

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

Erwann Rogard<sup>†</sup>

Released 2020/06/04

## Abstract

The package `ccool` for L<sup>A</sup>T<sub>E</sub>X is a *key-value* interface, `\Ccool`, on top of `xparse`'s document command parser. Global options control input processing and its expansion. By default, they are set to meet likely requirements, depending on context: the selected language, and which of text and math mode is active. These options can be overridden inline. This versatility could find its use, for example, 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 L<sup>A</sup>T<sub>E</sub>X met à disposition une interface de type *clé-valeur*, `\Ccool`, destinée à faciliter la génération de commandes. Les paramètres globaux contrôlant le traitement de ces *clé-valeur* sont fixés par défaut pour répondre aux besoins courants, suivant le contexte (langage, mode textuel ou mathématique). Un exemple d'application, est 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 v3.0, last revised 2020/06/04.

<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 <i>val<sub>i</sub></i> s . . . . .	5
2.3	Append to a hook . . . . .	5
2.4	Expand the <i>val<sub>i</sub></i> s . . . . .	5
2.5	Head . . . . .	6
2.6	Tail . . . . .	6
2.7	Parameterize the <i>key<sub>i</sub></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	And . . . . .	6
6.2	Expans . . . . .	7
6.4	Inner . . . . .	7
6.5	Param . . . . .	7
6.6	Outer . . . . .	7
6.7	Separ . . . . .	7
6.8	Write . . . . .	7
<b>7</b>	<b>\CcoolRead</b>	<b>8</b>
<b>8</b>	<b>\CcoolVers</b>	<b>8</b>
<b>9</b>	<b>Do's and dont's</b>	<b>8</b>
<b>II</b>	<b>Listing</b>	<b>10</b>
1.	\CcoolVers	10
2.	Let $\mathbb{N}$ and $\mathbb{R}$ denote...start of the tutorial)	10
3.	Equivalent to 2, with \NewDocumentCommand	10
4.	Equivalent to 3, with \Ccool	10
5.	Equivalent to 4, with expansion	10
6.	Equivalent to 4, parameterized (end of the tutorial)	10
7.	Language and mode	11
8.	Separators	11
9.	Hello, world! (testing)	12

10. Listing 9 read from file	12
11. Probability space	13
12. Listing 11 read from file	13
13. Mittelwertsatz für $n$ variable.	13
14. Listing 13 read from file	14
15. Families of polynomial functions	14
16. Listing 15 read from file	15
17. Same as Listing 15, but arbitrary number system	15
16. Listing 17 read from file	15
19. Fonction et fonctionnelle	15
20. Listing 19 read from file	16
21. CUSUM statistic.	16
22. Listing 21 read from file	17
 <b>III Other</b>	 <b>18</b>
1 Acknowledgment	18
2 Genealogy	18
3 Install	18
4 Issue	18
5 Support	18
6 Testing	18
6.1 Technicality . . . . .	18
6.2 Platform . . . . .	19
6.3 Engine . . . . .	19
6.4 Results . . . . .	19
6.5 Other . . . . .	19
7 To do	19
8 References	19
 <b>Change History</b>	 <b>20</b>

<b>Index</b>	<b>22</b>
<b>IV Implementation</b>	<b>25</b>
1 Opening	25
2 aux	25
3 lang	27
4 log	28
5 make_key	29
6 make_ccool	30
7 msg	31
8 option	32
9 prop	33
10 seq	34
11 Front-end	35
11.1 \CcoolClear . . . . .	35
11.2 \CcoolHook . . . . .	35
11.3 \CcoolLambda . . . . .	35
11.4 \CcoolOption . . . . .	35
11.4.1 And . . . . .	35
11.4.2 Expans . . . . .	35
11.4.3 File . . . . .	36
11.4.4 Inner . . . . .	36
11.4.5 Param . . . . .	36
11.4.6 Outer . . . . .	36
11.4.7 Separ . . . . .	37
11.4.8 Write . . . . .	37
11.5 \CcoolRead . . . . .	37
11.6 \CcoolVers . . . . .	37
12 Closing	37

## Part I

# Usage

## Convention

a) Loosely, those of [2], for example as to the meaning of  $\langle token\ list \rangle$ .

- b) Those of [5], 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**.

---

`\usepackage`    `\usepackage{ccool}`

---

### 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*

---

`\Ccool`    `\Ccool[⟨tl1⟩]⟨tl2⟩c{⟨code1⟩}{⟨kvl1⟩}+*s{⟨separators⟩}c{⟨code2⟩}[⟨tl6⟩]`  
 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{⟨kvl1⟩}` executes for each  $\langle key_i \rangle = \langle val_i \rangle$ ,

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

where `\function` is encoded in *global option* **Inner**. For instance, the side effect of `\Ccool{ Real = \mathbb{R} }` is  $\backslash Real \rightarrow \backslash mathbb{R}$ . To be sparingly used, *global option* **Expans** controls the type of expansion of  $\langle key_i \rangle$  and  $\langle val_i \rangle$ .

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

## 2.2 Process the $val_i$ s

`\Ccool c{⟨code1⟩}{⟨kvl1⟩}` 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{⟨kvl1⟩}+` 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 *vals*s

`\Ccool{<kvl>}*` supplements the **Core feature** with the expansion of the  $\langle val_i \rangle$ 's using typesetting rules encoded in *global 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>]{<kvl>}` 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{<kvl>}[<tl6>]{<kvl>}` is identical to `\Ccool{<kvl>}` followed by `\Ccool[<tl6>]{<kvl>}`.

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

## 2.7 Parameterize the *keys*

`\Ccool<tl2>{<kvl>}` is identical to the **Core feature**, except  $\langle key_i \rangle$  is replaced by  $\langle key_i < tl_2 > \rangle$ . The default value of  $\langle tl_2 \rangle$  is encoded in **Param**. In our example,  $\langle tl_2 \rangle$  could be **Style**.

## 2.8 Write

*global option* **Write** is identical to the **Core feature**, except that if **Write** is set to `\BooleanTrue`, the code is written to a file whose path is encoded in *global option* **File**.

