

The `ccool` package^{*}

Erwann Rogard[†]

Released 2020/04/21

Abstract

The package `ccool` for \LaTeX provides a *key-value* interface, `\Ccool`, meant to facilitate the generation of commands. Optional parameters that control the processing of the input and its expansion are set to their most likely usage. This can be used to encode notational conventions (such as `\Real` \rightarrow `\mathbb{R}`) at the point where they are introduced in the `document` (“Let \mathbb{R} denote real numbers”). Polymorphic commands can be generated by parameterizing the keys (for instance, one parameter value for style, another for a property). User input to `\Ccool` can optionally be serialized. This can be useful for typesetting documents sharing the same notation.

Résumé

L’extension `ccool` pour \LaTeX met à disposition une interface de type *clé-valeur*, `\Ccool`, destinée à faciliter la génération de commandes. Les paramètres optionnels contrôlant le traitement de ces *clé-valeur* sont fixés par défaut pour répondre aux besoins courants. Ceci peut-être utilisé pour la command-isation des conventions de notation (`\Reel` \rightarrow `\mathbb{R}`), au point dans le `document` où elles sont introduites (“Soit \mathbb{R} les nombres réels.”). Des commandes polymorphes peuvent être générées, en associant aux clés un paramètre (par exemple, une valeur pour le style typographique, une autre pour la description du concept associé). En option, les instructions passées à cette interface peuvent être sauvegardées, ce qui peut être utile pour la rédaction de documents faisant appel à des conventions typographiques communes.

Contents

I	Usage	4
0	Convention	4
1	Loading the package	5

^{*}This file describes version v2.5, last revised 2020/04/21.

[†]firstname dot lastname AusTria gmail dot com

2	\Ccool	5
2.1	Core feature	5
2.2	Process the <i>val_i</i> 's	5
2.3	Append to a hook	5
2.4	Expand the <i>val_i</i> 's	5
2.5	Head	5
2.6	Tail	6
2.7	Parameterize the <i>key_i</i> 's	6
2.8	Write	6
3	\CcoolClear	6
4	\CcoolHook	6
5	\CcoolLambda	6
6	\CcoolOption	6
6.1	Expans	6
6.3	Inner	6
6.4	Param	7
6.5	Outer	7
6.6	Separ	7
6.7	Write	7
7	\CcoolRead	7
8	\CcoolVers	7
9	Do's and dont's	7
II	Listing	9
1.	\CcoolVers	9
2.	“Let \mathbb{N} and \mathbb{R} denote...” (start of the tutorial)	9
3.	Equivalent to 2, with \NewDocumentCommand	9
4.	Equivalent to 3, with \Ccool	9
5.	Equivalent to 4, with expansion	9
6.	Equivalent to 4, parameterized (end of the tutorial)	10
7.	Separators.	10
8.	Hello, world! (testing)	10
9.	Listing 8 read from file	11
10.	Probability space	11

11. Listing 10 read from file	11
12. Mittelwertsatz für n Variable.	12
13. Listing 12 read from file	12
14. Polynôme	13
15. Listing 14 read from file	13
16. Same as Listing 14, but arbitrary number system	13
15. Listing 16 read from file	14
18. Fonction et fonctionnelle	14
19. Listing 18 read from file	14
20. CUSUM statistic.	14
21. Listing 20 read from file	15
III Other	16
1 Acknowledgment	16
2 Genealogy	16
3 Install	16
4 Issue	16
5 Support	16
6 Testing	16
6.1 Technicality	16
6.2 Platform	16
6.3 Engine	17
6.4 Results	17
6.5 Other	17
7 To do	17
8 References	17
IV Implementation	19
1 Opening	19
2 aux	19

3	<code>lambda</code>	21
4	<code>log</code>	21
5	<code>make_key</code>	22
6	<code>make_ccool</code>	23
7	<code>msg</code>	24
8	<code>option</code>	24
9	<code>prop</code>	25
10	<code>seq</code>	27
11	Front-end	27
	11.1 <code>\CcoolClear</code>	27
	11.2 <code>\CcoolHook</code>	27
	11.3 <code>\CcoolLambda</code>	27
	11.4 <code>\CcoolOption</code>	27
	11.4.1 <code>Expans</code>	28
	11.4.2 <code>File</code>	28
	11.4.3 <code>Inner</code>	28
	11.4.4 <code>Param</code>	28
	11.4.5 <code>Outer</code>	29
	11.4.6 <code>Separ</code>	29
	11.4.7 <code>Write</code>	29
	11.5 <code>\CcoolRead</code>	29
	11.6 <code>\CcoolVers</code>	30
12	Closing	30

Part I

Usage

Convention

- a) Loosely, those of [2], for example as to the meaning of $\langle token\ list \rangle$.
- b) Those of [4], for example `[arg]` is a ‘*o*’-type argument.
- c) $\langle X \rangle \leftarrow Y$: set $\langle X \rangle$ to `Y`
- d) $\backslash X \rightarrow Y$: `\X` expands to `Y`
- e) If unspecified, the environment in which a macro is to be used is **document**.

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

Requirement

1. `ccool.sty` is in the path of the \LaTeX engine. See [Part III, section 5](#).
2. Put in the *preamble*

<code>\Ccool</code>	<code>\Ccool[⟨tl₁⟩]<⟨tl₂⟩>c{⟨code₁⟩}{⟨kv₁⟩}+*s{⟨separators⟩}c{⟨code₂⟩}[⟨tl₆⟩]</code>
---------------------	--

where $\langle separators \rangle$ is either of: $\{\langle tl_3 \rangle\}$, $\{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\}$, and $\{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\}\{\langle tl_5 \rangle\}$.

Semantics See [subsection 2.1-2.8](#).

