

The `erw-l3` package ^{*}

Erwann Rogard[†]

Released 2020/05/23

Abstract

Utilities like `expl3[1]`.

Résumé

Utilitaires de type `expl3[1]`.

Contents

| | | |
|-----------|---------------------|----------|
| I | Usage | 1 |
| 1 | Loading the package | 1 |
| 2 | <code>cs</code> | 1 |
| 3 | <code>csint</code> | 2 |
| 4 | <code>int</code> | 2 |
| 5 | <code>keys</code> | 2 |
| 6 | <code>lambda</code> | 2 |
| 7 | <code>option</code> | 2 |
| 8 | <code>prop</code> | 3 |
| 9 | <code>seq</code> | 3 |
| 10 | <code>sys</code> | 3 |
| 11 | <code>tl</code> | 4 |
| II | Listing | 6 |

^{*}This file describes version v2.7, last revised 2020/05/23.

[†]firstname dot lastname AusTria gmail dot com

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

| | | |
|-----------|-------------------------|-----------|
| IV | Implementation | 15 |
| 1 | Opening | 15 |
| 2 | cs | 15 |
| | 2.1 backend | 15 |
| | 2.2 frontend | 15 |
| 3 | clist | 15 |
| | 3.1 backend | 15 |
| | 3.2 frontend | 15 |
| 4 | csint | 15 |
| | 4.1 backend | 15 |
| | 4.2 frontend | 15 |
| 5 | int | 16 |
| | 5.1 backend | 16 |
| | 5.2 frontend | 17 |
| 6 | keys | 17 |
| | 6.1 frontend | 17 |
| 7 | lambda | 17 |
| 8 | msg | 17 |
| | 8.1 backend | 17 |
| | 8.2 frontend | 18 |
| 9 | prop | 18 |
| | 9.1 backend | 18 |
| | 9.2 frontend | 18 |
| 10 | oper | 20 |
| | 10.1 backend | 20 |
| | 10.2 frontend | 20 |
| 11 | seq | 20 |
| | 11.1 backend | 20 |
| | 11.2 frontend | 21 |
| 12 | sys | 22 |
| | 12.1 backend | 22 |
| 13 | tl | 24 |
| | 13.1 backend | 24 |
| | 13.2 frontend | 25 |
| 14 | option | 28 |

Part I

Usage

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

Requirement

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

2 cs

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

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

3 csint

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

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

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

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

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

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

4 int

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

5 keys

| | |
|------------------------------------|---|
| <code>\erw_keyval_error:Nn</code> | <code>\erw_keyval_error:Nn<token>{<keyval list>}</code> |
| <code>\erw_keyval_error:Nnn</code> | <code>\erw_keyval_error:Nnn<token>{<clist>}</code> |

6 lambda

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

7 option

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

oper / fold_set_par
oper / fold_apply_par
sys / timestamp_delim

8 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<prop><cs1><cs2>{<keyval list>}</code> |
|--|---|

| | |
|-------------------------------------|---|
| <code>\erw_prop_map_item:NNN</code> | <code>\erw_prop_map_item:NNN<cs><prop1><prop2></code> |
|-------------------------------------|---|

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

9 seq

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

| | | |
|-------------|--|--|
| <hr/> <hr/> | $\backslash erw_seq_compose:nN$ | $\backslash erw_seq_compose:nN\{\{\langle cs_1 \rangle\} \dots\} \langle seq \rangle$ |
| <hr/> <hr/> | $\backslash erw_seq_compose_c:nN$ | $\backslash erw_seq_compose_c:nN\{\{\langle cs\ name_1 \rangle\} \dots\} \langle seq \rangle$ |
| <hr/> <hr/> | $\backslash erw_seq_compose_vers:nN$ | $\backslash erw_seq_compose:nN\{\{\langle cs\ or\ code_1 \rangle\} \dots\} \langle seq \rangle$ |
| <hr/> <hr/> | $\backslash erw_seq_put_right_clist:Nn$ $\backslash erw_seq_put_right_clist:cn$ | $\backslash erw_seq_put_right_clist:Nn \langle seq \rangle \{\langle clist \rangle\}$ |
| <hr/> <hr/> | $\backslash erw_seq_put_right_prop:Nnn$ | $\backslash erw_seq_put_right_prop:Nnn \langle seq \rangle \langle prop \rangle \{\langle clist \rangle\}$ |
| <hr/> <hr/> | $\backslash erw_seq_use:Nn$ | $\backslash erw_seq_use:Nn \langle seq \rangle \{\langle items \rangle\}$ |

