

# The `erw-l3` package <sup>\*</sup>

Erwann Rogard<sup>†</sup>

Released 2020/05/22

## Abstract

Utilities like `expl3[1]`.

## Résumé

Utilitaires de type `expl3[1]`.

## Contents

<b>I</b>	<b>Usage</b>	<b>4</b>
1	Loading the package	4
2	<code>cs</code>	4
3	<code>csint</code>	4
4	<code>int</code>	5
5	<code>lambda</code>	5
6	<code>option</code>	5
7	<code>prop</code>	5
8	<code>seq</code>	5
9	<code>sys</code>	6
10	<code>tl</code>	6
<b>II</b>	<b>Listing</b>	<b>8</b>
1	<b>constants</b>	<b>8</b>
1.	.....	8

---

<sup>\*</sup>This file describes version v2.6, last revised 2020/05/22.

<sup>†</sup>firstname dot lastname AusTria gmail dot com

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

<b>Index</b>	<b>18</b>
<b>IV Implementation</b>	<b>22</b>
<b>1 Opening</b>	<b>22</b>
<b>2 cs</b>	<b>22</b>
2.1 backend . . . . .	22
2.2 frontend . . . . .	22
<b>3 clist</b>	<b>22</b>
3.1 backend . . . . .	22
3.2 frontend . . . . .	22
<b>4 csint</b>	<b>22</b>
4.1 backend . . . . .	22
4.2 frontend . . . . .	22
<b>5 int</b>	<b>23</b>
5.1 backend . . . . .	23
5.2 frontend . . . . .	24
<b>6 keys</b>	<b>24</b>
6.1 frontend . . . . .	24
<b>7 lambda</b>	<b>24</b>
<b>8 msg</b>	<b>24</b>
8.1 backend . . . . .	24
8.2 frontend . . . . .	25
<b>9 prop</b>	<b>25</b>
9.1 backend . . . . .	25
9.2 frontend . . . . .	25
<b>10 oper</b>	<b>27</b>
10.1 backend . . . . .	27
10.2 frontend . . . . .	27
<b>11 seq</b>	<b>27</b>
11.1 backend . . . . .	27
11.2 frontend . . . . .	28
<b>12 sys</b>	<b>29</b>
12.1 backend . . . . .	29
<b>13 tl</b>	<b>31</b>
13.1 backend . . . . .	31
13.2 frontend . . . . .	32
<b>14 option</b>	<b>35</b>

## 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<sup>A</sup>T<sub>E</sub>X engine. See [Part III, section 3](#).
2. Goes in the *preamble*

## 2 cs

---

<code>\erw_cs_apply:Nn</code>	<code>\erw_cs_apply:Nn {\cs} {\token list<sub>1</sub>}</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{&lt;integer&gt;}</code>
-------------------------------	--

---



---

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

---

## 4 int

---

<code>\erw_int_range:n</code>	<code>\erw_int_range:n{&lt;integer&gt;}</code>
<code>\erw_int_range:nn</code>	

---

## 5 lambda

---

<code>\erw_lambda:nnn</code>	<code>\erw_lambda:nnn&lt;token&gt;{&lt;arg spec&gt;}{&lt;code&gt;}</code>
------------------------------	---

---

## 6 option

---

<code>\erw_option:n</code>	<code>\erw_option:n{&lt;keyval list&gt;}</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_keyval_parse:NNNn</code>	<code>\erw_prop_keyval_parse:NNNn&lt;prop&gt;&lt;cs<sub>1</sub>&gt;&lt;cs<sub>2</sub>&gt;{&lt;keyval list&gt;}</code>
--	---

---



---

<code>\erw_prop_map_item:NNN</code>	<code>\erw_prop_map_item:NNN&lt;cs&gt;&lt;prop<sub>1</sub>&gt;&lt;prop<sub>2</sub>&gt;</code>
-------------------------------------	---

---



---

<code>\erw_prop_to_clist:Nn</code>	<code>\erw_prop_to_clist:Nn&lt;prop&gt;{&lt;key<sub>1</sub>&gt;,...}</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{{&lt;cs<sub>1</sub>&gt;}...}&lt;seq&gt;</code>
----------------------------------	--

---



---

<code>\erw_seq_compose_c:nN</code>	<code>\erw_seq_compose_c:nN{{&lt;cs name<sub>1</sub>&gt;}...}&lt;seq&gt;</code>
------------------------------------	---

---

---



---

`\erw_seq_compose_vers:nN`    `\erw_seq_compose:nN{<{cs or code1>}...}<seq>`

---



---



---

`\erw_seq_put_right_clist:Nn`    `\erw_seq_put_right_clist:Nn<seq>{<clist>}`  
`\erw_seq_put_right_clist:cn`

---



---



---

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

---



---



---

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

---

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{<cs1>}...{<token list>}`  
`\erw_tl_compose:nn`

---



---



---

`\erw_tl_compose_c:nN`    `\erw_tl_compose_c:nn{<cs name1>}...{<token list>}`  
`\erw_tl_compose_c:nn`

---



---



---

`\erw_tl_compose_vers:nN`    `\erw_tl_compose_vers:nn{<cs or code1>}...{<token list>}`  
`\erw_tl_compose_vers:nn`

