

The `erw-l3` package ^{*}

Erwann Rogard[†]

Released 2020/05/21

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	5
9	sys	6
10	tl	6
II	Listing	8
1	constants	8
1.	8

^{*}This file describes version v2.4, last revised 2020/05/21.

[†]firstname dot lastname AusTria gmail dot com

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

1	Opening	17
2	basics	17
	2.1 backend	17
	2.2 frontend	17
3	clist	18
	3.1 backend	18
	3.2 frontend	18
4	csint	18
	4.1 backend	18
	4.2 frontend	18
5	int	18
	5.1 backend	18
	5.2 frontend	19
6	keyval	19
7	lambda	19
8	msg	20
	8.1 backend	20
	8.2 frontend	20
9	prop	20
	9.1 backend	20
	9.2 frontend	20
10	oper	21
	10.1 backend	21
	10.2 frontend	21
11	seq	21
	11.1 backend	21
	11.2 frontend	22
12	sys	22
	12.1 backend	22
13	tl	25
	13.1 backend	25
	13.2 frontend	26
14	option	29

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 \LaTeX 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_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.

<code>\erw_seq_compose:nN</code>	<code>\erw_seq_compose:nN{{⟨cs₁⟩}...}⟨seq⟩</code>
----------------------------------	--

<code>\erw_seq_compose_c:nN</code>	<code>\erw_seq_compose_c:nN{{⟨cs name₁⟩}...}⟨seq⟩</code>
------------------------------------	---

<code>\erw_seq_compose_vers:nN</code>	<code>\erw_seq_compose:nN{\langle cs or code_1 \rangle}...\langle seq \rangle</code>
---------------------------------------	--

<code>\erw_seq_from_clist:Nn</code>	<code>\erw_seq_from_clist:Nn\langle seq \rangle\langle clist \rangle</code>
<code>\erw_seq_from_clist:cn</code>	

<code>\erw_seq_from_prop:NNn</code>	<code>\erw_seq_from_prop:NNn\langle seq \rangle\langle prop \rangle\langle keyval list \rangle</code>
-------------------------------------	---

<code>\erw_seq_put_right:Nn</code>	<code>\erw_seq_put_right:Nn\langle seq \rangle\langle token list \rangle</code>
------------------------------------	---

<code>\erw_seq_use:Nn</code>	<code>\erw_seq_use:Nn\langle seq \rangle\langle items \rangle</code>
------------------------------	--

Also see [1, Section 8 of l3seq]

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

9 sys

<code>\erw_sys_jobnametimestamp:nn</code>	<code>\erw_sys_jobnametimestamp:nn{date time datetime}{10 16}</code>
<code>\erw_sys_jobnametimestamp:</code>	

<code>\erw_sys_timestamp:nn</code>	<code>\erw_sys_timestamp:nn{date time datetime}{10 16}</code>
<code>\erw_sys_timestamp:</code>	Semantics Timestamp in base 10 or 16

<code>\erw_sys_timestamp_delimiter:</code>	<code>\erw_sys_timestamp_delimiter:</code>
--	--

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{\langle arg list \rangle}\langle arg \rangle</code>
-------------------------------------	--

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

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

<code>\erw_tl_compose_vers:nN</code>	<code>\erw_tl_compose_vers:nn{\langle cs or code_1 \rangle}...\langle token list \rangle</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 list₁⟩}{⟨token list₂⟩}</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 $\langle items \rangle$:

- 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_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\}}
\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 10.

```
\ExplSyntaxOn
\erw_seq_from_prop:Nn \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
```

```

20200522-150
20200522-150
1343c4a%96
1343c4a%96
erw-l3%1343c4a%96

