

erw-l3*

Erwann Rogard[†]

Released 2018/6/22

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

1	Preliminaries	2
I	Usage	2
1	backend	2
1.1	compose	3
1.2	csutil	3
1.3	int	4
1.4	map	4
1.5	numbrdcs	5
2	frontend	5
2.1	disambig	5
2.2	numbrdcs	6
II	Listings	7
1	Backend	7
1.1	compose	7
1.2	csutil	9
1.3	int	10
1.4	map	10
1.5	numbrdcs	12
2	Frontend	13
2.1	disambig	13
2.2	numbrdcs	14

*This file describes version v0.1.4, last revised 2018/6/22.

[†]firstname dot lastname AusTria gmail dot com

III	Implementation	14
1	Back end	14
1.1	compose	14
1.2	csutil	15
1.3	map	18
1.4	map	19
1.5	numbrdcs	21
2	frontend	21
2.1	disambig	21
2.2	numbrdcs	22
IV	Other	23
1	Support	23
2	To do	23
3	Acknowledgment	23
	Change History	23
	Index	24

1 Preliminaries

See [Part IV section 1](#) on how to get this package. To use it, make sure the file `erw-l3.sty` is in the path of the \LaTeX engine. In the preamble of your \LaTeX document, put:

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

Part I

Usage

The naming conventions are (loosely) those of \LaTeX 3. For example, `<cs>` stands for *control sequence*, which is described in [\[1, Part l3basics\]](#).

1 backend

We call ‘backend’ commands that are `expl3`-like.

1.1 compose

<hr/> <code>\erw_compose:nV</code> <hr/>	<code>\erw_compose:nV{<cs list>}<var></code>
<code>\erw_compose:nn</code> <hr/>	Implements the mathematical concept $f_1 \circ f_2 \cdots \circ f_n$. See Listing 1
<hr/> <code>\erw_compose_c:nV</code> <hr/>	<code>\erw_compose_c:nV{<cs names>}<var></code>
<code>\erw_compose_c:nn</code> <hr/>	See Listing 2
<hr/> <code>\erw_compose_seq:nV</code> <hr/>	<code>\erw_compose_seq:nV{<cs list>}<seq></code>
	Same as <code>\erw_compose:nV</code> , but saves each intermediary step See Listing 3
<hr/> <code>\erw_compose_seq_c:nV</code> <hr/>	<code>\erw_compose_seq_c:nV{<cs names>}<seq></code>
	See Listing 4
<hr/> <code>\erw_compose_vers:nV</code> <hr/>	<code>\erw_compose_vers:nV{<list of cs or code>}<var></code>
<code>\erw_compose_vers:nn</code> <hr/>	See Listing 5. Only the nn version is implemented
<hr/> <code>\erw_compose_seq_vers:nV</code> <hr/>	<code>\erw_compose_seq_vers:nV{<list of cs or code>}<seq></code>
<code>\erw_compose_seq_vers:nn</code> <hr/>	Not implemented

1.2 csutil

<hr/> <code>\erw_accum:nn</code> <hr/>	<code>\erw_accum:nn{<token list>}{<item>}</code>
	Expands to a token list comprising the items of <code><token list></code> and <code><item></code>
<hr/> <code>\erw_apply:Nn</code> <hr/>	<code>\erw_apply:Nn<cs>{<arg>}</code>
<code>\erw_apply:cn</code> <hr/>	Expands to <code><cs>{<arg>}</code>
<code>\erw_apply:Nnn</code> <hr/>	
<code>\erw_apply:Nnnn</code> <hr/>	
<code>\erw_apply:Nnnnn</code> <hr/>	
<hr/> <code>\erw_cs_set_eq:NN</code> <hr/>	<code>\erw_cs_set_eq:NN<cs1><cs2></code>
<code>\erw_cs_set_eq:cN</code> <hr/>	<code><cs1>←<cs2></code>
<code>\erw_cs_gset_eq:NN</code> <hr/>	
<code>\erw_cs_gset_eq:cN</code> <hr/>	
<hr/> <code>\erw_cs_set_inline:Nn</code> <hr/>	<code>\erw_cs_set_inline:Nn<cs>{<code>}</code>
<code>\erw_cs_set_inline:cn</code> <hr/>	
<code>\erw_cs_gset_inline:Nn</code> <hr/>	
<code>\erw_cs_gset_inline:cn</code> <hr/>	
<hr/> <code>\erw_identity:n</code> <hr/>	<code>\erw_identity:n{<arg>}</code>
	Expands to <code><arg></code>

<code>\erw_is_matrix_p:n</code>	<code>\erw_is_matrix_p:n{<token list>}</code>
<code>\erw_is_matrix:nTF</code>	Checks if <code><token list></code> is a (square) matrix.