2.1 Core feature

`\Ccool{⟨kv1⟩}` creates for each $\langle key_i \rangle = \langle val_i \rangle$, the command $\backslash \langle key_i \rangle$, according to the following algorithm:

- 1) $\langle val_i \rangle \leftarrow \backslash \text{function}\{\langle val_i \rangle\}$
- 2) Creates $\backslash \langle key_i \rangle$, such that $\backslash \langle key_i \rangle \rightarrow \langle val_i \rangle$,

where `\function` is controled by option [Inner](#). For instance, the side effect of `\Ccool{ Real = \mathbb{R} }` is $\backslash \text{Real} \rightarrow \backslash \text{mathbb{R}}$. To be sparingly used, *option* [Expans](#) controls the way $\langle key_i \rangle$ and $\langle val_i \rangle$ are expanded.

See `\CcoolLambda` to allow command $\backslash \langle key_i \rangle$ to take arguments.

2.2 Process the val_i 's

`\Ccool c{⟨code1⟩}{⟨kv1⟩}` is identical to the [Core feature](#), except it overrides [Inner](#).

In our example, if multiple number systems are defined with `\Ccool` (natural, reals, ...), it is more efficient to omit `\mathbb{.}` inside $\langle val_i \rangle$ and, instead, use `c{\mathbb{#1}}`, where `#1` means “parameter to be replaced”.

2.3 Append to a hook

`\Ccool{⟨kv1⟩}+` is identical to the [Core feature](#), except it repeats after `\CcoolHook`. This is useful to make the side effect persist after a *local group* (such as `theorem`).

2.4 Expand the val_i 's

`\Ccool{⟨kv1⟩}*` supplements the [Core feature](#) with the expansion of the $\langle val_i \rangle$'s using typesetting rules controlled by *option* [Separ](#) and [Outer](#). The first are *separators* applied to the $\langle val_i \rangle$'s to form a *token list*, and the second a function applied to the latter.

They can be overridden inline by appending further `s{⟨separators⟩}` and `c{⟨code2⟩}`, respectively, to the list of arguments.

2.5 Head

`\Ccool[⟨tl1⟩]{⟨kv1⟩}` expands $\langle tl_1 \rangle$ and executes the [Core feature](#).

There may be situations where it is convenient to pass $\langle tl_1 \rangle$ as empty.

2.6 Tail

`\Ccool{⟨kv1⟩}[⟨tl6⟩]{⟨kv2⟩}` is identical to `\Ccool{⟨kv1⟩}` followed by `\Ccool[⟨tl6⟩]{⟨kv2⟩}`.

The combination of **Core feature**, **Head**, and **Tail** allows to integrate typesetting and the creation of commands.

2.7 Parameterize the *key*_{*i*}'s

`\Ccool<⟨tl2>⟩{⟨kv1⟩}` is identical to the **Core feature**, except *key*_{*i*} is replaced by *key*_{*i*}<⟨*tl*₂>⟩. The default parameter, that implicit in *key*_{*i*}, is controlled by **Param**. In our example, *tl*₂ could be **Style**.

2.8 Write

If *option* **Write** is set to `\BooleanTrue`, the **Core feature** is supplemented with the code written to a file, whose path is controlled by *option* **File**.

`\CcoolClear` `\CcoolClear<⟨tl2>⟩{⟨clist⟩}`

Semantics Clears all `\⟨keyi<⟨tl2>⟩`'s

`\CcoolHook` `\CcoolHook`

Semantics No side effect or expansion

`\CcoolLambda` `\CcoolLambda[⟨arg spec⟩]{⟨code⟩},`

where *arg spec* is by default an ‘*o*’-type *argument*.

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

Semantics Returns a command of type `\DeclareDocumentCommand`[\[4\]](#),

`\CcoolOption` `\CcoolOption{⟨keyval list⟩}`

Semantics Controls the default behavior of `\Ccool`.

Expans

Also see **Part IV**, **Expans**

Semantics See **Core feature**

Syntax `eo|ee|ex|xoxe|xx`

File

Also see **Part IV**, **File**

Semantics See **Write**

Syntax `⟨path⟩`

Inner

Also see [Part IV, Inner](#)

Semantics See [Process the \$val_i\$'s](#)

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

Param

Also see [Part IV, Param](#)

Semantics See [Parameterize the \$key_i\$'s](#)

Syntax $\langle token\ list \rangle$

Outer

Also see [Part IV, Outer](#)

Semantics See [Expand the \$val_i\$'s](#)

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

Separ

Also see [Part IV, Separ](#)

Semantics See [Expand the \$val_i\$'s](#)

Syntax That of *separators* in [\[2, Section 8 of l3seq\]](#)

Write

Also see [Part IV, Write](#)

Semantics See [Write](#)

Syntax $\langle boolean \rangle$

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

Also see [Part IV, \CcoolRead](#)

Semantics

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

\CcoolVers \CcoolVers

Semantics \rightarrow the package’s version

9 Do’s and dont’s

1)

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

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

2)

Don’t: $[a, b)$

Do: $\{ [] a, b \{ \}$

3)

Don't: `\cal F`.

