

erw-l3*

Erwann Rogard[†]

Released 2018/05/23

Abstract

L^AT_EX3 package defining narrow-purpose commands built around `expl3`[\[1\]](#).

1 Intro

This package consists of the following modules

1. `compose` Musings in recursion. Think $f_1 \circ f_2 \cdots \circ f_n$, where the f_i 's are either pre- or inline-defined commands
1. `csutil` Narrow-purpose, handy, commands
1. `disambig` Wrapper around `\NewDocumentCommand`[\[3\]](#) to prevent name conflicts with existing commands.
1. `numbrdcs` Numbered commands built from other commands or inline

Part I

Usage

We call front-end commands those that are for typesetting, and back-end commands those that go into the code of front-end commands. The first and second are recognizable by the absence and presence, respectively, of the prefix `erw_` in, and `_` and `:` inside, their identifier (a.k.a control sequence name). See [\[2, Part l3bootstrap\]](#).

1 Getting started

Make sure the file `erw-l3.sty` is in the path of the L^AT_EX engine. Load the package as follows in the preamble of the document:

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

2 Options

`disambig=<prefix>`

A prefix that is added to front-end command names, should they conflict with existing commands. For all the modules, except the `disambig` module itself.

3 csutil

<code>\erw_apply:Nn</code>	<code>\erw_apply:Nn<cs>{<arg>}</code>
<code>\erw_apply:cn</code>	Expands <code><cs>{<arg>}</code>

<code>\erw_cs_set_eq:NN</code>	<code>\erw_cs_set_eq:NN<cs1><cs2></code>
<code>\erw_cs_set_eq:cN</code>	<code><cs1>←<cs2></code>

<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_fold:NV</code>	<code>\erw_fold:NV<cs><var></code>
<code>\erw_fold:cV</code>	<code><var>←\erw_apply:NV<cs><var></code> . See Listing 9.

<code>\erw_map:Nn</code>	<code>\erw_map:Nn<cs>{<args>}</code>
	See Listing 10. Redundant with <code>\tl_map_function:nN</code> (but I use it to access internals in another package).

<code>\erw_map_inline:nn</code>	<code>\erw_map_inline:nn{<code>}{<args>}</code>
	See Listing 11

4 compose

<code>\erw_compose:nV</code>	<code>\erw_compose:nV{<cs list>}<var></code>
<code>\erw_compose:nn</code>	See Listing 3

<code>\erw_compose_c:nV</code>	<code>\erw_compose_c:nV{<cs names>}<var></code>
<code>\erw_compose_c:nn</code>	See Listing 4

<code>\erw_compose_seq:nV</code>	<code>\erw_compose_seq:nV{<cs list>}<seq></code>
	See Listing 5

<code>\erw_compose_seq_c:nV</code>	<code>\erw_compose_seq_c:nV{<cs names>}<seq></code>
	See Listing 6

*This file describes version v0.1.1, last revised 2018/05/23.

†firstname dot lastname AusTria gmail dot com

<u>\erw_compose_vers:nV</u>	\erw_compose_vers:nV{<list of cs or code>}{<var>}
<u>\erw_compose_vers:nn</u>	See Listing 7. Only the nn version is implemented
<u>\erw_compose_seq_vers:nV</u>	\erw_compose_seq_vers:nV{<list of cs or code>}{<seq>}
<u>\erw_compose_seq_vers:nn</u>	Not implemented.

