

The ccool package^{*}

Erwann Rogard[†]

Released 2020/04/17

Abstract

ccool stands for Custom COntent Oriented for L^AT_EX, that is “ give commands the ability to contain the mathematical meaning while retaining the typesetting versatility”, a concept pioneered by cool[1]. Here, the commands are not predefined, instead they are created (‘custom’) using a minimalist interface built upon xparse[4]. Specifically, `\Ccool` takes as mandatory argument a *keyval list*, the lhs of each element of which encodes a concept, and the rhs a typesetting instruction (for instance, the side effect of `{ Real = \mathbb{R} }` is that `\Real` expands to \mathbb{R}). A *keyval list* can optionally be expanded in place (append `*`), according to default or inline rules, and interspersed with optional arguments that are destined exclusively for expansion (`[Let~]` and `[~denote real numbers.]`). Thus, in theory, a document could be typeset, starting with a single `\Ccool` declaration. An optional parameter prepended to the *keyval list*, `<param>`, allows to parameterize the keys (for instance, one parameter value for the style, another for a property). In conjunction with lambda expressions, this tool allows for encoding the way complex mathematical objects are formatted (for instance, `functions and operators` having `(.)` and `[.]`, respectively, around their arguments). 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>	5
2.3	<code>i{<code₁>}</code>	5
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

^{*}This file describes version v2.1, last revised 2020/04/17.

[†]firstname dot lastname AusTria gmail dot com

2.8	<code>o{\code_2}</code>	5
2.9	<code>[{tl_6}]</code>	5
3	Other	5
4	Do's and dont's	6
II	Listing	7
1.	Preamble	7
2.	<code>\CcoolVers</code>	7
3.	Let \mathbb{N} and $\mathbb{R} \dots$ Plain	7
4.	Let \mathbb{N} and $\mathbb{R} \dots$ Equivalent to 3 with <code>\NewDocumentCommand</code>	7
5.	Let \mathbb{N} and $\mathbb{R} \dots$ Equivalent to 4 with <code>\Ccool</code>	7
6.	Let \mathbb{N} and $\mathbb{R} \dots$ Equivalent to 5 with <code>i{\code_1}</code>	8
7.	Let \mathbb{N} and $\mathbb{R} \dots$ Equivalent to 6 with <code>s{\tl_3}</code>	8
8.	Let \mathbb{N} and $\mathbb{R} \dots$ Extends 7 with descriptive keys, with <code><\tl_2></code>	8
9.	Let \mathbb{N} and $\mathbb{R} \dots$ Equivalent to 8 with <code>s{\tl_3}</code>	8
10.	Separators.	8
11.	Hello, world! (test case)	9
12.	Listing 11 read from file	9
13.	Probability space.	10
14.	Listing 13 read from file	10
15.	Mittelwertsatz für n Variable.	10
16.	Listing 15 read from file	11
17.	Fonction et fonctionelle	11
18.	Listing 17 read from file	11
19.	CUSUM statistic.	11
20.	Listing 19 read from file	12
III	Other	13
1	Acknowledgment	13

2	Install	13
3	Issue	13
4	Support	13
5	Testing	13
5.1	Technicality	13
5.2	Platform	13
5.3	Engine	14
5.4	Results	14
5.5	Other	14
	References	14
	Change History	15
	Index	16
IV	Implementation	19
1	Opening	19
2	aux	19
3	lambda	21
4	log	22
5	make_key	23
6	make_ccool	23
7	msg	25
8	option	25
9	prop	26
10	seq	27
11	sys	28

12	Front-end	29
12.1	<code>\CcoolClear</code>	29
12.2	<code>\CcoolHook</code>	29
12.3	<code>\CcoolLambda</code>	29
12.4	<code>\CcoolOption</code>	29
12.4.1	Expans	30
12.4.2	File	30
12.4.3	Inner	30
12.4.4	Param	30
12.4.5	Outer	31
12.4.6	Separ	31
12.4.7	Write	31
12.5	<code>\CcoolRead</code>	32
12.6	<code>\CcoolVers</code>	32
13	Closing	32

Part I

Usage

Convention

1. Loosely, those of [2] and [4], 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>
--------------------------	---------------------------------

Requirement

1. `ccool.sty` is in the path of the L^AT_EX engine. See [Part III, section 4](#).
2. Declare it in the *preamble*

<code>\Ccool</code>	<code>\Ccool</code> $[\langle t_1 \rangle]$ $\langle \langle t_2 \rangle \rangle$ $i\{\langle code_1 \rangle\}$ $\{\langle kv_1 \rangle\}$ $+$ $*$ $s\{\{\langle t_3 \rangle\} \{\langle t_3 \rangle\}\{\langle t_4 \rangle\} \{\langle t_3 \rangle\}\{\langle t_4 \rangle\}\{\langle t_5 \rangle\}\}$ $o\{\langle code_2 \rangle\}$ $[\langle t_6 \rangle]$
---------------------	---

Requirement $\langle kv_1 \rangle$ is specified (all others optional).

$\langle t_1 \rangle$

Example Let~

Semantics Expands $\langle tl_1 \rangle$

$\langle tl_2 \rangle$

Default **Param**'s

Example Default, Style and Describe, ModelA and ModelB

$\langle code_1 \rangle$

Default **Inner**'s

Example $\backslash\mathrm{mathbb{#1}}$

$\langle kvl_1 \rangle$

Example Real= $\{\backslash\mathrm{mathbb{R}}\}$

Semantics

- 1) $\langle val_i \rangle \leftarrow \langle code_1 \rangle$ applied to $\langle val_i \rangle$
- 2) $\backslash\langle key_i \rangle \langle tl_2 \rangle \leftarrow \langle val_i \rangle$ defined in step 1), using **Expans** for expansion.
- 3) If **Write**, writes the input used by step 2) to **File**

+

Semantics Repeats step 1), step 2), and step 3), at **\CcoolHook** (useful inside a *local group*)

*

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

$\langle tl_3 \rangle$

Default **Separ**'s

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

$\langle tl_4 \rangle$

Default **Separ**'s

Example $\{,\sim\}$

$\langle tl_5 \rangle$

Default **Separ**'s

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

$\langle code_2 \rangle$

Default **Outer**'s

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

$\langle tl_6 \rangle$

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

Other

Continued in [Part IV, section 12](#).