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

4)

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

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

5)

Don't: Use '*d*'-type or '*e*'-type arguments for `\CcoolLambda`

Do: Use only '*m*'-type and '*o*'-type arguments

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

Part II

Listing

NB:

1. These listings depend on the `\usepackage` statements of the source file’s **documentation**
2. Statements involving `Write` or `\CcoolClear` affect only the output of listings that come after that in which they appear. The demarcation is indicated by `%^A--->` and `%^A<---`, where applicable

Listing 1. `\CcoolVers`

```
\CcoolVers
```

2020/04/21 v2.5 cool — A key-value interface for generating commands

Listing 2. “Let \mathbb{N} and \mathbb{R} denote...” (start of the tutorial)

```
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 3. Equivalent to **2**, with `\NewDocumentCommand`

```
\DeclareDocumentCommand\Nat{}{\mathbb{N}}
\DeclareDocumentCommand\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 4. Equivalent to **3**, with `\Ccool`

```
%^A--->
\Ccool c{\mathbb{#1}}{ Nat = {N}, Real = {R} }
Let~$\Nat$ and $\Real$~denote the natural and real numbers.
%^A<---
\CcoolClear
```

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

Listing 5. Equivalent to **4**, with expansion

```
%^A--->
\Ccool[Let~]
c{\mathbb{#1}}{ Nat = {N}, Real = {R} }*s{\rm{and}~}
[-denote the natural and real numbers.]{ }
%^A<---
\CcoolClear
```

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

Listing 6. Equivalent to 4, parameterized (end of the tutorial)

```
%^A--->
\Cool<Style>c{\mathbb{#1}}{ Nat = {N}, Real = {R} }
[Let $\Nat<Style>$ and $\Real<Style>$ denote the natural and real
 numbers.]{ }
%^A<---
\CoolClear<Style>
```

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

Listing 7. Separators

```
%^A--->
\CoolOption{
  Separ={\ \char`@ \ }{\ \% \ }{\ \char`@ \ }}
\Cool{ X = x, Y = y }*[\]
{ X = x, Y = y }*s{\&~}\[\]
{ X = x, Y = y }*s{\{,~}\&~}\[\[1em]]
{ X = x, Y = y, Z = z }*[\]
{ X = x, Y = y, Z = z }*s{\&~}\[\]
{ X = x, Y = y, Z = z }*s{\{,~}\&~}\[\]
{ X = x, Y = y, Z = z }*s{\&~}\{,~}\&~}\[\]
%^A<---
\CoolClear
```

$x @ y$
 $x \& y$
 $x \& y$

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

Listing 8. Hello, world! (testing)

```
\CoolOption{ Write = \BooleanTrue }
%^A--->
\CoolOption{Separ = {\}\{.}\{.}\}, Outer = {###1}}
\Cool
<Test>{ KeyA = {\}, KeyB = {\!}, KeyC = {\%} }[]
<Test>{ KeyD = {d}, KeyE = {\%} }[]
<Test>c{\{1\}}{ KeyF = {H}, KeyG = {e}, KeyH = {1} }*[]
<Test>{ KeyI = {\%}, KeyJ = {\%}, KeyK = {\%} }[.\{1\}.\{o\}]
<Test>{ KeyL = {1}, KeyM = {\char`[]}, KeyN = {\char`[]} }[]
```

```

<Test>{ KeyO = {o}, KeyP = {\%}, KeyQ = {\%} }[{, \ }]
<Test>{ KeyR = {w}, KeyS = {o}, KeyT = {r} }*
s{{}}c{{\char`[]#1}[]}
<Test>{ KeyU = {\%}, KeyV = {\%}, KeyW = {\%} }[]
<Test>{ KeyX = {\%}, KeyY = {\%}, KeyZ = {\KeyB<Test>} }\nobreak
\KeyL<Test>\KeyD<Test>\KeyZ<Test>\KeyN<Test>\
%^^A<---
\CoolOption{ Write = \BooleanFalse }
\CoolClear

```

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

Listing 9. Listing 8 read from file

```

%^^A<--->
\CoolRead
\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
%^^A<---
\CoolClear

```

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

Listing 10. Probability space

```

\CoolOption{ Write = \BooleanTrue }
%^^A<--->
\Cool[Let~]
{ Space = \Omega, Field = \mathcal{F}, Meas = \mathcal{P} }
*s{{,}}c{{\#1\}}
[~denote the probability space, where~]{ PowerSet = { 2^{\Space} } }
[{\Field\subset \PowerSet$.]
{}
\CoolOption{ Write = \BooleanFalse }
%^^A<---
\CoolClear

```

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

Listing 11. Listing 10 read from file

```
%^A--->
\CcoolRead \tab $\Omega$ $\Field$ $\Meas$
%^A<---
\CcoolClear
```

Ω F P

Listing 12. Mittelwertsatz für n Variable[1, 17.3]

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

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 13. Listing 12 read from file

```
%^A--->
\CcoolRead \tab $N$ $R$ $\mathbb{D}$ $\mathbb{C}^1$ $\mathbb{D}$
%^A<---
```

```
\CcoolClear
```

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

Listing 14. Polynôme

```
\CcoolOption{ Write = \BooleanTrue }
% ^^A--->
\Ccool c{\mathbb{#1}}{ Nat = {N}, Reel = {R} }
[Soient~]
{ PolyR = \CcoolLambda[o]{\Reel\IfValueT{#1}{_#1}[X] } }
[ $\text{\PolyR}[n]$  et  $\text{\PolyR}$ , les familles de polynômes sur  $\text{\Reel}$ , de
degr\`e  $n$  et leur union pour  $n \in \text{\Nat}$ , respectivement. ]
{}
% ^^A<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear
```

Soient $\mathbb{R}_n[X]$ et $\mathbb{R}[X]$, les familles de polynômes sur \mathbb{R} , de degré n et leur union pour $n \in \mathbb{N}$, respectivement.

