

oops, an object oriented practical scribe’s package.*

Erwann Rogard†

Released 2020/04/04

Abstract

`oops` is a package for L^AT_EX (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, `\OpsNew{⟨token list₁⟩}`, where `⟨token list₁⟩` identifies an object, begins a series of instructions alternating between ‘text’ and definitions, that themselves optionally expand using predefined or inline rules. For example,

`\OpsNew[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 list₁⟩`, `Space` also works. Optionally, the definitions can be saved to a file, and restored, which can be useful for typesetting documents sharing the same notational conventions. Altogether, “practical”.

Contents

I	Usage	3
1	Convention	3
2	Loading the package	3
3	<code>\OpsClear</code>	3
4	<code>\OpsNew</code>	3
4.1	<code>{⟨token list₁⟩}</code>	4
4.2	<code>[⟨token list₂⟩]</code>	4
4.3	<code>i{⟨code₁⟩}</code>	4
4.4	<code>s{⟨token list₃⟩}{⟨token list₄⟩}{⟨token list₅⟩}</code>	4
4.5	<code>o{⟨code₂⟩}</code>	4
4.6	<code>{⟨keyval list₁⟩}</code>	4
4.7	Remainder	5

*This file describes version v1.1, last revised 2020/04/04.

†firstname dot lastname AusTria gmail dot com

5	\OpsOption	5
5.8	Inner	5
5.9	Name	5
5.10	Outer	5
5.11	Save	5
5.12	Separ	5
6	\OpsRestore	5
7	\OpsTest	5
7.13	A	5
7.14	B	5
II	Listing	6
	Listing 1.	6
	Listing 2.	6
	Listing 3.	6
	Listing 4.	7
	Listing 5.	7
	Listing 6.	7
	Listing 7.	7
	Listing 8.	7
	Listing 9.	8
III	Other	9
1	Acknowledgment	9
2	Bug	9
3	Install	9
4	Support	9
5	Unit testing	9
	Change History	10
	Index	10

IV	Implementation	12
1	aux	12
2	log	14
3	make	15
4	msg	17
5	option	17
6	prop	18
7	seq	19
8	test	20
9	Front-end	20
10	Misc	22

Part I

Usage

This part describes .

Convention

1. Loosely, those of [2] and [3], for example as to the meaning of $\langle token\ list \rangle$ and -NoValue-.
2. If unspecified, the environment in which a function must be declared is **document**.

<code>\usepackage</code>	<code>\usepackage{oops}</code>
--------------------------	--------------------------------

Environment Preamble

Requirement `oops.sty` is in the path of the L^AT_EX engine. See [Part III, section 4](#).

<code>\OpsClear</code>	<code>\OpsClear[\langle token\ list_1 \rangle]</code>
------------------------	---

<code>\OpsNew</code>	<code>\OpsNew{⟨token list₁⟩}</code> <code>[⟨token list₂⟩]</code> <code>i{⟨code₁⟩}</code> <code>s{⟨⟨token list₃⟩⟩{⟨token list₄⟩⟩{⟨token list₅⟩⟩}}</code> <code>o{⟨code₂⟩}</code> <code>{⟨keyval list₁⟩}</code> <code>i{⟨code₃⟩}</code> <code><⟨keyval list₂⟩></code> <code>[⟨token list₆⟩]</code>
----------------------	--

Requirement $\langle \text{token list}_1 \rangle$ and $\langle \text{keyval list}_1 \rangle$ are mandatory.

$\langle \text{token list}_1 \rangle$

Example Math, ModelA, ModelB

Semantics Registers a new object, if applicable

$\langle \text{token list}_2 \rangle$

Example Let~

Semantics Expands to $\langle \text{token list}_2 \rangle$

$\langle \text{code}_1 \rangle$

Example `\mathbb{#1}`

$\langle \text{token list}_3 \rangle$

Example `{~\&~}`

$\langle \text{token list}_4 \rangle$

Example `{,~}`

$\langle \text{token list}_5 \rangle$

Example `{~\&~}`

$\langle \text{code}_2 \rangle$

Example `\text{#1}`

$\langle \text{keyval list}_1 \rangle$

Example Sample= Ω

Semantics 1. Defines $\backslash \langle \text{key}_i \rangle [\langle \text{token list}_1 \rangle]$ as $\langle \text{code}_1 \rangle$ applied to $\langle \text{val}_i \rangle$.

