

The ccool package*

Erwann Rogard[†]

Released 2020/04/15

Abstract

ccool stands for Custom COntent Oriented for L^AT_EX, a concept pioneered by cool[1]¹. This is done using a minimalist interface built upon xparse[4]. Specifically, `\Ccool<name>` begins a series of instructions alternating between ‘text’ and macro definitions, that themselves optionally expand using predefined or inline rules. For example,

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

expands to: “Let \mathbb{N} and \mathbb{R} denote the natural and real numbers.” As a side effect, `\Nat<Math>` encodes “N” (and likewise for `\Real`). `Math` being the default for `<name>`, `<Math>` can be dropped. In conjunction with lambda expressions, this tool allows for encoding the way certain mathematical objects, such as functions, should be formatted. Optionally, the macros can be written to a file, and read, which can be useful for typesetting documents sharing the same notation.

Contents

| | | |
|----------|--|----------|
| I | Usage | 4 |
| 0 | Convention | 4 |
| 1 | Loading the package | 4 |
| 2 | \Ccool | 4 |
| 2.1 | <code><tl₁></code> | 4 |
| 2.2 | <code>[tl₂]</code> | 4 |
| 2.3 | <code>i{<code₁>}</code> | 4 |
| 2.4 | <code>{<kv₁>}</code> | 5 |
| 2.5 | <code>+</code> | 5 |
| 2.6 | <code>*</code> | 5 |
| 2.7 | <code>s{{<tl₃>}{<tl₃>}{<tl₄>}{<tl₃>}{<tl₄>}{<tl₅>}}</code> | 5 |
| 2.8 | <code>o{<code₂>}</code> | 5 |
| 2.9 | <code>[tl₆]</code> | 5 |

*This file describes version v2.0, last revised 2020/04/15.

[†]firstname dot lastname AusTria gmail dot com

¹Whereas cool provided predefined macros, ccool is tool for making macros, hence “custom”.

| | | |
|------------|---|-----------|
| 3 | Other | 5 |
| 4 | Do's and dont's | 6 |
| II | Listing | 7 |
| | Listing 1. Version. | 7 |
| | Listing 2. Preamble. | 7 |
| | Listing 3. Separators. | 7 |
| | Listing 4. Hello, world! | 7 |
| | Listing 5. Listing 4 read from file. | 8 |
| | Listing 6. Probability space. | 8 |
| | Listing 7. Listing 6 read from file. | 8 |
| | Listing 8. Mittelwertsatz für n Variable. | 9 |
| | Listing 9. Listing 8 read from file. | 9 |
| | Listing 10. Lambda expression. | 9 |
| | Listing 11. Listing 10 read from file. | 10 |
| | Listing 12. CUSUM statistic. | 10 |
| | Listing 13. Listing 12 read from file. | 11 |
| III | Other | 12 |
| 1 | Acknowledgment | 12 |
| 2 | Install | 12 |
| 3 | Issue | 12 |
| 4 | Support | 12 |
| 5 | Testing | 12 |
| | 5.1 Technicality | 12 |
| | 5.2 Platform | 12 |
| | 5.3 Engine | 12 |
| | 5.4 Results | 13 |
| | 5.5 Other | 13 |
| | References | 13 |

| | |
|--------------------------|-----------|
| Change History | 13 |
| Index | 14 |
| IV Implementation | 18 |
| 1 aux | 18 |
| 2 lambda | 20 |
| 3 log | 21 |
| 4 make_key | 22 |
| 5 make_ccool | 22 |
| 6 msg | 24 |
| 7 option | 24 |
| 8 prop | 25 |
| 9 seq | 26 |
| 10 sys | 27 |
| 11 Front-end | 28 |
| 12 \CcoolClear | 28 |
| 13 \CcoolHook | 28 |
| 13 \CcoolLambda | 28 |
| 13 \CcoolOption | 28 |
| 13.2 Expans | 29 |
| 13.2 File | 29 |
| 13.2 Inner | 29 |
| 13.2 Name | 29 |
| 13.2 Outer | 30 |
| 13.2 Separ | 30 |
| 13.3 Write | 30 |
| 14 \CcoolRead | 31 |
| 15 \CcoolVers | 31 |

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 L^AT_EX engine. See [Part III, section 4](#).
2. Declare it in the *preamble*

$$\begin{array}{l} \underline{\backslash\text{Ccool}} \quad \backslash\text{Ccool}\langle t_{l_1}\rangle> \\ \quad [\langle t_{l_2}\rangle] \\ \quad \text{i}\{\langle\text{code}_1\rangle\} \\ \quad \{\langle\text{kv}_{l_1}\rangle\} \\ \quad + \\ \quad * \\ \quad \text{s}\{\{\langle t_{l_3}\rangle\}|\{\langle t_{l_3}\rangle\}\{\langle t_{l_4}\rangle\}|\{\langle t_{l_3}\rangle\}\{\langle t_{l_4}\rangle\}\{\langle t_{l_5}\rangle\}\} \\ \quad \text{o}\{\langle\text{code}_2\rangle\} \\ \quad [\langle t_{l_6}\rangle] \end{array}$$

Requirement $\langle kvl_1 \rangle$ is specified (all others optional).

 $\langle tl_1 \rangle$

Default **Name**

Example Math, ModelA, ModelB

Semantics Identifies a group of macros

 $\langle t_{l_2} \rangle$

Example Let~

Semantics Expands $\langle tl_2 \rangle$

 $\langle code_1 \rangle$

Default Inner

Example `\mathbb{#1}`

Semantics

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

$\langle kv1_1 \rangle$

Example `Elms={\omega_1, \dots, \omega_n}, Sample=\Omega`

Semantics

2) $\backslash \langle key_i \rangle \langle tl_1 \rangle \leftarrow \langle val_i \rangle$ defined in step 1), using **Expans** for expansion.

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

+

Semantics Appends step 2) and step 3) to **\CcoolHook**²

*

Semantics

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

$\langle tl_3 \rangle$

Default **Separ**

Example `{~\in~}`

$\langle tl_4 \rangle$

Default **Separ**

Example `{,~}`

$\langle tl_5 \rangle$

Default **Separ**

Example `{~\&~}`

$\langle code_2 \rangle$

Default **Outer**

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

$\langle tl_6 \rangle$

Semantics **\Ccool** $\langle tl_1 \rangle > [\langle tl_6 \rangle]$

²Needed inside a *local group*, for the side effect of **\Ccool** to persist thereafter.