Also see [1, Section 8 of l3seq]

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

10 sys

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

11 tl

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

| | | |
|-------------|--|---|
| <hr/> <hr/> | $\backslash erw_tl_append_item:nn$ | $\backslash erw_tl_append_item:nn \{\langle arg\ list \rangle\} \{\langle arg \rangle\}$ |
| <hr/> <hr/> | $\backslash erw_tl_compose:nN$ $\backslash erw_tl_compose:nn$ | $\backslash erw_tl_compose:nn \{\{cs_1\} \dots\} \{\langle token\ list \rangle\}$ |

| | |
|--|---|
| <u><code>\erw_tl_compose_c:nN</code></u> <u><code>\erw_tl_compose_c:nn</code></u> | <code>\erw_tl_compose_c:nn{<cs name_1>...}{<token list>}</code> |
| <u><code>\erw_tl_compose_vers:nN</code></u> <u><code>\erw_tl_compose_vers:nn</code></u> | <code>\erw_tl_compose_vers:nn{<cs or code_1>...}{<token list>}</code> |
| <u><code>\erw_tl_fold:NN</code></u> <u><code>\erw_tl_fold:cN</code></u> | <code>\erw_tl_fold:NN<cs><tl var></code> |
| <u><code>\erw_tl_gset_function:N</code></u> <u><code>\erw_tl_gset_function:n</code></u> | <code>\erw_tl_gset_function:n{<code>}</code> |
| <u><code>\erw_tl_join:nn</code></u> <u><code>\erw_tl_join:nnn</code></u> <u><code>\erw_tl_join:nnnn</code></u> <u><code>\erw_tl_join:nnnnn</code></u> | <code>\erw_tl_join:nn{<token list₁₂</code> |
| <u><code>\erw_tl_last_item:n</code></u> | <code>\erw_tl_last_item:n{<token list>}</code> |
| <u><code>\erw_tl_map:n</code></u> <u><code>\erw_tl_map:Nn</code></u> | <code>\erw_tl_map:n{<items>}</code> Semantics Maps over <code><items></code> using the internal function set by <code>\erw_tl_gset_function:n</code> |
| <u><code>\erw_tl_map_inline:nn</code></u> | <code>\erw_tl_map_inline:nn{<code>}{<items>}</code> |
| <u><code>\erw_tl_map_thread:Nn</code></u> | <code>\erw_tl_math_thread:Nn<cs>{<items>}</code> |
| <u><code>\erw_tl_map_thread_at:Nnn</code></u> | <code>\erw_tl_math_thread_at:Nnn{<integer>}{<token list>}</code> |
| <u><code>\erw_tl_repeat:nn</code></u> | <code>\erw_tl_repeat:nn{<integer>}{<token list>}</code> |
| <u><code>\erw_tl_split:nnn</code></u> <u><code>\erw_tl_split:nn</code></u> | <code>\erw_tl_split:nn{<items>}{<delimiter>}</code> |
| <u><code>\erw_tl_separators:n</code></u> | <code>\erw_tl_separators:n{<items>}</code> Semantics According to the count of <code><items></code> : 1) <code>{<token list_{111 2) <code>{<token list₁₂₁ token list_{2 3) <code>{<token list₁₂₃</code>}</code>}</code> |

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 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
```

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

3 int

Listing 3.

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

2345
12345

4 lambda

Listing 4.

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

Hello, world!

5 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_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 7.

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

a,b,c

6 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

7 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
```

```

20200524-2232
20200524-2232
1343c4c%8b8
1343c4c%8b8
erw-l3%1343c4c%8b8

```

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:202005242232
D:20200524223240-04'00'
Hello, world!

```

8 tl

Listing 14.

