

The ccool package*

Erwann Rogard[†] and Olympia Hadjiliadis[‡]

Released 2020/04/12

Abstract

ccool stands for Custom COntent Oriented for L^AT_EX, a concept pioneered by cool[2]¹. This is done using a minimalist interface built upon xparse[6]. Specifically, `\Ccool<name>` begins a series of instructions alternating between ‘text’ and macro definitions, that themselves optionally expand using predefined or inline rules. For example,

```
\Ccool<Math>[Let~]i{\mathbb{#1}}{ Nat = N, Real = R }*s{{~and~}}
[~denote the natural and real numbers.]{}
```

expands to: “Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.” As a side effect, `\Nat<Math>` encodes “ \mathbb{N} ” (and likewise for `\Real`). `Math` being the default for `<name>`, `<Math>` can be dropped. Optionally, the macros can be written to a file, and restored, which can be useful for typesetting documents sharing the same notational conventions.

Contents

I	Usage	3
0	Convention	3
1	Loading the package	3
2	\Ccool	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

*This file describes version v1.7, last revised 2020/04/12.

[†]firstname dot lastname AusTria gmail dot com

[‡]<http://math.hunter.cuny.edu/~olympia/>

¹Whereas cool provided predefined macros, ccool is tool for making custom macros.

3	\CcoolClear	5
4	\CcoolHook	5
5	\CcoolOption	5
5.1	Expans	5
5.2	File	5
5.3	Inner	5
5.4	Name	6
5.5	Outer	6
5.6	Separ	6
5.7	Write	6
6	\CcoolRead	6
7	Do's and dont's	6
 II Listing		8
	Listing 1. Preamble.	8
	Listing 2. Separators.	8
	Listing 3. Hello, world!	8
	Listing 4. Listing 3 read from file.	9
	Listing 5. Probability space.	9
	Listing 6. Listing 5 read from file.	9
	Listing 7. Mittelwertsatz für n Variable.	9
	Listing 8. Listing 7 read from file.	10
	Listing 9. CUSUM statistic	10
 III Other		12
1	Acknowledgment	12
2	Install	12
3	Issue	12
4	Support	12
5	Testing	12
 Change History		14

Index	14
IV Implementation	17
1 aux	17
2 log	20
3 make_key	21
4 make_ccool	22
5 msg	23
6 option	23
7 prop	24
8 seq	26
9 Front-end	26
10 Misc	28

Part I

Usage

Convention

1. Loosely, those of [4] and [6], for example as to the meaning of $\langle token\ list \rangle$.
2. If unspecified, the environment in which a macro must be declared is `document`.

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

Environment *preamble*

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

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

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

$\langle tl_1 \rangle$

Example Math, ModelA, ModelB

Semantics Identifies a group of macros

$\langle tl_2 \rangle$

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= $\{\omega_1, \dots, \omega_n\}$, Sample= Ω

Semantics

2. $\backslash\langle key_i \rangle\langle tl_1 \rangle \leftarrow \langle val_i \rangle$ defined in step 1, using **Expans** for expansion.
3. If **Write**, writes the input used by step 2 to **File**

$+$

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

Semantics Appends step 2 and step 3 to **\CcoolHook**

$*$

Semantics

4. Expands $\langle code_2 \rangle$ applied to the list created in step 1, using the separator specified by $\langle tl_3 \rangle$, $\langle tl_4 \rangle$, $\langle tl_5 \rangle$.

$\langle tl_3 \rangle$

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

$\langle tl_4 \rangle$

Example $\{,\sim\}$

$\langle tl_5 \rangle$

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

$\langle code_2 \rangle$

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

$\langle tl_6 \rangle$

Semantics $\backslash\text{Ccool}\langle tl_1 \rangle > [\langle tl_6 \rangle]$

$\backslash\text{CcoolClear}$ $\backslash\text{CcoolClear}\langle keyval list \rangle$

Semantics Clears any data created by $\backslash\text{Ccool}\{\langle tl_1 \rangle\}$, for all $\langle tl_1 \rangle$ in $\langle keyval list \rangle$

$\backslash\text{CcoolHook}$ $\backslash\text{CcoolHook}$

Example $\backslash\text{AfterEndEnvironment}\{\text{theorem}\}\{\backslash\text{CcoolHook}\}$

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

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

Expans

Default xo

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

File

Default $\text{ccool}\backslash\text{pdfcreationdate}$

Syntax $Token$

Inner

Default $####1$

Semantics Default for $\langle code_1 \rangle$

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

Name

Default Math

Semantics Default for $\langle tl_1 \rangle$

Outer

Default `\ensuremath{####1}`

Semantics Default for $\langle code_2 \rangle$

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

Separ

Default `{ {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } }`

Semantics Default for **separators' parameter**

Syntax That of 'separators' in [4, Section 8 of l3seq]

Write

Default `\BooleanFalse`

Syntax *Boolean*

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

Other The default for $\langle path \rangle$ is the last write-file (see $\langle kvl_1 \rangle$)

Semantics

1. Reads the definitions in $\langle path \rangle$.
2. Writes to `ccool.log`: 'read from $\langle path \rangle$ '

Do's and dont's

1.

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

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

2.

Don't: $\$\langle key_i \rangle < x \$$.

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

3.

Don't: `\Ccool[[a, b]]`.

Do: `\Ccool{\{[]a, b{}}}`

4.

