The ccool package[*]

Erwann Rogard[†]

Released 2020/06/03

Abstract

The package ccool for LaTeX 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 overriden inline. This versality could find its use, for example, to encode notational conventions (such as `\Real` → `\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 useful for typesetting documents sharing the same notation.

Résumé

L'extension ccool pour LaTeX met à disposition une interface de type *clé-valeur*, `\Ccool`, destinée à faciliter la géneration 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` → `\mathbb{R}`), au point dans le `document` où elles sont introduites ("Soit $\mathbb{R}$ les nombres réels."). Des commandes polymorphes peuvent être générées, en associant aux clés un paramètre (par exemple, une valeur pour le style typographique, une autre pour la description du concept associé). En option, les instructions passées à cette interface peuvent être sauvegardées, ce qui peut être utile pour la rédaction de documents faisant appel à des conventions typographiques communes.

Contents

I Usage 4

0 Convention 4

1 Loading the package 5

[*]This file describes version v2.9, last revised 2020/06/03.

[†]firstname dot lastname AusTria gmail dot com

2 `\Ccool` **5**
- 2.1 Core feature . . . 5
- 2.2 Process the val_is . . . 5
- 2.3 Append to a hook . . . 5
- 2.4 Expand the val_is . . . 5
- 2.5 Head . . . 6
- 2.6 Tail . . . 6
- 2.7 Parameterize the key_is . . . 6
- 2.8 Write . . . 6

3 `\CcoolClear` **6**

4 `\CcoolHook` **6**

5 `\CcoolLambda` **6**

6 `\CcoolOption` **6**
- 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

7 `\CcoolRead` **8**

8 `\CcoolVers` **8**

9 **Do's and dont's** **8**

II Listing **10**

1.`\CcoolVers` **10**

2. Let ℕ and 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 $n$ariable. 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 fonctionelle 15

20. Listing 19 read from file 16

21. CUSUM statistic. 16

22. Listing 21 read from file 17

III Other 18

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

Change History 20

Index **22**

IV Implementation **25**

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 ⟨*token list*⟩.

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) `\X` $\rightarrow$ `Y`: `\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 LaTeX engine. See Part III, section 5.
2. Put in the *preamble*

`\Ccool` `\Ccool[`$\langle$*tl*$_1\rangle$`]<`$\langle$*tl*$_2\rangle$`>c{`$\langle$*code*$_1\rangle$`}{`$\langle$*kvl*$_1\rangle$`}+*s{`$\langle$*separators*$\rangle$`}c{`$\langle$*code*$_2\rangle$`}[`$\langle$*tl*$_6\rangle$`]`
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{`$\langle kvl_1\rangle$`}` executes for each $\langle key_i\rangle$=$\langle val_i\rangle$,

1) $\langle val_i\rangle \leftarrow$ `\function{`$\langle val_i\rangle$`}`

2) define `\`$\langle key_i\rangle$ such that `\`$\langle key_i\rangle \rightarrow \langle val_i\rangle$,

where `\function` is encoded in *global option* `Inner`. For instance, the side effect of `\Ccool{ Real = \mathbb{R} }` is `\Real` $\rightarrow$ `\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 `\`$\langle key_i\rangle$ to take arguments.

2.2 Process the *val*$_i$s

`\Ccool c{`$\langle code_1\rangle$`}{`$\langle kvl_1\rangle$`}` 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{`$\langle kvl_1\rangle$`}+` 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_is

`\Ccool{`$\langle kvl_1\rangle$`}*` 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 overriden inline by appending further `s{`$\langle separators\rangle$`}` and `c{`$\langle code_2\rangle$`}`, respectively, to the list of arguments.

2.5 Head

`\Ccool[`$\langle tl_1\rangle$`]{`$\langle kvl_1\rangle$`}` 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{`$\langle kvl_1\rangle$`}[`$\langle tl_6\rangle$`]{`$\langle kvl_2\rangle$`}` is identical to `\Ccool{`$\langle kvl_1\rangle$`}` followed by `\Ccool[`$\langle tl_6\rangle$`]{`$\langle kvl_2\rangle$`}`.

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

2.7 Parameterize the key_is

`\Ccool<`$\langle tl_2\rangle$`>{`$\langle kvl_1\rangle$`}` 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`.

`\CcoolClear`

`\CcoolClear<`$\langle tl_2\rangle$`>{...`$\langle key_i\rangle$`,...}`

Semantics Clears all `\`$\langle key_i$`<`tl_2`>`$\rangle$'s

`\CcoolHook`

`\CcoolHook`

Semantics No side effect or expansion

`\CcoolLambda`

`\CcoolLambda[`$\langle arg\ spec\rangle$`]{`$\langle code\rangle$`}`,

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

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

Semantics Returns a command of type `\DeclareDocumentCommand`[5],

`\CcoolOption` `\CcoolOption[...`⟨*key*$_i$⟩`|`⟨*key*$_i$⟩`=`⟨*val*$_i$⟩`,...]`

where ⟨*key*$_i$⟩ is either of `And`, `Expans`, `File`, `Inner`, `Param`, `Outer`, `Separ`, and `Write`.

