The ccool package[*]

Erwann Rogard[†]

Released 2020/04/28

Abstract

The package ccool for LaTeX provides a *key-value* interface,`\Ccool`, to facilitate the generation of commands. Optional parameters that control the processing of the input and its expansion are set through global options to their most likely usage. This can be used 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 paramètre optionnels globaux contrôlant le traitement de ces *clé-valeur* sont fixés par défaut pour répondre aux besoins courants. Ceci peut-être utilisé pour la command-isation des conventions de notation (`\Reel` → `\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 v2.7, last revised 2020/04/28.
[†]firstname dot lastname AusTria gmail dot com

2 `\Ccool` **5**
- 2.1 Core feature . . . 5
- 2.2 Process the val_i's . . . 5
- 2.3 Append to a hook . . . 5
- 2.4 Expand the val_i's . . . 5
- 2.5 Head . . . 6
- 2.6 Tail . . . 6
- 2.7 Parameterize the key_i's . . . 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 9

1.`\CcoolVers` **9**

2. "Let ℕ and ℝ denote. . . " (start of the tutorial) 9

3. Equivalent to 2, with `\NewDocumentCommand` **9**

4. Equivalent to 3, with `\Ccool` **9**

5. Equivalent to 4, with expansion 9

6. Equivalent to 4, parameterized (end of the tutorial) 10

7. Language. 10

8. Separators. 13

9. Hello, world! (testing) 14

10. Listing 9 read from file 14

11. Probability space 15

12. Listing 11 read from file 15

13. Mittelwertsatz für n Variable. 15

14. Listing 13 read from file 16

15. Families of polynomial functions 16

16. Listing 15 read from file 17

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

16. Listing 17 read from file 17

19. Fonction et fonctionelle 18

20. Listing 19 read from file 18

21. CUSUM statistic. 18

22. Listing 21 read from file 19

III Other 21

1 Acknowledgment 21

2 Genealogy 21

3 Install 21

4 Issue 21

5 Support 21

6 Testing 21

6.1 Technicality 21

6.2 Platform 21

6.3 Engine 22

6.4 Results 22

6.5 Other 22

7 To do 22

8 References 22

Change History 23

Index **24**

IV Implementation **27**

1 Opening **27**

2 aux **27**

3 lang **29**

4 log **31**

5 make_key **32**

6 make_ccool **33**

7 msg **34**

8 option **34**

9 prop **36**

10 seq **37**

11 Front-end **37**
11.1 `\CcoolClear` 37
11.2 `\CcoolHook` 37
11.3 `\CcoolLambda` 38
11.4 `\CcoolOption` 38
11.4.1 `And` 38
11.4.2 `Expans` 38
11.4.3 `File` 38
11.4.4 `Inner` 38
11.4.5 `Param` 39
11.4.6 `Outer` 39
11.4.7 `Separ` 39
11.4.8 `Write` 39
11.5 `\CcoolRead` 40
11.6 `\CcoolVers` 40

12 Closing **40**

Part I
Usage

Convention

a) Loosely, those of [2], for example as to the meaning of ⟨*token list*⟩.

b) Those of [4], for example `[arg]` is a *'o'-type argument*.

c) $\langle X\rangle \leftarrow$ `Y`: set $\langle X\rangle$ to `Y`

d) `\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{`$\langle kvl_1\rangle$`}` defines for each $\langle key_i\rangle$`=`$\langle val_i\rangle$, the command `\`$\langle key_i\rangle$, in two steps:

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

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

where `\function` is controled by 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 way $\langle key_i\rangle$ and $\langle val_i\rangle$ are expanded.

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*$_i$'s

`\Ccool{`⟨*kvl*$_1$⟩`}*` supplements the Core feature with the expansion of the ⟨*val*$_i$⟩'s using typesetting rules controlled by *global option* `Separ` and `Outer`. The first are *separators* applied to the ⟨*val*$_i$⟩'s to form a *token list*, and the second a function applied to the latter.

They can be overriden inline by appending further `s{`⟨*separators*⟩`}` and `c{`⟨*code*$_2$⟩`}`, respectively, to the list of arguments.