Don't: `\Ccool{ F = \cal F }`.

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

5. Also see [Part III, section 3](#)

Part II

Listing

Listing 1. Preamble^a

^aCheck the documentation portion of the source file, `ccool.dtx`, for exhaustive settings

```
% \usepackage{amsmath, amsthm, commath}
% \usepackage[T1]{fontenc}% \char`[
%
```

Listing 2. Separators

```
% \CcoolOption{
% ^^A% spaces betw. inner and outer brackets matter!->
% Separ={\ \char`@\ }{\ \%\ }{\ \char`@\ }}
% \Ccool<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 3. Hello, world!^a

^aIf this looks arcane, it's for the purpose of testing.

```
% \CcoolOption{ Separ = {\}\{.\}\{.}, Outer = {####1} }
% \CcoolOption{ Write = \BooleanTrue }
% \Ccool<Test>
% { KeyA = {.\}, KeyB = {!}, KeyC = {\%} }[]
% { KeyD = {d}, KeyE = {\%} }[]i{\#1\}
% { KeyF = {H}, KeyG = {e}, KeyH = {1} }*[]
% { KeyI = {\%}, KeyJ = {\%}, KeyK = {\%} }[.\{1\}.\{o\}]
% { KeyL = {1}, 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>\
% \CcoolOption{ Write = \BooleanFalse }
%
```

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

Listing 4. Listing 3 read from file.

```
% \CcoolRead
% \KeyF<Test>\KeyA<Test>\nobreak
% \KeyG<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}, [world!]

Listing 5. Probability space

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

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

Listing 6. Listing 5 read from file.

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

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

Listing 7. Mittelwertsatz für n Variable

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

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

(Check: \mathbb{N} , ξ)

Listing 8. Listing 7 read from file.

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

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

Listing 9. CUSUM statistic[7]

```
% \newtheorem{definition}{Definition}
% \AfterEndEnvironment{definition}{\CcoolHook}
% \NewDocumentCommand\EvalAt{m}{(1)}
%
% \Ccool{ SuchThat = { ;~ }, Time = { t }, Process = { \xi }, StopT
% = T }
% [The CUSUM statistic process and the corresponding one-sided CUSUM
```

```

stopping time are defined as follows:
% \begin{definition}\label{the CUSUM statistic} Let~]
% { Scale = { \lambda }, Real = {\mathcal{R}} }+*s{{\in}}[~and~]
% { CUSUMthresh = { \nu } }+*o{{\in}}\Real^{+}$.}
% [~Define the following processes:]
% { LogWald = { u }, CUSUMst = { \StopT_{c} }, CUSUM = { y },
LogWaldInf = { m } }+
% [\begin{enumerate}
% \item{\$ \LogWald_{\Time} \EvalAt{ \Scale } =
\Scale \Process_{\Time} - \frac{1}{2} \Scale^2 \Time$;
% \$ \LogWaldInf_{\Time} \EvalAt{ \Scale } = \inf_{0 \le s \le
\Time } \CUSUM_{s} \EvalAt{ \Scale }$.}
% \item{\$ \CUSUM_{\Time} \EvalAt{ \Scale } =
\LogWaldInf_{\Time} \EvalAt{ \Scale } - \LogWald_{\Time} \EvalAt{
\Scale } \ge 0$, which is the CUSUM statistic process.}
% \item{\$ \CUSUMst \EvalAt{ \Scale, \LogWaldInf } = \inf \left[
\Time \ge 0 \text{ \textit{SuchThat} } \CUSUM_{\Time} \EvalAt{\Scale} \ge \LogWaldInf
\right]$, which is the CUSUM stopping time.}
\end{enumerate}\end{definition}\par]{
%
% (Check: \$\Scale$, \$\CUSUM$)
%

```

The CUSUM statistic process and the corresponding one-sided CUSUM stopping time are defined as follows:

Definition 1 Let $\lambda \in \mathcal{R}$ and $\nu \in \mathcal{R}^+$. Define the following processes:

1. $u_t(\lambda) = \lambda \xi_t - \frac{1}{2} \lambda^2 t$; $m_t(\lambda) = \inf_{0 \leq s \leq t} y_s(\lambda)$.
2. $y_t(\lambda) = m_t(\lambda) - u_t(\lambda) \geq 0$, which is the CUSUM statistic process.
3. $T_c(\lambda, m) = \inf [t \geq 0; y_t(\lambda) \geq m]$, which is the CUSUM stopping time.

(Check: λ, y)

Part III

Other

1 Acknowledgment

This work has benefited from Q&A's from the L^AT_EX community, see here: <https://tex.stackexchange.com/users/112708/erwann?tab=questions>. Specific references are made in Part IV. Listing 5 and Listing 6 are from [1]. Listing 7 is from tcolbox[5, 17.3].

