The erw-l3 package *

Erwann Rogard†

Released 2020/05/22

Abstract

Utilities like expl3[1].

Résumé

Utilitaires de type expl3[1].

Contents

*This file describes version v2.6, last revised 2020/05/22.

†firstname dot lastname AusTria gmail dot com

Part I
Usage

\usepackage `\usepackage{erw-l3}`

Requirement

1. `erw-l3.sty` and its dependencies are in the path of the LaTeX engine. See Part III, section 3.
2. Goes in the *preamble*

2 cs

\erw_cs_apply:Nn
\erw_cs_apply:(No|Nf|Nx|cn)
\erw_cs_apply:Nnn
\erw_cs_apply:Nnnn
\erw_cs_apply:Nnnnn
`\erw_cs_apply:Nn {⟨cs⟩}{⟨token list$_1$⟩}`

\erw_cs_identity:n `\erw_cs_identity:n{⟨arg⟩}`

\erw_cs_set_inline:Nn
\erw_cs_set_inline:cn
`\erw_cs_set_inline:Nn{⟨cs⟩}{⟨code⟩}`

3 csint

\erw_csint:nn `\erw_csint:nn{⟨integer⟩}{⟨arg⟩}`

\erw_csint_name:n `\erw_csint_name:n{⟨integer⟩}`

\erw_csint_names:nnn `\erw_csint_names:nnn{⟨integer⟩}{⟨integer⟩}{⟨integer⟩}`

\erw_csint_names_braced:
\erw_csint_names_braced:n
\erw_csint_names_braced:nnn

\erw_csint_new:n \erw_csint_new:n{⟨*integer*⟩}

\erw_csint_reset: \erw_csint_reset:

4 int

\erw_int_range:n \erw_int_range:n{⟨*integer*⟩}
\erw_int_range:nn

5 lambda

\erw_lambda:nnn \erw_lambda:nnn⟨*token*⟩{⟨*arg spec*⟩}{⟨*code*⟩}

6 option

\erw_option:n \erw_option:n{⟨*keyval list*⟩}

oper / fold_set_par
oper / fold_apply_par
sys / timestamp_delim

7 prop

All functions that modify a ⟨*prop*⟩ check it exists, if not make sure it does.

\erw_prop_keyval_parse:NNNn \erw_prop_keyval_parse:NNNn⟨*prop*⟩⟨*cs*$_1$⟩⟨*cs*$_2$⟩{⟨*keyval list*⟩}

\erw_prop_map_item:NNN \erw_prop_map_item:NNN⟨*cs*⟩⟨*prop*$_1$⟩⟨*prop*$_2$⟩

\erw_prop_to_clist:Nn \erw_prop_to_clist:Nn⟨*prop*⟩{⟨*key*$_1$⟩,...}

8 seq

All functions that modify a ⟨*seq*⟩ check it exists, if not make sure it does.

\erw_seq_compose:nN \erw_seq_compose:nN{{⟨*cs*$_1$⟩}...}⟨*seq*⟩

\erw_seq_compose_c:nN \erw_seq_compose_c:nN{{⟨*cs name*$_1$⟩}...}⟨*seq*⟩

\erw_seq_compose_vers:nN \erw_seq_compose:nN{{⟨cs or code$_1$⟩}...}⟨seq⟩

\erw_seq_put_right_clist:Nn
\erw_seq_put_right_clist:cn \erw_seq_put_right_clist:Nn⟨seq⟩{⟨clist⟩}

\erw_seq_put_right_prop:NNn \erw_seq_put_right_prop:NNn⟨seq⟩⟨prop⟩{⟨keyval list⟩}

\erw_seq_use:Nn \erw_seq_use:Nn⟨seq⟩{⟨items⟩}

Also see [1, Section 8 of l3seq]

Semantics \seq_use:Nnnn⟨seq⟩\erw_tl_separators:n{⟨items⟩}

9 sys

\erw_sys_jobnametimestamp:nn
\erw_sys_jobnametimestamp: \erw_sys_jobnametimestamp:nn{date|time|datetime}{10|16}

\erw_sys_timestamp:nn
\erw_sys_timestamp: \erw_sys_timestamp:nn{date|time|datetime}{10|16}

Semantics Timestamp in base 10 or 16

\erw_sys_timestamp_delimiter: \erw_sys_timestamp_delimiter:

10 tl

All functions that modify a ⟨*token list*⟩ check it exists, if not make sure it does.

\erw_tl_append_item:nn \erw_tl_append_item:nn{⟨arg list⟩}{⟨arg⟩}

\erw_tl_compose:nN
\erw_tl_compose:nn \erw_tl_compose:nn{{cs_1}...}{⟨token list⟩}

\erw_tl_compose_c:nN
\erw_tl_compose_c:nn \erw_tl_compose_c:nn{{cs name_1}...}{⟨token list⟩}

\erw_tl_compose_vers:nN
\erw_tl_compose_vers:nn \erw_tl_compose_vers:nn{{cs or code_1}...}{⟨token list⟩}

\erw_tl_fold:NN
\erw_tl_fold:cN \erw_tl_fold:NN⟨cs⟩⟨tl var⟩

\erw_tl_gset_function:N
\erw_tl_gset_function:n
\erw_tl_gset_function:n{⟨*code*⟩}

\erw_tl_join:nn
\erw_tl_join:nnn
\erw_tl_join:nnnn
\erw_tl_join:nnnnn
\erw_tl_join:nn{⟨*token list*$_1$⟩}{⟨*token list*$_2$⟩}

\erw_tl_last_item:n
\erw_tl_last_item:n{⟨*token list*⟩}

\erw_tl_map:n
\erw_tl_map:Nn
\erw_tl_map:n{⟨*items*⟩}

Semantics Maps over ⟨*items*⟩ using the internal function set by \erw_tl_gset_function:n