Other

Continued in [Part IV, section 11](#).

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: `$\langle key_i \rangle <x$`.

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

3.

Don't: `[a, b)`

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

4.

Don't: `\Ccool{ F = \cal F }`.

Do: `\Ccool{ F = \cal{F} }` or `\Ccool{ F = \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.

```
% \CcoolVers
%
```

2020/04/15 v2.0 cool — A tool for encoding mathematical notation

Listing 2. Preamble^a

^aThese are the settings to replicate the listings. For exhaustivity, check the **documentation** section of `ccool.dtx`.

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

Listing 3. Separators

```
% \CcoolOption{
% ^^A% spaces betw. inner and outer brackets matter!->
% Separ={\ \char`@\ }{\ \% \ }{\ \char`@\ }}
% \Ccool<Test>{ 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 4. Hello, world!^a

^aIf this looks arcane, it's for the purpose of testing.

```
% \CcoolOption{ Separ = {\}\{.\}\}, Outer = { ####1 } }
% \CcoolOption{ Write = \BooleanTrue }
```

```

% \Ccool<Test>
% { KeyA = {.}, KeyB = {!}, KeyC = {\%} }[]
% { KeyD = {d}, KeyE = {\%} }[]i{\#1\}
% { KeyF = {H}, KeyG = {e}, KeyH = {l} }*[]
% { KeyI = {\%}, KeyJ = {\%}, KeyK = {\%} }[.\{l\}.\{o\}]
% { KeyL = {l}, KeyM = {\char`[]}, KeyN = {\char`[] } }[]
% { KeyO = {o}, KeyP = {\%}, KeyQ = {\%} }[{, \ }]
% { KeyR = {w}, KeyS = {o}, KeyT = {r} }*s{{}{}}o{{\char`[]\#1}[]
% { KeyU = {\%}, KeyV = {\%}, KeyW = {\%} }[]
% { KeyX = {\%}, KeyY = {\%}, KeyZ = {\KeyB<Test>} }\nobreak
% \KeyL<Test>\KeyD<Test>\KeyZ<Test>\KeyN<Test>\
% \CcoolOption{ Write = \BooleanFalse }
%

```

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

Listing 5. Listing 4 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 6. 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 7. Listing 6 read from file.

```
% \CcoolRead \tab $\Omega$ $\mathcal{F}$ $\mathcal{P}$
%
```

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

Listing 8. Mittelwertsatz für n Variable[1, 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] } }+
% [$n\in\mathbb{N}, \sim\mathcal{O}\text{ffMenge}\subseteq\mathbb{R}^n$ eine offene Menge und
% $f\in\mathcal{C}^1(\text{OffMenge},\mathbb{R})$.
% Dann gibt es auf jeder Strecke $\text{Strecke}\subseteq\text{OffMenge}$ einen
% Punkt $\xi\in\text{Strecke}$,~]
% { Steig = { \frac{ f(x)-f(x_0) }{ x-x_0 } }, Punkt = { \xi } }+
% [so dass gilt
% \begin{equation*}
% \text{Steig} = \text{Grad } f(\text{Punkt})^{\top}
% \end{equation*}
% \end{theorem}]
% {}
% (Check: $N$, $\xi$)
% \CcoolOption{ Write = \BooleanFalse }
%
```

Theorem 1 (Mittelwertsatz für n Variable) Es sei $n \in \mathbb{N}$, $D \subseteq \mathbb{R}^n$ eine offene Menge und $f \in C^1(D, \mathbb{R})$. Dann gibt es auf jeder Strecke $[x_0, x] \subset D$ einen Punkt $\xi \in [x_0, x]$, so dass gilt

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

(Check: \mathbb{N} , ξ)

Listing 9. Listing 8 read from file.

```
% \CcoolRead \tab $N$ $R$ $\mathcal{O}\text{ffMenge}$ $\mathcal{C}^1$ $\text{Strecke}$
%
```

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

Listing 10. Lambda expression.

```
% \CcoolOption{ Write = \BooleanTrue }
% \Ccool{ EvalAt = \CcoolLambda{(#1)}, ApplyOp =
% \CcoolLambda[2]{#1[#2]} }
% [Supposons une fonction  $f$ \EvalAt{t}$, et \etudions le probl\eme
% o\`u la fonctionnelle  $S$ \ApplyOp{S}{f}$ est donn\`ee par\dots]{ }
% \CcoolOption{ Write = \BooleanFalse }
%
```

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

Listing 11. Listing 10 read from file.