Semantics Modify the behavior of `\Ccool`

`And`

Also see Part IV `And`

Semantics Sets the translation of *and* in language ⟨*key*⟩ to ⟨*val*⟩

Syntax ⟨*keyval list*⟩

`Expans`

Also see Core feature and Part IV `Expans`

Syntax `eo|ee|ex|xo|xe|xx`

`File`

Also see Part I Write and Part IV `File`

Syntax ⟨*path*⟩

`Inner`

Also see Process the *val*$_i$s and Part IV `Inner`

Syntax ⟨*code*⟩, with `####1` as the *placeholder*

`Param`

Also see Parameterize the *key*$_i$s, and Part IV `Param`

Syntax ⟨*token list*⟩

`Outer`

Also see Expand the *val*$_i$s, and Part IV `Outer`

Default `\ensuremath{####1}`

Syntax ⟨*code*⟩, with `####1` as the *placeholder*

`Separ`

Also see Expand the *val*$_i$s; Listing 7; and Part IV `Separ`

Other Default behavior depends on whether babel and amsmath are loaded

Syntax That of *separators* in [2, Section 8 of l3seq]

`Write`

Also see Part I Write and Part IV `Write`

Syntax `\BooleanFalse|\BooleanTrue`

`\CcoolRead` `\CcoolRead[`⟨*path*⟩`]`

Also see Part IV `\CcoolRead`

Semantics

1. Reads the definitions in ⟨*path*⟩.
2. Writes to `ccool.log`: 'read from ⟨*path*⟩'

`\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: `$`⟨key_i⟩`<x$`.
Do: `$\`⟨key_i⟩`{<}x$`

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
```

2020/06/03 v2.9 cool — A key-value document command parser

Listing 2. "Let ℕ and ℝ denote. . . " (start of the tutorial)

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

Let ℕ and 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 ℕ and 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 ℕ and 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 ℕ and ℝ denote the natural and real numbers.

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

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

Let ℕ and ℝ denote the natural and real numbers.

Listing 7. Language and mode

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

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*[a])

[a] [bug]: Removing the closing \CcoolOption subsequently causes inconsistent separators between text and math mode (case replicated in uncommented form in dtx)

```
% ^^A--->
\CcoolOption[ Separ={{\ \char`@\ }{\ \%\ }{\ \char`@\ }} ]
\Ccool{ 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 = {\%} }[.\{l\}.\{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
```

```
{\{}\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 11. 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} } }
[$\Field\subset \PowerSet$.]
{}
% ^^A<---
\CcoolClear
\CcoolOption
```

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

Listing 12. Listing 11 read from file

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

Ω $\mathcal{F}$ $\mathcal{P}$

Listing 13. Mittelwertsatz für n Variable[4, 17.3]

```
\CcoolOption[ Write = \BooleanTrue ]
% ^^A--->
\selectlanguage{german}
\newtheorem{theorem}{Theorem}
\AfterEndEnvironment{theorem}{\CcoolHook}
\Ccool c{\mathbb{#1}}
{ N = { N } , R = { R } }+[]
{ Grad = { \operatorname{grad} } }+
[\begin{theorem}
  [Mittelwertsatz f\"ur $n$ Variable]Es~sei~]
  { OffMenge = {D}, Ci = {C^{1}}, Strecke = { \left[x_0,x\right] } }+
  [$n\in\N$,~$\OffMenge\subseteq\N^n$ eine offene Menge und
  $f\in\Ci(\OffMenge,\R)$.
  Dann gibt es auf jeder Strecke $\Strecke\subset\OffMenge$ einen Punkt
  $\xi\in\Strecke$,~]
  { Steig = { \frac{ f(x)-f(x_0) }{ x-x_0 } }, Punkt = { \xi } }+
  [so dass gilt
  \begin{equation*}
    \Steig = \Grad f(\Punkt)^{\top}
  \end{equation*}
\end{theorem}]
{}
(Check: $\N$, $\Punkt$)
% ^^A<---
\CcoolClear
\CcoolOption
```

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*

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

(Check: $\mathbb{N}$, ξ)

Listing 14. Listing 13 read from file

```
% ^^A--->
\CcoolRead \tab $\N$ $\R$ $\OffMenge$ $\Ci$ $\Strecke$
% ^^A<---
\CcoolClear
```

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

Listing 15. Families of polynomial functions

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

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

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

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

```
\CcoolOption[ Write = \BooleanTrue ]
% ^^A--->
\selectlanguage{french}
\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}}
}
[$\Poly[n]{\Corps}$ et $\Poly{\Corps}$, les familles de polyn\^omes 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<---
\CcoolClear
\CcoolOption
```

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

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

Listing 19. Fonction et fonctionelle

```
\CcoolOption[ 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 $\ApplyOp{S}{f}$ 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 $f\EvalAt{t}$, $\ApplyOp{S}{f}$
% ^^A<---
\CcoolClear
```

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

Listing 21. CUSUM statistic[6]

```
\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{
        $\LogWald_{\Time}\EvalAt{ \Scale } = \Scale\Process_{\Time} -
    \frac{1}{2}\Scale^2\Time$;
        $\LogWaldInf_{\Time}\EvalAt{ \Scale } = \inf_{ 0\le s \le \Time
    }\CUSUM_{s} \EvalAt{ \Scale }$.
      }
    \item{
        $\CUSUM_{\Time}\EvalAt{ \Scale } = \LogWaldInf_{\Time}\EvalAt{
    \Scale } - \LogWald_{\Time}\EvalAt{ \Scale }\ge0$,
        which is the CUSUM statistic process.
      }
    \item{
        $\CUSUMst \EvalAt{ \Scale, \LogWaldInf } = \inf\left[ \Time \ge 0
    \SuchThat \CUSUM_{\Time}\EvalAt{\Scale} \ge \LogWaldInf \right]$,
        which is the CUSUM stopping time.
      }
    \end{enumerate}
  \end{definition}\par]{}

