

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

Erwann Rogard<sup>†</sup>

Released 2020/06/03

## 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>keys</code>	5
6	<code>lambda</code>	5
7	<code>option</code>	5
8	<code>prop</code>	5
9	<code>seq</code>	6
10	<code>sys</code>	6
11	<code>tl</code>	6
<b>II</b>	<b>Listing</b>	<b>8</b>

---

<sup>\*</sup>This file describes version v3.0, last revised 2020/06/03.

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

<b>1</b>	<b>constants</b>	<b>8</b>
1.	.....	8
<b>2</b>	<b>cs</b>	<b>8</b>
2.	.....	8
3.	.....	8
<b>3</b>	<b>csint</b>	<b>9</b>
4.	.....	9
<b>4</b>	<b>int</b>	<b>9</b>
5.	.....	9
<b>5</b>	<b>lambda</b>	<b>9</b>
6.	.....	9
<b>6</b>	<b>prop</b>	<b>10</b>
7.	.....	10
8.	.....	10
9.	.....	10
<b>7</b>	<b>seq</b>	<b>10</b>
10.	.....	10
11.	.....	11
12.	.....	11
13.	.....	11
<b>8</b>	<b>sys</b>	<b>12</b>
14.	.....	12
15.	.....	12
<b>9</b>	<b>tl</b>	<b>13</b>
16.	.....	13
17.	.....	13
18.	.....	13
19.	.....	13
20.	.....	14
21.	.....	14
<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>5</b>	<b>To do</b>	<b>17</b>
	<b>Change History</b>	<b>17</b>
	<b>Index</b>	<b>19</b>
<b>IV</b>	<b>Implementation</b>	<b>22</b>
<b>1</b>	<b>Opening</b>	<b>22</b>
<b>2</b>	<b>cs</b>	<b>22</b>
	2.1 backend . . . . .	22
	2.2 frontend . . . . .	22
<b>3</b>	<b>csint</b>	<b>23</b>
	3.1 backend . . . . .	23
	3.2 frontend . . . . .	23
<b>4</b>	<b>int</b>	<b>24</b>
	4.1 backend . . . . .	24
	4.2 frontend . . . . .	24
<b>5</b>	<b>keys</b>	<b>25</b>
	5.1 frontend . . . . .	25
<b>6</b>	<b>lambda</b>	<b>25</b>
<b>7</b>	<b>msg</b>	<b>25</b>
	7.1 backend . . . . .	25
	7.2 frontend . . . . .	26
<b>8</b>	<b>prop</b>	<b>26</b>
	8.1 backend . . . . .	26
	8.2 frontend . . . . .	26
<b>9</b>	<b>oper</b>	<b>27</b>
	9.1 backend . . . . .	27
	9.2 frontend . . . . .	27
<b>10</b>	<b>option</b>	<b>27</b>
<b>11</b>	<b>seq</b>	<b>27</b>
	11.1 backend . . . . .	27
	11.2 frontend . . . . .	28
<b>12</b>	<b>sys</b>	<b>28</b>
	12.1 backend . . . . .	28

<b>13</b>	<b>tl</b>	<b>31</b>
13.1	backend	31
13.2	frontend	32
<b>14</b>	<b>Closing</b>	<b>34</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_compose:NnN</code>	<code>\erw_cs_compose:NnN&lt;cs&gt;{\&lt;items&gt;}\&lt;tl var&gt;</code>
----------------------------------	---

---



---

<code>\erw_cs_identity:n</code>	<code>\erw_cs_identity:n{\&lt;arg&gt;}</code>
---------------------------------	---

---



---

<code>\erw_cs_set_inline:Nn</code>	<code>\erw_cs_set_inline:Nn&lt;cs&gt;{\&lt;code&gt;}</code>
<code>\erw_cs_set_inline:(cn cn)</code>	
<code>\erw_cs_gset_inline:Nn</code>	

---

## 3 csint

---

<code>\erw_csint:nn</code>	<code>\erw_csint:nn{\&lt;integer&gt;}\&lt;arg&gt;}</code>
----------------------------	---

---



---

<code>\erw_csint_name:n</code>	<code>\erw_csint_name:n{\&lt;integer&gt;}</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 keys

---

<code>\erw_keyval_parse:NNNn</code>	<code>\erw_keyval_parse:NNNn &lt;container&gt;&lt;cs<sub>1</sub>&gt;&lt;cs<sub>2{&lt;token list&gt;<sub>1</sub>...}</sub></code>
-------------------------------------	--

---



---

<code>\erw_keyval_error:Nn</code>	<code>\erw_keyval_error:Nn&lt;token&gt;{&lt;keyval list&gt;}</code>
<code>\erw_keyval_error:Nnn</code>	<code>\erw_keyval_error:Nnn&lt;token&gt;{&lt;clist&gt;}</code>

---

## 6 lambda

---

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

---

## 7 option

---

<code>\erw_option:n</code>	<code>\erw_option:n{&lt;keyval list&gt;}</code>
----------------------------	---

---

tl / fold\_set\_par  
tl / fold\_apply\_par  
sys / timestamp\_delim

## 8 prop

All functions that modify a *<prop>* first create it if not exist.

---

<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></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>
------------------------------------	---

---

## 9 seq

All functions that modify a  $\langle seq \rangle$  first create it if not exists.

---

```
\erw_seq_fold:NN \erw_seq_fold:NN{\{ $\langle cs_1 \rangle$ \}...}
\erw_seq_fold:cN
```

---



---

```
\erw_seq_put_right_clist:Nn \erw_seq_put_right_clist:Nn $\langle seq \rangle$ \{ $\langle clist \rangle$ \}
\erw_seq_put_right_clist:cN
```

---



---

```
\erw_seq_put_right_prop:NNn \erw_seq_put_right_prop:NNn $\langle seq \rangle$ \(prop)\{ $\langle clist \rangle$ \}
```

---



---

```
\erw_seq_use:Nn \erw_seq_use:Nn $\langle seq \rangle$ \{ $\langle items \rangle$ \}
```

---

Also see [1, Section 8 of l3seq]

Semantics  $\backslash seq\_use:Nnn\langle seq \rangle \backslash \textcolor{red}{erw\_tl\_separators:n}\{ \langle items \rangle \}$

## 10 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:
```

---

## 11 tl

All functions that modify a  $\langle token list \rangle$  first create it if not exist.

---

```
\erw_tl_append_item:nn \erw_tl_append_item:nn{ $\langle arg list \rangle$ }\{ $\langle arg \rangle$ \}
```

---



---

```
\erw_tl_fold:NN \erw_tl_fold:NN $\langle cs \rangle$ \(tl var)
\erw_tl_fold:cN
```

---



---

```
\erw_tl_gset_function:N \erw_tl_gset_function:n{ $\langle code \rangle$ }
\erw_tl_gset_function:n
```

---

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

Listing 2.