---



---



---

`\erw_tl_fold:NN`    `\erw_tl_fold:NN<cs><tl var>`  
`\erw_tl_fold:cN`

---

<hr/> <hr/>	<code>\erw_tl_gset_function:N</code>	<code>\erw_tl_gset_function:n{⟨code⟩}</code>
<hr/>	<code>\erw_tl_gset_function:n</code>	
<hr/>	<code>\erw_tl_join:nn</code>	<code>\erw_tl_join:nn{⟨token list<sub>1</sub>⟩}{⟨token list<sub>2</sub>⟩}</code>
<hr/>	<code>\erw_tl_join:nnn</code>	
<hr/>	<code>\erw_tl_join:nnnn</code>	
<hr/>	<code>\erw_tl_join:nnnnn</code>	
<hr/>	<code>\erw_tl_last_item:n</code>	<code>\erw_tl_last_item:n{⟨token list⟩}</code>
<hr/>	<code>\erw_tl_map:n</code>	<code>\erw_tl_map:n{⟨items⟩}</code>
<hr/>	<code>\erw_tl_map:Nn</code>	<b>Semantics</b> Maps over $\langle items \rangle$ using the internal function set by <code>\erw_tl_gset_ function:n</code>
<hr/>	<code>\erw_tl_map_inline:nn</code>	<code>\erw_tl_map_inline:nn{⟨code⟩}{⟨items⟩}</code>
<hr/>	<code>\erw_tl_map_thread:Nn</code>	<code>\erw_tl_math_thread:Nn⟨cs⟩{⟨items⟩}</code>
<hr/>	<code>\erw_tl_map_thread_at:Nnn</code>	<code>\erw_tl_math_thread_at:Nnn{⟨integer⟩}{⟨token list⟩}</code>
<hr/>	<code>\erw_tl_repeat:nn</code>	<code>\erw_tl_repeat:nn{⟨integer⟩}{⟨token list⟩}</code>
<hr/>	<code>\erw_tl_split:nnn</code>	<code>\erw_tl_split:nn{⟨items⟩}{⟨delimiter⟩}</code>
<hr/>	<code>\erw_tl_split:nn</code>	
<hr/>	<code>\erw_tl_separators:n</code>	<code>\erw_tl_separators:n{⟨items⟩}</code>
	<b>Semantics</b> 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 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: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
```

```

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: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<sup>A</sup>T<sub>E</sub>X 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<sup>A</sup>T<sub>E</sub>X 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<sup>A</sup>T<sub>E</sub>X3 Project Team *The L<sup>A</sup>T<sub>E</sub>X3 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	v1.6	General: Fix: critical bug preventing erw-l3 from working without explicit inclusion of expl3 . . . . . 14
	disambig/backend: changes to the key, added	v1.7	General: Add: option . . . . . 14
	\ProcessPackageKeysOption; . . . 14		Add: sys . . . . . 14
	Brought all the modules under one file; renamed l3erw to erw-l3; . . . . 14		Move: \erw_fold_apply_par:n . . 14
v1.2	General: disambig: \disambignewcmd no longer takes a token name as arg, rather a token. . . . . 14		Move: \erw_fold_set_par:n . . . . 14
	disambig: pushed the code inside \keys_define; . . . . . 14		Rearrange: structure of implementation, e.g. section 10 . . 14
	Add: \erw_items_to . . . . . 14		Remove: document level functions, \numbrdcsnew, \numbrdcs . . . . . 14
	Add: \erw_last_item . . . . . 14		Replace: listing's implem with that of tocloft . . . . . 14
	Add: \erw_repeat . . . . . 14		Replace: vers. numb. from 3 to 2 digits . . . . . 14
	Add: \erw_split . . . . . 14	v1.8	General: Add: function for all frontend functions. . . . . 14
	Add: \map_thread . . . . . 14		Remove: \erw_cs_set_eq:NN and variants . . . . . 14
	Front end cmds no longer generated with module disambig; Option of the same name deleted; . . . . . 14		Remove: \erw_is_matrix:n (predicate must be expandable) . . 14
	Modify: \erw_compose, order in which functions composed ( $g \circ f$ means $f$ comes before $g$ ) . . . . . 14		Rename: all cs prefixes to agree with heading under which they come, e.g. \erw_identity:n by \erw_cs_identity:n . . . . . 14
	Rearrange: the doc to clearly separate frontend from backend . . 14		Replace: @@_map:n by @@_oper_function:n . . . . . 14
v1.3	General: Replace: versioning, should have been 0.1.2 . . . . . 14		Replace: \erw_seq_fold:NN by \erw_oper_fold_seq:NN and likewise for variants . . . . . 14
v1.4	General: Add: \erw_accum . . . . . 14	v1.9	General: Add:
	Add: \erw_int_range . . . . . 14		\erw_sys_timestamp_delimiter: 14
	Add: \erw_is_matrix (to check arg of \erw_tl_map_thread:Nn) . . . . 14		Add: \erw_tl_join:nn and variants 14
	Add: \erw_merge . . . . . 14		Rename: \erw_append_arg:nn to \erw_tl_append_item:nn . . . . . 14
	Add: \erw_set_map_inline . . . . 14		Rename:
	Add: \erw_set_map . . . . . 14		\erw_oper_gset_function:N to \erw_tl_gset_function:N (and variants) . . . . . 14
	Remove: \erw_items_to (redundant with \tl_range:nnn) . 14	v2.0	General: Add:
v1.5	General: Modify: source repository . . 14		\erw_jobnametimestamp:nn and variants . . . . . 14
	Rearrange: frontend/backend sections . . . . . 14		Remove: \merge:nn (redundant with \erw_join:nn) . . . . . 14
	Remove: disambig . . . . . 14		
	Split Section Preliminaries into Conventions and Requirement. . . 14		

	Rename: v0.0 to v1.0, etc. ....	14	v2.4		
v2.1	General: Add:			General: Add: \erw_lambda:nnn ....	14
	\erw_prop_to_clist:Nn,				
	\erw_prop_put:NN, and				
	\erw_prop_put:Nnn .....	14	v2.5	General: Add:	
	Add: \erw_seq_from_clist:Nn,			\erw_prop_put_keyval:Nn .....	14
	\erw_seq_from_prop:NNn, and				
	\erw_seq_put_right:Nn .....	14	v2.6	General: Add: \erw_cs_error:nn ...	14
	Move: all functions under section 10			Add: \erw_cs_error:n .....	14
	to section 13 or section 11, except			Add: \erw_keyval_parse:NNNn ..	14
	\@@_oper_compose:NnN .....	14		Add:	
	Replace: \erw_seq_fold:NN by			\erw_prop_keyval_parse:NNNn ..	14
	\__erw_seq_fold:NN .....	14		Add: \erw_prop_map_item:NNN ..	14
				Add: \msg_new:nnn, module erw,	
v2.2				messages: varnset .....	14
	General: Add: \erw_seq_use:Nn ....	14		Fix: inside implem of	
	Add: \erw_tl_separators:n ....	14		\erw_prop_put_keyval:Nn .....	14
v2.3				Remove: \erw_cs_apply .....	14
	General: Add: \msg_new:nnn, module			Remove: \erw_prop_put:NN .....	14
	erw, messages: csnset .....	14		Remove:	
	Add: \msg_new:nnn, module erw,			\erw_prop_put_keyval:Nn .....	14
	messages: keyval/... .....	14		Remove: \msg_new:nnn, module	
	Fix: 'mark as private code' (hiherto			erw, messages: keyval/... .....	14
	unnoticed) .....	14		Rename: basics to cs .....	14
	Modify: behavior of			Replace: \erw_seq_from_clist by	
	\erw_seq_use:Nn .....	14		\erw_seq_put_right_clist .....	14
	Move: all \msg_new:Nnnn			Replace: \erw_seq_from_prop by	
	statements under same heading ..	14		\erw_seq_put_right_prop .....	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.