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: `\cal F`.

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

5)

Don't: `\[x_0,x\]`

Do: `\left[x_0,x\right]`

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

Part II

Listing

Listing 1 are settings to replicate the listings. For exhaustivity, check the `documentation` section of `ccool.dtx`.

Listing 2 is self explanatory.

Listing 3-9 is a tutorial comprising different ways to typeset “Let \mathbb{N} and \mathbb{R} ...” The plain version is prone to errors, should the author change `\mathbb{R}` to, say, `\mathcal{R}` throughout the document. Listing 4 avoids that by separating style from meaning. Listing 5-7 achieve greater compactness (DRY and expand in place). Listing 8 extends Listing 5 with a description for each key. Listing 9 is to Listing 8, what Listing 7 is Listing 6.

Listing 10 shows the full range of uses of `separators' parameter`.

Listing 11 and Listing 12 are a contrived “Hello, world!” test case.

Listing 13, Listing 15, and Listing 19 typeset realistic mathematical text, and write it to a file. Listing 14, Listing 16, and Listing 20, read the corresponding files.

Listing 1. Preamble

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

Listing 2. \CcoolVers

```
% \CcoolVers
%
```

2020/04/17 v2.1 cool — A tool for encoding notational conventions (esp. Math)

Listing 3. Let \mathbb{N} and \mathbb{R} ... Plain

```
% Let~$\mathbb{N}$ and $\mathbb{R}$ denote the natural and real
% numbers.
%
```

Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.

Listing 4. Let \mathbb{N} and \mathbb{R} ... Equivalent to 3 with `\NewDocumentCommand`

```
% \NewDocumentCommand\Nat{}\{\mathbb{N}\}
% \NewDocumentCommand\Real{}\{\mathbb{R}\}
% Let~$\Nat$ and $\Real$ denote the natural and real numbers.
%
```

Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.

Listing 5. Let \mathbb{N} and \mathbb{R} ...Equivalent to 4 with `\Ccool`

```
% \Ccool { Nat = {\mathbb{N}}, Real = {\mathbb{R}} }
% Let~$Nat$ and $Real$~denote the natural and real numbers.
%
```

Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.

Listing 6. Let \mathbb{N} and \mathbb{R} ...Equivalent to 5 but with `i{\code_1}`

```
% \Ccool i{\mathbb{#1}}{ Nat = {N}, Real = {R} }
% Let~$Nat$ and $Real$~denote the natural and real numbers.
%
```

Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.

Listing 7. Let \mathbb{N} and \mathbb{R} ...Equivalent to 6 with `s{\langle tl_3 \rangle}`

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

Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.

Listing 8. Let \mathbb{N} and \mathbb{R} ...Equivalent to 7 with `<\tl_2>`

```
% \Ccool{ Nat = {\mathbb{N}}, Real = {\mathbb{R}} }[]
% <Describe>{ Nat = {natural numbers}, Real = {the real line} }
% [Let $Nat$ and $Real$ denote
% the~\Nat<Describe>~and~\Real<Describe>.]{}
%
```

Let \mathbb{N} and \mathbb{R} denote the natural numbers and the real line.

Listing 9. Let \mathbb{N} and \mathbb{R} ...Equivalent to 8 with `s{\langle tl_3 \rangle}`

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

Let \mathbb{N} and \mathbb{R} denote the natural numbers and the real line.


```
% \CcoolOption{
% ^^A% spaces betw. inner and outer brackets matter!->
% Separ={{\ \char`@ }{\ \% }{\ \char`@ }}}
% \Ccool{ 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{{\ \% }{, \ }}{, \ \% }{, \ }}[\]
%
```

Listing 11. Hello, world! (test case)

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

Listing 12. Listing 11 read from file

9

```
% \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 13. 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 14. Listing 13 read from file

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

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

Listing 15. 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 = { \left[x_0,x\right] } }+
% [\$n\in N$,~\$OffMenge\subseteq N^n$ eine offene Menge und
% \$f\in Ci(\OffMenge,\R)$].
% Dann gibt es auf jeder Strecke \$Strecke\subset OffMenge$ 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 16. Listing 15 read from file

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

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

Listing 17. Fonction et fonctionnelle

```
%      \CcoolOption{ Write = \BooleanTrue }
%      \Ccool{ EvalAt = \CcoolLambda{(#1)}, ApplyOp =
%      \CcoolLambda[mm]{#1[#2]} }
%      [Supposons une fonction $f\text{EvalAt}\{t\}$, et \etudions le probl\eme
%      o\`u la fonctionnelle $\text{ApplyOp}\{S\}\{f\}$ 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 18. Listing 17 read from file