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

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

Listing 12. CUSUM statistic

```
% \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{ $\LogWald_{\Time}\EvalAt{ \Scale } = \Scale\Process_{\Time}
% - \frac{1}{2}\Scale^2\Time$ ;
%  $\LogWaldInf_{\Time}\EvalAt{ \Scale } = \inf_{0 \leq s \leq \Time}
% \CUSUM_{s}\EvalAt{ \Scale }$ .}
% \item{ $\CUSUM_{\Time}\EvalAt{ \Scale } =
% \LogWaldInf_{\Time}\EvalAt{ \Scale } - \LogWald_{\Time}\EvalAt{
% \Scale } \geq 0$ , which is the CUSUM statistic process.}
% \item{ $\CUSUMst \EvalAt{ \Scale, \LogWaldInf } = \inf\left[ \Time
% \geq 0 \text{ SuchThat } \CUSUM_{\Time}\EvalAt{\Scale} \geq \LogWaldInf
% \right]$ , 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. $u_t(\lambda) = \lambda \xi_t - \frac{1}{2} \lambda^2 t$; $m_t(\lambda) = \inf_{0 \leq s \leq t} y_s(\lambda)$.
2. $y_t(\lambda) = m_t(\lambda) - u_t(\lambda) \geq 0$, which is the CUSUM statistic process.
3. $T_c(\lambda, m) = \inf [t \geq 0; y_t(\lambda) \geq m]$, which is the CUSUM stopping time.

(Check: λ, y)

Listing 13. Listing 12 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 L^AT_EXcommunity[6]. Specific attributions are made throughout this document.

2 Install

Compiling `ccool.dtx`³ will generate `ccool.sty` and `ccool.pdf`

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)

³Under Unix, `$tex ccool.dtx`

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*)

5.5 Other

Check [5] for testing ccool with llncs

References

- [1] Nick Setzer *The cool package*, 2005, <https://www.ctan.org/pkg/cool>
- [2] The L^AT_EX3 Project Team *The L^AT_EX3 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 L^AT_EX3 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 | Added:\OpsRestore | 13 |
| General: Initial version | Added:\OpsTest | 13 |
| v1.1 | Deleted: Listing 1-5 from v1.0 . . . | 13 |
| General: Added: Save | Fixed: apparent anomaly in v1.0’s | |
| Added: Listing 1., 2., 3., 4., 6., and | Listing 4, see Listing 3 | 13 |
| 9. | Replaced: | |
| | | |

| | | | | | |
|------|---|----|------|---|----|
| | <code>\OpsOptions</code> by <code>\OpsOption</code> .. | 13 | v1.6 | General: Added: Listing 2 (preamble) | 13 |
| | Replaced: | | | Renamed: <code>\OpsClear</code> to | |
| | <code>{\kvl_2}</code> by <code><kvl_2></code> given that | | | <code>\CcoolClear</code> | 13 |
| | option type <code>G</code> not recommended[4] | 13 | | Renamed: <code>\OpsDebug</code> to | |
| | Replaced: <code>GenericObject</code> by <code>Name</code> | 13 | | <code>\CcoolDebug</code> | 13 |
| | Replaced: <code>Separators</code> by <code>Separ</code> .. | 13 | | Renamed: <code>\OpsHook</code> to | |
| | Revamped: much of the | | | <code>\CcoolHook</code> | 13 |
| | implementation | 13 | | Renamed: <code>\OpsOption</code> to | |
| v1.2 | General: Deleted: <code>\OpsTest</code> | 13 | | <code>\CcoolOption</code> | 13 |
| | Deleted: <code><kvl_2></code> and <code><code_2></code> | 13 | | Renamed: <code>\OpsRead</code> to | |
| | Deleted: Listing 2-3 from v1.1. | 13 | | <code>\CcoolRead</code> | 13 |
| | Replaced: <code>\OpsClear{\tl_1}</code> by | | | Renamed: <code>\Ops</code> to <code>\Ccool</code> | 13 |
| | <code>\OpsClear[\keyval list]</code> | 13 | | Renamed: <code>oops</code> to <code>ccool</code> (better | |
| | Replaced: <code>\Restore</code> by <code>\Read</code> | 13 | | describes the purpose) | 13 |
| | Replaced: <code>\Save</code> by <code>\Write</code> | 13 | v1.7 | General: Added: Legends to listings . | 13 |
| v1.3 | General: Replaced: <code>\OpsNew</code> by <code>\Ops</code> | 13 | | Added: Listing 12 (CUSUM) | 13 |
| | Replaced: <code>{\tl_1}</code> and <code>[\tl_1]</code> by | | | Deleted: <code>\CcoolDebug</code> | 13 |
| | <code><\tl_1></code> | 13 | | Deleted: Listing 5 from v1.6 | 13 |
| v1.4 | General: Added: section 4 | 13 | v1.8 | General: Added: <code>\CcoolLambda</code> | 13 |
| | Added: <code>\OpsDebug</code> | 13 | | Added: <code>\CcoolVers</code> | 13 |
| | Added: <code>\OpsHook</code> | 13 | | Added: Listing 10, Listing 11 | 13 |
| | Added: <code>Expans</code> (for debugging' | | | Added: Listing 1 | 13 |
| | sake, but...) | 13 | v1.9 | General: Added: support for LuaTeX | 13 |
| | Added: Listing 1., 2., and 3. | 13 | | Moved: from Part I to Part IV, | |
| | Deleted: Listing 1., and 2. | 13 | | what is now that part's section 11 | 13 |
| | Replaced: <code>s{\tl_3}{\tl_4}{\tl_5}</code> | | v2.0 | General: Added: support for XeTeX . | 13 |
| | by | | | Deleted: File's dependency on | |
| | <code>s{\tl_3}{\tl_3}{\tl_4}{\tl_3}{\tl_4}{\tl_5}</code> | | | <code>texosquery</code> and <code>\pdfcreationdate</code> | 13 |
| | | 13 | | Updated: <code>\RequirePackage</code> , | |
| v1.5 | General: Added: <code>File</code> | 13 | | <code>\NeedsTeXFormat</code> 's second | |
| | Deleted: dependence on <code>datetime</code> . | 13 | | argument / TeX Live 2020 | 13 |

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 | |
|--|---------------------|
| <code>*</code> (option) | <u>5</u> |
| <code>+</code> (option) | <u>5</u> |
| <code>\key_i</code> | <u>5</u> , <u>6</u> |
| <code><code_1></code> (option) | <u>4</u> |
| <code><code_2></code> (option) | <u>5</u> |
| <code><kvl_1></code> (option) | <u>5</u> |
| <code><tl_1></code> (option) | <u>4</u> |
| <code><tl_2></code> (option) | <u>4</u> |
| <code><tl_3></code> (option) | <u>5</u> |
| <code><tl_4></code> (option) | <u>5</u> |
| <code><tl_5></code> (option) | <u>5</u> |
| <code><tl_6></code> (option) | <u>5</u> |
| <code>Expans</code> (option) | <u>29</u> |
| <code>File</code> (option) | <u>29</u> |
| <code>Inner</code> (option) | <u>29</u> |
| <code>Name</code> (option) | <u>29</u> |
| <code>Outer</code> (option) | <u>30</u> |
| <code>Separ</code> (option) | <u>30</u> |
| <code>Write</code> (option) | <u>30</u> |

| | | | |
|----------------------------|-----------------------------------|------------------------------------|--|
| _ | 411, 412 | _ccool_make_ccool_exp:nnn | 147, 195 |
| | | _ccool_make_ccool_key:nnn | 159, 173 |
| | | _ccool_make_ccool_sideeffect:nnn | |
| | | | 171, 192, 201 |
| A | | _ccool_make_key:N | 143, 169 |
| \AtEndDocument | 93 | _ccool_make_key:n | 138, 145 |
| | | _ccool_make_key:Nn | 128, 140 |
| B | | \\g_ccool_option_expans_tl | 31, 35, 349 |
| \begin | 294 | _ccool_option_inner:n | 217, 221, 358 |
| \begingroup | 178 | \\g_ccool_option_inner_tl | |
| bool commands: | | | 219, 223, 363, 377, 391, 405 |
| \\bool_gset_false:N | 98 | _ccool_option_name:n | 225, 372 |
| \\bool_gset_true:N | 105 | \\g_ccool_option_name_tl | |
| \\bool_if:nTF | 120, 174, 197, 414 | | 132, 227, 233, 332, 362, 376, 390, 404 |
