

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

Erwann Rogard[†]

Released 2020/04/10

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`[4]. 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
0	Convention	3
1	Loading the package	3
2	\Ops	3
2.1	<code><tl₁></code>	4
2.2	<code>[tl₂]</code>	4
2.3	<code>i{code₁}</code>	4
2.4	<code>{kv₁}</code>	4
2.5	<code>+</code>	4
2.6	<code>*</code>	4
2.7	<code>s{{tl₃}{tl₃}{tl₄}{tl₃}{tl₄}{tl₅}}</code>	4
2.8	<code>o{code₂}</code>	5
2.9	<code>[tl₆]</code>	5
3	\OpsClear	5
4	\OpsDebug	5

*This file describes version v1.5, last revised 2020/04/10.

[†]firstname dot lastname AusTria gmail dot com

5	\OpsHook	5
6	\OpsOption	5
6.1	Expans	5
6.2	File	5
6.3	Inner	5
6.4	Name	5
6.5	Outer	5
6.6	Separ	6
6.7	Write	6
7	\OpsRead	6
8	Do's and dont's	6
II Listing		7
	Listing 1.	7
	Listing 2.	7
	Listing 3.	8
	Listing 4.	8
	Listing 6.	8
	Listing 7.	9
	Listing 8.	9
III Other		10
1	Acknowledgment	10
2	Also see	10
3	Install	10
4	Issue	10
5	Support	10
6	Testing	10
Change History		12
Index		12

IV	Implementation	15
1	aux	15
2	log	17
3	make_key	19
4	make_oops	19
5	msg	21
6	option	21
7	prop	22
8	seq	24
9	Front-end	24
10	Misc	26

Part I

Usage

Convention

1. Loosely, those of [3] and [4], for example as to the meaning of $\langle token\ list \rangle$.
2. If unspecified, the environment in which a function must be declared is `document`.
3. Where $\langle tl_1 \rangle$ 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 5](#).

\Ops $\langle tl_1 \rangle$
 $[\langle tl_2 \rangle]$
 $i\{\langle code_1 \rangle\}$
 $\{\langle kvl_1 \rangle\}$
 $+$
 $*$
 $s\{\{\langle tl_3 \rangle\}|\{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\}|\{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\}\{\langle tl_5 \rangle\}\}$
 $o\{\langle code_2 \rangle\}$
 $[\langle tl_6 \rangle]$

Requirement $\langle kvl_1 \rangle$ is specified (all others optional).

$\langle tl_1 \rangle$

Example Math, ModelA, ModelB

Semantics Registers a new object, if applicable

$\langle tl_2 \rangle$

Example Let~

Semantics Expands $\langle tl_2 \rangle$

$\langle code_1 \rangle$

Example $\mathbb{\#1}$

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

$\langle kvl_1 \rangle$

Example Elms={ ω_1 , ..., ω_n }, Sample= Ω

Semantics 2. $\langle key_i \rangle \langle tl_1 \rangle \leftarrow \langle val_i \rangle$ defined in step 1, using subsection 6.2 for expansion.

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

$+$

Other Needed to make \Ops'side effect within a *local group* persist thereafter

Semantics Appends step 2 and step 3 to \OpsHook

$*$

Semantics 5. Expands $\langle code_2 \rangle$ applied to the list created in step 1, using the separator specified by subsection 2.7.1, subsection 2.7.2, subsection 2.7.3.

$\langle tl_3 \rangle$

Example $\{\sim\text{in}\sim\}$

$\langle tl_4 \rangle$

Example `{,~}`

$\langle tl_5 \rangle$

Example `{~\&~}`

$\langle code_2 \rangle$

Example `$$\left\{\#1\right\}\right\}$`

$\langle tl_6 \rangle$

Semantics `\Ops<\langle tl_1 \rangle>[\langle tl_6 \rangle]`

\OpsClear `\OpsClear<\langle keyval list \rangle>`

Semantics Clears any data created by `\Ops{\langle tl_1 \rangle}`, for all $\langle tl_1 \rangle$ in $\langle keyval list \rangle$

\OpsDebug **Semantics** See [Part IV](#)

\OpsHook `\OpsHook`

Example `\AfterEndEnvironment{theorem}{\OpsHook}`

\OpsOption `\OpsOption{\langle kv10 \rangle}`

Semantics Set default options for [section 2](#)

Expans

Default `xo`

Syntax Either of `eo`, `ee`, `ex`, `xe`, `xo`, `xe`, `xx`

File

Syntax *token*

Inner

Semantics Default for [subsection 2.3](#)

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

Name

Semantics Default for [subsection 2.1](#)

Outer

Semantics Default for [subsection 2.8](#)

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

Separ

Semantics Default for [subsection 2.7](#)

Syntax That of ‘separators’ in [\[3, Section 8 of l3seq\]](#)

Write

Syntax *Boolean*

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

Other The default for $\langle path \rangle$ is the last write-file (see $\langle kl_1 \rangle$)

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

Do’s and don’t’s

1.

Don’t: `\Oop{ A = a, B = b }[Hello, world!]`.

