

The ccool package*

Erwann Rogard†

Released 2020/04/14

Abstract

ccool stands for Custom COntent Oriented for L^AT_EX, a concept pioneered by cool[1]¹. This is done using a minimalist interface built upon xparse[4]. 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{{~\rm{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 “N” (and likewise for `\Real`). `Math` being the default for `<name>`, `<Math>` can be dropped. In conjunction with lambda expressions, this tool allows for encoding the way certain mathematical objects, such as functions, should be formatted. Optionally, the macros can be written to a file, and read, which can be useful for typesetting documents sharing the same notation.

Contents

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

*This file describes version v1.9, last revised 2020/04/14.

†firstname dot lastname AusTria gmail dot com

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

3	Other	5
4	Do's and dont's	6
II	Listing	7
	Listing 1. Version.	7
	Listing 2. Preamble.	7
	Listing 3. Separators.	7
	Listing 4. Hello, world!	7
	Listing 5. Listing 4 read from file.	8
	Listing 6. Probability space.	8
	Listing 7. Listing 6 read from file.	8
	Listing 8. Mittelwertsatz für n Variable.	9
	Listing 9. Listing 8 read from file.	9
	Listing 10. Lambda expression.	9
	Listing 11. Listing 10 read from file.	10
	Listing 12. CUSUM statistic.	10
	Listing 13. Listing 12 read from file.	11
III	Other	12
1	Acknowledgment	12
2	Install	12
3	Issue	12
4	Support	12
5	Testing	12
	5.1 Technicality	12
	5.2 Platform	12
	5.3 Engine	12
	5.4 Results	13
	5.5 Other	13
	References	13

Change History	13
Index	14
IV Implementation	18
1 aux	18
2 lambda	20
3 log	21
4 make_key	22
5 make_ccool	23
6 msg	24
7 option	24
8 prop	25
9 seq	27
10 Front-end	27
11 \CcoolClear	27
12 \CcoolHook	27
12 \CcoolLambda	27
12 \CcoolOption	28
12.2 Expans	28
12.2 File	28
12.2 Inner	28
12.2 Name	29
12.2 Outer	29
12.2 Separ	29
12.3 Write	30
13 \CcoolRead	30
14 \CcoolVers	30

Semantics

1) $\langle val_i \rangle \leftarrow \langle code_1 \rangle$ applied to $\langle val_i \rangle$

$\langle kv1_1 \rangle$

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

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**

+

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

*

Semantics

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

$\langle tl_3 \rangle$

Default **Separ**

Example `{~\in~}`

$\langle tl_4 \rangle$

Default **Separ**

Example `{,~}`

$\langle tl_5 \rangle$

Default **Separ**

Example `{~\&~}`

$\langle code_2 \rangle$

Default **Outer**

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

$\langle tl_6 \rangle$

Semantics **\Ccool** $\langle tl_1 \rangle > [\langle tl_6 \rangle]$

²Needed inside a *local group*, for the side effect of **\Ccool** to persist thereafter.

Other

Continued in [Part IV, section 10](#).

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: `[a, b)`

Do: `{[]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.

```
% \CcoolVers
%
```

2020/04/14 v1.9 cool — A tool for encoding mathematical notation

Listing 2. Preamble^a

^aThese are the settings to replicate the listings. For exhaustivity, check the **documentation** section of `ccool.dtx`.

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

Listing 3. 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 = 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 4. 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 = {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>\
% \CcoolOption{ Write = \BooleanFalse }
%

```

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

Listing 5. Listing 4 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 6. 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 7. Listing 6 read from file.

```
% \CcoolRead \tab $\Omega$ $\mathcal{F}$ $\mathcal{P}$
%
```

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

Listing 8. Mittelwertsatz für n Variable[1, 17.3]

```
% \CcoolOption{ Write = \BooleanTrue }
% \newtheorem{theorem}{Theorem}
% \AfterEndEnvironment{theorem}{\CcoolHook}
% \Ccool 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] } }+
% [$n\in\mathbb{N}$, $\mathcal{O}\subseteq\mathbb{R}^n$ eine offene Menge und
% $f\in C^1(\mathcal{O})$].
% Dann gibt es auf jeder Strecke $[x_0,x]$ einen
% Punkt $\xi\in[x_0,x]$
% { Steig = { \frac{ f(x)-f(x_0) }{ x-x_0 } }, Punkt = { \xi } }+
% [so dass gilt
% \begin{equation*}
% \operatorname{grad} f(\xi) = \frac{ f(x)-f(x_0) }{ x-x_0 }
% \end{equation*}
% \end{theorem}]
% {}
% (Check: $n$, $\xi$)
% \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

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

(Check: \mathbb{N} , ξ)

Listing 9. Listing 8 read from file.

```
% \CcoolRead \tab $n$ $R$ $\mathcal{O}$ $Ci$ $Strecke$
%
```

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

Listing 10. Lambda expression.

```
% \CcoolOption{ Write = \BooleanTrue }
% \Ccool{ EvalAt = \CcoolLambda{(#1)}, ApplyOp =
% \CcoolLambda[2]{#1[#2]} }
% [Supposons une fonction  $f$ \EvalAt{t}$, et \etudions le probl\eme
% o\`u la fonctionnelle  $S$ \ApplyOp{S}{f}$ est donn\`ee par\dots]{ }
% \CcoolOption{ Write = \BooleanFalse }
%
```

Supposons une fonction $f(t)$, et étudions le problème où la fonctionnelle $S[f]$ est donnée par...