2 Install

Compiling ccool.dtx² will generate ccool.sty and ccool.pdf

3 Issue

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

4 Support

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

5 Testing

It's not possible to check the expansion of a certain class of macros against predefined values[8]. Instead, one can check that Part II, as generated in section 2 on one's own machine, agrees with bench.pdf available at <https://github.com/rogard/ccool>,

References

- [1] A.N. Shiryaev *Probability* Springer, 1995
- [2] Nick Setzer *The cool package*, 2005, <https://www.ctan.org/pkg/cool>
- [3] Olympia Hadjiliadis *Change-point detection of two-sided alternatives in the Brownian motion model and its connection to the gambler's ruin problem with relative wealth perception*. PhD thesis. Columbia University, 2005
- [4] The L^AT_EX3 Project Team *The L^AT_EX3 interfaces*, 2019, <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/interface3.pdf>

²Under Unix, `$tex ccool.dtx`

- [5] Thomas F. Sturm *The tcolorbox package*, 2019, <http://www.texdoc.net/texmf-dist/doc/latex/tcolorbox/tcolorbox.pdf>
- [6] The L^AT_EX3 Project Team *The xparse package*, 2020, <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf>
- [7] Erwann Rogard and Olympia Hadjiliadis *Typesetting a math thesis with ccool*, 2020, <https://github.com/rogard/ccool/blob/master/thesis.pdf>
- [8] <https://tex.stackexchange.com/a/534100/112708>

Change History

v1.0		Added: <code>\OpsDebug</code>	13
General: Initial version	13	Added: <code>\OpsHook</code>	13
v1.1		Added: <code>Expans</code> (for debugging sake, but...)	13
General: Added: <code>Save</code>	13	Added: Listing 1., 2., and 3.	13
Added: Listing 1., 2., 3., 4., 6., and 9.	13	Deleted: Listing 1., and 2.	13
Added: <code>\OpsRestore</code>	13	Replaced: <code>s{\langle tl_3 \rangle}{\langle tl_4 \rangle}{\langle tl_5 \rangle}</code> by <code>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}</code>	13
Added: <code>\OpsTest</code>	13		
Deleted: Listing 1-5 from v1.0 . . .	13		
Fixed: apparent anomaly in v1.0's Listing 4, see Listing 2	13	v1.5	
Replaced: <code>\OpsOptions</code> by <code>\OpsOption</code> . .	13	General: Added: <code>File</code>	13
Replaced: <code>{\langle kvl_2 \rangle}</code> by <code><kvl_2></code> given that option type G not recommended[6] .	13	Deleted: dependence on <code>datetime</code> .	13
Replaced: <code>GenericObject</code> by <code>Name</code> .	13	v1.6	
Replaced: <code>Separators</code> by <code>Separ</code> . .	13	General: Added: Listing 1 (preamble) .	13
Revamped: much of the implementation	13	Renamed: <code>\OpsClear</code> to <code>\CcoolClear</code>	13
v1.2		Renamed: <code>\OpsDebug</code> to <code>\CcoolDebug</code>	13
General: Deleted: <code>\OpsTest</code>	13	Renamed: <code>\OpsHook</code> to <code>\CcoolHook</code>	13
Deleted: <code>\langle kvl_2 \rangle</code> and <code>\langle code_2 \rangle</code>	13	Renamed: <code>\OpsOption</code> to <code>\CcoolOption</code>	13
Deleted: Listing 2-3 from v1.1. . . .	13	Renamed: <code>\OpsRead</code> to <code>\CcoolRead</code>	13
Replaced: <code>\OpsClear{\langle tl_1 \rangle}</code> by <code>\OpsClear[\langle keyval list \rangle]</code>	13	Renamed: <code>\Ops</code> to <code>\Ccool</code>	13
Replaced: <code>\Restore</code> by <code>\Read</code>	13	Renamed: <code>oops</code> to <code>ccool</code> (better describes the purpose)	13
Replaced: <code>\Save</code> by <code>\Write</code>	13		
v1.3		v1.7	
General: Replaced: <code>\OpsNew</code> by <code>\Ops</code> .	13	General: Added: Co-author	13
Replaced: <code>{\langle tl_1 \rangle}</code> and <code>[\langle tl_1 \rangle]</code> by <code><\langle tl_1 \rangle></code>	13	Added: Legends to listings	13
v1.4		Added: Listing 9 (CUSUM)	13
General: Added: section 7	13	Removed: <code>\CcoolDebug</code>	13
		Removed: Listing 5 from v1.6	13

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)	<i>4</i>
+ (option)	<i>4</i>
<code>\key_i</code>	<i>4, 6</i>
<code>\code₁</code> (option)	<i>4</i>
<code>\code₂</code> (option)	<i>5</i>
<code>\kvl₁</code> (option)	<i>4</i>
<code>\tl₁</code> (option)	<i>4</i>
<code>\tl₂</code> (option)	<i>4</i>
<code>\tl₃</code> (option)	<i>4</i>
<code>\tl₄</code> (option)	<i>5</i>
<code>\tl₅</code> (option)	<i>5</i>
<code>\tl₆</code> (option)	<i>5</i>
<code>Expans</code> (option)	<i>5</i>
<code>File</code> (option)	<i>5</i>
<code>Inner</code> (option)	<i>5</i>

