version . . . 10

v1.1

General: Added: Save . . . 10

Added: Listing 1., 2., 3., 4., 6., and 9. . . . 10

Added:\OopsRestore . . . 10

Added:\OopsTest . . . 10

Deleted: Listing 1-5 from v1.0 . . . 10

Fixed: apparent anomaly in v1.0's Listing 4, see Listing 1 . . . 10

Replaced: \OopsOptions by \OopsOption . . 10

Replaced: {⟨*keyval list*$_2$⟩} by <keyval list$_2$> given that option type G not recommended[3] . . . 10

Replaced: GenericObject by Name 10

Replaced: Separators by Separ . . 10

Revamped: much of the implementation . . . 10

v1.2

General: Added: optional star to \OopsNew as instruction to expand keyval list$_1$ . . . 10

Deleted: \OopsTest . . . 10

Deleted: keyval list$_2$ and code$_3$. 10

Deleted: Listing 2-3 from v1.1. . . . 10

Replaced: \OopsClear{⟨*token list*$_1$⟩} by \OopsClear[⟨*keyval list*⟩] . . . 10

Replaced: \Restore by \Read 10

Replaced: \Save by \Write 10

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Part IV
Implementation

```
⟨@@=oops⟩
\NeedsTeXFormat{LaTeX2e}[2019/10/01]
\ExplSyntaxOn
```

1 aux

__oops_aux_key:w

#1 : ⟨ *key* ⟩
#2 : ⟨ *value* ⟩

```
\cs_new_protected:Npn \__oops_aux_key:w #1 = #2 \q_stop
{
  \seq_gput_right:Nx \g__oops_aux_key_seq { \tl_trim_spaces:n{ #1 } }
}
```

(*End definition for* __oops_aux_key:w.)

__oops_aux_key:n #1 : ⟨ *key = value* ⟩

```
\cs_new_protected:Nn \__oops_aux_key:n
{
  \__oops_aux_key:w #1 \q_stop
}
```

(*End definition for* __oops_aux_key:n*.*)

__oops_aux_key:N #1 : ⟨ *seq* ⟩

```
\cs_new_protected:Nn \__oops_aux_key:N
{
  \seq_gclear_new:N \g__oops_aux_key_seq
  \seq_map_function:NN #1 \__oops_aux_key:n
}
```

(*End definition for* __oops_aux_key:N*.*)

__oops_aux_name:n #1 : ⟨ *tl var name* ⟩

```
\cs_new:Nn \__oops_aux_name:n
{
  \tl_gset:Nn \g__oops_name_tl{ #1 }
}
\__oops_aux_name:n
{
  \msg_error:nnx{ __oops }
  { generic }
  { \exp_not:N\g__oops_name_tl~undefined }
}
```

(*End definition for* __oops_aux_name:n*.*)

__oops_aux_prop:w #1 : ⟨ *key* ⟩
#2 : ⟨ *value* ⟩

```
\prop_new:N \g__oops_aux_prop
\cs_new_protected:Nn \__oops_aux_prop:nn
{
  \prop_gput:Nnn \g__oops_aux_prop{ #1 } { #2 }
}
\cs_generate_variant:Nn \__oops_aux_prop:nn { eo }
\cs_new_protected:Npn \__oops_aux_prop:w #1 = #2 \q_stop
{
  \__oops_aux_prop:eo
  { \tl_trim_spaces:n{ #1 } }
  { \__oops_option_inner:e{ \tl_trim_spaces:n{ #2 } } }
% ^^A\prop_gput:Noo \g__oops_aux_prop % v1.1, FAIL with N = N (OK with N= N)
% ^^A  { \tl_trim_spaces:n{ #1 } } { \__oops_option_inner:n{ #2 } }
}
```

(*End definition for* __oops_aux_prop:w*.*)

__oops_aux_prop:n #1 : ⟨ *key = value* ⟩

```
\cs_new_protected:Nn \__oops_aux_prop:n
{
  \__oops_aux_prop:w #1 \q_stop
}
```

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

`\__oops_aux_prop:N`

#1 : ⟨*keyval list*⟩

```
\cs_new_protected:Nn \__oops_aux_prop:N
{
  \prop_gclear_new:N \g__oops_aux_prop
  \seq_if_empty:NTF #1
  { \c_empty_tl }
  {
    \seq_map_function:NN #1 \__oops_aux_prop:n
  }
}
```

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

`\__oops_aux_val:Nn`

#1 : ⟨ *seq* ⟩
#2 : ⟨ *tl var name* ⟩

```
\cs_new_protected:Nn \__oops_aux_val:Nn
{
  \seq_gclear_new:N \__oops_aux_val
  \__oops_seq_from_prop:NNn \__oops_aux_val #1 { \__oops_prop_name:n{ #2 } }
}
```

(*End definition for* `\__oops_aux_val:Nn`.)