2. If `Save=\BooleanTrue`,
writes `\OpsNew{⟨token list₁⟩i{⟨code₁⟩}{⟨keyval list₁⟩}` to file `oops⟨digits⟩.tex`,
where $\langle \text{digits} \rangle = \text{\pdfdate}$

3. If $\langle \text{token list}_2 \rangle \neq \text{-NoValue-}$,
expands to $\langle \text{code}_2 \rangle$ applied to the list created in 1.,
using $\{ \langle \text{token list}_3 \rangle \} \{ \langle \text{token list}_4 \rangle \} \{ \langle \text{token list}_5 \rangle \}$ as separator.

$\langle code_3 \rangle$, $\langle keyval list_2 \rangle$,
and $\langle token list_6 \rangle$ **Semantics** $\backslash OpsNew\{\langle token list_1 \rangle\}i\{\langle code_3 \rangle\}\{\langle keyval list_2 \rangle\}[\langle token list_6 \rangle]$

$\backslash OpsOption$ $\backslash OpsOption\{\langle kv10 \rangle\}$

Semantics Set default options for $\backslash OpsNew$

Inner

Semantics Default for $\langle code_1 \rangle$

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

Name

Semantics Default for $\langle token list_1 \rangle$

Outer

Semantics Default for $\langle code_2 \rangle$

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

Save

Syntax $\langle boolean \rangle$

Separ

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

Semantics Default for $\{\langle token list_3 \rangle\}\{\langle token list_4 \rangle\}\{\langle token list_5 \rangle\}$

$\backslash OpsRestore$ $\backslash OpsRestore[\langle path \rangle]$

Semantics Restores the definitions saved in $\langle path \rangle$ and writes to `oops.log`: ‘restore $\langle path \rangle$ ’

$\backslash OpsTest$ $\backslash OpsTest\{A|B\}\{\langle arg_1 \rangle\}$

A

Semantics $\backslash OpsClear\{Test\}\backslash OpsNew\{Test\}[A]\langle arg_1 \rangle\{ X = x, Y = y, Z = z \}$

B

Semantics $\backslash OpsNew\{Test\}[B]\{ W = w, X = x \}\langle arg_1 \rangle < Y = y, Z = z >$

Part II

Listing

Warning: To reproduce the listings in a L^AT_EX 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.

```
% \OpsOption{
% Inner={\{####1\}},
% ^^A% spaces betw. inner and outer brackets matter!->
% Separ={\ \char`@\ }{\ \char`@\ },
% Outer={\char`^####1\$}}
% \OpsTest{A}{\ \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsTest{A}{ i{(#1)} \ \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsTest{A}{ s{\ \&\ }{\ \&\ }} \\\
% \OpsTest{A}{ o{\char`[#1\char`]} \ }
```

A: $\{x\}\% \{y\} @ \{z\}$ $\{x\}\{y\}\{z\}$
A: $\hat{(x)}\% (y) @ (z)$ $(x)(y)(z)$
A: $\hat{\{x\}}, \{y\} \& \{z\}$
A: $\{\{x\}\% \{y\} @ \{z\}\}$

Listing 2.

```
% \OpsOption{ Inner, Separ, Outer }
% \OpsTest{A}{\ \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsTest{A}{ i{(#1)} \ \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsTest{A}{ s{\ \&\ }{\ \&\ }} \\\
% \OpsTest{A}{ o{\char`[#1\char`]} \ }
```

A: $x, y, \text{ and } z$ xyz
A: $(x), (y), \text{ and } (z)$ $(x)(y)(z)$
A: $x, y \& z$
A: $[x, y, \text{ and } z]$

Listing 3.

```
% \OpsTest{B}{\ \tab \W[Test]\X[Test]\Y[Test]\Z[Test]\\
% \OpsOption{ Save = \BooleanTrue }
% \OpsTest{B}{ i{(#1)} \ \tab \W[Test]\X[Test]\Y[Test]\Z[Test]
% \OpsOption{ Save = \BooleanFalse }
```

¹For 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`

%

B: w and x
B: w and x

$wxyz$
 $wx(y)(z)$

Listing 4.

```
% \OpsRestore \tab\W[Test]\X[Test]\Y[Test]\Z[Test]
%
```

$wx(y)(z)$

Listing 5.

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

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

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

the sample space.