Do: `\Oop{ A = a, B = b }[Hello, world!]{}`, or
`\Oop{ A = a, B = b } Hello, world!`

2.

Don’t: `$\langle key_i \rangle <x$`.

Do: `$\langle key_i \rangle \{<\}x$`

3.

Don’t: `\Oop[[a, b)]`.

Do: `\Oop{\{[a, b{)}`

4.

Don’t: `\Oop{ F = \cal F }`.

Do: `\Oop{ F = \cal{F} }` or `\Oop{ F = \mathcal{F} }`

5. Also see [Part III, section 4](#)

Part II

Listing

Tip: To replicate the listings in a L^AT_EX document, use the same setting as that of the documentation portion of `oops.dtx` (`\documentclass`, `\usepackage`, `\newtcblisting`, ...), and remove any `^^A`.

Listing 1.

```
% \OpsOption{
% ^^A% spaces betw. inner and outer brackets matter!->
% Separ={\ \char`@\ }{\ \% \ }{\ \char`@\ }}
% \Ops<Test>{ X = x, Y = y }*[\]
% { X = x, Y = y, Z = z }*[\]
% { X = x, Y = y }*s{\ \% \ }[\]
% { X = x, Y = y }*s{\ \% \ }{, \ }[\]
% { X = x, Y = y, Z = z }*s{\ \% \ }{, \ }[\]
% { X = x, Y = y, Z = z }*s{\ \% \ }{, \ }{, \ }[\]
% { X = x, Y = y, Z = z }*s{\ \% \ }{, \ }{\ \% \ }{\ \% \ }
%
```

```
x @ y
x \% y @ z
x & y
x & y
x & y & z
x, y & z
x, y & z
```

Listing 2.

```
% \OpsOption{ Separ = {\}\{.\}\{.}\}, Outer = {####1} }
% \OpsOption{ Write = \BooleanTrue }
% \Ops<Test>
% { KeyA = {.\}, KeyB = {!}, KeyC = {\%} }[]
% { KeyD = {d}, KeyE = {\%} }[]i{\#1\}
% { KeyF = {H}, KeyG = {e}, KeyH = {l} }*[]
% { KeyI = {\%}, KeyJ = {\%}, KeyK = {\%} }[.\{l\}.\{o\}]
% { KeyL = {l}, KeyM = {\char`[]}, KeyN = {\char`[] } }[]
% { KeyO = {o}, KeyP = {\%}, KeyQ = {\%} }[{, \ }
% { KeyR = {w}, KeyS = {o}, KeyT = {r} }*s{\}\{\}\}\o{\char`[]\#1}[]
% { KeyU = {\%}, KeyV = {\%}, KeyW = {\%} }[]
% { KeyX = {\%}, KeyY = {\%}, KeyZ = {\KeyB<Test>} } \nobreak
% \KeyL<Test>\KeyD<Test>\KeyZ<Test>\KeyN<Test>\
% \OpsOption{ Write = \BooleanFalse }
%
```

```
{H}.\{e}.\{l}.\{l}.\{o}, [world!]
```

Listing 3.

```
% \OpsRead
% \KeyF<Test>\KeyA<Test>\nobreak
% \KeyG<Test>\KeyA<Test>\nobreak
% \KeyH<Test>\KeyA<Test>\nobreak
% \KeyH<Test>\KeyA<Test>\nobreak
% \KeyH<Test>\KeyA<Test>\nobreak
% {\{ }\nobreak\KeyO<Test>\{ \}, {\ } \nobreak
% \KeyM<Test>\KeyR<Test>\nobreak
% \KeyO<Test>\nobreak
% \KeyT<Test>\nobreak
% \KeyL<Test>\nobreak
% \KeyD<Test>\nobreak
% \KeyZ<Test>\nobreak
% \KeyN<Test>\nobreak
%
```

$\{H\}.\{e\}.\{l\}.\{l\}.\{o\}, [\text{world!}]$

Listing 4.

```
% \Ops[We call~]{Elems={\omega_{1}}, {\dots}, {\omega_{n}}}}*
% [~the elementary events, and ]{Space=\Omega}
% [\begin{equation*}\Space=(\Elems)\end{equation*}~the sample space.]
% {}
%
```

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

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

the sample space.

Listing 5.

```
% \OpsOption{ Write = \BooleanTrue }
% \Ops[Let~]
% {Space=\Omega, Field=\mathcal{F}, Meas=\mathcal{P}}
% *s{\{, \}}o{\$ \{ \#1 \} \$}
% [~denote the probability space, where $\Field \subset 2^{\Space}$.]
% {}
% \OpsOption{ Write = \BooleanFalse }
```