Name (option)	6	_ccool_log_write:n	110, 169
Outer (option)	6	_ccool_make_ccool:nnnn	
Separ (option)	6	177, 311, 325, 339, 361	
Write (option)	6	_ccool_make_ccool_exp:nnn	140, 189
		_ccool_make_ccool_key:nnn	152, 166
		_ccool_make_ccool_sideeffect:nnn	164, 186, 195
_	370, 371	_ccool_make_key:N	136, 162
A			
\\AtEndDocument	86	_ccool_make_key:n	131, 138
B			
\\begin	288	_ccool_make_key:Nn	121, 133
\\begingroup	171	\\g_ccool_option_expans_tl	32, 36, 302
bool commands:		_ccool_option_inner:n	211, 215, 323
\\bool_gset_false:N	91	\\g_ccool_option_inner_tl	213, 217, 314, 328, 342, 364
\\bool_gset_true:N	98	_ccool_option_name:n	219, 309
\\bool_if:nTF	113, 167, 191, 351	\\g_ccool_option_name_tl	125, 221, 227, 313, 327, 341, 363, 374
\\bool_set_false:N	87	_ccool_option_outer:n	229, 337
\\BooleanFalse	356, 357	\\g_ccool_option_outer_tl	231, 235, 316, 330, 344, 366
C			
\\Ccool	1, 4, 4, 5, 5, 5, 7, 14, 172, 180, 201	_ccool_option_separ:n	237, 359
ccool internal commands:		\\g_ccool_option_separ_tl	239, 243, 315, 329, 343, 365
_ccool_aux_inner:n	6, 7, 38, 211	_ccool_prop_append:NN	245, 256
_ccool_aux_inner_set:n	4, 157	_ccool_prop_append:Nn	160, 254
_ccool_aux_key:N	17, 161	_ccool_prop_append:nn	247, 251
_ccool_aux_key:n	13, 20	_ccool_prop_clear_new:n	258, 265
_ccool_aux_key:w	9, 15	_ccool_prop_clear_new_map:n	262, 376
\\g_ccool_aux_key_seq	11, 19, 142, 162, 264, 265	_ccool_prop_if_exist:nTF	154, 267
\\g_ccool_aux_keyval_seq	158, 159, 161	_ccool_prop_item:nn	127, 271
_ccool_aux_outer:n	24, 144	_ccool_prop_name:n	83, 256, 260, 269, 273, 275, 278
_ccool_aux_outer_set:n	22, 143	_ccool_prop_new:n	156, 276
\\g_ccool_aux_prop	26, 29, 46, 160	_ccool_seq_from_prop:n	282, 286
_ccool_aux_prop:N	44, 159	_ccool_seq_from_prop:NNn	83, 280
_ccool_aux_prop:n	40, 50	_ccool_seq_use:Nn	147, 292
_ccool_aux_prop:nn	26	\\CcoolClear	2, 5, 14, 373
_ccool_aux_prop:w	32, 42	\\CcoolDebug	14
_ccool_aux_separ:n	76, 296	\\CcoolHook	2, 4, 5, 14, 177, 193
_ccool_aux_separ:nn	53, 78	\\CcoolOption	2, 5, 14, 378
_ccool_aux_val:Nn	80, 142	\\CcoolRead	2, 6, 12, 14, 383
\\g_ccool_aux_val_seq	82, 83, 148	cs commands:	
_ccool_erw_seq_use:Nn	288	\\cs_generate_variant:Nn	7, 31, 75, 105, 120, 130, 135, 253
_ccool_log_close:	85, 353	\\cs_gset:Npn	6, 24, 239
_ccool_log_entry	172, 173	\\cs_new:Nn	53, 76, 219, 267, 271, 292
\\g_ccool_log_file_tl	93, 96, 305	\\cs_new:Npn	275
\\g_ccool_log_iow	85, 86, 90, 97, 115, 118	\\cs_new_protected:Nn	4, 13, 17, 22, 27, 40, 44, 80, 88, 94, 100, 106, 111, 121, 131, 136, 140, 152, 164, 211, 229, 237, 254, 258, 262, 276, 280
_ccool_log_open:	93, 352	\\cs_new_protected:Npn	9, 33, 178, 245
\\g_ccool_log_open_bool	87, 91, 98, 113, 167	\\cs_set:Nn	247
_ccool_log_read:	106, 388		
_ccool_log_read:n	100, 108, 387		
\\g_ccool_log_log_to_tl	96, 97, 108, 110		