```

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

```

```

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

```

Listing 15.

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

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

Listing 16.

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

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

Listing 17.

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

```
f(X)
g[f(X)]
```

Listing 18.

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

```
xxx
```

Listing 19.

```

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

```

a==b==c

Listing 20.

```

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

```

(a)(b)(c)

Listing 21.

```

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

```

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

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

Listing 22.

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

(b+B)

Part III

Other

1 Acknowledgment

This work has benefited from Q&A's from the L^AT_EX community [3]. `lambda` originally appeared in [2].

2 Install

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

3 Support

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

3.1 Platform

- i) Linux laptop 4.15.0-20-generic #21-Ubuntu SMP Tue Apr 24
↪ 06:16:15 UTC 2018 x86_64 x86_64 x86_64 GNU/Linux

3.2 Engine

- a) pdfTeX 3.14159265-2.6-1.40.20 (TeX Live 2019)
- b) pdfTeX 3.14159265-2.6-1.40.21 (TeX Live 2020)
- c) LuaHBTeX, Version 1.12.0 (TeX Live 2020)
- d) XeTeX 3.14159265-2.6-0.999992 (TeX Live 2020)

3.3 Results

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

References

- [1] The L^AT_EX3 Project Team *The L^AT_EX3 interfaces*, 2019, <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/interface3.pdf>
- [2] @sean-allred's answer to "How to create lambda expressions?", <https://tex.stackexchange.com/a/188053/112708>
- [3] <https://tex.stackexchange.com/users/112708/erwann?tab=questions>

Change History

| | | | |
|------|---|------|---|
| v1.1 | General: \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 \ProcessPackageKeysOption; . . . 14 | v1.7 | General: Add: option 14 |
| | Brought all the modules under one file; renamed l3erw to erw-l3; . . . 14 | | Add: sys 14 |
| v1.2 | General: disambig: \disambignewcmd no longer takes a token name as arg, rather a token. 14 | | Move: \erw_fold_apply_par:n . . 14 |
| | disambig: pushed the code inside \keys_define; 14 | | Move: \erw_fold_set_par:n . . . 14 |
| | Add: \erw_items_to 14 | | Rearrange: structure of implementation, e.g. section 10 . . 14 |
| | Add: \erw_last_item 14 | | Remove: document level functions, \numbrdcsnew, \numbrdcs 14 |
| | Add: \erw_repeat 14 | | Replace: listing's implem with that of tocloft 14 |
| | Add: \erw_split 14 | | Replace: vers. numb. from 3 to 2 digits 14 |
| | Add: \map_thread 14 | v1.8 | General: (deleted) 14 |
| | Front end cmds no longer generated with module disambig; Option of the same name deleted; 14 | | Add: function for all frontend functions. 14 |
| | Modify: \erw_compose, order in which functions composed ($g \circ f$ means f comes before g) 14 | | Remove: \erw_cs_set_eq:NN and variants 14 |
| | Rearrange: the doc to clearly separate frontend from backend . . 14 | | Remove: \erw_is_matrix:n (predicate must be expandable) . . 14 |
