The ccool package[*]

Erwann Rogard[†]

Released 2020/04/16

Abstract

The package ccool for LaTeX provides a *key-value* interface,\Ccool, meant to facilitate the generation of commands. Optional parameters that control the processing of the input and its expansion are set to their most likely usage. This can be used to encode notational conventions (such as `\Real` → `\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 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

[*] This file describes version v2.4, last revised 2020/04/16.

[†] firstname dot lastname AusTria gmail dot com

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) ⟨X⟩ ← `Y`: set ⟨X⟩ to `Y`

d) `\X` → `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{`$\langle kvl_1\rangle$`}` creates for each $\langle key_i\rangle$`=`$\langle val_i\rangle$, the command `\`$\langle key_i\rangle$, according to the following algorithm:

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

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

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

See `\CcoolLambda` to allow command `\`$\langle key_i\rangle$ 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{`$\langle kvl_1\rangle$`}*` supplements the Core feature with the expansion of the $\langle val_i\rangle$'s using typesetting rules controlled by *option* `Separ` and `Outer`. The first are *separators* applied to the $\langle val_i\rangle$'s to form a *token list*, and the second a function applied to the latter.

They can be 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$_i$*'s

\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 parameter, that implicit in $\langle key_i\rangle$, is controlled by `Param`. In our example, $\langle tl_2\rangle$ could be `Style`.

2.8 Write

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

`\CcoolClear`

`\CcoolClear<`$\langle tl_2\rangle$`>{`$\langle clist\rangle$`}`

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`[4],

`\CcoolOption`

`\CcoolOption{`$\langle keyval\ list\rangle$`}`

Semantics Controls the default behavior of `\Ccool`.

`Expans`

Also see Part IV, `Expans`

Semantics See Core feature

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

`File`

Also see Part IV, `File`

Semantics See Write

Syntax $\langle path\rangle$

`Inner`

Also see Part IV, `Inner`

Semantics See Process the val_i's

Syntax ⟨*code*⟩, with `####1` as the argument to be replaced

`Param`

Also see Part IV, `Param`

Semantics See Parameterize the key_i's

Syntax ⟨*token list*⟩

`Outer`

Also see Part IV, `Outer`

Semantics See Expand the val_i's

Syntax ⟨*code*⟩, with `####1` as the argument to be replaced

`Separ`

Also see Part IV, `Separ`

Semantics See Expand the val_i's

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

`Write`

Also see Part IV, `Write`

Semantics See Write

Syntax ⟨*boolean*⟩

`\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: `$⟨key_i⟩<x$`.

Do: `$\⟨key_i⟩{<}x$`

2)

Don't: `[a, b)`

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

3)

Don't: `\cal F`.

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

4)

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

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

5)

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

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

6) Also see Part III, section 4

Part II
Listing

NB:

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

Listing 1. `\CcoolVers`

```
\CcoolVers
```

2020/04/16 v2.4 cool — A key-value interface for generating commands

Listing 2. "Let 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} }*s{{~\rm{and}~}}
[~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. Separators

```
%^^A--->
\CcoolOption{
  Separ={{\ \char`@\ }{\ \%\ }{\ \char`@\ }}}
\Ccool{ X = x, Y = y }*[\\]
{ X = x, Y = y, Z = z }*[\\]
{ X = x, Y = y }*s{{\ \&\ }}[\\]
{ X = x, Y = y }*s{{\ \&\ }{,\ }}[\\]
{ X = x, Y = y, Z = z }*s{{\ \&\ }}[\\]
{ X = x, Y = y, Z = z }*s{{\ \&\ }{,\ }}[\\]
{ X = x, Y = y, Z = z }*s{{\ \&\ }{,\ }{\ \&\ }}\\
%^^A<---
\CcoolClear
```

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

Listing 8. 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{ Write = \BooleanFalse }
\CcoolClear
```

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

Listing 9. Listing 8 read from file

```
%^^A--->
\CcoolRead
\KeyF<Test>\KeyA<Test>\nobreak
\KeyG<Test>\KeyA<Test>\nobreak
\KeyH<Test>\KeyA<Test>\nobreak
\KeyH<Test>\KeyA<Test>\nobreak
{\{}\nobreak\KeyO<Test>{\}},{\ }\nobreak
\KeyM<Test>\KeyR<Test>\nobreak
\KeyO<Test>\nobreak
\KeyT<Test>\nobreak
\KeyL<Test>\nobreak
\KeyD<Test>\nobreak
\KeyZ<Test>\nobreak
\KeyN<Test>\nobreak
%^^A<---
\CcoolClear
```

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

Listing 10. Probability space

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

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

Listing 11. Listing 10 read from file

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

Ω $\mathcal{F}$ $\mathcal{P}$

Listing 12. Mittelwertsatz für n Variable[3, 17.3]