\erw_tl_map_inline:nn
\erw_tl_map_inline:nn{⟨*code*⟩}{⟨*items*⟩}

\erw_tl_map_thread:Nn
\erw_tl_math_thread:Nn⟨*cs*⟩{⟨*items*⟩}

\erw_tl_map_thread_at:Nnn
\erw_tl_math_thread_at:Nnn{⟨*integer*⟩}{⟨*token list*⟩}

\erw_tl_repeat:nn
\erw_tl_repeat:nn{⟨*integer*⟩}{⟨*token list*⟩}

\erw_tl_split:nnn
\erw_tl_split:nn
\erw_tl_split:nn{⟨*items*⟩}{⟨*delimiter*⟩}

\erw_tl_separators:n
\erw_tl_separators:n{⟨*items*⟩}

Semantics According to the count of ⟨*items*⟩:

1) {⟨*token list*$_1$⟩}{⟨*token list*$_1$⟩}{⟨*token list*$_1$⟩}

2) {⟨*token list*$_1$⟩}{⟨*token list*$_2$⟩}{⟨*token list*$_1$ *token list*$_2$⟩}

3) {⟨*token list*$_1$⟩}{⟨*token list*$_2$⟩}{⟨*token list*$_3$⟩}

Part II
Listing

1 constants

Listing 1.

```
\ExplSyntaxOn
\seq_const_from_clist:Nn \foo_seq{ A, B, C }
\prop_const_from_keyval:Nn \foo_prop{ A = a, B = b, C = c }
\ExplSyntaxOff
```

2 cs

3 csint

Listing 2.

```
\ExplSyntaxOn
\cs_set:Nn\__foo:n{ f(#1) }
\cs_set:Nn\__baz:n{ h\{#1\} }
\tl_map_function:nN { {\__baz:n} {g[#1]} {\__foo:n} }\erw_csint_new:n
\exp_last_unbraced:Nx
\erw_tl_compose_c:nn
{{\erw_csint_names_braced:nnn{ 1 }{ 1 }{ 3 }}
  { X }}
\ExplSyntaxOff
```

h{g[f(X)]}

4 int

Listing 3.

```
\ExplSyntaxOn
\erw_int_range:nn{ 2 }{ 5 }\\
\erw_int_range:n{ 5 }
\ExplSyntaxOff
```

2345
12345

5 lambda

Listing 4.

```
\ExplSyntaxOn
\tl_set:Nn \l_tmpa_tl
{
  \erw_lambda:nnn \DeclareDocumentCommand{ m }{ Hello,~#1! }
}
\l_tmpa_tl{ world }
\ExplSyntaxOff
```

Hello, world!

6 prop

Listing 5.

```
\ExplSyntaxOn
\erw_prop_map_item:NNN \prop_put:Nnx \baz_prop \foo_prop
\prop_if_exist:NTF\baz_prop{T}{F}\\
\prop_item:Nn \baz_prop{ A }
,\prop_item:Nn \baz_prop{ B }
,\prop_item:Nn \baz_prop{ C }
\ExplSyntaxOff
```

T
a,b,c

Listing 6.

```
\ExplSyntaxOn
\erw_prop_keyval_parse:NNNn
\foo_prop
\erw_cs_error:n
\prop_put:Nnn{ X = x, Y = y, Z = z}
\prop_item:Nn \foo_prop{ X }
,\prop_item:Nn \foo_prop{ Y }
,\prop_item:Nn \foo_prop{ Z }
\ExplSyntaxOff
```

x,y,z

Listing 7.

```
\ExplSyntaxOn
\erw_prop_to_clist:Nn \foo_prop{ A, B, C }
\ExplSyntaxOff
```

a,b,c

7 seq

Listing 8.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { f(#1) }
\cs_set:Nn \__bar:n { g[#1] }
\cs_set:Nn \__baz:n { h\{#1\} }
\seq_new:N \l_tmp_seq
\seq_put_right:Nn \l_tmp_seq{X}
\erw_seq_compose:nN{ {\__baz:n}{\__bar:n}{\__foo:n} }\l_tmp_seq
\seq_item:Nn \l_tmp_seq{ 1 }\\
\seq_item:Nn \l_tmp_seq{ 2 }\\
\seq_item:Nn \l_tmp_seq{ 3 }\\
\seq_item:Nn \l_tmp_seq{ 4 }
\ExplSyntaxOff
```

X
f(X)
g[f(X)]
h{g[f(X)]}

Listing 9.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { f(#1) }
\cs_set:Nn \__bar:n { g[#1] }
\cs_set:Nn \__baz:n { h\{#1\} }
\seq_put_right:Nn \l_tmpa_seq{X}
\erw_seq_compose_c:nN{ {__baz:n}{__bar:n}{__foo:n} }\l_tmpa_seq
\seq_item:Nn \l_tmpa_seq{ 1 }\\
\seq_item:Nn \l_tmpa_seq{ 2 }\\
\seq_item:Nn \l_tmpa_seq{ 3 }\\
\seq_item:Nn \l_tmpa_seq{ 4 }
\ExplSyntaxOff
```

X
f(X)
g[f(X)]
h{g[f(X)]}

Listing 10.

```
\ExplSyntaxOn
\erw_seq_put_right_prop:NNn \bar_seq\foo_prop{ A, B, C }
\seq_use:Nn\bar_seq{,}
\ExplSyntaxOff
```

a,b,c

Listing 11.

```
\ExplSyntaxOn
\seq_put_right:Nn\l_tmpa_seq{ A }
\seq_put_right:Nn\l_tmpa_seq{ B }
\erw_seq_use:Nn \l_tmpa_seq{ {~and~} }\\
\erw_seq_use:Nn \l_tmpa_seq{ {,\ }{~and~} }\\
\erw_seq_use:Nn \l_tmpa_seq{ {~and~}{,\ }{,~and~} }\\[1em]
\seq_put_right:Nn\l_tmpa_seq{ C }
\erw_seq_use:Nn \l_tmpa_seq{ {~and~} }\\
\erw_seq_use:Nn \l_tmpa_seq{ {,\ }{and~} }\\
\erw_seq_use:Nn \l_tmpa_seq{ {~and~}{,\ }{,~and~} }\\
\ExplSyntaxOff
```

