

The `erw-l3` package ^{*}

Erwann Rogard[†]

Released 2020/05/22

Abstract

Utilities like `expl3[1]`.

Résumé

Utilitaires de type `expl3[1]`.

Contents

I	Usage	4
1	Loading the package	4
2	basics	4
3	csint	4
4	int	5
5	lambda	5
6	option	5
7	prop	5
8	seq	6
9	sys	6
10	tl	7
II	Listing	9
1	constants	9
1.	9

^{*}This file describes version v2.5, last revised 2020/05/22.

[†]firstname dot lastname AusTria gmail dot com

2	basics	9
2.	9
3	csint	9
3.	9
4	int	10
4.	10
5	lambda	10
5.	10
6	prop	10
6.	10
7.	10
8.	11
7	seq	11
9.	11
10.	11
11.	12
12.	12
8	sys	13
13.	13
14.	13
9	tl	13
15.	13
16.	14
18.	14
19.	15
20.	15
21.	15
22.	15
23.	16
III	Other	17
1	Acknowledgment	17
2	Install	17
3	Support	17
3.1	Platform	17
3.2	Engine	17
3.3	Results	17
4	References	17

Change History	18
Index	19
IV Implementation	22
1 Opening	22
2 basics	22
2.1 backend	22
2.2 frontend	22
3 clist	23
3.1 backend	23
3.2 frontend	23
4 csint	23
4.1 backend	23
4.2 frontend	23
5 int	23
5.1 backend	23
5.2 frontend	24
6 keyval	24
7 lambda	24
8 msg	25
8.1 backend	25
8.2 frontend	25
9 prop	25
9.1 backend	25
9.2 frontend	25
10 oper	26
10.1 backend	26
10.2 frontend	26
11 seq	27
11.1 backend	27
11.2 frontend	27
12 sys	28
12.1 backend	28
13 tl	30
13.1 backend	30
13.2 frontend	32

14	option	34
15	Closing	34

Part I

Usage

<code>\usepackage</code>	<code>\usepackage{erw-l3}</code>
--------------------------	----------------------------------

Requirement

1. `erw-l3.sty` and its dependencies are in the path of the L^AT_EX engine. See [Part III, section 3](#).
2. Goes in the *preamble*

2 basics

<code>\erw_cs_apply:Nn</code>	<code>\erw_cs_apply:Nn {<cs>}{<token list₁</code>
<code>\erw_cs_apply:(No Nf Nx cn)</code>	
<code>\erw_cs_apply:Nnn</code>	
<code>\erw_cs_apply:Nnnn</code>	
<code>\erw_cs_apply:Nnnnn</code>	

<code>\erw_cs_identity:n</code>	<code>\erw_cs_identity:n{<arg>}</code>
---------------------------------	----------------------------------------------

<code>\erw_cs_set_inline:Nn</code>	<code>\erw_cs_set_inline:Nn{<cs>}{<code>}</code>
<code>\erw_cs_set_inline:cn</code>	

3 csint

<code>\erw_csint:nn</code>	<code>\erw_csint:nn{<integer>}{<arg>}</code>
----------------------------	----------------------------------------------------------

<code>\erw_csint_name:n</code>	<code>\erw_csint_name:n{<integer>}</code>
--------------------------------	-------------------------------------------------

<code>\erw_csint_names:nnn</code>	<code>\erw_csint_names:nnn{<integer>}{<integer>}{<integer>}</code>
-----------------------------------	--------------------------------------------------------------------------------------

<code>\erw_csint_names_braced:</code>	
<code>\erw_csint_names_braced:n</code>	
<code>\erw_csint_names_braced:nnn</code>	

<code>\erw_csint_new:n</code>	<code>\erw_csint_new:n{<integer>}</code>
-------------------------------	------------------------------------------------

<code>\erw_csint_reset:</code>	<code>\erw_csint_reset:</code>
--------------------------------	--------------------------------

4 int

<code>\erw_int_range:n</code>	<code>\erw_int_range:n{<integer>}</code>
<code>\erw_int_range:nn</code>	

5 lambda

<code>\erw_lambda:nnn</code>	<code>\erw_lambda:nnn<token>{<arg spec>}{<code>}</code>
------------------------------	---------------------------------------------------------------------------

6 option

<code>\erw_option:n</code>	<code>\erw_option:n{<keyval list>}</code>
----------------------------	-------------------------------------------------

oper / fold_set_par
oper / fold_apply_par
sys / timestamp_delim

7 prop

All functions that modify a $\langle prop \rangle$ check it exists, if not make sure it does.

<code>\erw_prop_put:NN</code>	<code>\erw_prop_put:NN<prop₁><prop₂></code>
-------------------------------	-------------------------------------------------------------------------------

<code>\erw_prop_put:Nnn</code>	<code>\erw_prop_put:Nnn<prop>{<key>}{<val>}</code>
--------------------------------	----------------------------------------------------------------------

<code>\erw_prop_put_keyval:Nn</code>	<code>\erw_prop_put_keyval:Nn<prop>{<keyval list>}</code>
--------------------------------------	-----------------------------------------------------------------------

<code>\erw_prop_to_clist:Nn</code>	<code>\erw_prop_to_clist:Nn<prop>{<key₁>,...}</code>
------------------------------------	-----------------------------------------------------------------------------

8 seq

All functions that modify a $\langle seq \rangle$ check it exists, if not make sure it does.

<u>$\backslash erw_seq_compose:nN$</u>	$\backslash erw_seq_compose:nN\{\{\langle cs_1 \rangle\} \dots\} \langle seq \rangle$
<u>$\backslash erw_seq_compose_c:nN$</u>	$\backslash erw_seq_compose_c:nN\{\{\langle cs\ name_1 \rangle\} \dots\} \langle seq \rangle$
<u>$\backslash erw_seq_compose_vers:nN$</u>	$\backslash erw_seq_compose:nN\{\{\langle cs\ or\ code_1 \rangle\} \dots\} \langle seq \rangle$
<u>$\backslash erw_seq_from_clist:Nn$</u> <u>$\backslash erw_seq_from_clist:cn$</u>	$\backslash erw_seq_from_clist:Nn \langle seq \rangle \{\langle clist \rangle\}$
<u>$\backslash erw_seq_from_prop:NNn$</u>	$\backslash erw_seq_from_prop:NNn \langle seq \rangle \langle prop \rangle \{\langle keyval\ list \rangle\}$
<u>$\backslash erw_seq_put_right:Nn$</u>	$\backslash erw_seq_put_right:Nn \langle seq \rangle \{\langle token\ list \rangle\}$
<u>$\backslash erw_seq_use:Nn$</u>	$\backslash erw_seq_use:Nn \langle seq \rangle \{\langle items \rangle\}$

