

# The ccool package<sup>\*</sup>

Erwann Rogard<sup>†</sup>

Released 2020/04/16

## Abstract

The package `ccool` for  $\text{\LaTeX}$  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  $\text{\LaTeX}$  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

<b>I</b>	<b>Usage</b>	<b>4</b>
<b>0</b>	<b>Convention</b>	<b>4</b>
<b>1</b>	<b>Loading the package</b>	<b>5</b>

---

<sup>\*</sup>This file describes version v2.4, last revised 2020/04/16.

<sup>†</sup>firstname dot lastname AusTria gmail dot com

<b>2</b>	<b>\Ccool</b>	<b>5</b>
2.1	Core feature . . . . .	5
2.2	Process the $val_i$ 's . . . . .	5
2.3	Append to a hook . . . . .	5
2.4	Expand the $val_i$ 's . . . . .	5
2.5	Head . . . . .	5
2.6	Tail . . . . .	6
2.7	Parameterize the $key_i$ 's . . . . .	6
2.8	Write . . . . .	6
<b>3</b>	<b>\CcoolClear</b>	<b>6</b>
<b>4</b>	<b>\CcoolHook</b>	<b>6</b>
<b>5</b>	<b>\CcoolLambda</b>	<b>6</b>
<b>6</b>	<b>\CcoolOption</b>	<b>6</b>
6.1	Expans . . . . .	6
6.2	File . . . . .	6
6.3	Inner . . . . .	6
6.4	Param . . . . .	7
6.5	Outer . . . . .	7
6.6	Separ . . . . .	7
6.7	Write . . . . .	7
<b>7</b>	<b>\CcoolRead</b>	<b>7</b>
<b>8</b>	<b>\CcoolVers</b>	<b>7</b>
<b>9</b>	<b>Do's and dont's</b>	<b>7</b>
<b>II</b>	<b>Listing</b>	<b>9</b>
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 fonctionelle	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	<code>aux</code>	19
3	<code>lambda</code>	21
4	<code>log</code>	22
5	<code>make_key</code>	23
6	<code>make_ccool</code>	23
7	<code>msg</code>	25
8	<code>option</code>	25
9	<code>prop</code>	26
10	<code>seq</code>	27
11	<code>sys</code>	28
12	<b>Front-end</b>	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
	12.5 <code>Expans</code> . . . . .	30
	12.6 <code>File</code> . . . . .	30
	12.7 <code>Inner</code> . . . . .	30
	12.8 <code>Param</code> . . . . .	30
	12.9 <code>Outer</code> . . . . .	30
	12.10 <code>Separ</code> . . . . .	31
	12.11 <code>Write</code> . . . . .	31
	12.12 <code>\CcoolRead</code> . . . . .	31
	12.13 <code>\CcoolVers</code> . . . . .	31
13	<b>Closing</b>	32

## 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)  $\backslash X \rightarrow Y$ :  $\backslash X$  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 L<sup>A</sup>T<sub>E</sub>X engine. See [Part III, section 5](#).
2. Put in the *preamble*

---

<code>\Ccool</code>	<code>\Ccool[⟨tl<sub>1</sub>⟩]&lt;⟨tl<sub>2</sub>⟩&gt;c{⟨code<sub>1</sub>⟩}{⟨kv<sub>1</sub>⟩}+*s{⟨separators⟩}c{⟨code<sub>2</sub>⟩}[⟨tl<sub>6</sub>⟩]</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 `\⟨keyi⟩`, according to the following algorithm:

- 1)  $\langle val_i \rangle \leftarrow \text{\function}\{\langle val_i \rangle\}$
- 2) Creates `\⟨keyi⟩`, such that  $\text{\⟨key<sub>i</sub>⟩} \rightarrow \langle val_i \rangle$ ,

where `\function` is controled by option [Inner](#). For instance, the side effect of `\Ccool{ Real = \mathbb{R} }` is `\Real  $\rightarrow$  \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 `\⟨keyi⟩` 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*<sub>*i*</sub>'s