```

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:20200522150
D:20200522015023-04'00'
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: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 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 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' 16	v1.6	General: Fix: critical bug preventing <code>erw-l3</code> from working without explicit inclusion of <code>expl3</code> 16
	<code>disambig/backend</code> : changes to the key, added <code>\ProcessPackageKeysOption</code> ; . . . 16	v1.7	General: Add: <code>option</code> 16
	Brought all the modules under one file; renamed <code>l3erw</code> to <code>erw-l3</code> ; 16		Add: <code>sys</code> 16
v1.2	General: 16		Move: <code>\erw_fold_apply_par:n</code> . . 16
	<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) 16		Move: <code>\erw_fold_set_par:n</code> 16
	<code>disambig</code> : pushed the code inside <code>\keys_define:\disambignewcmd</code> no longer takes a token name as arg, rather a token. 16		Rearrange: structure of implementation, e.g. <code>section 10</code> . . 16
	Add: <code>\erw_items_to</code> 16		Remove: document level functions, <code>\numbrdcsnew</code> , <code>\numbrdcs</code> 16
	Add: <code>\erw_last_item</code> 16		Replace: listing's implem with that of <code>tocloft</code> 16
	Add: <code>\erw_repeat</code> 16		Replace: vers. numb. from 3 to 2 digits 16
	Add: <code>\erw_split</code> 16	v1.8	General: Add: function for all frontend functions. 16
	Add: <code>\map_thread</code> 16		Remove: <code>\erw_cs_set_eq:NN</code> and variants 16
	Front end cmds no longer generated with module <code>disambig</code> ; Option of the same name deleted; 16		Remove: <code>\erw_is_matrix:n</code> (predicate must be expandable) . . 16
	Re-arrange: the doc to clearly separate frontend from backend . . 16		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> 16
v1.3	General: Replace: versioning, should have been 0.1.2 16		Replace: <code>\@@_map:n</code> by <code>\@@_oper_function:n</code> 16
v1.4	General: Add: <code>\erw_accum</code> 16		Replace: <code>\erw_seq_fold:NN</code> by <code>\erw_oper_fold_seq:NN</code> and likewise for variants 16
	Add: <code>\erw_int_range</code> 16	v1.9	General: Add:
	Add: <code>\erw_is_matrix</code> (to check arg of <code>\erw_tl_map_thread:Nn</code>) 16		<code>\erw_sys_timestamp_delimiter:</code> . 16
	Add: <code>\erw_merge</code> 16		Add: <code>\erw_tl_join:nn</code> and variants . 16
	Add: <code>\erw_set_map_inline</code> 16		Rename: <code>\erw_append_arg:nn</code> to <code>\erw_tl_append_item:nn</code> 16
	Add: <code>\erw_set_map</code> 16		Rename:
	Remove: <code>\erw_items_to</code> (redundant with <code>\tl_range:nnn</code>) . 16		<code>\erw_oper_gset_function:N</code> to <code>\erw_tl_gset_function:N</code> (and variants) 16
v1.5	General: Modify: source repository . . 16	v2.0	General: Add:
	Rearrange: frontend/backend sections 16		<code>\erw_jobnametimestamp:nn</code> and variants 16
	Remove: <code>disambig</code> 16		Remove: <code>\merge:nn</code> (redundant with <code>\erw_join:nn</code>) 16
	Split Section Preliminaries into Conventions and Requirement. . . 16		

	Rename: v0.0 to v1.0, etc. 16		Add: \erw_tl_separators:n 16
v2.1	General: Add: \erw_prop_to_clist:Nn, \erw_prop_put:NN, and \erw_prop_put:Nnn 16 Add: \erw_seq_from_clist:Nn, \erw_seq_from_prop:Nnn, and \erw_seq_put_right:Nn 16 Move: all functions under section 10 to section 13 or section 11, except \@@oper_compose:NnN 16 Replace: \erw_seq_fold:NN by _erw_seq_fold:NN 16	v2.3	General: Add: \msg_new:nnn, module erw, messages: csnsset 16 Add: \msg_new:nnn, module erw, messages: keyval/... 16 Fix: 'mark as private code' (hitherto unnoticed) 16 Modify: behavior of \erw_seq_use:Nn 16 Move: all \msg_new:Nnnn statements under same heading .. 16
v2.2	General: Add: \erw_seq_use:Nn 16	v2.4	General: Add: \erw_lambda:nnn 16

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.

	B		E
\begin	299, 300, 335	erw commands:	
	C	\erw_cs_apply:Nn	4, 8, 12, 44, 365
cs commands:		\erw_cs_apply:Nnn	4, 13, 370
\cs_generate_variant:Nn	12, 30, 35, 139, 149, 161, 196, 202, 221, 228, 235, 242, 411, 455	\erw_cs_apply:Nnnnn	4, 17, 376
\cs_gset:Npn	33	\erw_cs_apply:Nnnnn	4, 21, 383
\cs_new:Nn 4, 8, 13, 17, 21, 26, 31, 36, 37, 38, 39, 42, 46, 47, 62, 67, 68, 98, 102, 108, 162, 203, 207, 211, 236, 243, 249, 258, 259, 260, 268, 269, 279, 280, 291, 292, 293, 301, 307, 313, 336, 337, 338, 342, 346, 389, 412, 416, 420, 426, 430, 436, 440, 449, 456, 460, 464, 486, 490, 501, 536		\erw_cs_gset:eq:NN	458
\cs_new_protected:Nn . 51, 72, 127, 141, 150, 185, 197, 215, 222, 229, 318, 351, 472, 476, 481, 505, 526, 540		\erw_cs_gset_inline:Nn ...	31, 35, 462
\cs_new_protected:Npn	112	\erw_cs_identity:n	4, 25
\cs_set:Nn	129, 143	\erw_cs_set_inline:Nn	4, 26, 30, 54, 164, 483
\cs_set:Npn	25, 28, 77	\erw_csint:nn	4, 42
\cs_set_eq:NN	478	\erw_csint_name:n ...	4, 41, 46, 49, 67
\cs_set_protected:Nn	187, 358, 363, 368, 374, 381	\erw_csint_names:nnn	4, 47
\cs_split_function:N	6	\erw_csint_names_braced: ..	4, 68, 446
		\erw_csint_names_braced:n ..	4, 64, 67
		\erw_csint_names_braced:nnn	4, 62, 70
		\erw_csint_new:n	5, 51, 443
		\erw_csint_reset:	5, 72, 442
		\erw_int_range:n	5, 102
		\erw_int_range:nn	5, 98
		\erw_keyval_keyonly:nn .	108, 135, 193
		\erw_lambda:nnn	5, 112
		\erw_option:n	5, 540
		\erw_prop_put:NN	5, 141, 149
		\erw_prop_put:Nnn ...	5, 150, 158, 161
		\erw_prop_to_clist:Nn	5, 127, 139, 200
		\erw_seq_compose:nN	5, 6, 203
		\erw_seq_compose_c:nN	5, 207
		\erw_seq_compose_vers:nN .	6, 211, 213