Also see [1, Section 8 of l3seq]

Semantics $\backslash seq_use:Nnnn \langle seq \rangle \backslash erw_tl_separators:n \{\langle items \rangle\}$

9 sys

<u>$\backslash erw_sys_jobnametimestamp:nn$</u> <u>$\backslash erw_sys_jobnametimestamp:$</u>	$\backslash erw_sys_jobnametimestamp:nn\{date time datetime\}\{10 16\}$
<u>$\backslash erw_sys_timestamp:nn$</u> <u>$\backslash erw_sys_timestamp:$</u>	$\backslash erw_sys_timestamp:nn\{date time datetime\}\{10 16\}$ Semantics Timestamp in base 10 or 16
<u>$\backslash erw_sys_timestamp_delimiter:$</u>	$\backslash erw_sys_timestamp_delimiter:$

10 tl

All functions that modify a $\langle token\ list \rangle$ check it exists, if not make sure it does.

<code>\erw_tl_append_item:nn</code>	<code>\erw_tl_append_item:nn{<arg\ list>}{<arg>}</code>
-------------------------------------	---------------------------------------------------------------------

<code>\erw_tl_compose:nN</code>	<code>\erw_tl_compose:nn{<cs_1>...}{<token\ list>}</code>
<code>\erw_tl_compose:nn</code>	

<code>\erw_tl_compose_c:nN</code>	<code>\erw_tl_compose_c:nn{<cs\ name_1>...}{<token\ list>}</code>
<code>\erw_tl_compose_c:nn</code>	

<code>\erw_tl_compose_vers:nN</code>	<code>\erw_tl_compose_vers:nn{<cs\ or\ code_1>...}{<token\ list>}</code>
<code>\erw_tl_compose_vers:nn</code>	

<code>\erw_tl_fold:NN</code>	<code>\erw_tl_fold:NN<cs><tl\ var></code>
<code>\erw_tl_fold:cN</code>	

<code>\erw_tl_gset_function:N</code>	<code>\erw_tl_gset_function:n{<code>}</code>
<code>\erw_tl_gset_function:n</code>	

<code>\erw_tl_join:nn</code>	<code>\erw_tl_join:nn{<token\ list_1>}{<token\ list_2>}</code>
<code>\erw_tl_join:nnn</code>	
<code>\erw_tl_join:nnnn</code>	
<code>\erw_tl_join:nnnnn</code>	

<code>\erw_tl_last_item:n</code>	<code>\erw_tl_last_item:n{<token\ list>}</code>
----------------------------------	-------------------------------------------------------

<code>\erw_tl_map:n</code>	<code>\erw_tl_map:n{<items>}</code>
<code>\erw_tl_map:Nn</code>	

Semantics Maps over $\langle items \rangle$ using the internal function set by `\erw_tl_gset_function:n`

<code>\erw_tl_map_inline:nn</code>	<code>\erw_tl_map_inline:nn{<code>}{<items>}</code>
------------------------------------	-----------------------------------------------------------------

<code>\erw_tl_map_thread:Nn</code>	<code>\erw_tl_math_thread:Nn<cs>{<items>}</code>
------------------------------------	--------------------------------------------------------------

<code>\erw_tl_map_thread_at:Nnn</code>	<code>\erw_tl_math_thread_at:Nnn{<integer>}{<token\ list>}</code>
----------------------------------------	-------------------------------------------------------------------------------

<code>\erw_tl_repeat:nn</code>	<code>\erw_tl_repeat:nn{<integer>}{<token\ list>}</code>
--------------------------------	----------------------------------------------------------------------

<code>\erw_tl_split:nnn</code>	<code>\erw_tl_split:nn{<items>}{<delimiter>}</code>
<code>\erw_tl_split:nn</code>	

<code>\erw_tl_separators:n</code>	<code>\erw_tl_separators:n{<items>}</code>
-----------------------------------	--------------------------------------------------

Semantics According to the count of *<items>*:

- 1) $\{\langle token\ list_1 \rangle\}\{\langle token\ list_1 \rangle\}\{\langle token\ list_1 \rangle\}$
- 2) $\{\langle token\ list_1 \rangle\}\{\langle token\ list_2 \rangle\}\{\langle token\ list_1\ token\ list_2 \rangle\}$
- 3) $\{\langle token\ list_1 \rangle\}\{\langle token\ list_2 \rangle\}\{\langle token\ list_3 \rangle\}$

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 basics

Listing 2.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { f(#1) }
\erw_cs_apply:Nn \__foo:n{ X }
\ExplSyntaxOff
```

f(X)

3 csint

Listing 3.

```
\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 4.

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

2345
12345

5 lambda

Listing 5.

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

Hello, world!

6 prop

Listing 6.

```
\ExplSyntaxOn
\erw_prop_put:Nnn \baz_prop { D } { d }
\erw_prop_put:NN \baz_prop \foo_prop
\prop_item:Nn \baz_prop{ A }
,\prop_item:Nn \baz_prop{ B }
,\prop_item:Nn \baz_prop{ C }
,\prop_item:Nn \baz_prop{ D }
\ExplSyntaxOff
```

a,b,c,d

Listing 7.

```
\ExplSyntaxOn
\erw_prop_put_keyval:Nn \foo_prop { 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 8.

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

a,b,c

7 seq

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_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 10.

```
\ExplSyntaxOn
\cs_set:Nn \__foo:n { f(#1) }
\cs_set:Nn \__bar:n { g[#1] }
\cs_set:Nn \__baz:n { h\{#1\} }
\erw_seq_put_right:Nn \l_tmp_seq{X}
\erw_seq_compose_c: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 11.

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

a,b,c

Listing 12.

```
\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 13.

```
\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
```

```
20200522-345
20200522-345
1343c4a%159
1343c4a%159
erw-l3%1343c4a%159
```

Listing 14.

```
\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:20200522345
D:20200522034515-04'00'
Hello, world!
```

9 tl

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: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 16.

```

\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:nN{ {\__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 17.

```

\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 18.

```

\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 19.

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

xxx

Listing 20.

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

a==b==c

Listing 21.

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

(a)(b)(c)

Listing 22.

```
\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 23.

```

\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 L^AT_EX community [3]. `lambda` originally appeared in [2].

2 Install

- 1) Compile `erw-13.dtx` (under Unix, `$tex timestamp.dtx`)
- 2) Put the generated `erw-13.sty` in the search path of the L^AT_EX engine

3 Support

This package is available from <https://www.ctan.org/pkg/erw-13> and <https://github.com/rogard/erw-13>.