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

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

Listing 19. CUSUM statistic [4]

```
% \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{\sim\in~}}[~and~]
% { CUSUMthresh = { \nu } } **o{\$#1\in\Real^{+}}$.}
% [~Define the following processes:]
% { LogWald = { u }, CUSUMst = { \StopT_{c} }, CUSUM = { y },
% LogWaldInf = { m } }+
% [\begin{enumerate}
% \item{\$LogWald_{\Time}\EvalAt{ \Scale } = \Scale\Process_{\Time}
% - \frac{1}{2}\Scale^2\Time$;
% \$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 \SuchThat \CUSUM_{\Time}\EvalAt{\Scale} \ge \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 20. Listing 19 read from file

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

$t \xi \lambda \tilde{\mathcal{R}} \nu u \tilde{T}_c y m$

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

- 1) Compile `ccool.dtx` (under Unix, `$tex ccool.dtx`)
- 2) Put the generated `ccool.sty` in the search path of the L^AT_EXengine

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[8]. 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)
- d)* XeTeX 3.14159265-2.6-0.999992 (TeX Live 2020)

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)*
- 4) ccool v2.0 satisfactory on platform *i)* and engines *b)*, *c)*, and *d)*
- 5) ccool v2.1 satisfactory on platform *i)* and engines *b)*, *c)*, and *d)*

5.5 Other

Check [5] for testing ccool with llnx

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] @sean-allred’s answer to “How to create lambda expressions?”, <https://tex.stackexchange.com/a/188053/112708>
- [8] @joseph-wright’s answer to “Checking a function’s expansion against a string”, <https://tex.stackexchange.com/a/534100>
- [9] @frougon’s answer to “Journaling calls to a function []”, <https://tex.stackexchange.com/a/536620>

Change History

v1.0		v1.5	
General: Initial version	14	General: Added: <code>\File</code>	14
v1.1		Deleted: dependence on <code>\datetime</code>	14
General: Added: <code>\Save</code>	14	v1.6	
Added: Listing 1., 2., 3., 4., 6., and 9.	14	General: Added: Listing 1 (preamble)	14
Added: <code>\OpsRestore</code>	14	Renamed: <code>\OpsClear</code> to <code>\CcoolClear</code>	14
Added: <code>\OpsTest</code>	14	Renamed: <code>\OpsDebug</code> to <code>\CcoolDebug</code>	14
Deleted: Listing 1-5 from v1.0	14	Renamed: <code>\OpsHook</code> to <code>\CcoolHook</code>	14
Fixed: apparent anomaly in v1.0's Listing 4, see Listing 11	14	Renamed: <code>\OpsOption</code> to <code>\CcoolOption</code>	14
Replaced: <code>\OpsOptions</code> by <code>\OpsOption</code>	14	Renamed: <code>\OpsRead</code> to <code>\CcoolRead</code>	14
Replaced: <code>\{<klv_2>\}</code> by <code><klv_2></code> given that option type <code>G</code> not recommended[4]	14	Renamed: <code>\Ops</code> to <code>\Ccool</code>	14
Replaced: <code>\GenericObject</code> by <code>\Name</code>	14	Renamed: <code>oops</code> to <code>ccool</code> (better describes the purpose)	14
Replaced: <code>\Separators</code> by <code>\Separ</code>	14	v1.7	
Revamped: much of the implementation	14	General: Added: Legends to listings	14
v1.2		Added: Listing 19 (CUSUM)	14
General: Added: optional <code>*to</code>		Deleted: <code>\CcoolDebug</code>	14
<code>\OpsNew</code> as instruction to expand <code>klv_1</code>	14	Deleted: Listing 5 from v1.6	14
Deleted: <code>\OpsTest</code>	14	v1.8	
Deleted: <code><klv_2></code> and <code><code_2></code>	14	General: Added: <code>\CcoolVers</code>	14
Deleted: Listing 2-3 from v1.1.	14	Added: <code>\CcoolLambda</code>	14
Replaced: <code>\OpsClear{<tl_2>}</code> by <code>\OpsClear[<keyval list>]</code>	14	Added: Listing 17, Listing 18	14
Replaced: <code>\Restore</code> by <code>\Read</code>	14	Added: Listing 2	14
Replaced: <code>\Save</code> by <code>\Write</code>	14	v1.9	
v1.3		General: Added: support for LuaTeX	14
General: Replaced: <code>\OpsNew</code> by <code>\Ops</code>	14	Moved: from Part I to Part IV, what is now that part's section 12	14
Replaced: <code>\{<tl_2>\}</code> and <code>[<tl_2>]</code> by <code><tl_2></code>	14	v2.0	
v1.4		General: Added: support for XeTeX	14
General: Added: section 4	14	Deleted: <code>\File</code> 's dependency on <code>\texosquery</code> and <code>\pdfcreationdate</code>	14
Added: <code>\OpsDebug</code>	14	Updated: <code>\RequirePackage</code> , <code>\NeedsTeXFormat</code> 's second argument / TeX Live 2020	14
Added: <code>\OpsHook</code>	14	v2.1	
Added: <code>\Expans</code> (for debugging' sake, but...)	14	General: Added: Listing 3, Listing 4, Listing 5, Listing 6, Listing 6, Listing 8, and Listing 9 (tutorial)	14
Added: Listing 1., 2., and 3.	14	Replaced: <code>\CcoolLambda</code> 's optional integer argument (number of <code>m</code> 's) by a standard argument list	14
Added: optional <code>+to</code> <code>\OpsNew</code> to make side effects persist beyond local group	14	Replaced: <code><tl_2></code> 's position within <code>\Ccool</code> 's argument list, from first to second. Greater versatility	14
Deleted: Listing 1., and 2.	14	Replaced: <code>\Name</code> by <code>\Param</code>	14
Replaced: <code>\s{\{<tl_3>\}\{<tl_4>\}\{<tl_5>\}}</code> by <code>\s{\{<tl_3>\}\{<tl_3>\}\{<tl_4>\}\{<tl_3>\}\{<tl_4>\}\{<tl_5>\}}</code>	14		

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)	5
\<key _i >	5, 6
<code ₁ > (option)	4
<code ₂ > (option)	5
<kv1 ₁ > (option)	5
<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)	26
File (option)	26
Inner (option)	26
Outer (option)	27
Param (option)	26
Separ (option)	27
Write (option)	27
_	411, 412
A	
\AfterEndEnvironment	25
\AtEndDocument	93
B	
\begin	294
\begingroup	178
bool commands:	
\bool_gset_false:N	98
\bool_gset_true:N	105
\bool_if:nTF	120, 174, 197, 414
\bool_set_false:N	94
\BooleanFalse	419, 420
C	