<code>\erw_fold:N</code>	<code>\erw_fold:NV<cs><var></code>
<code>\erw_fold:cV</code>	<code><var>←\erw_apply:NV<cs><var></code> . See Listing 7.

<code>\erw_last_item:nn</code>	<code>\erw_last_item:nn{<int>}{<token list>}</code>
--------------------------------	---

<code>\erw_merge:nn</code>	<code>\erw_merge:nn{<tl 1>}{<tl 2>}</code>
	Merges <code><tl 1></code> <code><tl 2></code>

<code>\erw_repeat:nn</code>	<code>\erw_repeat:nn{<int>}{<value>}</code>
	See Listing 9

<code>\erw_split:nn</code>	<code>\erw_split:nn{<token list>}{<delimiter>}</code>
	See Listing 10

1.3 int

<code>\erw_int_range:nn</code>	<code>\erw_int_range:nn{<first>}last</code>
	Returns a range of integers. Implementation different than <code>\int_step_inline</code>

<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

1.4 map

<code>\erw_set_map:N</code>	<code>\erw_set_map:N<cs></code>
<code>\erw_gset_map:N</code>	Sets the function used by <code>\erw_map:n</code> .

<code>\erw_set_map_inline:n</code>	<code>\erw_set_map_inline:n{<code>}</code>
<code>\erw_gset_map_inline:n</code>	Sets the function used by <code>\erw_map:n</code> .

<code>\erw_map:n</code>	<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>

<code>\erw_map:Nn</code>	<code>\erw_map:Nn<cs>{<token list>}</code>
	See Listing 12. Redundant with <code>\tl_map_function:nN</code>

<code>\erw_map_inline:nn</code>	<code>\erw_map_inline:nn{<code>}{<args>}</code>
	See Listing 13

<hr/> <hr/>	<code>\erw_map_indexed:Nnn</code>	<code>\erw_map_indexed:Nnn<cs>{\langle int \rangle}{\langle matrix of tokens \rangle}</code> Not implemented. See Listing 15.
<hr/> <hr/>	<code>\erw_map_thread:Nn</code>	<code>\erw_map_thread:Nn<cs>{\langle matrix of tokens \rangle}</code> Threads $\langle cs \rangle$ over the columns, where the arity of $\langle cs \rangle$ must be equal to the number of rows. See Listing 14
<hr/> <hr/>	<code>\erw_map_thread_at:Nnn</code>	<code>\erw_map_thread_at:Nnn<cs>{\langle matrix of tokens \rangle}</code>

1.5 numbrdcs

Part of these commands have a frontend counterpart, see subsection 2.2.

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

2 frontend

We call frontend commands those created with `xparse's \NewDocumentCommand`^[2]

2.1 disambig

<hr/> <hr/>	<code>\disambignewcmd</code> <code>\disambignewcmd*</code>	<code>\disambignewcmd{\langle token \rangle}{\langle pars \rangle}{\langle code \rangle}</code> Analogues of <code>\NewDocumentCommand</code> and <code>\RenewDocumentCommand</code> . See Listing 17
<hr/> <hr/>	<code>\disambignewenv</code> <code>\disambignewenv*</code>	<code>\disambignewenv{\langle token \rangle}{\langle pars \rangle}{\langle code1 \rangle}{\langle code2 \rangle}</code> Analogues of <code>\NewDocumentEnvironment</code> and <code>\RenewDocumentEnvironment</code> . See Listing 18
<hr/> <hr/>	<code>\disambigset</code>	<code>\disambigset{\langle prefix \rangle}</code>
<hr/> <hr/>	<code>\disambigunset</code>	<code>\disambigunset{}_</code>

2.2 numbrdcs

<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 19
<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 19

Part II

Listings

1 Backend

1.1 compose

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
```

1.2 csutil

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}}{==}          a==b==c
\ExplSyntaxOff
```

1.3 int

Listing 11

```
\ExplSyntaxOn
\erw_int_range:nn{2}{5}      2345
\erw_int_range:n{5}          12345
\ExplSyntaxOff
```

1.4 map

Listing 12

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

Listing 13

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

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
```

1.5 numbrdcs

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)]}
```