| v1.3 | General: Replace: versioning, should have been 0.1.2 14 | | Rename: all cs prefixes to agree with heading under which they come, e.g. \erw_identity:n by \erw_cs_identity:n 14 |
| v1.4 | General: Add: \erw_accum 14 | | Replace: \erw_seq_fold:NN by \erw_oper_fold_seq:NN and likewise for variants 14 |
| | Add: \erw_int_range 14 | v1.9 | General: Add: \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: \erw_oper_gset_function:N to \erw_tl_gset_function:N (and variants) 14 |
| | Add: \erw_set_map 14 | v2.0 | General: Add: \erw_jobnametimestamp:nn and variants 14 |
| | Remove: \erw_items_to (redundant with \tl_range:nnn) . 14 | | Remove: \merge:nn (redundant with \erw_join:nn) 14 |
| v1.5 | General: Modify: source repository . . 14 | | Rename: v0.0 to v1.0, etc. 14 |
| | Rearrange: frontend/backend sections 14 | | |
| | Remove: disambig 14 | | |
| | Split Section Preliminaries into Conventions and Requirement. . . 14 | | |

| | | | |
|------|--|------|---|
| v2.1 | General: Add: \erw_prop_to_clist:Nn, \erw_prop_put:NN, and \erw_prop_put:Nnn 14 Add: \erw_seq_from_clist:Nn, \erw_seq_from_prop:NNn, and \erw_seq_put_right:Nn 14 Move: all functions under section 10 to section 13 or section 11 , except \@@oper_compose:NnN 14 Replace: \erw_seq_fold:NN by __erw_seq_fold:NN 14 | v2.5 | General: Add: \erw_prop_put_keyval:Nn 14 |
| v2.2 | General: Add: \erw_seq_use:Nn 14 Add: \erw_tl_separators:n 14 | v2.6 | General: Add: \erw_cs_error:nn ... 14 Add: \erw_cs_error:n 14 Add: \erw_keyval_parse:NNNn .. 14 Add: \erw_prop_keyval_parse:NNNn .. 14 Add: \erw_prop_map_item:NNN .. 14 Add: \msg_new:nnn, module erw, messages: varnset 14 Remove: \erw_cs_apply 14 Remove: \erw_prop_put:NN 14 Remove: \erw_prop_put_keyval:Nn 14 Remove: \msg_new:nnn, module erw, messages: keyval/... 14 Rename: basics to cs 14 Replace: \erw_seq_from_clist by \erw_seq_put_right_clist 14 Replace: \erw_seq_from_prop by \erw_seq_put_right_prop 14 |
| v2.3 | General: Add: \msg_new:nnn, module erw, messages: csnset 14 Add: \msg_new:nnn, module erw, messages: keyval/... 14 Fix: 'mark as private code' (hitherto unnoticed) 14 Modify: behavior of \erw_seq_use:Nn 14 Move: all \msg_new:Nnn statements under same heading .. 14 | v2.7 | General: Add: \erw_keyval_error:Nnn 14 Add: \erw_keyval_error:Nn 14 Remove: \erw_cs_error:nn 14 Remove: \erw_cs_error:n 14 |
| v2.4 | General: Add: \erw_lambda:nnn 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 | |
| \begin | 389 |
| C | |
| cs commands: | |
| \cs_generate_variant:Nn | 13, 18, 136, 154, 166, 217, 252, 258, 277, 284, 291, 298, 465, 511 |
| \cs_gset:Npn | 16 |
| \cs_new:Nn | 4, 9, 14, 19, 20, 21, 22, 25, 29, 30, 45, 50, 51, 81, 85, 91, 92, 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 |
| \cs_new_protected:Nn .. | 34, 55, 93, 116, 124, 137, 146, 155, 167, 175, 183, 206, 241, 253, 271, 278, 285, 372, 405, 528, 532, 537, 561, 582, 596 |
| \cs_new_protected:Npn | 102 |
| \cs_set:Nn | 126, 148 |
| \cs_set:Npn | 8, 11, 60 |
| \cs_set_eq:NN | 534 |
| \cs_set_protected:Nn .. | 95, 96, 118, 192, 193, 243, 412, 417, 422, 428, 435 |
| \cs_split_function:N | 6 |
| E | |
| erw commands: | |
| \erw_cs_gset_eq:NN | 514 |
| \erw_cs_gset_inline:Nn ... | 14, 18, 518 |
| \erw_cs_identity:n | 1, 8 |

| | |
|---|--|
| \erw_cs_set_inline:Nn | \erw_tl_join:nn .. 4, 19, 351, 357, 363 |
| 1, 9, 13, 37, 220, 539 | \erw_tl_join:nnn |
| \erw_csint:nn | 4, 20, 335 |
| 2, 25 | \erw_tl_join:nnnn |
| \erw_csint_name:n ... 2, 24, 29, 32, 50 | 4, 21 |