\Ccool	1, 2, 4, 5, 8, 25, 25, 179, 186, 207
ccool internal commands:	
_ccool_aux_inner:n	5, 6, 37
_ccool_aux_inner_set:n	3, 164
_ccool_aux_key:N	16, 168
_ccool_aux_key:n	12, 19
_ccool_aux_key:w	8, 14
\g_ccool_aux_key_seq	10, 18, 149, 169, 270, 271
\g_ccool_aux_keyval_seq	165, 166, 168
_ccool_aux_outer:n	23, 151
_ccool_aux_outer_set:n	21, 150
\g_ccool_aux_prop	25, 28, 45, 167
_ccool_aux_prop:N	43, 166
_ccool_aux_prop:n	39, 49
_ccool_aux_prop:nn	25
_ccool_aux_prop:w	31, 41
_ccool_aux_separ:n	75, 302
_ccool_aux_separ:nn	52, 77
_ccool_aux_val:Nn	79, 149
\g_ccool_aux_val_seq	81, 82, 155
_ccool_erw_seq_use:Nn	294
_ccool_lambda:nn	84, 339
_ccool_lambda_expression	87, 90
_ccool_log_close:	92, 416
_ccool_log_entry	179, 180
\g_ccool_log_file_tl	100, 103, 353
\g_ccool_log_iow	92, 93, 97, 104, 122, 125
_ccool_log_open:	100, 415
\g_ccool_log_open_bool	94, 98, 105, 120, 174
_ccool_log_read:	113, 427
_ccool_log_read:n	107, 115, 426
\g_ccool_log_to_tl	103, 104, 115, 117
_ccool_log_write:n	117, 176
_ccool_make_ccool:nnnn	184, 360, 374, 388, 402
_ccool_make_ccool_exp:nnn	147, 195
_ccool_make_ccool_key:nnn	159, 173
_ccool_make_ccool_sideeffect:nnn	171, 192, 201
_ccool_make_key:N	143, 169
_ccool_make_key:n	138, 145
_ccool_make_key:Nn	128, 140
\g_ccool_option_expans_tl	31, 35, 349
_ccool_option_inner:n	217, 358
\g_ccool_option_inner_tl	219, 223, 363, 377, 391, 405
_ccool_option_outer:n	235, 386

<code>\g__ccool_option_outer_tl</code>	<code>\endgroup</code> 180
. 237, 241, 365, 379, 393, 407	<code>\ensuremath</code> 397, 398
<code>__ccool_option_param:n</code> 225, 372	exp commands:
<code>\g__ccool_option_param_tl</code>	<code>\exp_args:NNf</code> 153
132, 227, 233, 332, 362, 376, 390, 404	<code>\exp_args:NNx</code> 86, 130, 186
<code>__ccool_option_separ:n</code> 243, 400	<code>\exp_args:No</code> 164, 266
<code>\g__ccool_option_separ_tl</code>	<code>\exp_args:Nx</code> 34
. 245, 249, 364, 378, 392, 406	<code>\exp_last_unbraced:Nf</code>
<code>__ccool_prop_append:NN</code> 251, 262 359, 373, 387, 401
<code>__ccool_prop_append:Nn</code> 167, 260	<code>\exp_last_unbraced:NNf</code> 300
<code>__ccool_prop_append:nn</code> 253, 257	<code>\exp_not:N</code> 70, 223, 233, 241, 249
<code>__ccool_prop_clear_new:n</code> 264, 271	<code>\exp_not:n</code> 207
<code>__ccool_prop_clear_new_map:n</code>	<code>\expandafter</code> 179
. 268, 334	<code>\ExplSyntaxOff</code> 432
<code>__ccool_prop_if_exist:nTF</code> 161, 273	<code>\ExplSyntaxOn</code> 2
<code>__ccool_prop_item:nn</code> 134, 277	
<code>__ccool_prop_name:n</code>	F
. 82, 262, 266, 275, 279, 281, 284	file commands:
<code>__ccool_prop_new:n</code> 163, 282	<code>\file_input:n</code> 109
<code>__ccool_seq_from_prop:n</code> 288, 292	
<code>__ccool_seq_from_prop:NNn</code> 82, 286	G
<code>__ccool_seq_use:Nn</code> 154, 298	<code>\gappto</code> 199
<code>__ccool_sys_date:</code> 304, 314	
<code>__ccool_sys_date_hex:</code> 313, 328	I
<code>__ccool_sys_filename:</code> 325, 355, 356	<code>\IfBooleanT</code> 193
<code>__ccool_sys_time:</code> 315, 324	<code>\IfValueT</code> 191, 205
<code>__ccool_sys_time_hex:</code> 323, 329	<code>\IfValueTF</code> 425
<code>\CcoolClear</code> 3, 25, 331	int commands:
<code>\CcoolHook</code> 3, 5, 25, 199, 336	<code>\int_case:nnTF</code> 54
<code>\CcoolLambda</code> 3, 25, 337	<code>\int_eval:n</code> 306, 317
<code>\CcoolOption</code> 3, 26, 341	<code>\int_to_hex:n</code> 314, 324
<code>\CcoolRead</code> 3, 13, 28, 422	iow commands:
<code>\CcoolVers</code> 2, 3, 7, 28, 429	<code>\iow_close:N</code> 93, 97
cs commands:	<code>\iow_new:N</code> 92
<code>\cs_generate_variant:Nn</code>	<code>\iow_now:Nn</code> 122
. 6, 30, 74, 112, 127, 137, 142, 259	<code>\iow_open:Nn</code> 104
<code>\cs_gset:Npn</code> 5, 23, 245	<code>\item</code> 295, 296
<code>\cs_new:Nn</code> 52, 75, 225,	
273, 277, 298, 304, 313, 315, 323, 325	K
<code>\cs_new:Npn</code> 281	keys commands:
<code>\cs_new_protected:Nn</code> 3, 12, 16,	<code>\l_keys_choice_tl</code> 349
21, 26, 39, 43, 79, 95, 101, 107, 113,	<code>\keys_define:nn</code> 346
118, 128, 138, 143, 147, 159, 171,	<code>\keys_set:nn</code> 344
217, 235, 243, 260, 264, 268, 282, 286	
<code>\cs_new_protected:Npn</code>	M
. 8, 32, 84, 184, 251	<code>\meta</code> 295, 296
<code>\cs_set:Nn</code> 253	msg commands:
<code>\cs_set_protected:Nn</code> 288	<code>\msg_error:nnn</code> 125, 231
	<code>\msg_error:nnnn</code> 68
D	<code>\msg_new:nnn</code> 211, 212, 213, 214, 215, 216
<code>\DeclareDocumentCommand</code> 87, 186	<code>\msg_warning:nnn</code> 223, 241, 249
<code>\def</code> 179	
E	N
<code>\end</code> 297	<code>\NewDocumentCommand</code>
 2, 7, 331, 336, 341, 422, 429

O		S	
options:		seq commands:	
*	5	\seq_gclear_new:N	18, 81
+	5	\seq_gput_right:Nn	10, 290
<code ₁ >	4	\seq_if_empty:NTF	46
<code ₂ >	5	\seq_map_function:NN	
<kv ₁ >	5		19, 49, 145, 271, 292
<tl ₁ >	4	\seq_set_from_clist:Nn	165, 270
<tl ₂ >	4	\seq_use:Nnnn	301
<tl ₃ >	5	sys commands:	
<tl ₄ >	5	\c_sys_day_int	310
<tl ₅ >	5	\c_sys_hour_int	319
<tl ₆ >	5	\c_sys_jobname_str	327
Expans	26	\c_sys_minute_int	320
File	26	\c_sys_month_int	309
Inner	26	\c_sys_year_int	308
Outer	27		
Param	26	T	
Separ	27	tl commands:	
Write	27	\c_empty_tl	47, 66, 162, 182, 204, 336
P		\tl_count:n	77
prg commands:		\tl_gset:Nn	103, 219, 227, 237, 353
\prg_replicate:nn	57	\tl_gset_eq:NN	349
prop commands:		\tl_log:n	110, 123
\prop_clear_new:N	266	\tl_new:N	31, 100, 117
\prop_gclear_new:N	45	\tl_trim_spaces:n	10, 36, 37
\prop_gput:Nnn	28, 255	U	
\prop_if_exist:NTF	275	use commands:	
\prop_item:Nn	255, 279, 290	\use:N	35, 431
\prop_map_function:NN	257	\use_i:nn	60, 62
\prop_new:N	25, 284	\use_ii:nn	61
\ProvideDocumentCommand	131, 337	\usepackage	4
Q			
quark commands:			
\q_stop	8, 14, 32, 41		

Part IV

Implementation

1 Opening

