The ccool package[*]

Erwann Rogard[†]

Released 2020/04/17

Abstract

ccool stands for Custom COntent Oriented for LaTeX, that is " give commands the ability to contain the mathematical meaning while retaining the typesetting versatility", a concept pioneered by cool[1]. Here, the commands are not predefined, instead they are created ('custom') using a minimalist interface built upon xparse[4]. Specifically, `\Ccool` takes as mandatory argument a *keyval list*, the lhs of each element of which encodes a concept, and the rhs a typesetting instruction (for instance, the side effect of `{ Real = \mathbb{R} }` is that `\Real` expands to $\mathbb{R}$). A *keyval list* can optionally be expanded in place (append `*`), according to default or inline rules, and interspersed with optional arguments that are destined exclusively for expansion (`[Let~]` and `[~denote real numbers.]`). Thus, in theory, a document could be typeset, starting with a single `\Ccool` declaration. An optional parameter prepended to the *keyval list*, `<`$\langle param\rangle$`>`, allows to parameterize the keys (for instance, one parameter value for the style, another for a property). In conjunction with lamba expressions, this tool allows for encoding the way complex mathematical objects are formatted (for instance, functions and operators having (.) and [.], respectively, around their arguments). Optionally, the macros can be written to a file, and read, which can be useful for typesetting documents sharing the same notation.

Contents

[*]This file describes version v2.1, last revised 2020/04/17.
[†]firstname dot lastname AusTria gmail dot com

Part I
Usage

Convention

1. Loosely, those of [2] and [4], for example as to the meaning of ⟨*token list*⟩.
2. If unspecified, the environment in which a macro must be declared is `document`.

`\usepackage`

```
\usepackage{ccool}
```

Requirement

1. `ccool.sty` is in the path of the LaTeX engine. See Part III, section 4.
2. Declare it in the *preamble*

`\Ccool`

```
\Ccool
[⟨tl1⟩]
<⟨tl2⟩>
i{⟨code1⟩}
{⟨kvl1⟩}
+
*
s{{⟨tl3⟩}|{⟨tl3⟩}{⟨tl4⟩}|{⟨tl3⟩}{⟨tl4⟩}{⟨tl5⟩}}
o{⟨code2⟩}
[⟨tl6⟩]
```

Requirement ⟨*kvl*$_1$⟩ is specified (all others optional).

⟨`tl`$_1$⟩

Example `Let~`

Semantics Expands $\langle tl_1 \rangle$

$\langle$`tl`$_2\rangle$

Default `Param`'s

Example `Default`, `Style` and `Describe`, `ModelA` and `ModelB`

$\langle$`code`$_1\rangle$

Default `Inner`'s

Example `\mathbb{#1}`

$\langle$`kvl`$_1\rangle$

Example `Real={\mathbb{R}}`

Semantics

1) $\langle val_i \rangle \leftarrow \langle code_1 \rangle$ applied to $\langle val_i \rangle$

2) `\`$\langle$`key`$_i\rangle$`<`$\langle tl_2\rangle$`>` $\leftarrow \langle val_i \rangle$ defined in step *1)*, using `Expans` for expansion.

3) If `Write`, writes the input used by step *2)* to `File`

`+`

Semantics Repeats step *1)*, step *2)*, and step *3)*, at `\CcoolHook` (useful inside a *local group*)

`*`

Semantics Expands $\langle code_2 \rangle$ applied to the list created in step *1)*, using the separator specified by $\langle tl_3 \rangle$, $\langle tl_4 \rangle$, and $\langle tl_5 \rangle$.

$\langle$`tl`$_3\rangle$

Default `Separ`'s

Example `{~\in~}`

$\langle$`tl`$_4\rangle$

Default `Separ`'s

Example `{,~}`

$\langle$`tl`$_5\rangle$

Default `Separ`'s

Example `{~\&~}`

$\langle$`code`$_2\rangle$

Default `Outer`'s

Example `$\left\{#1\right\}$`

$\langle$`tl`$_6\rangle$

Semantics `\Ccool[`$\langle tl_6\rangle$`]`

Other

Continued in Part IV, section 12.

Do's and dont's

1)

Don't: `\Ccool{ A = a, B = b }[Hello, world!]`

Do: `\Ccool{ A = a, B = b }[Hello, world!]{}`
or `\Ccool{ A = a, B = b } Hello, world!`

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) Also see Part III, section 3

Part II
Listing

Listing 1 are settings to replicate the listings. For exhaustivity, check the `documentation` section of `ccool.dtx`.