Listing 15. Listing 14 read from file

```
% ^^A--->
\CcoolRead \tab  $\text{\PolyR}[n]$  et  $\text{\PolyR}$ 
% ^^A<---
\CcoolClear
```

$$\mathbb{R}_n[X] \text{ et } \mathbb{R}[X]$$

Listing 16. Same as Listing 16, but arbitrary number system

```
\CcoolOption{ Write = \BooleanTrue }
% ^^A--->
\Ccool c{\mathbb{#1}}{ Corps = {K}, Nat = {N}, Reel = {R} }
[Soient~]
{
  Poly = \CcoolLambda[om]{#2\IfValueT{#1}{_#1}[X] },
  PolyR = \CcoolLambda[o]{\Poly[#1]{\Reel}}
}
[ $\text{\Poly}[n]{\text{\Corps}}$  et  $\text{\Poly}{\text{\Corps}}$ , les familles de polynômes sur
 $\text{\Corps}$ , de degr\`e  $n$  et leur union pour  $n \in \text{\Nat}$ ,
respectivement. En particulier,
ils sont d\`enot\`es  $\text{\PolyR}[n]$  et  $\text{\PolyR}$ , pour  $\text{\Corps}=\text{\Reel}$ .]
{}
% ^^A<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear
```

Soient $\mathbb{K}_n[X]$ et $\mathbb{K}[X]$, les familles de polynômes sur \mathbb{K} , de degré n et leur union pour $n \in \mathbb{N}$, respectivement. En particulier, ils sont dénotés $\mathbb{R}_n[X]$ et $\mathbb{R}[X]$, pour $\mathbb{K} = \mathbb{R}$.

Listing 17. Listing 16 read from file

```
%^A--->
\CcoolRead \tab $\PolyR[n]$ et $\PolyR$
%^A<---
\CcoolClear
```

$\mathbb{R}_n[X]$ et $\mathbb{R}[X]$

Listing 18. Fonction et fonctionnelle

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

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

Listing 19. Listing 18 read from file

```
%^A--->
\CcoolRead \tab $f$ EvalAt{t}$, $\ApplyOp{S}{f}$
%^A<---
\CcoolClear
```

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

Listing 20. CUSUM statistic[

```
\CcoolOption{ Write = \BooleanTrue }
%^A--->
\newtheorem{definition}{Definition}
\AfterEndEnvironment{definition}{\CcoolHook}
\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 } }+*c{ $\in$ \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$, \$ \CUSUM$)
% ^A<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear

```

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 21. Listing 20 read from file

```

% ^A--->
% \CcoolRead \tab \$ \Time $ \$ \Process$ \$ \Scale$ \$ \Real$ \$ \CUSUMthresh$
\$ \LogWald$ \$ \CUSUMst$ \$ \CUSUM$ \$ \LogWaldInf$
% ^A<---
% \CcoolClear
%

```

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

Part III

Other

1 Acknowledgment

This work has benefited from Q&A's from the L^AT_EXcommunity[6][10]. Specific attributions are made throughout this document.

2 Genealogy

« Give commands the ability to contain the mathematical meaning while retaining the typesetting versatility » (`cool`[1]). The addition of 'c', in `ccool`, is for *custom*. With hindsight it is restrictive to describe `ccool` as a tool for encoding mathematical convention.

3 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

4 Issue

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

5 Support

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

6 Testing

6.1 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 3** on one's own machine, against that **of the repository** for the same version.

6.2 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

6.3 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)

6.4 Results

- 1) ccool v1.8 compiles satisfactorily on platform *i)* and engine *a)*
- 2) ccool v1.8 compiles satisfactorily on platform *i)* and engine *b)*
- 3) ccool v1.9 compiles satisfactorily on platform *i)* and engines *b)* and *c)*
- 4) ccool v2.0 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*
- 5) ccool v2.1 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*
- 6) ccool v2.3 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

6.5 Other

Check [5] for testing ccool with llncs

7 To do

- 1) Placeholder passed to Part IV \CcoolOption should be #1 not ####1
- 2) \CcoolOption should behave in away similar to that described in Part I subsection 6.7

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>
- [10] \Ccool, extension à L^AT_EX à vocation mathématique, <http://forum.mathematex.net/latex-f6/ccool-extension-latex-a-vocation-mathematique-t17314.html>

Change History

v1.0		Add: optional <code>*to</code> to <code>\OpsNew</code> to make side effects persist beyond local group	18
General: Initial version	18	Delete: Listing 1., and 2.	18
v1.1		Replace: <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>	18
General: Add: <code>\Save</code>	18	v1.5	
Add: Listing 1., 2., 3., 4., 6., and 9.	18	General: Add: <code>\File</code>	18
Add: <code>\OpsRestore</code>	18	Delete: dependence on <code>\datetime</code>	18
Add: <code>\OpsTest</code>	18	v1.6	
Delete: Listing 1-5 from v1.0	18	General: Add: Listing showing part of the preamble	18
Fix: apparent anomaly in v1.0’s Listing 4, see Listing 8	18	Rename: <code>\OpsClear</code> to <code>\CcoolClear</code>	18
Rearrange: much of the implementation	18	Rename: <code>\OpsDebug</code> to <code>\CcoolDebug</code>	18
Replace: <code>\OpsOptions</code> by <code>\OpsOption</code>	18	Rename: <code>\OpsHook</code> to <code>\CcoolHook</code>	18
Replace: <code>{\langle kvl_2 \rangle}</code> by <code><kvl_2></code> given that option type G not recommended[4]	18	Rename: <code>\OpsOption</code> to <code>\CcoolOption</code>	18
Replace: <code>\GenericObject</code> by <code>\Name</code>	18	Rename: <code>\OpsRead</code> to <code>\CcoolRead</code>	18
Replace: <code>\Separators</code> by <code>\Separ</code>	18	Rename: <code>\Ops</code> to <code>\Ccool</code>	18
v1.2		Rename: <code>oops</code> to <code>ccool</code> (better describes the purpose)	18
General: Add: optional <code>*to</code> to <code>\OpsNew</code> as instruction to expand <code>kvl_1</code>	18	v1.7	
Delete: <code>\OpsTest</code>	18	General: Add: Legends to listings	18
Delete: <code>\langle kvl_2 \rangle</code> and <code>\langle code_2 \rangle</code>	18	Add: Listing 20 (CUSUM)	18
Delete: Listing 2-3 from v1.1.	18	Delete: <code>\CcoolDebug</code>	18
Replace: <code>\OpsClear{\langle tl_2 \rangle}</code> by <code>\OpsClear[\langle keyval list \rangle]</code>	18	Delete: Listing 5 from v1.6	18
Replace: <code>\Restore</code> by <code>\Read</code>	18	v1.8	
Replace: <code>\Save</code> by <code>\Write</code>	18	General: Add: <code>\CcoolVers</code>	18
v1.3		Add: <code>\CcoolLambda</code>	18
General: Replace: <code>\OpsNew</code> by <code>\Ops</code>	18	Add: Listing 18, Listing 19	18
Replace: <code>{\langle tl_2 \rangle}</code> and <code>[\langle tl_2 \rangle]</code> by <code><\langle tl_2 \rangle></code>	18	Add: Listing 1	18
v1.4		v1.9	
General: Add: section 9	18	General: Add: support for LuaT _E X	18
Add: <code>\OpsDebug</code>	18	Move: from Part I to Part IV, what is now that part’s section 11	18
Add: <code>\OpsHook</code>	18	v2.0	
Add: <code>\Expans</code> (for debugging’ sake, but...)	18	General: Add: support for X _Y T _E X	18
Add: Listing 1., 2., and 3.	18	Delete: <code>\File</code> ’s dependency on <code>\texosquery</code> and <code>\pdfcreationdate</code>	18