Listing 6.

```
% \OpsOption{ Save = \BooleanTrue }
% \OpsNew{Math}[Let ]s{{,},{,},{,}}of{\ensuremath{\{\#1\}}}
% {Space=\Omega, SigmaField=\mathcal{F}, Measure=\mathcal{P}}
% [~denote the probability space, where $\SigmaField\subset
2^{\Space}$.]
% {}
% \OpsClear
% \OpsOption{ Save = \BooleanFalse }
%
```

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

Listing 7.

```
% \OpsRestore \tab $\Omega$ $\SigmaField$ $\Measure$
% \OpsClear
%
```

$\Omega \mathcal{F} \mathcal{P}$

Listing 8.

```
% \OpsOption{ Save = \BooleanTrue }
% \newtheorem{theorem}{Theorem}
% \OpsNew{Math}{i{\mathbb{#1}}}{ N = { N } , R = { R } }
% [\begin{theorem}[Mittelwertsatz f\"ur $n$ Variable]Es-sei~]{ }
% <OffeneMenge={D}, Ci={C^{1}}, Strecke={ [x_0,x] }>
% ^A% Strecke={\char`[x_0,x\char`]} % PASS
% ^A% Strecke={ [x_0,x] } % FAIL
% [$n\in\mathbb{N}, ~\mathbb{D}\subseteq\mathbb{R}^n$ eine offene Menge und
% $f\in C^1(\mathbb{D},\mathbb{R})$. Dann gibt es auf jeder Strecke
% $\gamma\subset\mathbb{D}$ einen Punkt $\xi\in\gamma$,~]{ }
% <yDifferenz={f(x)-f(x_0)},
% xDifferenz={x-x_0},
% Steigung={\frac{yDifferenz}{xDifferenz}}>
% [so dass gilt
% \begin{equation*}\text{Steigung} = \operatorname{grad}
% f(\xi)^{\top}\end{equation*}
% \end{theorem}]{ }
% \OpsClear
% \OpsOption{ Save = \BooleanFalse }
```

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

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

Listing 9.

```
% \OpsRestore \tab $\mathbb{N}$ $\mathbb{R}$ $\mathbb{D}\subseteq\mathbb{R}^n$ $C^1$ $[x_0,x]$
%
```

$$\mathbb{N} \quad \mathbb{R} \quad D \subseteq \mathbb{R}^n \quad C^1 \quad [x_0, x]$$

Part III

Other

1 Acknowledgment

This work has benefited from Q&A's from the L^AT_EX community, see here: <https://tex.stackexchange.com/users/112708/erwann?tab=questions>. Specific references are made in Part IV. Listing 5 and Listing 6 are from [1]. Listing 8 is from tcolorbox[4, 17.3].

2 Bug

1. Some characters don't work for $\langle \text{keyval } list_1 \rangle$, but there are workarounds, see Listing 8.

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> (on the source, a.k.a dtx, file) 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 L^AT_EX3 Project Team *The L^AT_EX3 interfaces* <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/interface3.pdf>
- [3] The L^AT_EX3 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		Replaced:	
General: Initial version	9	\OpsOptions by \OpsOption	9
v1.1		Replaced: {<keyval	
General: Added: Save	9	list ₂ >} by <keyval list ₂ > given	
Added: Listing 1, Listing 2, Listing		that option type G not	
3, Listing 4, Listing 7 and Listing 9	9	recommended[3]	9
Added:\OpsRestore	9	Replaced: GenericObject by Name	9
Added:\OpsTest	9	Replaced: Separators by Separ	9
Deleted: Listing 1-5 from v1.0	9	Revamped: much of the	
Fixed: apparent anomaly in v1.0's		implementation	9
Listing 4, see Listing 1	9		

Index

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

Symbols		\bool_set_false:N	62
<code ₁ > (option)	4	\BooleanFalse	317, 318
<code ₂ > (option)	4	\BooleanTrue	4
<code ₃ >, <keyval list ₂ >, and <token			
list ₆ > (option)	5	C	
<keyval list ₁ > (option)	4	cs commands:	
<token list ₁ > (option)	4	\cs_generate_variant:Nn	32, 79, 91, 101, 106, 219
<token list ₂ > (option)	4	\cs_gset:Npn	171, 193, 205
<token list ₃ > (option)	4	\cs_new:Nn	17, 181, 228, 232, 261
<token list ₄ > (option)	4	\cs_new:Npn	236
<token list ₅ > (option)	4	\cs_new_protected:Nn	8,
A (option)	5	12, 28, 42, 46, 55, 63, 68, 74, 80,	
B (option)	5	85, 92, 102, 107, 169, 173, 191,	
Inner (option)	5	195, 203, 220, 224, 237, 241, 249, 254	
Name (option)	5	\cs_new_protected:Npn	4, 33, 111, 211
Save (option)	5	\cs_set:Nn	213
Separ (option)	5	\cs_set_protected:Nn	243
		D	
_	330, 331	\DeclareDocumentCommand	113
A		\def	130
\AtEndDocument	61	\documentclass	6
B		E	
\begingroup	130	\endgroup	130
bool commands:		\ensuremath	309, 310
\bool_gset_false:N	66	exp commands:	
\bool_gset_true:N	72	\exp_args:NNx	94, 113
\bool_if:nTF	87, 125, 312	\exp_args:No	119