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

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
```

---

2020/04/16 v2.4 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--->
\CoolStyle>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, 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{\ \& \ }{, \ }{\ \& \ }[\]
%^A<---
\CoolClear
```

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

#### 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<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear

```

---

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

#### Listing 9. Listing 8 read from file

```

%~^A<--->
\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
%~^A<---
\CcoolClear

```

---

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

#### Listing 10. Probability space

```

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

```

---

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

Listing 11. Listing 10 read from file

```
%^A--->
\CoolRead \tab $\Omega$ $\Field$ $\Meas$
%^A<---
\CoolClear
```

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

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

```
\CoolOption{ Write = \BooleanTrue }
%^A--->
\newtheorem{theorem}{Theorem}
\AfterEndEnvironment{theorem}{\CoolHook}
\Cool 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] } }+
  [$n\in\mathbb{N}$,$\mathbb{R}$ eine offene Menge und
  $f\in C^1(\mathbb{R}^n,\mathbb{R})$].
  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*}
    \text{Steig} = \text{grad } f(\text{Punkt})
  \end{equation*}
  \end{theorem}]
{}
(Check: $N$, $Punkt$)
%^A<---
\CoolOption{ Write = \BooleanFalse }
\CoolClear
```

**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:  $\mathbb{N}$ ,  $\xi$ )

Listing 13. Listing 12 read from file

```
%^A--->
\CoolRead \tab $N$ $R$ $\mathbb{R}$ $\mathbb{R}$ $\mathbb{R}$
%^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$;
\$ \LogWaldInf_{\Time} \EvalAt{ \Scale } = \inf_{0 \le s \le \Time}
\Scale \Process_{\Time} \EvalAt{ \Scale }$.}
\item{\$ \CUSUM_{\Time} \EvalAt{ \Scale } = \LogWaldInf_{\Time} \EvalAt{
\Scale } - \LogWald_{\Time} \EvalAt{ \Scale } \ge 0$, which is the CUSUM
statistic process.}
\item{\$ \CUSUMst \EvalAt{ \Scale, \LogWaldInf } = \inf \left[ \Time \ge
0 \text{ \textit{SuchThat} } \CUSUM_{\Time} \EvalAt{ \Scale } \ge \LogWaldInf \right]$,
which is the CUSUM stopping time.}
\end{enumerate} \end{definition} \par]{}

(Check: \$ \Scale$, \$ \CUSUM$)
% ^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:  $\lambda, y$ )

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<sup>A</sup>T<sub>E</sub>Xcommunity[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<sup>A</sup>T<sub>E</sub>Xengine

### 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<sup>A</sup>T<sub>E</sub>X3 Project Team *The L<sup>A</sup>T<sub>E</sub>X3 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<sup>A</sup>T<sub>E</sub>X3 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<sup>A</sup>T<sub>E</sub>X 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>&lt;kvl_2&gt;</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>&lt;\langle tl_2 \rangle&gt;</code> . . . . .	18	Add: Listing 1 . . . . .	18
v1.4		v1.9	
General: Add: section 9 . . . . .	18	General: Add: support for Lua <sub>T</sub> <sub>E</sub> <sub>X</sub> . . . . .	18
Add: <code>\OpsDebug</code> . . . . .	18	Move: from Part I to Part IV, what is now that part’s section 12 . . . . .	18
Add: <code>\OpsHook</code> . . . . .	18	v2.0	
Add: <code>\Expans</code> (for debugging’ sake, but...) . . . . .	18	General: Add: support for X <sub>T</sub> <sub>E</sub> <sub>X</sub> . . . . .	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	v2.3	General: Add: Listing 15, Listing 16, and Listing 17 . . . . .	18
v2.1			Complete: Listing 14 . . . . .	18
General: Add: Listings 3, 4, 5, 6, 7, 8, and 9 . . . . .	18		Rearranged: <code>\Ccool</code> 's subsections. Previously, by argument. Now, by feature. . . . .	18
Replace: <code>\CcoolLambda</code> 's optional integer argument (number of m's) by a standard argument list . . . . .	18		Remove: Listing showing part of the preamble . . . . .	18
Replace: $\langle tl_2 \rangle$ 's position within <code>\Ccool</code> 's argument list, from first to second. Greater versatility . . . . .	18		Replace: for <code>\Ccool, i{}</code> by <code>c{}</code> . . . . .	18
Replace: Name by <b>Param</b> . . . . .	18		Replace: In step 2), the created command's implementation, from <code>\ProvideDocumentCommand</code> to <code>\DeclareDocumentCommand</code> . . . . .	18
Replace: as the de- fault of <b>Param</b> , <b>Math by Default</b> . . . . .	18	v2.4		
v2.2			General: Fix: minor error in the listings ( <code>\Real</code> rather than <code>\Reel</code> , hitherto unnoticed). . . . .	18
General: Remove: % from listings . . . . .	18		Remove: examples from <b>Part I</b> , section 2, as redundant with <b>Part II</b> Listing 2-6 . . . . .	18
Replace: part of the abstract's with more straightforward descriptions based on input from forum participants . . . . .	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.