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-13 v2.0 compiles satisfactorily on platform *i)* and engines *b)*, *c)*, and *d)*

References

- [1] 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>
- [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: <code>\numbrdcsnew</code> changed to <code>\newnumbrdcs</code> and made 'disambiguable' 17	v1.6	General: Fix: critical bug preventing <code>erw-l3</code> from working without explicit inclusion of <code>expl3</code> 17
	<code>disambig/backend</code> : changes to the key, added	v1.7	General: Add: <code>option</code> 17
	<code>\ProcessPackageKeysOption</code> ; . . . 17		Add: <code>sys</code> 17
	Brought all the modules under one file; renamed <code>l3erw</code> to <code>erw-l3</code> ; 17		Move: <code>\erw_fold_apply_par:n</code> . . 17
v1.2	General: 17		Move: <code>\erw_fold_set_par:n</code> 17
	<code>\erw_compose</code> reversed order in which the functions are composed, such that it now conforms to the mathematical convention ($g \circ f$ means f comes before g) 17		Rearrange: structure of implementation, e.g. section 10 . . 17
	<code>disambig</code> : pushed the code inside <code>\keys_define:\disambignewcmd</code> no longer takes a token name as arg, rather a token. 17		Remove: document level functions, <code>\numbrdcsnew</code> , <code>\numbrdcs</code> 17
	Add: <code>\erw_items_to</code> 17		Replace: listing's implem with that of <code>tocloft</code> 17
	Add: <code>\erw_last_item</code> 17		Replace: vers. numb. from 3 to 2 digits 17
	Add: <code>\erw_repeat</code> 17	v1.8	General: Add: function for all frontend functions. 17
	Add: <code>\erw_split</code> 17		Remove: <code>\erw_cs_set_eq:NN</code> and variants 17
	Add: <code>\map_thread</code> 17		Remove: <code>\erw_is_matrix:n</code> (predicate must be expandable) . . 17
	Front end cmds no longer generated with module <code>disambig</code> ; Option of the same name deleted; 17		Rename: all cs prefixes to agree with heading under which they come, e.g. <code>\erw_identity:n</code> by <code>\erw_cs_identity:n</code> 17
	Re-arrange: the doc to clearly separate frontend from backend . . 17		Replace: <code>@@_map:n</code> by <code>@@_oper_function:n</code> 17
v1.3	General: Replace: versioning, should have been 0.1.2 17		Replace: <code>\erw_seq_fold:NN</code> by <code>\erw_oper_fold_seq:NN</code> and likewise for variants 17
v1.4	General: Add: <code>\erw_accum</code> 17	v1.9	General: Add:
	Add: <code>\erw_int_range</code> 17		<code>\erw_sys_timestamp_delimiter:</code> . 17
	Add: <code>\erw_is_matrix</code> (to check arg of <code>\erw_tl_map_thread:Nn</code>) 17		Add: <code>\erw_tl_join:nn</code> and variants 17
	Add: <code>\erw_merge</code> 17		Rename: <code>\erw_append_arg:nn</code> to <code>\erw_tl_append_item:nn</code> 17
	Add: <code>\erw_set_map_inline</code> 17		Rename:
	Add: <code>\erw_set_map</code> 17		<code>\erw_oper_gset_function:N</code> to <code>\erw_tl_gset_function:N</code> (and variants) 17
	Remove: <code>\erw_items_to</code> (redundant with <code>\tl_range:nnn</code>) . 17	v2.0	General: Add:
v1.5	General: Modify: source repository . . 17		<code>\erw_jobnametimestamp:nn</code> and variants 17
	Rearrange: frontend/backend sections 17		Remove: <code>\merge:nn</code> (redundant with <code>\erw_join:nn</code>) 17
	Remove: <code>disambig</code> 17		
	Split Section Preliminaries into Conventions and Requirement. . . 17		

	Rename: v0.0 to v1.0, etc. 17		Add: \erw_tl_separators:n 17
v2.1	General: Add: \erw_prop_to_clist:Nn, \erw_prop_put:NN, and \erw_prop_put:Nnn 17 Add: \erw_seq_from_clist:Nn, \erw_seq_from_prop:Nnn, and \erw_seq_put_right:Nn 17 Move: all functions under section 10 to section 13 or section 11 , except \@@_oper_compose:NnN 17 Replace: \erw_seq_fold:NN by _erw_seq_fold:NN 17	v2.3	General: Add: \msg_new:nnn, module erw, messages: csnset 17 Add: \msg_new:nnn, module erw, messages: keyval/... 17 Fix: 'mark as private code' (hiherto unnoticed) 17 Modify: behavior of \erw_seq_use:Nn 17 Move: all \msg_new:Nnnn statements under same heading .. 17
v2.2	General: Add: \erw_seq_use:Nn 17	v2.4	General: Add: \erw_lambda:nnn 17
		v2.5	General: Add: \erw_prop_keyval:Nn 17

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			
@@ commands:		\cs_set:Npn 25, 28, 77	
\@@_map:n 18		\cs_set_eq:NN 499	
\@@_oper_compose:NnN 19		\cs_set_protected:Nn 208, 379, 384, 389, 395, 402	
\@@_oper_function:n 18		\cs_split_function:N 6	
B		D	
\begin 320, 321, 356		\disambignewcmd 18	
C		E	
cs commands:		erw commands:	
\cs_generate_variant:Nn 12, 30, 35, 144, 154, 166, 182, 217, 223, 242, 249, 256, 263, 432, 476		\erw_accum 18	
\cs_gset:Npn 33		\erw_append_arg:nn 18	
\cs_new:Nn 4, 8, 13, 17, 21, 26, 31, 36, 37, 38, 39, 42, 46, 47, 62, 67, 68, 98, 102, 108, 112, 183, 224, 228, 232, 257, 264, 270, 279, 280, 281, 289, 290, 300, 301, 312, 313, 314, 322, 328, 334, 357, 358, 359, 363, 367, 410, 433, 437, 441, 447, 451, 457, 461, 470, 477, 481, 485, 507, 511, 522, 557		\erw_compose 18	
\cs_new_protected:Nn 51, 72, 132, 146, 155, 167, 206, 218, 236, 243, 250, 339, 372, 493, 497, 502, 526, 547, 561		\erw_cs_apply:Nn 4, 8, 12, 44, 386	
\cs_new_protected:Npn 116		\erw_cs_apply:Nnn 4, 13, 391	
\cs_set:Nn 134, 148, 169		\erw_cs_apply:Nnnn 4, 17, 397	
		\erw_cs_apply:Nnnnn 4, 21, 404	
		\erw_cs_gset_eq:NN 479	
		\erw_cs_gset_inline:Nn ... 31, 35, 483	
		\erw_cs_identity:n 4, 18, 25	
		\erw_cs_set_eq:NN 18	
		\erw_cs_set_inline:Nn 4, 26, 30, 54, 185, 504	
		\erw_csint:nn 4, 42	
		\erw_csint_name:n ... 4, 41, 46, 49, 67	
		\erw_csint_names:nnn 4, 47	
		\erw_csint_names_braced: .. 4, 68, 467	
		\erw_csint_names_braced:n .. 4, 64, 67	
		\erw_csint_names_braced:nnn 4, 62, 70	

