

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, `\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 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	\OpsClear	3
4	\OpsNew	3
4.1	{⟨token list₁⟩}	3
4.2	[⟨token list₂⟩]	4
4.3	i{⟨code₁⟩}	4
4.4	{⟨keyval list₁⟩}	4
4.5	*	4
4.6	s{⟨token list₃⟩}{⟨token list₄⟩}{⟨token list₅⟩}	4
4.7	o{⟨code₂⟩}	4
4.8	[⟨token list₆⟩]	4

*This file describes version v1.2, 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	Bug	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	21
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>\OpsClear</code>	<code>\OpsClear[\langle keyval\ list \rangle]</code>
------------------------	--

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

<code>\OpsNew</code>	<code>\OpsNew{\langle token\ list_1 \rangle}</code> <code>[\langle token\ list_2 \rangle]</code> <code>i{\langle code_1 \rangle}</code> <code>{\langle keyval\ list_1 \rangle}</code> <code>*</code> <code>s{\langle token\ list_3 \rangle}{\langle token\ list_4 \rangle}{\langle token\ list_5 \rangle}</code> <code>o{\langle code_2 \rangle}</code> <code>[\langle token\ list_6 \rangle]</code>
----------------------	---

Requirement $\langle token\ list_1 \rangle$ and $\langle keyval\ list_1 \rangle$ are 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 $\mathbb{\text{\#1}}$

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

$\langle keyval\ list_1 \rangle$

Example Elms={ ω_1 , \dots , ω_n }, Sample= Ω

Semantics 2. $\langle key_i \rangle[\langle token\ list_1 \rangle] \leftarrow \langle val_i \rangle$ defined in 1.

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

*

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 {~\&~}

$\langle token\ list_4 \rangle$

Example {,~}

$\langle token\ list_5 \rangle$

Example {~\&~}

$\langle code_2 \rangle$

Example $\text{\text{\#1}}$

$\langle token\ list_6 \rangle$

Semantics $\backslash\text{OpsNew}\{\langle token\ list_1 \rangle\}[\langle token\ list_6 \rangle]$

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

Semantics Set default options for $\backslash\text{OpsNew}$

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

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 $\langle boolean \rangle$

`\OopsRead`

`\OopsRead[$\langle path \rangle$]`

Semantics

1. Reads the definitions in $\langle path \rangle$
2. Writes to `oops.log`: ‘read from $\langle path \rangle$ ’

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\$}}
% \OpsNew{Test}{ X =x, Y = y, Z = z }*
% \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsNew{Test}i{(#1)}{ X = x, Y = y, Z = z }*
% \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsNew{Test}{ X = x, Y = y, Z = z }*s{{\ \&\ }{\ \&\ }}
% \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsOption{ Write = \BooleanTrue }
% \OpsNew{Test}{ X = x, Y = y, Z = z }*o{\char`#1\char`}
% \tab \X[Test]\Y[Test]\Z[Test]\\
% \OpsClear[Test]
% \OpsOption{ Write = \BooleanFalse }
%
```

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

Listing 2.

```
% \OpsRead \tab \X[Test]\Y[Test]\Z[Test]
% \OpsClear[Test]
%
```

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

Listing 3.

```
% \OpsNew{Math}[We call~]{Elms={\omega_1, \dots, \omega_n}}*
% [-the elementary events, and ]{Space=\Omega}
% [\begin{equation*}\Space=(\Elms)\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`

```
% {}
% \OpsClear
%
```

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

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

the sample space.

Listing 4.

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

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

Listing 5.

