

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

Erwann Rogard[†]

Released 2020/04/06

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, `\Ops<object>` begins a series of instructions alternating between ‘text’ and definitions, that themselves optionally expand using predefined or inline rules. For example,

```
\Ops<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 `<object>`, it can be dropped. 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

I	Usage	3
1	Convention	3
2	Loading the package	3
3	\Ops	3
3.1	<code><token list₁></code>	3
3.2	<code>[token list₂]</code>	4
3.3	<code>i{code₁}</code>	4
3.4	<code>{keyval list₁}</code>	4
3.5	<code>*</code>	4
3.6	<code>s{{token list₃}{token list₄}{token list₅}}</code>	4
3.7	<code>o{code₂}</code>	4
3.8	<code>[token list₆]</code>	4
4	\OpsClear	4

*This file describes version v1.3, last revised 2020/04/06.

[†]firstname dot lastname AusTria gmail dot com

5	\OpsOption	4
5.9	Inner	4
5.10	Name	5
5.11	Outer	5
5.12	Separ	5
5.13	Write	5
6	\OpsRead	5
II	Listing	6
	Listing 1.	6
	Listing 2.	6
	Listing 3.	6
	Listing 4.	7
	Listing 5.	7
	Listing 6.	7
	Listing 7.	8
III	Other	9
1	Acknowledgment	9
2	Issues	9
3	Install	9
4	Support	9
5	Unit testing	9
	Change History	11
	Index	11
IV	Implementation	13
1	aux	13
2	log	15
3	make	16

4	msg	18
5	option	18
6	prop	19
7	seq	20
8	Front-end	21
9	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**.
3. Where $\langle token list_1 \rangle$ is an optional argument, its default is **Math**.

<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>\Ops</code>	<code>\Ops<token list₁></code> <code>[\token list₂]</code> <code>i{\code₁}</code> <code>{\keyval list₁}</code> <code>*</code> <code>s{\{\token list₃\}\{\token list₄\}\{\token list₅\}}</code> <code>o{\code₂}</code> <code>[\token list₆]</code>
-------------------	---

Requirement $\langle keyval list_1 \rangle$ is mandatory.

$\langle token list_1 \rangle$

Example **Math**, **ModelA**, **ModelB**

Semantics Registers a new object, if applicable

$\langle token\ list_2 \rangle$

Example Let~

Semantics Expands $\langle token\ list_2 \rangle$

$\langle code_1 \rangle$

Example $\mathbb{\#1}$

Semantics 1. $\langle val_i \rangle \leftarrow \langle code_1 \rangle$ applied to $\langle val_i \rangle$

$\langle keyval\ list_1 \rangle$

Example $\mathit{Elms}=\{\omega_1, \dots, \omega_n\}$, $\mathit{Sample}=\Omega$

Semantics 2. $\backslash\langle key_i \rangle\langle token\ list_1 \rangle < \leftarrow \langle val_i \rangle$ defined in 1.

3. If $\mathit{Write}=\mathit{BooleanTrue}$, writes the definitions made in 2. to file $\mathit{oops}\langle digits \rangle.\mathit{tex}$,
where $\langle digits \rangle=\mathit{pdfdate}$

*

Semantics 4. Expands $\langle code_2 \rangle$ applied to the list created in 1., using $\{\langle token\ list_3 \rangle\}\{\langle token\ list_4 \rangle\}\{\langle token\ list_5 \rangle\}$ as separator.

$\langle token\ list_3 \rangle$

Example $\{\sim\&\sim\}$

$\langle token\ list_4 \rangle$

Example $\{,\sim\}$

$\langle token\ list_5 \rangle$

Example $\{\sim\&\sim\}$

$\langle code_2 \rangle$

Example $\mathit{\$}\backslash\mathit{left}\{\mathit{\#1}\backslash\mathit{right}\backslash\mathit{\$}$

$\langle token\ list_6 \rangle$

Semantics $\backslash\mathit{Ops}\langle token\ list_1 \rangle > [\langle token\ list_6 \rangle]$

$\backslash\mathit{OpsClear}$

$\backslash\mathit{OpsClear}\langle keyval\ list \rangle >$

Semantics Clears any data created by $\backslash\mathit{Ops}\{\langle token\ list_1 \rangle\}$, for all $\langle token\ list_1 \rangle$ in
 $\langle keyval\ list \rangle$

$\backslash\mathit{OpsOption}$

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

Semantics Set default options for $\backslash\mathit{Ops}$

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

Separ

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

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

Write

Syntax $\langle boolean \rangle$

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

Semantics 1. Reads the definitions in $\langle path \rangle$.

2. Writes to `oops.log`: ‘read from $\langle path \rangle$ ’

Other The default for $\langle path \rangle$ is the last write-file (see $\langle keyval\ list_1 \rangle$)

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

$\widehat{\{x\}}\% \{y\} @ \{z\}\$$	$\{x\}\{y\}\{z\}$
$\widehat{(x)}\% (y) @ (z)\$$	$(x)(y)(z)$
$\widehat{\{x\}, \{y\} \& \{z\}}\$$	$\{x\}\{y\}\{z\}$
$[\{x\}\% \{y\} @ \{z\}]$	$\{x\}\{y\}\{z\}$

Listing 2.