A and B
A and B
A and B

A and B and C
A, B, and C
A, B, and C

8 sys

Listing 12.

```
\ExplSyntaxOn
\noindent\erw_sys_timestamp:nn{date}{10}{-}
\noindent\erw_sys_timestamp:nn{time}{10}\\
\noindent\erw_sys_timestamp:nn{datetime}{10}\\
\erw_sys_timestamp:nn{date}{16}{\%}
\erw_sys_timestamp:nn{time}{16}\\
\erw_option:n{ sys / timestamp_delim = {\%} }
\erw_sys_timestamp:nn{datetime}{16}\\
\erw_sys_jobnametimestamp:
\ExplSyntaxOff
```

```
20200524-2137
20200524-2137
1343c4c%859
1343c4c%859
erw-l3%1343c4c%859
```

Listing 13.

```
\ExplSyntaxOn
\erw_option:n{ sys / timestamp_delim = \c_empty_tl }
\iow_new:N \foo_iow
\tl_set:Nx \foo_dec { \erw_sys_timestamp:nn{datetime}{10} }
\tl_set:Nx \foo_hex { \erw_sys_timestamp: }
\iow_open:Nn \foo_iow{ \foo_hex }
\iow_now:Nn\foo_iow{ Hello,\ world! }
\iow_close:N \foo_iow
D:\foo_dec\\
\file_timestamp:n{ \foo_hex }\\
\file_input:n{ \foo_hex }
\ExplSyntaxOff
```

```
D:202005242137
D:20200524213720-04'00'
Hello, world!
```

9 tl

Listing 14.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { f(#1) }
\cs_set:Nn \__bar:n { g[#1] }
\cs_set:Nn \__baz:n { h\{#1\} }
\tl_set:Nn \l_tmpa_tl{ X }
\erw_tl_compose:nN{ {\__baz:n}{\__bar:n}{\__foo:n} }\l_tmpa_tl
\l_tmpa_tl\\
\tl_set:Nn \l_tmpa_tl{ X }
\erw_tl_compose:nn{ {\__baz:n}{\__bar:n}{\__foo:n}}{ X  }\\
\ExplSyntaxOff
```

```
h{g[f(X)]}
h{g[f(X)]}
```

Listing 15.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n  { f(#1) }
\cs_set:Nn \__bar:n  { g[#1] }
\cs_set:Nn \__baz:n  { h\{#1\} }
\tl_set:Nn \l_tmpa_tl{ X }
\erw_tl_compose_c:nnN{ {__baz:n}{__bar:n}{__foo:n} }\l_tmpa_tl
\l_tmpa_tl\\
\erw_tl_compose_c:nn{ {__baz:n}{__bar:n}{__foo:n}}{X }
\ExplSyntaxOff
```

---

h{g[f(X)]}
h{g[f(X)]}

Listing 16.

```
\ExplSyntaxOn
\cs_set:Npn \__foo #1 { f(#1) }
\cs_set:Npn \__bar #1 { g[#1] }
\cs_set:Npn \__baz #1 { h\{#1\} }
\erw_tl_compose_vers:nn{ {\__baz}{g[#1]}{\__foo}}{X }
\ExplSyntaxOff
```

---

h{g[f(X)]}

Listing 17.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { f(#1) }
\tl_set:Nn \l_tmpa_tl{ X }
\erw_tl_fold:NN\__foo:n\l_tmpa_tl
\l_tmpa_tl\\
\cs_set:Nn \__bar:n { g[#1] }
\erw_tl_fold:cN {__bar:n}\l_tmpa_tl
\l_tmpa_tl
\ExplSyntaxOff
```

---

f(X)
g[f(X)]

Listing 18.

```
\ExplSyntaxOn
\erw_tl_repeat:nn{ 3 }{ x }
\ExplSyntaxOff
```

---

xxx

Listing 19.

```
\ExplSyntaxOn
\erw_tl_split:nn{ {a} {b} {c} }{ == }
\ExplSyntaxOff
```

a==b==c

Listing 20.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { (#1) }
\erw_tl_map:Nn \__foo:n{ {a}{b}{c} }
\ExplSyntaxOff
```

(a)(b)(c)

Listing 21.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { (#1) }
\erw_tl_map_thread:Nn \__foo:n
{
  { {a}{b}{c}{d}{e}{f} }
}\\
\cs_set:Nn\__foo:nn { (#1+#2) }
\erw_tl_map_thread:Nn \__foo:nn
{
  { {a}{b}{c}{d}{e}{f} }
  { {A}{B}{C}{D}{E}{F} }
}\\
\cs_set:Nn \__foo:nnn { (#1+#2+#3) }
\erw_tl_map_thread:Nn \__foo:nnn
{
  { {a}{b}{c}{d}{e}{f} }
  { {A}{B}{C}{D}{E}{F} }
  { {k}{l}{m}{n}{o}{p} }
}\\
\cs_set:Nn \__foo:nnnn { (#1+#2+#3+#4) }
\erw_tl_map_thread:Nn \__foo:nnnn
{
  { {a}{b}{c}{d}{e}{f} }
  { {A}{B}{C}{D}{E}{F} }
  { {k}{l}{m}{n}{o}{p} }
  { {K}{L}{M}{N}{O}{P} }
}
\ExplSyntaxOff
```

(a)(b)(c)(d)(e)(f)
(a+A)(b+B)(c+C)(d+D)(e+E)(f+F)

(a+A+k)(b+B+l)(c+C+m)(d+D+n)(e+E+o)(f+F+p)
(a+A+k+K)(b+B+l+L)(c+C+m+M)(d+D+n+N)(e+E+o+O)(f+F+p+P)

Listing 22.