---

<code>\CcoolClear</code>	<code>\CcoolClear&lt;tl<sub>2</sub>&gt;{...&lt;key<sub>i</sub>&gt;,...}</code>
<b>Semantics</b>	Clears all $\langle key_i < tl_2 > \rangle$ 's

---

<code>\CcoolHook</code>	<code>\CcoolHook</code>
<b>Semantics</b>	No side effect or expansion

---

<code>\CcoolLambda</code>	<code>\CcoolLambda[&lt;arg spec&gt;]{&lt;code&gt;},</code> where <i>arg spec</i> is by default an ‘ <i>o</i> ’-type <i>argument</i> .
<b>Example</b>	<code>\Ccool{ EvalAt = \CcoolLambda{(#1)} }</code>
<b>Semantics</b>	Returns a command of type <code>\DeclareDocumentCommand[5]</code> ,

<u>\CcoolOption</u>	<p><code>\CcoolOption[...⟨key<sub>i</sub>⟩ ⟨key<sub>i</sub>⟩=⟨val<sub>i</sub>⟩,...]</code></p> <p>where <math>\langle key_i \rangle</math> is either of <b>And</b>, <b>Expans</b>, <b>File</b>, <b>Inner</b>, <b>Param</b>, <b>Outer</b>, <b>Separ</b>, and <b>Write</b>.</p> <p><b>Semantics</b> Modify the behavior of <code>\Ccool</code></p>
<b>And</b>	<p>Also see <b>Part IV And</b></p> <p><b>Semantics</b> Sets the translation of <i>and</i> in language <math>\langle key \rangle</math> to <math>\langle val \rangle</math></p> <p><b>Syntax</b> <math>\langle keyval list \rangle</math></p>
<b>Expans</b>	<p>Also see <b>Core feature</b> and <b>Part IV Expans</b></p> <p><b>Syntax</b> <code>eo ee ex x x xe xx</code></p>
<b>File</b>	<p>Also see <b>Part I Write</b> and <b>Part IV File</b></p> <p><b>Syntax</b> <math>\langle path \rangle</math></p>
<b>Inner</b>	<p>Also see <b>Process the val<sub>i</sub>s</b> and <b>Part IV Inner</b></p> <p><b>Syntax</b> <math>\langle code \rangle</math>, with <b>####1</b> as the <i>placeholder</i></p>
<b>Param</b>	<p>Also see <b>Parameterize the key<sub>i</sub>s</b>, and <b>Part IV Param</b></p> <p><b>Syntax</b> <math>\langle token list \rangle</math></p>
<b>Outer</b>	<p>Also see <b>Expand the val<sub>i</sub>s</b>, and <b>Part IV Outer</b></p> <p><b>Default</b> <code>\ensuremath{####1}</code></p> <p><b>Syntax</b> <math>\langle code \rangle</math>, with <b>####1</b> as the <i>placeholder</i></p>
<b>Separ</b>	<p>Also see <b>Expand the val<sub>i</sub>s</b>; Listing 7; and <b>Part IV Separ</b></p> <p><b>Other</b> Default behavior depends on whether <code>babel</code> and <code>amsmath</code> are loaded</p> <p><b>Syntax</b> That of <i>separators</i> in [2, Section 8 of l3seq]</p>
<b>Write</b>	<p>Also see <b>Part I Write</b> and <b>Part IV Write</b></p> <p><b>Syntax</b> <code>\BooleanFalse \BooleanTrue</code></p>

---

---

`\CcoolRead`    `\CcoolRead[⟨path⟩]`

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** → the package’s version

## 9 Do’s and dont’s

1)

Don’t: `Inner=\{####1\}`

Symptom: `\CcoolRead` fails

Do: `Inner={\char‘{####1\char‘}}`

2)

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

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

3)

Don’t: `[a, b)`

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

4)

Don’t: `\cal F`.

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

5)

Don’t: `\[x_0,x\]`

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

6)

Don’t: Use ‘*d*’-type or ‘*e*’-type arguments[5] for [\CcoolLambda](#)

Do: Use only ‘*m*’-type and ‘*o*’-type arguments

7)



Don't: `\usepackage[spanish]{babel}`

Do: `\usepackage[spanish.noquoting]{babel}`[\[11\]](#)

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

## Part II

# Listing

NB:

1. Some statements 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
```

---

2020/06/04 v3.0 cool — A key-value document command parser

### 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} }*
[~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. Language and mode

```
% ^A--->
\textbf{\language}{:}\Cool{ X = x, Y = y }*
\begin{otherlanguage}{spanish}
  \CoolOption[ Separ ]\
  \textbf{\language}{:}\Cool{ X = x, Y = y }*
\end{otherlanguage}\
\textbf{\language}{:}\Cool{ X = x, Y = y }*
\\[1em]
\CoolOption[ Outer = ###1 ]
\textbf{\language}{:}\Cool{ X = this, Y = that }*
\begin{otherlanguage}{spanish}
  \CoolOption[ Separ ]\
  \textbf{\language}{:}\Cool{ X = esto, Y = aquello }*
\end{otherlanguage}\
\textbf{\language}{:}\Cool{ X = this, Y = that }*
\CoolOption[ Separ ]\
% ^A<---
\CoolOption
```

---

**english:**  $x$  and  $y$

**spanish:**  $x$  y  $y$

**english:**  $x$  and  $y$

**english:** this and that

**spanish:** esto y aquello

**english:** this and that

### Listing 8. Separators (*Note<sup>a</sup>*)

<sup>a</sup>[bug]: Removing the closing `\CoolOption` subsequently causes inconsistent separators between text and math mode (case replicated in uncommented form in dtx)

```
% ^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<---
\CcoolOption
\CcoolClear

```

```

x @ y
x & y
x & y

x % y @ z
x & y & z
x, y, & z
x, y, & z

```

#### Listing 9. Hello, world! (testing)

```

\CcoolOption[ Write = \BooleanTrue ]
% ^A--->
\CcoolOption[Separ = {{}}{.}{.}], Outer = {###1}]
\Ccool
<Test>{ KeyA = {.), KeyB = {!}, KeyC = {\%} }[]
<Test>{ KeyD = {d}, KeyE = {\%} }[]
<Test>c\{#1\}{ KeyF = {H}, KeyG = {e}, KeyH = {l} }*[]
<Test>{ KeyI = {\%}, KeyJ = {\%}, KeyK = {\%} }[. \{1\} \{o\}]
<Test>{ KeyL = {l}, 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
\CcoolClear

```

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

#### Listing 10. Listing 9 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

```

```

{\}\nbreak\KeyO<Test>{\}},{\ }\nbreak
\KeyM<Test>\KeyR<Test>\nbreak
\KeyO<Test>\nbreak
\KeyT<Test>\nbreak
\KeyL<Test>\nbreak
\KeyD<Test>\nbreak
\KeyZ<Test>\nbreak
\KeyN<Test>\nbreak
% ^^A<---
\CoolClear

```

---

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

#### Listing 11. 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$. ]
{}
% ^^A<---
\CoolClear
\CoolOption

```

---

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

#### Listing 12. Listing 11 read from file

```

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

```

---

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

#### Listing 13. Mittelwertsatz für $n$ Variable[1, 17.3]

```

\CoolOption[ Write = \BooleanTrue ]
% ^^A<--->
\selectlanguage{german}
\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] } }+
[$n\in\mathbb{N}, ~\mathbb{N}^n$ eine offene Menge und
$f\in C^1(\mathbb{N}^n, \mathbb{R})$.
Dann gibt es auf jeder Strecke $\subset \mathbb{N}^n$ einen Punkt
$\xi\in \mathbb{N}^n$,
{ 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: $\mathbb{N}$, $\xi$)
% ^A<---
\CoolClear
\CoolOption

```

**Theorem 1 (Mittelwertsatz für  $n$  Variable)** *Es sei  $n \in \mathbb{N}$ ,  $D \subseteq \mathbb{N}^n$  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 14. Listing 13 read from file