Listing 11. Listing 10 read from file.

```
% \CcoolRead \tab $f\EvalAt{t}$,  $S$ \ApplyOp{S}{f}$
%
```

$$f(t), S[f]$$

Listing 12. CUSUM statistic

```
% \newtheorem{definition}{Definition}
% \AfterEndEnvironment{definition}{\CcoolHook}
%
% \CcoolOption{ Write = \BooleanTrue }
% \Ccool{ SuchThat = { ;~ }, Time = { t }, Process = { \xi }, StopT =
% { T }, EvalAt = \CcoolLambda{(#1)} }
% [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{#1\in\Real^{+}}$.}
% [-Define the following processes:]
% { LogWald = { u }, CUSUMst = { \StopT_{c} }, CUSUM = { y },
% LogWaldInf = { m } }+
% [\begin{enumerate}
% \item{ $\LogWald_{\Time}\EvalAt{ \Scale } = \Scale\Process_{\Time}$ 
% -  $\frac{1}{2}\Scale^2\Time$ ;}
% \item{ $\LogWaldInf_{\Time}\EvalAt{ \Scale } = \inf_{0 \leq s \leq \Time}$ 
% } $CUSUM_{s}\EvalAt{ \Scale }$ .}
% \item{ $CUSUM_{\Time}\EvalAt{ \Scale } =$ 
%  $\LogWaldInf_{\Time}\EvalAt{ \Scale } - \LogWald_{\Time}\EvalAt{$ 
%  $\Scale } \geq 0$ , which is the CUSUM statistic process.}
% \item{ $CUSUMst \EvalAt{ \Scale, \LogWaldInf } = \inf\left[ \Time$ 
%  $\geq 0 \text{ \textit{SuchThat} } CUSUM_{\Time}\EvalAt{\Scale} \geq \LogWaldInf$ 
%  $\right]$ , which is the CUSUM stopping time.}
% \end{enumerate}\end{definition}\par{}
%
% (Check:  $\Scale$ ,  $CUSUM$ )
% \CcoolOption{ Write = \BooleanFalse }
```

%

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)

Listing 13. Listing 12 read from file.

```
%      \CcoolRead \tab $Time$ $Process$ $Scale$ $Real$ $CUSUMthresh$
      $LogWald$ $CUSUMst$ $CUSUM$ $LogWaldInf$
%
```

$t \xi \lambda \mathcal{R} \nu u T_c y m$

Part III

Other

1 Acknowledgment

This work has benefited from Q&A's from the L^AT_EXcommunity[6]. Specific attributions are made throughout this document.

2 Install

Compiling `ccool.dtx`³ 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

5.1 Technicality

Not possible to compile-check the expansion of a certain class of macros against predefined values[9]. Instead, one can visually check **Part II**, as generated in **section 2** on one's own machine, against that **of the repository** for the same version.

5.2 Platform

- i)* Linux laptop 4.15.0-20-generic #21-Ubuntu SMP Tue Apr 24
↪ 06:16:15 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux

5.3 Engine

- a)* pdfTeX 3.14159265-2.6-1.40.20 (TeX Live 2019)
- b)* pdfTeX 3.14159265-2.6-1.40.21 (TeX Live 2020)
- c)* LuaHBTeX, Version 1.12.0 (TeX Live 2020)

³Under Unix, `$tex ccool.dtx`

5.4 Results

1. ccool v1.8 satisfactory on platform *i*) and engine *a*)
2. ccool v1.8 satisfactory on platform *i*) and engine *b*)
3. ccool v1.9 satisfactory on platform *i*) and engines *b*) and *c*)

5.5 Other

Check [5] for testing ccool with llncs

References

- [1] Nick Setzer *The cool package*, 2005, <https://www.ctan.org/pkg/cool>
- [2] 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>
- [3] Thomas F. Sturm *The tcolorbox package*, 2019, <http://www.texdoc.net/texmf-dist/doc/latex/tcolorbox/tcolorbox.pdf>
- [4] The L^AT_EX3 Project Team *The xparse package*, 2020, <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf>
- [5] Erwann Rogard and Olympia Hadjiliadis *Typesetting a math thesis with ccool*, 2020, <https://github.com/rogard/ccool/blob/master/thesis.pdf>
- [6] <https://tex.stackexchange.com/users/112708/erwann?tab=questions>
- [7] @egreg’s answer to “What is the XeTeX equivalent of \pdfcreationdate?”, <https://tex.stackexchange.com/a/41893>
- [8] @sean-allred’s answer to “How to create lambda expressions?”, <https://tex.stackexchange.com/a/188053/112708>
- [9] @joseph-wright’s answer to “Checking a function’s expansion against a string”, <https://tex.stackexchange.com/a/534100>
- [10] @frougon’s answer to “Journaling calls to a function []”, <https://tex.stackexchange.com/a/536620>