\exp_last_unbraced:Nf .	287, 300, 321	\g__oops_log_iow .	60, 61, 65, 71, 88, 89
\exp_last_unbraced:NNo	140	__oops_log_open:	68, 313
\exp_not:N	25, 179, 189, 201, 209	\g__oops_log_open_bool	
\exp_not:n	150, 153, 159	62, 66, 72, 87, 125
\expandafter	130	__oops_log_restore:	80, 348
\ExplSyntaxOff	354	__oops_log_restore:n	74, 82, 347
\ExplSyntaxOn	3	__oops_log_save:n	84, 128
F			
file commands:			
\file_input:n	76	\g__oops_log_to_tl	70, 71, 82, 84
I			
\IfValueT	133, 157	__oops_make_key:N	107, 124
\IfValueTF	146, 148, 346	__oops_make_key:n	102, 109
iow commands:			
\iow_close:N	61, 65	__oops_make_key:Nn	92, 104
\iow_new:N	60	__oops_make_new:nnn	111, 288, 301, 322
\iow_now:Nn	88	\g__oops_name_tl	19, 25, 96, 334
\iow_open:Nn	71	__oops_option_inner:n	119, 169
K			
keys commands:			
\keys_define:nn	277	__oops_option_inner:n	37, 40, 171
\keys_set:nn	341	__oops_option_inner_default:n	
M			
msg commands:			
\msg_error:nnn	23, 89, 187	173, 177, 286
\msg_error:nnnn	271	\g__oops_option_inner_tl	
\msg_new:nnn	164, 165, 166, 167, 168	175, 179, 290, 303, 324
\msg_warning:nnn	179, 201, 209	__oops_option_name:n	181
N			
\NeedsTeXFormat	2	\g__oops_option_name_tl	183, 189
\NewDocumentCommand	333, 338, 343, 350	__oops_option_outer:n	136, 191
\newtcblisting	6	__oops_option_outer:n	138, 193
O			
oops internal commands:			
__oops_aux_key:N	12, 123	__oops_option_outer_default:n	
__oops_aux_key:n	8, 15	191, 299
__oops_aux_key:w	4, 10	\g__oops_option_outer_tl	
\g__oops_aux_key_seq	6, 14, 124, 135	197, 201, 292, 305, 326
\g__oops_aux_keyval_seq	120, 121, 123	__oops_option_separ_default:n	
__oops_aux_name:n	17, 280	203, 320
\g__oops_aux_prop	27, 30, 39, 48, 122	\g__oops_option_separ_tl	
__oops_aux_prop:N	46, 121	205, 209, 291, 304, 325
__oops_aux_prop:n	42, 52	__oops_prop_append:NN	211, 222
__oops_aux_prop:nn	28, 32, 35	__oops_prop_append:Nn	122, 220
__oops_aux_prop:w	27, 44	__oops_prop_append:nn	213, 217
__oops_aux_val	57, 58, 142	__oops_prop_clear_new:n	224, 336
__oops_aux_val:Nn	55, 135	__oops_prop_if_exist:nTF	116, 228
__oops_log_close:	60, 314	__oops_prop_item:nn	98, 232
__oops_log_entry	130	__oops_prop_name:n	
		58, 222, 226, 230, 234, 236, 239
		__oops_prop_new:n	118, 237
		__oops_seq_from_prop:n	243, 247
		__oops_seq_from_prop:Nn	58, 241
		__oops_test:n	273
		__oops_test:nn	259, 352
		\c__oops_test_a	259, 266
		__oops_test_a:n	249, 266
		\c__oops_test_b	260, 267
		__oops_test_b:n	254, 267
		\OpsClear	1, 3, 5, 251, 256, 333
		\OpsNew	1, 4, 4, 5, 5,
			5, 5, 113, 130, 150, 153, 159, 252, 257
		\OpsOption	2, 5, 10, 338
		\OpsOptions	10
		\OpsRestore	2, 5, 10, 343


```

9 {
10   \__oops_aux_key:w #1 \q_stop
11 }

```

(End definition for __oops_aux_key:n.)

```

\__oops_aux_key:N #1: < seq >
12 \cs_new_protected:Nn \__oops_aux_key:N
13 {
14   \seq_gclear_new:N \g__oops_aux_key_seq
15   \seq_map_function:NN #1 \__oops_aux_key:n
16 }