```

% ^A--->
\CoolRead \tab $\mathbb{N}$ $\mathbb{R}$ $\mathbb{N}^n$ $\mathbb{C}^1$ $\mathbb{N}^n$
% ^A<---
\CoolClear

```

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

#### Listing 15. Families of polynomial functions

```

\CoolOption[ Write = \BooleanTrue ]
% ^A--->
\Cool c{\mathbb{#1}}{ Nat = {\mathbb{N}}, Real = {\mathbb{R}} }
[Let-]
{ PolyR = \CoolLambda[o]{\Real\IfValueT{#1}{_#1}[X] } }
[$\text{PolyR}[n]$ and $\text{PolyR}$, denote the families of polynomial functions
on $\mathbb{R}$, of order $n$ et and their union over $n \in \mathbb{N}$,
respectively. ]
{}
% ^A<---
\CoolClear
\CoolOption

```

Let  $\mathbb{R}_n[X]$  and  $\mathbb{R}[X]$ , denote the families of polynomial functions on  $\mathbb{R}$ , of order  $n$  et and their union over  $n \in \mathbb{N}$ , respectively.

**Listing 16. Listing 15 read from file**

```
% ^^A--->
\CoolRead \tab $\PolyR[n]$ et $\PolyR$
% ^^A<---
\CoolClear
```

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

**Listing 17. Same as Listing 15, but arbitrary number system**

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

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 18. Listing 17 read from file**

```
% ^^A--->
\CoolRead \tab $\PolyR[n]$ et $\PolyR$
% ^^A<---
\CoolClear
```

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

**Listing 19. Fonction et fonctionnelle**

```
\CoolOption[ Write = \BooleanTrue ]
% ^^A--->
```

```

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

```

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

#### Listing 20. Listing 19 read from file

```

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

```

$f(t), S[f]$

#### Listing 21. CUSUM statistic[1]

```

\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{ $1\in\Real^{+}$}.
[~Define the following processes:]
{
  LogWald = { u },
  CUSUMst = { \StopT_{c} },
  CUSUM = { y },
  LogWaldInf = { m }
}+
[\begin{enumerate}
\item{

```



```

    $\LogWald_{\Time}\EvalAt{ \Scale } = \Scale\Process_{\Time} -
    \frac{1}{2}\Scale^2\Time$;
    $\LogWaldInf_{\Time}\EvalAt{ \Scale } = \inf_{0 \leq s \leq \Time}
    \CUSUM_{\Scale} \EvalAt{ \Scale }$.
  }
  \item{
    $\CUSUM_{\Time}\EvalAt{ \Scale } = \LogWaldInf_{\Time}\EvalAt{
    \Scale } - \LogWald_{\Time}\EvalAt{ \Scale } \geq 0$,
    which is the CUSUM statistic process.
  }
  \item{
    $\CUSUMst \EvalAt{ \Scale, \LogWaldInf } = \inf\left[ \Time \geq 0
    \SuchThat \CUSUM_{\Time}\EvalAt{\Scale} \geq \LogWaldInf \right]$,
    which is the CUSUM stopping time.
  }
  \end{enumerate}
\end{definition}\par[]{}

(Check: $\Scale$, $\CUSUM$)
% ^A<---
\CoolClear
\CoolOption
%
```

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 22. Listing 21 read from file