Change History

v1.0	9.	13
General: Initial version	Added:\OpsRestore	13
v1.1	Added:\OpsTest	13
General: Added: Save	Deleted: Listing 1-5 from v1.0 . . .	13
Added: Listing 1., 2., 3., 4., 6., and	Fixed: apparent anomaly in v1.0’s	

Listing 4, see Listing 3	13	v1.5	General: Added: File	13
Replaced:			Deleted: dependence on <code>datetime</code>	13
\OpsOptions by \OpsOption	13	v1.6	General: Added: Listing 2 (preamble)	13
Replaced:			Renamed: \OpsClear to	
{\kvl ₂ } by <kvl ₂ > given that			\CcoolClear	13
option type G not recommended[4]	13		Renamed: \OpsDebug to	
Replaced: GenericObject by Name	13		\CcoolDebug	13
Replaced: Separators by Separ	13		Renamed: \OpsHook to	
Revamped: much of the			\CcoolHook	13
implementation	13		Renamed: \OpsOption to	
v1.2			\CcoolOption	13
General: Deleted: \OpsTest	13		Renamed: \OpsRead to	
Deleted: {\kvl ₂ } and {\code ₂ }	13		\CcoolRead	13
Deleted: Listing 2-3 from v1.1.	13		Renamed: \Ops to \Ccool	13
Replaced: \OpsClear{\tl ₁ }			Renamed: oops to ccool (better	
\OpsClear[\keyval list]	13		describes the purpose)	13
Replaced: \Restore by \Read	13	v1.7	General: Added: Legends to listings	13
Replaced: \Save by \Write	13		Added: Listing 12 (CUSUM)	13
v1.3			Deleted: \CcoolDebug	13
General: Replaced: \OpsNew by \Ops	13		Deleted: Listing 5 from v1.6	13
Replaced: {\tl ₁ } and [\tl ₁] by			v1.8	
<\tl ₁ >	13		General: Added: \CcoolLambda	13
v1.4			Added: \CcoolVers	13
General: Added: section 4	13		Added: Listing 10, Listing 11	13
Added: \OpsDebug	13		Added: Listing 1	13
Added: \OpsHook	13	v1.9	General: Added: support for LuaTeX	13
Added: Expans (for debugging			Moved: from Part I to Part IV,	
sake, but...)	13		what is now that part's section 10	13
Added: Listing 1., 2., and 3.	13			
Deleted: Listing 1., and 2.	13			
Replaced: s{\tl ₃ }{\tl ₄ }{\tl ₅ }				
by				
s{\tl ₃ }{\tl ₃ }{\tl ₄ }{\tl ₃ }{\tl ₄ }{\tl ₅ }				
	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)	5
+ (option)	4
\key _i	4, 5
\code ₁ (option)	4
\code ₂ (option)	5
\kvl ₁ (option)	4
\tl ₁ (option)	4
\tl ₂ (option)	4
\tl ₃ (option)	5
\tl ₄ (option)	5
\tl ₅ (option)	5
\tl ₆ (option)	5
Expans (option)	24
File (option)	24
Inner (option)	24
Name (option)	25
Outer (option)	25
Separ (option)	25
Write (option)	26
_	390, 391