<b>B</b>		116, 124, 137, 146, 155, 167, 175,
<code>\begin</code> .....	389	183, 206, 241, 253, 271, 278, 285,
		372, 405, 528, 532, 537, 561, 582, 596
<b>C</b>		
cs commands:		<code>\cs_new_protected:Npn</code> .... 102
<code>\cs_generate_variant:Nn</code> .....		<code>\cs_set:Nn</code> .... 126, 148
..... 13, 18, 136, 154, 166, 217,		<code>\cs_set:Npn</code> .... 8, 11, 62
252, 258, 277, 284, 291, 298, 465, 511		<code>\cs_set_eq:NN</code> .... 534
<code>\cs_gset:Npn</code> .... 16		<code>\cs_set_protected:Nn</code> . 95, 96, 118,
<code>\cs_new:Nn</code> .... 4, 9, 14, 19,		192, 193, 243, 412, 417, 422, 428, 435
20, 21, 22, 23, 24, 27, 31, 32, 47, 52,		<code>\cs_split_function:N</code> .... 6
53, 83, 87, 218, 259, 263, 267, 292,		
299, 305, 314, 315, 316, 324, 325,		
335, 336, 347, 348, 349, 355, 361,		
367, 390, 391, 392, 396, 400, 443,		
466, 470, 474, 480, 484, 490, 494,		
503, 512, 516, 520, 542, 546, 557, 592		
<code>\cs_new_protected:Nn</code> .. 36, 57, 93,		
		<b>E</b>
		erw commands:
		<code>\erw_cs_apply:Nn</code> .... 1
		<code>\erw_cs_apply:Nnn</code> .... 1
		<code>\erw_cs_apply:Nnnn</code> .... 1
		<code>\erw_cs_apply:Nnnnn</code> .... 1
		<code>\erw_cs_error:n</code> .... 23

<code>\erw_cs_error:nn</code>	24, 132	<code>\erw_tl_gset_function:N</code>	4, 512
<code>\erw_cs_gset_eq:NN</code>	514	<code>\erw_tl_gset_function:n</code>	4, 516
<code>\erw_cs_gset_inline:Nn</code>	14, 18, 518	<code>\erw_tl_join:nn</code>	4, 19, 351, 357, 363
<code>\erw_cs_identity:n</code>	1, 8	<code>\erw_tl_join:nnn</code>	4, 20, 335
<code>\erw_cs_set_inline:Nn</code>		<code>\erw_tl_join:nnnn</code>	4, 21
	1, 9, 13, 39, 220, 539	<code>\erw_tl_join:nnnnn</code>	4, 22
<code>\erw_csint:nn</code>	2, 27	<code>\erw_tl_last_item:n</code>	4, 520
<code>\erw_csint_name:n</code>	2, 26, 31, 34, 52	<code>\erw_tl_map:n</code>	4, 224, 528, 535, 540
<code>\erw_csint_names:nnn</code>	2, 32	<code>\erw_tl_map:Nn</code>	4, 532
<code>\erw_csint_names_braced:</code>	2, 53, 500	<code>\erw_tl_map_inline:nn</code>	4, 537
<code>\erw_csint_names_braced:n</code>	2, 49, 52	<code>\erw_tl_map_thread:Nn</code>	4, 582
<code>\erw_csint_names_braced:nnn</code>	2, 47, 55	<code>\erw_tl_map_thread_at:Nnn</code>	4, 561, 589
<code>\erw_csint_new:n</code>	2, 36, 497	<code>\erw_tl_math_thread:Nn</code>	4
<code>\erw_csint_reset:</code>	2, 57, 496	<code>\erw_tl_math_thread_at:Nnn</code>	4
<code>\erw_int_range:n</code>	2, 87	<code>\erw_tl_repeat:nn</code>	4, 542
<code>\erw_int_range:nn</code>	2, 83	<code>\erw_tl_separators:n</code>	5, 303, 592
<code>\erw_keyval_error:n</code>	179, 212	<code>\erw_tl_split:nn</code>	5, 557
<code>\erw_keyval_keyonly:nn</code>	172, 249	<code>\erw_tl_split:nnn</code>	5, 546, 559
<code>\erw_keyval_parse:NNNn</code>	93, 186	erw internal commands:	
<code>\erw_lambda:nnn</code>	2, 102	<code>\__erw_cs_name:N</code>	4
<code>\erw_option:n</code>	2, 596	<code>\__erw_csint_ext_tl</code>	60
<code>\erw_prop_gput_keyval:Nn</code>	206	<code>\g_erw_csint_int</code>	25, 26, 38, 55, 59
<code>\erw_prop_keyval_parse:NNn</code>	175	<code>\g_erw_csint_name_tl</code>	26, 39
<code>\erw_prop_keyval_parse:NNNn</code>		<code>\__erw_function:n</code>	243, 248
	3, 183, 189, 210	<code>\__erw_function:nn</code>	118, 122
<code>\erw_prop_keyval_parse_key:Nnn</code>	167	<code>\__erw_int_range:nnn</code>	62, 72, 85, 89
<code>\erw_prop_map_item:NNN</code>	3, 137, 143	<code>\__erw_keyval_function:n</code>	
<code>\erw_prop_put:NN</code>	146, 154		95, 98, 126, 131, 192, 197
<code>\erw_prop_put:Nnn</code>	155, 163, 166	<code>\__erw_keyval_function:nn</code>	
<code>\erw_prop_put_keyval:Nn</code>	217		96, 99, 193, 198
<code>\erw_prop_to_clist:Nn</code>	3, 124, 136, 256	<code>\__erw_lambda_expression</code>	105, 108
<code>\erw_put_right_prop:NNn</code>		<code>\__erw_map:nn</code>	412, 530
	3, 278, 282, 284	<code>\__erw_oper_compose:NnN</code>	
<code>\erw_seq_compose:nN</code>	3, 3, 259		218, 261, 265, 472, 482
<code>\erw_seq_compose_c:nN</code>	3, 263	<code>\g_erw_oper_fold_apply_par_tl</code>	
<code>\erw_seq_compose_vers:nN</code>	3, 267, 269		235, 508
<code>\erw_seq_put_right:Nn</code>	285, 289, 291	<code>\g_erw_oper_fold_set_par_tl</code>	231, 505
<code>\erw_seq_put_right_clist:Nn</code>		<code>\__erw_prop_append:nn</code>	148, 152
	3, 271, 275, 277	<code>\__erw_prop_fun_keyval:NNNn</code>	203
<code>\erw_seq_use:Nn</code>	3, 299	<code>\__erw_prop_keyval_parse:NNNn</code>	
<code>\erw_sys_jobnametimestamp:</code>	3, 391		169, 177
<code>\erw_sys_jobnametimestamp:nn</code>	3, 390	<code>\__erw_prop_map_item:NNN</code>	116, 140
<code>\erw_sys_timestamp:</code>	3, 365, 400	<code>\__erw_seq_fold:NN</code>	261, 265, 292, 298
<code>\erw_sys_timestamp:nn</code>	3, 359, 396	<code>\g_erw_seq_fold_item_tl</code>	
<code>\erw_sys_timestamp_delimiter:</code>	3, 392		240, 294, 295, 296
<code>\erw_tl_append_item:nn</code>	4, 74, 466	<code>\__erw_seq_put_right_clist:Nn</code>	
<code>\erw_tl_compose:nN</code>	4, 470, 477		241, 252, 255, 274
<code>\erw_tl_compose:nn</code>	4, 474	<code>\__erw_seq_put_right_prop:NNn</code>	
<code>\erw_tl_compose_c:nN</code>	4, 480, 487		253, 258, 281
<code>\erw_tl_compose_c:nn</code>	4, 484, 499	<code>\__erw_sys_date:N</code>	305
<code>\erw_tl_compose_vers:nN</code>	4, 490, 492	<code>\__erw_sys_date_dec:</code>	305, 347
<code>\erw_tl_compose_vers:nn</code>	4, 494	<code>\__erw_sys_date_hex:</code>	305, 348
<code>\erw_tl_fold:NN</code>		<code>\__erw_sys_datetime_base:n</code>	325, 370
	4, 295, 472, 482, 503, 511	<code>\__erw_sys_datetime_dec:</code>	347