  (Check: $\Scale$, $\CUSUM$)
  % ^^A<---
  \CcoolClear
  \CcoolOption
%
```

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. $u_t(\lambda) = \lambda\xi_t - \frac{1}{2}\lambda^2 t$; $m_t(\lambda) = \inf_{0\le s\le t} y_s(\lambda)$.
2. $y_t(\lambda) = m_t(\lambda) - u_t(\lambda) \ge 0$, *which is the CUSUM statistic process.*
3. $T_c(\lambda, m) = \inf\,[t \ge 0;\ y_t(\lambda) \ge m]$, *which is the CUSUM stopping time.*

(Check: λ, y)

Listing 22. Listing 21 read from file

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

t ξ λ $\mathcal{R}$ ν u T_c y m

Part III
Other

1 Acknowledgment

This work has benefited from Q&A's from the LaTeXcommunity[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 hinsdight 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 LaTeXengine

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 LaTeX3 Project Team *The LaTeX3 interfaces*, 2019, http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/interface3.pdf

[3] The LaTeX3 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 LaTeX3 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 LaTeX 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

General: Initial version 20

v1.1

General: Add: `Save` 20

Add: Listing 1., 2., 3., 4., 6., and 9. 20

Add:`\OopsRestore` 20

Add:`\OopsTest` 20

Delete: Listing 1-5 from v1.0 20

Fix: apparent anomaly in v1.0's Listing 4, see Listing 9 20

Rearrange: much of the implementation 20

Replace: `\OopsOptions` by `\OopsOption` . . . 20

Replace: `{`⟨*kvl*$_2$⟩`}` by <kvl_2> given that option type `G` not recommended[5] 20

Replace: `GenericObject` by `Name` . 20

Replace: `Separators` by `Separ` . . . 20

v1.2

General: Add: optional `*`to `\OopsNew` as instruction to expand kvl$_1$ 20

Delete: `\OopsTest` 20

Delete: ⟨*kvl*$_2$⟩ and ⟨*code*$_2$⟩ 20

Delete: Listing 2-3 from v1.1. 20

Replace: `\OopsClear{`⟨*tl*$_2$⟩`}` by `\OopsClear[`⟨*keyval list*⟩`]` 20

Replace: `\Restore` by `\Read` 20

Replace: `\Save` by `\Write` 20

v1.3

General: Replace: `\OopsNew` by `\Oops` 20

Replace: `{`$\langle tl_2\rangle$`}` and `[`$\langle tl_2\rangle$`]` by `<`$\langle tl_2\rangle$`>` 20

v1.4

General: Add: section 9 20

Add: `\OopsDebug` 20

Add: `\OopsHook` 20

Add: `Expans` (for debugging' sake, but...) 20

Add: Listing 1., 2., and 3. 20

Add:optional `+`to `\OopsNew` to make side effects presist beyond local group 20

Delete: Listing 1., and 2. 20

Replace: `s{{`$\langle tl_3\rangle$`}{`$\langle tl_4\rangle$`}{`$\langle tl_5\rangle$`}}` by `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

v1.5

General: Add: `File` 20

Delete: dependence on datetime . . 20

v1.6

General: Add: Listing showing part of the preamble 20

Rename: `\OopsClear` to `\CcoolClear` 20

Rename: `\OopsDebug` to `\CcoolDebug` 20

Rename: `\OopsHook` to `\CcoolHook` 20

Rename: `\OopsOption` to `\CcoolOption` 20

Rename: `\OopsRead` to `\CcoolRead` 20

Rename: `\Oops` to `\Ccool` 20

Rename: oops to ccool (better describes the purpose) 20

v1.7

General: Add: Legends to listings . . . 20

Add: Listing 21 (CUSUM) 20

Delete: `\CcoolDebug` 20

Delete: Listing 5 from v1.6 20

v1.8

General: Add: `\CcoolVers` 20

Add: `\CcoolLambda` 20

Add: Listing 19, Listing 20 20

Add: Listing 1 20

v1.9

General: Add: support for LuaTEX . . 20

Move: from Part I to Part IV, what is now that part's section 11 20

v2.0

General: Add: support for XƎTEX . . . 20

Delete: `File`'s dependency on texosquery and `\pdfcreationdate` 20

Update: `\RequirePackage`, `\NeedsTeXFormat`'s second argument / TeX Live 2020 20

v2.1

General: Add:Listings 3, 4, 5, 6, 7, 8, and 9 20

Replace: $\langle tl_2\rangle$'s position within `\Ccool`'s argument list, from first to second. Greater versatility 20

Replace: `\CcoolLambda`'s optional integer argument (number of `m`'s) by a standard argument list 20

Replace: *global option* `Name` by `Param` 20

Replace: as the default of `Param`, `Math` by `Default` . 20

v2.2

General: Remove: % from listings . . . 20

Replace: part of the abstract's with more straighforward descriptions based on input from forum participtants 20

v2.3

General: Add: Listing 16, Listing 17, and Listing 18 20

Complete: Listing 15 20

Rearranged: `\Ccool`'s subsections. Previously, by argument. Now, by feature. 20

Remove: Listing showing part of the preamble 20

Replace: for `\Ccool`, `i{}` by `c{}` . . 20

Replace: In step *2)*, the created command's implementation, from `\ProvideDocumentCommand` to `\DeclareDocumentCommand` 20

v2.4

General: Fix: minor error in the listings (`\Real` rather than `\Reel`, hitherto unnoticed). 20

Remove: examples from Part I, `\Ccool`, as redundant with Part II Listing 2-6 20

v2.5

General: Modify: `File`, rely on erw-l3's `\erw_jobnametimestamp:` 20

Modify: behavior of Part I Expand the *val*$_i$s, rely on erw-l3's `\erw_seq_use:Nn` 20

v2.6

General: Modify: `\CcoolLambda`, rely on erw-l3's `\erw_lambda:nnn` 20

v2.7

General: Add: *global option* `And` 20

Add: Listing 7 20

Modify: `\CcoolOption`'s 'm'-type argumentby 'o'-type argument . . 20

Modify: Separ's default rely on babel and amsmath, if applicable . 20

v2.8

General: Fix: conflict between \usepackage[spanish]{babel} and Parameterize the *key$_i$*s 20

v2.9

General: Other: polish the doc 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
\⟨*key$_i$*<*tl$_2$*>⟩ *6*
\⟨*key$_i$*⟩ *5*, *8*
\⟨*key$_i$*⟩ *5*
And (option) *6*, *31*
Expans (option) *7*, *31*
