

erw-l3^{*}

Erwann Rogard[†]

Released 2019/10/12

Abstract

L^AT_EX3 package defining commands built around `expl3`[1]. For example, `\erw_compose` implements the mathematical concept $f_1 \circ f_2 \cdots \circ f_n$.

Contents

I	Usage	3
1	compose	3
	1.1 backend	3
2	csutil	3
	2.1 backend	3
3	int	4
	3.1 backend	4
4	map	5
	4.1 backend	5
5	numbrdcs	5
	5.1 backend	5
	5.2 frontend	6
II	Listings	7
1	compose	7
	1.1 backend	7
2	csutil	9
	2.1 backend	9
3	int	10
	3.1 backend	10

^{*}This file describes version v0.1.5, last revised 2019/10/12.

[†]firstname dot lastname AusTria gmail dot com

4	map	10
4.1	backend	10
5	numbrdcs	12
5.1	backend	12
5.2	frontend	13
III	Implementation	13
1	compose	13
1.1	backend	13
2	csutil	15
2.1	backend	15
3	map	17
3.1	backend	17
4	map	18
4.1	backend	18
5	numbrdcs	20
5.1	backend	20
5.2	frontend	21
IV	Other	21
1	Support	21
2	To do	21
3	Acknowledgment	21
	Change History	22
	Index	22

Conventions

The naming conventions are (loosely) those of L^AT_EX3. For example, $\langle cs \rangle$ stands for *control sequence*, which is described in [1, Part l3basics].

Requirement

Have `erw-13.sty` is in the path of the L^AT_EX engine.

Part I

Usage

In the preamble of `\documentclass`, put:

```
\usepackage[<options>]{erw-l3}
```

1 compose

1.1 backend

<code>\erw_compose:nV</code>	<code>\erw_compose:nV{<cs list>}<var></code>
<code>\erw_compose:nn</code>	

Implements the mathematical concept $f_1 \circ f_2 \cdots \circ f_n$. See Listing 1

<code>\erw_compose_c:nV</code>	<code>\erw_compose_c:nV{<cs names>}<var></code>
<code>\erw_compose_c:nn</code>	

See Listing 2

<code>\erw_compose_seq:nV</code>	<code>\erw_compose_seq:nV{<cs list>}<seq></code>
----------------------------------	--

Same as `\erw_compose:nV`, but saves each intermediary step See Listing 3

<code>\erw_compose_seq_c:nV</code>	<code>\erw_compose_seq_c:nV{<cs names>}<seq></code>
------------------------------------	---

See Listing 4

<code>\erw_compose_vers:nV</code>	<code>\erw_compose_vers:nV{<list of cs or code>}<var></code>
<code>\erw_compose_vers:nn</code>	

See Listing 5. Only the `nn` version is implemented

<code>\erw_compose_seq_vers:nV</code>	<code>\erw_compose_seq_vers:nV{<list of cs or code>}<seq></code>
<code>\erw_compose_seq_vers:nn</code>	

Not implemented

2 csutil

2.1 backend

<code>\erw_accum:nn</code>	<code>\erw_accum:nn{<token list>}<item></code>
----------------------------	--

Expands to a token list comprising the items of `<token list>` and `<item>`

<hr/> <code>\erw_apply:Nn</code> <hr/>	<code>\erw_apply:Nn<cs>{\<arg>}</code>
<code>\erw_apply:cn</code>	Expands to $\langle cs \rangle \{ \langle arg \rangle \}$
<code>\erw_apply:Nnn</code>	
<code>\erw_apply:Nnnn</code>	
<code>\erw_apply:Nnnnn</code>	
<hr/>	
<code>\erw_cs_set_eq:NN</code>	<code>\erw_cs_set_eq:NN<cs1>\<cs2></code>
<code>\erw_cs_set_eq:cN</code>	$\langle cs1 \rangle \leftarrow \langle cs2 \rangle$
<code>\erw_cs_gset_eq:NN</code>	
<code>\erw_cs_gset_eq:cN</code>	
<hr/>	
<code>\erw_cs_set_inline:Nn</code>	<code>\erw_cs_set_inline:Nn<cs>{\<code>}</code>
<code>\erw_cs_set_inline:cn</code>	
<code>\erw_cs_gset_inline:Nn</code>	
<code>\erw_cs_gset_inline:cn</code>	
<hr/>	
<code>\erw_identity:n</code>	<code>\erw_identity:n{\<arg>}</code>
	Expands to $\langle arg \rangle$
<hr/>	
<code>\erw_is_matrix_p:n</code>	<code>\erw_is_matrix_p:n{\<token list>}</code>
<code>\erw_is_matrix:nTF</code>	Checks if $\langle token list \rangle$ is a (square) matrix.
<hr/>	
<code>\erw_fold:NV</code>	<code>\erw_fold:NV<cs>\<var></code>
<code>\erw_fold:cV</code>	$\langle var \rangle \leftarrow \text{\erw_apply:NV}\langle cs \rangle \langle var \rangle$. See Listing 7.
<hr/>	
<code>\erw_last_item:nn</code>	<code>\erw_last_item:nn{\<int>}\{\<token list>\}</code>
<hr/>	
<code>\erw_merge:nn</code>	<code>\erw_merge:nn{\<tl 1>}\{\<tl 2>\}</code>
	Merges $\langle tl 1 \rangle \langle tl 2 \rangle$
<hr/>	
<code>\erw_repeat:nn</code>	<code>\erw_repeat:nn{\<int>}\{\<value>\}</code>
	See Listing 9
<hr/>	
<code>\erw_split:nn</code>	<code>\erw_split:nn{\<token list>}\{\<delimiter>\}</code>
	See Listing 10