\erw_seq_from_clist:Nn	6, 215, 219, 221	\g__erw_seq_fold_item_tl
\erw_seq_from_prop:NNn	6, 222, 226, 228	184, 238, 239, 240
\erw_seq_put_right:Nn	6, 229, 233, 235	__erw_seq_set_from_clist:Nn	...
\erw_seq_use:Nn 6, 243	185, 196, 199, 218
\erw_sys_jobnametimestamp:..	6, 337	__erw_seq_set_from_prop:NNn	...
\erw_sys_jobnametimestamp:nn	6, 336	197, 202, 225
\erw_sys_timestamp:.....	6, 311, 346	__erw_sys_date:N 249
\erw_sys_timestamp:nn	... 6, 305, 342	__erw_sys_date_dec:.....	249, 291
\erw_sys_timestamp_delimiter:	6, 338	__erw_sys_date_hex:.....	249, 292
\erw_tl_append_item:nn	... 6, 89, 412	__erw_sys_datetime_base:n	.. 269, 316
\erw_tl_compose:nN 6, 416, 423	__erw_sys_datetime_dec:.....	291
\erw_tl_compose:nn 6, 420	__erw_sys_datetime_dec:n 269
\erw_tl_compose_c:nN	... 6, 426, 433	__erw_sys_datetime_hex:.....	292
\erw_tl_compose_c:nn	... 6, 430, 445	__erw_sys_datetime_hex:n 269
\erw_tl_compose_vers:nN	.. 6, 436, 438	__erw_sys_datetime_join:nn	... 269
\erw_tl_compose_vers:nn 6, 440	__erw_sys_datetime_period:n	269, 316
\erw_tl_fold:NN	__erw_sys_jobnametimestamp:..	...
.....	7, 239, 418, 428, 449, 455	299, 307, 337
\erw_tl_gset_function:N 7, 456	__erw_sys_jobnametimestamp:n	.. 299
\erw_tl_gset_function:n	... 7, 7, 460	__erw_sys_jobnametimestamp:nn	...
\erw_tl_join:nn	.. 7, 36, 295, 303, 309	301, 336
\erw_tl_join:nnn 7, 37, 279	__erw_sys_jobnametimestamp_	
\erw_tl_join:nnnn 7, 38	prefix:.....	293
\erw_tl_join:nnnnn 7, 39	__erw_sys_set_delim:nn	... 318, 328
\erw_tl_last_item:n 7, 464	__erw_sys_time_dec:.....	260, 291
\erw_tl_map:n	... 7, 168, 472, 479, 484	__erw_sys_time_hex 260
\erw_tl_map:Nn 7, 476	__erw_sys_time_hex:.....	268, 292
\erw_tl_map_inline:nn 7, 481	__erw_sys_timestamp:nn	313, 344, 348
\erw_tl_map_thread:Nn 7, 526	\g__erw_sys_timestamp_delim_str	..
\erw_tl_map_thread_at:Nnn	7, 505, 533	279, 297, 321, 340
\erw_tl_math_thread:Nn 7	\g__erw_tl_compose_tl
\erw_tl_math_thread_at:Nnn 7	350, 422, 423, 424, 432, 433, 434
\erw_tl_repeat:nn 7, 486	__erw_tl_map_thread_at:Nnn	363, 512
\erw_tl_separators:n	.. 6, 7, 247, 536	__erw_tl_map_thread_at:Nnnn	363, 513
\erw_tl_split:nn 7, 501	__erw_tl_map_thread_at:Nnnnn	...
\erw_tl_split:nnn 7, 490, 503	363, 514
erw internal commands:		__erw_tl_map_thread_at:Nnnnnn	..
__erw_cs_name:N 4	363, 515
__erw_csint_ext_tl 75	__erw_tl_separators:nn	... 389, 538
\g__erw_csint_int	.. 40, 41, 53, 70, 74	exp commands:	
\g__erw_csint_name_tl 41, 54	\exp_args:Nf
__erw_function:n 187, 192	130, 168, 366, 371, 372, 377,
__erw_int_range:nnn	77, 87, 100, 104	378, 379, 384, 385, 386, 387, 507, 530	
__erw_keyval_function:n	... 129, 134	\exp_args:NNx 114
__erw_lambda_expression	... 115, 118	\exp_args:No 315
__erw_map:nn 358, 474	\exp_args:Nof 466
__erw_oper_compose:NnN	\exp_args:Nx 89
.....	162, 205, 209, 418, 428	\exp_last_unbraced:Nf 6
\g__erw_oper_fold_apply_par_tl	..	\exp_last_unbraced:NNf 245
.....	179, 453	\exp_last_unbraced:No 327
\g__erw_oper_fold_set_par_tl	175, 451	\exp_last_unbraced:Nx 444
__erw_prop_append:nn 143, 147	\exp_not:N 407
__erw_seq_fold:NN	.. 205, 209, 236, 242	\ExplSyntaxOff 544
		\ExplSyntaxOn 3