Listing 2 is self explanatory.

Listing 3-9 is a tutorial comprising different ways to typeset "Let $\mathbb{N}$ and $\mathbb{R}$..." The plain version is prone to errors, should the author change `\mathbb{R}` to, say, `\mathcal{R}` throughout the document. Listing 4 avoids that by separating style from meaning. Listing 5-7 achieve greater compactness (DRY and expand in place). Listing 8 extends Listing 5 with a description for each key. Listing 9 is to Listing 8, what Listing 7 is Listing 6.

Listing 10 shows the full range of uses of separators' parameter.

Listing 11 and Listing 12 are a contrived "Hello, world!" test case.

Listing 13, Listing 15, and Listing 19 typeset realistic mathematical text, and write it to a file. Listing 14, Listing 16, and Listing 20, read the corresponding files.

Listing 1. Preamble

```
%     \usepackage{amsmath, amsthm, commath}
%     \usepackage[T1]{fontenc}% \char`[
%
```

Listing 2. `\CcoolVers`

```
%     \CcoolVers
%
```

2020/04/17 v2.1 cool — A tool for encoding notational conventions (esp. Math)

Listing 3. Let $\mathbb{N}$ and $\mathbb{R}$...Plain

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

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

Listing 4. Let $\mathbb{N}$ and $\mathbb{R}$...Equivalent to 3 with `\NewDocumentCommand`

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

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

Listing 5. Let $\mathbb{N}$ and $\mathbb{R}$...Equivalent to 4 with `\Ccool`

```
%     \Ccool { Nat = {\mathbb{N}}, Real = {\mathbb{R}} }
%     Let~$\Nat$ and $\Real$~denote the natural and real numbers.
%
```

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

Listing 6. Let $\mathbb{N}$ and $\mathbb{R}$...Equivalent to 5 but with i{$\langle code_1\rangle$}

```
%     \Ccool i{\mathbb{#1}}{ Nat = {N}, Real = {R} }
%     Let~$\Nat$ and $\Real$~denote the natural and real numbers.
%
```

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

Listing 7. Let $\mathbb{N}$ and $\mathbb{R}$...Equivalent to 6 with s{$\langle tl_3\rangle$}

```
%     \Ccool[Let~]
%     i{\mathbb{#1}}{ Nat = {N}, Real = {R} }*s{{~\rm{and}~}}
%     [~denote the natural and real numbers.]{}
%
```

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

Listing 8. Let $\mathbb{N}$ and $\mathbb{R}$...Equivalent to 7 with <$\langle tl_2\rangle$>

```
%     \Ccool{ Nat = {\mathbb{N}}, Real = {\mathbb{R}} }[]
%     <Describe>{ Nat = {natural numbers}, Real = {the real line} }
%     [Let $\Nat$ and $\Real$ denote
    the~\Nat<Describe>~and~\Real<Describe>.]{}
%
```

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

Listing 9. Let $\mathbb{N}$ and $\mathbb{R}$...Equivalent to 8 with s{$\langle tl_3\rangle$}

```
%     \Ccool[Let~]
%     i{\mathbb{#1}}{ Nat = {N}, Real = {R} }*s{{~\rm{and}~}}
%     [~denote~the~]
%     <Describe>
%     {
%     Nat = {natural numbers},
%     Real = {the real line}
%   }*s{{~and~}}o{#1.}
%
```

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

Listing 10. Separators

```
%      \CcoolOption{
%      ^^A% spaces betw. inner and outer brackets matter!->
%      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{{\ \&\ }{,\ }{\ \&\ }}\\
%
```

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

Listing 11. Hello, world! (test case)

```
%      \CcoolOption{Separ = {{}{.}{.}}, Outer = {####1}}
%      \CcoolOption{ Write = \BooleanTrue }
%      \Ccool
%      <Test>{ KeyA = {.}, KeyB = {!}, KeyC = {\%} }[]
%      <Test>{ KeyD = {d}, KeyE = {\%} }[]
%      <Test>i{\{#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{{}{}{}}o{{\char`[}#1}[]
%      <Test>{ KeyU = {\%}, KeyV = {\%}, KeyW = {\%} }[]
%      <Test>{ KeyX = {\%}, KeyY = {\%}, KeyZ = {\KeyB<Test>} }\nobreak
%      \KeyL<Test>\KeyD<Test>\KeyZ<Test>\KeyN<Test>\\
%      \CcoolOption{ Write = \BooleanFalse }
%
```