3 int

3.1 backend

<hr/> <code>\erw_int_range:nn</code> <hr/>	<code>\erw_int_range:nn{\<first>}\last</code>
	Returns a range of integers. Implementation different than <code>\int_step_inline</code>
<hr/>	
<code>\erw_int_range:n</code>	<code>\erw_int_range:n{\<count>}</code>
	Returns a range of integers. Implementation different than <code>\int_step_inline</code> . See Listing 11

4 map

4.1 backend

<hr/> <code>\erw_set_map:N</code> <hr/>	<code>\erw_set_map:N⟨cs⟩</code>
<code>\erw_gset_map:N</code> <hr/>	Sets the function used by <code>\erw_map:n</code> .
<hr/> <code>\erw_set_map_inline:n</code> <hr/>	<code>\erw_set_map_inline:n{⟨code⟩}</code>
<code>\erw_gset_map_inline:n</code> <hr/>	Sets the function used by <code>\erw_map:n</code> .
<hr/> <code>\erw_map:n</code> <hr/>	<code>\erw_map:n{⟨token list⟩}</code> Applies the stored <code>⟨cs⟩</code> to each item in <code>⟨token list⟩</code> . An application is <code>\erw_is_matrix</code>
<hr/> <code>\erw_map:Nn</code> <hr/>	<code>\erw_map:Nn⟨cs⟩{⟨token list⟩}</code> See Listing 12. Redundant with <code>\tl_map_function:nN</code>
<hr/> <code>\erw_map_inline:nn</code> <hr/>	<code>\erw_map_inline:nn{⟨code⟩}{⟨args⟩}</code> See Listing 13
<hr/> <code>\erw_map_indexed:Nnn</code> <hr/>	<code>\erw_map_indexed:Nnn⟨cs⟩{⟨int⟩}{⟨matrix of tokens⟩}</code> Not implemented. See Listing 15.
<hr/> <code>\erw_map_thread:Nn</code> <hr/>	<code>\erw_map_thread:Nn⟨cs⟩{⟨matrix of tokens⟩}</code> Threads <code>⟨cs⟩</code> over the columns, where the arity of <code>⟨cs⟩</code> must be equal to the number of rows. See Listing 14
<hr/> <code>\erw_map_thread_at:Nnn</code> <hr/>	<code>\erw_map_thread_at:Nnn⟨cs⟩{⟨matrix of tokens⟩}</code>

5 numbrdcs

5.1 backend

<hr/> <code>\erw_numbrd_cs_reset:</code> <hr/>	<code>\erw_numbrd_cs_reset:{}_</code> See Listing 16
<hr/> <code>\erw_numbrd_cs_new:n</code> <hr/>	<code>\erw_numbrd_cs_new:n {⟨cs or code⟩}</code> Use it as the first arg to <code>\tl_function_map:Nn</code>
<hr/> <code>\erw_numbrd_cs:nn</code> <hr/>	<code>\erw_numbrd_cs:nn {⟨cs or code⟩}</code>
<hr/> <code>\erw_numbrd_cs_names_braced:nnn</code> <hr/>	<code>\erw_numbrd_cs_names_braced:nnn{⟨first⟩}{⟨step⟩}{⟨last⟩}</code> See Listing 16

5.2 frontend

<hr/> <code>\numbrdcsnew</code>	<code>\numbrdcsnew{\textit{list of cs or code}}</code>
<code>\numbrdcsnew*</code>	Creates numbered control sequences. The starred version does not reset. See Listing 17
<hr/> <code>\numbrdcs</code>	<code>\numbrdcs{\textit{int}}{\textit{arg}}</code>
	Evaluates control sequence numbered $\langle int \rangle$ with argument $\langle arg \rangle$. See Listing 17

Part II

Listings

1 compose

1.1 backend

Listing 1

```

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

```

Listing 2

```

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

```

Listing 3

```
\ExplSyntaxOn
\cs_set:Npn \__foo #1 {f(#1)}
\cs_set:Npn \__bar #1 {g[#1]}
\cs_set:Npn \__baz #1 {h\{#1\}}
\seq_new:N\l_tmp_seq
\seq_put_right:Nn\l_tmp_seq{X}
\erw_compose_seq:nV{
  \__baz\__bar\__foo}
\l_tmp_seq
\seq_item:Nn\l_tmp_seq{1}      X
\seq_item:Nn\l_tmp_seq{2}      f(X)
\seq_item:Nn\l_tmp_seq{3}      g[f(X)]
\seq_item:Nn\l_tmp_seq{4}      h{g[f(X)]}
\ExplSyntaxOff
```