\__erw_sys_datetime_dec:n	325	\int_new:N	25
\__erw_sys_datetime_hex:	348	\int_step_function:nnnN	34, 49
\__erw_sys_datetime_hex:n	325	\int_step_inline:nn	91, 584
\__erw_sys_datetime_join:nn	325	\int_step_inline:nnnn	544
\__erw_sys_datetime_period:n	325, 370	\int_to_alph:n	29, 31
\__erw_sys_jobnametimestamp:	355, 391	\int_to_hex:n	314, 315, 324
\__erw_sys_jobnametimestamp:n	355	\int_zero:N	59
\__erw_sys_jobnametimestamp:nn	355, 390	<b>K</b>	
\__erw_sys_jobnametimestamp_		keys commands:	
prefix:	349, 358, 364	\keys_define:nn	229, 377
\__erw_sys_set_delim:nn	372, 382	\keys_set:nn	598
\__erw_sys_time_dec:	316, 347	keyval commands:	
\__erw_sys_time_hex	316	\keyval_parse:NNn	97, 130, 196, 247
\__erw_sys_time_hex:	324, 348	<b>M</b>	
\__erw_sys_timestamp:nn	367, 398, 402	msg commands:	
\g__erw_sys_timestamp_delim_str	335, 353, 375, 394	\msg_error	23, 24
\g__erw_tl_compose_tl	404, 476, 477, 478, 486, 487, 488	\msg_error:nnn	269, 333, 345, 407, 492, 577
\__erw_tl_map_thread_at:Nnn	417, 568	\msg_error:nnnn	459
\__erw_tl_map_thread_at:Nnnn	417, 569	\msg_new:nnn	110, 111, 112, 113, 114, 115
\__erw_tl_map_thread_at:Nnnnn	417, 570	<b>O</b>	
\__erw_tl_map_thread_at:Nnnnnn	417, 571	oper / fold_apply_par (option)	2
\__erw_tl_separators:nn	443, 594	oper / fold_set_par (option)	2
exp commands:		options:	
\exp_args:Nf	127, 224, 420, 425, 426, 431, 432, 433, 438, 439, 440, 441, 563, 586	oper / fold_apply_par	2
\exp_args:NNx	104	oper / fold_set_par	2
\exp_args:No	369	sys / timestamp_delim	2
\exp_args:Nof	522	<b>P</b>	
\exp_args:Nx	74	prg commands:	
\exp_last_unbraced:Nf	6	\prg_replicate:nn	448
\exp_last_unbraced:NNf	301	prop commands:	
\exp_last_unbraced:No	381	\prop_gput:Nnn	150, 213
\exp_last_unbraced:Nx	498	\prop_if_exist:NTF	139, 157, 185, 194, 208
\exp_not:N	461	\prop_item:Nn	126, 150
\ExplSyntaxOff	600	\prop_map_function:NN	122, 152
\ExplSyntaxOn	3	\prop_new:N	142, 162, 188, 202
		\prop_put:Nnn	159
<b>G</b>		<b>Q</b>	
g internal commands:		quark commands:	
\g__erw_tl_function:n	220, 405, 415, 514, 518, 534, 539	\quark_if_recursion_tail_stop:n	414
<b>I</b>		\q_recursion_stop	530
int commands:		\q_recursion_tail	530
\int_case:nnTF	327, 445, 563	<b>S</b>	
\int_compare:nNnTF	64	seq commands:	
\int_eval:n	66, 76, 79, 307, 318	\seq_get_right:NN	294
\int_incr:N	38	\seq_if_exist:NTF	273, 280, 287
		\seq_new:N	275, 282, 289
		\seq_put_right:Nn	245, 288, 296