```
% \OpsRead \tab \X<Test>\Y<Test>\Z<Test>
% \OpsClear<Test>
%
```

 $\{x\}\{y\}\{z\}$

Listing 3.

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

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`

```
% {}
% \Opclear
%
```

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

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

the sample space.

Listing 4.

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

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

Listing 5.

```
% \OpcRead \tab $\Omega$ $\SigmaField$ $\Measure$
% \OpcClear
%
```

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

Listing 6.

```
% \OpcOption{ Write = \BooleanTrue }
% \newtheorem{theorem}{Theorem}
% \Opc if{\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]} }
% [$n\in\mathbb{N}$,~$OffMenge\subseteq\mathbb{N}^n$ eine offene Menge und
% $f\in Ci(OffMenge,\mathbb{R})$.
% Dann gibt es auf jeder Strecke $Strecke\subseteq OffMenge$ 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}]
%      {}
%      \OpsClear
%      \OpsOption{ Write = \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*

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

Listing 7.

```

%      \OpsRead \tab $\mathbb{N}$ $\mathbb{R}$ $\text{OffMenge}$ $\mathbb{C}$ $\text{Strecke}$
%

```

$$\bar{N} \bar{R} \bar{D} \bar{C}^\top [\bar{x}_0, \bar{x}]$$

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 3 and Listing 4 are from [1]. Listing 6 is from tcolbox[4, 17.3].

2 Issues

1. **Input:** `Inner={\{####1\}}`
Symptom: `\OopsRead` fails
Workaround: `Inner={\char'####1\char'}}`
See: Listing 1
2. **Input:** Inside $\langle keyval list_1 \rangle$, $\{[a, b\]$
Workaround: $\{[a, b\}$ or $\{\char'a, b\char'\}$
See: Listing 6
3. **Input:** Inside $\langle token list_2 \rangle$, $\{[a, b\}$
Workaround: $\{[[]a, b\{\}$
4. **Input:** Inside $\langle token list_2 \rangle$, `\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 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	General: Initial version	10	Revamped: much of the implementation	10
v1.1	General: Added: Save	10	v1.2	General: Added: optional star to \OpsNew as instruction to expand
	Added: Listing 1., 2., 3., 4., 6., and 9.	10		keyval list ₁
	Added:\OpsRestore	10		Deleted: \OpsTest
	Added:\OpsTest	10		Deleted: keyval list ₂ and code ₃
	Deleted: Listing 1-5 from v1.0	10		Deleted: Listing 2-3 from v1.1.
	Fixed: apparent anomaly in v1.0's Listing 4, see Listing 1	10		Replaced: \OpsClear{<token list ₁ >} by \OpsClear[<keyval list>]
	Replaced: \OpsOptions by \OpsOption	10		Replaced: \Restore by \Read
	Replaced: {<keyval list ₂ >} by <keyval list ₂ > given that option type G not recommended[3]	10		Replaced: \Save by \Write
	Replaced: GenericObject by Name	10	v1.3	General: Replaced: \OpsNew by \Ops
	Replaced: Separators by Separ	10		Replaced: {<token list ₁ >} and [<token list ₁ >] by <token list ₁ >