Listing 4

```
\ExplSyntaxOn
\cs_set:Npn \__foo #1 {f(#1)}
\cs_set:Npn \__bar #1 {g[#1]}
\cs_set:Npn \__baz #1 {h\{#1\}}
\seq_new:N\l_tmp_seq
\seq_put_right:Nn\l_tmp_seq{X}
\erw_compose_seq_c:nV{
  \__baz\__bar\__foo}
\l_tmp_seq
\seq_item:Nn\l_tmp_seq{1}      X
\seq_item:Nn\l_tmp_seq{2}      f(X)
\seq_item:Nn\l_tmp_seq{3}      g[f(X)]
\seq_item:Nn\l_tmp_seq{4}      h{g[f(X)]}
\ExplSyntaxOff
```

Listing 5

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

2 csutil

2.1 backend

Listing 6

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

Listing 7

```
\ExplSyntaxOn
\cs_set:Npn \__foo #1 {f(#1)}
\tl_set:Nn \l_tmpa_tl{X}
\erw_fold_set_par:n{Nf}
\erw_fold_apply_par:n{Nf}
\erw_fold:NV \__foo \l_tmpa_tl
\l_tmpa_tl          f(X)
\cs_set:Npn \__bar #1 {g[#1]}
\erw_fold:cV{__bar} \l_tmpa_tl
\l_tmpa_tl          g[f(X)]
\ExplSyntaxOff
```

Listing 8

```
\ExplSyntaxOn
\erw_is_matrix:nTF
{
  { {a}{b}{c} }
  { {k}{l}{m} }
  { {x}{y}{z} }
}{T}{F}          T
\erw_is_matrix:nTF
{
  { {a}{c} }
  { {k} }
  { {x}{y}{z} }
}{T}{F}          F
\ExplSyntaxOff
```

Listing 9

```
\ExplSyntaxOn
\erw_repeat:nn
  {3}{abracad}abra          abracadabracadabracadabra
\ExplSyntaxOff
```

Listing 10

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

a==b==c

3 int

3.1 backend

Listing 11

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

2345
12345

4 map

4.1 backend

Listing 12

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

(a)(b)(c)

Listing 13

```
\ExplSyntaxOn
\cs_set:Npn \__foo #1 {(#1)}
\erw_map_inline:nn{
  (#1){{a}{b}{c}}
\ExplSyntaxOff
```

(a)(b)(c)

Listing 14

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

Listing 15 Debugging for `\erw_map_indexed`

```
\ExplSyntaxOn
\cs_set_protected:Npn \__foo:nn #1 #2
  {(#1+#2)}
\erw_map_thread:Nn
  \__foo:nn
  {
    {{1}{2}{3}}
    {{a}{b}{c}}
  }
  (1+a)(2+b)(3+c)
\exp_last_unbraced:Nx
\erw_map_thread:Nn
{
  \__foo:nn
  {
    {\erw_int_range:n{3}}
    {{a}{b}{c}}
  }
}
(123+a) (does not thread!)
\exp_last_unbraced:Nx
\erw_map_thread:Nn
{
  \__foo:nn
  {
    {\int_step_inline:nn{3}{#1}}
    {{a}{b}{c}}
  }
}
Illegal parameter number in definition of \l__exp_internal_tl!
\ExplSyntaxOff
```

5 numbrdcs

5.1 backend

Listing 16

```
\NewDocumentCommand{\myfoo}{m}{f{#1}}
\NewDocumentCommand{\mybar}{m}{g[#1]}
\NewDocumentCommand{\mybaz}{m}{h\{#1\}}
\numbrdcsnew{\mybaz}{g[#1]}{\myfoo}
\ExplSyntaxOn
\exp_last_unbraced:Nx
\erw_compose_c:nn
{
  {\erw_numbrd_cs_names_braced:
    nnn{1}{1}{3}}
  {X}
}
\ExplSyntaxOff
h{g[f(X)]}
```

5.2 frontend

Listing 17

```
\NewDocumentCommand{\thefoo}{m}{f(#1)}
\NewDocumentCommand{\thebar}{m}{g[#1]}
\NewDocumentCommand{\thebaz}{m}{h\{#1\}}
\numbrdcsnew{
  {\thefoo}
  {g[#1]}
  {\thebaz}}
\numbrdcs{1}{X}          f(X)
\numbrdcs{2}{X}          g[X]
\numbrdcs{3}{X}          h{X}
\numbrdcsnew*{
  {\thefoo}
  {g[#1]}
  {\thebaz}}
\numbrdcs{4}{X}          f(X)
\numbrdcs{5}{X}          g[X]
\numbrdcs{6}{X}          h{X}
```

Part III

Implementation

```
1 \NeedsTeXFormat{LaTeX2e}
2 \RequirePackage{expl3}[2018/06/01]
3 \RequirePackage{xparse}[2018/02/01]
4 \RequirePackage{l3keys2e}
5 \ExplSyntaxOn
6 \msg_new:nnn{erw}{generic}{#1}
```