| \erw_csint_names:nnn | \erw_tl_join:nnnnn |
| 2, 30 | 4, 22 |
| \erw_csint_names_braced: .. 2, 51, 500 | \erw_tl_last_item:n |
| \erw_csint_names_braced:n .. 2, 47, 50 | 4, 520 |
| \erw_csint_names_braced:nnn 2, 45, 53 | \erw_tl_map:n ... 4, 224, 528, 535, 540 |
| \erw_csint_new:n | \erw_tl_map:Nn |
| 2, 34, 497 | 4, 532 |
| \erw_csint_reset: | \erw_tl_map_inline:nn |
| 2, 55, 496 | 4, 537 |
| \erw_int_range:n | \erw_tl_map_thread:Nn |
| 2, 85 | 4, 582 |
| \erw_int_range:nn | \erw_tl_map_thread_at:Nnn 4, 561, 589 |
| 2, 81 | \erw_tl_math_thread:Nn |
| \erw_keyval_error:n | 4 |
| 179, 212 | \erw_tl_math_thread_at:Nnn |
| \erw_keyval_error:Nn | 4 |
| 2, 91 | \erw_tl_repeat:nn |
| \erw_keyval_error:Nnn | 4, 542 |
| 2, 92, 132 | \erw_tl_separators:n 5, 303, 592 |
| \erw_keyval_keyonly:nn | \erw_tl_split:nn |
| 172, 249 | 5, 557 |
| \erw_keyval_parse:NNNn | \erw_tl_split:nnn |
| 93, 186 | 5, 546, 559 |
| \erw_lambda:nnn | erw internal commands: |
| 2, 102 | __erw_cs_name:N |
| \erw_option:n | 4 |
| 2, 596 | __erw_csint_ext_tl |
| \erw_prop_gput_keyval:Nn | 58 |
| 206 | \g__erw_csint_int .. 23, 24, 36, 53, 57 |
| \erw_prop_keyval_parse:NNn | \g__erw_csint_name_tl |
| 175 | 24, 37 |
| \erw_prop_keyval_parse:NNNn | __erw_function:n |
| 3, 183, 189, 210 | 243, 248 |
| \erw_prop_keyval_parse_key:Nnn .. 167 | __erw_function:nn |
| \erw_prop_map_item:NNN ... 3, 137, 143 | 118, 122 |
| \erw_prop_put:NN | __erw_int_range:nnn .. 60, 70, 83, 87 |
| 146, 154 | __erw_keyval_function:n |
| \erw_prop_put:Nnn | 95, 98, 126, 131, 192, 197 |
| 155, 163, 166 | __erw_keyval_function:nn |
| \erw_prop_put_keyval:Nn | 96, 99, 193, 198 |
| 217 | __erw_lambda_expression ... 105, 108 |
| \erw_prop_to_clist:Nn 3, 124, 136, 256 | __erw_oper_compose:NnN |
| \erw_seq_compose:nN | 218, 261, 265, 472, 482 |
| 3, 3, 259 | \g__erw_oper_fold_apply_par_tl .. |
| \erw_seq_compose_c:nN | 235, 508 |
| 3, 263 | \g__erw_oper_fold_set_par_tl 231, 505 |
| \erw_seq_compose_vers:nN .. 3, 267, 269 | __erw_prop_append:nn |
| \erw_seq_put_right:Nn .. 285, 289, 291 | 148, 152 |
| \erw_seq_put_right_clist:Nn | __erw_prop_fun_keyval:NNNn ... 203 |
| 3, 271, 275, 277 | __erw_prop_keyval_parse:NNNn ... |
| \erw_seq_put_right_prop:NNn | 169, 177 |
| 3, 278, 282, 284 | __erw_prop_map_item:NNN ... 116, 140 |
| \erw_seq_use:Nn | __erw_seq_fold:NN .. 261, 265, 292, 298 |
| 3, 299 | \g__erw_seq_fold_item_tl |
| \erw_sys_jobnametimestamp: .. 3, 391 | 240, 294, 295, 296 |
| \erw_sys_jobnametimestamp:nn 3, 390 | __erw_seq_put_right_clist:Nn ... |
| \erw_sys_timestamp: | 241, 252, 255, 274 |
| 3, 365, 400 | __erw_seq_put_right_prop:NNn ... |
| \erw_sys_timestamp:nn ... 3, 359, 396 | 253, 258, 281 |
| \erw_sys_timestamp_delimiter: 3, 392 | __erw_sys_date:N |
| \erw_tl_append_item:nn 4, 72, 466 | 305 |
| \erw_tl_compose:nN | __erw_sys_date_dec: |
| 4, 470, 477 | 305, 347 |
| \erw_tl_compose:nn | __erw_sys_date_hex: |
| 4, 474 | 305, 348 |
| \erw_tl_compose_c:nN 4, 480, 487 | __erw_sys_datetime_base:n .. 325, 370 |
| \erw_tl_compose_c:nn 4, 484, 499 | __erw_sys_datetime_dec: |
| \erw_tl_compose_vers:nN .. 4, 490, 492 | 347 |
| \erw_tl_compose_vers:nn | __erw_sys_datetime_dec:n |
| 4, 494 | 325 |
| \erw_tl_fold:NN | __erw_sys_datetime_hex: |
| 4, 295, 472, 482, 503, 511 | 348 |
| \erw_tl_gset_function:N | __erw_sys_datetime_hex:n |
| 4, 512 | 325 |
| \erw_tl_gset_function:n | |
| 4, 516 | |

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}
```