```

% ^A--->
\CoolRead \tab $\Time$ $\Process$ $\Scale$ $\Real$ $\CUSUMthresh$
  $\LogWald$ $\CUSUMst$ $\CUSUM$ $\LogWaldInf$
% ^A<---
\CoolClear
```

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

## Part III

# Other

### 1 Acknowledgment

This work has benefited from Q&A's from the L<sup>A</sup>T<sub>E</sub>Xcommunity[7][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, `$pdflatex 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

Listed under:

- a) NOTE or \NB, tagged either `bug` or `fixed`, inside `ccool.dtx`

### 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

- a) Follow the steps in [section 3](#) on one's own machine to generate `ccool.pdf`
- b) Visually check [Part II](#), against that [of the repository](#), for the same version.

Also see:

- c) Also see: [section 7 a\)](#)

## 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)*
- 7)* ccool v2.7 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*
- 8)* ccool v2.8 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

## 6.5 Other

Check [6] for testing ccool with llncs

## 7 To do

- a)* Regression testing using [3, Section 3.2—Specifying expectations].

Also see:

- b)* NOTE or \NB tagged `abandon|done|todo` inside `ccool.dtx`

## 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] The L<sup>A</sup>T<sub>E</sub>X3 Project Team *The l3build package*, 2020, <http://mirror.utexas.edu/ctan/macros/latex/contrib/l3build/l3build.pdf>
- [4] Thomas F. Sturm *The tcolorbox package*, 2019, <http://www.texdoc.net/texmf-dist/doc/latex/tcolorbox/tcolorbox.pdf>
- [5] 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>
- [6] Erwann Rogard and Olympia Hadjiliadis *Typesetting a math thesis with ccool*, 2020, <https://github.com/rogard/ccool/blob/master/thesis.pdf>
- [7] <https://tex.stackexchange.com/users/112708/erwann?tab=questions>
- [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>
- [11] @Javier Bezos’s answer to <https://tex.stackexchange.com/a/547018/112708>

## Change History

v1.0		recommended[5] . . . . .	20
General: Initial version . . . . .	20	Replace: <b>GenericObject</b> by <b>Name</b> . . . . .	20
v1.1		Replace: <b>Separators</b> by <b>Separ</b> . . . . .	20
General: Add: <b>Save</b> . . . . .	20	v1.2	
Add: Listing 1., 2., 3., 4., 6., and 9. . . . .	20	General: Add: optional <b>*to \OpsNew</b>	
Add:\OpsRestore . . . . .	20	as instruction to expand <b>kv1</b> . . . . .	20
Add:\OpsTest . . . . .	20	Delete: <b>\OpsTest</b> . . . . .	20
Delete: Listing 1-5 from v1.0 . . . . .	20	Delete: $\langle kv2 \rangle$ and $\langle code2 \rangle$ . . . . .	20
Fix: apparent anomaly in v1.0’s		Delete: Listing 2-3 from v1.1. . . . .	20
Listing 4, see Listing 9 . . . . .	20	Replace: <b>\OpsClear{<math>\langle tl2 \rangle</math>}</b> by	
Rearrange: much of the		<b>\OpsClear[<math>\langle keyval list \rangle</math>]</b> . . . . .	20
implementation . . . . .	20	Replace: <b>\Restore</b> by <b>\Read</b> . . . . .	20
Replace:		Replace: <b>\Save</b> by <b>\Write</b> . . . . .	20
<b>\OpsOptions</b> by <b>\OpsOption</b> . . . . .	20	v1.3	
Replace: $\{ \langle kv2 \rangle \}$ by $\langle \langle kv2 \rangle \rangle$ given		General: Replace: <b>\OpsNew</b> by <b>\Ops</b> . . . . .	20
that option type <b>G</b> not			

	Replace: $\{\langle tl_2 \rangle\}$ and $[\langle tl_2 \rangle]$ by $\langle \langle tl_2 \rangle \rangle$ . . . . .	20	v2.1	General: Add: Listings 3, 4, 5, 6, 7, 8, and 9 . . . . .	20
v1.4	General: Add: <b>section 9</b> . . . . .	20		Replace: $\langle tl_2 \rangle$ 's position within $\backslash\text{Ccool}$ 's argument list, from first to second. Greater versatility . . . .	20
	Add: $\backslash\text{OpsDebug}$ . . . . .	20		Replace: $\backslash\text{CcoolLambda}$ 's optional integer argument (number of m's) by a standard argument list . . . .	20
	Add: $\backslash\text{OpsHook}$ . . . . .	20		Replace: <i>global option Name</i> by <b>Param</b> . . . .	20
	Add: <b>Expans</b> (for debugging' sake, but...) . . . . .	20		Replace: as the default of <b>Param</b> , <b>Math</b> by <b>Default</b> . . . .	20
	Add: Listing 1., 2., and 3. . . . .	20	v2.2	General: Remove: % from listings . . . .	20
	Add: optional <b>+to</b> $\backslash\text{OpsNew}$ to make side effects persist beyond local group . . . . .	20		Replace: part of the abstract's with more straightforward descriptions based on input from forum participants . . . . .	20
	Delete: Listing 1., and 2. . . . .	20			
	Replace: $\text{s}\{\{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\}\{\langle tl_5 \rangle\}\}$ by $\text{s}\{\{\{\langle tl_3 \rangle\} \{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\} \{\langle tl_3 \rangle\}\{\langle tl_4 \rangle\}\{\langle tl_5 \rangle\}\}$ . . . . .	20	v2.3	General: Add: Listing <b>16</b> , Listing <b>17</b> , and Listing <b>18</b> . . . . .	20
v1.5	General: Add: <b>File</b> . . . . .	20		Complete: Listing <b>15</b> . . . . .	20
	Delete: dependence on <b>datetime</b> . . . .	20		Rearranged: $\backslash\text{Ccool}$ 's subsections. Previously, by argument. Now, by feature. . . . .	20
v1.6	General: Add: Listing showing part of the preamble . . . . .	20		Remove: Listing showing part of the preamble . . . . .	20
	Rename: $\backslash\text{OpsClear}$ to $\backslash\text{CcoolClear}$ . . . . .	20		Replace: for $\backslash\text{Ccool}$ , <b>i{}</b> by <b>c{}</b> . . . .	20
	Rename: $\backslash\text{OpsDebug}$ to $\backslash\text{CcoolDebug}$ . . . . .	20		Replace: In step <b>2)</b> , the created command's implementation, from $\backslash\text{ProvideDocumentCommand}$ to $\backslash\text{DeclareDocumentCommand}$ . . . .	20
	Rename: $\backslash\text{OpsHook}$ to $\backslash\text{CcoolHook}$ . . . .	20	v2.4	General: Fix: minor error in the listings ( $\backslash\text{Real}$ rather than $\backslash\text{Reel}$ , hitherto unnoticed). . . . .	20
	Rename: $\backslash\text{OpsOption}$ to $\backslash\text{CcoolOption}$ . . . . .	20		Remove: examples from <b>Part I</b> , $\backslash\text{Ccool}$ , as redundant with <b>Part II</b> Listing <b>2-6</b> . . . . .	20
	Rename: $\backslash\text{OpsRead}$ to $\backslash\text{CcoolRead}$ . . . .	20	v2.5	General: Modify: <b>File</b> , rely on $\text{erw-l3's}$ $\backslash\text{erw\_jobnametimestamp}$ : . . . . .	20
	Rename: $\backslash\text{Ops}$ to $\backslash\text{Ccool}$ . . . . .	20		Modify: behavior of <b>Part I Expand</b> the <i>vals</i> , rely on $\text{erw-l3's}$ $\backslash\text{erw\_seq\_use:Nn}$ . . . . .	20
	Rename: <b>oops</b> to <b>ccool</b> (better describes the purpose) . . . . .	20	v2.6	General: Modify: $\backslash\text{CcoolLambda}$ , rely on $\text{erw-l3's}$ $\backslash\text{erw\_lambda:nnn}$ . . . .	20
v1.7	General: Add: Legends to listings . . . .	20	v2.7	General: Add: <i>global option</i> <b>And</b> . . . .	20
	Add: Listing <b>21</b> (CUSUM) . . . . .	20		Add: Listing <b>7</b> . . . . .	20
	Delete: $\backslash\text{CcoolDebug}$ . . . . .	20		Modify: $\backslash\text{CcoolOption}$ 's 'm'-type argument by 'o'-type argument . . . .	20
	Delete: Listing 5 from v1.6 . . . . .	20			
v1.8	General: Add: $\backslash\text{CcoolVers}$ . . . . .	20			
	Add: $\backslash\text{CcoolLambda}$ . . . . .	20			
	Add: Listing <b>19</b> , Listing <b>20</b> . . . . .	20			
	Add: Listing <b>1</b> . . . . .	20			
v1.9	General: Add: support for LuaTeX . . . .	20			
	Move: from <b>Part I</b> to <b>Part IV</b> , what is now that part's <b>section 11</b> . . . .	20			
v2.0	General: Add: support for XeTeX . . . .	20			
	Delete: <b>File</b> 's dependency on <b>texosquery</b> and $\backslash\text{pdfcreationdate}$ . . . .	20			
	Update: $\backslash\text{RequirePackage}$ , $\backslash\text{NeedsTeXFormat}$ 's second argument / TeX Live 2020 . . . . .	20			