1 compose

1.1 backend

```
7 \cs_set:Npn \erw_compose:NnV
8   #1 % method
9   #2 % funs
10  #3 % var
11  {
12    \erw_fold_set_par:n{Nf}
13    \erw_fold_apply_par:n{Nf}
14    \erw_cs_set_inline:Nn \__erw_map:n
15    {
16      #1{##1}#3
17    }
18    \exp_args:Nf\erw_map:n
19    {
```

```

20     \tl_reverse:n{#2}
21   }
22 }
23 \cs_set:Npn \erw_compose:nV #1 #2
24 {
25   \erw_compose:NnV \erw_fold:NV {#1} #2
26 }
27 \cs_set:Npn \erw_compose_c:nV #1 #2
28 {
29   \erw_compose:NnV \erw_fold:cV {#1} #2
30 }
31 \tl_new:N \__erw_compose_tl
32 \cs_set:Npn \erw_compose:nn #1 #2
33 {
34   \tl_set:Nn \__erw_compose_tl {#2}
35   \erw_compose:nV{#1}\__erw_compose_tl
36   \__erw_compose_tl
37 }
38 \cs_set:Npn \erw_compose_c:nn #1 #2
39 {
40   \tl_set:Nn \__erw_compose_tl {#2}
41   \erw_compose_c:nV{#1}\__erw_compose_tl
42   \__erw_compose_tl
43 }
44 \cs_set:Npn \erw_compose_seq:nV #1 #2
45 {
46   \erw_compose:NnV \erw_fold_seq:NV {#1} #2
47 }
48 \cs_set:Npn \erw_compose_seq_c:nV
49   #1 % funs
50   #2 % seq
51 {
52   \erw_compose:NnV \erw_fold_seq:cV {#1} #2
53 }
54 \cs_set:Npn \erw_compose_vers:nV #1 #2
55 {
56   \msg_error:nnn{erw}{generic}{erw_compose_vers:nV~yet-to-be-implemented}
57 }
58 \cs_set:Npn \erw_compose_seq_vers:nV #1 #2
59 {
60   \msg_error:nnn{erw}{generic}{erw_compose_vers:nV~yet-to-be-implemented}
61 }
62 \cs_set:Npn \erw_compose_vers:nn #1 #2
63 {
64   \erw_numbrd_cs_reset:{}
65   \tl_map_function:nN{#1}\erw_numbrd_cs_new:n
66   \exp_last_unbraced:Nx
67   \erw_compose_c:nn
68     {{\erw_numbrd_cs_names_braced:{}}}
69     {#2}
70 }

```

2 csutil

2.1 backend

```
71 \cs_set:Npn \erw_accum:nn #1 #2
72 {
73     {#1{#2}}
74 }
75 \cs_set:Npn \__erw_cs_name:N #1
76 {
77     \exp_last_unbraced:Nf \use_i:nnn {\cs_split_function:N #1}
78 }
79 \cs_set:Npn \erw_apply:Nn
80 #1 % fun
81 #2 % tl
82 {
83     #1{#2}
84 }
85 \cs_generate_variant:Nn \erw_apply:Nn {No, Nf, Nx, c}
86 \cs_set:Npn \erw_cs_set_eq:NN #1 #2
87 {
88     \cs_set:Npn #1 ##1{#2{##1}}
89 }
90 \cs_generate_variant:Nn \erw_cs_set_eq:NN {cN}
91 \cs_set:Npn \erw_cs_gset_eq:NN #1 #2
92 {
93     \cs_gset:Npn #1 ##1{#2{##1}}
94 }
95 \cs_generate_variant:Nn \erw_cs_gset_eq:NN {cN}
96 \cs_set:Npn \erw_cs_set_inline:Nn #1 #2
97 {
98     \cs_set:Npn #1 ##1{#2}
99 }
100 \cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}
101 \cs_set:Npn \erw_cs_gset_inline:Nn #1 #2
102 {
103     \cs_gset:Npn #1 ##1{#2}
104 }
105 \cs_generate_variant:Nn \erw_cs_gset_inline:Nn {cn}
106 \tl_set:Nn \__erw_fold_set_par_tl{\c_novalue_tl}
107 \tl_set:Nn \__erw_fold_apply_par_tl{\c_novalue_tl}
108 \cs_set:Npn \erw_fold_set_par:n #1
109 {
110     \tl_set:Nn \__erw_fold_set_par_tl{#1}
111 }
112 \cs_set:Npn \erw_fold_apply_par:n #1
113 {
114     \tl_set:Nn \__erw_fold_apply_par_tl{#1}
115 }
116 \cs_set:Npn \erw_fold:NV
117 #1 % fun
118 #2 % var
119 {
120     \use:c{tl_set:\__erw_fold_set_par_tl}
```

```

121     #2
122     {\use:c{erw_apply:\_erw_fold_apply_par_tl}{#1}{#2}}
123 }
124 \cs_generate_variant:Nn \erw_fold:NV {cV}
125 \tl_new:N \_erw_fold_seq_item_tl
126 \cs_set:Npn \erw_fold_seq:NV
127   #1 % fun
128   #2 % seq
129 {
130   \seq_get_right:NN #2 \_erw_fold_seq_item_tl
131   \erw_fold:NV #1 \_erw_fold_seq_item_tl
132   \seq_put_right:No #2 {\_erw_fold_seq_item_tl}
133 }
134 \cs_generate_variant:Nn \erw_fold_seq:NV {cV}
135 \cs_set:Npn \erw_identity:n #1{#1}
136 \prg_set_conditional:Npnn \erw_is_matrix:n #1 { p, TF }
137 {
138   \erw_gset_map_inline:n{==\tl_count:n{##1}}
139   \int_compare:nTF
140   {
141     \exp_args:Nf\tl_count:n{\tl_head:n{#1}}
142     \exp_args:Nf \erw_map:n
143     {
144       \tl_tail:n{#1}
145     }
146   }
147   {\prg_return_true:}
148   {\prg_return_false:}
149 }
150 % Deprecated in v0.1.4 after realizing \cs{tl_range:n} does the job
151 %\cs_set:Npn \_erw_items_to:nnn #1 #2 #3
152 %{
153 %   \int_compare:nNnTF
154 %   {#1}>{#2}
155 %   {
156 %     \exp_args:Nf \tl_head:n{#3}
157 %     \_erw_items_to:nnn
158 %     {#1}
159 %     {\int_eval:n{#2+1}}
160 %     {\exp_args:Nf \tl_tail:n{#3}}
161 %   }
162 %   {
163 %     \exp_args:Nf \tl_head:n{#3}
164 %   }
165 %}
166 %\cs_set:Npn \erw_items_to:nn #1 #2
167 %{
168 %   \_erw_items_to:nnn
169 %   {#1}
170 %   {1}
171 %   {#2}
172 %}
173 \cs_set:Npn \erw_last_item:n #1
174 {