2.5 Head

`\Ccool[`⟨*tl*$_1$⟩`]{`⟨*kvl*$_1$⟩`}` expands ⟨*tl*$_1$⟩ and executes the Core feature.

There may be situations where it is convenient to pass ⟨*tl*$_1$⟩ as empty.

2.6 Tail

`\Ccool{`⟨*kvl*$_1$⟩`}[`⟨*tl*$_6$⟩`]{`⟨*kvl*$_2$⟩`}` is identical to `\Ccool{`⟨*kvl*$_1$⟩`}` followed by `\Ccool[`⟨*tl*$_6$⟩`]{`⟨*kvl*$_2$⟩`}`.

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

2.7 Parameterize the *key*$_i$'s

`\Ccool<`⟨*tl*$_2$⟩`>{`⟨*kvl*$_1$⟩`}` is identical to the Core feature, except ⟨*key*$_i$⟩ is replaced by ⟨*key*$_i$`<`*tl*$_2$`>`⟩. The default value of ⟨*tl*$_2$⟩ is controlled by `Param`. In our example, ⟨*tl*$_2$⟩ 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 controlled by *global option* `File`.

`\CcoolClear`

`\CcoolClear<`⟨*tl*$_2$⟩`>{`⟨*clist*⟩`}`

Semantics Clears all `\`⟨*key*$_i$`<`*tl*$_2$`>`⟩'s

`\CcoolHook`

`\CcoolHook`

Semantics No side effect or expansion

`\CcoolLambda`

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

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

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

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

`\CcoolOption` `\CcoolOption[`⟨*keyval list*⟩`]`

where the ⟨key_i⟩'s are either of `And`, `Expans`, `File`, `Inner`, `Param`, `Outer`, `Separ`, and `Write`.

Semantics Set *global option* that control the default behavior of `\Ccool`; passing only the key's resets the behavior to the default.

`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 for `\CcoolLambda`
Do: Use only *‘m’-type* and *‘o’-type* arguments

7) Also see Part III, section 4

Part II
Listing

NB:

1. These listings depend on the `\usepackage` statements of the source file's `documentation`
2. 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
```

2020/04/28 v2.7 cool — A key-value interface for generating commands

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.

(*Note*[1])

Listing 7. Language

```
\textbf{\languagename}{:}\\
\Ccool{ X = x, Y = y }*
\selectlanguage{afrikaans}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{basque}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{catalan}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{croatian}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{czech}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{danish}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{dutch}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
% ^^A esperanto,                              % ERROR
\selectlanguage{estonian}\\
```

[1] [bug]: Some languages notably spanish incompatible

```
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{finnish}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{french}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
% ^^A galician,                           % ERROR
\selectlanguage{german}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{hungarian}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{icelandic}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{indonesian}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{irish}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{italian}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
% ^^A kurmanji,                           % ERROR
\selectlanguage{latin}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
% ^^A latvian,                            % ERROR
\selectlanguage{lithuanian}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{ngerman}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{polish}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{portuguese}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{romanian}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{slovak}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
% ^^A  \selectlanguage{spanish}                % ERROR
\selectlanguage{swedish}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{swissgerman}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{turkish}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{turkmen}\\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
\selectlanguage{welsh} \\
\noindent\textbf{\languagename}{:}\\
\CcoolOption[ Separ ]
\Ccool{ X = x, Y = y }*
```

english:
x and y
afrikaans:
x en y
basque:
x eta y
catalan:
x i y
croatian:
x i y
czech:
x a y
danish:
x og y
dutch:

x en y
estonian:
x ja y
finnish:
x ja y
french :
x et y
german:
x und y
hungarian:
x és y
icelandic:
x og y
indonesian:
x dan y
irish:
x agus y
italian:
x e y
latin:
x et y
lithuanian:
x ir y
ngerman:
x und y
polish:
x i y
portuguese:
x e y
romanian:
x şi y
slovak:
x a y
swedish:
x och y
swissgerman:
x und y
turkish:
x ve y
turkmen:
x we y
welsh:
x a y

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[3, 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*

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

(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 17, 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[5]

```
\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:

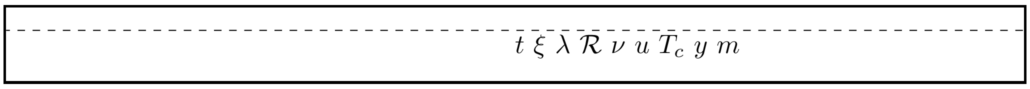
Definition 1 . *Let* $\lambda \in \mathcal{R}$ *and* $\nu \in \mathcal{R}^+$. *Define the following processes:*

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. $T_c(\lambda, m) = \inf\left[t \ge 0;\ y_t(\lambda) \ge m\right]$, *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[6][9]. 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, `$tex ccool.dtx`)
2) Put the generated `ccool.sty` in the search path of the LaTeXengine

4 Issue

Look for `NOTE` or `\NB{bug}` 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[7]. Instead, one can visually check Part II, as generated in section 3 on one's own machine, against that of the repository for the same version.

6.2 Platform

i)
```
Linux laptop 4.15.0-20-generic #21-Ubuntu SMP Tue Apr 24
↪  06:16:15 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux
```

6.3 Engine

a) `pdfTeX 3.14159265-2.6-1.40.20 (TeX Live 2019)`

b) `pdfTeX 3.14159265-2.6-1.40.21 (TeX Live 2020)`

c) `LuaHBTeX, Version 1.12.0 (TeX Live 2020)`

d) `XeTeX 3.14159265-2.6-0.999992 (TeX Live 2020)`

6.4 Results

1) ccool v1.8 compiles satisfactorily on platform *i)* and engine *a)*

2) ccool v1.8 compiles satisfactorily on platform *i)* and engine *b)*

3) ccool v1.9 compiles satisfactorily on platform *i)* and engines *b)* and *c)*

4) ccool v2.0 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

5) ccool v2.1 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

6) ccool v2.3 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

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

6.5 Other

Check [5] for testing ccool with llncs

7 To do

Look for `NOTE` or `\NB{`todo|abandon`}` 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] Thomas F. Sturm *The tcolorbox package*, 2019, http://www.texdoc.net/texmf-dist/doc/latex/tcolorbox/tcolorbox.pdf

[4] The LaTeX3 Project Team *The xparse package*, 2020, http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf

[5] Erwann Rogard and Olympia Hadjiliadis *Typesetting a math thesis with ccool*, 2020, https://github.com/rogard/ccool/blob/master/thesis.pdf

[6] https://tex.stackexchange.com/users/112708/erwann?tab=questions

[7] @joseph-wright's answer to "Checking a function's expansion against a string", `https://tex.stackexchange.com/a/534100`

[8] @frougon's answer to "Journaling calls to a function []", `https://tex.stackexchange.com/a/536620`

[9] `\Ccool`, extension à LaTeX à vocation mathématique, `http://forum.mathematex.net/latex-f6/ccool-extension-latex-a-vocation-mathematique-t17314.html`

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[4] . . . 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: `{`⟨*tl*$_2$⟩`}` and `[`⟨*tl*$_2$⟩`]` by `<`⟨*tl*$_2$⟩`>` . . . 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{{`⟨*tl*$_3$⟩`}{`⟨*tl*$_4$⟩`}{`⟨*tl*$_5$⟩`}}` by `s{{`⟨*tl*$_3$⟩`}|{`⟨*tl*$_3$⟩`}{`⟨*tl*$_4$⟩`}|{`⟨*tl*$_3$⟩`}{`⟨*tl*$_4$⟩`}{`⟨*tl*$_5$⟩`}}` . . . 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: ⟨*tl*$_2$⟩'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 participants 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: `Separ`'s default rely on babel and amsmath, if applicable . 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

`\'` 143
`\`⟨*key$_i$*`<`*tl$_2$*`>`⟩ *3*
`\`⟨*key$_i$*⟩ *2*, *5*
`\`⟨*key$_i$*⟩ *2*
`And` (option) *3*, *32*
`Expans` (option) *4*, *32*
`File` (option) *4*, *32*
`Inner` (option) *4*, *32*
`Outer` (option) *4*, *33*
`Param` (option) *4*, *33*
`Separ` (option) *4*, *33*
`Write` (option) *4*, *33*
`\^` 148
`\␣` 323, 324, 325, 331, 332, 333