File (option) *7*, *32*
Inner (option) *7*, *32*
Outer (option) *7*, *32*
Param (option) *7*, *32*
Separ (option) *7*, *33*
Write (option) *7*, *33*
\^ 108

\␣ 273, 274, 275, 281, 282, 283

A

\AtEndDocument 125

B

\begingroup 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*, 211, 218, 239
ccool internal commands:
  \__ccool_aux_inner:n 6, 7, 38
  \__ccool_aux_inner_set:n . . . . <u>4</u>, 196
  \__ccool_aux_key:N . . . . <u>17</u>, 200
  \__ccool_aux_key:n . . . . <u>13</u>, 20
  \__ccool_aux_key:w . . . . <u>9</u>, 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 . . . . <u>22</u>, 182
  \g__ccool_aux_prop . . . 26, 29, 46, 199
  \__ccool_aux_prop:N . . . . <u>44</u>, 198
  \__ccool_aux_prop:n . . . . <u>40</u>, 50
  \__ccool_aux_prop:nn . . . . <u>26</u>
  \__ccool_aux_prop:w . . . . <u>32</u>, 42
  \__ccool_aux_val:Nn . . . . <u>53</u>, 181
  \g__ccool_aux_val_seq . . . 55, 56, 187
  \__ccool_lang_and: <u>68</u>, 273, 275, 281, 283
  \__ccool_lang_and:n . . . . <u>68</u>
  \g__ccool_lang_and_prop 58, 62, 71, 73
  \c__ccool_lang_and_tl . . <u>86</u>, 347, 348
  \__ccool_lang_and_update:n . . <u>59</u>, 346
  \__ccool_log_close: . . . . <u>124</u>, 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: . . . . <u>132</u>, 416
  \g__ccool_log_open_bool . . . .
    . . . . 126, 130, 137, 152, 206
  \__ccool_log_read: . . . . <u>145</u>, 428
  \__ccool_log_read:n . . . <u>139</u>, 147, 427
  \g__ccool_log_to_tl 135, 136, 147, 149
  \__ccool_log_write:n . . . . . . <u>149</u>, 208
  \__ccool_make_ccool:nnnn . . . .
    . . . . <u>216</u>, 361, 375, 389, 403
  \__ccool_make_ccool_exp:nnn <u>179</u>, 227
  \__ccool_make_ccool_key:nnn <u>191</u>, 205
  \__ccool_make_ccool_sideeffect:nnn
    . . . . <u>203</u>, 224, 233
  \__ccool_make_key:N . . . . <u>175</u>, 201
  \__ccool_make_key:n . . . . <u>170</u>, 177
  \__ccool_make_key:Nn . . . . . . <u>160</u>, 172
  \g__ccool_option_expans_tl 32, 36, 350
  \__ccool_option_inner:n . . . . <u>249</u>, 359
  \g__ccool_option_inner_tl . . . . . .
    . . . . 249, 252, 364, 378, 392, 406
  \__ccool_option_outer:n . . . . <u>259</u>, 387

\g__ccool_option_outer_tl 259, 262, 366, 380, 394, 408
__ccool_option_param:n 254, 373
\g__ccool_option_param_tl 164, 254, 257, 330, 363, 377, 391, 405
__ccool_option_separ:n 264, 401
\c__ccool_option_separ_default_-tl 271, 279, 412, 413
\g__ccool_option_separ_tl 264, 267, 269, 365, 379, 393, 407
__ccool_prop_append:NN 286, 297
__ccool_prop_append:Nn 199, 295
__ccool_prop_append:nn 288, 292
__ccool_prop_clear_new:n . . 299, 306
__ccool_prop_clear_new_map:n 303, 332
__ccool_prop_if_exist:nTF . 193, 308
__ccool_prop_item:nn 166, 312
__ccool_prop_name:n 56, 297, 301, 310, 314, 316, 319
__ccool_prop_new:n 195, 317
__ccool_seq_from_prop:n . . . 323, 327
__ccool_seq_from_prop:NNn . . 56, 321
\CcoolClear *2*, *4*, *6*, 329
\CcoolHook *2*, *4*, *5*, *6*, 231, 334
\CcoolLambda *2*, *4*, *5*, *6*, *8*, 335
\CcoolOption *2*, *4*, *6*, 339
\CcoolRead *2*, *4*, *8*, *8*, 423
\CcoolVers *2*, *4*, *8*, *10*, 430
cs commands:
    \cs_generate_variant:Nn 7, 31, 67, 144, 159, 169, 174, 294
    \cs_gset:Npn 6, 24, 267
    \cs_new:Nn 68, 81, 84, 308, 312
    \cs_new:Npn 316
    \cs_new_protected:Nn 4, 13, 17, 22, 27, 40, 44, 53, 59, 127, 133, 139, 145, 150, 160, 170, 175, 179, 191, 203, 250, 255, 260, 265, 295, 299, 303, 317, 321
    \cs_new_protected:Npn   9, 33, 216, 286
    \cs_set:Nn 288
    \cs_set_protected:Nn 323

D

\DeclareDocumentCommand   *6*, 163, 218, 337
\def 211

E

\endgroup 212
\ensuremath 398, 399
erw commands:
    \erw_keyval_error:Nn 63
    \erw_lambda:nnn 337
    \erw_prop_keyval_parse:NNNn 61
    \erw_seq_use:Nn 186
    \erw_sys_jobnametimestamp: . 356, 357
exp commands:
    \exp_args:NNf 185
    \exp_args:NNx 162, 218
    \exp_args:No 81, 196, 301
    \exp_args:Nx 35
    \exp_last_unbraced:Nf 360, 374, 388, 402
    \exp_not:n 239
\expandafter 211
\ExplSyntaxOff 433
\ExplSyntaxOn 3

F

file commands:
    \file_input:n 141
\function *5*

G

\gappto 231

I

\IfBooleanT 225
\ifcsdef 79, 269
\IfValueT 223, 237
\IfValueTF 426
iow commands:
    \iow_close:N 125, 129
    \iow_new:N 124
    \iow_now:Nn 154
    \iow_open:Nn 136

K

keys commands:
    \l_keys_choice_tl 350
    \keys_define:nn 344
    \keys_set:nn 342

L

\languagename 81

M

msg commands:
    \msg_error:nnn 157
    \msg_new:nnn 243, 246
    \msg_warning:nnn 75

N

\NB *18*, *19*
\NewDocumentCommand *2*, *10*, 329, 334, 339, 423, 430

O

options:
    And *6*, *31*

Expans *7*, *31*
File *7*, *32*
Inner *7*, *32*
Outer *7*, *32*
Param *7*, *32*
Separ *7*, *33*
Write *7*, *33*

P

prop commands:
\prop_clear_new:N 301
\prop_gclear_new:N 46
\prop_gput:Nnn 29, 64, 290
\prop_if_exist:NTF 310
\prop_if_in:NnTF 70
\prop_item:Nn 73, 290, 314, 325
\prop_map_function:NN 292
\prop_new:N 26, 58, 319
\ProvideDocumentCommand 335

Q

quark commands:
\q_stop 9, 15, 33, 42

R

\Real *1*, *5*
\Reel *1*

S

seq commands:
\seq_gclear_new:N 19, 55
\seq_gput_right:Nn 11, 325
\seq_if_empty:NTF 47
\seq_map_function:NN . . . .
. . . . 20, 50, 177, 306, 327
\seq_set_from_clist:Nn 197, 305

T

\text 273, 274, 275
tl commands:
\c_empty_tl 48, 194, 214, 236, 334
\tl_const:Nn 86, 271, 279
\tl_gset:Nn . . . 135, 252, 257, 262, 354
\tl_gset_eq:NN 350
\tl_log:n 142, 155
\tl_new:N 32, 132, 149, 249, 254, 259, 264
\tl_trim_spaces:n 11, 37, 38

U

use commands:
\use:N 36, 432
\usepackage *5*

X

\X *4*

Part IV
Implementation

1 Opening