<b>Symbols</b>		<code>\bool_set_false:N</code> . . . . .	94
<code>\langle key_i \langle tl_2 \rangle \rangle</code> . . . . .	6	<code>\BooleanFalse</code> . . . . .	429, 430
<code>\langle key_i \rangle</code> . . . . .	5, 7	<code>\BooleanTrue</code> . . . . .	6
<code>\langle key_i \rangle</code> . . . . .	5		
Expans (option) . . . . .	6, 30	<b>C</b>	
File (option) . . . . .	6, 30	<code>\Ccool</code> . . . . .	1, 2, 5, 6, 6, 6, 9, 18, 179, 186, 207
Inner (option) . . . . .	6, 30	ccool internal commands:	
Outer (option) . . . . .	7, 30	<code>__ccool_aux_inner:n</code> . . . . .	5, 6, 37
Param (option) . . . . .	7, 30	<code>__ccool_aux_inner_set:n</code> . . . . .	3, 164
Separ (option) . . . . .	7, 31	<code>__ccool_aux_key:N</code> . . . . .	16, 168
Write (option) . . . . .	7, 31	<code>__ccool_aux_key:n</code> . . . . .	12, 19
		<code>__ccool_aux_key:w</code> . . . . .	8, 14
		<code>g_ccool_aux_key_seq</code> . . . . .	
<code>\sqcup</code> . . . . .	421, 422	. . . . .	10, 18, 149, 169, 270, 271
<b>A</b>		<code>g_ccool_aux_keyval_seq</code> . . . . .	165, 166, 168
<code>\AtEndDocument</code> . . . . .	93	<code>__ccool_aux_outer:n</code> . . . . .	23, 151
		<code>__ccool_aux_outer_set:n</code> . . . . .	21, 150
<b>B</b>		<code>g_ccool_aux_prop</code> . . . . .	25, 28, 45, 167
<code>\begin</code> . . . . .	294	<code>__ccool_aux_prop:N</code> . . . . .	43, 166
<code>\begingroup</code> . . . . .	178, 349	<code>__ccool_aux_prop:n</code> . . . . .	39, 49
bool commands:		<code>__ccool_aux_prop:nn</code> . . . . .	25
<code>\bool_gset_false:N</code> . . . . .	98	<code>__ccool_aux_prop:w</code> . . . . .	31, 41
<code>\bool_gset_true:N</code> . . . . .	105	<code>__ccool_aux_separ:n</code> . . . . .	75, 302
<code>\bool_if:nTF</code> . . . . .	120, 174, 197, 345, 424	<code>__ccool_aux_separ:nn</code> . . . . .	52, 77
		<code>__ccool_aux_val:Nn</code> . . . . .	79, 149





# 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   \DeclareDocumentCommand{#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>c{#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 : < seq<sub>1</sub> >  
 #3 : < seq<sub>2</sub> >  
 #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{ c }{#{2}} m t+ s E{ s c }{#{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:NNn.)