| \\bool_set_false:N | 94 | _ccool_option_outer:n | 235, 386 |
| \\BooleanFalse | 419, 420 | \\g_ccool_option_outer_tl | |
| | | | 237, 241, 365, 379, 393, 407 |
| C | | _ccool_option_separ:n | 243, 400 |
| \\Ccool | 1, 4, 5, 5, 14, 28, 179, 186, 207 | \\g_ccool_option_separ_tl | |
| ccool internal commands: | | | 245, 249, 364, 378, 392, 406 |
| _ccool_aux_inner:n | 5, 6, 37, 217 | _ccool_prop_append:NN | 251, 262 |
| _ccool_aux_inner_set:n | 3, 164 | _ccool_prop_append:Nn | 167, 260 |
| _ccool_aux_key:N | 16, 168 | _ccool_prop_append:nn | 253, 257 |
| _ccool_aux_key:n | 12, 19 | _ccool_prop_clear_new:n | 264, 271 |
| _ccool_aux_key:w | 8, 14 | _ccool_prop_clear_new_map:n | |
| \\g_ccool_aux_key_seq | | | 268, 334 |
| | 10, 18, 149, 169, 270, 271 | _ccool_prop_if_exist:nTF | 161, 273 |
| \\g_ccool_aux_keyval_seq | 165, 166, 168 | _ccool_prop_item:nn | 134, 277 |
| _ccool_aux_outer:n | 23, 151 | _ccool_prop_name:n | |
| _ccool_aux_outer_set:n | 21, 150 | | 82, 262, 266, 275, 279, 281, 284 |
| \\g_ccool_aux_prop | 25, 28, 45, 167 | _ccool_prop_new:n | 163, 282 |
| _ccool_aux_prop:N | 43, 166 | _ccool_seq_from_prop:n | 288, 292 |
| _ccool_aux_prop:n | 39, 49 | _ccool_seq_from_prop:NNn | 82, 286 |
| _ccool_aux_prop:nn | 25 | _ccool_seq_use:Nn | 154, 298 |
| _ccool_aux_prop:w | 31, 41 | _ccool_sys_date: | 304, 314 |
| _ccool_aux_separ:n | 75, 302 | _ccool_sys_date_hex: | 313, 328 |
| _ccool_aux_separ:nn | 52, 77 | _ccool_sys_filename: | 325, 355, 356 |
| _ccool_aux_val:Nn | 79, 149 | _ccool_sys_time: | 315, 324 |
| \\g_ccool_aux_val_seq | 81, 82, 155 | _ccool_sys_time_hex: | 323, 329 |
| _ccool_erw_seq_use:Nn | 294 | \\CcoolClear | 3, 14, 28, 331 |
| _ccool_lambda:nn | 84, 339 | \\CcoolDebug | 14 |
| _ccool_lambda_expression | 87, 90 | \\CcoolHook | 3, 5, 14, 28, 199, 336 |
| _ccool_log_close: | 92, 416 | \\CcoolLambda | 3, 14, 28, 337 |
| _ccool_log_entry | 179, 180 | \\CcoolOption | 3, 14, 29, 341 |
| \\g_ccool_log_file_tl | 100, 103, 353 | \\CcoolRead | 3, 12, 14, 31, 422 |
| \\g_ccool_log_iow | | \\CcoolVers | 3, 14, 31, 429 |
| | 92, 93, 97, 104, 122, 125 | cs commands: | |
| _ccool_log_open: | 100, 415 | \\cs_generate_variant:Nn | |
| \\g_ccool_log_open_bool | | | 6, 30, 74, 112, 127, 137, 142, 259 |
| | 94, 98, 105, 120, 174 | \\cs_gset:Npn | 5, 23, 245 |
| _ccool_log_read: | 113, 427 | \\cs_new:Nn | 52, 75, 225, |
| _ccool_log_read:n | 107, 115, 426 | | 273, 277, 298, 304, 313, 315, 323, 325 |
| \\g_ccool_log_to_tl | 103, 104, 115, 117 | \\cs_new:Npn | 281 |
| _ccool_log_write:n | 117, 176 | | |
| _ccool_make_ccool:nnnn | | | |
| | 184, 360, 374, 388, 402 | | |

| | |
|--|--|
| <code>\cs_new_protected:Nn</code> . . . 3, 12, 16, 21, 26, 39, 43, 79, 95, 101, 107, 113, 118, 128, 138, 143, 147, 159, 171, 217, 235, 243, 260, 264, 268, 282, 286 | <code>\keys_define:nn</code> 346 |
| <code>\cs_new_protected:Npn</code> 8, 32, 84, 184, 251 | <code>\keys_set:nn</code> 344 |
| <code>\cs_set:Nn</code> 253 | |
| <code>\cs_set_protected:Nn</code> 288 | |
| | M |
| D | <code>\meta</code> 295, 296 |
| <code>\DeclareDocumentCommand</code> 87, 186 | msg commands: |
| <code>\def</code> 179 | <code>\msg_error:nnn</code> 125, 231 |
| | <code>\msg_error:nnnn</code> 68 |
| | <code>\msg_new:nnn</code> 211, 212, 213, 214, 215, 216 |
| | <code>\msg_warning:nnn</code> 223, 241, 249 |
| | N |
| | <code>\NewDocumentCommand</code> 331, 336, 341, 422, 429 |
| E | |
| <code>\end</code> 297 | O |
| <code>\endgroup</code> 180 | <code>\Ops</code> 14 |
| <code>\ensuremath</code> 397, 398 | <code>\OpsClear</code> 14 |
| exp commands: | <code>\OpsDebug</code> 14 |
| <code>\exp_args:Nnf</code> 153 | <code>\OpsHook</code> 14 |
| <code>\exp_args:NNx</code> 86, 130, 186 | <code>\OpsNew</code> 14 |
| <code>\exp_args:No</code> 164, 266 | <code>\OpsOption</code> 14 |
| <code>\exp_args:Nx</code> 34 | <code>\OpsOptions</code> 14 |
| <code>\exp_last_unbraced:Nf</code> 359, 373, 387, 401 | <code>\OpsRead</code> 14 |
| <code>\exp_last_unbraced:NNf</code> 300 | <code>\OpsRestore</code> 13 |
| <code>\exp_not:N</code> 70, 223, 233, 241, 249 | <code>\OpsTest</code> 13, 14 |
| <code>\exp_not:n</code> 207 | options: |
| <code>\expandafter</code> 179 | <code>*</code> 5 |
| <code>\ExplSyntaxOff</code> 432 | <code>+</code> 5 |
| <code>\ExplSyntaxOn</code> 2 | <code><code₁></code> 4 |
| | <code><code₂></code> 5 |
| F | <code><kv₁></code> 5 |
| file commands: | <code><tl₁></code> 4 |
| <code>\file_input:n</code> 109 | <code><tl₂></code> 4 |
| | <code><tl₃></code> 5 |
| G | <code><tl₄></code> 5 |
| <code>\gappto</code> 199 | <code><tl₅></code> 5 |
| | <code><tl₆></code> 5 |
| I | <code>Expans</code> 29 |
| <code>\IfBooleanT</code> 193 | <code>File</code> 29 |
| <code>\IfValueT</code> 191, 205 | <code>Inner</code> 29 |
| <code>\IfValueTF</code> 425 | <code>Name</code> 29 |
| int commands: | <code>Outer</code> 30 |
| <code>\int_case:nnTF</code> 54 | <code>Separ</code> 30 |
| <code>\int_eval:n</code> 306, 317 | <code>Write</code> 30 |
| <code>\int_to_hex:n</code> 314, 324 | |
| iow commands: | P |
| <code>\iow_close:N</code> 93, 97 | <code>\pdfcreationdate</code> 14 |
| <code>\iow_new:N</code> 92 | prg commands: |
| <code>\iow_now:Nn</code> 122 | <code>\prg_replicate:nn</code> 57, 88 |
| <code>\iow_open:Nn</code> 104 | prop commands: |
| <code>\item</code> 295, 296 | <code>\prop_clear_new:N</code> 266 |
| | <code>\prop_gclear_new:N</code> 45 |
| K | <code>\prop_gput:Nnn</code> 28, 255 |
| keys commands: | <code>\prop_if_exist:NTF</code> 275 |
| <code>\l_keys_choice_tl</code> 349 | <code>\prop_item:Nn</code> 255, 279, 290 |

| | | | |
|-------------------------|-----------------------|--------------------|----------------------------|
| \prop_map_function:NN | 257 | \c_sys_jobname_str | 327 |
| \prop_new:N | 25, 284 | \c_sys_minute_int | 320 |
| \ProvideDocumentCommand | 131, 337 | \c_sys_month_int | 309 |
| | | \c_sys_year_int | 308 |
| Q | | | |
| quark commands: | | T | |
| \q_stop | 8, 14, 32, 41 | tl commands: | |
| R | | | |
| \Read | 14 | \c_empty_tl | 47, 66, 162, 182, 204, 336 |
| \Restore | 14 | \tl_count:n | 77 |
| S | | | |
| \Save | 14 | \tl_gset:Nn | 103, 219, 227, 237, 353 |
| seq commands: | | \tl_gset_eq:NN | 349 |
| \seq_gclear_new:N | 18, 81 | \tl_log:n | 110, 123 |
| \seq_gput_right:Nn | 10, 290 | \tl_new:N | 31, 100, 117 |
| \seq_if_empty:NTF | 46 | \tl_trim_spaces:n | 10, 36, 37 |
| \seq_map_function:NN | | U | |
| | 19, 49, 145, 271, 292 | use commands: | |
| \seq_set_from_clist:Nn | 165, 270 | \use:N | 35, 431 |
| \seq_use:Nnnn | 301 | \use_i:nn | 60, 62 |
| sys commands: | | \use_ii:nn | 61 |
| \c_sys_day_int | 310 | \usepackage | 4 |
| \c_sys_hour_int | 319 | W | |
| | | \Write | 14 |

Part IV

Implementation