5 disambig

<u>\disambigset</u>	\disambigset{<prefix>}
	See Listing 12
<u>\disambignewcmd</u> <u>\disambignewcmd*</u>	\disambignewcmd{<cs name>}{<pars>}{<code>}
	See Listing 13
<u>\disambignewenv</u> <u>\disambignewenv*</u>	\disambignewenv{<env name>}{<pars>}{<code1>}{<code2>}
	See Listing 14

6 numbrdcs

<u>\numbrdcsnew</u> <u>\numbrdcsnew*</u>	\numbrdcsnew{<list of cs or code>}
	Creates numbered control sequences. The starred version does not reset. See Listing 15
<u>\numbrdcs</u>	\numbrdcs{<int>}{<arg>}
	Evaluates control sequence numbered <int> with argument <arg>. See Listing 15
<u>\erw_numbrd_cs_reset:</u>	\erw_numbrd_cs_reset:{} See Listing 16
<u>\erw_numbrd_cs_new:n</u>	\erw_numbrd_cs_new:n {<cs or code>}
	Use it as the first arg to \tl_function_map:Nn
<u>\erw_numbrd_cs:nn</u>	\erw_numbrd_cs:nn {<cs or code>}
<u>\erw_numbrd_cs_names_braced:nnn</u>	\erw_numbrd_cs_names_braced:nnn{<first>}{<step>}{<last>}
	See Listing 16

Part II

Listings

Listing 1 Initialization

```
\NewDocumentCommand{\myfoo}{m}{f(#1)}
\NewDocumentCommand{\mybar}{m}{g(#1)}
\NewDocumentCommand{\mybaz}{m}{h(#1)}
```

Listing 2 Initialization

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

1 compose

Listing 3

```
\tl_set:Nn \l_tmpa_tl{X}
\erw_compose:nV{
  \__foo\__bar\__baz}
  \l_tmpa_tl
\l_tmpa_tl h{g[f(X)]}
\tl_set:Nn \l_tmpa_tl{X}
\erw_compose:nn{
  \__foo\__bar\__baz}
  {X} h{g[f(X)]}
```

Listing 4

```
\tl_set:Nn \l_tmpa_tl{X}
\erw_compose_c:nV{
  \__foo\__bar\__baz}
  \l_tmpa_tl
\l_tmpa_tl h{g[f(X)]}
\erw_compose_c:nn{
  \__foo\__bar\__baz}
  {X} h{g[f(X)]}
```

Listing 5

<code>\seq_new:N\l_tmp_seq</code>	
<code>\seq_put_right:Nn\l_tmp_seq{X}</code>	
<code>\l_tmp_seq</code>	
<code>\seq_item:Nn\l_tmp_seq{1}</code>	X
<code>\seq_item:Nn\l_tmp_seq{2}</code>	$f(X)$
<code>\seq_item:Nn\l_tmp_seq{3}</code>	$g[f(X)]$
<code>\seq_item:Nn\l_tmp_seq{4}</code>	$h\{g[f(X)]\}$

Listing 6

<code>\seq_new:N\l_tmp_seq</code>	
<code>\seq_put_right:Nn\l_tmp_seq{X}</code>	
<code>\erw_compose_seq_c:nV{</code>	
<code> {__foo}{__bar}{__baz}}</code>	
<code>\l_tmp_seq</code>	
<code>\seq_item:Nn\l_tmp_seq{1}</code>	X
<code>\seq_item:Nn\l_tmp_seq{2}</code>	$f(X)$
<code>\seq_item:Nn\l_tmp_seq{3}</code>	$g[f(X)]$
<code>\seq_item:Nn\l_tmp_seq{4}</code>	$h\{g[f(X)]\}$

Listing 7

<code>\erw_compose_vers:nn{</code>	
<code> {__foo}{g[#1]}{__baz}}</code>	
<code> {X}</code>	$h\{g[f(X)]\}$

2 csutil

Listing 8

<code>\ExplSyntaxOn</code>	
<code>\erw_apply:Nn__foo{X}</code>	$f(X)$
<code>\ExplSyntaxOff</code>	

Listing 9

```
\ExplSyntaxOn
\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 10

```
\ExplSyntaxOn
\erw_map:Nn \__foo{{a}{b}{c}} (a)(b)(c)
\ExplSyntaxOff
```

Listing 11

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

3 disambig

Listing 12

Input