A	
\AtEndDocument	101
B	
\begin	302
\begingroup	186
bool commands:	
\bool_gset_false:N	106
\bool_gset_true:N	113
\bool_if:nTF	128, 182, 205, 393
\bool_set_false:N	102
\BooleanFalse	398, 399
C	
\Ccool	1, 4, 4, 5, 23, 187, 194, 215
ccool internal commands:	
__ccool_aux_inner:n ...	6, 7, 38, 225
__ccool_aux_inner_set:n ...	4, 172
__ccool_aux_key:N	17, 176
__ccool_aux_key:n	13, 20
__ccool_aux_key:w	9, 15
\g__ccool_aux_key_seq	11, 19, 157, 177, 278, 279
\g__ccool_aux_keyval_seq	173, 174, 176
__ccool_aux_outer:n	24, 159
__ccool_aux_outer_set:n ...	22, 158
\g__ccool_aux_prop ...	26, 29, 46, 175
__ccool_aux_prop:N	44, 174
__ccool_aux_prop:n	40, 50
__ccool_aux_prop:nn	26
__ccool_aux_prop:w	32, 42
__ccool_aux_separ:n	76, 310
__ccool_aux_separ:nn	53, 78
__ccool_aux_timestamp ..	80, 334, 335
__ccool_aux_val:Nn	87, 157
\g__ccool_aux_val_seq ...	89, 90, 163
__ccool_erw_seq_use:Nn	302
__ccool_lambda:nn	92, 320
__ccool_lambda_expression ..	95, 98
__ccool_log_close:	100, 395
__ccool_log_entry	187, 188
\g__ccool_log_file_tl ..	108, 111, 333
\g__ccool_log_iow	100, 101, 105, 112, 130, 133
__ccool_log_open:	108, 394
\g__ccool_log_open_bool	102, 106, 113, 128, 182
__ccool_log_read:	121, 406
__ccool_log_read:n ...	115, 123, 405
\g__ccool_log_to_tl	111, 112, 123, 125
__ccool_log_write:n	125, 184
__ccool_make_ccool:nnnn	192, 339, 353, 367, 381
__ccool_make_ccool_exp:nnn	155, 203
__ccool_make_ccool_key:nnn	167, 181
__ccool_make_ccool_sideeffect:nnn	179, 200, 209
__ccool_make_key:N	151, 177
__ccool_make_key:n	146, 153
__ccool_make_key:Nn	136, 148
\g__ccool_option_expans_tl	32, 36, 330
__ccool_option_inner:n	225, 229, 337
\g__ccool_option_inner_tl	227, 231, 342, 356, 370, 384
__ccool_option_name:n	233, 351
\g__ccool_option_name_tl	140, 235, 241, 313, 341, 355, 369, 383
__ccool_option_outer:n ...	243, 365
\g__ccool_option_outer_tl	245, 249, 344, 358, 372, 386
__ccool_option_separ:n ...	251, 379
\g__ccool_option_separ_tl	253, 257, 343, 357, 371, 385
__ccool_prop_append:NN ...	259, 270
__ccool_prop_append:Nn ...	175, 268
__ccool_prop_append:nn ...	261, 265
__ccool_prop_clear_new:n ..	272, 279
__ccool_prop_clear_new_map:n ...	276, 315
__ccool_prop_if_exist:nTF ..	169, 281
__ccool_prop_item:nn	142, 285
__ccool_prop_name:n	90, 270, 274, 283, 287, 289, 292
__ccool_prop_new:n	171, 290
__ccool_seq_from_prop:n ...	296, 300
__ccool_seq_from_prop:NNn ..	90, 294
__ccool_seq_use:Nn	162, 306
\CcoolClear	3, 23, 312
\CcoolHook	3, 4, 23, 207, 317
\CcoolLambda	3, 24, 318
\CcoolOption	3, 24, 322
\CcoolRead	3, 12, 26, 401
\CcoolVers	3, 26, 408
cs commands:	
\cs_generate_variant:Nn	7, 31, 75, 120, 135, 145, 150, 267
\cs_gset:Npn	6, 24, 253
\cs_new:Nn ..	53, 76, 233, 281, 285, 306
\cs_new:Npn	289
\cs_new_protected:Nn	4, 13, 17, 22, 27, 40, 44, 87, 103, 109, 115, 121, 126, 136, 146, 151, 155, 167, 179, 225, 243, 251, 268, 272, 276, 290, 294
\cs_new_protected:Npn	9, 33, 92, 192, 259
\cs_set:Nn	261
\cs_set_protected:Nn	296
\csedef	81, 85

D		<code>\msg_new:nnn</code> 219, 220, 221, 222, 223, 224
<code>\DeclareDocumentCommand</code>	95, 194	<code>\msg_warning:nnn</code> 231, 249, 257
<code>\def</code>	187	
<code>\detokenize</code>	85	
E		N
<code>\end</code>	305	<code>\NeedsTeXFormat</code> 2
<code>\endgroup</code>	188	<code>\NewDocumentCommand</code> 312, 317, 322, 401, 408
<code>\ensuremath</code>	376, 377	
exp commands:		O
<code>\exp_args:NNf</code>	161	options:
<code>\exp_args:NNx</code>	94, 138, 194	<code>*</code> 5
<code>\exp_args:No</code>	172, 274	<code>+</code> 4
<code>\exp_args:Nx</code>	35	<code><code₁></code> 4
<code>\exp_last_unbraced:Nf</code>		<code><code₂></code> 5
.	338, 352, 366, 380	<code><kv₁></code> 4
<code>\exp_last_unbraced:NNf</code>	308	<code><tl₁></code> 4
<code>\exp_not:N</code>	71, 231, 241, 249, 257	<code><tl₂></code> 4
<code>\exp_not:n</code>	215, 333	<code><tl₃></code> 5
<code>\expandafter</code>	85, 187	<code><tl₄></code> 5
<code>\ExplSyntaxOff</code>	411	<code><tl₅></code> 5
<code>\ExplSyntaxOn</code>	3	<code><tl₆></code> 5
F		Expans 24
file commands:		File 24
<code>\file_input:n</code>	117	Inner 24
G		Name 25
<code>\gappto</code>	207	Outer 25
I		Separ 25
<code>\IfBooleanT</code>	201	Write 26
<code>\ifcsdef</code>	80	
<code>\IfValueT</code>	199, 213	P
<code>\IfValueTF</code>	404	<code>\pdfcreationdate</code> 81, 84, 85
int commands:		prg commands:
<code>\int_case:nnTF</code>	55	<code>\prg_replicate:nn</code> 58, 96
iow commands:		prop commands:
<code>\iow_close:N</code>	101, 105	<code>\prop_clear_new:N</code> 274
<code>\iow_new:N</code>	100	<code>\prop_gclear_new:N</code> 46
<code>\iow_now:Nn</code>	130	<code>\prop_gput:Nnn</code> 29, 263
<code>\iow_open:Nn</code>	112	<code>\prop_if_exist:NTF</code> 283
<code>\item</code>	303, 304	<code>\prop_item:Nn</code> 263, 287, 298
K		<code>\prop_map_function:NN</code> 265
keys commands:		<code>\prop_new:N</code> 26, 292
<code>\l_keys_choice_tl</code>	330	<code>\ProvideDocumentCommand</code> 139, 318
<code>\keys_define:nn</code>	327	
<code>\keys_set:nn</code>	325	Q
M		quark commands:
<code>\meta</code>	303, 304	<code>\q_stop</code> 9, 15, 33, 42
msg commands:		
<code>\msg_error:nnn</code>	133, 239	S
<code>\msg_error:nnnn</code>	69	seq commands:
N		<code>\seq_gclear_new:N</code> 19, 89
<code>\msg_new:nnn</code> 219, 220, 221, 222, 223, 224		<code>\seq_gput_right:Nn</code> 11, 298
<code>\msg_warning:nnn</code> 231, 249, 257		<code>\seq_if_empty:NTF</code> 47
O		<code>\seq_map_function:NN</code>
<code>\NeedsTeXFormat</code> 2	 20, 50, 153, 279, 300
<code>\NewDocumentCommand</code> 312, 317, 322, 401, 408		<code>\seq_set_from_clist:Nn</code> 173, 278
P		<code>\seq_use:Nnnn</code> 309
<code>\pdfcreationdate</code> 81, 84, 85		
prg commands:		
<code>\prg_replicate:nn</code> 58, 96		
prop commands:		
<code>\prop_clear_new:N</code> 274		
<code>\prop_gclear_new:N</code> 46		
<code>\prop_gput:Nnn</code> 29, 263		
<code>\prop_if_exist:NTF</code> 283		
<code>\prop_item:Nn</code> 263, 287, 298		
<code>\prop_map_function:NN</code> 265		
<code>\prop_new:N</code> 26, 292		
<code>\ProvideDocumentCommand</code> 139, 318		
Q		
quark commands:		
<code>\q_stop</code> 9, 15, 33, 42		
S		
seq commands:		
<code>\seq_gclear_new:N</code> 19, 89		
<code>\seq_gput_right:Nn</code> 11, 298		
<code>\seq_if_empty:NTF</code> 47		
<code>\seq_map_function:NN</code>		
. 20, 50, 153, 279, 300		
<code>\seq_set_from_clist:Nn</code> 173, 278		
<code>\seq_use:Nnnn</code> 309		