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 commands:	
* (option)	4	\bool_gset_false:N	64
<code ₁ > (option)	4	\bool_gset_true:N	70
<code ₂ > (option)	4	\bool_if:nTF	85, 126, 280
<keyval list ₁ > (option)	4	\bool_set_false:N	60
<token list ₁ > (option)	3	\BooleanFalse	285, 286
<token list ₂ > (option)	3	\BooleanTrue	4
<token list ₃ > (option)	4		
<token list ₄ > (option)	4		
<token list ₅ > (option)	4	C	
<token list ₆ > (option)	4	cs commands:	
Inner (option)	4	\cs_generate_variant:Nn	
Name (option)	5		7, 31, 77, 92, 102, 107, 199
Separ (option)	5	\cs_gset:Npn	6, 24, 185
Write (option)	5	\cs_new:Nn	165, 213, 217
		\cs_new:Npn	221
		\cs_new_protected:Nn	
_	299, 300		4, 13, 17, 22, 27, 40, 44,
A			53, 61, 66, 72, 78, 83, 93, 103, 108,
\AtEndDocument	59		157, 175, 183, 200, 204, 208, 222, 226
B		\cs_new_protected:Npn	9, 32, 112, 191
\begingroup	130	\cs_set:Nn	193
		\cs_set_protected:Nn	228

D		
\DeclareDocumentCommand	114	
\def	130	
\documentclass	6	
E		
\endgroup	130	
\ensuremath	277, 278	
exp commands:		
\exp_args:NNx	95, 114	
\exp_args:No	120, 206	
\exp_last_unbraced:Nf		
	239, 253, 267, 289	
\exp_last_unbraced:NNo	140	
\exp_not:N	163, 173, 181, 189	
\exp_not:n	148	
\expandafter	130	
\ExplSyntaxOff	319	
\ExplSyntaxOn	3	
F		
file commands:		
\file_input:n	74	
I		
\IfBooleanT	134	
\IfValueT	133, 146	
\IfValueTF	315	
iow commands:		
\iow_close:N	59, 63	
\iow_new:N	58	
\iow_now:Nn	87	
\iow_open:Nn	69	
K		
keys commands:		
\keys_define:nn	234	
\keys_set:nn	310	
M		
msg commands:		
\msg_error:nnn	90, 171	
\msg_new:nnn	152, 153, 154, 155, 156	
\msg_warning:nnn	163, 181, 189	
N		
\NeedsTeXFormat	2	
\NewDocumentCommand	302, 307, 312	
\newtcblisting	6	
O		
\Oops	1, 3, 4, 4, 4, 114, 130, 148	
oops internal commands:		
__oops_aux_inner:n	6, 7, 36, 38, 157	
__oops_aux_inner_set:n	4, 120	
__oops_aux_key:N	17, 124	
__oops_aux_key:n	13, 20	
__oops_aux_key:w	9, 15	
g__oops_aux_key_seq		
	11, 19, 125, 136, 210, 211	
g__oops_aux_keyval_seq	121, 122, 124	
__oops_aux_name:n	237	
__oops_aux_outer:n	24, 138	
__oops_aux_outer_set:n	22, 137	
g__oops_aux_prop	26, 29, 37, 46, 123	
__oops_aux_prop:N	44, 122	
__oops_aux_prop:n	40, 50	
__oops_aux_prop:nn	27, 31, 34	
__oops_aux_prop:w	26, 42	
__oops_aux_val	55, 56, 142	
__oops_aux_val:Nn	53, 136	
__oops_log_close:	58, 282	
__oops_log_entry	130	
g__oops_log_iow	58, 59, 63, 69, 87, 90	
__oops_log_open:	66, 281	
g__oops_log_open_bool		
	60, 64, 70, 85, 126	
__oops_log_read:	78, 317	
__oops_log_read:n	72, 80, 316	
g__oops_log_to_tl	68, 69, 80, 82	
__oops_log_write:n	82, 128	
__oops_make_key:N	108, 125	
__oops_make_key:n	103, 110	
__oops_make_key:Nn	93, 105	
__oops_make_new:nnnn		
	112, 240, 254, 268, 290	
__oops_option_inner:n	157, 161, 252	
g__oops_option_inner_tl		
	159, 163, 243, 257, 271, 293	
__oops_option_name:n	165, 238	
g__oops_option_name_tl		
	97, 167, 173, 242, 256, 270, 292, 303	
__oops_option_outer:n	175, 266	
g__oops_option_outer_tl		
	177, 181, 245, 259, 273, 295	
__oops_option_separ:n	183, 288	
g__oops_option_separ_tl		
	185, 189, 244, 258, 272, 294	
__oops_prop_append:NN	191, 202	
__oops_prop_append:Nn	123, 200	
__oops_prop_append:nn	193, 197	
__oops_prop_clear_new:n	204, 211	
__oops_prop_clear_new_map:n	208, 305	
__oops_prop_if_exist:nTF	117, 213	
__oops_prop_item:nn	99, 217	
__oops_prop_name:n		
	56, 202, 206, 215, 219, 221, 224	
__oops_prop_new:n	119, 222	
__oops_seq_from_prop:n	228, 232	