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

Listing 6.

```
%          \OpsRead \tab $\Omega$ $\Field$ $\Meas$
%
```

$$\Omega \not\subset \mathcal{P}$$

Listing 7.

```
%          \OpsOption{ Write = \BooleanTrue }
%          \newtheorem{theorem}{Theorem}
%          \AfterEndEnvironment{theorem}{\OpsHook}
%          \Ops 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] } }+
%            [$n\in\mathbb{N},~\text{\textit{OffMenge}}\subseteq\mathbb{N}^n$ eine offene Menge und
%            $f\in C^1(\text{\textit{OffMenge}},\mathbb{R})$.
%            Dann gibt es auf jeder Strecke $\text{\textit{Strecke}}\subseteq\text{\textit{OffMenge}}$ einen
%            Punkt $\xi\in\text{\textit{Strecke}}$,~]
%            { Steig = { \frac{ f(x)-f(x_0) }{ x-x_0 } }, Punkt = { \xi } }+
%            [so dass gilt
%            \begin{equation*}
%              \text{\textit{Steig}} = \text{\textit{Grad}} f(\text{\textit{Punkt}})^{\top}
%            \end{equation*}
%          \end{theorem}]
%          {}
%          \OpsOption{ Write = \BooleanFalse }
%
```

Theorem 1 (Mittelwertsatz für n Variable) *Es sei $n \in \mathbb{N}$, $D \subseteq \mathbb{N}^n$ 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 8.

```
%          \OpsRead \tab $\mathbb{N}$ $\mathbb{R}$ $\text{\textit{OffMenge}}$ $\text{\textit{Ci}}$ $\text{\textit{Strecke}}$
%
```

$$\mathbb{N} \not\subset \mathbb{R} \not\subset D \subset C^1[x_0, x]$$

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 4 and Listing 5 are from [1]. Listing 7 is from tcolbox[5, 17.3].

2 Also see

1. The package cool[2]

3 Install

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

4 Issue

1. **Don't:** `Inner={\####1\}`
Symptom: `\OpsRead` fails
Do: `Inner={\char'####1\char'}`

5 Support

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

6 Testing

It's not possible to check the expansion of a certain class of macros against predefined values[6]. 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] Nick Setzer *The cool package*, 2005 <https://www.ctan.org/pkg/cool>
- [3] 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>

- [4] The L^AT_EX3 Project Team *The xparse package* <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf>
- [5] Thomas F. Sturm *The tcolorbox package* <http://www.texdoc.net/texmf-dist/doc/latex/tcolorbox/tcolorbox.pdf>
- [6] <https://tex.stackexchange.com/a/534100/112708>

Change History

v1.0	General: Initial version	11	Replaced: <code>\OpsClear{<tl>}</code> by <code>\OpsClear[<keyval list>]</code>	11
v1.1	General: Added: <code>\Save</code>	11	Replaced: <code>\Restore</code> by <code>\Read</code>	11
	Added: Listing 1., 2., 3., 4., 6., and 9.	11	Replaced: <code>\Save</code> by <code>\Write</code>	11
	Added: <code>\OpsRestore</code>	11	v1.3	General: Replaced: <code>\OpsNew</code> by <code>\Ops</code>
	Added: <code>\OpsTest</code>	11		Replaced: <code>{<tl>}</code> and <code>[<tl>]</code> by <code><tl></code>
	Deleted: Listing 1-5 from v1.0	11	v1.4	General: Added: section 8
	Fixed: apparent anomaly in v1.0's Listing 4, see Listing 1	11		Added: <code>\OpsDebug</code>
	Replaced: <code>\OpsOptions</code> by <code>\OpsOption</code>	11		Added: <code>\OpsHook</code>
	Replaced: <code>{<kvl>}</code> by <code><kvl_2></code> given that option type G not recommended[4]	11		Added: <code>\Expans</code> (for debugging' sake, but...)
	Replaced: <code>\GenericObject</code> by <code>\Name</code>	11		Added: Listing 1., 2., and 3.
	Replaced: <code>\Separators</code> by <code>\Separ</code>	11		Deleted: Listing 1., and 2.
	Revamped: much of the implementation	11		Replaced: <code>s{<tl>}{<tl>}{<tl>}</code> by <code>s{<tl>}{<tl>}{<tl>}{<tl>}{<tl>}{<tl>}</code>
v1.2	General: Deleted: <code>\OpsTest</code>	11	v1.5	General: Added: <code>\File</code>
	Deleted: <code><kvl></code> and <code><code></code>	11		Deleted: dependence on <code>\datetime</code>
	Deleted: Listing 2-3 from v1.1.	11		

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	Write (option)	6
* (option)		4
+ (option)		4
<code>\<key></code>		4, 6
<code><code></code> (option)		4
<code><code></code> (option)		5
<code><kvl></code> (option)		4
<code><tl></code> (option)		4
<code><tl></code> (option)		4
<code><tl></code> (option)		4
<code><tl></code> (option)		4
<code><tl></code> (option)		5
<code><tl></code> (option)		5
<code>\Expans</code> (option)		5
<code>\File</code> (option)		5
<code>\Inner</code> (option)		5
<code>\Name</code> (option)		5
<code>\Separ</code> (option)		6
	A	
	<code>\AtEndDocument</code>	87
	B	
	<code>\begin</code>	289
	<code>\begingroup</code>	172
	bool commands:	
	<code>\bool_gset_false:N</code>	92
	<code>\bool_gset_true:N</code>	99
	<code>\bool_if:nTF</code>	114, 168, 192, 352
	<code>\bool_set_false:N</code>	88
	<code>\BooleanFalse</code>	357, 358
	<code>\BooleanTrue</code>	4