sys commands:		\tl_log:n	118, 131
\c_sys_jobname_str	334, 335	\tl_new:N	32, 108, 125
		\tl_trim_spaces:n	11, 37, 38
T			
\TeXOSQueryNow	84	U	
tl commands:		use commands:	
\c_empty_tl	48, 67, 170, 190, 212, 317	\use:N	36, 410
\tl_count:n	78	\use_i:nn	61, 63
\tl_gset:Nn	111, 227, 235, 245, 333	\use_ii:nn	62
\tl_gset_eq:NN	330	\usepackage	4, 83

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 { __ccool }
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_timestamp [7]
80 \ifcsdef{pdfcreationdate}
81 {\csedef{\__ccool_aux_timestamp}{\pdfcreationdate}}
82 {
83   \usepackage{texosquery}
84   \TeXOSQueryNow{\pdfcreationdate}
85   \csedef{\__ccool_aux_timestamp}{\detokenize\expandafter{\pdfcreationdate}}
86 }

```

(End definition for __ccool_aux_timestamp.)

```

\__ccool_aux_val:Nn #1: < seq >
#2: < tl var name >
87 \cs_new_protected:Nn \__ccool_aux_val:Nn
88 {
89   \seq_gclear_new:N \g__ccool_aux_val_seq
90   \__ccool_seq_from_prop:NNn \g__ccool_aux_val_seq #1 { \__ccool_prop_name:n{#2} }
91 }

```

(End definition for __ccool_aux_val:Nn.)

2 lambda

```

\__ccool_lambda:nn [8]
92 \cs_new_protected:Npn \__ccool_lambda:nn #1 #2
93 {
94   \exp_args:NNx

```

```

95 \DeclareDocumentCommand \_ccool_lambda_expression
96 { \prg_replicate:nn { #1 } { m } }
97 {#2}
98 \_ccool_lambda_expression
99 }

```

(End definition for _ccool_lambda:nn.)

3 log

_ccool_log_close:

```

100 \iow_new:N \g__ccool_log_iow
101 \AtEndDocument{\iow_close:N \g__ccool_log_iow}
102 \bool_set_false:N \g__ccool_log_open_bool
103 \cs_new_protected:Nn \_ccool_log_close:
104 {
105   \iow_close:N \g__ccool_log_iow
106   \bool_gset_false:N \g__ccool_log_open_bool
107 }

```

(End definition for _ccool_log_close:.)

_ccool_log_open:

```

108 \tl_new:N \g__ccool_log_file_tl
109 \cs_new_protected:Nn \_ccool_log_open:
110 {
111   \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
112   \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
113   \bool_gset_true:N \g__ccool_log_open_bool
114 }

```

(End definition for _ccool_log_open:.)

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

```

115 \cs_new_protected:Nn \_ccool_log_read:n
116 {
117   \file_input:n{#1}
118   \tl_log:n{read~from~#1}
119 }
120 \cs_generate_variant:Nn \_ccool_log_read:n { e }

```

(End definition for _ccool_log_read:n.)

_ccool_log_read:

```

121 \cs_new_protected:Nn \_ccool_log_read:
122 {
123   \_ccool_log_read:e{\g__ccool_log_to_tl}
124 }

```

(End definition for _ccool_log_read:.)