```



```

175     \exp_args:Nof \tl_item:nn
176       {#1}
177     {
178       \tl_count:n{#1}
179     }
180   }
181   \cs_set:Npn \erw_merge:nn #1 #2
182   {
183     {#1#2}
184   }
185   \cs_set:Npn \erw_repeat:nn #1 #2
186   {
187     \int_step_inline:nnnn{1}{1}{#1}{#2}
188   }
189   \cs_set:Npn \erw_split:nnn #1 #2 #3
190   {
191     \tl_head:n{#1}
192     \use:c{exp_args:#3} \tl_map_inline:nn
193     {
194       \tl_tail:n
195       {
196         #1
197       }
198     }{#2##1}
199   }
200   \cs_set:Npn \erw_split:nn #1 #2
201   {
202     \erw_split:nnn{#1}{#2}{Nf}
203   }

```

3 map

3.1 backend

```

204   \cs_set:Npn \__erw_int_range:nnn #1 #2 #3
205   {
206     \int_compare:nNnTF
207     {
208       \int_eval:n{#2+1}
209     }>{#3}
210     {
211       {#1}
212     }
213     {
214       \__erw_int_range:nnn
215       {
216         \exp_args:Nx\erw_accum:nn{#1}
217         {
218           \int_eval:n{#2+1}
219         }
220       }
221       {\int_eval:n{#2+1}}
222       {#3}
223     }

```

```

224 }
225 \cs_set:Npn \erw_int_range:nn #1 #2
226 {
227   \__erw_int_range:nnn {{#1}}{#1}{#2}
228 }
229 \cs_set:Npn \erw_int_range:n #1
230 {
231   \__erw_int_range:nnn {}{0}{#1}
232 % Alt to:
233 %   \int_step_inline:nn {#1}{##1}
234 }

```

4 map

4.1 backend

```

235 \cs_set:Npn \erw_gset_map:N #1
236 {
237   \erw_cs_gset_eq:NN \__erw_map:n #1
238 }
239 \cs_set:Npn \erw_gset_map_inline:n #1
240 {
241   \erw_cs_gset_inline:Nn \__erw_map:n {#1}
242 }
243 \cs_set:Npn \erw_map:n #1
244 {
245   \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
246 }
247 \cs_set:Npn \__erw_map:nn #1 #2
248 {
249   \quark_if_recursion_tail_stop:n{#1}
250   \__erw_map:n{#1} \__erw_map:nn{#2}
251 }
252 \cs_new:Npn \__erw_map:n #1
253 {
254   \msg_error:nnn
255     {erw}
256     {generic}
257     {\__erw_map:n~not~set}
258 }
259 \cs_set:Npn \erw_map:Nn
260   #1 % fun
261   #2 % tl
262 {
263   \erw_cs_set_eq:NN \__erw_map:n #1
264   \erw_map:n{#2}
265 }
266 \cs_set:Npn \erw_map_inline:nn
267   #1 % inl
268   #2 % tl
269 {
270   \erw_cs_set_inline:Nn \__erw_map:n {#1}
271   \erw_map:n{#2}
272 }