```

1 <@@=ccool>
2 \ExplSyntaxOn

```

1 aux

```

\__ccool_aux_inner_set:n #1: <code>

3 \cs_new_protected:Nn \__ccool_aux_inner_set:n
4 {
5   \cs_gset:Npn \__ccool_aux_inner:n ##1 {#1}
6   \cs_generate_variant:Nn \__ccool_aux_inner:n { e }
7 }

(End definition for \__ccool_aux_inner_set:n.)

\__ccool_aux_key:w #1: <key>
#2: <value>

8 \cs_new_protected:Npn \__ccool_aux_key:w #1 = #2 \q_stop
9 {
10  \seq_gput_right:Nx \g__ccool_aux_key_seq { \tl_trim_spaces:n{#1} }
11 }

(End definition for \__ccool_aux_key:w.)

\__ccool_aux_key:n #1: <key = value>

12 \cs_new_protected:Nn \__ccool_aux_key:n
13 {
14  \__ccool_aux_key:w #1 \q_stop
15 }

(End definition for \__ccool_aux_key:n.)

\__ccool_aux_key:N #1: <seq>

16 \cs_new_protected:Nn \__ccool_aux_key:N
17 {
18  \seq_gclear_new:N \g__ccool_aux_key_seq
19  \seq_map_function:NN #1 \__ccool_aux_key:n
20 }

(End definition for \__ccool_aux_key:N.)

\__ccool_aux_outer_set:n #1: <inline code>

21 \cs_new_protected:Nn \__ccool_aux_outer_set:n
22 {
23  \cs_gset:Npn \__ccool_aux_outer:n ##1 {#1}
24 }

(End definition for \__ccool_aux_outer_set:n.)

```

```

\__ccool_aux_prop:nn
25 \prop_new:N \g__ccool_aux_prop
26 \cs_new_protected:Nn \__ccool_aux_prop:nn
27 {
28   \prop_gput:Nnn \g__ccool_aux_prop{#1}{#2}
29 }
30 \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 >

31 \tl_new:N \g__ccool_option_expans_tl
32 \cs_new_protected:Npn \__ccool_aux_prop:w #1 = #2 \q_stop
33 {
34   \exp_args:Nx
35   \use:c{\__ccool_aux_prop:\g__ccool_option_expans_tl}
36   { \tl_trim_spaces:n{#1} }
37   { \__ccool_aux_inner:n{ \tl_trim_spaces:n{#2} } }
38 }