## Part IV

# Implementation

## 1 Opening

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

## 2 cs

### 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_set:Npn \erw_cs_identity:n #1{#1}
9 \cs_new:Nn \erw_cs_set_inline:Nn
10 {
11   \cs_set:Npn #1 ##1{#2}
12 }
13 \cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}
14 \cs_new:Nn \erw_cs_gset_inline:Nn
15 {
16   \cs_gset:Npn #1 ##1{#2}
17 }
18 \cs_generate_variant:Nn \erw_cs_gset_inline:Nn {cn}
19 \cs_new:Nn \erw_tl_join:nn{#1#2}
20 \cs_new:Nn \erw_tl_join:nnn{#1#2#3}
21 \cs_new:Nn \erw_tl_join:nnnn{#1#2#3#4}
22 \cs_new:Nn \erw_tl_join:nnnnn{#1#2#3#4#5}
23 \cs_new:Nn \erw_cs_error:n{\msg_error{__erw}{generic}{unary-function-not-allowed}}
24 \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

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

### 4.2 frontend

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

```

28 {
29   \use:c{__erw_csint\_int\_to\_alph:n{#1}:n}{#2}
30 }
31 \cs_new:Nn \erw_csint_name:n {__erw_csint\_int\_to\_alph:n{#1}:n}
32 \cs_new:Nn \erw_csint_names:nnn
33 {
34   \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_name:n
35 }
36 \cs_new_protected:Nn \erw_csint_new:n
37 {
38   \int_incr:N \g__erw_csint_int
39   \erw_cs_set_inline:cn{\g__erw_csint_name_tl}
40   {
41     \token_if_cs:NTF
42       {#1}
43       {#1{##1}}
44       {#1}
45   }
46 }
47 \cs_new:Nn \erw_csint_names_braced:nnn
48 {
49   \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_names_braced:n
50   % TODO \tl_range_braced:nnn?
51 }
52 \cs_new:Nn \erw_csint_names_braced:n {{\erw_csint_name:n{#1}}}
53 \cs_new:Nn \erw_csint_names_braced:
54 {
55   \erw_csint_names_braced:nnn{1}{1}{\g__erw_csint_int}
56 }
57 \cs_new_protected:Nn \erw_csint_reset:
58 {
59   \int_zero:N \g__erw_csint_int
60   \tl_set:Nn \__erw_csint_ext_tl{}%^A TODO remove?
61 }

```

## 5 int

### 5.1 backend

```

62 \cs_set:Npn \__erw_int_range:nnn #1 #2 #3
63 {
64   \int_compare:nNnTF
65     {
66       \int_eval:n{#2+1}
67     }>{#3}
68     {
69       {#1}
70     }
71     {
72       \__erw_int_range:nnn
73       {
74         \exp_args:Nx\erw_tl_append_item:nn{#1}
75         {
76           \int_eval:n{#2+1}

```

```

77     }
78   }
79   {\int_eval:n{#2+1}}
80   {#3}
81 }
82 }

```

## 5.2 frontend

```

83 \cs_new:Nn \erw_int_range:nn
84 {
85   \__erw_int_range:nnn {{#1}}{#1}{#2}
86 }
87 \cs_new:Nn \erw_int_range:n
88 {
89   \__erw_int_range:nnn {}{0}{#1}
90 % ^^A Alt to:
91 % ^^A   \int_step_inline:nn {#1}{##1}
92 }

```

## 6 keys

### 6.1 frontend

```

93 \cs_new_protected:Nn \erw_keyval_parse:NNNn
94 {
95   \cs_set_protected:Nn \__erw_keyval_function:n {#2 #1{##1}}
96   \cs_set_protected:Nn \__erw_keyval_function:nn {#3 #1{##1}{##2}}
97   \keyval_parse:NNn
98   \__erw_keyval_function:n
99   \__erw_keyval_function:nn
100   {#4}
101 }

```

## 7 lambda

`\erw_lambda:nnn`

```

102 \cs_new_protected:Npn \erw_lambda:nnn #1 #2 #3
103 {
104   \exp_args:NNx
105   #1 \__erw_lambda_expression
106   {#2}
107   {#3}
108   \__erw_lambda_expression
109 }

```

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

## 8 msg

### 8.1 backend

```

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

```



```

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

```

## 8.2 frontend

```

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

```

## 9 prop

### 9.1 backend

```

116 \cs_new_protected:Nn \__erw_prop_map_item:NNN
117 {
118   \cs_set_protected:Nn \__erw_function:nn
119   {
120     #1 #2 {##1}{##2}
121   }
122   \prop_map_function:NN #3 \__erw_function:nn
123 }

```

### 9.2 frontend

```

124 \cs_new_protected:Nn \erw_prop_to_clist:Nn
125 {
126   \cs_set:Nn \__erw_keyval_function:n {,\prop_item:Nn#1{##1}}
127   \exp_args:Nf
128   \tl_tail:n
129   {
130     \keyval_parse:NNn
131     \__erw_keyval_function:n
132     \erw_cs_error:nn
133     {#2}
134   }
135 }
136 \cs_generate_variant:Nn \erw_prop_to_clist:Nn { c }
137 \cs_new_protected:Nn \erw_prop_map_item:NNN
138 {
139   \prop_if_exist:NTF #2
140   {\__erw_prop_map_item:NNN #1#2#3}
141   {
142     \prop_new:N #2
143     \erw_prop_map_item:NNN #1#2#3
144   }
145 }
146 % ^^A\cs_new_protected:Nn \erw_prop_put:NN
147 % ^^A{
148 % ^^A \cs_set:Nn \__erw_prop_append:nn
149 % ^^A {
150 % ^^A \prop_gput:Nnx #1 {##1}{ \prop_item:Nn #2{##1} }
151 % ^^A }
152 % ^^A \prop_map_function:NN #2 \__erw_prop_append:nn
153 % ^^A}
154 % ^^A\cs_generate_variant:Nn \erw_prop_put:NN { cc }
155 % ^^A\cs_new_protected:Nn\erw_prop_put:Nnn
156 % ^^A{