2 log

`\__oops_log_close:`

```
\iow_new:N \g__oops_log_iow
\AtEndDocument{\iow_close:N \g__oops_log_iow}
\bool_set_false:N \g__oops_log_open_bool
\cs_new_protected:Nn \__oops_log_close:
{
  \iow_close:N \g__oops_log_iow
  \bool_gset_false:N \g__oops_log_open_bool
}
```

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

`\__oops_log_open:`

```
\cs_new_protected:Nn \__oops_log_open:
{
  \tl_gset:Nx \g__oops_log_to_tl{oops\pdfdate}
  \iow_open:Nn \g__oops_log_iow {\g__oops_log_to_tl}
  \bool_gset_true:N \g__oops_log_open_bool
}
```

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

__oops_log_read:n

#1 : ⟨*path*⟩

```
\cs_new_protected:Nn \__oops_log_read:n
{
  \file_input:n{#1}
  \tl_log:n{read~from~#1}
}
\cs_generate_variant:Nn \__oops_log_read:n { e }
```

(*End definition for* __oops_log_read:n*.*)

__oops_log_read:

```
\cs_new_protected:Nn \__oops_log_read:
{
  \__oops_log_read:e{\g__oops_log_to_tl}
}
```

(*End definition for* __oops_log_read:*.*)

__oops_log_write:n

```
\tl_new:N \g__oops_log_to_tl
\cs_new_protected:Nn \__oops_log_write:n
{
  \bool_if:nTF{ \g__oops_log_open_bool }
  {
    \iow_now:Nn \g__oops_log_iow { #1 }
    \tl_log:n{ write~to~#1 }
  }
  { \msg_error:nnn{ __oops }{ iow }{ \g__oops_log_iow }  }
}
\cs_generate_variant:Nn \__oops_log_write:n { e }
```

(*End definition for* __oops_log_write:n*.*)

3 make

__oops_make_key:Nn

#1 : ⟨ *token* ⟩
#2 : ⟨ *key* ⟩

```
\cs_new_protected:Nn \__oops_make_key:Nn
{
  \exp_args:NNx
  \ProvideDocumentCommand{ #1 }
  { O{\g__oops_name_tl} }
  {
    \__oops_prop_item:nn{ ##1 }{ #2 }
  }
}
\cs_generate_variant:Nn \__oops_make_key:Nn {c}
```

(*End definition for* __oops_make_key:Nn*.*)

__oops_make_key:n

#1 : $\langle$ *key* $\rangle$

```
\cs_new_protected:Nn \__oops_make_key:n
{
  \__oops_make_key:cn{#1}{#1}
}
\cs_generate_variant:Nn \__oops_make_key:n { e }
```

(*End definition for* \__*oops_make_key:n*.)

__oops_make_key:N

#1 : $\langle$ *seq* $\rangle$

```
\cs_new_protected:Nn \__oops_make_key:N
{
  \seq_map_function:NN #1 \__oops_make_key:e
}
```

(*End definition for* \__*oops_make_key:N*.)

__oops_make_new:nnn

#1 : $\langle seq_1 \rangle$
#2 : $\langle seq_2 \rangle$
#3 : $\langle prop \rangle$

```
\cs_new_protected:Npn \__oops_make_new:nnn #1 #2 #3
{
  \exp_args:NNx \DeclareDocumentCommand \OopsNew
  { m +o E{ i }{ { #1 } } m s E{ s o }{ { #2 }{ #3 } } +o }
  {
    \__oops_prop_if_exist:nTF{ ##1 }
    { \c_empty_tl }
    { \__oops_prop_new:n{ ##1 } }
    \exp_args:No \__oops_option_inner_set:n{ ##3 }
    \seq_set_from_clist:Nn \g__oops_aux_keyval_seq { ##4 }
    \__oops_aux_prop:N \g__oops_aux_keyval_seq
    \__oops_prop_append:Nn \g__oops_aux_prop { ##1 }
    \__oops_aux_key:N \g__oops_aux_keyval_seq
    \__oops_make_key:N \g__oops_aux_key_seq
    \bool_if:nTF{ \g__oops_log_open_bool }
    {%^^A https://tex.stackexchange.com/questions/536597
      \__oops_log_write:n
      {
        \begingroup \def \__oops_log_entry { \OopsNew{ ##1 }i{##3}{ ##4 } } \expandafter \end
      }
    }{\c_empty_tl}
    \IfValueT{ ##2 }{ ##2 }
    \IfBooleanT{ ##5 }
    {
      \__oops_aux_val:Nn \g__oops_aux_key_seq { ##1 }
      \__oops_option_outer_set:n{ ##7 }
      \__oops_option_outer:n
      {
        \exp_last_unbraced:NNo
        \seq_use:Nnnn
        \__oops_aux_val
        { ##6 }
      }
    }
    \IfValueT{ ##8 }
    {
      \exp_not:n{ \OopsNew{ ##1 }[ ##8 ] }
    }
  }
}
```

(*End definition for* __*oops_make_new:nnn*.)