<code>__oops_seq_from_prop:Nn</code> . . .	56, 226	<code>\prop_item:Nn</code>	195, 219, 230
<code>\OpsClear</code>	1, 4, 302	<code>\prop_map_function:Nn</code>	197
<code>\OpsOption</code>	2, 4, 307	<code>\prop_new:N</code>	26, 224
<code>\OpsRead</code>	2, 5, 9, 312	<code>\ProvideDocumentCommand</code>	96
options:			
<code>*</code>	4	Q	
<code><code₁></code>	4	quark commands:	
<code><code₂></code>	4	<code>\q_stop</code>	9, 15, 32, 42
<code><keyval list₁></code>	4	S	
<code><token list₁></code>	3	seq commands:	
<code><token list₂></code>	3	<code>\seq_gclear_new:N</code>	19, 55
<code><token list₃></code>	4	<code>\seq_gput_right:Nn</code>	11, 230
<code><token list₄></code>	4	<code>\seq_if_empty:NTF</code>	47
<code><token list₅></code>	4	<code>\seq_map_function:Nn</code>	
<code><token list₆></code>	4		20, 50, 110, 211, 232
<code>Inner</code>	4	<code>\seq_set_from_clist:Nn</code>	121, 210
<code>Name</code>	5	<code>\seq_use:Nnnn</code>	141
<code>Separ</code>	5	T	
<code>Write</code>	5	tl commands:	
<code>Outer</code>	5	<code>\c_empty_tl</code>	48, 118, 132
<code>Outer (option)</code>	5	<code>\tl_gset:Nn</code>	68, 159, 167, 177
P		<code>\tl_log:n</code>	75, 88
<code>\pdfdate</code>	68	<code>\tl_new:N</code>	82
prop commands:		<code>\tl_trim_spaces:n</code>	11, 35, 36, 38
<code>\prop_clear_new:N</code>	206	U	
<code>\prop_gclear_new:N</code>	46	<code>\usepackage</code>	3, 6
<code>\prop_gput:Nnn</code>	29, 37, 195		
<code>\prop_if_exist:NTF</code>	215		