C	
cs commands:	
\cs_generate_variant:Nn
.....	7, 31, 76, 106, 121, 131, 136, 254
\cs_gset:Npn 6, 24, 240
\cs_new:Nn	.. 54, 77, 220, 268, 272, 293
\cs_new:Npn 276
\cs_new_protected:Nn	.. 4, 13, 17,
22, 27, 41, 45, 81, 89, 95, 101, 107,	
112, 122, 132, 137, 141, 153, 165,	
212, 230, 238, 255, 259, 263, 277, 281	
\cs_new_protected:Npn	9, 33, 179, 246
\cs_set:Nn 248
\cs_set_protected:Nn 283
D	
\DeclareDocumentCommand 181
\def 173, 178
\documentclass 7
E	
\end 292
\endgroup 174
\ensuremath 349, 350
exp commands:	
\exp_args:NNf 147
\exp_args:NNx 124, 181
\exp_args:No 158, 261
\exp_args:Nx 35
\exp_last_unbraced:Nf
.....	311, 325, 339, 361
\exp_last_unbraced:NNf 295
\exp_not:N 72, 218, 228, 236, 244
\exp_not:n 202, 306
\expandafter 173
\ExplSyntaxOff 405
\ExplSyntaxOn 3
F	
file commands:	
\file_input:n 103
G	
\gappto 194, 385
I	
\IfBooleanT 188
\IfValueT 186, 200
\IfValueTF 401
int commands:	
\int_case:nnTF 56
iow commands:	
\iow_close:N 87, 91
\iow_new:N 86
\iow_now:Nn 116
\iow_open:Nn 98
\item 290, 291
K	
\KeyA 384, 390
keys commands:	
\l_keys_choice_tl 303
\keys_define:nn 299
\keys_set:nn 396
M	
\meta 290, 291
msg commands:	
\msg_error:nnn 119, 226
\msg_error:nnnn 70
\msg_new:nnn	206, 207, 208, 209, 210, 211
\msg_warning:nnn 218, 236, 244
N	
\NeedsTeXFormat 2
\NewDocumentCommand	... 374, 379, 393, 398
\newtcblisting 7
O	
\Oops 1, 4, 4, 5, 5, 6, 12, 173, 181, 202
oops internal commands:	
__oops_aux_inner:n	.. 6, 7, 38, 39, 212
__oops_aux_inner_set:n 4, 158
__oops_aux_key:N 17, 162
__oops_aux_key:n 13, 20
__oops_aux_key:w 9, 15
g__oops_aux_key_seq
.....	11, 19, 143, 163, 265, 266
g__oops_aux_keyval_seq	159, 160, 162
__oops_aux_outer:n 24, 145
__oops_aux_outer_set:n 22, 144
g__oops_aux_prop 26, 29, 47, 161
__oops_aux_prop:N 45, 160
__oops_aux_prop:n 41, 51
__oops_aux_prop:nn 27, 31
__oops_aux_prop:w 26, 43
__oops_aux_separ:n 77, 297
__oops_aux_separ:nn 54, 79
__oops_aux_val:Nn 81, 143
g__oops_aux_val_seq 83, 84, 149
__oops_erw_seq_use:Nn 289
__oops_log_close: 86, 354
__oops_log_entry 173, 174
g__oops_log_file_tl 94, 97, 306
g__oops_log_iow	86, 87, 91, 98, 116, 119
__oops_log_open: 94, 353
g__oops_log_open_bool
.....	88, 92, 99, 114, 168
__oops_log_read: 107, 403
__oops_log_read:n 101, 109, 402