```
\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_cs_compose:NnN \erw_tl_fold:NN{ \__baz:n}{\__bar:n}{\__foo:n}
\tl_use:N \l_tmpa_tl
\ExplSyntaxOff
```

---

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

Listing 3.

```
\ExplSyntaxOn
\tl_map_function:nN { {f(#1)} {g[#1]} {h\{#1\}} } \erw_csint_new:n
\tl_set:Nn \l_tmpa_tl{X}
\exp_args:NNx
\erw_cs_compose:NnN \erw_tl_fold:cN
{\erw_csint_names_braced:nnn{ 1 }{ 1 }{ 3 }}
\tl_tmpa_tl
\tl_use:N \l_tmpa_tl
\ExplSyntaxOff
```

---

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



### 3 csint

Listing 4.

```
\ExplSyntaxOn
\cs_set:Nn\__foo:n{ f(#1) }
\cs_set:Nn\__baz:n{ h\{#1\} }
\tl_map_function:nN { {\__foo:n} {g[#1]} {\__baz:n} }\erw_csint_new:n
\erw_csint:nn{1}{X},\
\erw_csint:nn{2}{X},\
\erw_csint:nn{3}{X}.
\erw_csint_reset:
\ExplSyntaxOff
```

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

### 4 int

Listing 5.

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

2345  
12345

### 5 lambda

Listing 6.

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

Hello, world!

## 6 prop

Listing 7.

```
\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 8.

```
\ExplSyntaxOn
\erw_prop_keyval_parse:NNNn
\foo_prop
\erw_keyval_error:Nn
\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 9.

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

a,b,c

## 7 seq

Listing 10.

```
\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_cs_compose:NNN \erw_seq_fold: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
\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_cs_compose:NnN \erw_seq_fold:cN{ \__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 12.

```

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

```

$a,b,c$

#### Listing 13.

```

\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 14.

```

\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

```

20200603-2228  
20200603-2228  
1343c9b%8b4  
1343c9b%8b4  
erw-l3%1343c9b%8b4

Listing 15.

```

\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:202006032228
D:20200603222859-04'00'
Hello, world!
```

## 9 tl

### Listing 16.

```
\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 17.

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

```
xxx
```

### Listing 18.

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

```
a==b==c
```

### Listing 19.

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

(a)(b)(c)

#### Listing 20.

```
\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 21.

```
\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>Xcommunity[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>Xengine

## 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>



## 5 To do

- a) Regression testing using [?, Section 3.2—Specifying expectations].

Also see:

- b) NOTE or \NB tagged `abandon|done|todo` inside `erw-l3.dtx`

## Change History

v1.1	General: \numbrdcsnew changed to \newnumbrdcs and made 'disambiguable' . . . . . 16	v1.5	General: Modify: source repository . . 16
	disambig/backend: changes to the key, added		Rearrange: frontend/backend sections . . . . . 16
	\ProcessPackageKeysOption; . . . 16		Remove: disambig . . . . . 16
	Brought all the modules under one file; renamed l3erw to erw-l3; . . . . 16		Split Section Preliminaries into Conventions and Requirement. . . 16
v1.2	General: disambig: \disambignewcmd no longer takes a token name as arg, rather a token. . . . . 16	v1.6	General: Fix: critical bug preventing erw-l3 from working without explicit inclusion of expl3 . . . . . 16
	disambig: pushed the code inside \keys_define; . . . . . 16	v1.7	General: (deleted) . . . . . 16
	Add: \erw_items_to . . . . . 16		Add: option . . . . . 16
	Add: \erw_last_item . . . . . 16		Add: sys . . . . . 16
	Add: \erw_repeat . . . . . 16		Move: \erw_fold_apply_par:n . . 16
	Add: \erw_split . . . . . 16		Move: \erw_fold_set_par:n . . . . 16
	Add: \map_thread . . . . . 16		Remove: document level functions, \numbrdcsnew, \numbrdcs . . . . . 16
	Front end cmds no longer generated with module disambig; Option of the same name deleted; . . . . . 16		Replace: listing's implem with that of tocloft . . . . . 16
	Modify: \erw_compose, order in which functions composed ( $g \circ f$ means $f$ comes before $g$ ) . . . . . 16		Replace: vers. numb. from 3 to 2 digits . . . . . 16
	Rearrange: the doc to clearly separate frontend from backend . . 16	v1.8	General: (deleted) . . . . . 16
v1.3	General: Replace: versioning, should have been 0.1.2 . . . . . 16		Add: function for all frontend functions. . . . . 16
v1.4	General: Add: \erw_accum . . . . . 16		Remove: \erw_cs_set_eq:NN and variants . . . . . 16
	Add: \erw_int_range . . . . . 16		Remove: \erw_is_matrix:n (predicate must be expandable) . . 16
	Add: \erw_is_matrix (to check arg of \erw_tl_map_thread:Nn) . . . . 16		Rename: all cs prefixes to agree with heading under which they come, e.g. \erw_identity:n by \erw_cs_identity:n . . . . . 16
	Add: \erw_merge . . . . . 16		Replace: \erw_seq_fold:NN by \erw_oper_fold_seq:NN and likewise for variants . . . . . 16
	Add: \erw_set_map_inline . . . . 16	v1.9	General: Add:
	Add: \erw_set_map . . . . . 16		
	Remove: \erw_items_to (redundant with \tl_range:nnn) . . . . . 16		