<code>\cs_set_protected:Nn</code>	282	<code>\msg_new:nnn</code> 205, 206, 207, 208, 209, 210	
		<code>\msg_warning:nnn</code>	217, 235, 243
D			
<code>\DeclareDocumentCommand</code>	180	N	
<code>\def</code>	172, 177	<code>\NeedsTeXFormat</code>	2
		<code>\NewDocumentCommand</code>	373, 378, 383
E			
<code>\end</code>	291	O	
<code>\endgroup</code>	173	<code>\Ops</code>	14
<code>\ensuremath</code>	348, 349	<code>\OpsClear</code>	14
exp commands:		<code>\OpsDebug</code>	14
<code>\exp_args:NNf</code>	146	<code>\OpsHook</code>	14
<code>\exp_args:NNx</code>	123, 180	<code>\OpsNew</code>	14
<code>\exp_args:No</code>	157, 260	<code>\OpsOption</code>	14
<code>\exp_args:Nx</code>	35	<code>\OpsOptions</code>	14
<code>\exp_last_unbraced:Nf</code>		<code>\OpsRead</code>	14
.	310, 324, 338, 360	<code>\OpsRestore</code>	14
<code>\exp_last_unbraced:NNf</code>	294	<code>\OpsTest</code>	14
<code>\exp_not:N</code>	71, 217, 227, 235, 243	options:	
<code>\exp_not:n</code>	201, 305	<code>*</code>	4
<code>\expandafter</code>	172	<code>+</code>	4
<code>\ExplSyntaxOff</code>	390	<code><code₁></code>	4
<code>\ExplSyntaxOn</code>	3	<code><code₂></code>	5
		<code><kv₁></code>	4
F			
file commands:		<code><tl₁></code>	4
<code>\file_input:n</code>	102	<code><tl₂></code>	4
		<code><tl₃></code>	4
G			
<code>\gappto</code>	193	<code><tl₄></code>	5
		<code><tl₅></code>	5
I			
<code>\IfBooleanT</code>	187	<code><tl₆></code>	5
<code>\IfValueT</code>	185, 199	Expans	5
<code>\IfValueTF</code>	386	File	5
int commands:		Inner	5
<code>\int_case:nnTF</code>	55	Name	6
iow commands:		Outer	6
<code>\iow_close:N</code>	86, 90	Separ	6
<code>\iow_new:N</code>	85	Write	6
<code>\iow_now:Nn</code>	115	P	
<code>\iow_open:Nn</code>	97	<code>\pdfcreationdate</code>	306, 307
<code>\item</code>	289, 290	prg commands:	
		<code>\prg_replicate:nn</code>	58
K			
keys commands:		prop commands:	
<code>\l_keys_choice_tl</code>	302	<code>\prop_clear_new:N</code>	260
<code>\keys_define:nn</code>	298	<code>\prop_gclear_new:N</code>	46
<code>\keys_set:nn</code>	381	<code>\prop_gput:Nnn</code>	29, 249
		<code>\prop_if_exist:NTF</code>	269
M			
<code>\meta</code>	289, 290	<code>\prop_item:Nn</code>	249, 273, 284
msg commands:		<code>\prop_map_function:NN</code>	251
<code>\msg_error:nnn</code>	118, 225	<code>\prop_new:N</code>	26, 278
<code>\msg_error:nnnn</code>	69	<code>\ProvideDocumentCommand</code>	124
		Q	
		quark commands:	
		<code>\q_stop</code>	9, 15, 33, 42

R		\tl_count:n 78
\Read 14		\tl_gset:Nn 96, 213, 221, 231, 305
\Restore 14		\tl_gset_eq:NN 302
S		\tl_log:n 103, 116
\Save 14		\tl_new:N 32, 93, 110
seq commands:		\tl_trim_spaces:n 11, 37, 38
\seq_gclear_new:N 19, 82		
\seq_gput_right:Nn 11, 284		U
\seq_if_empty:NTF 47	use commands:	
\seq_map_function:NN	\use:N 36	
. 20, 50, 138, 265, 286	\use_i:nn 61, 63	
\seq_set_from_clist:Nn 158, 264	\use_ii:nn 62	
\seq_use:Nnnn 295	\usepackage 3	
T		W
tl commands:		
\c_empty_tl 48, 67, 155, 175, 177, 198	\Write 14	

Part IV

Implementation

```

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

```

1 aux

```

\__ccool_aux_inner_set:n #1: <code>

4 \cs_new_protected:Nn \__ccool_aux_inner_set:n
5 {
6   \cs_gset:Npn \__ccool_aux_inner:n ##1 {#1}
7   \cs_generate_variant:Nn \__ccool_aux_inner:n { e }
8 }

(End definition for \__ccool_aux_inner_set:n.)

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

9 \cs_new_protected:Npn \__ccool_aux_key:w #1 = #2 \q_stop
10 {
11   \seq_gput_right:Nx \g__ccool_aux_key_seq { \tl_trim_spaces:n{#1} }
12 }

(End definition for \__ccool_aux_key:w.)

\__ccool_aux_key:n #1: <key = value>

13 \cs_new_protected:Nn \__ccool_aux_key:n
14 {
15   \__ccool_aux_key:w #1 \q_stop
16 }

```

(End definition for _ccool_aux_key:n.)

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

(End definition for _ccool_aux_key:N.)

```
\_ccool_aux_outer_set:n #1 : < inline code >
22 \cs_new_protected:Nn \_ccool_aux_outer_set:n
23 {
24   \cs_gset:Npn \_ccool_aux_outer:n ##1 {#1}
25 }
```

(End definition for _ccool_aux_outer_set:n.)

```
\_ccool_aux_prop:nn
26 \prop_new:N \g__ccool_aux_prop
27 \cs_new_protected:Nn \_ccool_aux_prop:nn
28 {
29   \prop_gput:Nnn \g__ccool_aux_prop{#1}{#2}
30 }
31 \cs_generate_variant:Nn \_ccool_aux_prop:nn { eo, ee, ex, xo, xe, xx }
```

(End definition for _ccool_aux_prop:nn.)

```
\_ccool_aux_prop:w #1 : < key >
#2 : < value >
32 \tl_new:N \g__ccool_option_expans_tl
33 \cs_new_protected:Npn \_ccool_aux_prop:w #1 = #2 \q_stop
34 {
35   \exp_args:Nx
36   \use:c{\_ccool_aux_prop:\g__ccool_option_expans_tl}
37   { \tl_trim_spaces:n{#1} }
38   { \_ccool_aux_inner:n{ \tl_trim_spaces:n{#2} } }
39 }
```

(End definition for _ccool_aux_prop:w.)