G	<code>\prop_put:Nnn</code>	154
g internal commands:		
<code>\g_erw_tl_function:n</code>		
.	164, 351, 361, 458, 462, 478, 483	
I		
int commands:		
<code>\int_case:nnTF</code>	271, 391, 507	
<code>\int_compare:nNnTF</code>	79	
<code>\int_eval:n</code>	81, 91, 94, 251, 262	
<code>\int_incr:N</code>	53	
<code>\int_new:N</code>	40	
<code>\int_step_function:nnnN</code>	49, 64	
<code>\int_step_inline:nn</code>	106, 528	
<code>\int_step_inline:nnnn</code>	488	
<code>\int_to_alph:n</code>	44, 46	
<code>\int_to_hex:n</code>	258, 259, 268	
<code>\int_zero:N</code>	74	
K		
keys commands:		
<code>\keys_define:nn</code>	173, 323	
<code>\keys_set:nn</code>	542	
keyval commands:		
<code>\keyval_parse:Nnn</code>	133, 191	
M		
msg commands:		
<code>\msg_error:nnn</code>		
.	110, 213, 277, 289, 353, 438, 521	
<code>\msg_error:nnnn</code>	405	
<code>\msg_new:nnn</code>		
.	120, 121, 122, 123, 124, 125, 126	
O		
oper / fold_apply_par (option)	5	
oper / fold_set_par (option)	5	
options:		
oper / fold_apply_par	5	
oper / fold_set_par	5	
sys / timestamp_delim	5	
P		
prg commands:		
<code>\prg_replicate:nn</code>	394	
prop commands:		
<code>\prop_gput:Nnn</code>	145	
<code>\prop_if_exist:NTF</code>	152	
<code>\prop_item:Nn</code>	129, 145	
<code>\prop_map_function:NN</code>	147	
<code>\prop_new:N</code>	157	
Q		
quark commands:		
<code>\quark_if_recursion_tail_stop:n</code>	360	
<code>\q_recursion_stop</code>	474	
<code>\q_recursion_tail</code>	474	
S		
seq commands:		
<code>\seq_get_right:NN</code>	238	
<code>\seq_if_exist:NTF</code>	217, 224, 231	
<code>\seq_new:N</code>	219, 226, 233	
<code>\seq_put_right:Nn</code>	189, 232, 240	
<code>\seq_use:Nnnn</code>	6, 246	
str commands:		
<code>\str_case:nnTF</code>	282	
<code>\subsection</code>	334	
sys / timestamp_delim (option)	5	
sys commands:		
<code>\c_sys_day_int</code>	255	
<code>\c_sys_hour_int</code>	264	
<code>\c_sys_jobname_str</code>	296	
<code>\c_sys_minute_int</code>	265	
<code>\c_sys_month_int</code>	254	
<code>\c_sys_year_int</code>	253	
T		
tl commands:		
<code>\c_empty_tl</code>	276, 288, 403	
<code>\tl_count:n</code>	469, 509, 530, 538	
<code>\tl_head:n</code>	492, 530	
<code>\tl_item:nn</code>	366, 371, 372,	
.	377, 378, 379, 384, 385, 386, 387, 466	
<code>\tl_map_function:nN</code>	443	
<code>\tl_map_inline:nn</code>	493	
<code>\tl_new:N</code>	184, 350	
<code>\tl_range_braced:nnn</code>	65	
<code>\tl_reverse:n</code>	170	
<code>\tl_set:Nn</code>	41, 75, 422, 432	
<code>\tl_tail:n</code>	131, 495	
token commands:		
<code>\token_if_cs:NTF</code>	56	
U		
use commands:		
<code>\use:N</code>	316, 320, 340, 451, 453, 493	
<code>\use_i:nn</code>	398, 399	
<code>\use_i:nnn</code>	6	
<code>\use_ii:nn</code>	397, 399	
<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_keyonly:nn
109 {
110   \msg_error:nnn{erw}{keyval/keyonly}{#1}{#2}
111 }

```

7 lambda

\erw_lambda:nnn