	\erw_sys_timestamp_delimiter: 16		Add:	
	Add: \erw_tl_join:nn and variants 16		\erw_prop_keyval_parse:NNNn .. 16	
	Rename: \erw_append_arg:nn to		Add: \erw_prop_map_item:NNN .. 16	
	\erw_tl_append_item:nn ..... 16		Add: \msg_new:nnn{erw}{varnset} 16	
	Rename:		Remove: \erw_cs_apply ..... 16	
	\erw_oper_gset_function:N to		Remove: \erw_prop_put:NN ..... 16	
	\erw_tl_gset_function:N (and		Remove:	
	variants) ..... 16		\erw_prop_put_keyval:Nn ..... 16	
v2.0	General: Add:		Remove: \msg_new:nnn, module erw,	
	\erw_jobnametimestamp:nn and		messages: keyval/... ..... 16	
	variants ..... 16		Rename: basics to cs ..... 16	
	Remove: \merge:nn (redundant		Replace: \erw_seq_from_clist by	
	with \erw_join:nn) ..... 16		\erw_seq_put_right_clist ..... 16	
	Rename: v0.0 to v1.0, etc. .... 16		Replace: \erw_seq_from_prop by	
			\erw_seq_put_right_prop ..... 16	
v2.1	General: (delete) ..... 16	v2.7	General: Add:	
	Add: \erw_prop_to_clist:Nn,		\erw_keyval_error:Nnn ..... 16	
	\erw_prop_put:NN, and		Add: \erw_keyval_error:Nn .... 16	
	\erw_prop_put:Nnn ..... 16		Remove: \erw_cs_error:nn ..... 16	
	Add: \erw_seq_from_clist:Nn,		Remove: \erw_cs_error:n ..... 16	
	\erw_seq_from_prop:NNn, and	v2.8	General: Add:	
	\erw_seq_put_right:Nn ..... 16		\msg_new:nnn{erw}{notset} .... 16	
	Replace: \erw_seq_fold:NN by		Remove:	
	\__erw_seq_fold:NN ..... 16		\msg_new:nnn{erw}{csnset} .... 16	
v2.2	General: Add: \erw_seq_use:Nn .... 16		Remove:	
	Add: \erw_tl_separators:n .... 16		\msg_new:nnn{erw}{varnset} .... 16	
v2.3	General: Add:	v2.9	General: Add: \erw_cs_compose:NnN 16	
	\msg_new:nnn{erw}{csnset} .... 16		Add: \erw_seq_fold:NN,	
	Add:		\erw_seq_fold:cN ..... 16	
	\msg_new:nnn{erw}{keyval/...} . 16		Remove: \erw_seq_-	
	Fix: 'mark as private code' (hitherto		compose:nN,\erw_seq_compose_-	
	unnoticed) ..... 16		c:nN,\erw_seq_compose_vers:nN . 16	
	Modify: behavior of		Remove: \erw_tl_compose:nN,	
	\erw_seq_use:Nn ..... 16		\erw_tl_compose:Nnn,	
	Move: all \msg_new:Nnnn statements		\erw_tl_compose:nn,	
	under same heading ..... 16		\erw_tl_compose_c:nN,	
v2.4			\erw_tl_compose_c:nn,	
	General: Add: \erw_lambda:nnn .... 16		\erw_tl_compose_vers:nN,	
v2.5	General: Add:		\erw_tl_compose_vers:nn ..... 16	
	\erw_prop_put_keyval:Nn ..... 16		Rename: oper / fold_apply_par	
v2.6	General: Add: \erw_cs_error:nn ... 16		to tl / fold_apply_par ..... 16	
	Add: \erw_cs_error:n ..... 16		Rename: oper / fold_set_par to	
	Add: \erw_keyval_parse:NNNn .. 16	v3.0	tl / fold_set_par ..... 16	
			General: Fix: warning csquotes+fvextra 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

`\begin` ..... 305

## C

cs commands:

`\cs_generate_variant:Nn` .. 24, 29,  
141, 187, 193, 200, 207, 214, 381, 394  
`\cs_gset:Npn` ..... 27  
`\cs_new:Nn` ..... 4,  
8, 25, 31, 32, 36, 49, 54, 55, 85,  
89, 95, 96, 215, 221, 230, 231, 232,  
240, 241, 251, 252, 263, 264, 265,  
271, 277, 283, 306, 307, 308, 312,  
316, 359, 382, 386, 395, 399, 403,  
411, 412, 413, 414, 429, 433, 444, 479  
`\cs_new_protected:Nn` .....  
..... 20, 37, 59, 97, 121, 129,  
142, 151, 171, 176, 188, 194, 201,  
208, 288, 321, 415, 419, 424, 448, 469  
`\cs_new_protected:Npn` ..... 106  
`\cs_set:Nn` ..... 131  
`\cs_set:Npn` ..... 19, 22, 64  
`\cs_set_eq:NN` ..... 421  
`\cs_set_protected:Nn` ..... 99,  
100, 123, 178, 328, 333, 338, 344, 351  
`\cs_split_function:N` ..... 6

## E

erw commands:

`\erw_cs_compose:NnN` ..... 4, 8  
`\erw_cs_gset_eq:NN` ..... 397  
`\erw_cs_gset_inline:Nn` . 4, 20, 25, 401  
`\erw_cs_identity:n` ..... 4, 19  
`\erw_cs_set_inline:Nn` 4, 10, 20, 41, 426  
`\erw_csint:nn` ..... 4, 32  
`\erw_csint_name:n` ... 4, 31, 34, 36, 54  
`\erw_csint_names_braced:` ..... 4, 49  
`\erw_csint_names_braced:n` .... 4, 49  
`\erw_csint_names_braced:nnn` .. 4, 49  
`\erw_csint_new:n` ..... 4, 37  
`\erw_csint_reset:` ..... 4, 59  
`\erw_int_range:n` ..... 5, 85  
`\erw_int_range:nn` ..... 5, 85  
`\erw_keyval_error:Nn` ..... 5, 95  
`\erw_keyval_error:Nnn` .... 5, 95, 137  
`\erw_keyval_keyonly:nn` ..... 184  
`\erw_keyval_parse:NNNn` .... 5, 97, 154  
`\erw_lambda:nnn` ..... 5, 106