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

(End definition for _ccool_aux_prop:n.)

```

\__ccool_aux_prop:N #1 : <keyval list>

44 \cs_new_protected:Nn \__ccool_aux_prop:N
45 {
46   \prop_gclear_new:N \g__ccool_aux_prop
47   \seq_if_empty:NTF #1
48   { \c_empty_tl }
49   {
50     \seq_map_function:NN #1 \__ccool_aux_prop:n
51   }
52 }

(End definition for \__ccool_aux_prop:N.)

\__ccool_aux_separ:nn #1 : <int>
#2 : <tokens>

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

(End definition for \__ccool_aux_separ:nn.)

\__ccool_aux_separ:n #1 : <tokens>

76 \cs_new:Nn \__ccool_aux_separ:n
77 {
78   \__ccool_aux_separ:en{ \tl_count:n{#1} }{#1}
79 }

(End definition for \__ccool_aux_separ:n.)

\__ccool_aux_val:Nn #1 : <seq>

```

```

#2 : < tl var name >

80 \cs_new_protected:Nn \__ccool_aux_val:Nn
81 {
82   \seq_gclear_new:N \g__ccool_aux_val_seq
83   \__ccool_seq_from_prop:Nn \g__ccool_aux_val_seq #1 { \__ccool_prop_name:n{#2} }
84 }

(End definition for \__ccool_aux_val:Nn.)

```

2 log

```

\__ccool_log_close:

85 \iow_new:N \g__ccool_log_iow
86 \AtEndDocument{\iow_close:N \g__ccool_log_iow}
87 \bool_set_false:N \g__ccool_log_open_bool
88 \cs_new_protected:Nn \__ccool_log_close:
89 {
90   \iow_close:N \g__ccool_log_iow
91   \bool_gset_false:N \g__ccool_log_open_bool
92 }

(End definition for \__ccool_log_close:.)

\__ccool_log_open:

93 \tl_new:N \g__ccool_log_file_tl
94 \cs_new_protected:Nn \__ccool_log_open:
95 {
96   \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
97   \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
98   \bool_gset_true:N \g__ccool_log_open_bool
99 }

(End definition for \__ccool_log_open:.)

\__ccool_log_read:n #1 : <path>

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

(End definition for \__ccool_log_read:n.)

\__ccool_log_read:

106 \cs_new_protected:Nn \__ccool_log_read:
107 {
108   \__ccool_log_read:e{\g__ccool_log_to_tl}
109 }

(End definition for \__ccool_log_read:.)

```

```

\__ccool_log_write:n
110 \tl_new:N \g__ccool_log_to_tl
111 \cs_new_protected:Nn \__ccool_log_write:n
112 {
113   \bool_if:nTF{ \g__ccool_log_open_bool }
114   {
115     \iow_now:Nn \g__ccool_log_iow {#1}
116     \tl_log:n{ write~to~#1 }
117   }
118   { \msg_error:nnn{ __ccool }{ iow }{ \g__ccool_log_iow } }
119 }
120 \cs_generate_variant:Nn \__ccool_log_write:n { e }

(End definition for \__ccool_log_write:n.)

```

3 make_key

```

\__ccool_make_key:Nn #1 : < token >
#2 : < key >
121 \cs_new_protected:Nn \__ccool_make_key:Nn
122 {
123   \exp_args:NNx
124   \ProvideDocumentCommand{#1}
125   { D<>{ \g__ccool_option_name_tl } }
126   {
127     \__ccool_prop_item:nn{##1}{#2}
128   }
129 }
130 \cs_generate_variant:Nn \__ccool_make_key:Nn { c }

(End definition for \__ccool_make_key:Nn.)

```

```

\__ccool_make_key:n #1 : < key >
131 \cs_new_protected:Nn \__ccool_make_key:n
132 {
133   \__ccool_make_key:cn{#1}{#1}
134 }
135 \cs_generate_variant:Nn \__ccool_make_key:n { e }

(End definition for \__ccool_make_key:n.)

```

```

\__ccool_make_key:N #1 : < seq >
136 \cs_new_protected:Nn \__ccool_make_key:N
137 {
138   \seq_map_function:NN #1 \__ccool_make_key:e
139 }

(End definition for \__ccool_make_key:N.)

```

4 make_ccool

_ccool_make_ccool_exp:nnn

```

140 \cs_new_protected:Nn \_ccool_make_ccool_exp:nnn
141 {
142   \_ccool_aux_val:Nn \g\_ccool_aux_key_seq {#1}
143   \_ccool_aux_outer_set:n{#3}
144   \_ccool_aux_outer:n
145   {
146     \exp_args:Nnf
147     \_ccool_seq_use:Nn
148     \g\_ccool_aux_val_seq
149     {#2}
150   }
151 }
```

(End definition for _ccool_make_ccool_exp:nnn.)

_ccool_make_ccool_key:nnn

```

152 \cs_new_protected:Nn \_ccool_make_ccool_key:nnn
153 {
154   \_ccool_prop_if_exist:nTF{#1}
155   { \c_empty_tl }
156   { \_ccool_prop_new:n{#1} }
157   \exp_args:No \_ccool_aux_inner_set:n{#2}
158   \seq_set_from_clist:Nn \g\_ccool_aux_keyval_seq {#3}
159   \_ccool_aux_prop:N \g\_ccool_aux_keyval_seq
160   \_ccool_prop_append:Nn \g\_ccool_aux_prop {#1}
161   \_ccool_aux_key:N \g\_ccool_aux_keyval_seq
162   \_ccool_make_key:N \g\_ccool_aux_key_seq
163 }
```

(End definition for _ccool_make_ccool_key:nnn.)