(End definition for \__ccool_aux_prop:w.)

```

```

\__ccool_aux_prop:n #1 : < key = value >

39 \cs_new_protected:Nn \__ccool_aux_prop:n
40 {
41   \__ccool_aux_prop:w #1 \q_stop
42 }

(End definition for \__ccool_aux_prop:n.)

```

```

\__ccool_aux_prop:N #1 : < keyval list >

43 \cs_new_protected:Nn \__ccool_aux_prop:N
44 {
45   \prop_gclear_new:N \g__ccool_aux_prop
46   \seq_if_empty:NTF #1
47   { \c_empty_tl }
48   {
49     \seq_map_function:NN #1 \__ccool_aux_prop:n
50   }
51 }

(End definition for \__ccool_aux_prop:N.)

```

```

\__ccool_aux_separ:nn #1 : < int >
#2 : < tokens >

52 \cs_new:Nn \__ccool_aux_separ:nn
53 {
54   \int_case:nnTF {#1}
55   {
56     {1}
57     { \prg_replicate:nn{ 3 }{#2} }
58     {2}
59     {

```

```

60     { \use_i:nn #2 }
61     { \use_ii:nn #2 }
62     { \use_i:nn #2 }
63   }
64   {3}{#2}
65 }
66 { \c_empty_tl }
67 {
68   \msg_error:nnnn { __ccool }
69   { separ }
70   { \exp_not:N \__ccool_aux_separ:nn }
71   {#2}
72 }
73 }
74 \cs_generate_variant:Nn \__ccool_aux_separ:nn { e }

```

(End definition for __ccool_aux_separ:nn.)

```

\__ccool_aux_separ:n #1: < tokens >
75 \cs_new:Nn \__ccool_aux_separ:n
76 {
77   \__ccool_aux_separ:en{ \tl_count:n{#1} }{#1}
78 }

```

(End definition for __ccool_aux_separ:n.)

```

\__ccool_aux_val:Nn #1: < seq >
#2: < tl var name >
79 \cs_new_protected:Nn \__ccool_aux_val:Nn
80 {
81   \seq_gclear_new:N \g__ccool_aux_val_seq
82   \__ccool_seq_from_prop:NNn \g__ccool_aux_val_seq #1 { \__ccool_prop_name:n{#2} }
83 }

```

(End definition for __ccool_aux_val:Nn.)

2 lambda

```

\__ccool_lambda:nn [7]
84 \cs_new_protected:Npn \__ccool_lambda:nn #1 #2
85 {
86   \exp_args:NNx
87   \DeclareDocumentCommand \__ccool_lambda_expression
88   { \prg_replicate:nn { #1 } { m } }
89   {#2}
90   \__ccool_lambda_expression
91 }

```

(End definition for __ccool_lambda:nn.)

3 log

_ccool_log_close:

```

92 \iow_new:N \g__ccool_log_iow
93 \AtEndDocument{\iow_close:N \g__ccool_log_iow}
94 \bool_set_false:N \g__ccool_log_open_bool
95 \cs_new_protected:Nn \_ccool_log_close:
96 {
97   \iow_close:N \g__ccool_log_iow
98   \bool_gset_false:N \g__ccool_log_open_bool
99 }

```

(End definition for _ccool_log_close:.)

_ccool_log_open:

```

100 \tl_new:N \g__ccool_log_file_tl
101 \cs_new_protected:Nn \_ccool_log_open:
102 {
103   \tl_gset:Nx \g__ccool_log_to_tl{\g__ccool_log_file_tl}
104   \iow_open:Nn \g__ccool_log_iow {\g__ccool_log_to_tl}
105   \bool_gset_true:N \g__ccool_log_open_bool
106 }

```

(End definition for _ccool_log_open:.)

_ccool_log_read:n #1 : $\langle path \rangle$

```

107 \cs_new_protected:Nn \_ccool_log_read:n
108 {
109   \file_input:n{#1}
110   \tl_log:n{read~from~#1}
111 }
112 \cs_generate_variant:Nn \_ccool_log_read:n { e }

```

(End definition for _ccool_log_read:n.)

_ccool_log_read:

```

113 \cs_new_protected:Nn \_ccool_log_read:
114 {
115   \_ccool_log_read:e{\g__ccool_log_to_tl}
116 }

```

(End definition for _ccool_log_read:.)

_ccool_log_write:n

```

117 \tl_new:N \g__ccool_log_to_tl
118 \cs_new_protected:Nn \_ccool_log_write:n
119 {
120   \bool_if:nTF{ \g__ccool_log_open_bool }
121   {
122     \iow_now:Nn \g__ccool_log_iow {#1}
123     \tl_log:n{ write~to~#1 }
124   }
125   { \msg_error:nnnn{ __ccool }{ iow }{ \g__ccool_log_iow } } }
126 }
127 \cs_generate_variant:Nn \_ccool_log_write:n { e }

```

(End definition for _ccool_log_write:n.)

4 make_key

```

__ccool_make_key:Nn #1 : < token >
#2 : < key >

128 \cs_new_protected:Nn \__ccool_make_key:Nn
129 {
130   \exp_args:NNx
131   \ProvideDocumentCommand{#1}
132   { D<>{\g__ccool_option_name_tl} }
133   {
134     \__ccool_prop_item:nn{#1}{#2}
135   }
136 }
137 \cs_generate_variant:Nn \__ccool_make_key:Nn {c}

(End definition for \__ccool_make_key:Nn.)

__ccool_make_key:n #1 : < key >

138 \cs_new_protected:Nn \__ccool_make_key:n
139 {
140   \__ccool_make_key:cn{#1}{#1}
141 }
142 \cs_generate_variant:Nn \__ccool_make_key:n { e }

(End definition for \__ccool_make_key:n.)