```

\__ccool_log_write:n
125 \tl_new:N \g__ccool_log_to_tl
126 \cs_new_protected:Nn \__ccool_log_write:n
127 {
128   \bool_if:nTF{ \g__ccool_log_open_bool }
129   {
130     \iow_now:Nn \g__ccool_log_iow {#1}
131     \tl_log:n{ write~to~#1 }
132   }
133   { \msg_error:nnn{ __ccool }{ iow }{ \g__ccool_log_iow } }
134 }
135 \cs_generate_variant:Nn \__ccool_log_write:n { e }

(End definition for \__ccool_log_write:n.)

```

4 make_key

```

\__ccool_make_key:Nn #1 : < token >
#2 : < key >
136 \cs_new_protected:Nn \__ccool_make_key:Nn
137 {
138   \exp_args:NNx
139   \ProvideDocumentCommand{#1}
140   { D<>{ \g__ccool_option_name_tl } }
141   {
142     \__ccool_prop_item:nn{##1}{#2}
143   }
144 }
145 \cs_generate_variant:Nn \__ccool_make_key:Nn {c}

(End definition for \__ccool_make_key:Nn.)

```

```

\__ccool_make_key:n #1 : < key >
146 \cs_new_protected:Nn \__ccool_make_key:n
147 {
148   \__ccool_make_key:cn{#1}{#1}
149 }
150 \cs_generate_variant:Nn \__ccool_make_key:n { e }

(End definition for \__ccool_make_key:n.)

```

```

\__ccool_make_key:N #1 : < seq >
151 \cs_new_protected:Nn \__ccool_make_key:N
152 {
153   \seq_map_function:NN #1 \__ccool_make_key:e
154 }

(End definition for \__ccool_make_key:N.)

```

5 make_ccool

_ccool_make_ccool_exp:nnn

```

155 \cs_new_protected:Nn \_ccool_make_ccool_exp:nnn
156 {
157   \_ccool_aux_val:Nn \g\_ccool_aux_key_seq {#1}
158   \_ccool_aux_outer_set:n{#3}
159   \_ccool_aux_outer:n
160   {
161     \exp_args:NNf
162     \_ccool_seq_use:Nn
163     \g\_ccool_aux_val_seq
164     {#2}
165   }
166 }
```

(End definition for _ccool_make_ccool_exp:nnn.)

_ccool_make_ccool_key:nnn

```

167 \cs_new_protected:Nn \_ccool_make_ccool_key:nnn
168 {
169   \_ccool_prop_if_exist:nTF{#1}
170   { \c_empty_tl }
171   { \_ccool_prop_new:n{#1} }
172   \exp_args:No \_ccool_aux_inner_set:n{#2}
173   \seq_set_from_clist:Nn \g\_ccool_aux_keyval_seq {#3}
174   \_ccool_aux_prop:N \g\_ccool_aux_keyval_seq
175   \_ccool_prop_append:Nn \g\_ccool_aux_prop {#1}
176   \_ccool_aux_key:N \g\_ccool_aux_keyval_seq
177   \_ccool_make_key:N \g\_ccool_aux_key_seq
178 }
```

(End definition for _ccool_make_ccool_key:nnn.)

_ccool_make_ccool_sideeffect:nnn [10]

```

179 \cs_new_protected:Nn \_ccool_make_ccool_sideeffect:nnn
180 {
181   \_ccool_make_ccool_key:nnn{#1}{#2}{#3}
182   \bool_if:nTF{ \g\_ccool_log_open_bool }
183   {
184     \_ccool_log_write:n
185     {
186       \begingroup
187       \def \_ccool_log_entry { \Ccool<#1>i{#2}{#3} } \expandafter
188       \endgroup \_ccool_log_entry
189     }
190   }{\c_empty_tl}
191 }
```

(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 >*

```
192 \cs_new_protected:Npn \__ccool_make_ccool:nnnn #1 #2 #3 #4
193 {
194   \exp_args:NNx \DeclareDocumentCommand \Ccool
195     {%^^A      2      3          4 5 6      7 8          9
196     D<>{#1} +o E{ i }{#{2}} m t+ s E{ s o }{#{3}{#4}} +o
197   }
198   {
199     \IfValueT{##2}{##2}
200     \__ccool_make_ccool_sideeffect:nnn{##1}{##3}{##4}
201     \IfBooleanT{##6}
202     {
203       \__ccool_make_ccool_exp:nnn{##1}{##7}{##8}
204     }
205     \bool_if:nTF{##5}
206     {
207       \gappto{\CcoolHook}
208       {
209         \__ccool_make_ccool_sideeffect:nnn{##1}{##3}{##4}
210       }
211     }
212     {\c_empty_tl}
213     \IfValueT{##9}
214     {
215       \exp_not:n{ \Ccool<##1>[##9] }
216     }
217   }
218 }
```

(End definition for __ccool_make_ccool:nnnn.)

6 msg