```

```

273 \cs_set:Npn \erw_apply:Nnn #1 #2 #3
274 {
275     #1{#2}{#3}
276 }
277 \cs_set:Npn \erw_apply:Nnnn #1 #2 #3 #4
278 {
279     #1{#2}{#3}{#4}
280 }
281 \cs_set:Npn \erw_apply:Nnnnn #1 #2 #3 #4 #5
282 {
283     #1{#2}{#3}{#4}{#5}
284 }
285 \cs_set:Npn \__erw_map_thread_at:Nnn #1 #2 #3
286 {
287     \erw_apply:Nn #1
288     {\exp_args:Nf\tl_item:nn {#3} {#2} }
289 }
290 \cs_set:Npn \__erw_map_thread_at:Nnnn #1 #2 #3 #4
291 {
292     \erw_apply:Nnn #1
293     {\exp_args:Nf\tl_item:nn {#3} {#2} }
294     {\exp_args:Nf\tl_item:nn {#4} {#2} }
295 }
296 \cs_set:Npn \__erw_map_thread_at:Nnnnn #1 #2 #3 #4 #5
297 {
298     \erw_apply:Nnnn #1
299     {\exp_args:Nf\tl_item:nn {#3} {#2} }
300     {\exp_args:Nf\tl_item:nn {#4} {#2} }
301     {\exp_args:Nf\tl_item:nn {#5} {#2} }
302 }
303 \cs_set:Npn \__erw_map_thread_at:Nnnnnn #1 #2 #3 #4 #5 #6
304 {
305     \erw_apply:Nnnnn #1
306     {\exp_args:Nf\tl_item:nn {#3} {#2} }
307     {\exp_args:Nf\tl_item:nn {#4} {#2} }
308     {\exp_args:Nf\tl_item:nn {#5} {#2} }
309     {\exp_args:Nf\tl_item:nn {#6} {#2} }
310 }
311 \cs_set:Npn \erw_map_thread_at:Nnn #1 #2 #3
312 {
313     \exp_args:Nf\int_case:nnTF
314     {
315         \tl_count:n{#3}
316     }
317     {
318         {1}{ \__erw_map_thread_at:Nnn #1{#2}#3 }
319         {2}{ \__erw_map_thread_at:Nnnn #1{#2}#3 }
320         {3}{ \__erw_map_thread_at:Nnnnn #1{#2}#3 }
321         {4}{ \__erw_map_thread_at:Nnnnnn #1{#2}#3 }
322     }
323     {
324         % Do nothing
325     }
326 {

```

```

327         \msg_error:nnn{erw}
328         {generic}
329         {erw_map_thread_at::~count~of~#3~not~withing~1~to~4}
330     }
331 }
332 \cs_set:Npn \erw_map_thread:Nn #1 #2
333 {
334     % TODO check that #2 is a matrix
335     \int_step_inline:nn
336     {
337         \exp_args:Nf \tl_count:n{ \tl_head:n{#2} }
338     }
339     {
340         \erw_map_thread_at:Nnn #1 {##1} {#2}
341     }
342 }

```

5 numbrdcs

5.1 backend

```

343 \int_new:N \__erw_numbrd_cs_int
344 \cs_set:Npn \erw_numbrd_cs_name:n #1{__erw_numbrd_cs_\int_to_alph:n{#1}:n}
345 \cs_set:Npn \erw_numbrd_cs_name_braced:n #1{{\erw_numbrd_cs_name:n{#1}}}
346 \tl_set:Nn \__erw_numbrd_cs_name_tl {\erw_numbrd_cs_name:n{\__erw_numbrd_cs_int}}
347 \cs_set:Npn \erw_numbrd_cs:nn #1 #2
348 {
349     \erw_apply:cn{\__erw_numbrd_cs_\int_to_alph:n{#1}:n}{#2}
350 }
351 \cs_new_protected:Npn \erw_numbrd_cs_reset:
352 {
353     \int_zero:N \__erw_numbrd_cs_int
354     \tl_set:Nn \__erw_numbrd_cs_ext_tl{}
355 }
356 \cs_new_protected:Npn \erw_numbrd_cs_new:n #1
357 {
358     \int_incr:N \__erw_numbrd_cs_int
359     \erw_cs_set_inline:cn{\__erw_numbrd_cs_name_tl}
360     {
361         \token_if_cs:NTF
362         {#1}
363         {#1{##1}}
364         {#1}
365     }
366 }
367 \cs_new:Npn \erw_numbrd_cs_names:nnn #1 #2 #3
368 {
369     \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_numbrd_cs_name:n
370 }
371 \cs_new:Npn \erw_numbrd_cs_names_braced:nnn #1 #2 #3
372 {
373     \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_numbrd_cs_name_braced:n
374     % TODO \tl_range_braced:nnn?
375 }

```

```

376 \cs_new:Npn \erw_numbrd_cs_names_braced:
377 {
378   \erw_numbrd_cs_names_braced:nnn{1}{1}{\_erw_numbrd_cs_int}
379 }

```

5.2 frontend

```

380 \NewDocumentCommand{\numbrdcsnew}{ s m }
381 {
382   \IfBooleanTF{#1}
383     {}
384     { \erw_numbrd_cs_reset:{} }
385   \tl_map_function:nN {#2}\erw_numbrd_cs_new:n
386 }
387 \NewDocumentCommand{\numbrdcs}{ m m }
388 {
389   \erw_numbrd_cs:nn{#1}{#2}
390 }
391 % \ProcessKeysPackageOptions{ erw }
392 \ExplSyntaxOff

```

Part IV

Other

1 Support

This package is available from <https://www.ctan.org/pkg/erw-13> (release) or <https://github.com/rogard/erw-13> (development) where you can report issues.