\g_ooops_log_to_tl ..	97, 98, 109, 111
__oops_log_write:n	111, 170
__oops_make_key:N	137, 163
__oops_make_key:n	132, 139
__oops_make_key:Nn	122, 134, 384, 390
__oops_make_oops:nnnn	
.....	178, 312, 326, 340, 362
__oops_make_oops_exp:nnn ..	141, 190
__oops_make_oops_key:nnn ..	153, 167
__oops_make_oops_sideeffect:nnn	
.....	165, 187, 196
\g_ooops_option_expans_tl	32, 36, 303
__oops_option_inner:n .	212, 216, 324
\g_ooops_option_inner_tl	
.....	214, 218, 315, 329, 343, 365
__oops_option_name:n	220, 310
\g_ooops_option_name_tl	
.....	126, 222, 228, 314, 328, 342, 364, 375
__oops_option_outer:n	230, 338
\g_ooops_option_outer_tl	
.....	232, 236, 317, 331, 345, 367
__oops_option_separ:n	238, 360
\g_ooops_option_separ_tl	
.....	240, 244, 316, 330, 344, 366
__oops_prop_append:NN	246, 257
__oops_prop_append:Nn	161, 255
__oops_prop_append:nn	248, 252
__oops_prop_clear_new:n ...	259, 266
__oops_prop_clear_new_map:n	263, 377
__oops_prop_if_exist:nTF	
.....	155, 268, 381, 387
__oops_prop_item:nn	128, 272
__oops_prop_name:n	
.....	84, 257, 261, 270, 274, 276, 279
__oops_prop_new:n .	157, 277, 383, 389
__oops_seq_from_prop:n	283, 287
__oops_seq_from_prop:NNn ...	84, 281
__oops_seq_use:Nn	148, 293
\OpsClear	1, 5, 12, 374
\OpsDebug	1, 5, 12, 379
\OpsHook	2, 4, 5, 12, 178, 194, 385
\OpsNew	12
\OpsOption	2, 5, 12, 393
\OpsOptions	12
\OpsRead	2, 6, 10, 398
\OpsRestore	12
\OpsTest	12
options:	
*	4
+	4
<code ₁ >	4
<code ₂ >	5
<kv _l ₁ >	4
<tl ₁ >	4
<tl ₂ >	4
<tl ₃ >	4
<tl ₄ >	4
<tl ₅ >	5
<tl ₆ >	5
Expans	5
File	5
Inner	5
Name	5
Separ	6
Write	6
Outer	5
Outer (option)	5
P	
\pdfcreationdate	307, 308
prg commands:	
\prg_replicate:nn	59
prop commands:	
\prop_clear_new:N	261
\prop_gclear_new:N	47
\prop_gput:Nnn	29, 250
\prop_if_exist:NNTF	270
\prop_item:Nn	250, 274, 285
\prop_map_function:NN	252
\prop_new:N	26, 279
\ProvideDocumentCommand	125
Q	
quark commands:	
\q_stop	9, 15, 33, 43
R	
\Read	12
\Restore	12
S	
\Save	12
seq commands:	
\seq_gclear_new:N	19, 83
\seq_gput_right:Nn	11, 285
\seq_if_empty:NNTF	48
\seq_map_function:NN	
.....	20, 51, 139, 266, 287
\seq_set_from_clist:Nn	159, 265
\seq_use:Nnnn	296
T	
tl commands:	
\c_empty_tl	
..	49, 68, 156, 176, 178, 199, 382, 388
\tl_count:n	79
\tl_gset:Nn	97, 214, 222, 232, 306
\tl_gset_eq:NN	303
\tl_log:n	104, 117

<code>\tl_new:N</code>	32, 94, 111	<code>\use_i:nn</code>	62, 64
<code>\tl_trim_spaces:n</code>	11, 37, 38, 39	<code>\use_ii:nn</code>	63
		<code>\usepackage</code>	3, 7
U			
use commands:		W	
<code>\use:N</code>	36	<code>\Write</code>	12

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:nn
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, ee, ex, xo, xe, xx }

(End definition for \__oops_aux_prop:nn.)

\__oops_aux_prop:w #1 : < key >
#2 : < value >
32 \tl_new:N \g__oops_option_expans_tl
33 \cs_new_protected:Npn \__oops_aux_prop:w #1 = #2 \q_stop
34 {
35   \exp_args:Nx
36   \use:c{\__oops_aux_prop:\g__oops_option_expans_tl}
37   { \tl_trim_spaces:n{#1} }
38   { \__oops_aux_inner:n{ \tl_trim_spaces:n{#2} } }
39   %^^A { \__oops_aux_inner:e{ \tl_trim_spaces:n{#2} } }% DEBUG
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_separ:nn #1 : < int >

```



```

#2 : < tokens >
54 \cs_new:Nn \__oops_aux_separ:nn
55 {
56   \int_case:nnTF {#1}
57   {
58     {1}
59     { \prg_replicate:nn{ 3 }{#2} }
60     {2}
61     {
62       { \use_i:nn #2 }
63       { \use_ii:nn #2 }
64       { \use_i:nn #2 }
65     }
66     {3}{#2}
67   }
68   { \c_empty_tl }
69   {
70     \msg_error:nnnn { __erw }
71     { separ }
72     { \exp_not:N \__oops_aux_separ:nn }
73     {#2}
74   }
75 }
76 \cs_generate_variant:Nn \__oops_aux_separ:nn { e }

```

(End definition for __oops_aux_separ:nn.)

```

\__oops_aux_separ:n #1 : < tokens >
77 \cs_new:Nn \__oops_aux_separ:n
78 {
79   \__oops_aux_separ:en{ \tl_count:n{#1} }{#1}
80 }

```

(End definition for __oops_aux_separ:n.)

```

\__oops_aux_val:Nn #1 : < seq >
#2 : < tl var name >
81 \cs_new_protected:Nn \__oops_aux_val:Nn
82 {
83   \seq_gclear_new:N \g__oops_aux_val_seq
84   \__oops_seq_from_prop:NNn \g__oops_aux_val_seq #1 { \__oops_prop_name:n{#2} }
85 }

```

(End definition for __oops_aux_val:Nn.)

2 log

```

\__oops_log_close:
86 \iow_new:N \g__oops_log_iow
87 \AtEndDocument{\iow_close:N \g__oops_log_iow}
88 \bool_set_false:N \g__oops_log_open_bool
89 \cs_new_protected:Nn \__oops_log_close:
90 {

```

```

91 \iow_close:N \g__oops_log_iow
92 \bool_gset_false:N \g__oops_log_open_bool
93 }

```

(End definition for __oops_log_close:.)

__oops_log_open:

```

94 \tl_new:N \g__oops_log_file_tl
95 \cs_new_protected:Nn \__oops_log_open:
96 {
97   \tl_gset:Nx \g__oops_log_to_tl{\g__oops_log_file_tl}
98   \iow_open:Nn \g__oops_log_iow {\g__oops_log_to_tl}
99   \bool_gset_true:N \g__oops_log_open_bool
100 }

```

(End definition for __oops_log_open:.)

__oops_log_read:n #1 : $\langle path \rangle$

```

101 \cs_new_protected:Nn \__oops_log_read:n
102 {
103   \file_input:n{#1}
104   \tl_log:n{read~from~#1}
105 }
106 \cs_generate_variant:Nn \__oops_log_read:n { e }

```

(End definition for __oops_log_read:n.)

__oops_log_read:

```

107 \cs_new_protected:Nn \__oops_log_read:
108 {
109   \__oops_log_read:ef\g__oops_log_to_tl}
110 }

```

(End definition for __oops_log_read:.)

__oops_log_write:n

```

111 \tl_new:N \g__oops_log_to_tl
112 \cs_new_protected:Nn \__oops_log_write:n
113 {
114   \bool_if:nTF{ \g__oops_log_open_bool }
115   {
116     \iow_now:Nn \g__oops_log_iow {#1}
117     \tl_log:n{ write~to~#1 }
118   }
119   { \msg_error:nnn{ __oops }{ iow }{ \g__oops_log_iow } }
120 }
121 \cs_generate_variant:Nn \__oops_log_write:n { e }

```

(End definition for __oops_log_write:n.)

3 make_key

```

\__oops_make_key:Nn #1 : < token >
#2 : < key >

122 \cs_new_protected:Nn \__oops_make_key:Nn
123 {
124   \exp_args:NNx
125   \ProvideDocumentCommand{#1}
126   { D<>{\g__oops_option_name_tl} }
127   {
128     \__oops_prop_item:nn{##1}{#2}
129   }
130 }
131 \cs_generate_variant:Nn \__oops_make_key:Nn {c}

(End definition for \__oops_make_key:Nn.)

\__oops_make_key:n #1 : < key >

132 \cs_new_protected:Nn \__oops_make_key:n
133 {
134   \__oops_make_key:cn{#1}{#1}
135 }
136 \cs_generate_variant:Nn \__oops_make_key:n { e }

(End definition for \__oops_make_key:n.)

\__oops_make_key:N #1 : < seq >

137 \cs_new_protected:Nn \__oops_make_key:N
138 {
139   \seq_map_function:NN #1 \__oops_make_key:e
140 }

(End definition for \__oops_make_key:N.)

```

4 make_oops

```

\__oops_make_oops_exp:nnn

141 \cs_new_protected:Nn \__oops_make_oops_exp:nnn
142 {
143   \__oops_aux_val:Nn \g__oops_aux_key_seq {#1}
144   \__oops_aux_outer_set:n{#3}
145   \__oops_aux_outer:n
146   {
147     \exp_args:NNf
148     \__oops_seq_use:Nn
149     \g__oops_aux_val_seq
150     {#2}
151   }
152 }

(End definition for \__oops_make_oops_exp:nnn.)

```

__oops_make_oops_key:nnn

```
153 \cs_new_protected:Nn \__oops_make_oops_key:nnn
154 {
155   \__oops_prop_if_exist:nTF{#1}
156   { \c_empty_tl }
157   { \__oops_prop_new:n{#1} }
158   \exp_args:No \__oops_aux_inner_set:n{#2}
159   \seq_set_from_clist:Nn \g__oops_aux_keyval_seq {#3}
160   \__oops_aux_prop:N \g__oops_aux_keyval_seq
161   \__oops_prop_append:Nn \g__oops_aux_prop {#1}
162   \__oops_aux_key:N \g__oops_aux_keyval_seq
163   \__oops_make_key:N \g__oops_aux_key_seq
164 }
```

(End definition for __oops_make_oops_key:nnn.)

__oops_make_oops_sideeffect:nnn

```
165 \cs_new_protected:Nn \__oops_make_oops_sideeffect:nnn
166 {
167   \__oops_make_oops_key:nnn{#1}{#2}{#3}
168   \bool_if:nTF{ \g__oops_log_open_bool }
169   {%^^A https://tex.stackexchange.com/questions/536597
170     \__oops_log_write:n
171     {
172       \begingroup
173       \def \__oops_log_entry { \Oops<#1>i{#2}{#3} } \expandafter
174       \endgroup \__oops_log_entry
175     }
176   }\c_empty_tl}
177 }
```

(End definition for __oops_make_oops_sideeffect:nnn.)

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