```

```

157 % ^^A \prop_if_exist:NTF#1
158 % ^^A {
159 % ^^A \prop_put:Nnn #1 {#2}{#3}
160 % ^^A }
161 % ^^A {
162 % ^^A \prop_new:N #1
163 % ^^A \erw_prop_put:Nnn #1{#2}{#3}
164 % ^^A }
165 % ^^A}
166 % ^^A\cs_generate_variant:Nn \erw_prop_put:Nnn { c }
167 % ^^A\cs_new_protected:Nn\erw_prop_keyval_parse_key:Nnn
168 % ^^A{
169 % ^^A \__erw_prop_keyval_parse:NNNn
170 % ^^A #1
171 % ^^A #2
172 % ^^A \erw_keyval_keyonly:nn
173 % ^^A {#3}
174 % ^^A}
175 % ^^A\cs_new_protected:Nn \erw_prop_keyval_parse:NNn
176 % ^^A{
177 % ^^A \__erw_prop_keyval_parse:NNNn
178 % ^^A #1
179 % ^^A \erw_keyval_error:n
180 % ^^A #2
181 % ^^A {#3}
182 % ^^A}
183 \cs_new_protected:Nn\erw_prop_keyval_parse:NNNn
184 {
185 \prop_if_exist:NTF#1
186 {\erw_keyval_parse:NNNn #1#2#3{#4}}
187 {
188 \prop_new:N #1
189 \erw_prop_keyval_parse:NNNn#1#2#3{#4}
190 }
191 }
192 % ^^A \cs_set_protected:Nn \__erw_keyval_function:n {#2 #1{##1}}
193 % ^^A \cs_set_protected:Nn \__erw_keyval_function:nn {#3 #1{##1}{##2}}
194 % ^^A \prop_if_exist:NTF#1
195 % ^^A {
196 % ^^A \keyval_parse:NNn
197 % ^^A \__erw_keyval_function:n
198 % ^^A \__erw_keyval_function:nn
199 % ^^A {#4}
200 % ^^A }
201 % ^^A {
202 % ^^A \prop_new:N #1
203 % ^^A \__erw_prop_fun_keyval:NNNn #1#2#3{#4}
204 % ^^A }
205 % ^^A}
206 % ^^A%^^A\cs_new_protected:Nn\erw_prop_gput_keyval:Nn
207 % ^^A{
208 % ^^A \prop_if_exist:NTF#1
209 % ^^A {
210 % ^^A \erw_prop_keyval_parse:NNNn

```

```

211 % ^^A      #1
212 % ^^A      \erw_keyval_error:n
213 % ^^A      \prop_gput:Nnn
214 % ^^A      {#2}
215 % ^^A    }
216 % ^^A}
217 % ^^A\cs_generate_variant:Nn \erw_prop_put_keyval:Nn { c }

```

## 10 oper

### 10.1 backend

```

218 \cs_new:Nn \__erw_oper_compose:NnN
219 {
220   \erw_cs_set_inline:Nn \g__erw_tl_function:n
221   {
222     #1{##1}#3
223   }
224   \exp_args:Nf\erw_tl_map:n
225   {
226     \tl_reverse:n{#2}
227   }
228 }

```

### 10.2 frontend

```

229 \keys_define:nn{__erw}
230 {
231   oper/fold_set_par.tl_gset:N = \g__erw_oper_fold_set_par_tl,
232   oper/fold_set_par.value_required:n = true,
233   oper/fold_set_par.default:n = {Nf},
234   oper/fold_set_par.initial:n = {Nf},
235   oper/fold_apply_par.tl_gset:N = \g__erw_oper_fold_apply_par_tl,
236   oper/fold_apply_par.value_required:n = true,
237   oper/fold_apply_par.default:n = {Nf},
238   oper/fold_apply_par.initial:n = {Nf}
239 }

```

## 11 seq

### 11.1 backend

```

240 \tl_new:N \g__erw_seq_fold_item_tl
241 \cs_new_protected:Nn\__erw_seq_put_right_clist:Nn
242 {
243   \cs_set_protected:Nn \__erw_function:n
244   {
245     \seq_put_right:Nn #1{##1}
246   }
247   \keyval_parse:NNn
248   \__erw_function:n
249   \erw_keyval_keyonly:nn
250   {#2}
251 }
252 \cs_generate_variant:Nn \__erw_seq_put_right_clist:Nn { c }

```

```

253 \cs_new_protected:Nn\__erw_seq_put_right_prop:NNn
254 {
255   \__erw_seq_put_right_clist:Nn #1
256   {\erw_prop_to_clist:Nn #2 {#3}}
257 }
258 \cs_generate_variant:Nn \__erw_seq_put_right_prop:NNn { cc }

```

## 11.2 frontend

```

259 \cs_new:Nn \erw_seq_compose:nN
260 {
261   \__erw_oper_compose:NnN \__erw_seq_fold:NN {#1} #2
262 }
263 \cs_new:Nn \erw_seq_compose_c:nN
264 {
265   \__erw_oper_compose:NnN \__erw_seq_fold:cN {#1} #2
266 }
267 \cs_new:Nn \erw_seq_compose_vers:nN
268 {
269   \msg_error:nnn{\__erw}{csnset}{\erw_seq_compose_vers:nN}
270 }
271 \cs_new_protected:Nn\erw_seq_put_right_clist:Nn
272 {
273   \seq_if_exist:NTF#1
274   {\__erw_seq_put_right_clist:Nn#1{#2}}
275   {\seq_new:N#1\erw_seq_put_right_clist:Nn#1{#2}}
276 }
277 \cs_generate_variant:Nn \erw_seq_put_right_clist:Nn { c }
278 \cs_new_protected:Nn\erw_seq_put_right_prop:NNn
279 {
280   \seq_if_exist:NTF#1
281   {\__erw_seq_put_right_prop:NNn#1#2{#3}}
282   {\seq_new:N#1\erw_seq_put_right_prop:NNn#1#2{#3}}
283 }
284 \cs_generate_variant:Nn \erw_seq_put_right_prop:NNn { cc }
285 % ^^A\cs_new_protected:Nn\erw_seq_put_right:Nn
286 % ^^A{
287 % ^^A \seq_if_exist:NTF#1
288 % ^^A {\seq_put_right:Nn#1{#2}}
289 % ^^A {\seq_new:N#1\erw_seq_put_right:Nn #1{#2}}
290 % ^^A}
291 % ^^A\cs_generate_variant:Nn\erw_seq_put_right:Nn { c }
292 \cs_new:Nn \__erw_seq_fold:NN
293 {
294   \seq_get_right:NN #2 \g__erw_seq_fold_item_tl
295   \erw_tl_fold:NN #1 \g__erw_seq_fold_item_tl
296   \seq_put_right:No #2 {\g__erw_seq_fold_item_tl}
297 }
298 \cs_generate_variant:Nn \__erw_seq_fold:NN { cN }
299 \cs_new:Nn \erw_seq_use:Nn
300 {
301   \exp_last_unbraced:NNf
302   \seq_use:Nnnn #1
303   \erw_tl_separators:n{#2}
304 }