A

`\AtEndDocument` 165

B

`\begingroup` 250
bool commands:
  `\bool_gset_false:N` 170
  `\bool_gset_true:N` 177
  `\bool_if:nTF` 192, 246, 269, 465
  `\bool_set_false:N` 166
`\BooleanFalse` *4*, 470, 471

\BooleanTrue *3*, *4*

C

\c 155
\Ccool *1*, *2*, *3*, *6*, *20*, 251, 258, 279
ccool internal commands:
- __ccool_aux_inner:n 6, 7, 38
- __ccool_aux_inner_set:n 4, 236
- __ccool_aux_key:N 17, 240
- __ccool_aux_key:n 13, 20
- __ccool_aux_key:w 9, 15
- \g__ccool_aux_key_seq 11, 19, 221, 241, 355, 356
- \g__ccool_aux_keyval_seq 237, 238, 240
- __ccool_aux_outer:n 24, 223
- __ccool_aux_outer_set:n 22, 222
- \g__ccool_aux_prop 26, 29, 46, 239
- __ccool_aux_prop:N 44, 238
- __ccool_aux_prop:n 40, 50
- __ccool_aux_prop:nn 26
- __ccool_aux_prop:w 32, 42
- __ccool_aux_val:Nn 53, 221
- \g__ccool_aux_val_seq 55, 56, 227
- __ccool_lang_and: 68, 323, 325, 331, 333
- __ccool_lang_and:n 68
- \g__ccool_lang_and_prop 58, 62, 71, 73
- \c__ccool_lang_and_tl 86, 397, 398
- __ccool_lang_and_update:n 59, 396
- __ccool_log_close: 164, 467
- __ccool_log_entry 251, 252
- \g__ccool_log_file_tl 172, 175, 404
- \g__ccool_log_iow 164, 165, 169, 176, 194, 197
- __ccool_log_open: 172, 466
- \g__ccool_log_open_bool 166, 170, 177, 192, 246
- __ccool_log_read: 185, 478
- __ccool_log_read:n 179, 187, 477
- \g__ccool_log_to_tl 175, 176, 187, 189
- __ccool_log_write:n 189, 248
- __ccool_make_ccool:nnnn 256, 411, 425, 439, 453
- __ccool_make_ccool_exp:nnn 219, 267
- __ccool_make_ccool_key:nnn 231, 245
- __ccool_make_ccool_sideeffect:nnn 243, 264, 273
- __ccool_make_key:N 215, 241
- __ccool_make_key:n 210, 217
- __ccool_make_key:Nn 200, 212
- \g__ccool_option_expans_tl 32, 36, 400
- __ccool_option_inner:n 285, 409
- \g__ccool_option_inner_tl 287, 291, 414, 428, 442, 456
- __ccool_option_outer:n 303, 437
- \g__ccool_option_outer_tl 305, 309, 416, 430, 444, 458
- __ccool_option_param:n 293, 423
- \g__ccool_option_param_tl 204, 295, 301, 380, 413, 427, 441, 455
- __ccool_option_separ:n 311, 451
- \c__ccool_option_separ_default_-tl 321, 329, 462, 463
- \g__ccool_option_separ_tl 313, 317, 319, 415, 429, 443, 457
- __ccool_prop_append:NN 336, 347
- __ccool_prop_append:Nn 239, 345
- __ccool_prop_append:nn 338, 342
- __ccool_prop_clear_new:n 349, 356
- __ccool_prop_clear_new_map:n 353, 382
- __ccool_prop_if_exist:nTF 233, 358
- __ccool_prop_item:nn 206, 362
- __ccool_prop_name:n 56, 347, 351, 360, 364, 366, 369
- __ccool_prop_new:n 235, 367
- __ccool_seq_from_prop:n 373, 377
- __ccool_seq_from_prop:NNn 56, 371