```

112 \cs_new_protected:Npn \erw_lambda:nnn #1 #2 #3
113 {
114   \exp_args:NNx
115   #1 \__erw_lambda_expression
116   {#2}
117   {#3}
118   \__erw_lambda_expression
119 }

```

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

8 msg

8.1 backend

```
120 \msg_new:nnn{__erw}{generic}{#1}
121 \msg_new:nnn{__erw}{separ}{#1~expects~1~to~3~items,~#2}
122 \msg_new:nnn{__erw}{timestamp / base}{Calling~#1,~arg~must~be~'dec|hex'}
123 \msg_new:nnn{__erw}{timestamp / period}{Calling~#1,~arg~must~be~'date|time|datetime'}
```

8.2 frontend

```
124 \msg_new:nnn{erw}{csnset}{#1~not~set}
125 \msg_new:nnn{erw}{keyval/keyonly}{passed~key~#1~val~#2~where~keyonly}
126 \msg_new:nnn{erw}{keyval/mandatval}{key~#1~has~no~matching~val}
```

9 prop

9.1 backend

9.2 frontend

```
127 \cs_new_protected:Nn \erw_prop_to_clist:Nn
128 {
129   \cs_set:Nn \__erw_keyval_function:n {,\prop_item:Nn#1{##1}}
130   \exp_args:Nf
131   \tl_tail:n
132   {
133     \keyval_parse:NNn
134     \__erw_keyval_function:n
135     \erw_keyval_keyonly:nn
136     {#2}
137   }
138 }
139 \cs_generate_variant:Nn \erw_prop_to_clist:Nn { c }
140
141 \cs_new_protected:Nn \erw_prop_put:NN
142 {
143   \cs_set:Nn \__erw_prop_append:nn
144   {
145     \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
146   }
147   \prop_map_function:NN #2 \__erw_prop_append:nn
148 }
149 \cs_generate_variant:Nn \erw_prop_put:NN { cc }
150 \cs_new_protected:Nn \erw_prop_put:Nnn
151 {
152   \prop_if_exist:NTF#1
153   {
154     \prop_put:Nnn #1 {#2}{#3}
155   }
156   {
157     \prop_new:N #1
158     \erw_prop_put:Nnn #1{#2}{#3}
159   }
160 }
```



```
161 \cs_generate_variant:Nn \erw_prop_put:Nnn { c }
```

10 oper

10.1 backend

```
162 \cs_new:Nn \__erw_oper_compose:NnN
163 {
164   \erw_cs_set_inline:Nn \g__erw_tl_function:n
165   {
166     #1{##1}#3
167   }
168   \exp_args:Nf\erw_tl_map:n
169   {
170     \tl_reverse:n{#2}
171   }
172 }
```

10.2 frontend

```
173 \keys_define:nn{__erw}
174 {
175   oper/fold_set_par.tl_gset:N = \g__erw_oper_fold_set_par_tl,
176   oper/fold_set_par.value_required:n = true,
177   oper/fold_set_par.default:n = {Nf},
178   oper/fold_set_par.initial:n = {Nf},
179   oper/fold_apply_par.tl_gset:N = \g__erw_oper_fold_apply_par_tl,
180   oper/fold_apply_par.value_required:n = true,
181   oper/fold_apply_par.default:n = {Nf},
182   oper/fold_apply_par.initial:n = {Nf}
183 }
```

11 seq

11.1 backend

```
184 \tl_new:N \g__erw_seq_fold_item_tl
185 \cs_new_protected:Nn\__erw_seq_set_from_clist:Nn
186 {
187   \cs_set_protected:Nn \__erw_function:n
188   {
189     \seq_put_right:Nn #1{##1}
190   }
191   \keyval_parse:NNn
192   \__erw_function:n
193   \erw_keyval_keyonly:nn
194   {#2}
195 }
196 \cs_generate_variant:Nn \__erw_seq_set_from_clist:Nn { c }
197 \cs_new_protected:Nn\__erw_seq_set_from_prop:NNn
198 {
199   \__erw_seq_set_from_clist:Nn #1
200   {\erw_prop_to_clist:Nn #2 {#3}}
201 }
202 \cs_generate_variant:Nn \__erw_seq_set_from_prop:NNn { cc }
```

11.2 frontend