```
⟨*package⟩
⟨@@=ccool⟩
\ExplSyntaxOn
```

2 aux

__ccool_aux_inner_set:n

#1 : ⟨*code*⟩

```
\cs_new_protected:Nn \__ccool_aux_inner_set:n
{
  \cs_gset:Npn \__ccool_aux_inner:n ##1 {#1}
  \cs_generate_variant:Nn \__ccool_aux_inner:n { e }
}
```

(*End definition for* `\__ccool_aux_inner_set:n`.)

__ccool_aux_key:w

#1 : ⟨ *key* ⟩
#2 : ⟨ *value* ⟩

```
\cs_new_protected:Npn \__ccool_aux_key:w #1 = #2 \q_stop
{
  \seq_gput_right:Nx \g__ccool_aux_key_seq { \tl_trim_spaces:n{#1} }
}
```

(*End definition for* `\__ccool_aux_key:w`.)

__ccool_aux_key:n

#1 : ⟨ *key* = *value* ⟩

```
\cs_new_protected:Nn \__ccool_aux_key:n
{
  \__ccool_aux_key:w #1 \q_stop
}
```

(*End definition for* `\__ccool_aux_key:n`.)

__ccool_aux_key:N

#1 : ⟨ *seq* ⟩

```
\cs_new_protected:Nn \__ccool_aux_key:N
{
  \seq_gclear_new:N \g__ccool_aux_key_seq
  \seq_map_function:NN #1 \__ccool_aux_key:n
}
```

(*End definition for* `\__ccool_aux_key:N`.)

__ccool_aux_outer_set:n

#1 : ⟨ *inline code* ⟩

```
\cs_new_protected:Nn \__ccool_aux_outer_set:n
{
  \cs_gset:Npn \__ccool_aux_outer:n ##1 {#1}
}
```

(*End definition for* `\__ccool_aux_outer_set:n`.)

`\__ccool_aux_prop:nn`

```
\prop_new:N \g__ccool_aux_prop
\cs_new_protected:Nn \__ccool_aux_prop:nn
{
  \prop_gput:Nnn \g__ccool_aux_prop{#1}{#2}
}
\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* ⟩

```
\tl_new:N \g__ccool_option_expans_tl
\cs_new_protected:Npn \__ccool_aux_prop:w #1 = #2 \q_stop
{
  \exp_args:Nx
  \use:c{__ccool_aux_prop:\g__ccool_option_expans_tl}
  { \tl_trim_spaces:n{#1} }
  { \__ccool_aux_inner:n{ \tl_trim_spaces:n{#2} } }
}
```

(*End definition for* `\__ccool_aux_prop:w`.)

`\__ccool_aux_prop:n` #1 : ⟨ *key* = *value* ⟩

```
\cs_new_protected:Nn \__ccool_aux_prop:n
{
  \__ccool_aux_prop:w #1 \q_stop
}
```

(*End definition for* `\__ccool_aux_prop:n`.)