```
219 \msg_new:nnn {__ccool}{ generic }{#1}
220 \msg_new:nnn {__ccool}{ iow }{#1~is~closed~can't~write}
221 \msg_new:nnn {__ccool}{ keyonly }{#1~does~not~take~values;~keyval~is~#2}
222 \msg_new:nnn {__ccool}{ keywrong }{#1~does~not~recognize~key~#2}
223 \msg_new:nnn {__ccool}{ separ }{#1~expects~1~to~3~items,~#2}
224 \msg_new:nnn {__ccool}{ unset }{#1~unset}
```

7 option

__ccool_aux_inner:n #1 : *< code >*

```
225 \cs_new_protected:Nn \__ccool_option_inner:n
226 {
227   \tl_gset:Nn \g__ccool_option_inner_tl {#1}
228 }
229 \__ccool_option_inner:n
230 {
231   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_inner_tl }
232 }
```

(End definition for __ccool_aux_inner:n.)


```

\__ccool_option_name:n #1:  $\langle token list \rangle$ 

233 \cs_new:Nn \__ccool_option_name:n
234 {
235   \tl_gset:Nn \g__ccool_option_name_tl{#1}
236 }
237 \__ccool_option_name:n
238 {
239   \msg_error:nnx{ __ccool }
240   { generic }
241   { \exp_not:N\g__ccool_option_name_tl~undefined }
242 }

(End definition for \__ccool_option_name:n.)

\__ccool_option_outer:n #1:  $\langle inline code \rangle$ 

243 \cs_new_protected:Nn \__ccool_option_outer:n
244 {
245   \tl_gset:Nn \g__ccool_option_outer_tl {#1}
246 }
247 \__ccool_option_outer:n
248 {
249   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_outer_tl }
250 }

(End definition for \__ccool_option_outer:n.)

\__ccool_option_separ:n #1: { $\langle tl_1 \rangle$ }{ $\langle tl_2 \rangle$ }{ $\langle tl_3 \rangle$ }

251 \cs_new_protected:Nn \__ccool_option_separ:n
252 {
253   \cs_gset:Npn \g__ccool_option_separ_tl {#1}
254 }
255 \__ccool_option_separ:n
256 {
257   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_separ_tl }
258 }

(End definition for \__ccool_option_separ:n.)

```

8 prop

```

\__ccool_prop_append:NN #1:  $\langle prop_1 \rangle$ 
#2:  $\langle prop_2 \rangle$ 

259 \cs_new_protected:Npn \__ccool_prop_append:NN #1 #2
260 {
261   \cs_set:Nn \__ccool_prop_append:nn
262   {
263     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
264   }
265   \prop_map_function:NN #2 \__ccool_prop_append:nn
266 }
267 \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 >

268 \cs_new_protected:Nn \__ccool_prop_append:Nn
269 {
270   \__ccool_prop_append:cn{ \__ccool_prop_name:n {#2} } #1
271 }

(End definition for \__ccool_prop_append:Nn.)

\__ccool_prop_clear_new:n #1 : < tl var name >

272 \cs_new_protected:Nn \__ccool_prop_clear_new:n
273 {
274   \exp_args:No \prop_clear_new:c{ \__ccool_prop_name:n {#1} }
275 }

(End definition for \__ccool_prop_clear_new:n.)

\__ccool_prop_clear_new_map:n #1 : < keyval list >

276 \cs_new_protected:Nn \__ccool_prop_clear_new_map:n
277 {
278   \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
279   \seq_map_function:NN \g__ccool_aux_key_seq \__ccool_prop_clear_new:n
280 }

(End definition for \__ccool_prop_clear_new_map:n.)

\__ccool_prop_if_exist:nTF #1 : < tl1 >
#2 : < tl2 >
#3 : < tl3 >

281 \cs_new:Nn \__ccool_prop_if_exist:nTF
282 {
283   \prop_if_exist:CTF{ \__ccool_prop_name:n {#1} }{#2}{#3}
284 }

(End definition for \__ccool_prop_if_exist:nTF.)

\__ccool_prop_item:nn #1 : < tl var name >
#2 : < key >

285 \cs_new:Nn \__ccool_prop_item:nn
286 {
287   \prop_item:cn { \__ccool_prop_name:n {#1} } {#2}
288 }

(End definition for \__ccool_prop_item:nn.)