Modify: <b>Separ</b> 's default rely on babel and amsmath, if applicable . 20	Parameterize the <i>key</i> s . . . . . 20
v2.8	v2.9
General: Fix: conflict between \usepackage[spanish]{babel} and	General: Miscellaneous . . . . . 20
	v3.0
	General: Miscellaneous . . . . . 20

## 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	
\'	103
\langle key <sub>i</sub> <tl <sub>2</sub> >	6
\langle key <sub>i</sub> \rangle	5, 8
\langle key <sub>i</sub> \rangle	5
And (option)	6, 35
Expans (option)	7, 35
File (option)	7, 36
Inner (option)	7, 36
Outer (option)	7, 36
Param (option)	7, 36
Seper (option)	7, 37
Write (option)	7, 37
\^	108
\_	273, 274, 275, 281, 282, 283
A	
\AtEndDocument	125
B	
\beginngroup	210
bool commands:	
\bool_gset_false:N	130
\bool_gset_true:N	137
\bool_if:nTF	152, 206, 229, 415
\bool_set_false:N	126
\BooleanFalse	7, 420, 421
\BooleanTrue	6, 7
C	
\c	115
\Ccool	1,
	2, 5, 6, 6, 6, 10, 20, 21, 211, 218, 239
ccool internal commands:	
\_ccool_aux_inner:n	6, 7, 38
\_ccool_aux_inner_set:n	4, 196
\_ccool_aux_key:N	17, 200
\_ccool_aux_key:n	13, 20
\_ccool_aux_key:w	9, 15
\_g_ccool_aux_key_seq	
	11, 19, 181, 201, 305, 306
\_g_ccool_aux_keyval_seq	197, 198, 200
\_ccool_aux_outer:n	24, 183
\_ccool_aux_outer_set:n	22, 182
\_g_ccool_aux_prop	26, 29, 46, 199
\_ccool_aux_prop:N	44, 198
\_ccool_aux_prop:n	40, 50
\_ccool_aux_prop:nn	26
\_ccool_aux_prop:w	32, 42
\_ccool_aux_val:Nn	53, 181
\_g_ccool_aux_val_seq	55, 56, 187
\_ccool_lang_and:	68, 273, 275, 281, 283
\_ccool_lang_and:n	68
\_g_ccool_lang_and_prop	58, 62, 71, 73
\_c_ccool_lang_and_tl	86, 347, 348
\_ccool_lang_and_update:n	59, 346
\_ccool_log_close:	124, 417
\_ccool_log_entry	211, 212
\_g_ccool_log_file_tl	132, 135, 354
\_g_ccool_log_iow	
	124, 125, 129, 136, 154, 157
\_ccool_log_open:	132, 416
\_g_ccool_log_open_bool	
	126, 130, 137, 152, 206
\_ccool_log_read:	145, 428
\_ccool_log_read:n	139, 147, 427
\_g_ccool_log_to_tl	135, 136, 147, 149
\_ccool_log_write:n	149, 208
\_ccool_make_ccool:nnnn	
	216, 361, 375, 389, 403
\_ccool_make_ccool_exp:nnn	179, 227
\_ccool_make_ccool_key:nnn	191, 205
\_ccool_make_ccool_sideeffect:nnn	
	203, 224, 233
\_ccool_make_key:N	175, 201
\_ccool_make_key:n	170, 177
\_ccool_make_key:Nn	160, 172
\_g_ccool_option_expans_tl	32, 36, 350
\_ccool_option_inner:n	249, 359

\g_ccool_option_inner_tl . . . . .	\ensuremath . . . . . 398, 399
. . . . . 249, 252, 364, 378, 392, 406	erw commands:
\_ccool_option_outer:n . . . . 259, 387	\erw_jobnametimestamp: . . . . . 21
\g_ccool_option_outer_tl . . . . .	\erw_keyval_error:Nn . . . . . 63
. . . . . 259, 262, 366, 380, 394, 408	\erw_lambda:nnn . . . . . 21, 337
\_ccool_option_param:n . . . . 254, 373	\erw_prop_keyval_parse:NNNn . . . . 61
\g_ccool_option_param_tl . . . . .	\erw_seq_use:Nn . . . . . 21, 186
164, 254, 257, 330, 363, 377, 391, 405	\erw_sys_jobnametimestamp: . 356, 357
\_ccool_option_separ:n . . . . 264, 401	exp commands:
\c_ccool_option_separ_default_-	\exp_args:NNf . . . . . 185
tl . . . . . 271, 279, 412, 413	\exp_args:NNx . . . . . 162, 218
\g_ccool_option_separ_tl . . . . .	\exp_args:No . . . . . 81, 196, 301
. . . . 264, 267, 269, 365, 379, 393, 407	\exp_args:Nx . . . . . 35
\_ccool_prop_append:NN . . . . 286, 297	\exp_last_unbraced:Nf 360, 374, 388, 402
\_ccool_prop_append:Nn . . . . 199, 295	\exp_not:n . . . . . 239
\_ccool_prop_append:nn . . . . 288, 292	\expandafter . . . . . 211
\_ccool_prop_clear_new:n . . . 299, 306	\ExplSyntaxOff . . . . . 433
\_ccool_prop_clear_new_map:n . . .	\ExplSyntaxOn . . . . . 3
. . . . . 303, 332	
\_ccool_prop_if_exist:nTF . 193, 308	<b>F</b>
\_ccool_prop_item:nn . . . . 166, 312	file commands:
\_ccool_prop_name:n . . . . .	\file_input:n . . . . . 141
. . . . 56, 297, 301, 310, 314, 316, 319	\function . . . . . 5
\_ccool_prop_new:n . . . . . 195, 317	
\_ccool_seq_from_prop:n . . . 323, 327	<b>G</b>
\_ccool_seq_from_prop:NNn . . 56, 321	\gappto . . . . . 231
\CcoolClear . . . . . 2, 4, 6, 21, 329	
\CcoolDebug . . . . . 21	<b>I</b>
\CcoolHook . . . . . 2, 4, 5, 6, 21, 231, 334	\IfBooleanT . . . . . 225
\CcoolLambda . . . . . 2, 4, 5, 6, 8, 21, 335	\ifcsdef . . . . . 79, 269
\CcoolOption . . . . . 2, 4, 6, 21, 339	\IfValueT . . . . . 223, 237
\CcoolRead . . . . . 2, 4, 8, 8, 21, 423	\IfValueTF . . . . . 426
\CcoolVers . . . . . 2, 4, 8, 10, 21, 430	iow commands:
cs commands:	\iow_close:N . . . . . 125, 129
\cs_generate_variant:Nn . . . . .	\iow_new:N . . . . . 124
. . . . 7, 31, 67, 144, 159, 169, 174, 294	\iow_now:Nn . . . . . 154
\cs_gset:Npn . . . . . 6, 24, 267	\iow_open:Nn . . . . . 136
\cs_new:Nn . . . . . 68, 81, 84, 308, 312	
\cs_new:Npn . . . . . 316	<b>K</b>
\cs_new_protected:Nn . . . . .	keys commands:
. . . . . 4, 13, 17, 22, 27, 40,	\l_keys_choice_tl . . . . . 350
44, 53, 59, 127, 133, 139, 145, 150,	\keys_define:nn . . . . . 344
160, 170, 175, 179, 191, 203, 250,	\keys_set:nn . . . . . 342
255, 260, 265, 295, 299, 303, 317, 321	
\cs_new_protected:Npn 9, 33, 216, 286	<b>L</b>
\cs_set:Nn . . . . . 288	\language_name . . . . . 81
\cs_set_protected:Nn . . . . . 323	
	<b>M</b>
<b>D</b>	msg commands:
\DeclareDocumentCommand 6, 21, 163, 218, 337	\msg_error:nnn . . . . . 157
\def . . . . . 211	\msg_new:nnn . . . . . 243, 246
	\msg_warning:nnn . . . . . 75
<b>E</b>	
\endgroup . . . . . 212	<b>N</b>
	\NB . . . . . 18, 19

<code>\NeedsTeXFormat</code> .....	21			<b>R</b>	
<code>\NewDocumentCommand</code> .....				<code>\Read</code> .....	20
..... 2, 10, 329, 334, 339, 423, 430				<code>\Real</code> .....	1, 5, 21
				<code>\Reel</code> .....	1, 21
<b>O</b>				<code>\RequirePackage</code> .....	21
<code>\Ops</code> .....	20, 21			<code>\Restore</code> .....	20
<code>\OpsClear</code> .....	20, 21				
<code>\OpsDebug</code> .....	21			<b>S</b>	
<code>\OpsHook</code> .....	21			<code>\Save</code> .....	20
<code>\OpsNew</code> .....	20, 21			seq commands:	
<code>\OpsOption</code> .....	20, 21			<code>\seq_gclear_new:N</code> .....	19, 55
<code>\OpsOptions</code> .....	20			<code>\seq_gput_right:Nn</code> .....	11, 325
<code>\OpsRead</code> .....	21			<code>\seq_if_empty:NTF</code> .....	47
<code>\OpsRestore</code> .....	20			<code>\seq_map_function:NN</code> .....	
<code>\OpsTest</code> .....	20			..... 20, 50, 177, 306, 327	
options:				<code>\seq_set_from_clist:Nn</code> .....	197, 305
<code>And</code> .....	6, 35				
<code>Expans</code> .....	7, 35			<b>T</b>	
<code>File</code> .....	7, 36			<code>\text</code> .....	273, 274, 275
<code>Inner</code> .....	7, 36			tl commands:	
<code>Outer</code> .....	7, 36			<code>\c_empty_tl</code> ....	48, 194, 214, 236, 334
<code>Param</code> .....	7, 36			<code>\tl_const:Nn</code> .....	86, 271, 279
<code>Separ</code> .....	7, 37			<code>\tl_gset:Nn</code> ...	135, 252, 257, 262, 354
<code>Write</code> .....	7, 37			<code>\tl_gset_eq:NN</code> .....	350
				<code>\tl_log:n</code> .....	142, 155
<b>P</b>				<code>\tl_new:N</code> 32, 132, 149, 249, 254, 259, 264	
<code>\pdfcreationdate</code> .....	21			<code>\tl_trim_spaces:n</code> .....	11, 37, 38
prop commands:					
<code>\prop_clear_new:N</code> .....	301			<b>U</b>	
<code>\prop_gclear_new:N</code> .....	46			use commands:	
<code>\prop_gput:Nnn</code> .....	29, 64, 290			<code>\use:N</code> .....	36, 432
<code>\prop_if_exist:NTF</code> .....	310			<code>\usepackage</code> .....	5, 22
<code>\prop_if_in:NnTF</code> .....	70				
<code>\prop_item:Nn</code> .....	73, 290, 314, 325			<b>W</b>	
<code>\prop_map_function:NN</code> .....	292			<code>\Write</code> .....	20
<code>\prop_new:N</code> .....	26, 58, 319				
<code>\ProvideDocumentCommand</code> .....	21, 335			<b>X</b>	
<b>Q</b>					
quark commands:					
<code>\q_stop</code> .....	9, 15, 33, 42			<code>\X</code> .....	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 lang