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

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

Listing 6.

```
% \OpsOption{ Write = \BooleanTrue }
% \newtheorem{theorem}{Theorem}
% \OpsNew{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]} }
% [$n\in\mathbb{N}$,~$\text{OffMenge}\subseteq\mathbb{N}^n$ eine offene Menge und
$f\in Ci(\text{OffMenge},\mathbb{R})$.
% Dann gibt es auf jeder Strecke $\text{Strecke}\subseteq\text{OffMenge}$ einen
Punkt $\xi\in\text{Strecke}$,~]
% { 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 Bug

1. **Input:** `Inner={\{####1\}}`
Symptom: `\OpsRead` 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 $\langle token list_2 \rangle$, $\backslash cal F$
Workaround: $\backslash mathcal{F}$

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	Replaced: GenericObject by Name 10
v1.1		Replaced: Separators by Separ .. 10
General: Added: Save	10	Revamped: much of the implementation 10
Added: Listing 1., 2., 3., 4., 6., and 9.	10	v1.2
Added: \OpsRestore	10	General: Added: optional star to \OpsNew as instruction to expand keyval list₁ 10
Added: \OpsTest	10	Deleted: \OpsTest 10
Deleted: Listing 1-5 from v1.0	10	Deleted: keyval list₂ and code₃ . 10
Fixed: apparent anomaly in v1.0's Listing 4, see Listing 1	10	Deleted: Listing 2-3 from v1.1. 10
Replaced: \OpsOptions by \OpsOption .. 10		Replaced: \OpsClear{<token list₁>} by \OpsClear[<keyval list>] 10
Replaced: {<keyval list₂>} by <keyval list₂> given that option type G not recommended[3] 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.

Symbols		
* (option)	4	\bool_gset_true:N 71
<code ₁ > (option)	4	\bool_if:nTF 86, 127, 279
<code ₂ > (option)	4	\bool_set_false:N 61
<keyval list ₁ > (option)	4	\BooleanFalse 284, 285
<token list ₁ > (option)	4	\BooleanTrue 4
<token list ₂ > (option)	4	
<token list ₃ > (option)	4	C
<token list ₄ > (option)	4	cs commands:
<token list ₅ > (option)	4	\cs_generate_variant:Nn 32, 78, 93, 103, 108, 161, 209
<token list ₆ > (option)	4	\cs_gset:Npn 160, 183, 195
Inner (option)	4	\cs_new:Nn 17, 171, 223, 227
Name (option)	5	\cs_new:Npn 231
Separ (option)	5	\cs_new_protected:Nn 8, 12, 28, 41, 45, 54, 62, 67, 73, 79, 84, 94, 104, 109, 158, 163, 181, 185, 193, 210, 214, 218, 232, 236
Write (option)	5	\cs_new_protected:Npn 4, 33, 113, 201
		\cs_set:Nn 203
_ 297, 298		\cs_set_protected:Nn 238
A		
\AtEndDocument 60		
B		
\begingroup 131		D
bool commands:		\DeclareDocumentCommand 115
\bool_gset_false:N 65		\def 131
		\documentclass 6

E	
<code>\endgroup</code>	131
<code>\ensuremath</code>	276, 277
exp commands:	
<code>\exp_args:NNx</code>	96, 115
<code>\exp_args:No</code>	121, 216
<code>\exp_last_unbraced:Nf</code> ..	254, 267, 288
<code>\exp_last_unbraced:NNo</code>	141
<code>\exp_not:N</code>	25, 169, 179, 191, 199
<code>\exp_not:n</code>	149
<code>\expandafter</code>	131
<code>\ExplSyntaxOff</code>	317
<code>\ExplSyntaxOn</code>	3
F	
file commands:	
<code>\file_input:n</code>	75
I	
<code>\IfBooleanT</code>	135
<code>\IfValueT</code>	134, 147
<code>\IfValueTF</code>	313
iow commands:	
<code>\iow_close:N</code>	60, 64
<code>\iow_new:N</code>	59
<code>\iow_now:Nn</code>	88
<code>\iow_open:Nn</code>	70
K	
keys commands:	
<code>\keys_define:nn</code>	244
<code>\keys_set:nn</code>	308
M	
msg commands:	
<code>\msg_error:nnn</code>	23, 91, 177
<code>\msg_new:nnn</code> ..	153, 154, 155, 156, 157
<code>\msg_warning:nnn</code>	169, 191, 199
N	
<code>\NeedsTeXFormat</code>	2
<code>\NewDocumentCommand</code>	300, 305, 310
<code>\newtcblisting</code>	6
O	
oops internal commands:	
<code>__oops_aux_key:N</code>	12, 125
<code>__oops_aux_key:n</code>	8, 15
<code>__oops_aux_key:w</code>	4, 10
<code>\g__oops_aux_key_seq</code>	6, 14, 126, 137, 220, 221
<code>\g__oops_aux_keyval_seq</code> ..	122, 123, 125