Part IV

Implementation

```

1 <@@=oops>
2 \NeedsTeXFormat{LaTeX2e}[2019/10/01]
3 \ExplSyntaxOn

```

1 aux

```

\__oops_aux_inner_set:n #1 : <code>
4 \cs_new_protected:Nn \__oops_aux_inner_set:n
5 {
6   \cs_gset:Npn \__oops_aux_inner:n ##1 { #1 }
7   \cs_generate_variant:Nn \__oops_aux_inner:n { e }
8 }

(End definition for \__oops_aux_inner_set:n.)

\__oops_aux_key:w #1 : <key>
#2 : <value>
9 \cs_new_protected:Npn \__oops_aux_key:w #1 = #2 \q_stop

```

```

10 {
11   \seq_gput_right:Nx \g__oops_aux_key_seq { \tl_trim_spaces:n{ #1 } }
12 }

```

(End definition for __oops_aux_key:w.)

```

\__oops_aux_key:n #1 : < key = value >
13 \cs_new_protected:Nn \__oops_aux_key:n
14 {
15   \__oops_aux_key:w #1 \q_stop
16 }

```

(End definition for __oops_aux_key:n.)

```

\__oops_aux_key:N #1 : < seq >
17 \cs_new_protected:Nn \__oops_aux_key:N
18 {
19   \seq_gclear_new:N \g__oops_aux_key_seq
20   \seq_map_function:NN #1 \__oops_aux_key:n
21 }

```

(End definition for __oops_aux_key:N.)

```

\__oops_aux_outer_set:n #1 : < inline code >
22 \cs_new_protected:Nn \__oops_aux_outer_set:n
23 {
24   \cs_gset:Npn \__oops_aux_outer:n ##1 { #1 }
25 }

```

(End definition for __oops_aux_outer_set:n.)

```

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

```

(End definition for __oops_aux_prop:w.)

```

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

```

(End definition for _oops_aux_prop:n.)

```
\_oops_aux_prop:N #1 : <keyval list>
44 \cs_new_protected:Nn \_oops_aux_prop:N
45 {
46   \prop_gclear_new:N \g__oops_aux_prop
47   \seq_if_empty:NTF #1
48   { \c_empty_tl }
49   {
50     \seq_map_function:NN #1 \_oops_aux_prop:n
51   }
52 }
```

(End definition for _oops_aux_prop:N.)

```
\_oops_aux_val:Nn #1 : <seq>
#2 : <tl var name>
53 \cs_new_protected:Nn \_oops_aux_val:Nn
54 {
55   \seq_gclear_new:N \_oops_aux_val
56   \_oops_seq_from_prop:NNn \_oops_aux_val #1 { \_oops_prop_name:n{ #2 } }
57 }
```

(End definition for _oops_aux_val:Nn.)

2 log

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

(End definition for _oops_log_close:.)

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

(End definition for _oops_log_open:.)

```

\__oops_log_read:n #1: <path>
72 \cs_new_protected:Nn \__oops_log_read:n
73 {
74   \file_input:n{#1}
75   \tl_log:n{read~from~#1}
76 }
77 \cs_generate_variant:Nn \__oops_log_read:n { e }

(End definition for \__oops_log_read:n.)

\__oops_log_read:
78 \cs_new_protected:Nn \__oops_log_read:
79 {
80   \__oops_log_read:ef\g__oops_log_to_tl}
81 }

(End definition for \__oops_log_read:.)

\__oops_log_write:n
82 \tl_new:N \g__oops_log_to_tl
83 \cs_new_protected:Nn \__oops_log_write:n
84 {
85   \bool_if:nTF{ \g__oops_log_open_bool }
86   {
87     \iow_now:Nn \g__oops_log_iow { #1 }
88     \tl_log:n{ write~to~#1 }
89   }
90   { \msg_error:nnn{ __oops }{ iow }{ \g__oops_log_iow } }
91 }
92 \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 >
93 \cs_new_protected:Nn \__oops_make_key:Nn
94 {
95   \exp_args:NNx
96   \ProvideDocumentCommand{ #1 }
97   { D<>{\g__oops_option_name_tl} }
98   {
99     \__oops_prop_item:nn{ ##1 }{ #2 }
100   }
101 }
102 \cs_generate_variant:Nn \__oops_make_key:Nn { c }

(End definition for \__oops_make_key:Nn.)