`\__ccool_aux_prop:N` #1 : ⟨*keyval list*⟩

```
\cs_new_protected:Nn \__ccool_aux_prop:N
{
  \prop_gclear_new:N \g__ccool_aux_prop
  \seq_if_empty:NTF #1
  { \c_empty_tl }
  {
    \seq_map_function:NN #1 \__ccool_aux_prop:n
  }
}
```

(*End definition for* `\__ccool_aux_prop:N`.)

`\__ccool_aux_val:Nn` #1 : ⟨ *seq* ⟩
#2 : ⟨ *tl var name* ⟩

```
\cs_new_protected:Nn \__ccool_aux_val:Nn
{
  \seq_gclear_new:N \g__ccool_aux_val_seq
  \__ccool_seq_from_prop:NNn \g__ccool_aux_val_seq #1 { \__ccool_prop_name:n{#2} }
}
```

(*End definition for* `\__ccool_aux_val:Nn`.)

3 lang

```
\prop_new:N \g__ccool_lang_and_prop
```

\__ccool_lang_and_update:n

```
\cs_new_protected:Nn \__ccool_lang_and_update:n
{
  \erw_prop_keyval_parse:NNNn
  \g__ccool_lang_and_prop
  \erw_keyval_error:Nn
  \prop_gput:Nnn
  { #1 }
}
\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:

```
\cs_new:Nn \__ccool_lang_and:n
{
  \prop_if_in:NnTF
  \g__ccool_lang_and_prop
  {#1}
  {\prop_item:Nn\g__ccool_lang_and_prop{#1}}
  {
    \msg_warning:nnn{__ccool}{lang_and}{#1}
    \__ccool_lang_and:n{english}
  }
}
\ifcsdef{languagename}
{
  \cs_new:Nn \__ccool_lang_and:{\exp_args:No\__ccool_lang_and:n{\languagename}}
}
{
  \cs_new:Nn \__ccool_lang_and:{english}
}
```

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

\\c__ccool_lang_and_tl (*Note*[1])

```
\tl_const:Nn \c__ccool_lang_and_tl
{
%^^A https://www.overleaf.com/learn/latex/International_language_support
  afrikaans=en,
  basque=eta,
  catalan=i,
  croatian=i,
  czech=a,
  danish=og,
  dutch=en,
  english=and,
  esperanto=kaj,
  estonian=ja,
```

[1] [todo]: Non latin-alphabet languages

```
  finnish=ja,
  french=et,
  galician=e,
  german=und,
  hungarian=\'es,
  icelandic=og,
  indonesian=dan,
  irish=agus,
  italian=e,
  kurmanji=\^u,
  latin=et,
  latvian=un,
  lithuanian=ir,
  ngerman=und,
  polish=i,
  portuguese=e,
  romanian=\c{s}i,
  slovak=a,
  spanish=y,
  swedish=och,
  swissgerman=und,
  turkish=ve,
  turkmen=we,
  welsh=a
}
```

(*End definition for* `\c__ccool_lang_and_tl`.)

4 log

`\__ccool_log_close:`

```
\iow_new:N \g__ccool_log_iow
\AtEndDocument{\iow_close:N \g__ccool_log_iow}
\bool_set_false:N \g__ccool_log_open_bool
\cs_new_protected:Nn \__ccool_log_close:
{
  \iow_close:N \g__ccool_log_iow
  \bool_gset_false:N \g__ccool_log_open_bool
}
```

(*End definition for* `\__ccool_log_close:`.)

`\__ccool_log_open:`

```
\tl_new:N \g__ccool_log_file_tl
\cs_new_protected:Nn \__ccool_log_open:
{
  \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
  \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
  \bool_gset_true:N \g__ccool_log_open_bool
}
```

(*End definition for* `\__ccool_log_open:`.)

__ccool_log_read:n #1 : ⟨*path*⟩

```
\cs_new_protected:Nn \__ccool_log_read:n
{
  \file_input:n{#1}
  \tl_log:n{read~from~#1}
}
\cs_generate_variant:Nn \__ccool_log_read:n { e }
```

(*End definition for* __ccool_log_read:n.)

__ccool_log_read:

```
\cs_new_protected:Nn \__ccool_log_read:
{
  \__ccool_log_read:e{\g__ccool_log_to_tl}
}
```

(*End definition for* __ccool_log_read:.)

__ccool_log_write:n

```
\tl_new:N \g__ccool_log_to_tl
\cs_new_protected:Nn \__ccool_log_write:n
{
  \bool_if:nTF{ \g__ccool_log_open_bool }
  {
    \iow_now:Nn \g__ccool_log_iow {#1}
    \tl_log:n{ write~to~#1 }
  }
  { \msg_error:nnn{ __ccool }{ iow }{ \g__ccool_log_iow }  }
}
\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* ⟩

```
\cs_new_protected:Nn \__ccool_make_key:Nn
{
  \exp_args:NNx
  \DeclareDocumentCommand{#1}
  { D<>{\g__ccool_option_param_tl} }
  {
    \__ccool_prop_item:nn{##1}{#2}
  }
}
\cs_generate_variant:Nn \__ccool_make_key:Nn {c}
```

(*End definition for* __ccool_make_key:Nn.)