<code>__oops_aux_name:n</code>	17, 247
<code>\g__oops_aux_prop</code> ..	27, 30, 38, 47, 124
<code>__oops_aux_prop:N</code>	45, 123
<code>__oops_aux_prop:n</code>	41, 51
<code>__oops_aux_prop:nn</code>	28, 32, 35
<code>__oops_aux_prop:w</code>	27, 43
<code>__oops_aux_val</code>	56, 57, 143
<code>__oops_aux_val:Nn</code>	54, 137
<code>__oops_log_close:</code>	59, 281
<code>__oops_log_entry</code>	131
<code>\g__oops_log_iow</code> ..	59, 60, 64, 70, 88, 91
<code>__oops_log_open:</code>	67, 280
<code>\g__oops_log_open_bool</code>	61, 65, 71, 86, 127
<code>__oops_log_read:</code>	79, 315
<code>__oops_log_read:n</code>	73, 81, 314
<code>\g__oops_log_to_tl</code>	69, 70, 81, 83
<code>__oops_log_write:n</code>	83, 129
<code>__oops_make_key:N</code>	109, 126
<code>__oops_make_key:n</code>	104, 111
<code>__oops_make_key:Nn</code>	94, 106
<code>__oops_make_new:nnn</code> ..	113, 255, 268, 289
<code>\g__oops_name_tl</code>	19, 25, 98, 301
<code>__oops_option_inner:n</code> ..	37, 39, 160, 161
<code>__oops_option_inner_default:n</code> ..	163, 167, 253
<code>__oops_option_inner_set:n</code> ..	121, 158
<code>\g__oops_option_inner_tl</code>	165, 169, 257, 270, 291
<code>__oops_option_name:n</code>	171
<code>\g__oops_option_name_tl</code>	173, 179
<code>__oops_option_outer:n</code>	139, 183
<code>__oops_option_outer_default:n</code> ..	181, 266
<code>__oops_option_outer_set:n</code> ..	138, 181
<code>\g__oops_option_outer_tl</code>	187, 191, 259, 272, 293
<code>__oops_option_separ_default:n</code> ..	193, 287
<code>\g__oops_option_separ_tl</code>	195, 199, 258, 271, 292
<code>__oops_prop_append:NN</code>	201, 212
<code>__oops_prop_append:Nn</code>	124, 210
<code>__oops_prop_append:nn</code>	203, 207
<code>__oops_prop_clear_new:n</code>	214, 221
<code>__oops_prop_clear_new_map:n</code> ..	218, 303
<code>__oops_prop_if_exist:nTF</code> ..	118, 223
<code>__oops_prop_item:nn</code>	100, 227
<code>__oops_prop_name:n</code>	57, 212, 216, 225, 229, 231, 234
<code>__oops_prop_new:n</code>	120, 232
<code>__oops_seq_from_prop:n</code>	238, 242
<code>__oops_seq_from_prop:NNn</code>	57, 236
<code>\OpsClear</code>	1, 3, 11, 300
<code>\OpsNew</code>	1, 3, 3, 4, 11, 115, 131, 149
<code>\OpsOption</code>	2, 4, 11, 305
<code>\OpsOptions</code>	11

<code>\OpsRead</code>	2, 5, 9, 310	Q	
<code>\OpsRestore</code>	11	quark commands:	
<code>\OpsTest</code>	11	<code>\q_stop</code>	4, 10, 33, 43
options:			
<code>*</code>	4	R	
<code><code₁></code>	4	<code>\Read</code>	11
<code><code₂></code>	4	<code>\Restore</code>	11
<code><keyval list₁></code>	4	S	
<code><token list₁></code>	4	<code>\Save</code>	11
<code><token list₂></code>	4	seq commands:	
<code><token list₃></code>	4	<code>\seq_gclear_new:N</code>	14, 56
<code><token list₄></code>	4	<code>\seq_gput_right:Nn</code>	6, 240
<code><token list₅></code>	4	<code>\seq_if_empty:NTF</code>	48
<code><token list₆></code>	4	<code>\seq_map_function:NN</code>	
<code>Inner</code>	4	15, 51, 111, 221, 242
<code>Name</code>	5	<code>\seq_set_from_clist:Nn</code>	122, 220
<code>Separ</code>	5	<code>\seq_use:Nnnn</code>	142
<code>Write</code>	5	T	
<code>Outer</code>	5	tl commands:	
<code>Outer (option)</code>	5	<code>\c_empty_tl</code>	49, 119, 133
P		<code>\tl_gset:Nn</code>	19, 69, 165, 173, 187
<code>\pdfdate</code>	69	<code>\tl_log:n</code>	76, 89
prop commands:		<code>\tl_new:N</code>	83
<code>\prop_clear_new:N</code>	216	<code>\tl_trim_spaces:n</code>	6, 36, 37, 39
<code>\prop_gclear_new:N</code>	47	U	
<code>\prop_gput:Nnn</code>	30, 38, 205	<code>\usepackage</code>	3, 6
<code>\prop_if_exist:NTF</code>	225	W	
<code>\prop_item:Nn</code>	205, 229, 240	<code>\Write</code>	11
<code>\prop_map_function:NN</code>	207		
<code>\prop_new:N</code>	27, 234		
<code>\ProvideDocumentCommand</code>	97		

Part IV

Implementation