```



```

\__oops_make_key:n #1 : < key >

103 \cs_new_protected:Nn \__oops_make_key:n
104 {
105   \__oops_make_key:cn{#1}{#1}
106 }
107 \cs_generate_variant:Nn \__oops_make_key:n { e }

(End definition for \__oops_make_key:n.)

\__oops_make_key:N #1 : < seq >

108 \cs_new_protected:Nn \__oops_make_key:N
109 {
110   \seq_map_function:NN #1 \__oops_make_key:e
111 }

(End definition for \__oops_make_key:N.)

\__oops_make_new:nnnn #1 : < token list >
#2 : < seq1 >
#3 : < seq2 >
#4 : < prop >

112 \cs_new_protected:Npn \__oops_make_new:nnnn #1 #2 #3 #4
113 {
114   \exp_args:NNx \DeclareDocumentCommand \Ops
115   { D<>{#1} +o E{ i }{ { #2 } } m s E{ s o }{ { #3 }{ #4 } } +o }
116   {
117     \__oops_prop_if_exist:nTF{ ##1 }
118     { \c_empty_tl }
119     { \__oops_prop_new:n{ ##1 } }
120     \exp_args:No \__oops_aux_inner_set:n{ ##3 }
121     \seq_set_from_clist:Nn \g__oops_aux_keyval_seq { ##4 }
122     \__oops_aux_prop:N \g__oops_aux_keyval_seq
123     \__oops_prop_append:Nn \g__oops_aux_prop { ##1 }
124     \__oops_aux_key:N \g__oops_aux_keyval_seq
125     \__oops_make_key:N \g__oops_aux_key_seq
126     \bool_if:nTF{ \g__oops_log_open_bool }
127     {%^A https://tex.stackexchange.com/questions/536597
128       \__oops_log_write:n
129       {
130         \begingroup \def \__oops_log_entry { \Ops< ##1 >i{##3}{ ##4 } } \expandafter \endgroup
131       }
132     }\c_empty_tl}
133     \IfValueT{ ##2 }{ ##2 }
134     \IfBooleanT{ ##5 }
135     {
136       \__oops_aux_val:Nn \g__oops_aux_key_seq { ##1 }
137       \__oops_aux_outer_set:n{ ##7 }
138       \__oops_aux_outer:n
139       {
140         \exp_last_unbraced:NNo
141         \seq_use:Nnnn
142         \__oops_aux_val
143         { ##6 }

```

```

144     }
145   }
146   \IfValueT{ ##8 }
147   {
148     \exp_not:n{ \0ops< ##1 >[ ##8 ] }
149   }
150 }
151 }

```

(End definition for __oops_make_new:nnnn.)

4 msg

```

152 \msg_new:nnn {__oops}{ generic }{ #1 }
153 \msg_new:nnn {__oops}{ iow }{ #1~is~closed~can't~write }
154 \msg_new:nnn {__oops}{ keyonly }{ #1~does~not~take~values;~keyval~is~#2 }
155 \msg_new:nnn {__oops}{ keywrong }{ #1~does~not~recognize~key~#2 }
156 \msg_new:nnn {__oops}{ unset }{ #1~unset }