```

1 <@@=ccool>
2 \ExplSyntaxOn

```

2 aux

```

\__ccool_aux_inner_set:n #1: <code>
3 \cs_new_protected:Nn \__ccool_aux_inner_set:n
4 {
5   \cs_gset:Npn \__ccool_aux_inner:n ##1 {#1}
6   \cs_generate_variant:Nn \__ccool_aux_inner:n { e }
7 }

```

(End definition for __ccool_aux_inner_set:n.)

```

\__ccool_aux_key:w #1: <key>
#2: <value>
8 \cs_new_protected:Npn \__ccool_aux_key:w #1 = #2 \q_stop
9 {
10  \seq_gput_right:Nx \g__ccool_aux_key_seq { \tl_trim_spaces:n{#1} }
11 }

```

(End definition for __ccool_aux_key:w.)

```

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

```

(End definition for __ccool_aux_key:n.)

```

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

```

(End definition for __ccool_aux_key:N.)

```

\__ccool_aux_outer_set:n #1: <inline code>
21 \cs_new_protected:Nn \__ccool_aux_outer_set:n
22 {
23   \cs_gset:Npn \__ccool_aux_outer:n ##1 {#1}
24 }

```

(End definition for __ccool_aux_outer_set:n.)

```

\__ccool_aux_prop:nn
25 \prop_new:N \g__ccool_aux_prop
26 \cs_new_protected:Nn \__ccool_aux_prop:nn
27 {
28   \prop_gput:Nnn \g__ccool_aux_prop{#1}{#2}
29 }
30 \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 >

31 \tl_new:N \g__ccool_option_expans_tl
32 \cs_new_protected:Npn \__ccool_aux_prop:w #1 = #2 \q_stop
33 {
34   \exp_args:Nx
35   \use:c{\__ccool_aux_prop:\g__ccool_option_expans_tl}
36   { \tl_trim_spaces:n{#1} }
37   { \__ccool_aux_inner:n{ \tl_trim_spaces:n{#2} } }
38 }

(End definition for \__ccool_aux_prop:w.)

```

```

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

39 \cs_new_protected:Nn \__ccool_aux_prop:n
40 {
41   \__ccool_aux_prop:w #1 \q_stop
42 }

(End definition for \__ccool_aux_prop:n.)

```

```

\__ccool_aux_prop:N #1 : < keyval list >

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

(End definition for \__ccool_aux_prop:N.)

```

```

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

52 \cs_new:Nn \__ccool_aux_separ:nn
53 {
54   \int_case:nnTF {#1}
55   {
56     {1}
57     { \prg_replicate:nn{ 3 }{#2} }
58     {2}
59     {

```

```

60      { \use_i:nn #2 }
61      { \use_ii:nn #2 }
62      { \use_i:nn #2 }
63    }
64    {3}{#2}
65  }
66  { \c_empty_tl }
67  {
68    \msg_error:nnnn { __ccool }
69    { separ }
70    { \exp_not:N \__ccool_aux_separ:nn }
71    {#2}
72  }
73 }
74 \cs_generate_variant:Nn \__ccool_aux_separ:nn { e }

```

(End definition for __ccool_aux_separ:nn.)

```

\__ccool_aux_separ:n #1 : < tokens >
75 \cs_new:Nn \__ccool_aux_separ:n
76 {
77   \__ccool_aux_separ:en{ \tl_count:n{#1} }{#1}
78 }

```

(End definition for __ccool_aux_separ:n.)