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

Listing 12. Listing 11 read from file

```
%      \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
%
```

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

Listing 13. Probability space

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

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

Listing 14. Listing 13 read from file

```
%     \CcoolRead \tab $\Omega$ $\Field$ $\Meas$
%
```

Ω $\mathcal{F}$ $\mathcal{P}$

Listing 15. Mittelwertsatz für n Variable[3, 17.3]

```
%     \CcoolOption{ Write = \BooleanTrue }
%     \newtheorem{theorem}{Theorem}
%     \AfterEndEnvironment{theorem}{\CcoolHook}
%     \Ccool i{\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: $\N$, $\Punkt$)
%     \CcoolOption{ Write = \BooleanFalse }
%
```

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 16. Listing 15 read from file

```
%     \CcoolRead \tab $\N$ $\R$ $\OffMenge$ $\Ci$ $\Strecke$
%
```

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

Listing 17. Fonction et fonctionelle

```
%     \CcoolOption{ Write = \BooleanTrue }
%     \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]{}
%     \CcoolOption{ Write = \BooleanFalse }
%
```

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

Listing 18. Listing 17 read from file

```
%     \CcoolRead \tab $f\EvalAt{t}$, $\ApplyOp{S}{f}$
%
```

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

Listing 19. CUSUM statistic[5]

```
%     \newtheorem{definition}{Definition}
%     \AfterEndEnvironment{definition}{\CcoolHook}
%
%     \CcoolOption{ Write = \BooleanTrue }
%     \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 } }+*o{$#1\in\Real^{+}$.}
%       [~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: $\Scale$, $\CUSUM$)
%     \CcoolOption{ Write = \BooleanFalse }
%
```

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 20. Listing 19 read from file

```
%     \CcoolRead \tab $\Time $ $\Process$ $\Scale$ $\Real$ $\CUSUMthresh$
    $\LogWald$  $\CUSUMst$ $\CUSUM$ $\LogWaldInf$
%
```

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

Part III
Other

1 Acknowledgment

This work has benefited from Q&A's from the LaTeXcommunity[6]. Specific attributions are made throughout this document.

2 Install

1) Compile `ccool.dtx` (under Unix, `$tex ccool.dtx`)

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

3 Issue

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

Symptom: `\CcoolRead` fails

Do: `Inner={\char‘{####1\char‘}}`

4 Support

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

5 Testing

5.1 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 2 on one's own machine, against that of the repository for the same version.