__ccool_make_key:N #1 : < seq >

143 \cs_new_protected:Nn \__ccool_make_key:N
144 {
145   \seq_map_function:NN #1 \__ccool_make_key:e
146 }

(End definition for \__ccool_make_key:N.)

```

5 make_ccool

```

__ccool_make_ccool_exp:nnn

147 \cs_new_protected:Nn \__ccool_make_ccool_exp:nnn
148 {
149   \__ccool_aux_val:Nn \g__ccool_aux_key_seq {#1}
150   \__ccool_aux_outer_set:n{#3}
151   \__ccool_aux_outer:n
152   {
153     \exp_args:NNf
154     \__ccool_seq_use:Nn
155     \g__ccool_aux_val_seq
156     {#2}
157   }
158 }

(End definition for \__ccool_make_ccool_exp:nnn.)

```

_ccool_make_ccool_key:nnn

```

159 \cs_new_protected:Nn \_ccool\_make\_ccool\_key:nnn
160 {
161   \_ccool\_prop\_if\_exist:nTF{#1}
162   { \c_empty_tl }
163   { \_ccool\_prop\_new:n{#1} }
164   \exp_args:No \_ccool\_aux\_inner\_set:n{#2}
165   \seq_set_from_clist:Nn \g\_ccool\_aux\_keyval\_seq {#3}
166   \_ccool\_aux\_prop:N \g\_ccool\_aux\_keyval\_seq
167   \_ccool\_prop\_append:Nn \g\_ccool\_aux\_prop {#1}
168   \_ccool\_aux\_key:N \g\_ccool\_aux\_keyval\_seq
169   \_ccool\_make\_key:N \g\_ccool\_aux\_key\_seq
170 }

```

(End definition for _ccool_make_ccool_key:nnn.)

_ccool_make_ccool_sideeffect:nnn [9]

```

171 \cs_new_protected:Nn \_ccool\_make\_ccool\_sideeffect:nnn
172 {
173   \_ccool\_make\_ccool\_key:nnn{#1}{#2}{#3}
174   \bool_if:nTF{ \g\_ccool\_log\_open\_bool }
175   {
176     \_ccool\_log\_write:n
177     {
178       \begingroup
179       \def \_ccool\_log\_entry { \Ccool<#1>i{#2}{#3} } \expandafter
180       \endgroup \_ccool\_log\_entry
181     }
182   }{\c_empty_tl}
183 }

```

(End definition for _ccool_make_ccool_sideeffect:nnn.)

_ccool_make_ccool:nnnn #1 : < token list >
 #2 : < seq₁ >
 #3 : < seq₂ >
 #4 : < prop >

```

184 \cs_new_protected:Npn \_ccool\_make\_ccool:nnnn #1 #2 #3 #4
185 {
186   \exp_args:NNx \DeclareDocumentCommand \Ccool
187   {%^^A      2      3      4 5 6      7 8      9
188     D<>{#1} +o E{ i }{{#2}} m t+ s E{ s o }{{#3}{#4}} +o
189   }
190   {
191     \IfValueT{##2}{##2}
192     \_ccool\_make\_ccool\_sideeffect:nnn{##1}{##3}{##4}
193     \IfBooleanT{##6}
194     {
195       \_ccool\_make\_ccool\_exp:nnn{##1}{##7}{##8}
196     }
197     \bool_if:nTF{##5}
198     {
199       \gappto{\CcoolHook}
200     }

```

```

201     \__ccool_make_ccool_sideeffect:nnn{##1}{##3}{##4}
202   }
203 }
204 {\c_empty_tl}
205 \IfValueT{##9}
206 {
207   \exp_not:n{ \Ccool<##1>[##9] }
208 }
209 }
210 }

```

(End definition for __ccool_make_ccool:nnnn.)

6 msg

```

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

```

7 option

__ccool_aux_inner:n #1: *<code>*

```

217 \cs_new_protected:Nn \__ccool_option_inner:n
218 {
219   \tl_gset:Nn \g__ccool_option_inner_tl {#1}
220 }
221 \__ccool_option_inner:n
222 {
223   \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_inner_tl }
224 }

```

(End definition for __ccool_aux_inner:n.)

__ccool_option_name:n #1: *<token list>*

```

225 \cs_new:Nn \__ccool_option_name:n
226 {
227   \tl_gset:Nn \g__ccool_option_name_tl{#1}
228 }
229 \__ccool_option_name:n
230 {
231   \msg_error:nnx{ __ccool }
232   { generic }
233   { \exp_not:N\g__ccool_option_name_tl~undefined }
234 }

```

(End definition for __ccool_option_name:n.)

__ccool_option_outer:n #1: *<inline code>*

```

235 \cs_new_protected:Nn \__ccool_option_outer:n
236 {

```

```

237 \tl_gset:Nn \g__ccool_option_outer_tl {#1}
238 }
239 \__ccool_option_outer:n
240 {
241 \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_outer_tl }
242 }

```

(End definition for __ccool_option_outer:n.)

```

\__ccool_option_separ:n #1 : {< tl1 >}{< tl2 >}{< tl3 >}
243 \cs_new_protected:Nn \__ccool_option_separ:n
244 {
245 \cs_gset:Npn \g__ccool_option_separ_tl {#1}
246 }
247 \__ccool_option_separ:n
248 {
249 \msg_warning:nnn{ __ccool }{ unset }{ \exp_not:N \g__ccool_option_separ_tl }
250 }

```

(End definition for __ccool_option_separ:n.)

8 prop

```

\__ccool_prop_append:NN #1 : < prop1 >
#2 : < prop2 >
251 \cs_new_protected:Npn \__ccool_prop_append:NN #1 #2
252 {
253 \cs_set:Nn \__ccool_prop_append:nn
254 {
255 \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
256 }
257 \prop_map_function:NN #2 \__ccool_prop_append:nn
258 }
259 \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 >
260 \cs_new_protected:Nn \__ccool_prop_append:Nn
261 {
262 \__ccool_prop_append:cN{ \__ccool_prop_name:n {#2} } #1
263 }

```

(End definition for __ccool_prop_append:Nn.)

```

\__ccool_prop_clear_new:n #1 : < tl var name >
264 \cs_new_protected:Nn \__ccool_prop_clear_new:n
265 {
266 \exp_args:No \prop_clear_new:c{ \__ccool_prop_name:n {#1} }
267 }

```

(End definition for __ccool_prop_clear_new:n.)