<code>\erw_csint_new:n</code>	5, 51, 464
<code>\erw_csint_reset:</code>	5, 72, 463
<code>\erw_fold_apply_par:n</code>	18
<code>\erw_fold_set_par:n</code>	18
<code>\erw_identity:n</code>	18
<code>\erw_int_range</code>	18
<code>\erw_int_range:n</code>	5, 102
<code>\erw_int_range:nn</code>	5, 98
<code>\erw_is_matrix</code>	18
<code>\erw_is_matrix:n</code>	18
<code>\erw_items_to</code>	18
<code>\erw_jobnametimestamp:nn</code>	18
<code>\erw_join:nn</code>	18
<code>\erw_keyval_ignorekey:n</code>	173
<code>\erw_keyval_keyonly:nn</code> ..	112, 140, 214
<code>\erw_keyval_keyval:n</code>	108
<code>\erw_lambda:nnn</code>	5, 19, 116
<code>\erw_last_item</code>	18
<code>\erw_merge</code>	18
<code>\erw_oper_fold_seq:NN</code>	18
<code>\erw_oper_gset_function:N</code>	18
<code>\erw_option:n</code>	5, 561
<code>\erw_prop_keyval:Nn</code>	19
<code>\erw_prop_put:NN</code>	5, 19, 146, 154
<code>\erw_prop_put:Nn</code>	179
<code>\erw_prop_put:Nnn</code> ..	5, 19, 155, 163, 166
<code>\erw_prop_put_keyval:Nn</code> ..	5, 167, 182
<code>\erw_prop_to_clist:Nn</code>	5, 19, 132, 144, 221
<code>\erw_repeat</code>	18
<code>\erw_seq_compose:nN</code>	6, 6, 224
<code>\erw_seq_compose_c:nN</code>	6, 228
<code>\erw_seq_compose_vers:nN</code> ..	6, 232, 234
<code>\erw_seq_fold:NN</code>	18, 19
<code>\erw_seq_from_clist:Nn</code>	6, 19, 236, 240, 242
<code>\erw_seq_from_prop:Nnn</code>	6, 19, 243, 247, 249
<code>\erw_seq_put_right:Nn</code>	6, 19, 250, 254, 256
<code>\erw_seq_use:Nn</code>	6, 19, 264
<code>\erw_set_map</code>	18
<code>\erw_set_map_inline</code>	18
<code>\erw_split</code>	18
<code>\erw_sys_jobnametimestamp:</code> ..	6, 358
<code>\erw_sys_jobnametimestamp:nn</code>	6, 357
<code>\erw_sys_timestamp:</code>	6, 332, 367
<code>\erw_sys_timestamp:nn</code> ..	6, 326, 363
<code>\erw_sys_timestamp_delimiter:</code> ..	6, 18, 359
<code>\erw_tl_append_item:nn</code> ..	7, 18, 89, 433
<code>\erw_tl_compose:nN</code>	7, 437, 444
<code>\erw_tl_compose:nn</code>	7, 441
<code>\erw_tl_compose_c:nN</code>	7, 447, 454
<code>\erw_tl_compose_c:nn</code>	7, 451, 466
<code>\erw_tl_compose_vers:nN</code> ..	7, 457, 459
<code>\erw_tl_compose_vers:nn</code>	7, 461
<code>\erw_tl_fold:NN</code>	7, 260, 439, 449, 470, 476
<code>\erw_tl_gset_function:N</code> ..	7, 18, 477
<code>\erw_tl_gset_function:n</code> ..	7, 7, 481
<code>\erw_tl_join:nn</code> ..	7, 18, 36, 316, 324, 330
<code>\erw_tl_join:nnn</code>	7, 37, 300
<code>\erw_tl_join:nnnn</code>	7, 38
<code>\erw_tl_join:nnnnn</code>	7, 39
<code>\erw_tl_last_item:n</code>	7, 485
<code>\erw_tl_map:n</code> ..	7, 189, 493, 500, 505
<code>\erw_tl_map:Nn</code>	7, 497
<code>\erw_tl_map_inline:nn</code>	7, 502
<code>\erw_tl_map_thread:Nn</code>	7, 18, 547
<code>\erw_tl_map_thread_at:Nnn</code>	7, 526, 554
<code>\erw_tl_math_thread:Nn</code>	7
<code>\erw_tl_math_thread_at:Nnn</code>	7
<code>\erw_tl_repeat:nn</code>	7, 507
<code>\erw_tl_separators:n</code> ..	6, 8, 19, 268, 557
<code>\erw_tl_split:nn</code>	8, 522
<code>\erw_tl_split:nnn</code>	8, 511, 524
erw internal commands:	
<code>__erw_cs_name:N</code>	4
<code>__erw_csint_ext_tl</code>	75
<code>\g_erw_csint_int</code> ..	40, 41, 53, 70, 74
<code>\g_erw_csint_name_tl</code>	41, 54
<code>__erw_function:n</code>	208, 213
<code>__erw_int_range:nnn</code> ..	77, 87, 100, 104
<code>__erw_keyval_function:n</code> ..	134, 139
<code>__erw_keyval_function:nn</code> ..	169, 174
<code>__erw_lambda_expression</code> ..	119, 122
<code>__erw_map:nn</code>	379, 495
<code>__erw_oper_compose:NnN</code>	183, 226, 230, 439, 449
<code>\g_erw_oper_fold_apply_par_tl</code> ..	200, 474
<code>\g_erw_oper_fold_set_par_tl</code>	196, 472
<code>__erw_prop_append:nn</code>	148, 152
<code>__erw_seq_fold:NN</code>	19, 226, 230, 257, 263
<code>\g_erw_seq_fold_item_tl</code>	205, 259, 260, 261
<code>__erw_seq_set_from_clist:Nn</code> ..	206, 217, 220, 239
<code>__erw_seq_set_from_prop:Nnn</code> ..	218, 223, 246
<code>__erw_sys_date:N</code>	270
<code>__erw_sys_date_dec:</code>	270, 312
<code>__erw_sys_date_hex:</code>	270, 313
<code>__erw_sys_datetime_base:n</code> ..	290, 337
<code>__erw_sys_datetime_dec:</code>	312
<code>__erw_sys_datetime_dec:n</code>	290