\\_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**

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

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

**\CcoolHook**

```
336 \NewDocumentCommand{\CcoolHook}{\c_empty_tl}
```

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

**\CcoolLambda**

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

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

**\CcoolOption**

```
341 \NewDocumentCommand{ \CcoolOption }
342 { m }
343 {
344   \keys_set:nn{ __ccool }{#1}
345   %%^A \bool_if:nTF{ \g__ccool_log_open_bool }
346   %%^A {
347     %%^A \__ccool_log_write:n
348     %%^A {
349       %%^A \beginingroup
350       %%^A \def \__ccool_log_entry { \CcoolOption{ #1 } \expandafter
351       %%^A \endgroup \__ccool_log_entry
352       %%^A }
353       %%^A }{\c_empty_tl}
354       %%^A }
355 }
```



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

```
356 \keys_define:nn { __ccool }
357 {
```

#### Expans

```
358 Expans .multichoices:nn = { eo, ee, ex, xo, xe, xx }
359 { \tl_gset_eq:NN \g__ccool_option_expans_tl \l_keys_choice_tl },
360 Expans .default:n = { xo },
361 Expans .initial:n = { xo },
```

#### File

```
362 File .code:n = {
363   \tl_gset:Nx \g__ccool_log_file_tl{#1}
364 },
365 File .default:n = { \__ccool_sys_filename: },
366 File .initial:n = { \__ccool_sys_filename: },
```

#### Inner

```
367 Inner .code:n={
368   \__ccool_option_inner:n{#1}
369   \exp_last_unbraced:Nf
370   \__ccool_make_ccool:nnnn
371   {
372     { \g__ccool_option_param_tl }
373     { \g__ccool_option_inner_tl }
374     { \g__ccool_option_separ_tl }
375     { \g__ccool_option_outer_tl }
376   }
377 },
378 Inner .value_required:n = false,
379 Inner .default:n = {###1},
380 Inner .initial:n = {###1},
```

#### Param

```
381 Param .code:n={
382   \__ccool_option_param:n{#1}
383   \exp_last_unbraced:Nf
384   \__ccool_make_ccool:nnnn
385   {
386     { \g__ccool_option_param_tl }
387     { \g__ccool_option_inner_tl }
388     { \g__ccool_option_separ_tl }
389     { \g__ccool_option_outer_tl }
390   }
391 },
392 Param .value_required:n = false,
393 Param .default:n = { Default },
394 Param .initial:n = { Default },
```

#### Outer

```
395 Outer .code:n={
396   \__ccool_option_outer:n{#1}
397   \exp_last_unbraced:Nf
398   \__ccool_make_ccool:nnnn
399   {
```

```

400     { \g__ccool_option_param_tl }
401     { \g__ccool_option_inner_tl }
402     { \g__ccool_option_separ_tl }
403     { \g__ccool_option_outer_tl }
404   }
405 },
406 Outer .value_required:n = false,
407 Outer .default:n = { \ensuremath{####1} },
408 Outer .initial:n = { \ensuremath{####1} },

```

**Separ**

```

409 Separ .code:n={
410   \__ccool_option_separ:n{#1}
411   \exp_last_unbraced:Nf
412   \__ccool_make_ccool:nnnn
413   {
414     { \g__ccool_option_param_tl }
415     { \g__ccool_option_inner_tl }
416     { \g__ccool_option_separ_tl }
417     { \g__ccool_option_outer_tl }
418   }
419 },
420 Separ .value_required:n = false,
421 Separ .default:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
422 Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },

```

**Write**

```

423 Write .code:n = {
424   \bool_if:nTF{#1}
425   {\__ccool_log_open:}
426   {\__ccool_log_close:}
427 },
428 Write .value_required:n = false,
429 Write .default:n = \BooleanFalse,
430 Write .initial:n = \BooleanFalse
431 }

```

**\CcoolRead**

```

432 \NewDocumentCommand{\CcoolRead}
433 {o}
434 {
435   \IfValueTF{#1}
436   {\__ccool_log_read:e{#1}}
437   {\__ccool_log_read:}
438 }

```

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

**\CcoolVers**

```

439 \NewDocumentCommand{\CcoolVers}
440 {}
441 {\use:c{ver@ccool.sty}}

```

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

## 13 Closing

442 \ExplSyntaxOff