\__ccool_prop_name:n #1 : < tl var name >

289 \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 >

290 \cs_new_protected:Nn \__ccool_prop_new:n
291 {
292   \prop_new:c{ \__ccool_prop_name:n {#1} }
293 }

(End definition for \__ccool_prop_new:n.)

```

9 seq

```

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

294 \cs_new_protected:Nn \__ccool_seq_from_prop:NNn
295 {
296   \cs_set_protected:Nn \__ccool_seq_from_prop:n
297   {
298     \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
299   }
300   \seq_map_function:NN #2 \__ccool_seq_from_prop:n
301 }

(End definition for \__ccool_seq_from_prop:NNn.)

```

```

\__ccool_erw_seq_use:Nn

302 % \begin{arguments}
303 % \item \meta{ seq }
304 % \item \meta{ tokens }
305 % \end{arguments}
306 \cs_new:Nn \__ccool_seq_use:Nn
307 {
308   \exp_last_unbraced:NNf
309   \seq_use:Nnnn #1
310   \__ccool_aux_separ:n{#2}
311 }

(End definition for \__ccool_erw_seq_use:Nn.)

```

10 Front-end

\CcoolClear #1 : <token list>

Semantics Clears any data created by **\Ccool**{<token list>}

```

312 \NewDocumentCommand{ \CcoolClear }
313 { D<>{\g__ccool_option_name_tl} }
314 {
315   \__ccool_prop_clear_new_map:n{#1}
316 }

```

\CcoolHook

Example \AfterEndEnvironment{theorem}{\CcoolHook}

```

317 \NewDocumentCommand{\CcoolHook}{\c_empty_tl}

```

\CcoolLambda

#1 : $\langle integer \rangle$
#2 : $\langle code \rangle$

Example \Ccool{ EvalAt = \CcoolLambda{(#1)} }

Semantics Creates a lambda expression with $\langle integer \rangle$ arguments for $\langle code \rangle$

```
318 \ProvideDocumentCommand \CcoolLambda { 0{1} m }
319 {
320   \__ccool_lambda:nn { #1 } { #2 }
321 }
```

\CcoolOption

#1 : $\langle keyval list \rangle$

```
322 \NewDocumentCommand{ \CcoolOption }
323 { m }
324 {
325   \keys_set:nn{ __ccool }{#1}
326 }
327 \keys_define:nn { __ccool }
328 {
```

Expans

Value eo|ee|ex|x0|xe|xx

```
329 Expans .multichoices:nn = { eo, ee, ex, xo, xe, xx }
330 { \tl_gset_eq:NN \g__ccool_option_expans_tl \l_keys_choice_tl },
331 Expans .default:n = { xo },
332 Expans .initial:n = { xo },
```

File

Value $\langle path \rangle$

```
333 File .code:n = { \tl_gset:Nn \g__ccool_log_file_tl{ \exp_not:n{ #1 } } },
334 File .default:n = { \c_sys_jobname_str\__ccool_aux_timestamp },
335 File .initial:n = { \c_sys_jobname_str\__ccool_aux_timestamp },
```

Inner

Value $\langle code \rangle$, with **###1** as the argument to be replaced

```
336 Inner .code:n={
337   \__ccool_option_inner: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 Inner .value_required:n = false,
348 Inner .default:n = {####1},
349 Inner .initial:n = {####1},

```

Name

Value *(token list)*

```

350 Name .code:n={
351   \__ccool_option_name:n{#1}
352   \exp_last_unbraced:Nf
353   \__ccool_make_ccool:nnnn
354   {
355     { \g__ccool_option_name_tl }
356     { \g__ccool_option_inner_tl }
357     { \g__ccool_option_separ_tl }
358     { \g__ccool_option_outer_tl }
359   }
360 },
361 Name .value_required:n = false,
362 Name .default:n = { Math },
363 Name .initial:n = { Math },

```

Outer

Value *(code)*, with ####1 as the argument to be replaced

```

364 Outer .code:n={
365   \__ccool_option_outer:n{#1}
366   \exp_last_unbraced:Nf
367   \__ccool_make_ccool:nnnn
368   {
369     { \g__ccool_option_name_tl }
370     { \g__ccool_option_inner_tl }
371     { \g__ccool_option_separ_tl }
372     { \g__ccool_option_outer_tl }
373   }
374 },
375 Outer .value_required:n = false,
376 Outer .default:n = { \ensuremath{####1} },
377 Outer .initial:n = { \ensuremath{####1} },

```

Separ

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

```

378 Separ .code:n={
379   \__ccool_option_separ:n{#1}
380   \exp_last_unbraced:Nf
381   \__ccool_make_ccool:nnnn
382   {
383     { \g__ccool_option_name_tl }
384     { \g__ccool_option_inner_tl }
385     { \g__ccool_option_separ_tl }
386     { \g__ccool_option_outer_tl }
387   }

```

```

388 },
389 Separ .value_required:n = false,
390 Separ .default:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
391 Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },

```

Write

Value $\langle boolean \rangle$

```

392 Write .code:n = {
393   \bool_if:nTF{#1}
394   {\__ccool_log_open:}
395   {\__ccool_log_close:}
396 },
397 Write .value_required:n = false,
398 Write .default:n = \BooleanFalse,
399 Write .initial:n = \BooleanFalse
400 }

```

\CcoolRead

#1 : $\langle path \rangle$

Semantics

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

```

401 \NewDocumentCommand{\CcoolRead}
402 {o}
403 {
404   \IfValueTF{#1}
405   {\__ccool_log_read:e{#1}}
406   {\__ccool_log_read:}
407 }

```

\CcoolVers

Semantics Expands to the package’s version

```

408 \NewDocumentCommand{\CcoolVers}
409 {}
410 {\use:c{ver@ccool.sty}}

```

15 Misc

```

411 \ExplSyntaxOff

```