```

(End definition for __oops_aux_key:N.)

```

\__oops_aux_name:n #1: < tl var name >
17 \cs_new:Nn \__oops_aux_name:n
18 {
19   \tl_gset:Nn \g__oops_name_tl{ #1 }
20 }
21 \__oops_aux_name:n
22 {
23   \msg_error:nnx{ __oops }
24   { generic }
25   { \exp_not:N\g__oops_name_tl~undefined }
26 }

```

(End definition for __oops_aux_name:n.)

```

\__oops_aux_prop:w #1: < key >
#2: < value >
27 \prop_new:N \g__oops_aux_prop
28 \cs_new_protected:Nn \__oops_aux_prop:nn
29 {
30   \prop_gput:Nnn \g__oops_aux_prop{ #1 } { #2 }
31 }
32 \cs_generate_variant:Nn \__oops_aux_prop:nn { eo }
33 \cs_new_protected:Npn \__oops_aux_prop:w #1 = #2 \q_stop
34 {
35   \__oops_aux_prop:eo
36   { \tl_trim_spaces:n{ #1 } }
37   { \__oops_option_inner:n{ #2 } }
38   % ^^A\prop_gput:Noo \g__oops_aux_prop % v1.1, FAIL with N = N (OK with N= N)
39   % ^^A { \tl_trim_spaces:n{ #1 } } { \__oops_option_inner:n{ #2 } }
40 }

```

(End definition for __oops_aux_prop:w.)

```

\__oops_aux_prop:n #1: < key = value >
41 \cs_new_protected:Nn \__oops_aux_prop:n
42 {
43   \__oops_aux_prop:w #1 \q_stop
44 }

```

(End definition for __oops_aux_prop:n.)

```

\__oops_aux_prop:N #1 :  $\langle keyval list \rangle$ 
45 \cs_new_protected:Nn \__oops_aux_prop:N
46 {
47   \prop_gclear_new:N \g__oops_aux_prop
48   \seq_if_empty:NTF #1
49   { \c_empty_tl }
50   {
51     \seq_map_function:NN #1 \__oops_aux_prop:n
52   }
53 }

```

(End definition for __oops_aux_prop:N.)

```

\__oops_aux_val:Nn #1 :  $\langle seq \rangle$ 
#2 :  $\langle tl var name \rangle$ 
54 \cs_new_protected:Nn \__oops_aux_val:Nn
55 {
56   \seq_gclear_new:N \__oops_aux_val
57   \__oops_seq_from_prop:NNn \__oops_aux_val #1 { \__oops_prop_name:n{ #2 } }
58 }

```

(End definition for __oops_aux_val:Nn.)

2 log

```

\__oops_log_close:
59 \iow_new:N \g__oops_log_iow
60 \AtEndDocument{\iow_close:N \g__oops_log_iow}
61 \bool_set_false:N \g__oops_log_open_bool
62 \cs_new_protected:Nn \__oops_log_close:
63 {
64   \iow_close:N \g__oops_log_iow
65   \bool_gset_false:N \g__oops_log_open_bool
66 }

```

(End definition for __oops_log_close:.)

```

\__oops_log_open:
67 \cs_new_protected:Nn \__oops_log_open:
68 {
69   \tl_gset:Nx \g__oops_log_to_tl{oops\pdfdate}
70   \iow_open:Nn \g__oops_log_iow {\g__oops_log_to_tl}
71   \bool_gset_true:N \g__oops_log_open_bool
72 }

```

(End definition for __oops_log_open:.)

```

\__oops_log_restore:n #1 :  $\langle path \rangle$ 
73 \cs_new_protected:Nn \__oops_log_restore:n
74 {
75   \file_input:n{#1}
76   \tl_log:n{restore~#1}
77 }
78 \cs_generate_variant:Nn \__oops_log_restore:n { e }

```

(End definition for _oops_log_restore:n.)

_oops_log_restore:

```

79 \cs_new_protected:Nn \_oops_log_restore:
80 {
81   \_oops_log_restore:e{\g__oops_log_to_tl}
82 }

```

(End definition for _oops_log_restore:.)

_oops_log_save:n

```

83 \tl_new:N \g__oops_log_to_tl
84 \cs_new_protected:Nn \_oops_log_save:n
85 {
86   \bool_if:nTF{ \g__oops_log_open_bool }
87   { \iow_now:Nn \g__oops_log_iow { #1 } }
88   { \msg_error:nnn{ __oops }{ iow }{ \g__oops_log_iow } }
89 }
90 \cs_generate_variant:Nn \_oops_log_save:n { e }

```

(End definition for _oops_log_save:n.)