Q

quark commands:

<code>\quark_if_recursion_tail_stop:n</code>	381
<code>\q_recursion_stop</code>	495
<code>\q_recursion_tail</code>	495

S

seq commands:

<code>\seq_get_right:NN</code>	259
<code>\seq_if_exist:NTF</code>	238, 245, 252
<code>\seq_new:N</code>	240, 247, 254
<code>\seq_put_right:Nn</code>	210, 253, 261
<code>\seq_use:Nnnn</code>	6, 267

str commands:

<code>\str_case:nnTF</code>	303
<code>\subsection</code>	355

sys / timestamp_delim (option) 5

sys commands:

<code>\c_sys_day_int</code>	276
<code>\c_sys_hour_int</code>	285
<code>\c_sys_jobname_str</code>	317
<code>\c_sys_minute_int</code>	286
<code>\c_sys_month_int</code>	275
<code>\c_sys_year_int</code>	274

T

tl commands:

<code>\c_empty_tl</code>	297, 309, 424
<code>\tl_count:n</code>	490, 530, 551, 559
<code>\tl_head:n</code>	513, 551
<code>\tl_item:nn</code>	387, 392, 393, 398, 399, 400, 405, 406, 407, 408, 487
<code>\tl_map_function:nN</code>	464
<code>\tl_map_inline:nn</code>	514
<code>\tl_new:N</code>	205, 371
<code>\tl_range:nnn</code>	18
<code>\tl_range_braced:nnn</code>	65
<code>\tl_reverse:n</code>	191
<code>\tl_set:Nn</code>	41, 75, 443, 453
<code>\tl_tail:n</code>	136, 516

token commands:

<code>\token_if_cs:NTF</code>	56
-------------------------------	----

U

use commands:

<code>\use:N</code>	337, 341, 361, 472, 474, 514
<code>\use_i:nn</code>	419, 420
<code>\use_i:nnn</code>	6
<code>\use_ii:nn</code>	418, 420
<code>\usepackage</code>	4

Part IV

Implementation

1 Opening

```

1 <*package>
2 <@@=erw>
3 % \ExplSyntaxOn

```

2 basics

2.1 backend

```

4 \cs_new:Nn \__erw_cs_name:N
5 {
6   \exp_last_unbraced:Nf \use_i:nnn {\cs_split_function:N #1}
7 }

```

2.2 frontend

```

8 \cs_new:Nn \erw_cs_apply:Nn
9 {
10   #1{#2}
11 }
12 \cs_generate_variant:Nn \erw_cs_apply:Nn {No, Nf, Nx, c}
13 \cs_new:Nn \erw_cs_apply:Nnn
14 {
15   #1{#2}{#3}
16 }
17 \cs_new:Nn \erw_cs_apply:Nnnn
18 {
19   #1{#2}{#3}{#4}
20 }
21 \cs_new:Nn \erw_cs_apply:Nnnnn
22 {
23   #1{#2}{#3}{#4}{#5}
24 }
25 \cs_set:Npn \erw_cs_identity:n #1{#1}
26 \cs_new:Nn \erw_cs_set_inline:Nn
27 {
28   \cs_set:Npn #1 ##1{#2}
29 }
30 \cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}
31 \cs_new:Nn \erw_cs_gset_inline:Nn
32 {
33   \cs_gset:Npn #1 ##1{#2}
34 }
35 \cs_generate_variant:Nn \erw_cs_gset_inline:Nn {cn}
36 \cs_new:Nn \erw_tl_join:nn{#1#2}
37 \cs_new:Nn \erw_tl_join:nnn{#1#2#3}
38 \cs_new:Nn \erw_tl_join:nnnn{#1#2#3#4}
39 \cs_new:Nn \erw_tl_join:nnnnn{#1#2#3#4#5}

```

3 clist

3.1 backend

3.2 frontend

4 csint

4.1 backend

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

4.2 frontend

```
42 \cs_new:Nn \erw_csint:nn
43 {
44   \erw_cs_apply:cn{\__erw_csint_\int_to_alph:n{#1}:n}{#2}
45 }
46 \cs_new:Nn \erw_csint_name:n {\__erw_csint_\int_to_alph:n{#1}:n}
47 \cs_new:Nn \erw_csint_names:nnn
48 {
49   \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_name:n
50 }
51 \cs_new_protected:Nn \erw_csint_new:n
52 {
53   \int_incr:N \g__erw_csint_int
54   \erw_cs_set_inline:cn{\g__erw_csint_name_tl}
55   {
56     \token_if_cs:NTF
57     {#1}
58     {#1{##1}}
59     {#1}
60   }
61 }
62 \cs_new:Nn \erw_csint_names_braced:nnn
63 {
64   \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_names_braced:n
65   % TODO \tl_range_braced:nnn?
66 }
67 \cs_new:Nn \erw_csint_names_braced:n {\erw_csint_name:n{#1}}
68 \cs_new:Nn \erw_csint_names_braced:
69 {
70   \erw_csint_names_braced:nnn{1}{1}{\g__erw_csint_int}
71 }
72 \cs_new_protected:Nn \erw_csint_reset:
73 {
74   \int_zero:N \g__erw_csint_int
75   \tl_set:Nn \__erw_csint_ext_tl{}%^A TODO remove?
76 }
```

5 int

5.1 backend


```

77 \cs_set:Npn \__erw_int_range:nnn #1 #2 #3
78 {
79   \int_compare:nNnTF
80   {
81     \int_eval:n{#2+1}
82   }>{#3}
83   {
84     {#1}
85   }
86   {
87     \__erw_int_range:nnn
88     {
89       \exp_args:Nx\erw_tl_append_item:nn{#1}
90       {
91         \int_eval:n{#2+1}
92       }
93     }
94     {\int_eval:n{#2+1}}
95     {#3}
96   }
97 }

```

5.2 frontend

```

98 \cs_new:Nn \erw_int_range:nn
99 {
100   \__erw_int_range:nnn {{#1}}{#1}{#2}
101 }
102 \cs_new:Nn \erw_int_range:n
103 {
104   \__erw_int_range:nnn {}{0}{#1}
105 % ^^A Alt to:
106 % ^^A   \int_step_inline:nn {#1}{##1}
107 }

```

6 keyval

```

108 \cs_new:Nn \erw_keyval_keyval:n
109 {
110   \msg_error:nnn{erw}{keyval/keyval}{#1}
111 }
112 \cs_new:Nn \erw_keyval_keyonly:nn
113 {
114   \msg_error:nnn{erw}{keyval/keyonly}{#1}{#2}
115 }