```
\CcoolOption{ Write = \BooleanTrue }
%^^A--->
\newtheorem{theorem}{Theorem}
\AfterEndEnvironment{theorem}{\CcoolHook}
\Ccool c{\mathbb{#1}}
{ N = { N } , R = { R } }+[]
{ Grad = { \operatorname{grad} } }+
[\begin{theorem}
  [Mittelwertsatz f\"ur $n$ Variable]Es~sei~]
  { OffMenge = {D}, Ci = {C^{1}}, Strecke = { \left[x_0,x\right] } }+
  [$n\in\N$,~$\OffMenge\subseteq\N^n$ 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<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear
```

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 13. Listing 12 read from file

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

```
\CcoolClear
```

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

Listing 14. Polynôme

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

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

Listing 15. Listing 14 read from file

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

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

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

```
  \CcoolOption{ Write = \BooleanTrue }
%^^A--->
\Ccool c{\mathbb{#1}}{ Corps = {K}, Nat = {N}, Reel = {R} }
[Soient~]
{
  Poly = \CcoolLambda[om]{#2\IfValueT{#1}{_#1}[X] },
  PolyR = \CcoolLambda[o]{\Poly[#1]{\Reel}}
}
[$\Poly[n]{\Corps}$ 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<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear
```

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

Listing 17. Listing 16 read from file

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

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

Listing 18. Fonction et fonctionelle

```
\CcoolOption{ Write = \BooleanTrue }
%^^A--->
\Ccool{ EvalAt = \CcoolLambda{(#1)}, ApplyOp = \CcoolLambda[mm]{#1[#2]} }
[Supposons une fonction $f\EvalAt{t}$, et \'etudions le probl\`eme o\`u
  la fonctionnelle $\ApplyOp{S}{f}$ est donn\'ee par\dots]{}
%^^A<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear
```

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

Listing 19. Listing 18 read from file

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

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

Listing 20. CUSUM statistic[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<---
\CcoolOption{ Write = \BooleanFalse }
\CcoolClear
```

The CUSUM statistic process and the corresponding one-sided CUSUM stopping time are defined as follows:

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

1. $u_t(\lambda) = \lambda\xi_t - \frac{1}{2}\lambda^2 t$; $m_t(\lambda) = \inf_{0\le s\le t} y_s(\lambda)$.
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 21. Listing 20 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][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, `$tex ccool.dtx`)

2) Put the generated `ccool.sty` in the search path of the LaTeXengine

4 Issue

1) **Don't:** `Inner=\{####1\}`

Symptom: `\CcoolRead` fails

Do: ``Inner={\char`{####1\char`}}``

5 Support

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

6 Testing

6.1 Technicality

Not possible to compile-check the expansion of a certain class of macros against predefined values[8]. Instead, one can visually check Part II, as generated in section 3 on one's own machine, against that of the repository for the same version.

6.2 Platform

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

6.3 Engine

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

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

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

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

6.4 Results

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

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

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

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

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

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

6.5 Other

Check [5] for testing ccool with llncs

7 To do

1) *Placeholder* passed to Part IV `\CcoolOption` should be `#1` not `####1`

2) `\CcoolOption` should behave in away similar to that described in Part I subsection 6.7

References

[1] Nick Setzer *The cool package*, 2005, `https://www.ctan.org/pkg/cool`

[2] The 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] @sean-allred's answer to "How to create lambda expressions?", `https://tex.stackexchange.com/a/188053/112708`

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

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

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

Part IV
Implementation

1 Opening

```
⟨@@=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_separ:nn

#1 : ⟨ *int* ⟩
#2 : ⟨ *tokens* ⟩

```
\cs_new:Nn \__ccool_aux_separ:nn
{
  \int_case:nnTF {#1}
  {
    {1}
    { \prg_replicate:nn{ 3 }{#2} }
    {2}
    {
      { \use_i:nn #2 }
      { \use_ii:nn #2 }
      { \use_i:nn #2 }
    }
    {3}{#2}
  }
  { \c_empty_tl }
  {
    \msg_error:nnnn { __ccool }
    { separ }
    { \exp_not:N \__ccool_aux_separ:nn }
    {#2}
  }
}
\cs_generate_variant:Nn \__ccool_aux_separ:nn { e }
```

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

`\__ccool_aux_separ:n`

`#1` : ⟨ *tokens* ⟩