```

\__ccool_aux_val:Nn #1 : < seq >
#2 : < tl var name >
79 \cs_new_protected:Nn \__ccool_aux_val:Nn
80 {
81   \seq_gclear_new:N \g__ccool_aux_val_seq
82   \__ccool_seq_from_prop:NNn \g__ccool_aux_val_seq #1 { \__ccool_prop_name:n{#2} }
83 }

```

(End definition for __ccool_aux_val:Nn.)

3 lambda

```

\__ccool_lambda:nn [7]
84 \cs_new_protected:Npn \__ccool_lambda:nn #1 #2
85 {
86   \exp_args:NNx
87   \DeclareDocumentCommand \__ccool_lambda_expression
88     {#1}
89     {#2}
90     \__ccool_lambda_expression
91 }

```

(End definition for __ccool_lambda:nn.)

4 log

_ccool_log_close:

```

92 \iow_new:N \g__ccool_log_iow
93 \AtEndDocument{\iow_close:N \g__ccool_log_iow}
94 \bool_set_false:N \g__ccool_log_open_bool
95 \cs_new_protected:Nn \_ccool_log_close:
96 {
97   \iow_close:N \g__ccool_log_iow
98   \bool_gset_false:N \g__ccool_log_open_bool
99 }

```

(End definition for _ccool_log_close:.)

_ccool_log_open:

```

100 \tl_new:N \g__ccool_log_file_tl
101 \cs_new_protected:Nn \_ccool_log_open:
102 {
103   \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
104   \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
105   \bool_gset_true:N \g__ccool_log_open_bool
106 }

```

(End definition for _ccool_log_open:.)

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

```

107 \cs_new_protected:Nn \_ccool_log_read:n
108 {
109   \file_input:n{#1}
110   \tl_log:n{read~from~#1}
111 }
112 \cs_generate_variant:Nn \_ccool_log_read:n { e }

```

(End definition for _ccool_log_read:n.)

_ccool_log_read:

```

113 \cs_new_protected:Nn \_ccool_log_read:
114 {
115   \_ccool_log_read:e{\g__ccool_log_to_tl}
116 }

```

(End definition for _ccool_log_read:.)

_ccool_log_write:n

```

117 \tl_new:N \g__ccool_log_to_tl
118 \cs_new_protected:Nn \_ccool_log_write:n
119 {
120   \bool_if:nTF{ \g__ccool_log_open_bool }
121   {
122     \iow_now:Nn \g__ccool_log_iow {#1}
123     \tl_log:n{ write~to~#1 }
124   }
125   { \msg_error:nnnn{ __ccool }{ iow }{ \g__ccool_log_iow } }
126 }
127 \cs_generate_variant:Nn \_ccool_log_write:n { e }

```

(End definition for _ccool_log_write:n.)

5 make_key

```

__ccool_make_key:Nn #1 : < token >
#2 : < key >

128 \cs_new_protected:Nn \__ccool_make_key:Nn
129 {
130   \exp_args:NNx
131   \ProvideDocumentCommand{#1}
132   { D<>{\g__ccool_option_param_tl} }
133   {
134     \__ccool_prop_item:nn{#1}{#2}
135   }
136 }
137 \cs_generate_variant:Nn \__ccool_make_key:Nn {c}

(End definition for \__ccool_make_key:Nn.)

__ccool_make_key:n #1 : < key >

138 \cs_new_protected:Nn \__ccool_make_key:n
139 {
140   \__ccool_make_key:cn{#1}{#1}
141 }
142 \cs_generate_variant:Nn \__ccool_make_key:n { e }

(End definition for \__ccool_make_key:n.)

__ccool_make_key:N #1 : < seq >

143 \cs_new_protected:Nn \__ccool_make_key:N
144 {
145   \seq_map_function:NN #1 \__ccool_make_key:e
146 }

(End definition for \__ccool_make_key:N.)

```

6 make_ccool

```

__ccool_make_ccool_exp:nnn

147 \cs_new_protected:Nn \__ccool_make_ccool_exp:nnn
148 {
149   \__ccool_aux_val:Nn \g__ccool_aux_key_seq {#1}
150   \__ccool_aux_outer_set:n{#3}
151   \__ccool_aux_outer:n
152   {
153     \exp_args:NNf
154     \__ccool_seq_use:Nn
155     \g__ccool_aux_val_seq
156     {#2}
157   }
158 }

(End definition for \__ccool_make_ccool_exp:nnn.)

```

_ccool_make_ccool_key:nnn

```

159 \cs_new_protected:Nn \_ccool\_make\_ccool\_key:nnn
160 {
161   \_ccool\_prop\_if\_exist:nTF{#1}
162   { \c_empty_tl }
163   { \_ccool\_prop\_new:n{#1} }
164   \exp_args:No \_ccool\_aux\_inner\_set:n{#2}
165   \seq_set_from_clist:Nn \g\_ccool\_aux\_keyval\_seq {#3}
166   \_ccool\_aux\_prop:N \g\_ccool\_aux\_keyval\_seq
167   \_ccool\_prop\_append:Nn \g\_ccool\_aux\_prop {#1}
168   \_ccool\_aux\_key:N \g\_ccool\_aux\_keyval\_seq
169   \_ccool\_make\_key:N \g\_ccool\_aux\_key\_seq
170 }

```

(End definition for _ccool_make_ccool_key:nnn.)

_ccool_make_ccool_sideeffect:nnn [9]

```

171 \cs_new_protected:Nn \_ccool\_make\_ccool\_sideeffect:nnn
172 {
173   \_ccool\_make\_ccool\_key:nnn{#1}{#2}{#3}
174   \bool_if:nTF{ \g\_ccool\_log\_open\_bool }
175   {
176     \_ccool\_log\_write:n
177     {
178       \begin{group}
179       \def \_ccool\_log\_entry { \Ccool<#1>i{#2}{#3} } \expandafter
180       \end{group} \_ccool\_log\_entry
181     }
182   }{\c_empty_tl}
183 }

```

(End definition for _ccool_make_ccool_sideeffect:nnn.)

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

```

184 \cs_new_protected:Npn \_ccool\_make\_ccool:nnnn #1 #2 #3 #4
185 {
186   \exp_args:NNx \DeclareDocumentCommand \Ccool
187   {%^^A 2 3 4 5 6 7 8 9
188     +o D<>{#1} E{ i }{#2}} m t+ s E{ s o }{#3}{#4}} +o
189   }
190   {
191     \IfValueT{##1}{##1}
192     \_ccool\_make\_ccool\_sideeffect:nnn{##2}{##3}{##4}
193     \IfBooleanT{##6}
194     {
195       \_ccool\_make\_ccool\_exp:nnn{##2}{##7}{##8}
196     }
197     \bool_if:nTF{##5}
198     {
199       \gappto{\CcoolHook}
200       {

```