```

7 lambda

\erw_lambda:nnn

```

116 \cs_new_protected:Npn \erw_lambda:nnn #1 #2 #3
117 {
118   \exp_args:NNx
119   #1 \__erw_lambda_expression
120   {#2}
121   {#3}

```

```

122  \__erw_lambda_expression
123 }

```

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

8 msg

8.1 backend

```

124 \msg_new:nnn{\__erw}{generic}{#1}
125 \msg_new:nnn{\__erw}{separ}{#1~expects~1~to~3~items,~#2}
126 \msg_new:nnn{\__erw}{timestamp / base}{Calling~#1,~arg~must~be~'dec|hex'}
127 \msg_new:nnn{\__erw}{timestamp / period}{Calling~#1,~arg~must~be~'date|time|datetime'}

```

8.2 frontend

```

128 \msg_new:nnn{erw}{csnset}{#1~not~set}
129 \msg_new:nnn{erw}{keyval/keyval}{passed~key~#1~without~a~val}
130 \msg_new:nnn{erw}{keyval/keyonly}{passed~key~#1~val~#2~where~keyonly}
131 \msg_new:nnn{erw}{keyval/mandatval}{key~#1~has~no~matching~val}

```

9 prop

9.1 backend

9.2 frontend

```

132 \cs_new_protected:Nn \erw_prop_to_clist:Nn
133 {
134   \cs_set:Nn \__erw_keyval_function:n {,\prop_item:Nn#1{##1}}
135   \exp_args:Nf
136   \tl_tail:n
137   {
138     \keyval_parse:NNn
139     \__erw_keyval_function:n
140     \erw_keyval_keyonly:nn
141     {#2}
142   }
143 }
144 \cs_generate_variant:Nn \erw_prop_to_clist:Nn { c }
145
146 \cs_new_protected:Nn \erw_prop_put:NN
147 {
148   \cs_set:Nn \__erw_prop_append:nn
149   {
150     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
151   }
152   \prop_map_function:NN #2 \__erw_prop_append:nn
153 }
154 \cs_generate_variant:Nn \erw_prop_put:NN { cc }
155 \cs_new_protected:Nn \erw_prop_put:Nnn
156 {
157   \prop_if_exist:NTF#1
158   {

```

```

159   \prop_put:Nnn #1 {#2}{#3}
160 }
161 {
162   \prop_new:N #1
163   \erw_prop_put:Nnn #1{#2}{#3}
164 }
165 }
166 \cs_generate_variant:Nn \erw_prop_put:Nnn { c }
167 \cs_new_protected:Nn\erw_prop_put_keyval:Nn
168 {
169   \cs_set:Nn \__erw_keyval_function:nn {\prop_put:Nnn #1{##1}{##2}}
170   \prop_if_exist:NTF#1
171   {
172     \keyval_parse:NNn
173     \erw_keyval_ignorekey:n
174     \__erw_keyval_function:nn
175     {#2}
176   }
177   {
178     \prop_new:N #1
179     \erw_prop_put:Nn #1{#2}
180   }
181 }
182 \cs_generate_variant:Nn \erw_prop_put_keyval:Nn { c }

```

10 oper

10.1 backend

```

183 \cs_new:Nn \__erw_oper_compose:NnN
184 {
185   \erw_cs_set_inline:Nn \g__erw_tl_function:n
186   {
187     #1{##1}#3
188   }
189   \exp_args:Nf\erw_tl_map:n
190   {
191     \tl_reverse:n{#2}
192   }
193 }

```

10.2 frontend

```

194 \keys_define:nn{__erw}
195 {
196   oper/fold_set_par.tl_gset:N = \g__erw_oper_fold_set_par_tl,
197   oper/fold_set_par.value_required:n = true,
198   oper/fold_set_par.default:n = {Nf},
199   oper/fold_set_par.initial:n = {Nf},
200   oper/fold_apply_par.tl_gset:N = \g__erw_oper_fold_apply_par_tl,
201   oper/fold_apply_par.value_required:n = true,
202   oper/fold_apply_par.default:n = {Nf},
203   oper/fold_apply_par.initial:n = {Nf}
204 }

```

11 seq

11.1 backend

```
205 \tl_new:N \g__erw_seq_fold_item_tl
206 \cs_new_protected:Nn\__erw_seq_set_from_clist:Nn
207 {
208   \cs_set_protected:Nn \__erw_function:n
209   {
210     \seq_put_right:Nn #1{##1}
211   }
212   \keyval_parse:NNn
213   \__erw_function:n
214   \erw_keyval_keyonly:nn
215   {#2}
216 }
217 \cs_generate_variant:Nn \__erw_seq_set_from_clist:Nn { c }
218 \cs_new_protected:Nn\__erw_seq_set_from_prop:NNn
219 {
220   \__erw_seq_set_from_clist:Nn #1
221   {\erw_prop_to_clist:Nn #2 {#3}}
222 }
223 \cs_generate_variant:Nn \__erw_seq_set_from_prop:NNn { cc }
```

11.2 frontend

```
224 \cs_new:Nn \erw_seq_compose:nN
225 {
226   \__erw_oper_compose:NnN \__erw_seq_fold:NN {#1} #2
227 }
228 \cs_new:Nn \erw_seq_compose_c:nN
229 {
230   \__erw_oper_compose:NnN \__erw_seq_fold:cN {#1} #2
231 }
232 \cs_new:Nn \erw_seq_compose_vers:nN
233 {
234   \msg_error:nnn{__erw}{csnset}{\erw_seq_compose_vers:nN}
235 }
236 \cs_new_protected:Nn\erw_seq_from_clist:Nn
237 {
238   \seq_if_exist:NTF#1
239   {\__erw_seq_set_from_clist:Nn#1{#2}}
240   {\seq_new:N#1\erw_seq_from_clist:Nn#1{#2}}
241 }
242 \cs_generate_variant:Nn \erw_seq_from_clist:Nn { c }
243 \cs_new_protected:Nn\erw_seq_from_prop:NNn
244 {
245   \seq_if_exist:NTF#1
246   {\__erw_seq_set_from_prop:NNn#1#2{#3}}
247   {\seq_new:N#1\erw_seq_from_prop:NNn#1#2{#3}}
248 }
249 \cs_generate_variant:Nn \erw_seq_from_prop:NNn { cc }
250 \cs_new_protected:Nn\erw_seq_put_right:Nn
251 {
252   \seq_if_exist:NTF#1
```

```

253   {\seq_put_right:Nn#1{#2}}
254   {\seq_new:N#1\erw_seq_put_right:Nn #1{#2}}
255 }
256 \cs_generate_variant:Nn\erw_seq_put_right:Nn { c }
257 \cs_new:Nn \__erw_seq_fold:NN
258 {
259   \seq_get_right:NN #2 \g__erw_seq_fold_item_tl
260   \erw_tl_fold:NN #1 \g__erw_seq_fold_item_tl
261   \seq_put_right:No #2 {\g__erw_seq_fold_item_tl}
262 }
263 \cs_generate_variant:Nn \__erw_seq_fold:NN {cN}
264 \cs_new:Nn \erw_seq_use:Nn
265 {
266   \exp_last_unbraced:NNf
267   \seq_use:Nnnn #1
268   \erw_tl_separators:n{#2}
269 }