```
\ExplSyntaxOn
\cs_set:Nn\__foo:nn { (#1+#2) }
\erw_tl_map_thread_at:Nnn \__foo:nn{ 2 }
{
  { {a}{b}{c}{d}{e}{f} }
  { {A}{B}{C}{D}{E}{F} }
}
\ExplSyntaxOff
```

(b+B)

Part III
Other

1 Acknowledgment

This work has benefited from Q&A's from the LaTeXcommunity[3]. lambda originally appeared in [2].

2 Install

1) Compile `erw-l3.dtx` (under Unix, `$tex timestamp.dtx`)

2) Put the generated `erw-l3.sty` in the search path of the LaTeXengine

3 Support

This package is available from https://www.ctan.org/pkg/erw-l3 and https://github.com/rogard/erw-l3.

3.1 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
```

3.2 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)
```

3.3 Results

1) erw-l3 v2.0 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

References

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

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

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

Change History

v1.1

General: \numbrdcsnew changed to \newnumbrdcs and made 'disambiguable' 14

disambig/backend: changes to the key, added \ProcessPackageKeysOption; . . . 14

Brought all the modules under one file; renamed l3erw to erw-l3; 14

v1.2

General: disambig: \disambignewcmd no longer takes a token name as arg, rather a token. 14

disambig: pushed the code inside \keys_define; 14

Add: \erw_items_to 14

Add: \erw_last_item 14

Add: \erw_repeat 14

Add: \erw_split 14

Add: \map_thread 14

Front end cmds no longer generated with module disambig; Option of the same name deleted; 14

Modify: \erw_compose, order in which functions composed ($g \circ f$ means f comes before g) 14

Rearrange: the doc to clearly separate frontend from backend . . 14

v1.3

General: Replace: versioning, should have been 0.1.2 14

v1.4

General: Add: \erw_accum 14

Add: \erw_int_range 14

Add: \erw_is_matrix (to check arg of \erw_tl_map_thread:Nn) 14

Add: \erw_merge 14

Add: \erw_set_map_inline 14

Add: \erw_set_map 14

Remove: \erw_items_to (redundant with \tl_range:nnn) . 14

v1.5

General: Modify: source repository . . 14

Rearrange: frontend/backend sections 14

Remove: disambig 14

Split Section Preliminaries into Conventions and Requirement. . . 14

v1.6

General: Fix: critical bug preventing erw-l3 from working without explicit inclusion of expl3 14

v1.7

General: Add: option 14

Add: sys 14

Move: \erw_fold_apply_par:n . . 14

Move: \erw_fold_set_par:n 14

Rearrange: structure of implementation, e.g. section 10 . . 14

Remove: document level functions,\numbrdcsnew, \numbrdcs 14

Replace: listing's implem with that of tocloft 14

Replace: vers. numb. from 3 to 2 digits 14

v1.8

General: Add: function for all frontend functions. 14

Remove: \erw_cs_set_eq:NN and variants 14

Remove: \erw_is_matrix:n (predicate must be expandable) . . 14

Rename: all cs prefixes to agree with heading under which they come, e.g. \erw_identity:n by \erw_cs_identity:n 14

Replace: \@@_map:n by \@@_oper_function:n 14

Replace: \erw_seq_fold:NN by \erw_oper_fold_seq:NN and likewise for variants 14

v1.9

General: Add: \erw_sys_timestamp_delimiter: 14

Add: \erw_tl_join:nn and variants 14

Rename: \erw_append_arg:nn to \erw_tl_append_item:nn 14

Rename: \erw_oper_gset_function:N to \erw_tl_gset_function:N (and variants) 14

v2.0

General: Add: \erw_jobnametimestamp:nn and variants 14

Remove: \merge:nn (redundant with \erw_join:nn) 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.

erw internal commands:

G

I

K

M

O

P

Q

S

T

U

Part IV
Implementation

1 Opening

```
⟨*package⟩
⟨@@=erw⟩
%       \ExplSyntaxOn
```

2 cs

2.1 backend

```
\cs_new:Nn \__erw_cs_name:N
{
  \exp_last_unbraced:Nf \use_i:nnn {\cs_split_function:N #1}
}
```

2.2 frontend

```
\cs_set:Npn \erw_cs_identity:n #1{#1}
\cs_new:Nn \erw_cs_set_inline:Nn
{
  \cs_set:Npn #1 ##1{#2}
}
\cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}
\cs_new:Nn \erw_cs_gset_inline:Nn
{
  \cs_gset:Npn #1 ##1{#2}
}
\cs_generate_variant:Nn \erw_cs_gset_inline:Nn {cn}
\cs_new:Nn \erw_tl_join:nn{#1#2}
\cs_new:Nn \erw_tl_join:nnn{#1#2#3}
\cs_new:Nn \erw_tl_join:nnnn{#1#2#3#4}
\cs_new:Nn \erw_tl_join:nnnnn{#1#2#3#4#5}
\cs_new:Nn \erw_cs_error:n{\msg_error{__erw}{generic}{unary~function~not~allowed}}
\cs_new:Nn \erw_cs_error:nn{\msg_error{__erw}{generic}{binary~function~not~allowed}}
```

3 clist

3.1 backend

3.2 frontend

4 csint

4.1 backend

```
\int_new:N \g__erw_csint_int
\tl_set:Nn \g__erw_csint_name_tl {\erw_csint_name:n{\g__erw_csint_int}}
```

4.2 frontend