```

5 option

```

\__oops_aux_inner:n #1: <code>
157 \cs_new_protected:Nn \__oops_option_inner:n
158 {
159   \tl_gset:Nn \g__oops_option_inner_tl { #1 }
160 }
161 \__oops_option_inner:n
162 {
163   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_inner_tl }
164 }

```

(End definition for __oops_aux_inner:n.)

```

\__oops_option_name:n #1: <token list>
165 \cs_new:Nn \__oops_option_name:n
166 {
167   \tl_gset:Nn \g__oops_option_name_tl{ #1 }
168 }
169 \__oops_option_name:n
170 {
171   \msg_error:nnx{ __oops }
172   { generic }
173   { \exp_not:N\g__oops_option_name_tl~undefined }
174 }

```

(End definition for __oops_option_name:n.)

```

\__oops_option_outer:n #1: <inline code>
175 \cs_new_protected:Nn \__oops_option_outer:n
176 {
177   \tl_gset:Nn \g__oops_option_outer_tl { #1 }
178 }
179 \__oops_option_outer:n

```

```

180 {
181   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_outer_tl }
182 }

```

(End definition for __oops_option_outer:n.)

```

\__oops_option_separ:n #1 : { \ token list_1 } { \ token list_2 } { \ token list_3 }
183 \cs_new_protected:Nn \__oops_option_separ:n
184 {
185   \cs_gset:Npn \g__oops_option_separ_tl { #1 }
186 }
187 \__oops_option_separ:n
188 {
189   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_separ_tl }
190 }

```

(End definition for __oops_option_separ:n.)

6 prop

```

\__oops_prop_append:NN #1 : \ prop_1
\__oops_prop_append:cN #2 : \ prop_2
191 \cs_new_protected:Npn \__oops_prop_append:NN #1 #2
192 {
193   \cs_set:Nn \__oops_prop_append:nn
194   {
195     \prop_gput:Nnx #1 { ##1 } { \prop_item:Nn #2 { ##1 } }
196   }
197   \prop_map_function:NN #2 \__oops_prop_append:nn
198 }
199 \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
200 \cs_new_protected:Nn \__oops_prop_append:Nn
201 {
202   \__oops_prop_append:cN{ \__oops_prop_name:n { #2 } } #1
203 }

```

(End definition for __oops_prop_append:Nn.)

```

\__oops_prop_clear_new:n #1 : \ tl var name
204 \cs_new_protected:Nn \__oops_prop_clear_new:n
205 {
206   \exp_args:No \prop_clear_new:c{ \__oops_prop_name:n { #1 } }
207 }

```

(End definition for __oops_prop_clear_new:n.)

```

\__oops_prop_clear_new_map:n #1 : < keyval list >
208 \cs_new_protected:Nn \__oops_prop_clear_new_map:n
209 {
210   \seq_set_from_clist:Nn \g__oops_aux_key_seq { #1 }
211   \seq_map_function:NN \g__oops_aux_key_seq \__oops_prop_clear_new:n
212 }

(End definition for \__oops_prop_clear_new_map:n.)

\__oops_prop_if_exist:nTF #1 : < token list1 >
#2 : < token list2 >
#3 : < token list3 >
213 \cs_new:Nn \__oops_prop_if_exist:nTF
214 {
215   \prop_if_exist:cTF{ \__oops_prop_name:n { #1 } }{ #2 }{ #3 }
216 }

(End definition for \__oops_prop_if_exist:nTF.)

\__oops_prop_item:nn #1 : < tl var name >
#2 : < key >
217 \cs_new:Nn \__oops_prop_item:nn
218 {
219   \prop_item:cn { \__oops_prop_name:n { #1 } } { #2 }
220 }

(End definition for \__oops_prop_item:nn.)

\__oops_prop_name:n #1 : < tl var name >
221 \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 >
222 \cs_new_protected:Nn \__oops_prop_new:n
223 {
224   \prop_new:c{ \__oops_prop_name:n { #1 } }
225 }

(End definition for \__oops_prop_new:n.)