```

203 \cs_new:Nn \erw_seq_compose:nN
204 {
205   \__erw_oper_compose:NnN \__erw_seq_fold:NN {#1} #2
206 }
207 \cs_new:Nn \erw_seq_compose_c:nN
208 {
209   \__erw_oper_compose:NnN \__erw_seq_fold:cN {#1} #2
210 }
211 \cs_new:Nn \erw_seq_compose_vers:nN
212 {
213   \msg_error:nnn{\__erw}{csnset}{\erw_seq_compose_vers:nN}
214 }
215 \cs_new_protected:Nn\erw_seq_from_clist:Nn
216 {
217   \seq_if_exist:NTF#1
218   {\__erw_seq_set_from_clist:Nn#1{#2}}
219   {\seq_new:N#1\erw_seq_from_clist:Nn#1{#2}}
220 }
221 \cs_generate_variant:Nn \erw_seq_from_clist:Nn { c }
222 \cs_new_protected:Nn\erw_seq_from_prop:NNn
223 {
224   \seq_if_exist:NTF#1
225   {\__erw_seq_set_from_prop:NNn#1#2{#3}}
226   {\seq_new:N#1\erw_seq_from_prop:NNn#1#2{#3}}
227 }
228 \cs_generate_variant:Nn \erw_seq_from_prop:NNn { cc }
229 \cs_new_protected:Nn\erw_seq_put_right:Nn
230 {
231   \seq_if_exist:NTF#1
232   {\seq_put_right:Nn#1{#2}}
233   {\seq_new:N#1\erw_seq_put_right:Nn #1{#2}}
234 }
235 \cs_generate_variant:Nn\erw_seq_put_right:Nn { c }
236 \cs_new:Nn \__erw_seq_fold:NN
237 {
238   \seq_get_right:NN #2 \g__erw_seq_fold_item_tl
239   \erw_tl_fold:NN #1 \g__erw_seq_fold_item_tl
240   \seq_put_right:No #2 {\g__erw_seq_fold_item_tl}
241 }
242 \cs_generate_variant:Nn \__erw_seq_fold:NN { cN}
243 \cs_new:Nn \erw_seq_use:Nn
244 {
245   \exp_last_unbraced:NNf
246   \seq_use:Nnnn #1
247   \erw_tl_separators:n{#2}
248 }

```

12 sys

12.1 backend

```

\__erw_sys_date:N
\__erw_sys_date_dec:
\__erw_sys_date_hex:

```

```

249 \cs_new:Nn \__erw_sys_date_dec:
250 {
251   \int_eval:n
252   {
253     \c_sys_year_int * 10000
254     +\c_sys_month_int * 100
255     +\c_sys_day_int * 1
256   }
257 }
258 \cs_new:Nn \__erw_sys_date:N{\int_to_hex:n{\__erw_sys_date_dec:}}
259 \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`

```

260 \cs_new:Nn \__erw_sys_time_dec:
261 {
262   \int_eval:n
263   {
264     \c_sys_hour_int * 100
265     +\c_sys_minute_int * 1
266   }
267 }
268 \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`

```

269 \cs_new:Nn \__erw_sys_datetime_base:n
270 {
271   \int_case:nnTF{#1}
272   {
273     {10}{dec}
274     {16}{hex}
275   }
276   {\c_empty_tl}
277   {\msg_error:nnn{\__erw}{timestamp / base}{\__erw_sys_datetime_base:n{#1}}}
278 }
279 \cs_new:Nn \__erw_sys_datetime_join:nn{\erw_tl_join:nnn{#1}{\g__erw_sys_timestamp_delim_str}{#2}}
280 \cs_new:Nn \__erw_sys_datetime_period:n
281 {
282   \str_case:nnTF{#1}
283   {
284     {date}{date}
285     {time}{time}
286     {datetime}{datetime}
287   }
288   {\c_empty_tl}
289   {\msg_error:nnn{\__erw}{timestamp / period}{\__erw_sys_datetime_period:n{#1}}}
290 }
291 \cs_new:Nn \__erw_sys_datetime_dec: {\__erw_sys_datetime_join:nn{\__erw_sys_date_dec:}{\__erw_sys_time_dec:}}
292 \cs_new:Nn \__erw_sys_datetime_hex: {\__erw_sys_datetime_join:nn{\__erw_sys_date_hex:}{\__erw_sys_time_hex:}}

```

(End definition for __erw_sys_datetime_base:n and others.)

_erw_sys_jobnametimestamp_prefix:

```

293 \cs_new:Nn\_erw_sys_jobnametimestamp_prefix:
294 {
295   \erw_tl_join:nn
296   {\c_sys_jobname_str}
297   {\g__erw_sys_timestamp_delim_str}
298 }
299 % \begin{macro}{\_erw_sys_jobnametimestamp:n, \_erw_sys_jobnametimestamp:}
300 %   \begin{macrocode}
301 \cs_new:Nn\_erw_sys_jobnametimestamp:nn
302 {
303   \erw_tl_join:nn
304   {\_erw_sys_jobnametimestamp_prefix:}
305   {\erw_sys_timestamp:nn{#1}{#2}}
306 }
307 \cs_new:Nn\_erw_sys_jobnametimestamp:
308 {
309   \erw_tl_join:nn
310   {\_erw_sys_jobnametimestamp_prefix:}
311   {\erw_sys_timestamp:}
312 }

```

(End definition for _erw_sys_jobnametimestamp_prefix:.)

_erw_sys_timestamp:nn