3 make

_oops_make_key:Nn #1 : $\langle token \rangle$
 #2 : $\langle key \rangle$

```

91 \cs_new_protected:Nn \_oops_make_key:Nn
92 {
93   \exp_args:NNx
94   \ProvideDocumentCommand{ #1 }
95   { 0{\g__oops_name_tl} }
96   {
97     \_oops_prop_item:nn{ ##1 }{ #2 }
98   }
99 }
100 \cs_generate_variant:Nn \_oops_make_key:Nn { c }

```

(End definition for _oops_make_key:Nn.)

_oops_make_key:n #1 : $\langle key \rangle$

```

101 \cs_new_protected:Nn \_oops_make_key:n
102 {
103   \_oops_make_key:cn{#1}{#1}
104 }
105 \cs_generate_variant:Nn \_oops_make_key:n { e }

```

(End definition for _oops_make_key:n.)

_oops_make_key:N #1 : $\langle seq \rangle$

```

106 \cs_new_protected:Nn \_oops_make_key:N
107 {
108   \seq_map_function:NN #1 \_oops_make_key:e
109 }

```

(End definition for _oops_make_key:N.)

```

\_oops\_make\_new:nnn #1 : < seq1 >
#2 : < seq2 >
#3 : < prop >

110 \cs\_new\_protected:Npn \_oops\_make\_new:nnn #1 #2 #3
111 {
112   \exp\_args:NNx \DeclareDocumentCommand \OpsNew
113   { m +o E{ i s o }{ { #1 }{ #2 }{ #3 } } m E{ i }{ { #1 } } d<> +o }
114   {
115     \_oops\_prop\_if\_exist:nTF{ ##1 }
116     { \c\_empty\_tl }
117     { \_oops\_prop\_new:n{ ##1 } }
118     \exp\_args:No \_oops\_option\_inner:n{ ##3 }
119     \seq\_set\_from\_clist:Nn \g\_oops\_aux\_keyval\_seq { ##6 }
120     \_oops\_aux\_prop:N \g\_oops\_aux\_keyval\_seq
121     \_oops\_prop\_append:Nn \g\_oops\_aux\_prop { ##1 }
122     \_oops\_aux\_key:N \g\_oops\_aux\_keyval\_seq
123     \_oops\_make\_key:N \g\_oops\_aux\_key\_seq
124     \bool\_if:nTF{ \g\_oops\_log\_open\_bool }
125     {
126       %^^A https://tex.stackexchange.com/questions/536597
127       \_oops\_log\_save:n
128       {
129         \begingroup \def \_oops\_log\_entry { \OpsNew{ ##1 }i{##3}{ ##6 } } \expandafter \endgroup
130       }
131     }{\c\_empty\_tl}
132     \IfValueT{ ##2 }
133     {
134       \_oops\_aux\_val:Nn \g\_oops\_aux\_key\_seq { ##1 }
135       \_oops\_option\_outer:n{ ##5 }
136       ##2
137       \_oops\_option\_outer_:n
138       {
139         \exp\_last\_unbraced:NNo
140         \seq\_use:Nnnn
141         \_oops\_aux\_val
142         { ##4 }
143       }
144     }
145     \IfValueTF{ ##8 }
146     {
147       \IfValueTF{ ##9 }
148       {
149         \exp\_not:n{ \OpsNew{ ##1 }i{ ##7 }{ ##8 }[ ##9 ] }
150       }
151       {
152         \exp\_not:n{ \OpsNew{ ##1 }i{ ##7 }{ ##8 } }
153       }
154     }
155     {
156       \IfValueT{##9}
157       {
158         \exp\_not:n{ \OpsNew{ ##1 }[ ##9 ] }

```



```

159 }
160 }
161 }
162 }

```

(End definition for `__oops_make_new:nnn`.)

4 msg

```

163 \msg_new:nnn {\__oops}{ generic }{ #1 }
164 \msg_new:nnn {\__oops}{ iow }{ #1~is~closed~can't~save }
165 \msg_new:nnn {\__oops}{ keyonly }{ #1~does~not~take~values;~keyval~is~#2 }
166 \msg_new:nnn {\__oops}{ keywrong }{ #1~does~not~recognize~key~#2 }
167 \msg_new:nnn {\__oops}{ unset }{ #1~unset }

```

5 option

`__oops_option_inner:n` #1: *<inlinecode>*

```

168 \cs_new_protected:Nn \__oops_option_inner:n
169 {
170   \cs_gset:Npn \__oops_option_inner_:n ##1 { #1 }
171 }
172 \cs_new_protected:Nn \__oops_option_inner_default:n
173 {
174   \tl_gset:Nn \g__oops_option_inner_tl { #1 }
175 }
176 \__oops_option_inner_default:n
177 {
178   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_inner_tl }
179 }

```

(End definition for `__oops_option_inner:n`.)