3 clist

3.1 backend

3.2 frontend

4 csint

4.1 backend

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

4.2 frontend

```
25 \cs_new:Nn \erw_csint:nn
26 {
27   \use:c{__erw_csint_\int_to_alph:n{#1}:n}{#2}
```

```

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

```

5 int

5.1 backend

```

60 \cs_set:Npn \__erw_int_range:nnn #1 #2 #3
61 {
62   \int_compare:nNnTF
63   {
64     \int_eval:n{#2+1}
65   }>{#3}
66   {
67     {#1}
68   }
69   {
70     \__erw_int_range:nnn
71     {
72       \exp_args:Nx\erw_tl_append_item:nn{#1}
73       {
74         \int_eval:n{#2+1}
75       }
76     }

```

```

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

```

5.2 frontend

```

81 \cs_new:Nn \erw_int_range:nn
82 {
83   \__erw_int_range:nnn {{#1}}{#1}{#2}
84 }
85 \cs_new:Nn \erw_int_range:n
86 {
87   \__erw_int_range:nnn {}{0}{#1}
88   % ^^A Alt to:
89   % ^^A   \int_step_inline:nn {#1}{##1}
90 }

```

6 keys

6.1 frontend

```

91 \cs_new:Nn \erw_keyval_error:Nn{\msg_error{__erw}{generic}{unary~function~not~allowed}}
92 \cs_new:Nn \erw_keyval_error:Nnn{\msg_error{__erw}{generic}{binary~function~not~allowed}}
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 2.)

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_keyval_error:Nnn
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_tl_map:nn

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

(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

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_tl_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>

```