```

313 \cs_new:Nn\_erw_sys_timestamp:nn
314 {
315   \exp_args:No
316   \use:c{\_erw_sys\_erw_sys_datetime_period:n{#1}\_erw_sys_datetime_base:n{#2}:}
317 }
318 \cs_new_protected:Nn \_erw_sys_set_delim:nn
319 {
320   \use:c{tl_gset:N#1}
321   \g__erw_sys_timestamp_delim_str{#2}
322 }

```

(End definition for _erw_sys_timestamp:nn.)

```

323 \keys_define:nn{\_erw}
324 {
325   sys / timestamp_delim .code:n =
326   {
327     \exp_last_unbraced:No
328     \_erw_sys_set_delim:nn{n}{#1}
329   },
330   sys / timestamp_delim .value_required:n = true,
331   sys / timestamp_delim .default:n = {-},
332   sys / timestamp_delim .initial:n = {-}
333 }
334 % \subsection{frontend}
335 %   \begin{macrocode}
336 \cs_new:Nn\erw_sys_jobnametimestamp:nn{\_erw_sys_jobnametimestamp:nn{#1}{#2}}
337 \cs_new:Nn\erw_sys_jobnametimestamp:{\_erw_sys_jobnametimestamp:}
338 \cs_new:Nn\erw_sys_timestamp_delimiter:
339 {

```

```

340   \use:N \g__erw_sys_timestamp_delim_str
341 }
342 \cs_new:Nn\erw_sys_timestamp:nn
343 {
344   \__erw_sys_timestamp:nn{#1}{#2}
345 }
346 \cs_new:Nn\erw_sys_timestamp:
347 {
348   \__erw_sys_timestamp:nn{datetime}{16}
349 }

```

13 tl

13.1 backend

```

350 \tl_new:N \g__erw_tl_compose_tl

\g__erw_tl_function:n

351 \cs_new_protected:Nn \g__erw_tl_function:n
352 {
353   \msg_error:nnn
354   {erw}
355   {csnset}
356   {\g__erw_tl_function:n}
357 }

(End definition for \g__erw_tl_function:n.)

\__erw_map:nn

358 \cs_set_protected:Nn \__erw_map:nn
359 {
360   \quark_if_recursion_tail_stop:n{#1}
361   \g__erw_tl_function:n{#1} \__erw_map:nn{#2}
362 }

(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

363 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnn
364 {
365   \erw_cs_apply:Nn #1
366   {\exp_args:Nf\tl_item:nn {#3} {#2} }
367 }
368 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnn
369 {
370   \erw_cs_apply:Nnn #1
371   {\exp_args:Nf\tl_item:nn {#3} {#2} }
372   {\exp_args:Nf\tl_item:nn {#4} {#2} }
373 }
374 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnn
375 {
376   \erw_cs_apply:Nnnn #1
377   {\exp_args:Nf\tl_item:nn {#3} {#2} }
378   {\exp_args:Nf\tl_item:nn {#4} {#2} }

```

```

379   {\exp_args:Nf\tl_item:nn {#5} {#2} }
380 }
381 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnnn
382 {
383   \erw_cs_apply:Nnnnn #1
384   {\exp_args:Nf\tl_item:nn {#3} {#2} }
385   {\exp_args:Nf\tl_item:nn {#4} {#2} }
386   {\exp_args:Nf\tl_item:nn {#5} {#2} }
387   {\exp_args:Nf\tl_item:nn {#6} {#2} }
388 }

```

(End definition for __erw_tl_map_thread_at:Nnn and others.)

```

\__erw_tl_separators:nn #1 : < int >
                        #2 : < items >

389 \cs_new:Nn \__erw_tl_separators:nn
390 {
391   \int_case:nnTF {#1}
392   {
393     {1}
394     { \prg_replicate:nn{ 3 }{#2} }
395     {2}
396     {
397       { \use_ii:nn #2 }
398       { \use_i:nn #2 }
399       { \use_i:nn #2 \use_ii:nn #2 }
400     }
401     {3}{#2}
402   }
403   { \c_empty_tl }
404   {
405     \msg_error:nnnn { __erw }
406     { separ }
407     { \exp_not:N \__erw_tl_separators:nn }
408     {#2}
409   }
410 }
411 \cs_generate_variant:Nn \__erw_tl_separators:nn { e }

```

(End definition for __erw_tl_separators:nn.)

13.2 frontend

```

412 \cs_new:Nn \erw_tl_append_item:nn
413 {
414   {#1{#2}}
415 }
416 \cs_new:Nn \erw_tl_compose:nN
417 {
418   \__erw_oper_compose:NnN \erw_tl_fold:NN {#1} #2
419 }
420 \cs_new:Nn \erw_tl_compose:nn
421 {
422   \tl_set:Nn \g__erw_tl_compose_tl {#2}

```