```
178 \def\OopsHook{\c_empty_tl}
179 \cs_new_protected:Npn \__oops_make_oops:nnnn #1 #2 #3 #4
180 {
181   \exp_args:NNx \DeclareDocumentCommand \Oops
182   {%^^A      2      3      4 5 6      7 8      9
183     D<>{#1} +o E{ i }{{#2}} m t+ s E{ s o }{{#3}{#4}} +o
184   }
185   {
186     \IfValueT{##2}{##2}
187     \__oops_make_oops_sideeffect:nnn{##1}{##3}{##4}
188     \IfBooleanT{##6}
189     {
190       \__oops_make_oops_exp:nnn{##1}{##7}{##8}
191     }
192     \bool_if:nTF{##5}
193     {
194       \gappto{\OopsHook}
```

```

195     {
196       \__oops_make_oops_sideeffect:nnn{##1}{##3}{##4}
197     }
198   }
199   {\c_empty_tl}
200   \IfValueT{##9}
201   {
202     \exp_not:n{ \Ops<##1>[##9] }
203   }
204 }
205 }

```

(End definition for __oops_make_oops:nnnn.)

5 msg

```

206 \msg_new:nnn {\__oops}{ generic }{#1}
207 \msg_new:nnn {\__oops}{ iow }{#1~is~closed~can't~write}
208 \msg_new:nnn {\__oops}{ keyonly }{#1~does~not~take~values;~keyval~is~#2}
209 \msg_new:nnn {\__oops}{ keywrong }{#1~does~not~recognize~key~#2}
210 \msg_new:nnn {\__oops}{ separ }{#1~expects~1~to~3~items,~#2}
211 \msg_new:nnn {\__oops}{ unset }{#1~unset}

```

6 option

```

\__oops_aux_inner:n #1: <code>
212 \cs_new_protected:Nn \__oops_option_inner:n
213 {
214   \tl_gset:Nn \g__oops_option_inner_tl {#1}
215 }
216 \__oops_option_inner:n
217 {
218   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_inner_tl }
219 }

```

(End definition for __oops_aux_inner:n.)

```

\__oops_option_name:n #1: <token list>
220 \cs_new:Nn \__oops_option_name:n
221 {
222   \tl_gset:Nn \g__oops_option_name_tl{#1}
223 }
224 \__oops_option_name:n
225 {
226   \msg_error:nnx{ __oops }
227   { generic }
228   { \exp_not:N\g__oops_option_name_tl~undefined }
229 }

```

(End definition for __oops_option_name:n.)

```

\__oops_option_outer:n #1: < inline code >
230 \cs_new_protected:Nn \__oops_option_outer:n
231 {
232   \tl_gset:Nn \g__oops_option_outer_tl {#1}
233 }
234 \__oops_option_outer:n
235 {
236   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_outer_tl }
237 }

```

(End definition for __oops_option_outer:n.)

```

\__oops_option_separ:n #1: {< tl1>}{< tl2>}{< tl3>}
238 \cs_new_protected:Nn \__oops_option_separ:n
239 {
240   \cs_gset:Npn \g__oops_option_separ_tl {#1}
241 }
242 \__oops_option_separ:n
243 {
244   \msg_warning:nnn{ __oops }{ unset }{ \exp_not:N \g__oops_option_separ_tl }
245 }

```

(End definition for __oops_option_separ:n.)

7 prop

```

\__oops_prop_append:NN #1: < prop1 >
#2: < prop2 >
246 \cs_new_protected:Npn \__oops_prop_append:NN #1 #2
247 {
248   \cs_set:Nn \__oops_prop_append:nn
249   {
250     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
251   }
252   \prop_map_function:NN #2 \__oops_prop_append:nn
253 }
254 \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 >
255 \cs_new_protected:Nn \__oops_prop_append:Nn
256 {
257   \__oops_prop_append:cN{ \__oops_prop_name:n {#2} } #1
258 }

```

(End definition for __oops_prop_append:Nn.)

```

\__oops_prop_clear_new:n #1: < tl var name >
259 \cs_new_protected:Nn \__oops_prop_clear_new:n
260 {
261   \exp_args:No \prop_clear_new:c{ \__oops_prop_name:n {#1} }
262 }

```

(End definition for _oops_prop_clear_new:n.)

```
\_oops_prop_clear_new_map:n #1 : < keyval list >
263 \cs_new_protected:Nn \_oops_prop_clear_new_map:n
264 {
265   \seq_set_from_clist:Nn \g__oops_aux_key_seq {#1}
266   \seq_map_function:NN \g__oops_aux_key_seq \_oops_prop_clear_new:n
267 }
```

(End definition for _oops_prop_clear_new_map:n.)

```
\_oops_prop_if_exist:nTF #1 : < tl_1 >
#2 : < tl_2 >
#3 : < tl_3 >
268 \cs_new:Nn \_oops_prop_if_exist:nTF
269 {
270   \prop_if_exist:cTF{ \_oops_prop_name:n {#1} }{#2}{#3}
271 }
```

(End definition for _oops_prop_if_exist:nTF.)

```
\_oops_prop_item:nn #1 : < tl var name >
#2 : < key >
272 \cs_new:Nn \_oops_prop_item:nn
273 {
274   \prop_item:cn { \_oops_prop_name:n {#1} } {#2}
275 }
```

(End definition for _oops_prop_item:nn.)