`\erw_option:n` ..... 5, 171  
`\erw_prop_keyval_parse:NNNn` . 5, 151  
`\erw_prop_map_item:NNN` ..... 5, 142  
`\erw_prop_to_clist:Nn` ... 5, 129, 191  
`\erw_seq_fold:NN` ..... 6, 208, 214  
`\erw_seq_put_right_clist:Nn` ....  
..... 6, 194, 198, 200  
`\erw_seq_put_right_prop:NNn` ....  
..... 6, 201, 205, 207  
`\erw_seq_use:Nn` ..... 6, 215  
`\erw_sys_jobnametimestamp:` .. 6, 307  
`\erw_sys_jobnametimestamp:nn` 6, 306  
`\erw_sys_timestamp:` ..... 6, 281, 316  
`\erw_sys_timestamp:nn` ... 6, 275, 312  
`\erw_sys_timestamp_delimiter:` 6, 308  
`\erw_tl_append_item:nn` .... 6, 76, 382  
`\erw_tl_fold:NN` .... 6, 211, 386, 394  
`\erw_tl_gset_function:N` ..... 6, 395  
`\erw_tl_gset_function:n` .... 6, 7, 399  
`\erw_tl_join:nn` . 7, 267, 273, 279, 411  
`\erw_tl_join:nnn` ..... 7, 251, 411  
`\erw_tl_join:nnnn` ..... 7, 411  
`\erw_tl_join:nnnnn` ..... 7, 411  
`\erw_tl_last_item:n` ..... 7, 403  
`\erw_tl_map:n` .... 7, 14, 415, 422, 427  
`\erw_tl_map:Nn` ..... 7, 419  
`\erw_tl_map_inline:nn` ..... 7, 424  
`\erw_tl_map_thread:Nn` ..... 7, 469  
`\erw_tl_map_thread_at:Nnn` 7, 448, 476  
`\erw_tl_math_thread:Nn` ..... 7  
`\erw_tl_math_thread_at:Nnn` ..... 7  
`\erw_tl_repeat:nn` ..... 7, 429  
`\erw_tl_separators:n` .. 6, 7, 219, 479  
`\erw_tl_split:nn` ..... 7, 444  
`\erw_tl_split:nnn` ..... 7, 433, 446

erw internal commands:

`\__erw_cs_name:N` ..... 4  
`\__erw_csint_ext_tl` ..... 62  
`\g_erw_csint_int` .. 30, 31, 39, 57, 61  
`\__erw_csint_name:` ..... 31, 41  
`\__erw_function:n` ..... 178, 183  
`\__erw_function:nn` ..... 123, 127  
`\__erw_int_range:nnn` .. 64, 74, 87, 91  
`\__erw_keyval_function:n` .....  
..... 99, 102, 131, 136  
`\__erw_keyval_function:nn` .. 100, 103  
`\__erw_lambda_expression` ... 109, 112  
`\__erw_prop_map_item:NNN` ... 121, 145

\g__erw_seq_fold_item_tl .....	175, 210, 211, 212
\__erw_seq_put_right_clist:Nn ...	176, 187, 190, 197
\__erw_seq_put_right_prop:NNn ...	188, 193, 204
\__erw_sys_date:N .....	221
\__erw_sys_date_dec: .....	221, 263
\__erw_sys_date_hex: .....	221, 264
\__erw_sys_datetime_base:n .	241, 286
\__erw_sys_datetime_dec: .....	263
\__erw_sys_datetime_dec:n .....	241
\__erw_sys_datetime_hex: .....	264
\__erw_sys_datetime_hex:n .....	241
\__erw_sys_datetime_join:nn ...	241
\__erw_sys_datetime_period:n	241, 286
\__erw_sys_jobnametimestamp: ..	271, 307
\__erw_sys_jobnametimestamp:n ..	271
\__erw_sys_jobnametimestamp:nn ..	271, 306
\__erw_sys_jobnametimestamp_- prefix: .....	265, 274, 280
\__erw_sys_set_delim:nn ....	288, 298
\__erw_sys_time_dec: .....	232, 263
\__erw_sys_time_hex .....	232
\__erw_sys_time_hex: .....	240, 264
\__erw_sys_timestamp:nn	283, 314, 318
\g__erw_sys_timestamp_delim_str .	251, 269, 291, 310
\g__erw_tl_compose_tl .....	320
\g__erw_tl_fold_apply_par_tl	166, 391
\g__erw_tl_fold_set_par_tl .	162, 388
\__erw_tl_map:nn .....	328, 417
\__erw_tl_map_thread_at:Nnn	333, 455
\__erw_tl_map_thread_at:Nnnn	333, 456
\__erw_tl_map_thread_at:Nnnnn ..	333, 457
\__erw_tl_map_thread_at:Nnnnnn ..	333, 458
\__erw_tl_separators:nn ....	359, 481
exp commands:	
\exp_args:Nf .....	14, 132, 336, 341, 342, 347, 348, 349, 354, 355, 356, 357, 450, 473
\exp_args:NNx .....	108
\exp_args:No .....	34, 40, 285
\exp_args:Nof .....	405
\exp_args:Nx .....	76
\exp_last_unbraced:Nf .....	6
\exp_last_unbraced:NNf .....	217
\exp_last_unbraced:No .....	297
\ExplSyntaxOff .....	483
\ExplSyntaxOn .....	3
	<b>G</b>
g internal commands:	
\g__erw_tl_function:n .....	10, 321, 331, 397, 401, 421, 426
	<b>I</b>
int commands:	
\int_case:nnTF .....	243, 361, 450
\int_compare:nNnTF .....	66
\int_eval:n .....	68, 78, 81, 223, 234
\int_incr:N .....	39
\int_new:N .....	30
\int_step_function:nnnN .....	51
\int_step_inline:nn .....	93, 471
\int_step_inline:nnnn .....	431
\int_to_alph:n .....	36
\int_to_hex:n .....	230, 231, 240
\int_zero:N .....	61
	<b>K</b>
keys commands:	
\keys_define:nn .....	160, 293
\keys_set:nn .....	173
keyval commands:	
\keyval_parse:NNn .....	101, 135, 182
	<b>M</b>
msg commands:	
\msg_error:nnn ....	249, 261, 323, 464
\msg_error:nnnn .....	375
\msg_error:nnnnn .....	95
\msg_error:nnnnnn .....	96
\msg_new:nnn .....	114, 115, 116, 117, 118, 119, 120
	<b>O</b>
options:	
sys / timestamp_delim .....	5
tl / fold_apply_par .....	5
tl / fold_set_par .....	5
	<b>P</b>
prg commands:	
\prg_replicate:nn .....	364
prop commands:	
\prop_if_exist:NTF .....	144, 153
\prop_item:Nn .....	131
\prop_map_function:NN .....	127
\prop_new:N .....	147, 156
	<b>Q</b>
quark commands:	
\quark_if_recursion_tail_stop:n	330
\q_recursion_stop .....	417
\q_recursion_tail .....	417