5.2 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
```

5.3 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)`

5.4 Results

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

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

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

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

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

5.5 Other

Check [5] for testing ccool with llncs

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

Change History

v1.0

- General: Initial version 14

v1.1

- General: Added: `Save` 14
  - Added: Listing 1., 2., 3., 4., 6., and 9. 14
  - Added:`\OopsRestore` 14
  - Added:`\OopsTest` 14
  - Deleted: Listing 1-5 from v1.0 . . . 14
  - Fixed: apparent anomaly in v1.0's Listing 4, see Listing 11 14
  - Replaced: `\OopsOptions` by `\OopsOption` . . 14
  - Replaced: `{`⟨*kvl*$_2$⟩`}` by <kvl_2> given that option type `G` not recommended[4] 14
  - Replaced: `GenericObject` by `Name` 14
  - Replaced: `Separators` by `Separ` . . 14
  - Revamped: much of the implementation 14

v1.2

- General: Added: optional `*`to `\OopsNew` as instruction to expand kvl$_1$ 14
  - Deleted: `\OopsTest` 14
  - Deleted: ⟨*kvl*$_2$⟩ and ⟨*code*$_2$⟩ 14
  - Deleted: Listing 2-3 from v1.1. . . . 14
  - Replaced: `\OopsClear{`⟨*tl*$_2$⟩`}` by `\OopsClear[`⟨*keyval list*⟩`]` 14
  - Replaced: `\Restore` by `\Read` 14
  - Replaced: `\Save` by `\Write` 14

v1.3

- General: Replaced: `\OopsNew` by `\Oops` 14
  - Replaced: `{`⟨*tl*$_2$⟩`}` and `[`⟨*tl*$_2$⟩`]` by `<`⟨*tl*$_2$⟩`>` 14

v1.4

- General: Added: section 4 14
  - Added: `\OopsDebug` 14
  - Added: `\OopsHook` 14
  - Added: `Expans` (for debugging' sake, but...) 14
  - Added: Listing 1., 2., and 3. 14
  - Added:optional `+`to `\OopsNew` to make side effects presist beyond local group 14
  - Deleted: Listing 1., and 2. 14
  - Replaced: `s{{`⟨*tl*$_3$⟩`}{`⟨*tl*$_4$⟩`}{`⟨*tl*$_5$⟩`}}` by `s{{`⟨*tl*$_3$⟩`}|{`⟨*tl*$_3$⟩`}{`⟨*tl*$_4$⟩`}|{`⟨*tl*$_3$⟩`}{`⟨*tl*$_4$⟩`}{`⟨*tl*$_5$⟩`}}` 14

v1.5

- General: Added: `File` 14
  - Deleted: dependence on `datetime` . 14

v1.6

- General: Added: Listing 1 (preamble) 14
  - Renamed: `\OopsClear` to `\CcoolClear` 14
  - Renamed: `\OopsDebug` to `\CcoolDebug` 14
  - Renamed: `\OopsHook` to `\CcoolHook` 14
  - Renamed: `\OopsOption` to `\CcoolOption` 14
  - Renamed: `\OopsRead` to `\CcoolRead` 14
  - Renamed: `\Oops` to `\Ccool` 14
  - Renamed: `oops` to `ccool` (better describes the purpose) 14

v1.7

- General: Added: Legends to listings . 14
  - Added: Listing 19 (CUSUM) 14
  - Deleted: `\CcoolDebug` 14
  - Deleted: Listing 5 from v1.6 14

v1.8

- General: Added: `\CcoolVers` 14
  - Added: `\CcoolLambda` 14
  - Added: Listing 17, Listing 18 14
  - Added: Listing 2 14

v1.9

- General: Added: support for LuaTEX 14
  - Moved: from Part I to Part IV, what is now that part's section 12 14

v2.0

- General: Added: support for XƎTEX . 14
  - Deleted: `File`'s dependency on `texosquery` and `\pdfcreationdate` 14
  - Updated: `\RequirePackage`, `\NeedsTeXFormat`'s second argument / TeX Live 2020 14

v2.1

- General: Added: Listing 3, Listing 4, Listing 5, Listing 6, Listing 6, Listing 8, and Listing 9 (tutorial) . 14
  - Replaced: `\CcoolLambda`'s optional integer argument (number of `m`'s) by a standard argument list 14
  - Replaced: ⟨*tl*$_2$⟩'s position within `\Ccool`'s argument list, from first to second. Greater versatility 14
  - Replaced: `Name` by `Param` 14

Replaced: as the default of `Param`, `Math` by `Default` . 14

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.

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
  \ProvideDocumentCommand{#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>i{#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{ i }{{#2}} m t+ s E{ s o }{{#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 : {⟨ 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{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_separ_tl }
}
```

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

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`.)

`\__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`

`#1` : ⟨*token list*⟩

Semantics Clears any data created by `\Ccool{`⟨*token list*⟩`}`

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

`\CcoolHook`

Example `\AfterEndEnvironment{theorem}{\CcoolHook}`

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

`\CcoolLambda`

`#1` : ⟨*arg spec*⟩
`#2` : ⟨*code*⟩

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

Semantics Creates a lambda expression with ⟨*integer*⟩ arguments for ⟨*code*⟩

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

`\CcoolOption` #1 : ⟨*keyval list*⟩

```
\NewDocumentCommand{ \CcoolOption }
{ m }
{
  \keys_set:nn{ __ccool }{#1}
}
```

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

`Expans`

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

```
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`

Value ⟨*path*⟩

```
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`

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

```
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`

Value ⟨*token list*⟩

```
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`

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

```
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`

Value That of 'separators' in [2, Section 8 of l3seq]

```
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`

Value ⟨*boolean*⟩

```
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` #1 : ⟨*path*⟩

Semantics

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

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

`\CcoolVers`

Semantics Expands to the package's version

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

13 Closing

```
\ExplSyntaxOff
```