```

12 sys

12.1 backend

```

\__erw_sys_date:N
\__erw_sys_date_dec: 270 \cs_new:Nn \__erw_sys_date_dec:
\__erw_sys_date_hex: 271 {
272   \int_eval:n
273   {
274     \c_sys_year_int * 10000
275     +\c_sys_month_int * 100
276     +\c_sys_day_int * 1
277   }
278 }
279 \cs_new:Nn \__erw_sys_date:N{\int_to_hex:n{\__erw_sys_date_dec:}}
280 \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 281 \cs_new:Nn \__erw_sys_time_dec:
282 {
283   \int_eval:n
284   {
285     \c_sys_hour_int * 100
286     +\c_sys_minute_int * 1
287   }
288 }
289 \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 290 \cs_new:Nn\__erw_sys_datetime_base:n
\__erw_sys_datetime_join:nn 291 {
\__erw_sys_datetime_hex:n 292   \int_case:nnTF{#1}
\__erw_sys_datetime_period:n

```

```

293 {
294     {10}{dec}
295     {16}{hex}
296 }
297 {\c_empty_tl}
298 {\msg_error:nnn{__erw}{timestamp / base}{\__erw_sys_datetime_base:n{#1}}}
299 }
300 \cs_new:Nn\__erw_sys_datetime_join:nn{\erw_tl_join:nnn{#1}{\g__erw_sys_timestamp_delim_str}{#2}}
301 \cs_new:Nn\__erw_sys_datetime_period:n
302 {
303     \str_case:nnTF{#1}
304     {
305         {date}{date}
306         {time}{time}
307         {datetime}{datetime}
308     }
309     {\c_empty_tl}
310     {\msg_error:nnn{__erw}{ timestamp / period }{\__erw_sys_datetime_period:n{#1}}}
311 }
312 \cs_new:Nn\__erw_sys_datetime_dec: {\__erw_sys_datetime_join:nn{\__erw_sys_date_dec:}{\__erw_sys_timestamp_delim_str}{#1}}
313 \cs_new:Nn\__erw_sys_datetime_hex: {\__erw_sys_datetime_join:nn{\__erw_sys_date_hex:}{\__erw_sys_timestamp_delim_str}{#1}}

```

(End definition for __erw_sys_datetime_base:n and others.)

__erw_sys_jobnametimestamp_prefix:

```

314 \cs_new:Nn\__erw_sys_jobnametimestamp_prefix:
315 {
316     \erw_tl_join:nn
317     {\c_sys_jobname_str}
318     {\g__erw_sys_timestamp_delim_str}
319 }
320 % \begin{macro}{\__erw_sys_jobnametimestamp:n, \__erw_sys_jobnametimestamp:}
321 %     \begin{macrocode}
322 \cs_new:Nn\__erw_sys_jobnametimestamp:nn
323 {
324     \erw_tl_join:nn
325     {\__erw_sys_jobnametimestamp_prefix:}
326     {\erw_sys_timestamp:nn{#1}{#2}}
327 }
328 \cs_new:Nn\__erw_sys_jobnametimestamp:
329 {
330     \erw_tl_join:nn
331     {\__erw_sys_jobnametimestamp_prefix:}
332     {\erw_sys_timestamp:}
333 }

```

(End definition for __erw_sys_jobnametimestamp_prefix:.)

__erw_sys_timestamp:nn

```

334 \cs_new:Nn\__erw_sys_timestamp:nn
335 {
336     \exp_args:No
337     \use:c{\__erw_sys\___erw_sys_datetime_period:n{#1}\__erw_sys_datetime_base:n{#2}:}
338 }
339 \cs_new_protected:Nn \__erw_sys_set_delim:nn

```

```

340 {
341   \use:c{tl_gset:N#1}
342   \g__erw_sys_timestamp_delim_str{#2}
343 }

(End definition for \__erw_sys_timestamp:nn.)

344 \keys_define:nn{\__erw}
345 {
346   sys / timestamp_delim .code:n =
347   {
348     \exp_last_unbraced:No
349     \__erw_sys_set_delim:nn{n}{#1}
350   },
351   sys / timestamp_delim .value_required:n = true,
352   sys / timestamp_delim .default:n = {-},
353   sys / timestamp_delim .initial:n = {-}
354 }
355 % \subsection{frontend}
356 %   \begin{macrocode}
357 \cs_new:Nn\erw_sys_jobnametimestamp:nn{\__erw_sys_jobnametimestamp:nn{#1}{#2}}
358 \cs_new:Nn\erw_sys_jobnametimestamp:{\__erw_sys_jobnametimestamp:}
359 \cs_new:Nn\erw_sys_timestamp_delimiter:
360 {
361   \use:N \g__erw_sys_timestamp_delim_str
362 }
363 \cs_new:Nn\erw_sys_timestamp:nn
364 {
365   \__erw_sys_timestamp:nn{#1}{#2}
366 }
367 \cs_new:Nn\erw_sys_timestamp:
368 {
369   \__erw_sys_timestamp:nn{datetime}{16}
370 }

```

13 tl

13.1 backend

```

371 \tl_new:N \g__erw_tl_compose_tl

\g__erw_tl_function:n

372 \cs_new_protected:Nn \g__erw_tl_function:n
373 {
374   \msg_error:nnn
375   {erw}
376   {csnset}
377   {\g__erw_tl_function:n}
378 }

(End definition for \g__erw_tl_function:n.)

\__erw_map:nn

379 \cs_set_protected:Nn \__erw_map:nn
380 {

```

```

381 \quark_if_recursion_tail_stop:n{#1}
382 \g__erw_tl_function:n{#1} \__erw_map:nn{#2}
383 }

```

(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
384 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnn
385 {
386   \erw_cs_apply:Nn #1
387   {\exp_args:Nf\__erw_tl_map_thread_at:Nnn {#3} {#2} }
388 }
389 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnn
390 {
391   \erw_cs_apply:Nnn #1
392   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnn {#3} {#2} }
393   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnn {#4} {#2} }
394 }
395 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnn
396 {
397   \erw_cs_apply:Nnnn #1
398   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnn {#3} {#2} }
399   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnnn {#4} {#2} }
400   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnnnn {#5} {#2} }
401 }
402 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnnn
403 {
404   \erw_cs_apply:Nnnnn #1
405   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnnn {#3} {#2} }
406   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnnnn {#4} {#2} }
407   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnnnnn {#5} {#2} }
408   {\exp_args:Nf\__erw_tl_map_thread_at:Nnnnnnnnn {#6} {#2} }
409 }

```

(End definition for __erw_tl_map_thread_at:Nnn and others.)