S		tl commands:	
seq commands:		\c_empty_tl	248, 260, 373
\seq_get_right:NN	210	\tl_count:n	408, 452, 473, 481
\seq_if_exist:NTF	196, 203	\tl_head:n	435, 473
\seq_new:N	198, 205	\tl_item:nn	336, 341, 342,
\seq_put_right:Nn	180, 212		347, 348, 349, 354, 355, 356, 357, 405
\seq_use:Nnnn	6, 218	\tl_map_inline:nn	436
str commands:		\tl_new:N	175, 320
\str_case:nnTF	254	\tl_range_braced:nnn	52
\subsection	304	\tl_reverse:n	16
sys / timestamp_delim (option)	5	\tl_set:Nn	62
sys commands:		\tl_tail:n	133, 438
\c_sys_day_int	227	token commands:	
\c_sys_hour_int	236	\token_if_cs:NTF	43
\c_sys_jobname_str	268		
\c_sys_minute_int	237	U	
\c_sys_month_int	226	use commands:	
\c_sys_year_int	225	\use:N	34, 286, 290, 310, 388, 391, 436
		\use_i:nn	368, 369
		\use_i:nnn	6
		\use_ii:nn	367, 369
		\usepackage	4
T			
tl / fold_apply_par (option)	5		
tl / fold_set_par (option)	5		

# 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

**\erw\_cs\_compose:NnN**

```

8 \cs_new:Nn \erw_cs_compose:NnN
9 {
10   \erw_cs_set_inline:Nn \g__erw_tl_function:n
11   {
12     #1{##1}#3
13   }
14   \exp_args:Nf\erw_tl_map:n
15   {
16     \tl_reverse:n{#2}
17   }
18 }

```

(End definition for \erw\_cs\_compose:NnN. This function is documented on page 4.)

**\erw\_cs\_identity:n**

```

19 \cs_set:Npn \erw_cs_identity:n #1{#1}

```

(End definition for \erw\_cs\_identity:n. This function is documented on page 4.)

**\erw\_cs\_set\_inline:Nn**

**\erw\_cs\_gset\_inline:Nn**

```

20 \cs_new_protected:Nn \erw_cs_set_inline:Nn
21 {
22   \cs_set:Npn #1 ##1{#2}
23 }
24 \cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}

```

(End definition for \erw\_cs\_set\_inline:Nn and \erw\_cs\_gset\_inline:Nn. These functions are documented on page 4.)

`\erw_cs_gset_inline:Nn`

```

25 \cs_new:Nn \erw_cs_gset_inline:Nn
26 {
27   \cs_gset:Npn #1 ##1{#2}
28 }
29 \cs_generate_variant:Nn \erw_cs_gset_inline:Nn {cn}

```

(End definition for `\erw_cs_gset_inline:Nn`. This function is documented on page 4.)

## 3 csint

### 3.1 backend

```

30 \int_new:N \g__erw_csint_int
31 \cs_new:Nn \__erw_csint_name: {\erw_csint_name:n{\g__erw_csint_int}}

```

### 3.2 frontend

`\erw_csint:nn`

```

32 \cs_new:Nn \erw_csint:nn
33 {
34   \exp_args:No \use:c{\erw_csint_name:n{#1}}{#2}
35 }

```

(End definition for `\erw_csint:nn`. This function is documented on page 4.)

`\erw_csint_name:n`

```

36 \cs_new:Nn \erw_csint_name:n {\__erw_csint\_int_to_alph:n{#1}:n}

```

(End definition for `\erw_csint_name:n`. This function is documented on page 4.)

`\erw_csint_new:n`

```

37 \cs_new_protected:Nn \erw_csint_new:n
38 {
39   \int_incr:N \g__erw_csint_int
40   \exp_args:No
41   \erw_cs_set_inline:cn{\__erw_csint_name:}
42   {
43     \token_if_cs:NTF
44     {#1}
45     {#1{##1}}
46     {#1}
47   }
48 }

```

(End definition for `\erw_csint_new:n`. This function is documented on page 4.)

`\erw_csint_names_braced:nnn`

`\erw_csint_names_braced:n`

`\erw_csint_names_braced:`

```

49 \cs_new:Nn \erw_csint_names_braced:nnn
50 {
51   \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_csint_names_braced:n
52   % TODO \tl_range_braced:nnn?
53 }
54 \cs_new:Nn \erw_csint_names_braced:n {\erw_csint_name:n{#1}}

```

```

55 \cs_new:Nn \erw_csint_names_braced:
56 {
57   \erw_csint_names_braced:nnn{1}{1}{\g__erw_csint_int}
58 }

```

(End definition for `\erw_csint_names_braced:nnn`, `\erw_csint_names_braced:n`, and `\erw_csint_names_braced:..` These functions are documented on page 4.)

`\erw_csint_reset:`

```

59 \cs_new_protected:Nn \erw_csint_reset:
60 {
61   \int_zero:N \g__erw_csint_int
62   \tl_set:Nn \__erw_csint_ext_tl{}%^^A TODO remove?
63 }

```

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

## 4 int

### 4.1 backend

```

64 \cs_set:Npn \__erw_int_range:nnn #1 #2 #3
65 {
66   \int_compare:nNnTF
67   {
68     \int_eval:n{#2+1}
69   }>{#3}
70   {
71     {#1}
72   }
73   {
74     \__erw_int_range:nnn
75     {
76       \exp_args:Nx\erw_tl_append_item:nn{#1}
77       {
78         \int_eval:n{#2+1}
79       }
80     }
81     {\int_eval:n{#2+1}}
82     {#3}
83   }
84 }

```

### 4.2 frontend

`\erw_int_range:nn`

`\erw_int_range:n`

```

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

```



```

93 % ^^A \int_step_inline:nn {#1}{##1}
94 }

```

(End definition for `\erw_int_range:nn` and `\erw_int_range:n`. These functions are documented on page 5.)