```

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

```

1 aux

```

\__oops_aux_key:w #1: < key >
                  #2: < value >

4 \cs_new_protected:Npn \__oops_aux_key:w #1 = #2 \q_stop
5 {
6   \seq_gput_right:Nx \g__oops_aux_key_seq { \tl_trim_spaces:n{ #1 } }
7 }

```

(End definition for `__oops_aux_key:w`.)

```

__oops_aux_key:n #1: < key = value >
8 \cs_new_protected:Nn \__oops_aux_key:n
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:e{ \tl_trim_spaces: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 : <keyval list>
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 : <seq>
#2 : <tl var name>
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_read:n #1: <path>

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

(End definition for \__oops_log_read:n.)

\__oops_log_read:

79 \cs_new_protected:Nn \__oops_log_read:
80 {
81   \__oops_log_read:ef\g__oops_log_to_tl}
82 }

(End definition for \__oops_log_read:.)

\__oops_log_write:n

83 \tl_new:N \g__oops_log_to_tl
84 \cs_new_protected:Nn \__oops_log_write:n
85 {
86   \bool_if:nTF{ \g__oops_log_open_bool }
87   {
88     \iow_now:Nn \g__oops_log_iow { #1 }
89     \tl_log:n{ write~to~#1 }
90   }
91   { \msg_error:nnn{ __oops }{ iow }{ \g__oops_log_iow } }
92 }
93 \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 >

94 \cs_new_protected:Nn \__oops_make_key:Nn
95 {
96   \exp_args:NNx
97   \ProvideDocumentCommand{ #1 }
98   { 0{ \g__oops_name_tl } }
99   {
100     \__oops_prop_item:nn{ ##1 }{ #2 }
101   }
102 }
103 \cs_generate_variant:Nn \__oops_make_key:Nn { c }

(End definition for \__oops_make_key:Nn.)

```



```

\__oops_make_key:n #1: < key >

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

(End definition for \__oops_make_key:n.)

\__oops_make_key:N #1: < seq >

109 \cs_new_protected:Nn \__oops_make_key:N
110 {
111   \seq_map_function:NN #1 \__oops_make_key:e
112 }

(End definition for \__oops_make_key:N.)