```
\_oops_prop_name:n #1 : < tl var name >
276 \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 >
277 \cs_new_protected:Nn \_oops_prop_new:n
278 {
279   \prop_new:c{ \_oops_prop_name:n {#1} }
280 }
```

(End definition for _oops_prop_new:n.)

8 seq

```

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

281 \cs_new_protected:Nn \__oops_seq_from_prop:NNn
282 {
283   \cs_set_protected:Nn \__oops_seq_from_prop:n
284   {
285     \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
286   }
287   \seq_map_function:NN #2 \__oops_seq_from_prop:n
288 }

```

(End definition for __oops_seq_from_prop:NNn.)

__oops_erw_seq_use:Nn

```

289 % \begin{arguments}
290 % \item \meta{ seq }
291 % \item \meta{ tokens }
292 % \end{arguments}
293 \cs_new:Nn \__oops_seq_use:Nn
294 {
295   \exp_last_unbraced:NNf
296   \seq_use:Nnnn #1
297   \__oops_aux_separ:n{#2}
298 }

```

(End definition for __oops_erw_seq_use:Nn.)

9 Front-end

```

299 \keys_define:nn { __oops }
300 {
301   Expans .multichoices:nn =
302   { eo, ee, ex, xo, xe, xx }
303   { \tl_gset_eq:NN \g__oops_option_expans_tl \l_keys_choice_tl },
304   Expans .default:n = { xo },
305   Expans .initial:n = { xo },
306   File .code:n = { \tl_gset:Nn \g__oops_log_file_tl{ \exp_not:n{ #1 } } },
307   File .default:n = { oops\pdfcreationdate },
308   File .initial:n = { oops\pdfcreationdate },
309   Name .code:n={
310     \__oops_option_name:n{#1}
311     \exp_last_unbraced:Nf
312     \__oops_make_oops:nnnn
313     {
314       { \g__oops_option_name_tl }
315       { \g__oops_option_inner_tl }
316       { \g__oops_option_separ_tl }
317       { \g__oops_option_outer_tl }
318     }
319   },

```



```

320 Name .value_required:n = false,
321 Name .default:n = { Math },
322 Name .initial:n = { Math },
323 Inner .code:n={
324   \__oops_option_inner:n{#1}
325   \exp_last_unbraced:Nf
326   \__oops_make_oops:nnnn
327   {
328     { \g__oops_option_name_tl }
329     { \g__oops_option_inner_tl }
330     { \g__oops_option_separ_tl }
331     { \g__oops_option_outer_tl }
332   }
333 },
334 Inner .value_required:n = false,
335 Inner .default:n = {####1},
336 Inner .initial:n = {####1},
337 Outer .code:n={
338   \__oops_option_outer:n{#1}
339   \exp_last_unbraced:Nf
340   \__oops_make_oops:nnnn
341   {
342     { \g__oops_option_name_tl }
343     { \g__oops_option_inner_tl }
344     { \g__oops_option_separ_tl }
345     { \g__oops_option_outer_tl }
346   }
347 },
348 Outer .value_required:n = false,
349 Outer .default:n = { \ensuremath{####1} },
350 Outer .initial:n = { \ensuremath{####1} },
351 Write .code:n = {
352   \bool_if:nTF{#1}
353   {\__oops_log_open:}
354   {\__oops_log_close:}
355 },
356 Write .value_required:n = false,
357 Write .default:n = \BooleanFalse,
358 Write .initial:n = \BooleanFalse,
359 Separ .code:n={
360   \__oops_option_separ:n{#1}
361   \exp_last_unbraced:Nf
362   \__oops_make_oops:nnnn
363   {
364     { \g__oops_option_name_tl }
365     { \g__oops_option_inner_tl }
366     { \g__oops_option_separ_tl }
367     { \g__oops_option_outer_tl }
368   }
369 },
370 Separ .value_required:n = false,
371 Separ .default:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
372 Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } }
373 }

```

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

```
374 \NewDocumentCommand{ \OpsClear }
375 { D<>{\g__oops_option_name_tl} }
376 {
377   \__oops_prop_clear_new_map:n{#1}
378 }
```

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

\OpsDebug

```
379 \NewDocumentCommand\OpsDebug{m}
380 {
381   \__oops_prop_if_exist:nTF{#1}
382   { \c_empty_tl }
383   { \__oops_prop_new:n{#1} }
384   \__oops_make_key:Nn \KeyA{KeyA}
385   \gappto{\OpsHook}
386   {
387     \__oops_prop_if_exist:nTF{#1}
388     { \c_empty_tl }
389     { \__oops_prop_new:n{#1} }
390     \__oops_make_key:Nn \KeyA{KeyA}
391   }
392 }
```

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

\OpsOption

```
393 \NewDocumentCommand{ \OpsOption }
394 { m }
395 {
396   \keys_set:nn{ __oops }{#1}
397 }
```

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

\OpsRead

```
398 \NewDocumentCommand{\OpsRead}
399 {o}
400 {
401   \IfValueTF{#1}
402   {\__oops_log_read:e{#1}}
403   {\__oops_log_read:}
404 }
```

(End definition for \OpsRead. This function is documented on page 6.)

10 Misc

```
405 \ExplSyntaxOff
```