## 5 keys

### 5.1 frontend

```

\erw_keyval_error:Nn
\erw_keyval_error:Nnn

```

```

95 \cs_new:Nn \erw_keyval_error:Nn{\msg_error:nnnnn{__erw}{keyval/n}{\erw_keyval_error:Nn}{#1}{#2}}
96 \cs_new:Nn \erw_keyval_error:Nnn{\msg_error:nnnnnn{__erw}{keyval/nn}{\erw_keyval_error:Nnn}{#1}{#2}{#3}}

```

(End definition for `\erw_keyval_error:Nn` and `\erw_keyval_error:Nnn`. These functions are documented on page 5.)

```

\erw_keyval_parse:NNNn

```

```

97 \cs_new_protected:Nn\erw_keyval_parse:NNNn
98 {
99   \cs_set_protected:Nn \__erw_keyval_function:n {#2 #1{##1}}
100   \cs_set_protected:Nn \__erw_keyval_function:nn {#3 #1{##1}{##2}}
101   \keyval_parse:NNn
102   \__erw_keyval_function:n
103   \__erw_keyval_function:nn
104   {#4}
105 }

```

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

## 6 lambda

```

\erw_lambda:nnn

```

```

106 \cs_new_protected:Npn \erw_lambda:nnn #1 #2 #3
107 {
108   \exp_args:NNx
109   #1 \__erw_lambda_expression
110   {#2}
111   {#3}
112   \__erw_lambda_expression
113 }

```

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

## 7 msg

### 7.1 backend

```

114 \msg_new:nnn{__erw}{generic}{#1}
115 \msg_new:nnn{__erw}{keyval/nn}{#1#2{#3}{#4}};~encountered~key=val~where~only~key~required}
116 \msg_new:nnn{__erw}{keyval/n}{#1#2{#3}};~encountered~key~~where~only~key=val~required}
117 \msg_new:nnn{__erw}{separ}{#1~expects~1~to~3~items,~#2}

```

```

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

```

## 7.2 frontend

```

120 \msg_new:nnn{erw}{notset}{#1~not~set}

```

## 8 prop

### 8.1 backend

```

121 \cs_new_protected:Nn \__erw_prop_map_item:NNN
122 {
123   \cs_set_protected:Nn \__erw_function:nn
124   {
125     #1 #2 {##1}{##2}
126   }
127   \prop_map_function:NN #3 \__erw_function:nn
128 }

```

### 8.2 frontend

**\erw\_prop\_to\_clist:Nn**

```

129 \cs_new_protected:Nn \erw_prop_to_clist:Nn
130 {
131   \cs_set:Nn \__erw_keyval_function:n {,\prop_item:Nn#1{##1}}
132   \exp_args:Nf
133   \tl_tail:n
134   {
135     \keyval_parse:NNn
136     \__erw_keyval_function:n
137     \erw_keyval_error:Nnn
138     {#2}
139   }
140 }
141 \cs_generate_variant:Nn \erw_prop_to_clist:Nn { c }

```

(End definition for \erw\_prop\_to\_clist:Nn. This function is documented on page 5.)

**\erw\_prop\_map\_item:NNN**

```

142 \cs_new_protected:Nn \erw_prop_map_item:NNN
143 {
144   \prop_if_exist:NTF #2
145   {\__erw_prop_map_item:NNN #1#2#3}
146   {
147     \prop_new:N #2
148     \erw_prop_map_item:NNN #1#2#3
149   }
150 }

```

(End definition for \erw\_prop\_map\_item:NNN. This function is documented on page 5.)

**\erw\_prop\_keyval\_parse:NNNn**

```

151 \cs_new_protected:Nn \erw_prop_keyval_parse:NNNn
152 {
153   \prop_if_exist:NTF#1

```

```

154   {\erw_keyval_parse:NNNn #1#2#3{#4}}
155   {
156     \prop_new:N #1
157     \erw_prop_keyval_parse:NNNn#1#2#3{#4}
158   }
159 }

```

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

## 9 oper

### 9.1 backend

### 9.2 frontend

```

160 \keys_define:nn{__erw}
161 {
162   tl/fold_set_par.tl_gset:N = \g__erw_tl_fold_set_par_tl,
163   tl/fold_set_par.value_required:n = true,
164   tl/fold_set_par.default:n = {Nf},
165   tl/fold_set_par.initial:n = {Nf},
166   tl/fold_apply_par.tl_gset:N = \g__erw_tl_fold_apply_par_tl,
167   tl/fold_apply_par.value_required:n = true,
168   tl/fold_apply_par.default:n = {Nf},
169   tl/fold_apply_par.initial:n = {Nf}
170 }

```

## 10 option

```

171 \cs_new_protected:Nn\erw_option:n
172 {
173   \keys_set:nn{__erw}{#1}
174 }

```

## 11 seq

### 11.1 backend

```

175 \tl_new:N \g__erw_seq_fold_item_tl
176 \cs_new_protected:Nn\__erw_seq_put_right_clist:Nn
177 {
178   \cs_set_protected:Nn \__erw_function:n
179   {
180     \seq_put_right:Nn #1{##1}
181   }
182   \keyval_parse:NNn
183   \__erw_function:n
184   \erw_keyval_keyonly:nn
185   {#2}
186 }
187 \cs_generate_variant:Nn \__erw_seq_put_right_clist:Nn { c }
188 \cs_new_protected:Nn\__erw_seq_put_right_prop:NNn
189 {
190   \__erw_seq_put_right_clist:Nn #1

```

```

191   {\erw_prop_to_clist:Nn #2 {#3}}
192 }
193 \cs_generate_variant:Nn \__erw_seq_put_right_prop:NNn { cc }