\CcoolClear *3*, 379
\CcoolHook *3*, 271, 384
\CcoolLambda *3*, 385
\CcoolOption *3*, 389
\CcoolRead *5*, *5*, 473
\CcoolVers *5*, *6*, 480
cs commands:
- \cs_generate_variant:Nn 7, 31, 67, 184, 199, 209, 214, 344
- \cs_gset:Npn 6, 24, 313
- \cs_new:Nn 68, 81, 84, 293, 358, 362
- \cs_new:Npn 366
- \cs_new_protected:Nn 4, 13, 17, 22, 27, 40, 44, 53, 59, 167, 173, 179, 185, 190, 200, 210, 215, 219, 231, 243, 285, 303, 311, 345, 349, 353, 367, 371
- \cs_new_protected:Npn 9, 33, 256, 336
- \cs_set:Nn 338
- \cs_set_protected:Nn 373

D

\DeclareDocumentCommand *3*, 203, 258, 387
\def 251

E

\endgroup 252
\ensuremath 448, 449
erw commands:
- \erw_keyval_error:Nn 63
- \erw_lambda:nnn 387
- \erw_prop_keyval_parse:NNNn 61

\erw_seq_use:Nn 226
\erw_sys_jobnametimestamp: . 406, 407
exp commands:
\exp_args:NNf 225
\exp_args:NNx 202, 258
\exp_args:No 81, 236, 351
\exp_args:Nx 35
\exp_last_unbraced:Nf 410, 424, 438, 452
\exp_not:N 291, 301, 309, 317
\exp_not:n 279
\expandafter 251
\ExplSyntaxOff 483
\ExplSyntaxOn 3

F

file commands:
\file_input:n 181
\function *2*

G

\gappto 271

I

\IfBooleanT 265
\ifcsdef 79, 319
\IfValueT 263, 277
\IfValueTF 476
iow commands:
\iow_close:N 165, 169
\iow_new:N 164
\iow_now:Nn 194
\iow_open:Nn 176

K

keys commands:
\l_keys_choice_tl 400
\keys_define:nn 394
\keys_set:nn 392

L

\languagename 81

M

msg commands:
\msg_error:nnn 197, 299
\msg_new:nnn 283, 284
\msg_warning:nnn 75, 291, 309, 317

N

\NB *18*, *19*
\NewDocumentCommand . . . .
. . . . *6*, 379, 384, 389, 473, 480

O

options:
And *3*, *32*
Expans *4*, *32*
File *4*, *32*
Inner *4*, *32*
Outer *4*, *33*
Param *4*, *33*
Separ *4*, *33*
Write *4*, *33*

P

prop commands:
\prop_clear_new:N 351
\prop_gclear_new:N 46
\prop_gput:Nnn 29, 64, 340
\prop_if_exist:NTF 360
\prop_if_in:NnTF 70
\prop_item:Nn 73, 340, 364, 375
\prop_map_function:NN 342
\prop_new:N 26, 58, 369
\ProvideDocumentCommand 385

Q

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

R

\Real *1*, *2*
\Reel *1*

S

seq commands:
\seq_gclear_new:N 19, 55
\seq_gput_right:Nn 11, 375
\seq_if_empty:NTF 47
\seq_map_function:NN . . . .
. . . . 20, 50, 217, 356, 377
\seq_set_from_clist:Nn 237, 355

T

\text 323, 324, 325
tl commands:
\c_empty_tl 48, 234, 254, 276, 384
\tl_const:Nn 86, 321, 329
\tl_gset:Nn . . . 175, 287, 295, 305, 404
\tl_gset_eq:NN 400
\tl_log:n 182, 195
\tl_new:N 32, 172, 189
\tl_trim_spaces:n 11, 37, 38