2 Frontend

2.1 disambig

Listing 17

Input

```
\disambigset{my}  
\disambignewcmd{\foo}{m}{#1~world!}  
\noindent\myfoo{Hello}  
\disambignewcmd*{\foo}{m}{#1~universe!}  
\myfoo{Hello}  
\disambigunset  
\disambignewcmd{\foo}{m}{#1~world!}  
\myfoo{Hello}
```

Output

```
Hello world!  
Hello universe!  
Hello world!
```

Listing 18

Input

```
\disambigset{my}  
\disambignewenv{bar}{}{H}{!}  
\noindent\begin{mybar}ello~world\end{mybar}  
\disambignewenv*{bar}{}{J}{!}  
\begin{mybar}ello~world\end{mybar}
```

Output

```
Hello world!  
Jello world!
```

2.2 numbrdcs

Listing 19

```

\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 Back end

1.1 compose

```

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 }

```

1.2 csutil

```

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 }

```

1.3 map

```

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 }

```

1.4 map

```

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 }

```

1.5 numbrdcs

```

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 }

```

2 frontend

2.1 disambig

```

380 \cs_set:Npn \__erw_disambig:NN #1 #2 {#1{#2}}
381 \cs_generate_variant:Nn \__erw_disambig:NN { Nc }
382 \NewDocumentCommand{\disambignewcmd}{ s m m m }
383 {
384     \msg_error:nnn{erw}{generic}{disambignewcmd~undefined}
385 }

```

```

386 \NewDocumentCommand{\disambignewenv}{ s m m m m }
387 {
388   \msg_error:nnn{erw}{generic}{disambignewenv~undefined}
389 }
390 \keys_define:nn { erw }
391 {
392   disambig .code:n =
393   {
394     \RenewDocumentCommand{\disambignewcmd}{ s m m m }
395     {
396       \IfBooleanTF{##1}
397       { \__erw_disambig:Nc{\RenewDocumentCommand}}
398       { \__erw_disambig:Nc{\NewDocumentCommand}}
399       {#1 \__erw_cs_name:N ##2}
400       {##3}
401       {##4}
402     }
403   \RenewDocumentCommand{\disambignewenv}{ s m m m m }
404   {
405     \IfBooleanTF{##1}
406     { \RenewDocumentEnvironment }
407     { \NewDocumentEnvironment }
408     {#1##2}
409     {##3}
410     {##4}
411     {##5}
412   }
413 },
414   disambig .initial:n = \c_empty_tl
415 }
416 \NewDocumentCommand{\disambigset}{ m }
417 {
418   \keys_set:nn { erw }
419   {
420     disambig={#1}
421   }
422 }
423 \NewDocumentCommand{\disambigunset}{ }
424 {
425   \disambigset{\c_empty_tl}
426 }

```

2.2 numbrdcs

```

427 \NewDocumentCommand{\numbrdcsnew}{ s m }
428 {
429   \IfBooleanTF{#1}
430   { }
431   { \erw_numbrd_cs_reset:{} }
432   \tl_map_function:nN {#2}\erw_numbrd_cs_new:n
433 }
434 \NewDocumentCommand{\numbrdcs}{ m m }
435 {
436   \erw_numbrd_cs:nn{#1}{#2}
437 }

```

Part IV

Other

1 Support

This package is available from <https://www.ctan.org/pkg/erw-l3> (release) or <https://github.com/er-cpp/erw-l3> (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>