2 To do

- Missing variants of `\erw_compose`
- `\erw_map_indexed`. See Listing 15
- Need to give some thought to ‘protected’

3 Acknowledgment

I thank those that have answered my questions on forums pertaining to L^AT_EX3. See here: <https://tex.stackexchange.com/users/112708/erwann?tab=questions> and here: <https://latex.org/forum/memberlist.php?mode=viewprofile&u=61329>

References

- [1] The L^AT_EX3 Project Team *The L^AT_EX3 interfaces* <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/interface3.pdf>

[2] The L^AT_EX3 Project Team *The xparse package* <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf>

Change History

0.1	General: Initial version	21	Added <code>\erw_split</code>	21
0.1.1	General:	21	Added <code>\map_thread</code>	21
	<code>\numbrdcsnew</code> changed to		Front end cmds no longer generated	
	<code>\newnumbrdcs</code> and made		with module <code>disambig</code> ; Option of	
	'disambiguable'	21	the same name deleted;	21
	<code>disambig/backend</code> : changes to the		Re-arranged the doc to clearly	
	key, added		separate frontend from backend . .	21
	<code>\ProcessPackageKeysOption</code> ; . . .	21	0.1.3	
	Brought all the modules under one		General: Wrong versioning, should	
	file; renamed <code>l3erw</code> to <code>erw-l3</code> ; . . .	21	have been 0.1.2	21
0.1.2	General:	21	0.1.4	
	<code>\erw_compose</code> reversed order in		General:	21
	which the functions are composed,		Added <code>\erw_accum</code>	21
	such that it now conforms to the		Added <code>\erw_int_range</code>	21
	mathematical convention ($g \circ f$		Added <code>\erw_is_matrix</code>	21
	means f comes before g)	21	Added <code>\erw_merge</code>	21
	<code>disambig</code> : pushed the code inside		Added <code>\erw_set_map_inline</code>	21
	<code>\keys_define:\disambignewcmd</code>		Added <code>\erw_set_map</code>	21
	no longer takes a token name as		Removed <code>\erw_items_to</code>	
	arg, rather a token.	21	(redundant with <code>\tl_range:nnn</code>) .	21
	Added <code>\erw_items_to</code>	21	0.1.5	
	Added <code>\erw_last_item</code>	21	General: Rearranged frontend/backend	
	Added <code>\erw_repeat</code>	21	sections	21
			Removed <code>disambig</code>	21
			Split Section Preliminaries into	
			Conventions and Requirement. . .	21

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.

	C	44, 48, 54, 58, 62, 71, 75, 79, 86, 88,
<code>\cs</code>	150	91, 96, 98, 101, 108, 112, 116, 126,
cs commands:		135, 151, 166, 173, 181, 185, 189,
<code>\cs_generate_variant:Nn</code>		200, 204, 225, 229, 235, 239, 243,
.	85, 90, 95, 100, 105, 124, 134	247, 259, 266, 273, 277, 281, 285,
<code>\cs_gset:Npn</code>	93, 103	290, 296, 303, 311, 332, 344, 345, 347
<code>\cs_new:Npn</code>	252, 367, 371, 376	<code>\cs_split_function:N</code>
<code>\cs_new_protected:Npn</code>	351, 356	77
<code>\cs_set:Npn</code>	7, 23, 27, 32, 38,	D
		<code>\disambignewcmd</code>
		22

\documentclass 3

E

erw commands:

\erw_accum 22
\erw_accum:nn 3, 71, 216
\erw_apply:Nn ... 4, 4, 79, 85, 287, 349
\erw_apply:Nnn 4, 273, 292
\erw_apply:Nnnn 4, 277, 298
\erw_apply:Nnnnn 4, 281, 305
\erw_compose 1, 21, 22
\erw_compose:nn 3, 3, 23, 32, 35
\erw_compose:Nnn 7, 25, 29, 46, 52
\erw_compose_c:nn ... 3, 27, 38, 41, 67
\erw_compose_seq:nn 3, 44
\erw_compose_seq_c:nn 3, 48
\erw_compose_seq_vers:nn 3, 58
\erw_compose_vers:nn 3, 54, 62
\erw_cs_gset_eq:NN ... 4, 91, 95, 237
\erw_cs_gset_inline:Nn 4, 101, 105, 241
\erw_cs_set_eq:NN 4, 86, 90, 263
\erw_cs_set_inline:Nn
..... 4, 14, 96, 100, 270, 359
\erw_fold:Nn .. 4, 25, 29, 116, 124, 131
\erw_fold_apply_par:n 13, 112
\erw_fold_seq:Nn 46, 52, 126, 134
\erw_fold_set_par:n 12, 108
\erw_gset_map:N 5, 235
\erw_gset_map_inline:n ... 5, 138, 239
\erw_identity:n 4, 135
\erw_int_range 22
\erw_int_range:n 4, 229
\erw_int_range:nn 4, 225
\erw_is_matrix 5, 22
\erw_is_matrix:n 136
\erw_is_matrix:nTF 4
\erw_is_matrix_p:n 4
\erw_items_to 22
\erw_items_to:nn 166
\erw_last_item 22
\erw_last_item:n 173
\erw_last_item:nn 4
\erw_map:n 5, 5, 5, 18, 142, 243, 264, 271
\erw_map:Nn 5, 259
\erw_map_indexed 12, 21
\erw_map_indexed:Nnn 5
\erw_map_inline:nn 5, 266
\erw_map_thread:Nn 5, 332
\erw_map_thread_at:Nnn ... 5, 311, 340
\erw_merge 22
\erw_merge:nn 4, 181
\erw_numbrd_cs:nn 5, 347, 389
\erw_numbrd_cs_name:n
..... 344, 345, 346, 369