```

## 11.2 frontend

```

194 \cs_new_protected:Nn\erw_seq_put_right_clist:Nn
195 {
196   \seq_if_exist:NTF#1
197   {\__erw_seq_put_right_clist:Nn#1{#2}}
198   {\seq_new:N#1\erw_seq_put_right_clist:Nn#1{#2}}
199 }
200 \cs_generate_variant:Nn \erw_seq_put_right_clist:Nn { c }
201 \cs_new_protected:Nn\erw_seq_put_right_prop:NNn
202 {
203   \seq_if_exist:NTF#1
204   {\__erw_seq_put_right_prop:NNn#1#2{#3}}
205   {\seq_new:N#1\erw_seq_put_right_prop:NNn#1#2{#3}}
206 }
207 \cs_generate_variant:Nn \erw_seq_put_right_prop:NNn { cc }
208 \cs_new_protected:Nn \erw_seq_fold:NN
209 {
210   \seq_get_right:NN #2 \g__erw_seq_fold_item_tl
211   \erw_tl_fold:NN #1 \g__erw_seq_fold_item_tl
212   \seq_put_right:No #2 {\g__erw_seq_fold_item_tl}
213 }
214 \cs_generate_variant:Nn \erw_seq_fold:NN {cN}
215 \cs_new:Nn \erw_seq_use:Nn
216 {
217   \exp_last_unbraced:NNf
218   \seq_use:Nnnn #1
219   \erw_tl_separators:n{#2}
220 }

```

## 12 sys

### 12.1 backend

```

__erw_sys_date:N
__erw_sys_date_dec: 221 \cs_new:Nn \__erw_sys_date_dec:
__erw_sys_date_hex: 222 {
223   \int_eval:n
224   {
225     \c_sys_year_int * 10000
226     +\c_sys_month_int * 100
227     +\c_sys_day_int * 1
228   }
229 }
230 \cs_new:Nn \__erw_sys_date:N{\int_to_hex:n{\__erw_sys_date_dec:}}
231 \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

```

```

232 \cs_new:Nn \__erw_sys_time_dec:
233 {
234   \int_eval:n
235   {
236     \c_sys_hour_int * 100
237     +\c_sys_minute_int * 1
238   }
239 }
240 \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
241 \cs_new:Nn\__erw_sys_datetime_base:n
242 {
243   \int_case:nnTF{#1}
244   {
245     {10}{dec}
246     {16}{hex}
247   }
248   {\c_empty_tl}
249   {\msg_error:nnn{\__erw}{timestamp / base}{\__erw_sys_datetime_base:n{#1}}}
250 }
251 \cs_new:Nn\__erw_sys_datetime_join:nn{\erw_tl_join:nnn{#1}{\g__erw_sys_timestamp_delim_str}{#2}}
252 \cs_new:Nn\__erw_sys_datetime_period:n
253 {
254   \str_case:nnTF{#1}
255   {
256     {date}{date}
257     {time}{time}
258     {datetime}{datetime}
259   }
260   {\c_empty_tl}
261   {\msg_error:nnn{\__erw}{timestamp / period}{\__erw_sys_datetime_period:n{#1}}}
262 }
263 \cs_new:Nn\__erw_sys_datetime_dec: {\__erw_sys_datetime_join:nn{\__erw_sys_date_dec:}{\__erw_sys_time_dec:}}
264 \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:

```

265 \cs_new:Nn\__erw_sys_jobnametimestamp_prefix:
266 {
267   \erw_tl_join:nn
268   {\c_sys_jobname_str}
269   {\g__erw_sys_timestamp_delim_str}
270 }

```

(End definition for \\_\_erw\_sys\_jobnametimestamp\_prefix:.)

\\_\_erw\_sys\_jobnametimestamp:

```

\__erw_sys_jobnametimestamp:
271 \cs_new:Nn\__erw_sys_jobnametimestamp:nn
272 {
273   \erw_tl_join:nn
274   {\__erw_sys_jobnametimestamp_prefix:}

```

```

275   {\erw_sys_timestamp:nn{#1}{#2}}
276 }
277 \cs_new:Nn\__erw_sys_jobnametimestamp:
278 {
279   \erw_tl_join:nn
280   {\__erw_sys_jobnametimestamp_prefix:}
281   {\erw_sys_timestamp:}
282 }

```

(End definition for \\_\_erw\_sys\_jobnametimestamp:n and \\_\_erw\_sys\_jobnametimestamp:.)

\\_\_erw\_sys\_timestamp:nn

```

283 \cs_new:Nn\__erw_sys_timestamp:nn
284 {
285   \exp_args:No
286   \use:c{__erw_sys\___erw_sys_datetime_period:n{#1}\__erw_sys_datetime_base:n{#2}:}
287 }
288 \cs_new_protected:Nn \__erw_sys_set_delim:nn
289 {
290   \use:c{tl_gset:N#1}
291   \g__erw_sys_timestamp_delim_str{#2}
292 }

```

(End definition for \\_\_erw\_sys\_timestamp:nn.)

```

293 \keys_define:nn{__erw}
294 {
295   sys / timestamp_delim .code:n =
296   {
297     \exp_last_unbraced:No
298     \__erw_sys_set_delim:nn{n}{#1}
299   },
300   sys / timestamp_delim .value_required:n = true,
301   sys / timestamp_delim .default:n = {-},
302   sys / timestamp_delim .initial:n = {-}
303 }
304 % \subsection{frontend}
305 % \begin{macrocode}
306 \cs_new:Nn\erw_sys_jobnametimestamp:nn{\__erw_sys_jobnametimestamp:nn{#1}{#2}}
307 \cs_new:Nn\erw_sys_jobnametimestamp:{\__erw_sys_jobnametimestamp:}
308 \cs_new:Nn\erw_sys_timestamp_delimiter:
309 {
310   \use:N \g__erw_sys_timestamp_delim_str
311 }
312 \cs_new:Nn\erw_sys_timestamp:nn
313 {
314   \__erw_sys_timestamp:nn{#1}{#2}
315 }
316 \cs_new:Nn\erw_sys_timestamp:
317 {
318   \__erw_sys_timestamp:nn{datetime}{16}
319 }

```

## 13 tl

### 13.1 backend

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

```
\g__erw_tl_function:n
```

```
321 \cs_new_protected:Nn \g__erw_tl_function:n
322 {
323   \msg_error:nnn
324   {erw}
325   {notset}
326   {\g__erw_tl_function:n}
327 }
```

*(End definition for \g\_\_erw\_tl\_function:n.)*