\__oops_make_new:nnn #1: < seq1 >
#2: < seq2 >
#3: < prop >

113 \cs_new_protected:Npn \__oops_make_new:nnn #1 #2 #3
114 {
115   \exp_args:NNx \DeclareDocumentCommand \OpsNew
116   { m +o E{ i }{ { #1 } } m s E{ s o }{ { #2 }{ #3 } } +o }
117   {
118     \__oops_prop_if_exist:nTF{ ##1 }
119     { \c_empty_tl }
120     { \__oops_prop_new:n{ ##1 } }
121     \exp_args:No \__oops_option_inner_set:n{ ##3 }
122     \seq_set_from_clist:Nn \g__oops_aux_keyval_seq { ##4 }
123     \__oops_aux_prop:N \g__oops_aux_keyval_seq
124     \__oops_prop_append:Nn \g__oops_aux_prop { ##1 }
125     \__oops_aux_key:N \g__oops_aux_keyval_seq
126     \__oops_make_key:N \g__oops_aux_key_seq
127     \bool_if:nTF{ \g__oops_log_open_bool }
128     {%^A https://tex.stackexchange.com/questions/536597
129       \__oops_log_write:n
130       {
131         \begingroup \def \__oops_log_entry { \OpsNew{ ##1 }i{##3}{ ##4 } } \expandafter \endgroup
132       }
133     }\c_empty_tl}
134     \IfValueT{ ##2 }{ ##2 }
135     \IfBooleanT{ ##5 }
136     {
137       \__oops_aux_val:Nn \g__oops_aux_key_seq { ##1 }
138       \__oops_option_outer_set:n{ ##7 }
139       \__oops_option_outer:n
140       {
141         \exp_last_unbraced:NNo
142         \seq_use:Nnnn
143         \__oops_aux_val
144         { ##6 }
145       }
146     }

```

```

147     \IfValueT{ ##8 }
148     {
149         \exp_not:n{ \OpsNew{ ##1 }[ ##8 ] }
150     }
151 }
152 }

```

(End definition for `__oops_make_new:nnn`.)

4 msg

```

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

```

5 option

`__oops_option_inner_set:n` #1: *⟨inlinecode⟩*

```

158 \cs_new_protected:Nn \__oops_option_inner_set:n
159 {
160     \cs_gset:Npn \__oops_option_inner:n ##1 { #1 }
161     \cs_generate_variant:Nn \__oops_option_inner:n { e }
162 }
163 \cs_new_protected:Nn \__oops_option_inner_default:n
164 {
165     \tl_gset:Nn \g__oops_option_inner_tl { #1 }
166 }
167 \__oops_option_inner_default:n
168 {
169     \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_inner_tl }
170 }

```

(End definition for `__oops_option_inner_set:n`.)

`__oops_option_name:n` #1: *⟨token list⟩*

```

171 \cs_new:Nn \__oops_option_name:n
172 {
173     \tl_gset:Nn \g__oops_option_name_tl{ #1 }
174 }
175 \__oops_option_name:n
176 {
177     \msg_error:nnx{ __oops }
178     { generic }
179     { \exp_not:N \g__oops_option_name_tl~undefined }
180 }

```

(End definition for `__oops_option_name:n`.)

`__oops_option_outer_set:n` #1: *⟨ inline code ⟩*

```

\__oops_option_outer_default:n
181 \cs_new_protected:Nn \__oops_option_outer_set:n
182 {

```

```

183 \cs_gset:Npn \__oops_option_outer:n ##1 { #1 }
184 }
185 \cs_new_protected:Nn \__oops_option_outer_default:n
186 {
187 \tl_gset:Nn \g__oops_option_outer_tl { #1 }
188 }
189 \__oops_option_outer_default:n
190 {
191 \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_outer_tl }
192 }

```

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

```

\__oops_option_separ_default:n #1 : { \< token list1 \> } { \< token list2 \> } { \< token list3 \> }
193 \cs_new_protected:Nn \__oops_option_separ_default:n
194 {
195 \cs_gset:Npn \g__oops_option_separ_tl { #1 }
196 }
197 \__oops_option_separ_default:n
198 {
199 \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_separ_tl }
200 }

```

(End definition for __oops_option_separ_default:n.)

6 prop

```

\__oops_prop_append:NN #1 : \< prop1 \>
\__oops_prop_append:cN #2 : \< prop2 \>
201 \cs_new_protected:Npn \__oops_prop_append:NN #1 #2
202 {
203 \cs_set:Nn \__oops_prop_append:nn
204 {
205 \prop_gput:Nnx #1 { ##1 } { \prop_item:Nn #2 { ##1 } }
206 }
207 \prop_map_function:NN #2 \__oops_prop_append:nn
208 }
209 \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 \>
210 \cs_new_protected:Nn \__oops_prop_append:Nn
211 {
212 \__oops_prop_append:cN{ \__oops_prop_name:n { #2 } } #1
213 }

```

(End definition for __oops_prop_append:Nn.)