```

201         \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
202     }
203 }
204 {\c_empty_tl}
205 \IfValueT{##9}
206 {
207     \exp_not:n{ \Ccool[##9] }
208 }
209 }
210 }

```

(End definition for __ccool_make_ccool:nnnn.)

7 msg

```

211 \msg_new:nnn {\__ccool}{ generic }{#1}
212 \msg_new:nnn {\__ccool}{ iow }{#1~is~closed~can't~write}
213 \msg_new:nnn {\__ccool}{ keyonly }{#1~does~not~take~values;~keyval~is~#2}
214 \msg_new:nnn {\__ccool}{ keywrong }{#1~does~not~recognize~key~#2}
215 \msg_new:nnn {\__ccool}{ separ }{#1~expects~1~to~3~items,~#2}
216 \msg_new:nnn {\__ccool}{ unset }{#1~unset}

```

8 option

__ccool_option_inner:n #1: *<code>*

```

217 \cs_new_protected:Nn \__ccool_option_inner:n
218 {
219     \tl_gset:Nn \g__ccool_option_inner_tl {#1}
220 }
221 \__ccool_option_inner:n
222 {
223     \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_inner_tl }
224 }

```

(End definition for __ccool_option_inner:n.)

__ccool_option_param:n #1: *<token list>*

```

225 \cs_new:Nn \__ccool_option_param:n
226 {
227     \tl_gset:Nn \g__ccool_option_param_tl{#1}
228 }
229 \__ccool_option_param:n
230 {
231     \msg_error:nnx{ __ccool }
232     { generic }
233     { \exp_not:N\g__ccool_option_param_tl~undefined }
234 }

```

(End definition for __ccool_option_param:n.)

__ccool_option_outer:n #1: *<inline code>*

```

235 \cs_new_protected:Nn \__ccool_option_outer:n
236 {

```

```

237 \tl_gset:Nn \g__ccool_option_outer_tl {#1}
238 }
239 \__ccool_option_outer:n
240 {
241 \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_outer_tl }
242 }

```

(End definition for __ccool_option_outer:n.)

```

\__ccool_option_separ:n #1 : {< tl1 >}{< tl2 >}{< tl3 >}
243 \cs_new_protected:Nn \__ccool_option_separ:n
244 {
245 \cs_gset:Npn \g__ccool_option_separ_tl {#1}
246 }
247 \__ccool_option_separ:n
248 {
249 \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_separ_tl }
250 }

```

(End definition for __ccool_option_separ:n.)

9 prop

```

\__ccool_prop_append:NN #1 : < prop1 >
#2 : < prop2 >
251 \cs_new_protected:Npn \__ccool_prop_append:NN #1 #2
252 {
253 \cs_set:Nn \__ccool_prop_append:nn
254 {
255 \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
256 }
257 \prop_map_function:NN #2 \__ccool_prop_append:nn
258 }
259 \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 >
260 \cs_new_protected:Nn \__ccool_prop_append:Nn
261 {
262 \__ccool_prop_append:cN{ \__ccool_prop_name:n {#2} } #1
263 }

```

(End definition for __ccool_prop_append:Nn.)

```

\__ccool_prop_clear_new:n #1 : < tl var name >
264 \cs_new_protected:Nn \__ccool_prop_clear_new:n
265 {
266 \exp_args:No \prop_clear_new:c{ \__ccool_prop_name:n {#1} }
267 }

```

(End definition for __ccool_prop_clear_new:n.)

```

\__ccool_prop_clear_new_map:n #1 : < keyval list >
268 \cs_new_protected:Nn \__ccool_prop_clear_new_map:n
269 {
270   \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
271   \seq_map_function:NN \g__ccool_aux_key_seq \__ccool_prop_clear_new:n
272 }
(End definition for \__ccool_prop_clear_new_map:n.)

```

```

\__ccool_prop_if_exist:nTF #1 : < tl1 >
#2 : < tl2 >
#3 : < tl3 >
273 \cs_new:Nn \__ccool_prop_if_exist:nTF
274 {
275   \prop_if_exist:cTF{ \__ccool_prop_name:n {#1} }{#2}{#3}
276 }
(End definition for \__ccool_prop_if_exist:nTF.)

```

```

\__ccool_prop_item:nn #1 : < tl var name >
#2 : < key >
277 \cs_new:Nn \__ccool_prop_item:nn
278 {
279   \prop_item:cn { \__ccool_prop_name:n {#1} } {#2}
280 }
(End definition for \__ccool_prop_item:nn.)

```

```

\__ccool_prop_name:n #1 : < tl var name >
281 \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 >
282 \cs_new_protected:Nn \__ccool_prop_new:n
283 {
284   \prop_new:c{ \__ccool_prop_name:n {#1} }
285 }
(End definition for \__ccool_prop_new:n.)

```

10 seq

```

\__ccool_seq_from_prop:NNn #1 : < seq1 >
#2 : < seq2 > (keys)
#3 : < prop >
286 \cs_new_protected:Nn \__ccool_seq_from_prop:NNn
287 {
288   \cs_set_protected:Nn \__ccool_seq_from_prop:n
289   {
290     \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
291   }
292   \seq_map_function:NN #2 \__ccool_seq_from_prop:n
293 }

```

(End definition for _ccool_seq_from_prop:Nn.)

_ccool_erw_seq_use:Nn

```

294 %      \begin{arguments}
295 %      \item \meta{ seq }
296 %      \item \meta{ tokens }
297 %      \end{arguments}
298 \cs_new:Nn \_ccool_seq_use:Nn
299 {
300   \exp_last_unbraced:NNf
301   \seq_use:Nnnn #1
302   \_ccool_aux_separ:n{#2}
303 }

```

(End definition for _ccool_erw_seq_use:Nn.)

11 sys

_ccool_sys_date:

```

304 \cs_new:Nn \_ccool_sys_date:
305 {
306   \int_eval:n
307   {
308     \c_sys_year_int * 10000
309     +\c_sys_month_int * 100
310     +\c_sys_day_int * 1
311   }
312 }

```

(End definition for _ccool_sys_date:.)

_ccool_sys_date_hex:

```

313 \cs_new:Nn \_ccool_sys_date_hex:
314 {\int_to_hex:n{\_ccool_sys_date:}}

```

(End definition for _ccool_sys_date_hex:.)

_ccool_sys_time:

```

315 \cs_new:Nn \_ccool_sys_time:
316 {
317   \int_eval:n
318   {
319     \c_sys_hour_int * 100
320     +\c_sys_minute_int * 1
321   }
322 }

```

(End definition for _ccool_sys_time:.)

_ccool_sys_time_hex:

```

323 \cs_new:Nn \_ccool_sys_time_hex:
324 {\int_to_hex:n{\_ccool_sys_time:}}

```

(End definition for _ccool_sys_time_hex:.)

```

\__ccool_sys_filename:
325 \cs_new:Nn\__ccool_sys_filename:
326 {
327   \c_sys_jobname_str--
328   \__ccool_sys_date_hex--
329   \__ccool_sys_time_hex:
330 }
(End definition for \__ccool_sys_filename:.)

```

12 Front-end

\CcoolClear #1 : $\langle token\ list \rangle$

Semantics Clears any data created by `\Ccool{ $\langle token\ list \rangle$ }`

```

331 \NewDocumentCommand{ \CcoolClear }
332 { D<>{\g__ccool_option_param_tl} }
333 {
334   \__ccool_prop_clear_new_map:n{#1}
335 }

```

\CcoolHook

Example `\AfterEndEnvironment{theorem}{\CcoolHook}`

```

336 \NewDocumentCommand{\CcoolHook}{\c_empty_tl}

```

\CcoolLambda

#1 : $\langle arg\ spec \rangle$
 #2 : $\langle code \rangle$

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

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

```

337 \ProvideDocumentCommand \CcoolLambda { 0{m} m }
338 {
339   \__ccool_lambda:nn { #1 } { #2 }
340 }

```

\CcoolOption

#1 : *<keyval list>*

```
341 \NewDocumentCommand{ \CcoolOption }
342 { m }
343 {
344   \keys_set:nn{ __ccool }{#1}
345 }

346 \keys_define:nn { __ccool }
347 {
```

Expans

Value *eo|ee|ex|xo|xe|xx*

```
348 Expans .multichoices:nn = { eo, ee, ex, xo, xe, xx }
349 { \tl_gset_eq:NN \g__ccool_option_expans_tl \l_keys_choice_tl },
350 Expans .default:n = { xo },
351 Expans .initial:n = { xo },
```

File

Value *<path>*

```
352 File .code:n = {
353   \tl_gset:Nx \g__ccool_log_file_tl{#1}
354 },
355 File .default:n = { \__ccool_sys_filename: },
356 File .initial:n = { \__ccool_sys_filename: },
```

Inner

Value *<code>*, with **###1** as the argument to be replaced

```
357 Inner .code:n={
358   \__ccool_option_inner:n{#1}
359   \exp_last_unbraced:Nf
360   \__ccool_make_ccool:nnnn
361   {
362     { \g__ccool_option_param_tl }
363     { \g__ccool_option_inner_tl }
364     { \g__ccool_option_separ_tl }
365     { \g__ccool_option_outer_tl }
366   }
367 },
368 Inner .value_required:n = false,
369 Inner .default:n = {###1},
370 Inner .initial:n = {###1},
```

Param

Value *<token list>*

```
371 Param .code:n={
372   \__ccool_option_param:n{#1}
373   \exp_last_unbraced:Nf
374   \__ccool_make_ccool:nnnn
375   {
```

```

376     { \g__ccool_option_param_tl }
377     { \g__ccool_option_inner_tl }
378     { \g__ccool_option_separ_tl }
379     { \g__ccool_option_outer_tl }
380   }
381 },
382 Param .value_required:n = false,
383 Param .default:n = { Default },
384 Param .initial:n = { Default },

```

Outer

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

```

385 Outer .code:n={
386   \__ccool_option_outer:n{#1}
387   \exp_last_unbraced:Nf
388   \__ccool_make_ccool:nnnn
389   {
390     { \g__ccool_option_param_tl }
391     { \g__ccool_option_inner_tl }
392     { \g__ccool_option_separ_tl }
393     { \g__ccool_option_outer_tl }
394   }
395 },
396 Outer .value_required:n = false,
397 Outer .default:n = { \ensuremath{####1} },
398 Outer .initial:n = { \ensuremath{####1} },

```

Separ

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

```

399 Separ .code:n={
400   \__ccool_option_separ:n{#1}
401   \exp_last_unbraced:Nf
402   \__ccool_make_ccool:nnnn
403   {
404     { \g__ccool_option_param_tl }
405     { \g__ccool_option_inner_tl }
406     { \g__ccool_option_separ_tl }
407     { \g__ccool_option_outer_tl }
408   }
409 },
410 Separ .value_required:n = false,
411 Separ .default:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
412 Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },

```

Write

Value $\langle boolean \rangle$

```

413 Write .code:n = {
414   \bool_if:nTF{#1}
415   {\__ccool_log_open:}
416   {\__ccool_log_close:}
417 },

```

```

418 Write .value_required:n = false,
419 Write .default:n = \BooleanFalse,
420 Write .initial:n = \BooleanFalse
421 }

```

\CcoolRead #1 : $\langle path \rangle$

Semantics

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

```

422 \NewDocumentCommand{\CcoolRead}
423 {o}
424 {
425   \IfValueTF{#1}
426   {\_ccool_log_read:e{#1}}
427   {\_ccool_log_read:}
428 }

```

\CcoolVers

Semantics Expands to the package’s version

```

429 \NewDocumentCommand{\CcoolVers}
430 {}
431 {\use:c{ver@ccool.sty}}

```

13 Closing

```

432 \ExplSyntaxOff

```