```

\__erw_tl_separators:nn #1 : < int >
                        #2 : < items >
410 \cs_new:Nn \__erw_tl_separators:nn
411 {
412   \int_case:nnTF {#1}
413   {
414     {1}
415     { \prg_replicate:nn{ 3 }{#2} }
416     {2}
417     {
418       { \use_ii:nn #2 }
419       { \use_i:nn #2 }
420       { \use_i:nn #2 \use_ii:nn #2 }
421     }
422     {3}{#2}
423   }
424   { \c_empty_tl }
425   {

```



```

426 \msg_error:nnnn { __erw }
427 { separ }
428 { \exp_not:N \__erw_tl_separators:nn }
429 {#2}
430 }
431 }
432 \cs_generate_variant:Nn \__erw_tl_separators:nn { e }

(End definition for \__erw_tl_separators:nn.)

```

13.2 frontend

```

433 \cs_new:Nn \erw_tl_append_item:nn
434 {
435   {#1{#2}}
436 }
437 \cs_new:Nn \erw_tl_compose:nN
438 {
439   \__erw_oper_compose:NnN \erw_tl_fold:NN {#1} #2
440 }
441 \cs_new:Nn \erw_tl_compose:nn
442 {
443   \tl_set:Nn \g__erw_tl_compose_tl {#2}
444   \erw_tl_compose:nN{#1}\g__erw_tl_compose_tl
445   \g__erw_tl_compose_tl
446 }
447 \cs_new:Nn \erw_tl_compose_c:nN
448 {
449   \__erw_oper_compose:NnN \erw_tl_fold:cN {#1} #2
450 }
451 \cs_new:Nn \erw_tl_compose_c:nn
452 {
453   \tl_set:Nn \g__erw_tl_compose_tl {#2}
454   \erw_tl_compose_c:nN{#1}\g__erw_tl_compose_tl
455   \g__erw_tl_compose_tl
456 }
457 \cs_new:Nn \erw_tl_compose_vers:nN
458 {
459   \msg_error:nnn{__erw}{csnset}{\erw_tl_compose_vers:nN}
460 }
461 \cs_new:Nn \erw_tl_compose_vers:nn
462 {
463   \erw_csint_reset:{}
464   \tl_map_function:nN{#1}\erw_csint_new:n
465   \exp_last_unbraced:Nx
466   \erw_tl_compose_c:nn
467   {{\erw_csint_names_braced:{}}}
468   {#2}
469 }
470 \cs_new:Nn \erw_tl_fold:NN
471 {
472   \use:c{tl_set:\g__erw_oper_fold_set_par_tl}
473   #2
474   {\use:c{erw_cs_apply:\g__erw_oper_fold_apply_par_tl}{#1}{#2}}

```

```

475 }
476 \cs_generate_variant:Nn \erw_tl_fold:NN {cN}
477 \cs_new:Nn \erw_tl_gset_function:N
478 {
479   \erw_cs_gset_eq:NN \g__erw_tl_function:n #1
480 }
481 \cs_new:Nn \erw_tl_gset_function:n
482 {
483   \erw_cs_gset_inline:Nn \g__erw_tl_function:n {#1}
484 }
485 \cs_new:Nn \erw_tl_last_item:n
486 {
487   \exp_args:Nof \tl_item:nn
488     {#1}
489   {
490     \tl_count:n{#1}
491   }
492 }
493 \cs_new_protected:Nn \erw_tl_map:n
494 {
495   \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
496 }
497 \cs_new_protected:Nn \erw_tl_map:Nn
498 {
499   \cs_set_eq:NN \g__erw_tl_function:n #1
500   \erw_tl_map:n{#2}
501 }
502 \cs_new_protected:Nn \erw_tl_map_inline:nn
503 {
504   \erw_cs_set_inline:Nn \g__erw_tl_function:n {#1}
505   \erw_tl_map:n{#2}
506 }
507 \cs_new:Nn \erw_tl_repeat:nn
508 {
509   \int_step_inline:nnnn{1}{1}{#1}{#2}
510 }
511 \cs_new:Nn \erw_tl_split:nnn
512 {
513   \tl_head:n{#1}
514   \use:c{exp_args:#3} \tl_map_inline:nn
515     {
516       \tl_tail:n
517       {
518         #1
519       }
520     }{#2##1}
521 }
522 \cs_new:Nn \erw_tl_split:nn
523 {
524   \erw_tl_split:nnn{#1}{#2}{Nf}
525 }
526 \cs_new_protected:Nn \erw_tl_map_thread_at:Nnn
527 {
528   \exp_args:Nf\int_case:nnTF

```

```

529 {
530   \tl_count:n{#3}
531 }
532 {
533   {1}{ \__erw_tl_map_thread_at:Nnn #1{#2}#3 }
534   {2}{ \__erw_tl_map_thread_at:Nnnn #1{#2}#3 }
535   {3}{ \__erw_tl_map_thread_at:Nnnnn #1{#2}#3 }
536   {4}{ \__erw_tl_map_thread_at:Nnnnnn #1{#2}#3 }
537 }
538 {
539   % Do nothing
540 }
541 {
542   \msg_error:nnn{__erw}
543   {generic}
544   {erw_tl_map_thread_at:~count~of~#3~not~withing~1~to~4}
545 }
546 }
547 \cs_new_protected:Nn \erw_tl_map_thread:Nn
548 {
549   \int_step_inline:nn
550   {
551     \exp_args:Nf \tl_count:n{ \tl_head:n{#2} }
552   }
553   {
554     \erw_tl_map_thread_at:Nnn #1 {##1} {#2}
555   }
556 }
557 \cs_new:Nn \erw_tl_separators:n
558 {
559   \__erw_tl_separators:en{ \tl_count:n{#1} }{#1}
560 }

```

14 option

```

561 \cs_new_protected:Nn\erw_option:n
562 {
563   \keys_set:nn{__erw}{#1}
564 }

```

15 Closing

```

565 \ExplSyntaxOff
566 \</package>

```