Update: <code>\RequirePackage</code> , <code>\NeedsTeXFormat</code> 's second argument / TeX Live 2020	18	Complete: Listing 14	18
v2.1		Rearranged: <code>\Ccool</code> 's subsections. Previously, by argument. Now, by feature.	18
General: Add: Listings 3, 4, 5, 6, 7, 8, and 9	18	Remove: Listing showing part of the preamble	18
Replace: <code>\tl2</code> 's position within <code>\Ccool</code> 's argument list, from first to second. Greater versatility	18	Replace: for <code>\Ccool</code> , <code>i{}</code> by <code>c{}</code> . .	18
Replace: <code>\CcoolLambda</code> 's optional integer argument (number of m's) by a standard argument list	18	Replace: In step 2), the created command's implementation, from <code>\ProvideDocumentCommand</code> to <code>\DeclareDocumentCommand</code>	18
Replace: Name by Param	18	v2.4	
Replace: as the de- fault of Param , Math by Default . .	18	General: Fix: minor error in the listings (<code>\Real</code> rather than <code>\Reel</code> , hitherto unnoticed).	18
v2.2		Remove: examples from Part I , <code>\Ccool</code> , as redundant with Part II Listing 2-6	18
General: Remove: % from listings . .	18	v2.5	
Replace: part of the abstract's with more straightforward descriptions based on input from forum participants	18	General: Modify: File , rely on <code>erw-l3</code> 's <code>\erw_jobnametimestamp</code> :	18
v2.3		Modify: behavior of Part I , Expand the <i>val_i</i> 's, rely on <code>erw-l3</code> 's <code>\erw_seq_use:Nn</code>	18
General: Add: Listing 15, Listing 16, and Listing 17	18		

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		<code>\bool_gset_true:N</code>	79
<code>\<key_i<tl2></code>	6	<code>\bool_if:nTF</code>	94, 148, 171, 277, 356
<code>\<key_i></code>	5, 7	<code>\bool_set_false:N</code>	68
<code>\<key_i></code>	5	<code>\BooleanFalse</code>	361, 362
Expans (option)	6, 28	<code>\BooleanTrue</code>	6
File (option)	6, 28	C	
Inner (option)	6, 28	<code>\Ccool</code>	1, 2, 5, 6, 6, 6, 9, 18, 153, 160, 181
Outer (option)	7, 29	ccool internal commands:	
Param (option)	7, 28	<code>__ccool_aux_inner:n</code>	6, 7, 38
Separ (option)	7, 29	<code>__ccool_aux_inner_set:n</code>	4, 138
Write (option)	7, 29	<code>__ccool_aux_key:N</code>	17, 142
<code>_</code>	353, 354	<code>__ccool_aux_key:n</code>	13, 20
A		<code>__ccool_aux_key:w</code>	9, 15
<code>\AtEndDocument</code>	67	<code>\g__ccool_aux_key_seq</code>	11, 19, 123, 143, 239, 240
B		<code>\g__ccool_aux_keyval_seq</code> 139, 140, 142	
<code>\begingroup</code>	152, 281	<code>__ccool_aux_outer:n</code>	24, 125
bool commands:		<code>__ccool_aux_outer_set:n</code>	22, 124
<code>\bool_gset_false:N</code>	72	<code>\g__ccool_aux_prop</code>	26, 29, 46, 141
		<code>__ccool_aux_prop:N</code>	44, 140
		<code>__ccool_aux_prop:n</code>	40, 50

\iow_now:Nn	96	\prop_new:N	26, 253
\iow_open:Nn	78	\ProvideDocumentCommand	269
K			
keys commands:			
\l_keys_choice_tl	291		
\keys_define:nn	288		
\keys_set:nn	276		
M			
msg commands:			
\msg_error:nnn	99, 200		
\msg_new:nnn	185		
\msg_warning:nnn	192, 210, 218		
N			
\NewDocumentCommand			
.....	2, 9, 263, 268, 273, 364, 371		
O			
options:			
Expans	6, 28		
File	6, 28		
Inner	6, 28		
Outer	7, 29		
Param	7, 28		
Separ	7, 29		
Write	7, 29		
P			
prop commands:			
\prop_clear_new:N	235		
\prop_gclear_new:N	46		
\prop_gput:Nnn	29, 224		
\prop_if_exist:NTF	244		
\prop_item:Nn	224, 248, 259		
\prop_map_function:NN	226		
Q			
quark commands:			
\q_stop	9, 15, 33, 42		
R			
\Real	1, 5		
\Reel	1		
S			
seq commands:			
\seq_gclear_new:N	19, 55		
\seq_gput_right:Nn	11, 259		
\seq_if_empty:NTF	47		
\seq_map_function:NN			
.....	20, 50, 119, 240, 261		
\seq_set_from_clist:Nn	139, 239		
T			
tl commands:			
\c_empty_tl ..	48, 136, 156, 178, 268, 285		
\tl_gset:Nn	77, 188, 196, 206, 295		
\tl_gset_eq:NN	291		
\tl_log:n	84, 97		
\tl_new:N	32, 74, 91		
\tl_trim_spaces:n	11, 37, 38		
U			
use commands:			
\use:N	36, 373		
\usepackage	5, 9		
X			
\X	4		