```
\cs_new:Nn \erw_csint:nn
```

```
{
  \use:c{__erw_csint_\int_to_alph:n{#1}:n}{#2}
}
\cs_new:Nn \erw_csint_name:n {__erw_csint_\int_to_alph:n{#1}:n}
\cs_new:Nn \erw_csint_names:nnn
{
  \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_name:n
}
\cs_new_protected:Nn \erw_csint_new:n
{
  \int_incr:N \g__erw_csint_int
  \erw_cs_set_inline:cn{\g__erw_csint_name_tl}
  {
    \token_if_cs:NTF
    {#1}
    {#1{##1}}
    {#1}
  }
}
\cs_new:Nn \erw_csint_names_braced:nnn
{
  \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_names_braced:n
  % TODO \tl_range_braced:nnn?
}
\cs_new:Nn \erw_csint_names_braced:n {{\erw_csint_name:n{#1}}}
\cs_new:Nn \erw_csint_names_braced:
{
  \erw_csint_names_braced:nnn{1}{1}{\g__erw_csint_int}
}
\cs_new_protected:Nn \erw_csint_reset:
{
  \int_zero:N \g__erw_csint_int
  \tl_set:Nn \__erw_csint_ext_tl{}%^^A TODO remove?
}
```

5 int

5.1 backend

```
\cs_set:Npn \__erw_int_range:nnn #1 #2 #3
{
  \int_compare:nNnTF
  {
    \int_eval:n{#2+1}
  }>{#3}
  {
    {#1}
  }
  {
    \__erw_int_range:nnn
    {
      \exp_args:Nx\erw_tl_append_item:nn{#1}
      {
        \int_eval:n{#2+1}
```

```
      }
    }
    {\int_eval:n{#2+1}}
    {#3}
  }
}
```

5.2 frontend

```
\cs_new:Nn \erw_int_range:nn
{
  \__erw_int_range:nnn {{#1}}{#1}{#2}
}
\cs_new:Nn \erw_int_range:n
{
  \__erw_int_range:nnn {}{0}{#1}
% ^^A Alt to:
% ^^A    \int_step_inline:nn {#1}{##1}
}
```

6 keys

6.1 frontend

```
\cs_new_protected:Nn\erw_keyval_parse:NNNn
{
  \cs_set_protected:Nn \__erw_keyval_function:n {#2 #1{##1}}
  \cs_set_protected:Nn \__erw_keyval_function:nn {#3 #1{##1}{##2}}
  \keyval_parse:NNn
  \__erw_keyval_function:n
  \__erw_keyval_function:nn
  {#4}
}
```

7 lambda

\erw_lambda:nnn

```
\cs_new_protected:Npn \erw_lambda:nnn #1 #2 #3
{
  \exp_args:NNx
  #1 \__erw_lambda_expression
  {#2}
  {#3}
  \__erw_lambda_expression
}
```

(*End definition for* `\erw_lambda:nnn`. *This function is documented on page* 5.)

8 msg

8.1 backend

```
\msg_new:nnn{__erw}{generic}{#1}
\msg_new:nnn{__erw}{separ}{#1~expects~1~to~3~items,~#2}
```

```
\msg_new:nnn{__erw}{timestamp / base}{Calling~#1,~arg~must~be~'dec|hex'}
\msg_new:nnn{__erw}{timestamp / period}{Calling~#1,~arg~must~be~'date|time|datetime'}
```

8.2 frontend

```
\msg_new:nnn{erw}{csnset}{#1~not~set}
\msg_new:nnn{erw}{varnset}{#1~not~set}
```

9 prop

9.1 backend

```
\cs_new_protected:Nn \__erw_prop_map_item:NNN
{
  \cs_set_protected:Nn \__erw_function:nn
  {
    #1 #2 {##1}{##2}
  }
  \prop_map_function:NN #3 \__erw_function:nn
}
```

9.2 frontend

```
\cs_new_protected:Nn \erw_prop_to_clist:Nn
{
  \cs_set:Nn \__erw_keyval_function:n {,\prop_item:Nn#1{##1}}
  \exp_args:Nf
  \tl_tail:n
  {
    \keyval_parse:NNn
    \__erw_keyval_function:n
    \erw_cs_error:nn
    {#2}
  }
}
\cs_generate_variant:Nn \erw_prop_to_clist:Nn { c }
\cs_new_protected:Nn \erw_prop_map_item:NNN
{
  \prop_if_exist:NTF #2
  {\__erw_prop_map_item:NNN #1#2#3}
  {
    \prop_new:N #2
    \erw_prop_map_item:NNN #1#2#3
  }
}
% ^^A\cs_new_protected:Nn \erw_prop_put:NN
% ^^A{
% ^^A  \cs_set:Nn \__erw_prop_append:nn
% ^^A  {
% ^^A    \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
% ^^A  }
% ^^A  \prop_map_function:NN #2 \__erw_prop_append:nn
% ^^A}
% ^^A\cs_generate_variant:Nn \erw_prop_put:NN { cc }
% ^^A\cs_new_protected:Nn\erw_prop_put:Nnn
% ^^A{
```