4 msg

```
\msg_new:nnn {__oops}{ generic }{ #1 }
\msg_new:nnn {__oops}{ iow }{ #1~is~closed~can’t~write }
\msg_new:nnn {__oops}{ keyonly }{ #1~does~not~take~values;~keyval~is~#2 }
\msg_new:nnn {__oops}{ keywrong }{ #1~does~not~recognize~key~#2 }
\msg_new:nnn {__oops}{ unset }{ #1~unset }
```

5 option

__oops_option_inner_set:n

#1 : ⟨*inlinecode*⟩

```
\cs_new_protected:Nn \__oops_option_inner_set:n
{
  \cs_gset:Npn \__oops_option_inner:n ##1 { #1 }
  \cs_generate_variant:Nn \__oops_option_inner:n { e }
}
\cs_new_protected:Nn \__oops_option_inner_default:n
{
  \tl_gset:Nn \g__oops_option_inner_tl { #1 }
}
\__oops_option_inner_default:n
{
  \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_inner_tl }
}
```

(*End definition for* __*oops_option_inner_set:n*.)

__oops_option_name:n

#1 : ⟨*token list*⟩

```
\cs_new:Nn \__oops_option_name:n
{
  \tl_gset:Nn \g__oops_option_name_tl{ #1 }
}
\__oops_option_name:n
{
  \msg_error:nnx{ __oops }
  { generic }
  { \exp_not:N\g__oops_option_name_tl~undefined }
}
```

(*End definition for* __*oops_option_name:n*.)

__oops_option_outer_set:n
__oops_option_outer_default:n

#1 : ⟨ *inline code* ⟩

```
\cs_new_protected:Nn \__oops_option_outer_set:n
{
  \cs_gset:Npn \__oops_option_outer:n ##1 { #1 }
}
\cs_new_protected:Nn \__oops_option_outer_default:n
{
  \tl_gset:Nn \g__oops_option_outer_tl { #1 }
}
\__oops_option_outer_default:n
{
  \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_outer_tl }
}
```

(*End definition for* __oops_option_outer_set:n *and* __oops_option_outer_default:n.)

__oops_option_separ_default:n

#1 : {⟨ *token list*$_1$ ⟩}{⟨ *token list*$_2$ ⟩}{⟨ *token list*$_3$ ⟩}

```
\cs_new_protected:Nn \__oops_option_separ_default:n
{
  \cs_gset:Npn \g__oops_option_separ_tl { #1 }
}
\__oops_option_separ_default:n
{
  \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_separ_tl }
}
```

(*End definition for* __oops_option_separ_default:n.)

6 prop

__oops_prop_append:NN
__oops_prop_append:cN

#1 : ⟨ *prop*$_1$ ⟩
#2 : ⟨ *prop*$_2$ ⟩

```
\cs_new_protected:Npn \__oops_prop_append:NN #1 #2
{
  \cs_set:Nn \__oops_prop_append:nn
  {
    \prop_gput:Nnx #1 { ##1 }{ \prop_item:Nn #2{ ##1 } }
  }
  \prop_map_function:NN #2 \__oops_prop_append:nn
}
\cs_generate_variant:Nn \__oops_prop_append:NN { cN }
```

(*End definition for* __oops_prop_append:NN.)

__oops_prop_append:Nn

#1 : ⟨ *prop* ⟩
#2 : ⟨ *tl var name* ⟩

```
\cs_new_protected:Nn \__oops_prop_append:Nn
{
  \__oops_prop_append:cN{ \__oops_prop_name:n { #2 } } #1
}
```

(*End definition for* __oops_prop_append:Nn.)