`__oops_option_name:n` #1: *<token list>*

```

180 \cs_new:Nn \__oops_option_name:n
181 {
182   \tl_gset:Nn \g__oops_option_name_tl{ #1 }
183 }
184 \__oops_option_name:n
185 {
186   \msg_error:nnx{ __oops }
187   { generic }
188   { \exp_not:N \g__oops_option_name_tl~undefined }
189 }

```

(End definition for `__oops_option_name:n`.)

`__oops_option_outer:n` #1: *<inline code>*

```

\__oops_option_outer_default:n
190 \cs_new_protected:Nn \__oops_option_outer:n
191 {
192   \cs_gset:Npn \__oops_option_outer_:n ##1 { #1 }
193 }
194 \cs_new_protected:Nn \__oops_option_outer_default:n

```

```

195 {
196   \tl_gset:Nn \g__oops_option_outer_tl { #1 }
197 }
198 \__oops_option_outer_default:n
199 {
200   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_outer_tl }
201 }

```

(End definition for __oops_option_outer:n and __oops_option_outer_default:n.)

```

\__oops_option_separ_default:n #1 : { \ token list1 } { \ token list2 } { \ token list3 }
202 \cs_new_protected:Nn \__oops_option_separ_default:n
203 {
204   \cs_gset:Npn \g__oops_option_separ_tl { #1 }
205 }
206 \__oops_option_separ_default:n
207 {
208   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_separ_tl }
209 }

```

(End definition for __oops_option_separ_default:n.)

6 prop

```

\__oops_prop_append:NN #1 : \ prop1 \
\__oops_prop_append:cN #2 : \ prop2 \
210 \cs_new_protected:Npn \__oops_prop_append:NN #1 #2
211 {
212   \cs_set:Nn \__oops_prop_append:nn
213   {
214     \prop_gput:Nnx #1 { ##1 } { \prop_item:Nn #2 { ##1 } }
215   }
216   \prop_map_function:NN #2 \__oops_prop_append:nn
217 }
218 \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 \
219 \cs_new_protected:Nn \__oops_prop_append:Nn
220 {
221   \__oops_prop_append:cN{ \__oops_prop_name:n { #2 } } #1
222 }

```

(End definition for __oops_prop_append:Nn.)

```

\__oops_prop_clear_new:n #1 : \ tl var name \
223 \cs_new_protected:Nn \__oops_prop_clear_new:n
224 {
225   \prop_clear_new:c{ \__oops_prop_name:n { #1 } }
226 }

```

(End definition for `__oops_prop_clear_new:n`.)

```
\__oops_prop_if_exist:nTF #1 :  $\langle token\ list_1 \rangle$ 
                          #2 :  $\langle token\ list_2 \rangle$ 
                          #3 :  $\langle token\ list_3 \rangle$ 
227 \cs_new:Nn \__oops_prop_if_exist:nTF
228 {
229   \prop_if_exist:cTF{ \__oops_prop_name:n { #1 } }{ #2 }{ #3 }
230 }
```

(End definition for `__oops_prop_if_exist:nTF`.)

```
\__oops_prop_item:nn #1 :  $\langle tl\ var\ name \rangle$ 
                     #2 :  $\langle key \rangle$ 
231 \cs_new:Nn \__oops_prop_item:nn
232 {
233   \prop_item:cn { \__oops_prop_name:n { #1 } } { #2 }
234 }
```

(End definition for `__oops_prop_item:nn`.)

```
\__oops_prop_name:n #1 :  $\langle tl\ var\ name \rangle$ 
235 \cs_new:Npn \__oops_prop_name:n #1{ __oops_#1 }
```

(End definition for `__oops_prop_name:n`.)

```
\__oops_prop_new:n #1 :  $\langle tl\ var\ name \rangle$ 
236 \cs_new_protected:Nn \__oops_prop_new:n
237 {
238   \prop_new:c{ \__oops_prop_name:n { #1 } }
239 }
```

(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 :  $\langle prop \rangle$ 
240 \cs_new_protected:Nn \__oops_seq_from_prop:NNn
241 {
242   \cs_set_protected:Nn \__oops_seq_from_prop:n
243   {
244     \seq_gput_right:No #1 { \prop_item:cn{ #3 }{ ##1 } }
245   }
246   \seq_map_function:NN #2 \__oops_seq_from_prop:n
247 }
```

(End definition for `__oops_seq_from_prop:NNn`.)

8 test

`__oops_test_a:n`