```

## 12 sys

### 12.1 backend

```
__erw_sys_date:N
__erw_sys_date_dec:
__erw_sys_date_hex:
305 \cs_new:Nn __erw_sys_date_dec:
306 {
307   \int_eval:n
308   {
309     \c_sys_year_int * 10000
310     +\c_sys_month_int * 100
311     +\c_sys_day_int * 1
312   }
313 }
314 \cs_new:Nn __erw_sys_date:N{\int_to_hex:n{__erw_sys_date_dec:}}
315 \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
316 \cs_new:Nn __erw_sys_time_dec:
317 {
318   \int_eval:n
319   {
320     \c_sys_hour_int * 100
321     +\c_sys_minute_int * 1
322   }
323 }
324 \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
325 \cs_new:Nn __erw_sys_datetime_base:n
326 {
327   \int_case:nnTF{#1}
328   {
329     {10}{dec}
330     {16}{hex}
331   }
332   {\c_empty_tl}
333   {\msg_error:nnn{__erw}{timestamp / base}{__erw_sys_datetime_base:n{#1}}}
334 }
335 \cs_new:Nn __erw_sys_datetime_join:nn{\erw_tl_join:nnn{#1}{\g__erw_sys_timestamp_delim_str}{#2}}
336 \cs_new:Nn __erw_sys_datetime_period:n
337 {
338   \str_case:nnTF{#1}
339   {
340     {date}{date}
341     {time}{time}
342     {datetime}{datetime}
343   }
344   {\c_empty_tl}
345   {\msg_error:nnn{__erw}{timestamp / period}{__erw_sys_datetime_period:n{#1}}}
```

```

346 }
347 \cs_new:Nn\__erw_sys_datetime_dec: {\__erw_sys_datetime_join:nn{\__erw_sys_date_dec:}{\__erw_
348 \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:

```

349 \cs_new:Nn\__erw_sys_jobnametimestamp_prefix:
350 {
351   \erw_tl_join:nn
352   {\c_sys_jobname_str}
353   {\g__erw_sys_timestamp_delim_str}
354 }

(End definition for \__erw_sys_jobnametimestamp_prefix:.)

```

\\_\_erw\_sys\_jobnametimestamp:n

```

\__erw_sys_jobnametimestamp:
355 \cs_new:Nn\__erw_sys_jobnametimestamp:nn
356 {
357   \erw_tl_join:nn
358   {\__erw_sys_jobnametimestamp_prefix:}
359   {\erw_sys_timestamp:nn{#1}{#2}}
360 }
361 \cs_new:Nn\__erw_sys_jobnametimestamp:
362 {
363   \erw_tl_join:nn
364   {\__erw_sys_jobnametimestamp_prefix:}
365   {\erw_sys_timestamp:}
366 }

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

```

\\_\_erw\_sys\_timestamp:nn

```

367 \cs_new:Nn\__erw_sys_timestamp:nn
368 {
369   \exp_args:No
370   \use:c{\__erw_sys\___erw_sys_datetime_period:n{#1}\__erw_sys_datetime_base:n{#2}:}
371 }
372 \cs_new_protected:Nn \__erw_sys_set_delim:nn
373 {
374   \use:c{tl_gset:N#1}
375   \g__erw_sys_timestamp_delim_str{#2}
376 }

(End definition for \__erw_sys_timestamp:nn.)

```

```

377 \keys_define:nn{\__erw}
378 {
379   sys / timestamp_delim .code:n =
380   {
381     \exp_last_unbraced:No
382     \__erw_sys_set_delim:nn{n}{#1}
383   },
384   sys / timestamp_delim .value_required:n = true,
385   sys / timestamp_delim .default:n = {-},
386   sys / timestamp_delim .initial:n = {-}

```

```

387 }
388 % \subsection{frontend}
389 % \begin{macrocode}
390 \cs_new:Nn\erw_sys_jobnametimestamp:nn{\__erw_sys_jobnametimestamp:nn{#1}{#2}}
391 \cs_new:Nn\erw_sys_jobnametimestamp:{\__erw_sys_jobnametimestamp:}
392 \cs_new:Nn\erw_sys_timestamp_delimiter:
393 {
394   \use:N \g__erw_sys_timestamp_delim_str
395 }
396 \cs_new:Nn\erw_sys_timestamp:nn
397 {
398   \__erw_sys_timestamp:nn{#1}{#2}
399 }
400 \cs_new:Nn\erw_sys_timestamp:
401 {
402   \__erw_sys_timestamp:nn{datetime}{16}
403 }

```

## 13 tl

### 13.1 backend

```

404 \tl_new:N \g__erw_tl_compose_tl

\g__erw_tl_function:n

405 \cs_new_protected:Nn \g__erw_tl_function:n
406 {
407   \msg_error:nnn
408   {erw}
409   {csnset}
410   {\g__erw_tl_function:n}
411 }

(End definition for \g__erw_tl_function:n.)

\__erw_map:nn

412 \cs_set_protected:Nn \__erw_map:nn
413 {
414   \quark_if_recursion_tail_stop:n{#1}
415   \g__erw_tl_function:n{#1} \__erw_map:nn{#2}
416 }

(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

417 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnn
418 {
419   #1
420   {\exp_args:Nf\tl_item:nn {#3} {#2} }
421 }
422 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnn
423 {
424   #1
425   {\exp_args:Nf\tl_item:nn {#3} {#2} }

```

```

426   {\exp_args:Nf\tl_item:nn {#4} {#2} }
427 }
428 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnnn
429 {
430   #1
431   {\exp_args:Nf\tl_item:nn {#3} {#2} }
432   {\exp_args:Nf\tl_item:nn {#4} {#2} }
433   {\exp_args:Nf\tl_item:nn {#5} {#2} }
434 }
435 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnnn
436 {
437   #1
438   {\exp_args:Nf\tl_item:nn {#3} {#2} }
439   {\exp_args:Nf\tl_item:nn {#4} {#2} }
440   {\exp_args:Nf\tl_item:nn {#5} {#2} }
441   {\exp_args:Nf\tl_item:nn {#6} {#2} }
442 }

```

(End definition for \\_\_erw\_tl\_map\_thread\_at:Nnn and others.)