```

__oops_prop_clear_new:n #1 : < tl var name >

214 \cs_new_protected:Nn __oops_prop_clear_new:n
215 {
216   \exp_args:No \prop_clear_new:c{ __oops_prop_name:n { #1 } }
217 }

(End definition for __oops_prop_clear_new:n.)

__oops_prop_clear_new_map:n #1 : < keyval list >

218 \cs_new_protected:Nn __oops_prop_clear_new_map:n
219 {
220   \seq_set_from_clist:Nn \g__oops_aux_key_seq { #1 }
221   \seq_map_function:NN \g__oops_aux_key_seq __oops_prop_clear_new:n
222 }

(End definition for __oops_prop_clear_new_map:n.)

__oops_prop_if_exist:nTF #1 : < token list1 >
#2 : < token list2 >
#3 : < token list3 >

223 \cs_new:Nn __oops_prop_if_exist:nTF
224 {
225   \prop_if_exist:CTF{ __oops_prop_name:n { #1 } }{ #2 }{ #3 }
226 }

(End definition for __oops_prop_if_exist:nTF.)

__oops_prop_item:nn #1 : < tl var name >
#2 : < key >

227 \cs_new:Nn __oops_prop_item:nn
228 {
229   \prop_item:cn { __oops_prop_name:n { #1 } } { #2 }
230 }

(End definition for __oops_prop_item:nn.)

__oops_prop_name:n #1 : < tl var name >

231 \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 >

232 \cs_new_protected:Nn __oops_prop_new:n
233 {
234   \prop_new:c{ __oops_prop_name:n { #1 } }
235 }

(End definition for __oops_prop_new:n.)

```

7 seq

```

\__oops_seq_from_prop:NNn #1: < seq1 >
#2: < seq2 > (keys)
#3: < prop >

236 \cs_new_protected:Nn \__oops_seq_from_prop:NNn
237 {
238   \cs_set_protected:Nn \__oops_seq_from_prop:n
239   {
240     \seq_gput_right:No #1 { \prop_item:cn{ #3 }{ ##1 } }
241   }
242   \seq_map_function:NN #2 \__oops_seq_from_prop:n
243 }

```

(End definition for __oops_seq_from_prop:NNn.)

8 Front-end

```

244 \keys_define:nn { __oops }
245 {
246   Name .code:n={
247     \__oops_aux_name:n{ #1 }
248   },
249   Name .value_required:n = false,
250   Name .default:n = { Math },
251   Name .initial:n = { Math },
252   Inner .code:n={
253     \__oops_option_inner_default:n{ #1 }
254     \exp_last_unbraced:Nf
255     \__oops_make_new:nnn
256     {
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_default:n{ #1 }
267     \exp_last_unbraced:Nf
268     \__oops_make_new:nnn
269     {
270       { \g__oops_option_inner_tl }
271       { \g__oops_option_separ_tl }
272       { \g__oops_option_outer_tl }
273     }
274   },
275   Outer .value_required:n = false,
276   Outer .default:n = { \ensuremath{####1} },
277   Outer .initial:n = { \ensuremath{####1} },
278   Write .code:n = {

```

```

279     \bool_if:nTF{#1}
280     {\__oops_log_open:}
281     {\__oops_log_close:}
282   },
283   Write .value_required:n = false,
284   Write .default:n = \BooleanFalse,
285   Write .initial:n = \BooleanFalse,
286   Separ .code:n={
287     \__oops_option_separ_default:n{ #1 }
288     \exp_last_unbraced:Nf
289     \__oops_make_new:nnn
290     {
291       { \g__oops_option_inner_tl }
292       { \g__oops_option_separ_tl }
293       { \g__oops_option_outer_tl }
294     }
295   },
296   Separ .value_required:n = false,
297   Separ .default:n = { { {\ }and{\ } } { {\ } } { {\ }and{\ } } },
298   Separ .initial:n = { { {\ }and{\ } } { {\ } } { {\ }and{\ } } }
299 }

```

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

```

300 \NewDocumentCommand{ \OpsClear }
301 { 0{\g__oops_name_tl} }
302 {
303   \__oops_prop_clear_new_map:n{ #1 } % TODO record
304 }

```

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

\OpsOption

```

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

```

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

\OpsRead

```

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

```

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

9 Misc

```

317 \ExplSyntaxOff

```