__ccool_make_key:n #1 : ⟨ *key* ⟩

```
\cs_new_protected:Nn \__ccool_make_key:n
{
  \__ccool_make_key:cn{#1}{#1}
}
\cs_generate_variant:Nn \__ccool_make_key:n { e }
```

(*End definition for* `\__ccool_make_key:n`*.*)

__ccool_make_key:N #1 : ⟨ *seq* ⟩

```
\cs_new_protected:Nn \__ccool_make_key:N
{
  \seq_map_function:NN #1 \__ccool_make_key:e
}
```

(*End definition for* `\__ccool_make_key:N`*.*)

6 make_ccool

__ccool_make_ccool_exp:nnn

```
\cs_new_protected:Nn \__ccool_make_ccool_exp:nnn
{
  \__ccool_aux_val:Nn \g__ccool_aux_key_seq {#1}
  \__ccool_aux_outer_set:n{#3}
  \__ccool_aux_outer:n
  {
    \exp_args:NNf
    \erw_seq_use:Nn
    \g__ccool_aux_val_seq
    {#2}
  }
}
```

(*End definition for* `\__ccool_make_ccool_exp:nnn`*.*)

__ccool_make_ccool_key:nnn

```
\cs_new_protected:Nn \__ccool_make_ccool_key:nnn
{
  \__ccool_prop_if_exist:nTF{#1}
  { \c_empty_tl }
  { \__ccool_prop_new:n{#1} }
  \exp_args:No \__ccool_aux_inner_set:n{#2}
  \seq_set_from_clist:Nn \g__ccool_aux_keyval_seq {#3}
  \__ccool_aux_prop:N \g__ccool_aux_keyval_seq
  \__ccool_prop_append:Nn \g__ccool_aux_prop {#1}
  \__ccool_aux_key:N \g__ccool_aux_keyval_seq
  \__ccool_make_key:N \g__ccool_aux_key_seq
}
```

(*End definition for* `\__ccool_make_ccool_key:nnn`*.*)

__ccool_make_ccool_sideeffect:nnn [9]

```
\cs_new_protected:Nn \__ccool_make_ccool_sideeffect:nnn
{
  \__ccool_make_ccool_key:nnn{#1}{#2}{#3}
  \bool_if:nTF{ \g__ccool_log_open_bool }
  {
    \__ccool_log_write:n
    {
      \begingroup
      \def \__ccool_log_entry { \Ccool<#1>c{#2}{#3} } \expandafter
      \endgroup \__ccool_log_entry
    }
  }{\c_empty_tl}
}
```

(*End definition for* `\__ccool_make_ccool_sideeffect:nnn`.)

__ccool_make_ccool:nnnn

#1 : ⟨ *token list* ⟩
#2 : ⟨ seq_1 ⟩
#3 : ⟨ seq_2 ⟩
#4 : ⟨ *prop* ⟩

```
\cs_new_protected:Npn \__ccool_make_ccool:nnnn #1 #2 #3 #4
{
  \exp_args:NNx \DeclareDocumentCommand \Ccool
  {%^^A      2         3    4 5  6           7   8    9
    +o D<>{#1} E{ c }{{#2}} m t+ s E{ s c }{{#3}{#4}} +o
  }
  {
    \IfValueT{##1}{##1}
    \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
    \IfBooleanT{##6}
    {
      \__ccool_make_ccool_exp:nnn{##2}{##7}{##8}
    }
    \bool_if:nTF{##5}
    {
      \gappto{\CcoolHook}
      {
        \__ccool_make_ccool_sideeffect:nnn{##2}{##3}{##4}
      }
    }
    {\c_empty_tl}
    \IfValueT{##9}
    {
      \exp_not:n{ \Ccool[##9] }
    }
  }
}
```

(*End definition for* `\__ccool_make_ccool:nnnn`.)

7 msg

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

8 option

__ccool_option_inner:n

#1 : ⟨*code*⟩

```
\tl_new:N \g__ccool_option_inner_tl
\cs_new_protected:Nn \__ccool_option_inner:n
{
  \tl_gset:Nn \g__ccool_option_inner_tl {#1}
}
```

(*End definition for* `\__ccool_option_inner:n`.)

__ccool_option_param:n

#1 : ⟨*token list*⟩

```
\tl_new:N \g__ccool_option_param_tl
\cs_new_protected:Nn \__ccool_option_param:n
{
  \tl_gset:Nn \g__ccool_option_param_tl{#1}
}
```

(*End definition for* `\__ccool_option_param:n`.)

__ccool_option_outer:n

#1 : ⟨ *inline code* ⟩

```
\tl_new:N \g__ccool_option_outer_tl
\cs_new_protected:Nn \__ccool_option_outer:n
{
  \tl_gset:Nn \g__ccool_option_outer_tl {#1}
}
```

(*End definition for* `\__ccool_option_outer:n`.)

__ccool_option_separ:n

#1 : {⟨ tl_1 ⟩}{⟨ tl_2 ⟩}{⟨ tl_3 ⟩}

```
\tl_new:N \g__ccool_option_separ_tl
\cs_new_protected:Nn \__ccool_option_separ:n
{
  \cs_gset:Npn \g__ccool_option_separ_tl {#1}
}
```

(*End definition for* `\__ccool_option_separ:n`.)

\g__ccool_option_separ_tl

```
\ifcsdef{text}
{
  \tl_const:Nn \c__ccool_option_separ_default_tl
  {
    { \text{{\ }\__ccool_lang_and:{\ }} }
    { \text{,{\ }} }
    { \text{,{\ }\__ccool_lang_and:{\ }} }
  }
}
{
  \tl_const:Nn \c__ccool_option_separ_default_tl
  {
    { {\ }\__ccool_lang_and:{\ } }
    { ,{\ } }
    { ,{\ }\__ccool_lang_and:{\ } }
  }
}
```

(*End definition for* `\g__ccool_option_separ_tl`*.*)

9 prop

`\__ccool_prop_append:NN`

#1 : ⟨ *prop*$_1$ ⟩
#2 : ⟨ *prop*$_2$ ⟩

```
\cs_new_protected:Npn \__ccool_prop_append:NN #1 #2
{
  \cs_set:Nn \__ccool_prop_append:nn
  {
    \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
  }
  \prop_map_function:NN #2 \__ccool_prop_append:nn
}
\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* ⟩