```

58 \prop_new:N \g__ccool_lang_and_prop

\__ccool_lang_and_update:n

59 \cs_new_protected:Nn \__ccool_lang_and_update:n
60 {
61   \erw_prop_keyval_parse:NNNn
62   \g__ccool_lang_and_prop
63   \erw_keyval_error:Nn
64   \prop_gput:Nnn
65   { #1 }
66 }
67 \cs_generate_variant:Nn \__ccool_lang_and_update:n { e }

(End definition for \__ccool_lang_and_update:n.)

\__ccool_lang_and:n
\__ccool_lang_and:
68 \cs_new:Nn \__ccool_lang_and:n
69 {
70   \prop_if_in:NnTF
71   \g__ccool_lang_and_prop
72   {#1}
73   {\prop_item:Nn\g__ccool_lang_and_prop{#1}}
74   {
75     \msg_warning:nnn{__ccool}{lang_and}{#1}
76     \__ccool_lang_and:n{english}
77   }
78 }
79 \ifcsdef{language_name}
80 {
81   \cs_new:Nn \__ccool_lang_and:{\exp_args:No\__ccool_lang_and:n{\language_name}}
82 }
83 {
84   \cs_new:Nn \__ccool_lang_and:{english}
85 }

(End definition for \__ccool_lang_and:n and \__ccool_lang_and:.)

\c__ccool_lang_and_tl (Note1)
86 \tl_const:Nn \c__ccool_lang_and_tl
87 {
88   %%^A https://www.overleaf.com/learn/latex/International\_language\_support
89   afrikaans=en,
90   basque=eta,
91   catalan=i,
92   croatian=i,
93   czech=a,
94   danish=og,
95   dutch=en,
96   english=and,
97   esperanto=kaj,
98   estonian=ja,

```

---

<sup>1</sup>[todo]: Non latin-alphabet languages

```

99   finnish=ja,
100  french=et,
101  galician=e,
102  german=und,
103  hungarian=\’es,
104  icelandic=og,
105  indonesian=dan,
106  irish=agus,
107  italian=e,
108  kurmanji=\^u,
109  latin=et,
110  latvian=un,
111  lithuanian=ir,
112  ngerman=und,
113  polish=i,
114  portuguese=e,
115  romanian=\c{s}i,
116  slovak=a,
117  spanish=y,
118  swedish=och,
119  swissgerman=und,
120  turkish=ve,
121  turkmen=we,
122  welsh=a
123 }

```

*(End definition for \c\_\_ccool\_lang\_and\_tl.)*

## 4 log

\\_\_ccool\_log\_close:

```

124 \iow_new:N \g__ccool_log_iow
125 \AtEndDocument{\iow_close:N \g__ccool_log_iow}
126 \bool_set_false:N \g__ccool_log_open_bool
127 \cs_new_protected:Nn \__ccool_log_close:
128 {
129   \iow_close:N \g__ccool_log_iow
130   \bool_gset_false:N \g__ccool_log_open_bool
131 }

```

*(End definition for \\_\_ccool\_log\_close:.)*

\\_\_ccool\_log\_open:

```

132 \tl_new:N \g__ccool_log_file_tl
133 \cs_new_protected:Nn \__ccool_log_open:
134 {
135   \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
136   \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
137   \bool_gset_true:N \g__ccool_log_open_bool
138 }

```

*(End definition for \\_\_ccool\_log\_open:.)*

```

\__ccool_log_read:n #1: <path>

139 \cs_new_protected:Nn \__ccool_log_read:n
140 {
141   \file_input:n{#1}
142   \tl_log:n{read~from~#1}
143 }
144 \cs_generate_variant:Nn \__ccool_log_read:n { e }

```

(End definition for \\_\_ccool\_log\_read:n.)

```

\__ccool_log_read:

145 \cs_new_protected:Nn \__ccool_log_read:
146 {
147   \__ccool_log_read:e{\g__ccool_log_to_tl}
148 }

```

(End definition for \\_\_ccool\_log\_read:.)