_ccool_make_ccool_sideeffect:nnn

```

164 \cs_new_protected:Nn \_ccool_make_ccool_sideeffect:nnn
165 {
166   \_ccool_make_ccool_key:nnn{#1}{#2}{#3}
167   \bool_if:nTF{ \g\_ccool_log_open_bool }
168   {%^A https://tex.stackexchange.com/questions/536597
169     \_ccool_log_write:n
170     {
171       \begingroup
172       \def \_ccool_log_entry { \Ccool<#1>i{#2}{#3} } \expandafter
173       \endgroup \_ccool_log_entry
174     }
175   }{\c_empty_tl}
176 }
```

(End definition for _ccool_make_ccool_sideeffect:nnn.)

_ccool_make_ccool:nnnn #1 : \langle token list \rangle
 #2 : \langle seq₁ \rangle
 #3 : \langle seq₂ \rangle

```

#4 : < prop >

177 \def\CcoolHook{\c_empty_tl}
178 \cs_new_protected:Npn \__ccool_make_ccool:nnnn #1 #2 #3 #4
179 {
180   \exp_args:NNx \DeclareDocumentCommand \Ccool
181     {%^~A      2      3      4 5 6      7 8      9
182     D<>{#1} +o E{ i }{{#2}} m t+ s E{ s o }{{#3}{#4}} +o
183   }
184   {
185     \IfValueT{##2}{##2}
186     \__ccool_make_ccool_sideeffect:nnn{##1}{##3}{##4}
187     \IfBooleanT{##6}
188     {
189       \__ccool_make_ccool_exp:nnn{##1}{##7}{##8}
190     }
191     \bool_if:nTF{##5}
192     {
193       \gappto{\CcoolHook}
194       {
195         \__ccool_make_ccool_sideeffect:nnn{##1}{##3}{##4}
196       }
197     }
198     {\c_empty_tl}
199     \IfValueT{##9}
200     {
201       \exp_not:n{ \Ccool<##1>[##9] }
202     }
203   }
204 }

```

(End definition for __ccool_make_ccool:nnnn.)

5 msg

```

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

```

6 option

```

\__ccool_aux_inner:n #1 : <code>

211 \cs_new_protected:Nn \__ccool_option_inner:n
212 {
213   \tl_gset:Nn \g__ccool_option_inner_tl {#1}
214 }
215 \__ccool_option_inner:n
216 {
217   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_inner_tl }
218 }

```

(End definition for _ccool_aux_inner:n.)

```
\_ccool_option_name:n #1 : < token list >
219 \cs_new:Nn \_ccool_option_name:n
220 {
221   \tl_gset:Nn \g_ccool_option_name_tl{#1}
222 }
223 \_ccool_option_name:n
224 {
225   \msg_error:nnx{ __ccool }
226   { generic }
227   { \exp_not:N\g_ccool_option_name_tl~undefined }
228 }
```

(End definition for _ccool_option_name:n.)

```
\_ccool_option_outer:n #1 : < inline code >
229 \cs_new_protected:Nn \_ccool_option_outer:n
230 {
231   \tl_gset:Nn \g_ccool_option_outer_tl {#1}
232 }
233 \_ccool_option_outer:n
234 {
235   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g_ccool_option_outer_tl }
236 }
```

(End definition for _ccool_option_outer:n.)

```
\_ccool_option_separ:n #1 : {< tl1>}{< tl2>}{< tl3>}
237 \cs_new_protected:Nn \_ccool_option_separ:n
238 {
239   \cs_gset:Npn \g_ccool_option_separ_tl {#1}
240 }
241 \_ccool_option_separ:n
242 {
243   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g_ccool_option_separ_tl }
244 }
```

(End definition for _ccool_option_separ:n.)

7 prop

```
\_ccool_prop_append:NN #1 : < prop1 >
#2 : < prop2 >
245 \cs_new_protected:Npn \_ccool_prop_append:NN #1 #2
246 {
247   \cs_set:Nn \_ccool_prop_append:nn
248   {
249     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
250   }
251   \prop_map_function:NN #2 \_ccool_prop_append:nn
252 }
253 \cs_generate_variant:Nn \_ccool_prop_append:NN { cN }
```


(End definition for _ccool_prop_append:NN.)

```
\_ccool_prop_append:Nn #1 : < prop >
#2 : < tl var name >
254 \cs_new_protected:Nn \_ccool_prop_append:Nn
255 {
256   \_ccool_prop_append:cN{ \_ccool_prop_name:n {#2} } #1
257 }
```

(End definition for _ccool_prop_append:Nn.)

```
\_ccool_prop_clear_new:n #1 : < tl var name >
258 \cs_new_protected:Nn \_ccool_prop_clear_new:n
259 {
260   \exp_args:No \prop_clear_new:c{ \_ccool_prop_name:n {#1} }
261 }
```

(End definition for _ccool_prop_clear_new:n.)

```
\_ccool_prop_clear_new_map:n #1 : < keyval list >
262 \cs_new_protected:Nn \_ccool_prop_clear_new_map:n
263 {
264   \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
265   \seq_map_function:NN \g__ccool_aux_key_seq \_ccool_prop_clear_new:n
266 }
```