```
\disambigset{my}
```

Output

Listing 13

Input

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

Output

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

Listing 14

Input

```
\disambignewenv{bar}{}{\textrightarrow}{\textleftarrow}
\begin{mybar}
  Hello~world
\end{mybar}
\disambignewenv*{bar}{}{>}{<}
\\begin{mybar}
  Hello~world
\end{mybar}
```

Output

```
→ Hello world ←
> Hello world <
```

4 numbrdcs

Listing 15

```
\numbrdcsnew*{\myfoo}{g[#1]}\mybaz}}

\numbrdcs{1}{X}          f(X)
\numbrdcs{2}{X}          g[X]
\numbrdcs{3}{X}          h(X)
\numbrdcsnew*{\myfoo}{g[#1]}\mybaz}}

\numbrdcs{4}{X}          f(X)
\numbrdcs{5}{X}          g[X]
\numbrdcs{6}{X}          h(X)
```

Listing 16

```
\ExplSyntaxOn
\exp_last_unbraced:Nx
  \erw_compose_c:nn
  {
    {\erw_numbrd_cs_names
      _braced:nnn{1}{1}{3}}
    {X}
  }
\ExplSyntaxOff          h(g[f(X)])
```

Part III

Other

1 Acknowledgment

The idea to create `l3erw-numbrdcs` arose while developing `l3erw-compose` and stumbling upon a problem discussed in [4]. The use of `\exp_last_unbraced:Nx` originated in [5].

References

- [1] The L^AT_EX3 Project Team *The expl3 package and L^AT_EX3 programming* <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3kernel/expl3.pdf>
- [2] 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>
- [3] The L^AT_EX3 Project Team *The xparse package* <http://ftp.math.purdue.edu/mirrors/ctan.org/macros/latex/contrib/l3packages/xparse.pdf>
- [4] <https://tex.stackexchange.com/questions/431046/calling-expl3s-usec-on-an-expression-expanding-to-a-cs-name-causes-error>
- [5] <https://tex.stackexchange.com/questions/432171/expl3-making-arguments-from-a-loop>

Part IV

Implementation

```
1 \NeedsTeXFormat{LaTeX2e}
2 \RequirePackage{expl3}[2018/02/21]
3 \RequirePackage{xparse}[2018/02/21]
4 \RequirePackage{l3keys2e}
5 \ExplSyntaxOn
```

1 compose

```
6 \msg_new:nnn{erw_compose}{generic}{#1}
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   \erw_map:n{#2}
19 }
20 \cs_set:Npn \erw_compose:nV #1 #2
21 {
22   \erw_compose:NnV \erw_fold:NV {#1} #2
23 }
24 \cs_set:Npn \erw_compose_c:nV #1 #2
25 {
26   \erw_compose:NnV \erw_fold:cV {#1} #2
27 }
28 \tl_new:N \__erw_compose_tl
29 \cs_set:Npn \erw_compose:nn #1 #2
30 {
31   \tl_set:Nn \__erw_compose_tl {#2}
32   \erw_compose:nV{#1}\__erw_compose_tl
33   \__erw_compose_tl
34 }
35 \cs_set:Npn \erw_compose_c:nn #1 #2
36 {
37   \tl_set:Nn \__erw_compose_tl {#2}
38   \erw_compose_c:nV{#1}\__erw_compose_tl
39   \__erw_compose_tl
40 }
41 \tl_new:N \__erw_fold_seq_item_tl
42 \cs_set:Npn \erw_fold_seq:NV
43   #1 % fun
44   #2 % seq
45 {
46   \seq_get_right:NN #2 \__erw_fold_seq_item_tl
47   \erw_fold:NV #1 \__erw_fold_seq_item_tl
48   \seq_put_right:No #2 {\__erw_fold_seq_item_tl}
49 }
50 \cs_generate_variant:Nn \erw_fold_seq:NV {cV}
51 \cs_set:Npn \erw_compose_seq:nV #1 #2
52 {
53   \erw_compose:NnV \erw_fold_seq:NV {#1} #2
54 }
55 \cs_set:Npn \erw_compose_seq_c:nV
56   #1 % funs
57   #2 % seq
58 {
59   \erw_compose:NnV \erw_fold_seq:cV {#1} #2
60 }
61 \cs_set:Npn \erw_compose_vers:nV #1 #2
62 {
63   \msg_error:nnn{erw_rec}{generic}{erw_compose_vers:nV~to~be~defined}
64 }
65 \cs_set:Npn \erw_compose_seq_vers:nV #1 #2
66 {
67   \msg_error:nnn{erw_rec}{generic}{erw_compose_seq_vers:nV~to~be~defined}
68 }
69 \cs_set:Npn \erw_compose_vers:nn #1 #2
70 {
71   \erw_numbrd_cs_reset:{}

```