```
\cs_new_protected:Nn \__ccool_prop_append:Nn
{
  \__ccool_prop_append:cN{ \__ccool_prop_name:n {#2} } #1
}
```

(*End definition for* `\__ccool_prop_append:Nn`*.*)

`\__ccool_prop_clear_new:n`

#1 : ⟨ *tl var name* ⟩

```
\cs_new_protected:Nn \__ccool_prop_clear_new:n
{
  \exp_args:No \prop_clear_new:c{ \__ccool_prop_name:n {#1} }
}
```

(*End definition for* `\__ccool_prop_clear_new:n`*.*)

`\__ccool_prop_clear_new_map:n`

#1 : ⟨ *keyval list* ⟩

```
\cs_new_protected:Nn \__ccool_prop_clear_new_map:n
{
  \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
  \seq_map_function:NN \g__ccool_aux_key_seq \__ccool_prop_clear_new:n
}
```

(*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 : $\langle tl_3 \rangle$

```
\cs_new:Nn \__ccool_prop_if_exist:nTF
{
  \prop_if_exist:cTF{ \__ccool_prop_name:n {#1} }{#2}{#3}
}
```

(*End definition for* `\__ccool_prop_if_exist:nTF`*.*)

__ccool_prop_item:nn

#1 : $\langle$ *tl var name* $\rangle$
#2 : $\langle$ *key* $\rangle$

```
\cs_new:Nn \__ccool_prop_item:nn
{
  \prop_item:cn { \__ccool_prop_name:n {#1} } {#2}
}
```

(*End definition for* `\__ccool_prop_item:nn`*.*)

__ccool_prop_name:n

#1 : $\langle$ *tl var name* $\rangle$

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

(*End definition for* `\__ccool_prop_name:n`*.*)

__ccool_prop_new:n

#1 : $\langle$ *tl var name* $\rangle$

```
\cs_new_protected:Nn \__ccool_prop_new:n
{
  \prop_new:c{ \__ccool_prop_name:n {#1} }
}
```

(*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* $\rangle$

```
\cs_new_protected:Nn \__ccool_seq_from_prop:NNn
{
  \cs_set_protected:Nn \__ccool_seq_from_prop:n
  {
    \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
  }
  \seq_map_function:NN #2 \__ccool_seq_from_prop:n
}
```

(*End definition for* `\__ccool_seq_from_prop:NNn`*.*)

11 Front-end

\CcoolClear

```
\NewDocumentCommand{ \CcoolClear }
{ D<>{\g__ccool_option_param_tl} }
{
  \__ccool_prop_clear_new_map:n{#1}
}
```

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

\CcoolHook

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

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

\CcoolLambda (*Note*[2])

```
\ProvideDocumentCommand \CcoolLambda { O{m} m }
{
  \erw_lambda:nnn \DeclareDocumentCommand { #1 } { #2 }
}
```

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

\CcoolOption (*Note*[3]) (*Note*[4])

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

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

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

And

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

Expans

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

File

```
File .code:n = {
  \tl_gset:Nx \g__ccool_log_file_tl{#1}
},
File .default:n = { \erw_sys_jobnametimestamp: },
File .initial:n = { \erw_sys_jobnametimestamp: },
```

Inner

```
Inner .code:n={
  \__ccool_option_inner:n{#1}
  \exp_last_unbraced:Nf
  \__ccool_make_ccool:nnnn
  {
    { \g__ccool_option_param_tl }
    { \g__ccool_option_inner_tl }
    { \g__ccool_option_separ_tl }
    { \g__ccool_option_outer_tl }
  }
},
Inner .value_required:n = false,
Inner .default:n = {####1},
Inner .initial:n = {####1},
```

Param

```
Param .code:n={
  \__ccool_option_param:n{#1}
  \exp_last_unbraced:Nf
  \__ccool_make_ccool:nnnn
  {
    { \g__ccool_option_param_tl }
    { \g__ccool_option_inner_tl }
    { \g__ccool_option_separ_tl }
    { \g__ccool_option_outer_tl }
  }
},
Param .value_required:n = false,
Param .default:n = { Default },
Param .initial:n = { Default },
```

Outer

```
Outer .code:n={
  \__ccool_option_outer:n{#1}
  \exp_last_unbraced:Nf
  \__ccool_make_ccool:nnnn
  {
    { \g__ccool_option_param_tl }
    { \g__ccool_option_inner_tl }
    { \g__ccool_option_separ_tl }
    { \g__ccool_option_outer_tl }
  }
},
```

[2] [todo]: allow only m- or o-type arguments

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

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

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

Separar

```
Separ .code:n={
  \__ccool_option_separ:n{#1}
  \exp_last_unbraced:Nf
  \__ccool_make_ccool:nnnn
  {
    { \g__ccool_option_param_tl }
    { \g__ccool_option_inner_tl }
    { \g__ccool_option_separ_tl }
    { \g__ccool_option_outer_tl }
  }
},
Separ .value_required:n = false,
Separ .default:n = { \c__ccool_option_separ_default_tl },
Separ .initial:n = { \c__ccool_option_separ_default_tl },
```

Write

```
Write .code:n = {
  \bool_if:nTF{#1}
  {\__ccool_log_open:}
  {\__ccool_log_close:}
},
Write .value_required:n = false,
Write .default:n = \BooleanFalse,
Write .initial:n = \BooleanFalse
}
```

\CcoolRead

```
\NewDocumentCommand{\CcoolRead}
{o}
{
  \IfValueTF{#1}
  {\__ccool_log_read:e{#1}}
  {\__ccool_log_read:}
}
```

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

\CcoolVers

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

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

12 Closing

```
\ExplSyntaxOff
⟨/package⟩
```