```
% ^^A  \prop_if_exist:NTF#1
% ^^A  {
% ^^A    \prop_put:Nnn #1 {#2}{#3}
% ^^A  }
% ^^A  {
% ^^A    \prop_new:N #1
% ^^A    \erw_prop_put:Nnn #1{#2}{#3}
% ^^A  }
% ^^A}
% ^^A\cs_generate_variant:Nn \erw_prop_put:Nnn { c }
% ^^A\cs_new_protected:Nn\erw_prop_keyval_parse_key:Nnn
% ^^A{
% ^^A  \__erw_prop_keyval_parse:NNNn
% ^^A  #1
% ^^A  #2
% ^^A  \erw_keyval_keyonly:nn
% ^^A  {#3}
% ^^A}
% ^^A\cs_new_protected:Nn \erw_prop_keyval_parse:NNn
% ^^A{
% ^^A  \__erw_prop_keyval_parse:NNNn
% ^^A  #1
% ^^A  \erw_keyval_error:n
% ^^A  #2
% ^^A  {#3}
% ^^A}
\cs_new_protected:Nn\erw_prop_keyval_parse:NNNn
{
  \prop_if_exist:NTF#1
  {\erw_keyval_parse:NNNn #1#2#3{#4}}
  {
    \prop_new:N #1
    \erw_prop_keyval_parse:NNNn#1#2#3{#4}
  }
}
% ^^A  \cs_set_protected:Nn \__erw_keyval_function:n {#2 #1{##1}}
% ^^A  \cs_set_protected:Nn \__erw_keyval_function:nn {#3 #1{##1}{##2}}
% ^^A  \prop_if_exist:NTF#1
% ^^A  {
% ^^A    \keyval_parse:NNn
% ^^A    \__erw_keyval_function:n
% ^^A    \__erw_keyval_function:nn
% ^^A    {#4}
% ^^A  }
% ^^A  {
% ^^A    \prop_new:N #1
% ^^A    \__erw_prop_fun_keyval:NNNn #1#2#3{#4}
% ^^A  }
% ^^A}
% ^^A%^^A\cs_new_protected:Nn\erw_prop_gput_keyval:Nn
% ^^A{
% ^^A  \prop_if_exist:NTF#1
% ^^A  {
% ^^A    \erw_prop_keyval_parse:NNNn
```

```
% ^^A    #1
% ^^A    \erw_keyval_error:n
% ^^A    \prop_gput:Nnn
% ^^A    {#2}
% ^^A  }
% ^^A}
% ^^A\cs_generate_variant:Nn \erw_prop_put_keyval:Nn { c }
```

10 oper

10.1 backend

```
\cs_new:Nn \__erw_oper_compose:NnN
{
  \erw_cs_set_inline:Nn \g__erw_tl_function:n
  {
    #1{##1}#3
  }
  \exp_args:Nf\erw_tl_map:n
  {
    \tl_reverse:n{#2}
  }
}
```

10.2 frontend

```
\keys_define:nn{__erw}
{
  oper/fold_set_par.tl_gset:N = \g__erw_oper_fold_set_par_tl,
  oper/fold_set_par.value_required:n = true,
  oper/fold_set_par.default:n = {Nf},
  oper/fold_set_par.initial:n = {Nf},
  oper/fold_apply_par.tl_gset:N = \g__erw_oper_fold_apply_par_tl,
  oper/fold_apply_par.value_required:n = true,
  oper/fold_apply_par.default:n = {Nf},
  oper/fold_apply_par.initial:n = {Nf}
}
```

11 seq

11.1 backend

```
\tl_new:N \g__erw_seq_fold_item_tl
\cs_new_protected:Nn\__erw_seq_put_right_clist:Nn
{
  \cs_set_protected:Nn \__erw_function:n
  {
    \seq_put_right:Nn #1{##1}
  }
  \keyval_parse:NNn
  \__erw_function:n
  \erw_keyval_keyonly:nn
  {#2}
}
\cs_generate_variant:Nn \__erw_seq_put_right_clist:Nn { c }
```

```
\cs_new_protected:Nn\__erw_seq_put_right_prop:NNn
{
  \__erw_seq_put_right_clist:Nn #1
  {\erw_prop_to_clist:Nn #2 {#3}}
}
\cs_generate_variant:Nn \__erw_seq_put_right_prop:NNn { cc }
```

11.2 frontend

```
\cs_new:Nn \erw_seq_compose:nN
{
  \__erw_oper_compose:NnN \__erw_seq_fold:NN {#1} #2
}
\cs_new:Nn \erw_seq_compose_c:nN
{
  \__erw_oper_compose:NnN \__erw_seq_fold:cN {#1} #2
}
\cs_new:Nn \erw_seq_compose_vers:nN
{
  \msg_error:nnn{__erw}{csnset}{\erw_seq_compose_vers:nN}
}
\cs_new_protected:Nn\erw_seq_put_right_clist:Nn
{
  \seq_if_exist:NTF#1
  {\__erw_seq_put_right_clist:Nn#1{#2}}
  {\seq_new:N#1\erw_seq_put_right_clist:Nn#1{#2}}
}
\cs_generate_variant:Nn \erw_seq_put_right_clist:Nn { c }
\cs_new_protected:Nn\erw_seq_put_right_prop:NNn
{
  \seq_if_exist:NTF#1
  {\__erw_seq_put_right_prop:NNn#1#2{#3}}
  {\seq_new:N#1\erw_seq_put_right_prop:NNn#1#2{#3}}
}
\cs_generate_variant:Nn \erw_seq_put_right_prop:NNn { cc }
% ^^A\cs_new_protected:Nn\erw_seq_put_right:Nn
% ^^A{
% ^^A  \seq_if_exist:NTF#1
% ^^A  {\seq_put_right:Nn#1{#2}}
% ^^A  {\seq_new:N#1\erw_seq_put_right:Nn #1{#2}}
% ^^A}
% ^^A\cs_generate_variant:Nn\erw_seq_put_right:Nn { c }
\cs_new:Nn \__erw_seq_fold:NN
{
  \seq_get_right:NN #2 \g__erw_seq_fold_item_tl
  \erw_tl_fold:NN #1 \g__erw_seq_fold_item_tl
  \seq_put_right:No #2 {\g__erw_seq_fold_item_tl}
}
\cs_generate_variant:Nn \__erw_seq_fold:NN {cN}
\cs_new:Nn \erw_seq_use:Nn
{
  \exp_last_unbraced:NNf
  \seq_use:Nnnn #1
  \erw_tl_separators:n{#2}
}
```

12 sys

12.1 backend

__erw_sys_date:N
__erw_sys_date_dec:
__erw_sys_date_hex:

```
\cs_new:Nn \__erw_sys_date_dec:
{
  \int_eval:n
  {
    \c_sys_year_int * 10000
    +\c_sys_month_int * 100
    +\c_sys_day_int *  1
  }
}
\cs_new:Nn \__erw_sys_date:N{\int_to_hex:n{\__erw_sys_date_dec:}}
\cs_new:Nn \__erw_sys_date_hex:{\int_to_hex:n{\__erw_sys_date_dec:}}
```

(*End definition for* `\__erw_sys_date:N`, `\__erw_sys_date_dec:`, *and* `\__erw_sys_date_hex:`.)

__erw_sys_time_dec:
__erw_sys_time_hex

```
\cs_new:Nn \__erw_sys_time_dec:
{
  \int_eval:n
  {
    \c_sys_hour_int * 100
    +\c_sys_minute_int * 1
  }
}
\cs_new:Nn\__erw_sys_time_hex:{\int_to_hex:n{\__erw_sys_time_dec:}}
```

(*End definition for* `\__erw_sys_time_dec:` *and* `\__erw_sys_time_hex`.)

__erw_sys_datetime_base:n
__erw_sys_datetime_dec:n
__erw_sys_datetime_join:nn
__erw_sys_datetime_hex:n
__erw_sys_datetime_period:n

```
\cs_new:Nn\__erw_sys_datetime_base:n
{
  \int_case:nnTF{#1}
  {
    {10}{dec}
    {16}{hex}
  }
  {\c_empty_tl}
  {\msg_error:nnn{__erw}{timestamp / base}{\__erw_sys_datetime_base:n{#1}}}
}
\cs_new:Nn\__erw_sys_datetime_join:nn{\erw_tl_join:nnn{#1}{\g__erw_sys_timestamp_delim_str}{#
\cs_new:Nn\__erw_sys_datetime_period:n
{
  \str_case:nnTF{#1}
  {
    {date}{date}
    {time}{time}
    {datetime}{datetime}
  }
  {\c_empty_tl}
  {\msg_error:nnn{__erw}{ timestamp / period }{\__erw_sys_datetime_period:n{#1}}}
```

```
}
\cs_new:Nn\__erw_sys_datetime_dec: {\__erw_sys_datetime_join:nn{\__erw_sys_date_dec:}{\__erw_
\cs_new:Nn\__erw_sys_datetime_hex: {\__erw_sys_datetime_join:nn{\__erw_sys_date_hex:}{\__erw_
```

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

`\__erw_sys_jobnametimestamp_prefix:`

```
\cs_new:Nn\__erw_sys_jobnametimestamp_prefix:
{
  \erw_tl_join:nn
  {\c_sys_jobname_str}
  {\g__erw_sys_timestamp_delim_str}
}
```

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

`\__erw_sys_jobnametimestamp:n`
`\__erw_sys_jobnametimestamp:`

```
\cs_new:Nn\__erw_sys_jobnametimestamp:nn
{
  \erw_tl_join:nn
  {\__erw_sys_jobnametimestamp_prefix:}
  {\erw_sys_timestamp:nn{#1}{#2}}
}
\cs_new:Nn\__erw_sys_jobnametimestamp:
{
  \erw_tl_join:nn
  {\__erw_sys_jobnametimestamp_prefix:}
  {\erw_sys_timestamp:}
}
```

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

`\__erw_sys_timestamp:nn`

```
\cs_new:Nn\__erw_sys_timestamp:nn
{
  \exp_args:No
  \use:c{__erw_sys_\__erw_sys_datetime_period:n{#1}_\__erw_sys_datetime_base:n{#2}:}
}
\cs_new_protected:Nn \__erw_sys_set_delim:nn
{
  \use:c{tl_gset:N#1}
  \g__erw_sys_timestamp_delim_str{#2}
}
```

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

```
\keys_define:nn{__erw}
{
  sys / timestamp_delim .code:n =
  {
    \exp_last_unbraced:No
    \__erw_sys_set_delim:nn{n}{#1}
  },
  sys / timestamp_delim  .value_required:n = true,
  sys / timestamp_delim  .default:n = {-},
  sys / timestamp_delim  .initial:n = {-}
```

```
}
% \subsection{frontend}
%    \begin{macrocode}
\cs_new:Nn\erw_sys_jobnametimestamp:nn{\__erw_sys_jobnametimestamp:nn{#1}{#2}}
\cs_new:Nn\erw_sys_jobnametimestamp:{\__erw_sys_jobnametimestamp:}
\cs_new:Nn\erw_sys_timestamp_delimiter:
{
  \use:N \g__erw_sys_timestamp_delim_str
}
\cs_new:Nn\erw_sys_timestamp:nn
{
  \__erw_sys_timestamp:nn{#1}{#2}
}
\cs_new:Nn\erw_sys_timestamp:
{
  \__erw_sys_timestamp:nn{datetime}{16}
}
```

13 tl

13.1 backend

```
\tl_new:N \g__erw_tl_compose_tl
```

\g__erw_tl_function:n

```
\cs_new_protected:Nn \g__erw_tl_function:n
{
  \msg_error:nnn
  {erw}
  {csnset}
  {\g__erw_tl_function:n}
}
```

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

__erw_map:nn

```
\cs_set_protected:Nn \__erw_map:nn
{
  \quark_if_recursion_tail_stop:n{#1}
  \g__erw_tl_function:n{#1}  \__erw_map:nn{#2}
}
```

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

__erw_tl_map_thread_at:Nnn
__erw_tl_map_thread_at:Nnnn
__erw_tl_map_thread_at:Nnnnn
__erw_tl_map_thread_at:Nnnnnn