```

72     \tl_map_function:nn{#1}\erw_numbrd_cs_new:n
73     \exp_last_unbraced:Nx
74     \erw_compose_c:nn
75     {{\erw_numbrd_cs_names_braced:{}}}
76     {#2}
77 }

```

2 disambig

```

78 \tl_new:N \__erw_disambig_tl
79 \keys_define:nn { erw }
80 {
81   disambig .tl_set:N = \__erw_disambig_tl,
82   disambig .initial:n = \c_empty_tl
83 }
84 \cs_set:Npn \__erw_disambig:NN #1 #2 {#1{#2}}
85 \cs_generate_variant:Nn \__erw_disambig:NN { Nc }
86 \NewDocumentCommand{\disambignewcmd}{ s m m m }
87 {
88   \IfBooleanTF{#1}
89     {\__erw_disambig:Nc{\RenewDocumentCommand}}
90     {\__erw_disambig:Nc{\NewDocumentCommand}}
91     {\__erw_disambig_tl #2}
92     {#3}
93     {#4}
94 }
95 \NewDocumentCommand{\disambignewenv}{ s m m m m }
96 {
97   \IfBooleanTF{#1}
98     {\RenewDocumentEnvironment}
99     {\NewDocumentEnvironment}
100   {\__erw_disambig_tl #2}
101   {#3}
102   {#4}
103   {#5}
104 }
105 \NewDocumentCommand{\disambigset}{ m }
106 {
107   \keys_set:nn { erw }
108   {
109     disambig={#1}
110   }
111 }
112 \ProcessKeysPackageOptions{ erw }

```

3 csutil

```

113 \msg_new:nnn
114   {erw_csutil}
115   {generic}
116   {#1}
117 \cs_set:Npn \erw_cs_set_eq:NN #1 #2
118 {
119   \cs_set:Npn #1 ##1{#2{##1}}

```

```

120 }
121 \cs_generate_variant:Nn \erw_cs_set_eq:NN {cN}
122 \cs_set:Npn \erw_cs_set_inline:Nn #1 #2
123 {
124   \cs_set:Npn #1 ##1{#2}
125 }
126 \cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}
127 \cs_set:Npn \erw_map:n #1
128 {
129   \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
130 }
131 \cs_set:Npn \__erw_map:nn #1 #2
132 {
133   \quark_if_recursion_tail_stop:n{#1}
134   \__erw_map:n{#1} \__erw_map:nn{#2}
135 }
136 \cs_new:Npn \__erw_map:n #1
137 {
138   \msg_error:nnn
139     {erw_csutil}
140     {generic}
141     {\__erw_map:n~not~set}
142 }
143 \cs_set:Npn \erw_map:Nn
144   #1 % fun
145   #2 % tl
146 {
147   \erw_cs_set_eq:NN \__erw_map:n #1
148   \erw_map:n{#2}
149 }
150 \cs_set:Npn \erw_map_inline:nn
151   #1 % inl
152   #2 % tl
153 {
154   \erw_cs_set_inline:Nn \__erw_map:n {#1}
155   \erw_map:n{#2}
156 }
157 \cs_set:Npn \erw_apply:Nn
158   #1 % fun
159   #2 % tl
160 {
161   #1{#2}
162 }
163 \cs_generate_variant:Nn \erw_apply:Nn {No, Nf, Nx, c}
164 \tl_set:Nn \__erw_fold_set_par_tl{c_novalue_tl}
165 \tl_set:Nn \__erw_fold_apply_par_tl{c_novalue_tl}
166 \cs_set:Npn \erw_fold_set_par:n #1
167 {
168   \tl_set:Nn \__erw_fold_set_par_tl{#1}
169 }
170 \cs_set:Npn \erw_fold_apply_par:n #1
171 {
172   \tl_set:Nn \__erw_fold_apply_par_tl{#1}
173 }

```