```

248 \cs_new_protected:Nn \__oops_test_a:n
249 {
250   \OupsClear[Test]
251   \OupsNew{Test}[A:~]#1{ X = x, Y = y, Z = z }
252 }
```

(End definition for __oops_test_a:n.)

`__oops_test_b:n`

```

253 \cs_new_protected:Nn \__oops_test_b:n
254 {
255   \OupsClear[Test]
256   \OupsNew{Test}[B:~]{ W = w, X = x }#1< Y = y, Z = z >
257 }
```

(End definition for __oops_test_b:n.)

`__oops_test:nn`

```

258 \tl_const:Nn \c__oops_test_a { A }
259 \tl_const:Nn \c__oops_test_b { B }
260 \cs_new:Nn \__oops_test:nn
261 {
262   \tl_set:Nn \l_tmpa_tl { #1 }
263   \tl_case:NnTF \l_tmpa_tl
264   {
265     \c__oops_test_a { \__oops_test_a:n{#2} }
266     \c__oops_test_b { \__oops_test_b:n{#2} }
267   }
268   { \c_empty_tl }
269   {
270     \msg_error:nnnn{ __oops }
271     { keywrong }
272     { \__oops_test:n }
273     { #1 }
274   }
275 }
```

(End definition for __oops_test:nn.)

9 Front-end

```

276 \keys_define:nn { __oops }
277 {
278   Name .code:n={
279     \__oops_aux_name:n{ #1 }
280   },
281   Name .value_required:n = false,
282   Name .default:n = { Math },
283   Name .initial:n = { Math },
284   Inner .code:n={
285     \__oops_option_inner_default:n{ #1 }
```

```

286 \exp_last_unbraced:Nf
287 \__oops_make_new:nnn
288 {
289   { \g__oops_option_inner_tl }
290   { \g__oops_option_separ_tl }
291   { \g__oops_option_outer_tl }
292 }
293 },
294 Inner .value_required:n = false,
295 Inner .default:n = { #####1 },
296 Inner .initial:n = { #####1 },
297 Outer .code:n={
298   \__oops_option_outer_default:n{ #1 }
299   \exp_last_unbraced:Nf
300   \__oops_make_new:nnn
301   {
302     { \g__oops_option_inner_tl }
303     { \g__oops_option_separ_tl }
304     { \g__oops_option_outer_tl }
305   }
306 },
307 Outer .value_required:n = false,
308 Outer .default:n = { \ensuremath{#####1} },
309 Outer .initial:n = { \ensuremath{#####1} },
310 Save .code:n = {
311   \bool_if:nTF{#1}
312   {\__oops_log_open:}
313   {\__oops_log_close:}
314 },
315 Save .value_required:n = false,
316 Save .default:n = \BooleanFalse,
317 Save .initial:n = \BooleanFalse,
318 Separ .code:n={
319   \__oops_option_separ_default:n{ #1 }
320   \exp_last_unbraced:Nf
321   \__oops_make_new:nnn
322   {
323     { \g__oops_option_inner_tl }
324     { \g__oops_option_separ_tl }
325     { \g__oops_option_outer_tl }
326   }
327 },
328 Separ .value_required:n = false,
329 Separ .default:n = { { { \ }and{ \ } } { , { \ } } { , { \ }and{ \ } } },
330 Separ .initial:n = { { { \ }and{ \ } } { , { \ } } { , { \ }and{ \ } } }
331 }

```

\OpsClear #1 : $\langle tl\ var\ name \rangle$

```

332 \NewDocumentCommand{ \OpsClear }
333 { 0{\g__oops_name_tl} }
334 {
335   \__oops_prop_clear_new:n{ #1 }
336 }

```

(End definition for \OpsClear. This function is documented on page 3.)

\OpsOption

```
337 \NewDocumentCommand{ \OpsOption }  
338 { m }  
339 {  
340   \keys_set:nn{ __oops }{ #1 }  
341 }
```

(End definition for \OpsOption. This function is documented on page 5.)

\OpsRestore

```
342 \NewDocumentCommand{\OpsRestore}  
343 {o}  
344 {  
345   \IfValueTF{#1}  
346   {\__oops_log_restore:e{#1}}  
347   {\__oops_log_restore:}  
348 }
```

(End definition for \OpsRestore. This function is documented on page 5.)

\OpsTest

```
349 \NewDocumentCommand\OpsTest{mm}  
350 {  
351   \__oops_test:nn{ #1 }{ #2 }  
352 }
```

(End definition for \OpsTest. This function is documented on page 5.)

10 Misc

```
353 \ExplSyntaxOff
```