```

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

443 \cs_new:Nn \__erw_tl_separators:nn
444 {
445   \int_case:nnTF {#1}
446   {
447     {1}
448     { \prg_replicate:nn{ 3 }{#2} }
449     {2}
450     {
451       { \use_ii:nn #2 }
452       { \use_i:nn #2 }
453       { \use_i:nn #2 \use_ii:nn #2 }
454     }
455     {3}{#2}
456   }
457   { \c_empty_tl }
458   {
459     \msg_error:nnnn { __erw }
460     { separ }
461     { \exp_not:N \__erw_tl_separators:nn }
462     {#2}
463   }
464 }
465 \cs_generate_variant:Nn \__erw_tl_separators:nn { e }

```

(End definition for \\_\_erw\_tl\_separators:nn.)

## 13.2 frontend

```

466 \cs_new:Nn \erw_tl_append_item:nn
467 {
468   {#1{#2}}
469 }

```



```

470 \cs_new:Nn \erw_tl_compose:nN
471 {
472   \__erw_oper_compose:NnN \erw_tl_fold:NN {#1} #2
473 }
474 \cs_new:Nn \erw_tl_compose:nn
475 {
476   \tl_set:Nn \g__erw_tl_compose_tl {#2}
477   \erw_tl_compose:nN{#1}\g__erw_tl_compose_tl
478   \g__erw_tl_compose_tl
479 }
480 \cs_new:Nn \erw_tl_compose_c:nN
481 {
482   \__erw_oper_compose:NnN \erw_tl_fold:cN {#1} #2
483 }
484 \cs_new:Nn \erw_tl_compose_c:nn
485 {
486   \tl_set:Nn \g__erw_tl_compose_tl {#2}
487   \erw_tl_compose_c:nN{#1}\g__erw_tl_compose_tl
488   \g__erw_tl_compose_tl
489 }
490 \cs_new:Nn \erw_tl_compose_vers:nN
491 {
492   \msg_error:nnn{__erw}{csnset}{\erw_tl_compose_vers:nN}
493 }
494 \cs_new:Nn \erw_tl_compose_vers:nn
495 {
496   \erw_csint_reset:{}
497   \tl_map_function:nN{#1}\erw_csint_new:n
498   \exp_last_unbraced:Nx
499   \erw_tl_compose_c:nn
500   {{\erw_csint_names_braced:{}}}
501   {#2}
502 }
503 \cs_new:Nn \erw_tl_fold:NN
504 {
505   \use:c{tl_set:\g__erw_oper_fold_set_par_tl}
506   #2
507   {
508     \use:c{exp_args:\g__erw_oper_fold_apply_par_tl}{#1}{#2}
509   }
510 }
511 \cs_generate_variant:Nn \erw_tl_fold:NN {cN}
512 \cs_new:Nn \erw_tl_gset_function:N
513 {
514   \erw_cs_gset_eq:NN \g__erw_tl_function:n #1
515 }
516 \cs_new:Nn \erw_tl_gset_function:n
517 {
518   \erw_cs_gset_inline:Nn \g__erw_tl_function:n {#1}
519 }
520 \cs_new:Nn \erw_tl_last_item:n
521 {
522   \exp_args:Nof \tl_item:nn
523   {#1}

```

```

524     {
525         \tl_count:n{#1}
526     }
527 }
528 \cs_new_protected:Nn \erw_tl_map:n
529 {
530     \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
531 }
532 \cs_new_protected:Nn \erw_tl_map:Nn
533 {
534     \cs_set_eq:NN \g__erw_tl_function:n #1
535     \erw_tl_map:n{#2}
536 }
537 \cs_new_protected:Nn \erw_tl_map_inline:nn
538 {
539     \erw_cs_set_inline:Nn \g__erw_tl_function:n {#1}
540     \erw_tl_map:n{#2}
541 }
542 \cs_new:Nn \erw_tl_repeat:nn
543 {
544     \int_step_inline:nnnn{1}{1}{#1}{#2}
545 }
546 \cs_new:Nn \erw_tl_split:nnn
547 {
548     \tl_head:n{#1}
549     \use:c{exp_args:#3} \tl_map_inline:nn
550     {
551         \tl_tail:n
552         {
553             #1
554         }
555     }{#2##1}
556 }
557 \cs_new:Nn \erw_tl_split:nn
558 {
559     \erw_tl_split:nnn{#1}{#2}{Nf}
560 }
561 \cs_new_protected:Nn \erw_tl_map_thread_at:Nnn
562 {
563     \exp_args:Nf\int_case:nnTF
564     {
565         \tl_count:n{#3}
566     }
567     {
568         {1}{ \__erw_tl_map_thread_at:Nnn #1{#2}#3 }
569         {2}{ \__erw_tl_map_thread_at:Nnnn #1{#2}#3 }
570         {3}{ \__erw_tl_map_thread_at:Nnnnn #1{#2}#3 }
571         {4}{ \__erw_tl_map_thread_at:Nnnnnn #1{#2}#3 }
572     }
573     {
574         % Do nothing
575     }
576     {
577         \msg_error:nnn{__erw}

```

```

578     {generic}
579     {erw_tl_map_thread_at:~count~of~#3~not~withing~1~to~4}
580   }
581 }
582 \cs_new_protected:Nn \erw_tl_map_thread:Nn
583 {
584   \int_step_inline:nn
585   {
586     \exp_args:Nf \tl_count:n{ \tl_head:n{#2} }
587   }
588   {
589     \erw_tl_map_thread_at:Nnn #1 {##1} {#2}
590   }
591 }
592 \cs_new:Nn \erw_tl_separators:n
593 {
594   \__erw_tl_separators:en{ \tl_count:n{#1} }{#1}
595 }

```

## 14 option

```

596 \cs_new_protected:Nn\erw_option:n
597 {
598   \keys_set:nn{__erw}{#1}
599 }

```

## 15 Closing

```

600 \ExplSyntaxOff
601 \</package>

```