U

use commands:
\use:N 36, 482
\usepackage *2*, *6*

X

\X *2*

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

```
\tl_const:Nn \c__ccool_lang_and_tl
{
% ^^A https://www.overleaf.com/learn/latex/International_language_support
% ^^A  ancientgreek,     % not latin
% ^^A  arabic,           % not latin
% ^^A  armenian,         % not latin
% ^^A  assamese,         % translation unknown
% ^^A  bengali,          % not latin
% ^^A  bokmal,           % translation unknown
% ^^A  bulgarian,        % not latin
% ^^A  coptic,           % translation unknown
% ^^A  dumylang,         % translation unknown
% ^^A  ethiopic,         % translation unknown
% ^^A  farsi,            % not latin
% ^^A  friulan,          % translation unknown
% ^^A  greek,            % not latin
% ^^A  gujarati,         % not latin
% ^^A  hindi,            % not latin
% ^^A  ibycus,           % translation unknown
% ^^A  interlingua,      % translation unknown
% ^^A  kannada,          % not latin
% ^^A  lao,              % not latin
% ^^A  malayalam,        % not latin
% ^^A  marathi,          % not latin
% ^^A  mongolian,        % not latin
% ^^A  mongolianlmc,     % translation unknown
% ^^A  monogreek,        % not latin
% ^^A  nynorsk,          % translation unknown
% ^^A  oriya,            % translation unknown
% ^^A  panjabi,          % not latin
% ^^A  pinyin,           % translation unknown
% ^^A  romansh,          % translation unknown
% ^^A  russian,          % not latin
% ^^A  sanskrit,         % not latin
% ^^A  serbian,          % not latin
% ^^A  serbianc,         % not latin
% ^^A  tamil,            % not latin
% ^^A  telugu,           % not latin
% ^^A  uppersorbian,     % translation unknown
% ^^A slovenian=in,     % not supported by babel
% ^^A ukenglish=and,    % not supported by babel
% ^^A ukrainian=i,      % not latin
% ^^A usenglishmax=and, % not supported by babel
afrikaans=en,
basque=eta,
catalan=i,
croatian=i,
czech=a,
danish=og,
dutch=en,
english=and,
esperanto=kaj,
estonian=ja,
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 [8]

```
\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 : $\langle$ *token list* $\rangle$
#2 : $\langle$ seq_1 $\rangle$
#3 : $\langle$ seq_2 $\rangle$

#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~'en
```

8 option

`\__ccool_option_inner:n` #1 : ⟨*code*⟩

```
\cs_new_protected:Nn \__ccool_option_inner:n
{
  \tl_gset:Nn \g__ccool_option_inner_tl {#1}
}
\__ccool_option_inner:n
{
  \msg_warning:nnn{ erw }{ notset }{ \exp_not:N \g__ccool_option_inner_tl }
}
```

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

`\__ccool_option_param:n` #1 : ⟨*token list*⟩

```
\cs_new:Nn \__ccool_option_param:n
{
  \tl_gset:Nn \g__ccool_option_param_tl{#1}
}
\__ccool_option_param:n
{
  \msg_error:nnx{ __ccool }
  { generic }
  { \exp_not:N\g__ccool_option_param_tl~undefined }
}
```

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

`\__ccool_option_outer:n` `#1` : ⟨ *inline code* ⟩

```
\cs_new_protected:Nn \__ccool_option_outer:n
{
  \tl_gset:Nn \g__ccool_option_outer_tl {#1}
}
\__ccool_option_outer:n
{
  \msg_warning:nnn{ erw }{ notset }{ \exp_not:N \g__ccool_option_outer_tl }
}
```

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

`\__ccool_option_separ:n` `#1` : `{`⟨ tl_1 ⟩`}{`⟨ tl_2 ⟩`}{`⟨ tl_3 ⟩`}`

```
\cs_new_protected:Nn \__ccool_option_separ:n
{
  \cs_gset:Npn \g__ccool_option_separ_tl {#1}
}
\__ccool_option_separ:n
{
  \msg_warning:nnn{ erw }{ notset }{ \exp_not:N \g__ccool_option_separ_tl }
}
```

(*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 : ⟨tl_1⟩
#2 : ⟨tl_2⟩
#3 : ⟨tl_3⟩

```
\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 : ⟨ *tl var name* ⟩

#2 : ⟨ *key* ⟩

```
\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 : ⟨ *tl var name* ⟩

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

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

`\__ccool_prop_new:n` #1 : ⟨ *tl var name* ⟩

```
\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 : ⟨ *seq_1* ⟩
#2 : ⟨ *seq_2* ⟩ (keys)
#3 : ⟨ *prop* ⟩

```
\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 }
  }
```

[2] [todo]: allow only m- or o-type arguments
[3] [todo]: Fix placeholders passed to options requiring code
[4] [abandon]: Requirement: write to file if Write; Update: redundant with \cs {Ccool}+Write

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

Separ

```
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⟩
```