```
\cs_set_protected:Nn \__erw_tl_map_thread_at:Nnn
{
  #1
  {\exp_args:Nf\tl_item:nn {#3} {#2} }
}
\cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnn
{
  #1
  {\exp_args:Nf\tl_item:nn {#3} {#2} }
```

```
  {\exp_args:Nf\tl_item:nn {#4} {#2} }
}
\cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnn
{
  #1
  {\exp_args:Nf\tl_item:nn {#3} {#2} }
  {\exp_args:Nf\tl_item:nn {#4} {#2} }
  {\exp_args:Nf\tl_item:nn {#5} {#2} }
}
\cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnnn
{
  #1
  {\exp_args:Nf\tl_item:nn {#3} {#2} }
  {\exp_args:Nf\tl_item:nn {#4} {#2} }
  {\exp_args:Nf\tl_item:nn {#5} {#2} }
  {\exp_args:Nf\tl_item:nn {#6} {#2} }
}
```

(*End definition for* `\__erw_tl_map_thread_at:Nnn` *and others.*)

`\__erw_tl_separators:nn`

#1 : ⟨ *int* ⟩
#2 : ⟨ *items* ⟩

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

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

13.2 frontend

```
\cs_new:Nn \erw_tl_append_item:nn
{
  {#1{#2}}
}
```

```
\cs_new:Nn \erw_tl_compose:nN
{
  \__erw_oper_compose:NnN \erw_tl_fold:NN {#1} #2
}
\cs_new:Nn \erw_tl_compose:nn
{
  \tl_set:Nn \g__erw_tl_compose_tl {#2}
  \erw_tl_compose:nN{#1}\g__erw_tl_compose_tl
  \g__erw_tl_compose_tl
}
\cs_new:Nn \erw_tl_compose_c:nN
{
  \__erw_oper_compose:NnN \erw_tl_fold:cN {#1} #2
}
\cs_new:Nn \erw_tl_compose_c:nn
{
  \tl_set:Nn \g__erw_tl_compose_tl {#2}
  \erw_tl_compose_c:nN{#1}\g__erw_tl_compose_tl
  \g__erw_tl_compose_tl
}
\cs_new:Nn \erw_tl_compose_vers:nN
{
  \msg_error:nnn{__erw}{csnset}{\erw_tl_compose_vers:nN}
}
\cs_new:Nn \erw_tl_compose_vers:nn
{
  \erw_csint_reset:{}
  \tl_map_function:nN{#1}\erw_csint_new:n
  \exp_last_unbraced:Nx
  \erw_tl_compose_c:nn
  {{\erw_csint_names_braced:{}}}
  {#2}
}
\cs_new:Nn \erw_tl_fold:NN
{
  \use:c{tl_set:\g__erw_oper_fold_set_par_tl}
  #2
  {
    \use:c{exp_args:\g__erw_oper_fold_apply_par_tl}{#1}{#2}
  }
}
\cs_generate_variant:Nn \erw_tl_fold:NN {cN}
\cs_new:Nn \erw_tl_gset_function:N
{
  \erw_cs_gset_eq:NN \g__erw_tl_function:n #1
}
\cs_new:Nn \erw_tl_gset_function:n
{
  \erw_cs_gset_inline:Nn \g__erw_tl_function:n {#1}
}
\cs_new:Nn \erw_tl_last_item:n
{
  \exp_args:Nof \tl_item:nn
  {#1}
```

```
  {
    \tl_count:n{#1}
  }
}
\cs_new_protected:Nn \erw_tl_map:n
{
  \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
}
\cs_new_protected:Nn \erw_tl_map:Nn
{
  \cs_set_eq:NN \g__erw_tl_function:n #1
  \erw_tl_map:n{#2}
}
\cs_new_protected:Nn \erw_tl_map_inline:nn
{
  \erw_cs_set_inline:Nn \g__erw_tl_function:n {#1}
  \erw_tl_map:n{#2}
}
\cs_new:Nn \erw_tl_repeat:nn
{
  \int_step_inline:nnnn{1}{1}{#1}{#2}
}
\cs_new:Nn \erw_tl_split:nnn
{
  \tl_head:n{#1}
  \use:c{exp_args:#3} \tl_map_inline:nn
  {
    \tl_tail:n
    {
      #1
    }
  }{#2##1}
}
\cs_new:Nn \erw_tl_split:nn
{
  \erw_tl_split:nnn{#1}{#2}{Nf}
}
\cs_new_protected:Nn \erw_tl_map_thread_at:Nnn
{
  \exp_args:Nf\int_case:nnTF
  {
    \tl_count:n{#3}
  }
  {
    {1}{ \__erw_tl_map_thread_at:Nnn #1{#2}#3 }
    {2}{ \__erw_tl_map_thread_at:Nnnn #1{#2}#3 }
    {3}{ \__erw_tl_map_thread_at:Nnnnn #1{#2}#3 }
    {4}{ \__erw_tl_map_thread_at:Nnnnnn #1{#2}#3 }
  }
  {
    % Do nothing
  }
  {
    \msg_error:nnn{__erw}
```

```
    {generic}
    {erw_tl_map_thread_at:~count~of~#3~not~within~1~to~4}
  }
}
\cs_new_protected:Nn \erw_tl_map_thread:Nn
{
  \int_step_inline:nn
  {
    \exp_args:Nf \tl_count:n{ \tl_head:n{#2} }
  }
  {
    \erw_tl_map_thread_at:Nnn #1 {##1} {#2}
  }
}
\cs_new:Nn \erw_tl_separators:n
{
  \__erw_tl_separators:en{ \tl_count:n{#1} }{#1}
}
```

14 option

```
\cs_new_protected:Nn\erw_option:n
{
  \keys_set:nn{__erw}{#1}
}
```

15 Closing

```
\ExplSyntaxOff
⟨/package⟩
```