```
\__erw_tl_map:nn
```

```
328 \cs_set_protected:Nn \__erw_tl_map:nn
329 {
330   \quark_if_recursion_tail_stop:n{#1}
331   \g__erw_tl_function:n{#1} \__erw_tl_map:nn{#2}
332 }
```

*(End definition for \\_\_erw\_tl\_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
```

```
333 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnn
334 {
335   #1
336   {\exp_args:Nf\tl_item:nn {#3} {#2} }
337 }
338 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnn
339 {
340   #1
341   {\exp_args:Nf\tl_item:nn {#3} {#2} }
342   {\exp_args:Nf\tl_item:nn {#4} {#2} }
343 }
344 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnn
345 {
346   #1
347   {\exp_args:Nf\tl_item:nn {#3} {#2} }
348   {\exp_args:Nf\tl_item:nn {#4} {#2} }
349   {\exp_args:Nf\tl_item:nn {#5} {#2} }
350 }
351 \cs_set_protected:Nn \__erw_tl_map_thread_at:Nnnnnn
352 {
353   #1
354   {\exp_args:Nf\tl_item:nn {#3} {#2} }
355   {\exp_args:Nf\tl_item:nn {#4} {#2} }
356   {\exp_args:Nf\tl_item:nn {#5} {#2} }
357   {\exp_args:Nf\tl_item:nn {#6} {#2} }
358 }
```

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

```

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

359 \cs_new:Nn \__erw_tl_separators:nn
360 {
361   \int_case:nnTF {#1}
362   {
363     {1}
364     { \prg_replicate:nn{ 3 }{#2} }
365     {2}
366     {
367       { \use_ii:nn #2 }
368       { \use_i:nn #2 }
369       { \use_i:nn #2 \use_ii:nn #2 }
370     }
371     {3}{#2}
372   }
373   { \c_empty_tl }
374   {
375     \msg_error:nnnn { __erw }
376     { separ }
377     { \__erw_tl_separators:nn }
378     {#2}
379   }
380 }
381 \cs_generate_variant:Nn \__erw_tl_separators:nn { e }

(End definition for \__erw_tl_separators:nn.)

```

## 13.2 frontend

```

382 \cs_new:Nn \erw_tl_append_item:nn
383 {
384   {#1{#2}}
385 }
386 \cs_new:Nn \erw_tl_fold:NN
387 {
388   \use:c{tl_set:\g__erw_tl_fold_set_par_tl}
389   #2
390   {
391     \use:c{exp_args:\g__erw_tl_fold_apply_par_tl}{#1}{#2}
392   }
393 }
394 \cs_generate_variant:Nn \erw_tl_fold:NN {cN}
395 \cs_new:Nn \erw_tl_gset_function:N
396 {
397   \erw_cs_gset_eq:NN \g__erw_tl_function:n #1
398 }
399 \cs_new:Nn \erw_tl_gset_function:n
400 {
401   \erw_cs_gset_inline:Nn \g__erw_tl_function:n {#1}
402 }
403 \cs_new:Nn \erw_tl_last_item:n
404 {
405   \exp_args:Nof \tl_item:nn

```



```

406     {#1}
407     {
408         \tl_count:n{#1}
409     }
410 }

\erw_tl_join:nn
\erw_tl_join:nnn
\erw_tl_join:nnnn
\erw_tl_join:nnnnn
411 \cs_new:Nn \erw_tl_join:nn{#1#2}
412 \cs_new:Nn \erw_tl_join:nnn{#1#2#3}
413 \cs_new:Nn \erw_tl_join:nnnn{#1#2#3#4}
414 \cs_new:Nn \erw_tl_join:nnnnn{#1#2#3#4#5}

(End definition for \erw_tl_join:nn and others. These functions are documented on page 7.)

415 \cs_new_protected:Nn \erw_tl_map:n
416 {
417     \__erw_tl_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
418 }
419 \cs_new_protected:Nn \erw_tl_map:Nn
420 {
421     \cs_set_eq:NN \g__erw_tl_function:n #1
422     \erw_tl_map:n{#2}
423 }
424 \cs_new_protected:Nn \erw_tl_map_inline:nn
425 {
426     \erw_cs_set_inline:Nn \g__erw_tl_function:n {#1}
427     \erw_tl_map:n{#2}
428 }
429 \cs_new:Nn \erw_tl_repeat:nn
430 {
431     \int_step_inline:nnnn{1}{1}{#1}{#2}
432 }
433 \cs_new:Nn \erw_tl_split:nnn
434 {
435     \tl_head:n{#1}
436     \use:c{exp_args:#3} \tl_map_inline:nn
437     {
438         \tl_tail:n
439         {
440             #1
441         }
442     }{#2##1}
443 }
444 \cs_new:Nn \erw_tl_split:nn
445 {
446     \erw_tl_split:nnn{#1}{#2}{Nf}
447 }
448 \cs_new_protected:Nn \erw_tl_map_thread_at:Nnn
449 {
450     \exp_args:Nf\int_case:nnTF
451     {
452         \tl_count:n{#3}
453     }
454     {
455         {1}{ \__erw_tl_map_thread_at:Nnn #1{#2}#3 }

```

```

456 {2}{ \_erw_tl_map_thread_at:Nnnn #1{#2}#3 }
457 {3}{ \_erw_tl_map_thread_at:Nnnnn #1{#2}#3 }
458 {4}{ \_erw_tl_map_thread_at:Nnnnnn #1{#2}#3 }
459 }
460 {
461   % Do nothing
462 }
463 {
464   \msg_error:nnn{\_erw}
465   {generic}
466   {erw_tl_map_thread_at:~count~of~#3~not~withing~1~to~4}
467 }
468 }
469 \cs_new_protected:Nn \erw_tl_map_thread:Nn
470 {
471   \int_step_inline:nn
472   {
473     \exp_args:Nf \tl_count:n{ \tl_head:n{#2} }
474   }
475   {
476     \erw_tl_map_thread_at:Nnn #1 {##1} {#2}
477   }
478 }
479 \cs_new:Nn \erw_tl_separators:n
480 {
481   \_erw_tl_separators:en{ \tl_count:n{#1} }{#1}
482 }

```

## 14 Closing

```

483 \ExplSyntaxOff
484 \endpackage

```