Part IV

Implementation

1 Opening

```

1 <*package>
2 <@@=ccool>
3 \ExplSyntaxOn

```

2 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_val:Nn

```

#1 : < seq >
#2 : < tl var name >

53 \cs_new_protected:Nn \_ccool\_aux\_val:Nn
54 {
55   \seq_gclear_new:N \g__ccool\_aux\_val\_seq
56   \_ccool\_seq\_from\_prop:NNn \g__ccool\_aux\_val\_seq #1 { \_ccool\_prop\_name:n{#2} }
57 }

```

(End definition for _ccool_aux_val:Nn.)

3 lambda

```

\__ccool_lambda:nn [7]
58 \cs_new_protected:Npn \__ccool_lambda:nn #1 #2
59 {
60   \exp_args:NNx
61   \DeclareDocumentCommand \__ccool_lambda_expression
62     {#1}
63     {#2}
64     \__ccool_lambda_expression
65 }

(End definition for \__ccool_lambda:nn.)

```

4 log

```

\__ccool_log_close:
66 \iow_new:N \g__ccool_log_iow
67 \AtEndDocument{\iow_close:N \g__ccool_log_iow}
68 \bool_set_false:N \g__ccool_log_open_bool
69 \cs_new_protected:Nn \__ccool_log_close:
70 {
71   \iow_close:N \g__ccool_log_iow
72   \bool_gset_false:N \g__ccool_log_open_bool
73 }

(End definition for \__ccool_log_close:.)

```

```

\__ccool_log_open:
74 \tl_new:N \g__ccool_log_file_tl
75 \cs_new_protected:Nn \__ccool_log_open:
76 {
77   \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
78   \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
79   \bool_gset_true:N \g__ccool_log_open_bool
80 }

(End definition for \__ccool_log_open:.)

```

```

\__ccool_log_read:n #1: <path>
81 \cs_new_protected:Nn \__ccool_log_read:n
82 {
83   \file_input:n{#1}
84   \tl_log:n{read~from~#1}
85 }
86 \cs_generate_variant:Nn \__ccool_log_read:n { e }

(End definition for \__ccool_log_read:n.)

```

```

\__ccool_log_read:
87 \cs_new_protected:Nn \__ccool_log_read:
88 {
89   \__ccool_log_read:e{\g__ccool_log_to_tl}
90 }

```


(End definition for _ccool_log_read:.)

_ccool_log_write:n

```

91 \tl_new:N \g__ccool_log_to_tl
92 \cs_new_protected:Nn \_ccool_log_write:n
93 {
94   \bool_if:nTF{ \g__ccool_log_open_bool }
95   {
96     \iow_now:Nn \g__ccool_log_iow {#1}
97     \tl_log:n{ write-to~#1 }
98   }
99   { \msg_error:nnn{ __ccool }{ iow }{ \g__ccool_log_iow } }
100 }
101 \cs_generate_variant:Nn \_ccool_log_write:n { e }

```

(End definition for _ccool_log_write:n.)

5 make_key

_ccool_make_key:Nn #1 : $\langle token \rangle$
 #2 : $\langle key \rangle$

```

102 \cs_new_protected:Nn \_ccool_make_key:Nn
103 {
104   \exp_args:NNx
105   \DeclareDocumentCommand{#1}
106   { D<>{ \g__ccool_option_param_tl } }
107   {
108     \_ccool_prop_item:nn{##1}{#2}
109   }
110 }
111 \cs_generate_variant:Nn \_ccool_make_key:Nn { c }

```

(End definition for _ccool_make_key:Nn.)

_ccool_make_key:n #1 : $\langle key \rangle$

```

112 \cs_new_protected:Nn \_ccool_make_key:n
113 {
114   \_ccool_make_key:cn{#1}{#1}
115 }
116 \cs_generate_variant:Nn \_ccool_make_key:n { e }

```

(End definition for _ccool_make_key:n.)

_ccool_make_key:N #1 : $\langle seq \rangle$

```

117 \cs_new_protected:Nn \_ccool_make_key:N
118 {
119   \seq_map_function:NN #1 \_ccool_make_key:e
120 }

```

(End definition for _ccool_make_key:N.)

6 make_ccool

_ccool_make_ccool_exp:nnn

```

121 \cs_new_protected:Nn \_ccool_make_ccool_exp:nnn
122 {
123   \_ccool_aux_val:Nn \g\_ccool_aux_key_seq {#1}
124   \_ccool_aux_outer_set:n{#3}
125   \_ccool_aux_outer:n
126   {
127     \exp_args:Nnf
128     \erw_seq_use:Nn
129     \g\_ccool_aux_val_seq
130     {#2}
131   }
132 }

```

(End definition for _ccool_make_ccool_exp:nnn.)

_ccool_make_ccool_key:nnn

```

133 \cs_new_protected:Nn \_ccool_make_ccool_key:nnn
134 {
135   \_ccool_prop_if_exist:nTF{#1}
136   { \c_empty_tl }
137   { \_ccool_prop_new:n{#1} }
138   \exp_args:No \_ccool_aux_inner_set:n{#2}
139   \seq_set_from_clist:Nn \g\_ccool_aux_keyval_seq {#3}
140   \_ccool_aux_prop:N \g\_ccool_aux_keyval_seq
141   \_ccool_prop_append:Nn \g\_ccool_aux_prop {#1}
142   \_ccool_aux_key:N \g\_ccool_aux_keyval_seq
143   \_ccool_make_key:N \g\_ccool_aux_key_seq
144 }

```

(End definition for _ccool_make_ccool_key:nnn.)