```

7 seq

```

\__oops_seq_from_prop:NNn #1 : < seq1 >
#2 : < seq2 > (keys)
#3 : < prop >
226 \cs_new_protected:Nn \__oops_seq_from_prop:NNn
227 {
228   \cs_set_protected:Nn \__oops_seq_from_prop:n
229   {
230     \seq_gput_right:No #1 { \prop_item:cn{ #3 }{ ##1 } }
231   }
232   \seq_map_function:NN #2 \__oops_seq_from_prop:n
233 }

(End definition for \__oops_seq_from_prop:NNn.)

```

8 Front-end

```

234 \keys_define:nn { __oops }
235 {
236   Name .code:n={
237     % ^A    \__oops_aux_name:n{ #1 }
238     \__oops_option_name:n{ #1 }
239     \exp_last_unbraced:Nf
240     \__oops_make_new:nnnn
241     {
242       { \g__oops_option_name_tl }
243       { \g__oops_option_inner_tl }
244       { \g__oops_option_separ_tl }
245       { \g__oops_option_outer_tl }
246     }
247   },
248   Name .value_required:n = false,
249   Name .default:n = { Math },
250   Name .initial:n = { Math },
251   Inner .code:n={
252     \__oops_option_inner:n{ #1 }
253     \exp_last_unbraced:Nf
254     \__oops_make_new:nnnn
255     {
256       { \g__oops_option_name_tl }
257       { \g__oops_option_inner_tl }
258       { \g__oops_option_separ_tl }
259       { \g__oops_option_outer_tl }
260     }
261   },
262   Inner .value_required:n = false,
263   Inner .default:n = { #####1 },
264   Inner .initial:n = { #####1 },
265   Outer .code:n={
266     \__oops_option_outer:n{ #1 }
267     \exp_last_unbraced:Nf
268     \__oops_make_new:nnnn
269     {
270       { \g__oops_option_name_tl }
271       { \g__oops_option_inner_tl }
272       { \g__oops_option_separ_tl }
273       { \g__oops_option_outer_tl }
274     }
275   },
276   Outer .value_required:n = false,
277   Outer .default:n = { \ensuremath{#####1} },
278   Outer .initial:n = { \ensuremath{#####1} },
279   Write .code:n = {
280     \bool_if:nTF{#1}
281     {\__oops_log_open:}
282     {\__oops_log_close:}
283   },
284   Write .value_required:n = false,
285   Write .default:n = \BooleanFalse,

```

```

286 Write .initial:n = \BooleanFalse,
287 Separ .code:n={
288   \__oops_option_separ:n{ #1 }
289   \exp_last_unbraced:Nf
290   \__oops_make_new:nnnn
291   {
292     { \g__oops_option_name_tl }
293     { \g__oops_option_inner_tl }
294     { \g__oops_option_separ_tl }
295     { \g__oops_option_outer_tl }
296   }
297 },
298 Separ .value_required:n = false,
299 Separ .default:n = { { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } } },
300 Separ .initial:n = { { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } } }
301 }

```

\OpsClear #1 : $\langle \textit{tl var name} \rangle$

```

302 \NewDocumentCommand{ \OpsClear }
303 { D<>{\g__oops_option_name_tl} }
304 {
305   \__oops_prop_clear_new_map:n{ #1 }
306 }

```

(End definition for \OpsClear. This function is documented on page 4.)

\OpsOption

```

307 \NewDocumentCommand{ \OpsOption }
308 { m }
309 {
310   \keys_set:nn{ __oops }{ #1 } % TODO record
311 }

```

(End definition for \OpsOption. This function is documented on page 4.)

\OpsRead

```

312 \NewDocumentCommand{\OpsRead}
313 {o}
314 {
315   \IfValueTF{#1}
316   {\__oops_log_read:e{#1}}
317   {\__oops_log_read:}
318 }

```

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

9 Misc

```

319 \ExplSyntaxOff

```