```

\__ccool_log_write:n

149 \tl_new:N \g__ccool_log_to_tl
150 \cs_new_protected:Nn \__ccool_log_write:n
151 {
152   \bool_if:nTF{ \g__ccool_log_open_bool }
153   {
154     \iow_now:Nn \g__ccool_log_iow {#1}
155     \tl_log:n{ write~to~#1 }
156   }
157   { \msg_error:nnn{ __ccool }{ iow }{ \g__ccool_log_iow } }
158 }
159 \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 >

160 \cs_new_protected:Nn \__ccool_make_key:Nn
161 {
162   \exp_args:NNx
163   \DeclareDocumentCommand{#1}
164   { D<>{\g__ccool_option_param_tl} }
165   {
166     \__ccool_prop_item:nn{##1}{#2}
167   }
168 }
169 \cs_generate_variant:Nn \__ccool_make_key:Nn { c }

```

(End definition for \\_\_ccool\_make\_key:Nn.)

```

\__ccool_make_key:n #1: < key >

170 \cs_new_protected:Nn \__ccool_make_key:n
171 {
172   \__ccool_make_key:cn{#1}{#1}
173 }
174 \cs_generate_variant:Nn \__ccool_make_key:n { e }

(End definition for \__ccool_make_key:n.)

```

```

\__ccool_make_key:N #1: < seq >

175 \cs_new_protected:Nn \__ccool_make_key:N
176 {
177   \seq_map_function:NN #1 \__ccool_make_key:e
178 }

(End definition for \__ccool_make_key:N.)

```

## 6 make\_ccool

```

\__ccool_make_ccool_exp:nnn

179 \cs_new_protected:Nn \__ccool_make_ccool_exp:nnn
180 {
181   \__ccool_aux_val:Nn \g__ccool_aux_key_seq {#1}
182   \__ccool_aux_outer_set:n{#3}
183   \__ccool_aux_outer:n
184   {
185     \exp_args:NNf
186     \erw_seq_use:Nn
187     \g__ccool_aux_val_seq
188     {#2}
189   }
190 }

(End definition for \__ccool_make_ccool_exp:nnn.)

```

```

\__ccool_make_ccool_key:nnn

191 \cs_new_protected:Nn \__ccool_make_ccool_key:nnn
192 {
193   \__ccool_prop_if_exist:nTF{#1}
194   { \c_empty_tl }
195   { \__ccool_prop_new:n{#1} }
196   \exp_args:No \__ccool_aux_inner_set:n{#2}
197   \seq_set_from_clist:Nn \g__ccool_aux_keyval_seq {#3}
198   \__ccool_aux_prop:N \g__ccool_aux_keyval_seq
199   \__ccool_prop_append:Nn \g__ccool_aux_prop {#1}
200   \__ccool_aux_key:N \g__ccool_aux_keyval_seq
201   \__ccool_make_key:N \g__ccool_aux_key_seq
202 }

(End definition for \__ccool_make_ccool_key:nnn.)

```

```

\__ccool_make_ccool_sideeffect:nnn [9]
203 \cs_new_protected:Nn \__ccool_make_ccool_sideeffect:nnn
204 {
205   \__ccool_make_ccool_key:nnn{#1}{#2}{#3}
206   \bool_if:nTF{ \g__ccool_log_open_bool }
207   {
208     \__ccool_log_write:n
209     {
210       \begin{group}
211       \def \__ccool_log_entry { \Ccool<#1>c{#2}{#3} } \expandafter
212       \endgroup \__ccool_log_entry
213     }
214   }\c_empty_tl}
215 }

```

(End definition for \\_\_ccool\_make\_ccool\_sideeffect:nnn.)

```

\__ccool_make_ccool:nnnn #1 : < token list >
#2 : < seq1 >
#3 : < seq2 >
#4 : < prop >
216 \cs_new_protected:Npn \__ccool_make_ccool:nnnn #1 #2 #3 #4
217 {
218   \exp_args:NNx \DeclareDocumentCommand \Ccool
219   {%^^A      2      3      4 5 6      7 8 9
220   +o D<>{#1} E{ c }{{#2}} m t+ s E{ s c }{{#3}{#4}} +o
221   }
222   {
223     \IfValueT{##1}{##1}
224     \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
225     \IfBooleanT{##6}
226     {
227       \__ccool_make_ccool_exp:nnn{##2}{##7}{##8}
228     }
229     \bool_if:nTF{##5}
230     {
231       \gappto{\CcoolHook}
232       {
233         \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
234       }
235     }
236     {\c_empty_tl}
237     \IfValueT{##9}
238     {
239       \exp_not:n{ \Ccool[##9] }
240     }
241   }
242 }

```

(End definition for \\_\_ccool\_make\_ccool:nnnn.)

## 7 msg

```

243 \msg_new:nnn {__ccool}
244 { iow }
245 {#1~is~closed~can't~write}
246 \msg_new:nnn {__ccool}
247 {lang_and}
248 {~key~#1~missing~for~global~option~'And';~falling~back~on~'english'}

```

## 8 option

```

\__ccool_option_inner:n #1: <code>

249 \tl_new:N \g__ccool_option_inner_tl
250 \cs_new_protected:Nn \__ccool_option_inner:n
251 {
252   \tl_gset:Nn \g__ccool_option_inner_tl {#1}
253 }

```

(End definition for \\_\_ccool\_option\_inner:n.)

```

\__ccool_option_param:n #1: <token list>

254 \tl_new:N \g__ccool_option_param_tl
255 \cs_new_protected:Nn \__ccool_option_param:n
256 {
257   \tl_gset:Nn \g__ccool_option_param_tl{#1}
258 }

```

(End definition for \\_\_ccool\_option\_param:n.)

```

\__ccool_option_outer:n #1: < inline code >

259 \tl_new:N \g__ccool_option_outer_tl
260 \cs_new_protected:Nn \__ccool_option_outer:n
261 {
262   \tl_gset:Nn \g__ccool_option_outer_tl {#1}
263 }

```

(End definition for \\_\_ccool\_option\_outer:n.)

```

\__ccool_option_separ:n #1: {< tl1 >}{< tl2 >}{< tl3 >}

264 \tl_new:N \g__ccool_option_separ_tl
265 \cs_new_protected:Nn \__ccool_option_separ:n
266 {
267   \cs_gset:Npn \g__ccool_option_separ_tl {#1}
268 }

```

(End definition for \\_\_ccool\_option\_separ:n.)