_ccool_make_ccool_sideeffect:nnn [9]

```

145 \cs_new_protected:Nn \_ccool_make_ccool_sideeffect:nnn
146 {
147   \_ccool_make_ccool_key:nnn{#1}{#2}{#3}
148   \bool_if:nTF{ \g\_ccool_log_open_bool }
149   {
150     \_ccool_log_write:n
151     {
152       \begingroup
153       \def \_ccool_log_entry { \Ccool<#1>c{#2}{#3} } \expandafter
154       \endgroup \_ccool_log_entry
155     }
156   }{\c_empty_tl}
157 }

```

(End definition for _ccool_make_ccool_sideeffect:nnn.)

_ccool_make_ccool:nnnn #1 : $\langle \text{token list} \rangle$
 #2 : $\langle \text{seq}_1 \rangle$
 #3 : $\langle \text{seq}_2 \rangle$

```

#4 : < prop >

158 \cs_new_protected:Npn \__ccool_make_ccool:nnnn #1 #2 #3 #4
159 {
160   \exp_args:NNx \DeclareDocumentCommand \Ccool
161   {%^~A      2      3      4 5 6      7 8      9
162   +o D<>{#1} E{ c }{{#2}} m t+ s E{ s c }{{#3}{#4}} +o
163   }
164   {
165     \IfValueT{##1}{##1}
166     \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
167     \IfBooleanT{##6}
168     {
169       \__ccool_make_ccool_exp:nnn{##2}{##7}{##8}
170     }
171     \bool_if:nTF{##5}
172     {
173       \gappto{\CcoolHook}
174       {
175         \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
176       }
177     }
178     {\c_empty_tl}
179     \IfValueT{##9}
180     {
181       \exp_not:n{ \Ccool[##9] }
182     }
183   }
184 }

```

(End definition for __ccool_make_ccool:nnnn.)

7 msg

```

185 \msg_new:nnn {__ccool}{ iow }{#1~is~closed~can't~write}

```

8 option

__ccool_option_inner:n #1 : <code>

```

186 \cs_new_protected:Nn \__ccool_option_inner:n
187 {
188   \tl_gset:Nn \g__ccool_option_inner_tl {#1}
189 }
190 \__ccool_option_inner:n
191 {
192   \msg_warning:nnn{ erw }{ csnset }{ \exp_not:N \g__ccool_option_inner_tl }
193 }

```

(End definition for __ccool_option_inner:n.)

__ccool_option_param:n #1 : <token list>

```

194 \cs_new:Nn \__ccool_option_param:n
195 {
196   \tl_gset:Nn \g__ccool_option_param_tl{#1}

```

```

197 }
198 \__ccool_option_param:n
199 {
200   \msg_error:nnx{ __ccool }
201   { generic }
202   { \exp_not:N\g__ccool_option_param_tl-undefined }
203 }

```

(End definition for __ccool_option_param:n.)

```

\__ccool_option_outer:n #1: < inline code >
204 \cs_new_protected:Nn \__ccool_option_outer:n
205 {
206   \tl_gset:Nn \g__ccool_option_outer_tl {#1}
207 }
208 \__ccool_option_outer:n
209 {
210   \msg_warning:nnn{ erw }{ csnset }{ \exp_not:N \g__ccool_option_outer_tl }
211 }

```

(End definition for __ccool_option_outer:n.)

```

\__ccool_option_separ:n #1: {< tl1>}{< tl2>}{< tl3>}
212 \cs_new_protected:Nn \__ccool_option_separ:n
213 {
214   \cs_gset:Npn \g__ccool_option_separ_tl {#1}
215 }
216 \__ccool_option_separ:n
217 {
218   \msg_warning:nnn{ erw }{ csnset }{ \exp_not:N \g__ccool_option_separ_tl }
219 }

```

(End definition for __ccool_option_separ:n.)

9 prop

```

\__ccool_prop_append:NN #1: < prop1 >
#2: < prop2 >
220 \cs_new_protected:Npn \__ccool_prop_append:NN #1 #2
221 {
222   \cs_set:Nn \__ccool_prop_append:nn
223   {
224     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
225   }
226   \prop_map_function:NN #2 \__ccool_prop_append:nn
227 }
228 \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 >
229 \cs_new_protected:Nn \__ccool_prop_append:Nn
230 {
231   \__ccool_prop_append:cN{ \__ccool_prop_name:n {#2} } #1
232 }

```

(End definition for __ccool_prop_append:Nn.)

```

\__ccool_prop_clear_new:n #1 : < tl var name >
233 \cs_new_protected:Nn \__ccool_prop_clear_new:n
234 {
235   \exp_args:No \prop_clear_new:c{ \__ccool_prop_name:n {#1} }
236 }

```

(End definition for __ccool_prop_clear_new:n.)

```

\__ccool_prop_clear_new_map:n #1 : < keyval list >
237 \cs_new_protected:Nn \__ccool_prop_clear_new_map:n
238 {
239   \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
240   \seq_map_function:NN \g__ccool_aux_key_seq \__ccool_prop_clear_new:n
241 }

```

(End definition for __ccool_prop_clear_new_map:n.)

```

\__ccool_prop_if_exist:nTF #1 : < tl1 >
#2 : < tl2 >
#3 : < tl3 >
242 \cs_new:Nn \__ccool_prop_if_exist:nTF
243 {
244   \prop_if_exist:cTF{ \__ccool_prop_name:n {#1} }{#2}{#3}
245 }

```

(End definition for __ccool_prop_if_exist:nTF.)

```

\__ccool_prop_item:nn #1 : < tl var name >
#2 : < key >
246 \cs_new:Nn \__ccool_prop_item:nn
247 {
248   \prop_item:cn { \__ccool_prop_name:n {#1} } {#2}
249 }

```

(End definition for __ccool_prop_item:nn.)

```

\__ccool_prop_name:n #1 : < tl var name >
250 \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 >
251 \cs_new_protected:Nn \__ccool_prop_new:n
252 {
253   \prop_new:c{ \__ccool_prop_name:n {#1} }
254 }

```

(End definition for __ccool_prop_new:n.)