```

\__ccool_prop_clear_new_map:n #1 : < keyval list >
268 \cs_new_protected:Nn \__ccool_prop_clear_new_map:n
269 {
270   \seq_set_from_clist:Nn \g__ccool_aux_key_seq {#1}
271   \seq_map_function:NN \g__ccool_aux_key_seq \__ccool_prop_clear_new:n
272 }
(End definition for \__ccool_prop_clear_new_map:n.)

```

```

\__ccool_prop_if_exist:nTF #1 : < tl1 >
#2 : < tl2 >
#3 : < tl3 >
273 \cs_new:Nn \__ccool_prop_if_exist:nTF
274 {
275   \prop_if_exist:cTF{ \__ccool_prop_name:n {#1} }{#2}{#3}
276 }
(End definition for \__ccool_prop_if_exist:nTF.)

```

```

\__ccool_prop_item:nn #1 : < tl var name >
#2 : < key >
277 \cs_new:Nn \__ccool_prop_item:nn
278 {
279   \prop_item:cn { \__ccool_prop_name:n {#1} } {#2}
280 }
(End definition for \__ccool_prop_item:nn.)

```

```

\__ccool_prop_name:n #1 : < tl var name >
281 \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 >
282 \cs_new_protected:Nn \__ccool_prop_new:n
283 {
284   \prop_new:c{ \__ccool_prop_name:n {#1} }
285 }
(End definition for \__ccool_prop_new:n.)

```

9 seq

```

\__ccool_seq_from_prop:NNn #1 : < seq1 >
#2 : < seq2 > (keys)
#3 : < prop >
286 \cs_new_protected:Nn \__ccool_seq_from_prop:NNn
287 {
288   \cs_set_protected:Nn \__ccool_seq_from_prop:n
289   {
290     \seq_gput_right:No #1 { \prop_item:cn{#3}{##1} }
291   }
292   \seq_map_function:NN #2 \__ccool_seq_from_prop:n
293 }

```

(End definition for _ccool_seq_from_prop:Nn.)

_ccool_erw_seq_use:Nn

```

294 %      \begin{arguments}
295 %      \item \meta{ seq }
296 %      \item \meta{ tokens }
297 %      \end{arguments}
298 \cs_new:Nn \_ccool_seq_use:Nn
299 {
300   \exp_last_unbraced:NNf
301   \seq_use:Nnnn #1
302   \_ccool_aux_separ:n{#2}
303 }

```

(End definition for _ccool_erw_seq_use:Nn.)

10 sys

_ccool_sys_date:

```

304 \cs_new:Nn \_ccool_sys_date:
305 {
306   \int_eval:n
307   {
308     \c_sys_year_int * 10000
309     +\c_sys_month_int * 100
310     +\c_sys_day_int * 1
311   }
312 }

```

(End definition for _ccool_sys_date:.)

_ccool_sys_date_hex:

```

313 \cs_new:Nn \_ccool_sys_date_hex:
314 {\int_to_hex:n{\_ccool_sys_date:}}

```

(End definition for _ccool_sys_date_hex:.)

_ccool_sys_time:

```

315 \cs_new:Nn \_ccool_sys_time:
316 {
317   \int_eval:n
318   {
319     \c_sys_hour_int * 100
320     +\c_sys_minute_int * 1
321   }
322 }

```

(End definition for _ccool_sys_time:.)

_ccool_sys_time_hex:

```

323 \cs_new:Nn \_ccool_sys_time_hex:
324 {\int_to_hex:n{\_ccool_sys_time:}}

```

(End definition for _ccool_sys_time_hex:.)

`_ccool_sys_filename:`

```

325 \cs_new:Nn\_ccool_sys_filename:
326 {
327   \c_sys_jobname_str--
328   \_ccool_sys_date_hex--
329   \_ccool_sys_time_hex:
330 }

```

(End definition for _ccool_sys_filename:.)

11 Front-end

\CcoolClear

#1 : $\langle token\ list \rangle$

Semantics Clears any data created by `\Ccool{ $\langle token\ list \rangle$ }`

```

331 \NewDocumentCommand{ \CcoolClear }
332 { D<>{\g_ccool_option_name_tl} }
333 {
334   \_ccool_prop_clear_new_map:n{#1}
335 }

```

\CcoolHook

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

```

336 \NewDocumentCommand{\CcoolHook}{*}{\c_empty_tl}

```

\CcoolLambda

#1 : $\langle integer \rangle$

#2 : $\langle code \rangle$

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

Semantics Creates a lambda expression with $\langle integer \rangle$ arguments for $\langle code \rangle$

```

337 \ProvideDocumentCommand \CcoolLambda { 0{1} m }
338 {
339   \_ccool_lambda:nn { #1 } { #2 }
340 }

```

\CcoolOption**#1** : *<keyval list>*

```
341 \NewDocumentCommand{ \CcoolOption }
342 { m }
343 {
344   \keys_set:nn{ __ccool }{#1}
345 }

346 \keys_define:nn { __ccool }
347 {
```

Expans**Value** *eo|ee|ex|xo|xe|xx*

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

File**Value** *<path>*

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

Inner**Value** *<code>*, with **####1** as the argument to be replaced

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

Name**Value** *<token list>*

```
371 Name .code:n={
372   \__ccool_option_name:n{#1}
373   \exp_last_unbraced:Nf
374   \__ccool_make_ccool:nnnn
375   {
```

```

376     { \g__ccool_option_name_tl }
377     { \g__ccool_option_inner_tl }
378     { \g__ccool_option_separ_tl }
379     { \g__ccool_option_outer_tl }
380   }
381 },
382 Name .value_required:n = false,
383 Name .default:n = { Math },
384 Name .initial:n = { Math },

```

Outer

Value $\langle code \rangle$, with `####1` as the argument to be replaced

```

385 Outer .code:n={
386   \__ccool_option_outer:n{#1}
387   \exp_last_unbraced:Nf
388   \__ccool_make_ccool:nnnn
389   {
390     { \g__ccool_option_name_tl }
391     { \g__ccool_option_inner_tl }
392     { \g__ccool_option_separ_tl }
393     { \g__ccool_option_outer_tl }
394   }
395 },
396 Outer .value_required:n = false,
397 Outer .default:n = { \ensuremath{####1} },
398 Outer .initial:n = { \ensuremath{####1} },

```

Separ

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

```

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

```

Write

Value $\langle boolean \rangle$

```

413 Write .code:n = {
414   \bool_if:nTF{#1}
415   {\__ccool_log_open:}
416   {\__ccool_log_close:}
417 },

```

```

418 Write .value_required:n = false,
419 Write .default:n = \BooleanFalse,
420 Write .initial:n = \BooleanFalse
421 }

```

\CcoolRead #1 : $\langle path \rangle$

Semantics

1. Reads the definitions in $\langle path \rangle$.
2. Writes to `ccool.log`: ‘read from $\langle path \rangle$ ’

```

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

```

\CcoolVers

Semantics Expands to the package’s version

```

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

```

16 Misc

```

432 \ExplSyntaxOff

```