```
\cs_new:Nn \__ccool_aux_separ:n
{
  \__ccool_aux_separ:en{ \tl_count:n{#1} }{#1}
}
```

(*End definition for* `\__ccool_aux_separ: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 lambda

`\__ccool_lambda:nn`

[7]

```
\cs_new_protected:Npn \__ccool_lambda:nn #1 #2
{
  \exp_args:NNx
  \DeclareDocumentCommand \__ccool_lambda_expression
  {#1}
  {#2}
  \__ccool_lambda_expression
}
```

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

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
    \__ccool_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}{ generic }{#1}
\msg_new:nnn {__ccool}{ iow }{#1~is~closed~can't~write}
\msg_new:nnn {__ccool}{ keyonly }{#1~does~not~take~values;~keyval~is~#2}
\msg_new:nnn {__ccool}{ keywrong }{#1~does~not~recognize~key~#2}
\msg_new:nnn {__ccool}{ separ }{#1~expects~1~to~3~items,~#2}
\msg_new:nnn {__ccool}{ unset }{#1~unset}
```

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{ __ccool }{ unset }{ \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{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_outer_tl }
}
```

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

__ccool_option_separ:n

#1 : {$\langle$ tl_1 $\rangle$}{$\langle$ tl_2 $\rangle$}{$\langle$ tl_3 $\rangle$}

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

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

9 prop

__ccool_prop_append:NN

#1 : $\langle$ prop_1 $\rangle$
#2 : $\langle$ prop_2 $\rangle$

```
\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 : $\langle$ *prop* $\rangle$
#2 : $\langle$ *tl var name* $\rangle$

```
\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 : $\langle$ *tl var name* $\rangle$

```
\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`.)

__ccool_erw_seq_use:Nn

```
%      \begin{arguments}
%      \item \meta{ seq }
%      \item \meta{ tokens }
%      \end{arguments}
\cs_new:Nn \__ccool_seq_use:Nn
{
  \exp_last_unbraced:NNf
  \seq_use:Nnnn #1
  \__ccool_aux_separ:n{#2}
}
```

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

11 sys

__ccool_sys_date:

```
\cs_new:Nn \__ccool_sys_date:
{
  \int_eval:n
  {
    \c_sys_year_int * 10000
    +\c_sys_month_int * 100
    +\c_sys_day_int *  1
  }
}
```

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

__ccool_sys_date_hex:

```
\cs_new:Nn \__ccool_sys_date_hex:
{\int_to_hex:n{\__ccool_sys_date:}}
```

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

__ccool_sys_time:

```
\cs_new:Nn \__ccool_sys_time:
{
  \int_eval:n
  {
    \c_sys_hour_int * 100
    +\c_sys_minute_int * 1
  }
}
```

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

__ccool_sys_time_hex:

```
\cs_new:Nn\__ccool_sys_time_hex:
{\int_to_hex:n{\__ccool_sys_time:}}
```

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

__ccool_sys_filename:

```
\cs_new:Nn\__ccool_sys_filename:
{
  \c_sys_jobname_str--
  \__ccool_sys_date_hex:--
  \__ccool_sys_time_hex:
}
```

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

12 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

```
\ProvideDocumentCommand \CcoolLambda { O{m} m }
{
  \__ccool_lambda:nn { #1 } { #2 }
}
```

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

\CcoolOption

```
\NewDocumentCommand{ \CcoolOption }
{ m }
{
  \keys_set:nn{ __ccool }{#1}
%^^A  \bool_if:nTF{ \g__ccool_log_open_bool }
%^^A  {
%^^A    \__ccool_log_write:n
%^^A    {
%^^A      \begingroup
%^^A      \def \__ccool_log_entry { \CcoolOption{ #1 } \expandafter
%^^A        \endgroup \__ccool_log_entry
%^^A      }
%^^A    }{\c_empty_tl}
%^^A  }
}
```

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

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

`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 = { \__ccool_sys_filename: },
File .initial:n = { \__ccool_sys_filename: },
```

`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 }
  }
},
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 = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
Separ .initial:n = { {\ }and{\ } } { ,{\ } } { ,{\ }and{\ } },
```

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 7.*)

\CcoolVers

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

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

13 Closing

```
\ExplSyntaxOff
```