(End definition for _ccool_prop_clear_new_map:n.)

```
\_ccool_prop_if_exist:nTF #1 : < tl1 >
#2 : < tl2 >
#3 : < tl3 >
267 \cs_new:Nn \_ccool_prop_if_exist:nTF
268 {
269   \prop_if_exist:ctF{ \_ccool_prop_name:n {#1} }{#2}{#3}
270 }
```

(End definition for _ccool_prop_if_exist:nTF.)

```
\_ccool_prop_item:nn #1 : < tl var name >
#2 : < key >
271 \cs_new:Nn \_ccool_prop_item:nn
272 {
273   \prop_item:cn { \_ccool_prop_name:n {#1} } {#2}
274 }
```

(End definition for _ccool_prop_item:nn.)

```
\_ccool_prop_name:n #1 : < tl var name >
275 \cs_new:Npn \_ccool_prop_name:n #1{ __ccool_#1 }
```

(End definition for _ccool_prop_name:n.)

```

\__ccool_prop_new:n #1: < tl var name >
276 \cs_new_protected:Nn \__ccool_prop_new:n
277 {
278   \prop_new:c{ \__ccool_prop_name:n {#1} }
279 }

(End definition for \__ccool_prop_new:n.)

```

8 seq

```

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

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

(End definition for \__ccool_seq_from_prop:NNn.)

```

```

\__ccool_erw_seq_use:Nn
288 % \begin{arguments}
289 % \item \meta{ seq }
290 % \item \meta{ tokens }
291 % \end{arguments}
292 \cs_new:Nn \__ccool_seq_use:Nn
293 {
294   \exp_last_unbraced:NNf
295   \seq_use:Nnnn #1
296   \__ccool_aux_separ:n{#2}
297 }

(End definition for \__ccool_erw_seq_use:Nn.)

```

9 Front-end

```

298 \keys_define:nn { __ccool }
299 {
300   Expans .multichoices:nn =
301   { eo, ee, ex, xo, xe, xx }
302   { \tl_gset_eq:NN \g__ccool_option_expans_tl \l_keys_choice_tl },
303   Expans .default:n = { xo },
304   Expans .initial:n = { xo },
305   File .code:n = { \tl_gset:Nn \g__ccool_log_file_tl{ \exp_not:n{ #1 } } },
306   File .default:n = { ccool\pdfcreationdate },
307   File .initial:n = { ccool\pdfcreationdate },
308   Name .code:n={
309     \__ccool_option_name:n{#1}

```

```

310 \exp_last_unbraced:Nf
311 \__ccool_make_ccool:nnnn
312 {
313   { \g__ccool_option_name_tl }
314   { \g__ccool_option_inner_tl }
315   { \g__ccool_option_separ_tl }
316   { \g__ccool_option_outer_tl }
317 }
318 },
319 Name .value_required:n = false,
320 Name .default:n = { Math },
321 Name .initial:n = { Math },
322 Inner .code:n={
323   \__ccool_option_inner:n{#1}
324   \exp_last_unbraced:Nf
325   \__ccool_make_ccool:nnnn
326   {
327     { \g__ccool_option_name_tl }
328     { \g__ccool_option_inner_tl }
329     { \g__ccool_option_separ_tl }
330     { \g__ccool_option_outer_tl }
331   }
332 },
333 Inner .value_required:n = false,
334 Inner .default:n = {####1},
335 Inner .initial:n = {####1},
336 Outer .code:n={
337   \__ccool_option_outer:n{#1}
338   \exp_last_unbraced:Nf
339   \__ccool_make_ccool:nnnn
340   {
341     { \g__ccool_option_name_tl }
342     { \g__ccool_option_inner_tl }
343     { \g__ccool_option_separ_tl }
344     { \g__ccool_option_outer_tl }
345   }
346 },
347 Outer .value_required:n = false,
348 Outer .default:n = { \ensuremath{####1} },
349 Outer .initial:n = { \ensuremath{####1} },
350 Write .code:n = {
351   \bool_if:nTF{#1}
352   {\__ccool_log_open:}
353   {\__ccool_log_close:}
354 },
355 Write .value_required:n = false,
356 Write .default:n = \BooleanFalse,
357 Write .initial:n = \BooleanFalse,
358 Separ .code:n={
359   \__ccool_option_separ:n{#1}
360   \exp_last_unbraced:Nf
361   \__ccool_make_ccool:nnnn
362   {
363     { \g__ccool_option_name_tl }

```

```

364     { \g__ccool_option_inner_tl }
365     { \g__ccool_option_separ_tl }
366     { \g__ccool_option_outer_tl }
367   }
368 },
369 Separ .value_required:n = false,
370 Separ .default:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
371 Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } }
372 }

```

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

```

373 \NewDocumentCommand{ \CcoolClear }
374 { D<>{\g__ccool_option_name_tl} }
375 {
376   \__ccool_prop_clear_new_map:n{#1}
377 }

```

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

\CcoolOption

```

378 \NewDocumentCommand{ \CcoolOption }
379 { m }
380 {
381   \keys_set:nn{ __ccool }{#1}
382 }

```

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

\CcoolRead

```

383 \NewDocumentCommand{\CcoolRead}
384 {o}
385 {
386   \IfValueTF{#1}
387   {\__ccool_log_read:e{#1}}
388   {\__ccool_log_read:}
389 }

```

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

10 Misc

```

390 \ExplSyntaxOff

```