```

\g__ccool_option_separ_tl

269 \ifcsdef{text}
270 {
271   \tl_const:Nn \c__ccool_option_separ_default_tl
272   {
273     { \text{{\ } \__ccool_lang_and:{\ }} }
274     { \text{,{\ }} }
275     { \text{,{\ } \__ccool_lang_and:{\ }} }
276   }

```



```

277 }
278 {
279   \tl_const:Nn \c_ccool_option_separ_default_tl
280   {
281     { {\ } \__ccool_lang_and:{\ } }
282     { ,{\ } }
283     { ,{\ } \__ccool_lang_and:{\ } }
284   }
285 }

```

(End definition for \g\_ccool\_option\_separ\_tl.)

## 9 prop

```

\__ccool_prop_append:NN #1 : < prop1 >
#2 : < prop2 >

286 \cs_new_protected:Npn \__ccool_prop_append:NN #1 #2
287 {
288   \cs_set:Nn \__ccool_prop_append:nn
289   {
290     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
291   }
292   \prop_map_function:NN #2 \__ccool_prop_append:nn
293 }
294 \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 >

295 \cs_new_protected:Nn \__ccool_prop_append:Nn
296 {
297   \__ccool_prop_append:cN{ \__ccool_prop_name:n {#2} } #1
298 }

```

(End definition for \\_\_ccool\_prop\_append:Nn.)

```

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

299 \cs_new_protected:Nn \__ccool_prop_clear_new:n
300 {
301   \exp_args:No \prop_clear_new:c{ \__ccool_prop_name:n {#1} }
302 }

```

(End definition for \\_\_ccool\_prop\_clear\_new:n.)

```

\__ccool_prop_clear_new_map:n #1 : < keyval list >

303 \cs_new_protected:Nn \__ccool_prop_clear_new_map:n
304 {
305   \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
306   \seq_map_function:NN \g__ccool_aux_key_seq \__ccool_prop_clear_new:n
307 }

```

(End definition for \\_\_ccool\_prop\_clear\_new\_map:n.)

```

\__ccool_prop_if_exist:nTF #1 :  $\langle tl_1 \rangle$ 
#2 :  $\langle tl_2 \rangle$ 
#3 :  $\langle tl_3 \rangle$ 

308 \cs_new:Nn \__ccool_prop_if_exist:nTF
309 {
310   \prop_if_exist:cTF{ \__ccool_prop_name:n {#1} }{#2}{#3}
311 }

```

*(End definition for \\_\_ccool\_prop\_if\_exist:nTF.)*

```

\__ccool_prop_item:nn #1 :  $\langle tl \text{ var name} \rangle$ 
#2 :  $\langle key \rangle$ 

312 \cs_new:Nn \__ccool_prop_item:nn
313 {
314   \prop_item:cn { \__ccool_prop_name:n {#1} } {#2}
315 }

```

*(End definition for \\_\_ccool\_prop\_item:nn.)*

```

\__ccool_prop_name:n #1 :  $\langle tl \text{ var name} \rangle$ 

316 \cs_new:Npn \__ccool_prop_name:n #1{ __ccool_#1 }

```

*(End definition for \\_\_ccool\_prop\_name:n.)*

```

\__ccool_prop_new:n #1 :  $\langle tl \text{ var name} \rangle$ 

317 \cs_new_protected:Nn \__ccool_prop_new:n
318 {
319   \prop_new:c{ \__ccool_prop_name:n {#1} }
320 }

```

*(End definition for \\_\_ccool\_prop\_new:n.)*

## 10 seq

```

\__ccool_seq_from_prop:NNn #1 :  $\langle seq_1 \rangle$ 
#2 :  $\langle seq_2 \rangle$  (keys)
#3 :  $\langle prop \rangle$ 

321 \cs_new_protected:Nn \__ccool_seq_from_prop:NNn
322 {
323   \cs_set_protected:Nn \__ccool_seq_from_prop:n
324   {
325     \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
326   }
327   \seq_map_function:NN #2 \__ccool_seq_from_prop:n
328 }

```

*(End definition for \\_\_ccool\_seq\_from\_prop:NNn.)*

## 11 Front-end

### `\CcoolClear`

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

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

### `\CcoolHook`

```
334 \NewDocumentCommand{\CcoolHook}{\c_empty_tl}
```

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

### `\CcoolLambda` (Note<sup>2</sup>)

```
335 \ProvideDocumentCommand \CcoolLambda { O{m} m }
336 {
337   \erw_lambda:nnn \DeclareDocumentCommand { #1 } { #2 }
338 }
```

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

### `\CcoolOption` (Note<sup>3</sup>) (Note<sup>4</sup>)

```
339 \NewDocumentCommand{ \CcoolOption }
340 { O{ And, Expans, File, Inner, Param, Outer, Separ, Write } }
341 {
342   \keys_set:nn{ __ccool }{#1}
343 }
```

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

```
344 \keys_define:nn { __ccool }
345 {
```

`And`

```
346 And .code:n = { \__ccool_lang_and_update:e{ #1 } },
347 And .default:n = { \c__ccool_lang_and_tl },
348 And .initial:n = { \c__ccool_lang_and_tl },
```

`Expans`

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

## File

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

## Inner

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

## Param

```
372 Param .code:n={
373   \__ccool_option_param:n{#1}
374   \exp_last_unbraced:Nf
375   \__ccool_make_ccool:nnnn
376   {
377     { \g__ccool_option_param_tl }
378     { \g__ccool_option_inner_tl }
379     { \g__ccool_option_separ_tl }
380     { \g__ccool_option_outer_tl }
381   }
382 },
383 Param .value_required:n = false,
384 Param .default:n = { Default },
385 Param .initial:n = { Default },
```

## Outer

```
386 Outer .code:n={
387   \__ccool_option_outer:n{#1}
388   \exp_last_unbraced:Nf
389   \__ccool_make_ccool:nnnn
390   {
391     { \g__ccool_option_param_tl }
392     { \g__ccool_option_inner_tl }
393     { \g__ccool_option_separ_tl }
394     { \g__ccool_option_outer_tl }
395   }
396 },
```

---

<sup>2</sup>[todo]: allow only m- or o-type arguments

<sup>3</sup>[todo]: Fix placeholders passed to options requiring code (only one pound sign)

<sup>4</sup>[abandon]: Requirement: write to file if Write; Update: redundant with \cs  
{Ccool}+Write

```

397 Outer .value_required:n = false,
398 Outer .default:n = { \ensuremath{####1} },
399 Outer .initial:n = { \ensuremath{####1} },

```

Separ

```

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

```

Write

```

414 Write .code:n = {
415   \bool_if:nTF{#1}
416   {\__ccool_log_open:}
417   {\__ccool_log_close:}
418 },
419 Write .value_required:n = false,
420 Write .default:n = \BooleanFalse,
421 Write .initial:n = \BooleanFalse
422 }

```

**\CcoolRead**

```

423 \NewDocumentCommand{\CcoolRead}
424 {o}
425 {
426   \IfValueTF{#1}
427   {\__ccool_log_read:e{#1}}
428   {\__ccool_log_read:}
429 }

```

(End definition for \CcoolRead. This function is documented on page 8.)

**\CcoolVers**

```

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

```

(End definition for \CcoolVers. This function is documented on page 8.)

## 12 Closing

```

433 \ExplSyntaxOff
434 \</package>

```