```

423 \erw_tl_compose:nN{#1}\g__erw_tl_compose_tl
424 \g__erw_tl_compose_tl
425 }
426 \cs_new:Nn \erw_tl_compose_c:nN
427 {
428 \__erw_oper_compose:NnN \erw_tl_fold:cN {#1} #2
429 }
430 \cs_new:Nn \erw_tl_compose_c:nn
431 {
432 \tl_set:Nn \g__erw_tl_compose_tl {#2}
433 \erw_tl_compose_c:nN{#1}\g__erw_tl_compose_tl
434 \g__erw_tl_compose_tl
435 }
436 \cs_new:Nn \erw_tl_compose_vers:nN
437 {
438 \msg_error:nnn{__erw}{csnset}{\erw_tl_compose_vers:nN}
439 }
440 \cs_new:Nn \erw_tl_compose_vers:nn
441 {
442 \erw_csint_reset:{}
443 \tl_map_function:nN{#1}\erw_csint_new:n
444 \exp_last_unbraced:Nx
445 \erw_tl_compose_c:nn
446 {{\erw_csint_names_braced:{}}}
447 {#2}
448 }
449 \cs_new:Nn \erw_tl_fold:NN
450 {
451 \use:c{tl_set:\g__erw_oper_fold_set_par_tl}
452 #2
453 {\use:c{erw_cs_apply:\g__erw_oper_fold_apply_par_tl}{#1}{#2}}
454 }
455 \cs_generate_variant:Nn \erw_tl_fold:NN {cN}
456 \cs_new:Nn \erw_tl_gset_function:N
457 {
458 \erw_cs_gset_eq:NN \g__erw_tl_function:n #1
459 }
460 \cs_new:Nn \erw_tl_gset_function:n
461 {
462 \erw_cs_gset_inline:Nn \g__erw_tl_function:n {#1}
463 }
464 \cs_new:Nn \erw_tl_last_item:n
465 {
466 \exp_args:Nof \tl_item:nn
467 {#1}
468 {
469 \tl_count:n{#1}
470 }
471 }
472 \cs_new_protected:Nn \erw_tl_map:n
473 {
474 \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
475 }
476 \cs_new_protected:Nn \erw_tl_map:Nn

```

```

477 {
478   \cs_set_eq:NN \g__erw_tl_function:n #1
479   \erw_tl_map:n{#2}
480 }
481 \cs_new_protected:Nn \erw_tl_map_inline:nn
482 {
483   \erw_cs_set_inline:Nn \g__erw_tl_function:n {#1}
484   \erw_tl_map:n{#2}
485 }
486 \cs_new:Nn \erw_tl_repeat:nn
487 {
488   \int_step_inline:nnnn{1}{1}{#1}{#2}
489 }
490 \cs_new:Nn \erw_tl_split:nnn
491 {
492   \tl_head:n{#1}
493   \use:c{exp_args:#3} \tl_map_inline:nn
494   {
495     \tl_tail:n
496     {
497       #1
498     }
499   }{#2##1}
500 }
501 \cs_new:Nn \erw_tl_split:nn
502 {
503   \erw_tl_split:nnn{#1}{#2}{Nf}
504 }
505 \cs_new_protected:Nn \erw_tl_map_thread_at:Nnn
506 {
507   \exp_args:Nf\int_case:nnTF
508   {
509     \tl_count:n{#3}
510   }
511   {
512     {1}{ \__erw_tl_map_thread_at:Nnn #1{#2}#3 }
513     {2}{ \__erw_tl_map_thread_at:Nnnn #1{#2}#3 }
514     {3}{ \__erw_tl_map_thread_at:Nnnnn #1{#2}#3 }
515     {4}{ \__erw_tl_map_thread_at:Nnnnnn #1{#2}#3 }
516   }
517   {
518     % Do nothing
519   }
520   {
521     \msg_error:nnn{__erw}
522     {generic}
523     {erw_tl_map_thread_at:~count~of~#3~not~withing~1~to~4}
524   }
525 }
526 \cs_new_protected:Nn \erw_tl_map_thread:Nn
527 {
528   \int_step_inline:nn
529   {
530     \exp_args:Nf \tl_count:n{ \tl_head:n{#2} }

```



```

531   }
532   {
533     \erw_tl_map_thread_at:Nnn #1 {##1} {#2}
534   }
535 }
536 \cs_new:Nn \erw_tl_separators:n
537 {
538   \__erw_tl_separators:en{ \tl_count:n{#1} }{#1}
539 }

```

14 option

```

540 \cs_new_protected:Nn\erw_option:n
541 {
542   \keys_set:nn{__erw}{#1}
543 }

```

15 Closing

```

544 \ExplSyntaxOff
545 \</package>

```