```

1 % \ProcessKeysPackageOptions{ erw }
2 \ExplSyntaxOff

```

Change History

0.1		Brought all the modules under one file; renamed l3erw to erw-l3;	23
General: Initial version	23		
0.1.1		0.1.2	
General:	23	General:	23
\numbrdcsnew changed to \newnumbrdcs and made 'disambiguable'	23	\erw_compose reversed order in which the functions are composed, such that it now conforms to the mathematical convention ($g \circ f$ means f comes before g)	23
disambig/backend: changes to the key, added			
\ProcessPackageKeysOption; . . .	23	disambig: pushed the code inside	

\keys_define;\disambignewcmd	0.1.3
no longer takes a token name as	General: Wrong versioning, should
arg, rather a token. 23	have been 0.1.2 23
Added \erw_items_to 23	0.1.4
Added \erw_last_item 23	General: 23
Added \erw_repeat 23	Added \erw_accum 23
Added \erw_split 23	Added \erw_int_range 23
Added \map_thread 23	Added \erw_is_matrix 23
Front end cmds no longer generated	Added \erw_merge 23
with module disambig; Option of	Added \erw_set_map_inline 23
the same name deleted; 23	Added \erw_set_map 23
Re-arranged the doc to clearly	Removed \erw_items_to
separate frontend from backend .. 23	(redundant with \tl_range:nnn) . 23

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	
\cs 150	\erw_apply:Nnnnn 3, 281, 305
cs commands:	\erw_compose 1, 23, 23
\cs_generate_variant:Nn	\erw_compose:nn 3, 3, 23, 32, 35
... 85, 90, 95, 100, 105, 124, 134, 381	\erw_compose:Nnn 7, 25, 29, 46, 52
\cs_gset:Npn 93, 103	\erw_compose_c:nn 3, 27, 38, 41, 67
\cs_new:Npn 252, 367, 371, 376	\erw_compose_seq:nn 3, 44
\cs_new_protected:Npn 351, 356	\erw_compose_seq_c:nn 3, 48
\cs_set:Npn .. 7, 23, 27, 32, 38, 44,	\erw_compose_seq_vers:nn 3, 58
48, 54, 58, 62, 71, 75, 79, 86, 88, 91,	\erw_compose_vers:nn 3, 54, 62
96, 98, 101, 108, 112, 116, 126, 135,	\erw_cs_gset_eq:NN 3, 91, 95, 237
151, 166, 173, 181, 185, 189, 200,	\erw_cs_gset_inline:Nn 3, 101, 105, 241
204, 225, 229, 235, 239, 243, 247,	\erw_cs_set_eq:NN 3, 86, 90, 263
259, 266, 273, 277, 281, 285, 290,	\erw_cs_set_inline:Nn
296, 303, 311, 332, 344, 345, 347, 380 3, 14, 96, 100, 270, 359
\cs_split_function:N	\erw_fold:Nn .. 4, 25, 29, 116, 124, 131
77	\erw_fold_apply_par:n 13, 112
D	
\disambignewcmd 5, 24, 382, 394	\erw_fold_seq:Nn 46, 52, 126, 134
\disambignewcmd*	\erw_fold_set_par:n 12, 108
5	\erw_gset_map:N
\disambignewenv 5, 386, 403	4, 235
\disambignewenv*	\erw_gset_map_inline:n ... 4, 138, 239
5	\erw_identity:n
\disambigset 5, 416, 425	3, 135
\disambigunset 5, 423	\erw_int_range 24
E	
erw commands:	\erw_int_range:n 4, 229
\erw_accum 24	\erw_int_range:nn 4, 225
\erw_accum:nn 3, 71, 216	\erw_is_matrix 4, 24
\erw_apply:Nn ... 3, 4, 79, 85, 287, 349	\erw_is_matrix:n 136
\erw_apply:Nnn 3, 273, 292	\erw_is_matrix:nTF
\erw_apply:Nnnn 3, 277, 298	4
	\erw_is_matrix_p:n
	4
	\erw_items_to
	24
	\erw_items_to:nn
	166
	\erw_last_item
	24

<code>\ProcessPackageKeysOption</code>	23	<code>\tl_count:n</code>	138, 141, 178, 315, 337
Q		<code>\tl_function_map:Nn</code>	5
quark commands:		<code>\tl_head:n</code>	141, 156, 163, 191, 337
<code>\quark_if_recursion_tail_stop:n</code>	249	<code>\tl_item:nn</code>	175, 288, 293,
<code>\q_recursion_stop</code>	245	294, 299, 300, 301, 306, 307, 308, 309	
<code>\q_recursion_tail</code>	245	<code>\tl_map_function:nN</code>	4, 65, 432
R		<code>\tl_map_inline:nn</code>	192
<code>\RenewDocumentCommand</code>	5, 394, 397, 403	<code>\tl_new:N</code>	31, 125
<code>\RenewDocumentEnvironment</code>	5, 406	<code>\tl_range:nnn</code>	24
<code>\RequirePackage</code>	2, 3, 4	<code>\tl_range_braced:nnn</code>	374
S		<code>\tl_reverse:n</code>	20
seq commands:		<code>\tl_set:Nn</code>	
<code>\seq_get_right:NN</code>	130	34, 40, 106, 107, 110, 114, 346, 354
<code>\seq_put_right:Nn</code>	132	<code>\tl_tail:n</code>	144, 160, 194
T		token commands:	
tl commands:		<code>\token_if_cs:NTF</code>	361
<code>\c_empty_tl</code>	414, 425	U	
<code>\c_novalue_tl</code>	106, 107	use commands:	
		<code>\use:N</code>	120, 122, 192
		<code>\use_i:nnn</code>	77
		<code>\usepackage</code>	2