\erw_numbrd_cs_name_braced:n 345, 373
\erw_numbrd_cs_names:nnn 367
\erw_numbrd_cs_names_braced: 68, 376
\erw_numbrd_cs_names_braced:nnn .
..... 5, 371, 378
\erw_numbrd_cs_new:n . 5, 65, 356, 385
\erw_numbrd_cs_reset: 5, 64, 351, 384
\erw_repeat 22
\erw_repeat:nn 4, 185
\erw_set_map 22
\erw_set_map:N 5
\erw_set_map_inline 22
\erw_set_map_inline:n 5
\erw_split 22
\erw_split:nn 4, 200
\erw_split:nnn 189, 202

erw internal commands:

_erw_compose_tl
..... 31, 34, 35, 36, 40, 41, 42
_erw_cs_name:N 75
_erw_fold_apply_par_tl 107, 114, 122
_erw_fold_seq_item_tl
..... 125, 130, 131, 132
_erw_fold_set_par_tl . 106, 110, 120
_erw_int_range:nnn 204, 214, 227, 231
_erw_items_to:nnn ... 151, 157, 168
_erw_map:n
..... 14, 237, 241, 250, 252, 263, 270
_erw_map:nn 245, 247, 250
_erw_map_thread_at:Nnn ... 285, 318
_erw_map_thread_at:Nnnn .. 290, 319
_erw_map_thread_at:Nnnnn . 296, 320
_erw_map_thread_at:Nnnnnn 303, 321
_erw_numbrd_cs_ext_tl 354
_erw_numbrd_cs_int
..... 343, 346, 353, 358, 378
_erw_numbrd_cs_name_tl ... 346, 359

exp commands:

\exp_args:Nf 18, 141, 142,
156, 160, 163, 288, 293, 294, 299,
300, 301, 306, 307, 308, 309, 313, 337
\exp_args:Nof 175
\exp_args:Nx 216
\exp_last_unbraced:Nf 77
\exp_last_unbraced:Nx 66
\ExplSyntaxOff 392
\ExplSyntaxOn 5

I

\IfBooleanTF 382

int commands:

\int_case:nnTF 313
\int_compare:nNnTF 153, 206
\int_compare:nTF 139

<code>\int_eval:n</code>	159, 208, 218, 221	<code>\q_recursion_stop</code>	245
<code>\int_incr:N</code>	358	<code>\q_recursion_tail</code>	245
<code>\int_new:N</code>	343		
<code>\int_step_function:nnnN</code>	369, 373		
<code>\int_step_inline</code>	4, 4		
<code>\int_step_inline:nn</code>	233, 335		
<code>\int_step_inline:nnnn</code>	187		
<code>\int_to_alph:n</code>	344, 349		
<code>\int_zero:N</code>	353		
	K		R
keys commands:		<code>\RequirePackage</code>	2, 3, 4
<code>\keys_define</code>	22		
			S
	M	seq commands:	
map commands:		<code>\seq_get_right:NN</code>	130
<code>\map_thread</code>	22	<code>\seq_put_right:Nn</code>	132
msg commands:			
<code>\msg_error:nnn</code>	56, 60, 254, 327		
<code>\msg_new:nnn</code>	6		
			T
	N	tl commands:	
<code>\NeedsTeXFormat</code>	1	<code>\c_novalue_tl</code>	106, 107
<code>\NewDocumentCommand</code>	380, 387	<code>\tl_count:n</code>	138, 141, 178, 315, 337
<code>\newnumbrdcs</code>	22	<code>\tl_function_map:Nn</code>	5
<code>\numbrdcs</code>	6, 387	<code>\tl_head:n</code>	141, 156, 163, 191, 337
<code>\numbrdcsnew</code>	6, 22, 380	<code>\tl_item:nn</code>	175, 288, 293,
<code>\numbrdcsnew*</code>	6		294, 299, 300, 301, 306, 307, 308, 309
		<code>\tl_map_function:nN</code>	5, 65, 385
		<code>\tl_map_inline:nn</code>	192
	P	<code>\tl_new:N</code>	31, 125
prg commands:		<code>\tl_range:nnn</code>	22
<code>\prg_return_false:</code>	148	<code>\tl_range_braced:nnn</code>	374
<code>\prg_return_true:</code>	147	<code>\tl_reverse:n</code>	20
<code>\prg_set_conditional:Npnn</code>	136	<code>\tl_set:Nn</code>	
<code>\ProcessKeysPackageOptions</code>	391		34, 40, 106, 107, 110, 114, 346, 354
<code>\ProcessPackageKeysOption</code>	22	<code>\tl_tail:n</code>	144, 160, 194
		token commands:	
	Q	<code>\token_if_cs:NTF</code>	361
quark commands:			
<code>\quark_if_recursion_tail_stop:n</code> . .	249		
			U
		use commands:	
		<code>\use:N</code>	120, 122, 192
		<code>\use_i:nnn</code>	77
		<code>\usepackage</code>	3