10 seq

```

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

255 \cs_new_protected:Nn \__ccool_seq_from_prop:NNn
256 {
257   \cs_set_protected:Nn \__ccool_seq_from_prop:n
258   {
259     \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
260   }
261   \seq_map_function:NN #2 \__ccool_seq_from_prop:n
262 }

(End definition for \__ccool_seq_from_prop:NNn.)

```

11 Front-end

\CcoolClear

```

263 \NewDocumentCommand{ \CcoolClear }
264 { D<>{\g__ccool_option_param_tl} }
265 {
266   \__ccool_prop_clear_new_map:n{#1}
267 }

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

```

\CcoolHook

```

268 \NewDocumentCommand{\CcoolHook}{\c_empty_tl}

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

```

\CcoolLambda

```

269 \ProvideDocumentCommand \CcoolLambda { O{m} m }
270 {
271   \__ccool_lambda:nn { #1 } { #2 }
272 }

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

```

\CcoolOption

```

273 \NewDocumentCommand{ \CcoolOption }
274 { m }
275 {
276   \keys_set:nn{ __ccool }{#1}
277   %%^A \bool_if:nTF{ \g__ccool_log_open_bool }
278   %%^A {
279     %%^A \__ccool_log_write:n
280     %%^A {

```

```

281 %^^A      \begingroup
282 %^^A      \def \__ccool_log_entry { \CcoolOption{ #1 } \expandafter
283 %^^A      \endgroup \__ccool_log_entry
284 %^^A      }
285 %^^A      }{\c_empty_tl}
286 %^^A      }
287 }

```

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

```

288 \keys_define:nn { __ccool }
289 {

```

Expans

```

290 Expans .multichoices:nn = { eo, ee, ex, xo, xe, xx }
291 { \tl_gset_eq:NN \g__ccool_option_expans_tl \l_keys_choice_tl },
292 Expans .default:n = { xo },
293 Expans .initial:n = { xo },

```

File

```

294 File .code:n = {
295   \tl_gset:Nx \g__ccool_log_file_tl{#1}
296 },
297 File .default:n = { \erw_sys_jobnametimestamp: },
298 File .initial:n = { \erw_sys_jobnametimestamp: },

```

Inner

```

299 Inner .code:n={
300   \__ccool_option_inner:n{#1}
301   \exp_last_unbraced:Nf
302   \__ccool_make_ccool:nnnn
303   {
304     { \g__ccool_option_param_tl }
305     { \g__ccool_option_inner_tl }
306     { \g__ccool_option_separ_tl }
307     { \g__ccool_option_outer_tl }
308   }
309 },
310 Inner .value_required:n = false,
311 Inner .default:n = {###1},
312 Inner .initial:n = {###1},

```

Param

```

313 Param .code:n={
314   \__ccool_option_param:n{#1}
315   \exp_last_unbraced:Nf
316   \__ccool_make_ccool:nnnn
317   {
318     { \g__ccool_option_param_tl }
319     { \g__ccool_option_inner_tl }
320     { \g__ccool_option_separ_tl }
321     { \g__ccool_option_outer_tl }
322   }
323 },
324 Param .value_required:n = false,
325 Param .default:n = { Default },
326 Param .initial:n = { Default },

```

Outer

```

327 Outer .code:n={
328   \__ccool_option_outer:n{#1}
329   \exp_last_unbraced:Nf
330   \__ccool_make_ccool:nnnn
331   {
332     { \g__ccool_option_param_tl }
333     { \g__ccool_option_inner_tl }
334     { \g__ccool_option_separ_tl }
335     { \g__ccool_option_outer_tl }
336   }
337 },
338 Outer .value_required:n = false,
339 Outer .default:n = { \ensuremath{####1} },
340 Outer .initial:n = { \ensuremath{####1} },

```

Separ

```

341 Separ .code:n={
342   \__ccool_option_separ:n{#1}
343   \exp_last_unbraced:Nf
344   \__ccool_make_ccool:nnnn
345   {
346     { \g__ccool_option_param_tl }
347     { \g__ccool_option_inner_tl }
348     { \g__ccool_option_separ_tl }
349     { \g__ccool_option_outer_tl }
350   }
351 },
352 Separ .value_required:n = false,
353 Separ .default:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
354 Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },

```

Write

```

355 Write .code:n = {
356   \bool_if:nTF{#1}
357   {\__ccool_log_open:}
358   {\__ccool_log_close:}
359 },
360 Write .value_required:n = false,
361 Write .default:n = \BooleanFalse,
362 Write .initial:n = \BooleanFalse
363 }

```

\CcoolRead

```

364 \NewDocumentCommand{\CcoolRead}
365 {o}
366 {
367   \IfValueTF{#1}
368   {\__ccool_log_read:e{#1}}
369   {\__ccool_log_read:}
370 }

```


(End definition for \CcoolRead. This function is documented on page 7.)

\CcoolVers

```
371 \NewDocumentCommand{\CcoolVers}  
372 {}  
373 {\use:c{ver@ccool.sty}}
```

(End definition for \CcoolVers. This function is documented on page 7.)

12 Closing

```
374 \ExplSyntaxOff  
375 \</package>
```