```

174 \cs_set:Npn \erw_fold:NV
175   #1 % fun
176   #2 % var
177 {
178   \use:c{tl_set:\_erw_fold_set_par_tl}
179     #2
180     {\use:c{erw_apply:\_erw_fold_apply_par_tl}{#1}{#2}}
181 }
182 \cs_generate_variant:Nn \erw_fold:NV {cV}

```

4 numbrdcs

```

183 \disambignewcmd{numbrdcsnew}{ s m }
184 {
185   \IfBooleanTF{#1}
186   {}
187   { \erw_numbrd_cs_reset:{} }
188   \tl_map_function:nN {#2}\erw_numbrd_cs_new:n
189 }
190 \disambignewcmd{numbrdcs}{ m m }
191 {
192   \erw_numbrd_cs:nn{#1}{#2}
193 }
194 \msg_new:nnn
195   {erw_numbrdcs}
196   {generic}
197   {#1}
198 \int_new:N \_erw_numbrd_cs_int
199 \cs_set:Npn \erw_numbrd_cs_name:n #1{\_erw_numbrd_cs\_int_to_alph:n{#1}:n}
200 \cs_set:Npn \erw_numbrd_cs_name_braced:n #1{{\erw_numbrd_cs_name:n{#1}}}
201 \tl_set:Nn \_erw_numbrd_cs_name_tl {\erw_numbrd_cs_name:n{\_erw_numbrd_cs_int}}
202 \cs_set:Npn \erw_numbrd_cs:nn #1 #2
203 {
204   \erw_apply:cn{\_erw_numbrd_cs\_int_to_alph:n{#1}:n}{#2}
205 }
206 \cs_new_protected:Npn \erw_numbrd_cs_reset:
207 {
208   \int_zero:N \_erw_numbrd_cs_int
209   \tl_set:Nn \_erw_numbrd_cs_ext_tl{}
210 }
211 \cs_new_protected:Npn \erw_numbrd_cs_new:n #1
212 {
213   \int_incr:N \_erw_numbrd_cs_int
214   \erw_cs_set_inline:cn{\_erw_numbrd_cs_name_tl}
215   {
216     \token_if_cs:NTF
217     {#1}
218     {#1{##1}}
219     {#1}
220   }
221 }
222 \cs_new:Npn \erw_numbrd_cs_names:nnn #1 #2 #3
223 {
224   \int_step_function:nnnN { #1 } { #2 } { #3 } \erw_numbrd_cs_name:n

```

```

225 }
226 \cs_new:Npn \erw_numbrd_cs_names_braced:nnn #1 #2 #3
227 {
228     \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_numbrd_cs_name_braced:n
229     % TODO \tl_range_braced:nnn?
230 }
231 \cs_new:Npn \erw_numbrd_cs_names_braced:
232 {
233     \erw_numbrd_cs_names_braced:nnn{1}{1}{\_erw_numbrd_cs_int}
234 }
235 \ExplSyntaxOff

```