__oops_prop_clear_new:n

#1 : ⟨ *tl var name* ⟩

```
\cs_new_protected:Nn \__oops_prop_clear_new:n
{
  \exp_args:No \prop_clear_new:c{ \__oops_prop_name:n { #1 } }
}
```

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

__oops_prop_clear_new_map:n

#1 : ⟨ *keyval list* ⟩

```
\cs_new_protected:Nn \__oops_prop_clear_new_map:n
{
  \seq_set_from_clist:Nn \g__oops_aux_key_seq { #1 }
  \seq_map_function:NN \g__oops_aux_key_seq \__oops_prop_clear_new:n
}
```

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

__oops_prop_if_exist:nTF

#1 : $\langle \textit{token list}_1 \rangle$

#2 : $\langle \textit{token list}_2 \rangle$

#3 : $\langle \textit{token list}_3 \rangle$

```
\cs_new:Nn \__oops_prop_if_exist:nTF
{
  \prop_if_exist:cTF{ \__oops_prop_name:n { #1 } }{ #2 }{ #3 }
}
```

(*End definition for* `\__oops_prop_if_exist:nTF`*.*)

__oops_prop_item:nn

#1 : ⟨ *tl var name* ⟩

#2 : ⟨ *key* ⟩

```
\cs_new:Nn \__oops_prop_item:nn
{
  \prop_item:cn { \__oops_prop_name:n { #1 } } { #2 }
}
```

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

__oops_prop_name:n

#1 : ⟨ *tl var name* ⟩

```
\cs_new:Npn \__oops_prop_name:n #1{ __oops_#1 }
```

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

__oops_prop_new:n

#1 : ⟨ *tl var name* ⟩

```
\cs_new_protected:Nn \__oops_prop_new:n
{
  \prop_new:c{ \__oops_prop_name:n { #1 } }
}
```

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

7 seq

__oops_seq_from_prop:NNn

#1 : $\langle\, seq_1 \,\rangle$
#2 : $\langle\, seq_2 \,\rangle$ (keys)
#3 : $\langle\, prop \,\rangle$

```
\cs_new_protected:Nn \__oops_seq_from_prop:NNn
{
  \cs_set_protected:Nn \__oops_seq_from_prop:n
  {
    \seq_gput_right:No #1 { \prop_item:cn{ #3 }{ ##1 } }
  }
  \seq_map_function:NN #2 \__oops_seq_from_prop:n
}
```

(*End definition for* __oops_seq_from_prop:NNn.)

8 Front-end

```
\keys_define:nn { __oops }
{
  Name .code:n={
    \__oops_aux_name:n{ #1 }
  },
  Name .value_required:n = false,
  Name .default:n = { Math },
  Name .initial:n = { Math },
  Inner .code:n={
    \__oops_option_inner_default:n{ #1 }
    \exp_last_unbraced:Nf
    \__oops_make_new:nnn
    {
      { \g__oops_option_inner_tl }
      { \g__oops_option_separ_tl }
      { \g__oops_option_outer_tl }
    }
  },
  Inner .value_required:n = false,
  Inner .default:n = { ####1 },
  Inner .initial:n = { ####1 },
  Outer .code:n={
    \__oops_option_outer_default:n{ #1 }
    \exp_last_unbraced:Nf
    \__oops_make_new:nnn
    {
      { \g__oops_option_inner_tl }
      { \g__oops_option_separ_tl }
      { \g__oops_option_outer_tl }
    }
  },
  Outer .value_required:n = false,
  Outer .default:n = { \ensuremath{####1} },
  Outer .initial:n = { \ensuremath{####1} },
  Write .code:n = {
    \bool_if:nTF{#1}
    {\__oops_log_open:}
    {\__oops_log_close:}
  },
  Write .value_required:n = false,
  Write .default:n = \BooleanFalse,
  Write .initial:n = \BooleanFalse,
  Separ .code:n={
    \__oops_option_separ_default:n{ #1 }
    \exp_last_unbraced:Nf
    \__oops_make_new:nnn
    {
      { \g__oops_option_inner_tl }
      { \g__oops_option_separ_tl }
      { \g__oops_option_outer_tl }
    }
  },
  Separ .value_required:n = false,
  Separ .default:n = { { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } } },
  Separ .initial:n = { { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } } }
}
```

\OopsClear #1 : ⟨ *tl var name* ⟩

```
\NewDocumentCommand{ \OopsClear }
{ O{\g__oops_name_tl} }
{
  \__oops_prop_clear_new_map:n{ #1 } % TODO record
}
```

(*End definition for* `\OopsClear`. *This function is documented on page 3.*)

\OopsOption

```
\NewDocumentCommand{ \OopsOption }
{ m }
{
  \keys_set:nn{ __oops }{ #1 } % TODO record
}
```

(*End definition for* `\OopsOption`. *This function is documented on page 4.*)

\OopsRead

```
\NewDocumentCommand{\OopsRead}
{o}
{
  \IfValueTF{#1}
  {\__oops_log_read:e{#1}}
  {\__oops_log_read:}
}
```

(*End definition for* `\OopsRead`. *This function is documented on page 5.*)

9 Misc

```
\ExplSyntaxOff
```