erw-l3*

Erwann Rogard†

Released 2018/05/20

Abstract

LaTeX3 package defining narrow-purpose commands built around expl3.

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` 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 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 Part 'l3bootstrap' of [**?**].

1 Getting started

Make sure the file `erw-l3.sty` is in the path of the LaTeX 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

`\erw_apply:Nn` `\erw_apply:cn`
`\erw_apply:Nn`⟨*cs*⟩{⟨*args*⟩}

`\erw_cs_set_eq:NN` `\erw_cs_set_eq:cN`
`\erw_cs_set_eq:NN`⟨*cs*⟩⟨*cs*⟩

`\erw_cs_set_inline:Nn` `\erw_cs_set_inline:cn`
`\erw_cs_set_inline:Nn`⟨*cs*⟩{⟨*code*⟩}

`\erw_fold:NV` `\erw_fold:cV`
`\erw_fold:NV`⟨*cs*⟩⟨*var*⟩
See Listing 9

`\erw_map:Nn`
`\erw_map:Nn`⟨*cs*⟩{⟨*args*⟩}
See Listing 10. Redundant with `\tl_map_function:nN` (but I use it to access internals in another package).

`\erw_map_inline:nn`
`\erw_map_inline:nn`{⟨*code*⟩}{⟨*args*⟩}
See Listing 11

4 compose

`\erw_compose:nV` `\erw_compose:nn`
`\erw_compose:nV`{⟨*cs list*⟩}⟨*var*⟩
See Listing 3

`\erw_compose_c:nV` `\erw_compose_c:nn`
`\erw_compose_c:nV`{⟨*cs names*⟩}⟨*var*⟩
See Listing 4

`\erw_compose_seq:nV`
`\erw_compose_seq:nV`{⟨*cs list*⟩}⟨*seq*⟩
See Listing 5

`\erw_compose_seq_c:nV`
`\erw_compose_seq_c:nV`{⟨*cs names*⟩}⟨*seq*⟩
See Listing 6

*This file describes version v0.1.1, last revised 2018/05/20.
†firstname dot lastname AusTria gmail dot com

\erw_compose_vers:nV
\erw_compose_vers:nn

\erw_compose_vers:nV{⟨*list of cs or code*⟩}⟨*var*⟩

See Listing 7. Only the nn version is implemented

\erw_compose_seq_vers:nV
\erw_compose_seq_vers:nn

\erw_compose_seq_vers:nV{⟨*list of cs or code*⟩}⟨*seq*⟩

Not implemented.

5 disambig

\disambigset

\disambigset{⟨*prefix*⟩}

See Listing 12

\disambignewcmd
\disambignewcmd*

\disambignewcmd{⟨*cs name*⟩}{⟨*pars*⟩}{⟨*code*⟩}

See Listing 13

\disambignewenv
\disambignewenv*

\disambignewenv{⟨*env name*⟩}{⟨*pars*⟩}{⟨*code1*⟩}{⟨*code2*⟩}

See Listing 14

6 numbrdcs

\numbrdcsnew
\numbrdcsnew*

\numbrdcsnew{⟨*list of cs or code*⟩}

Creates numbered control sequences. The starred version does not reset. See Listing 15

\numbrdcs

\numbrdcs{⟨*int*⟩}{⟨*arg*⟩}

Evaluates control sequence numbered ⟨*int*⟩ with argument ⟨*arg*⟩. See Listing 15

\erw_numbrd_cs_reset:

\erw_numbrd_cs_reset:{}

See Listing 16

\erw_numbrd_cs_new:n

\erw_numbrd_cs_new:n {⟨*cs or code*⟩}

Use it as the first arg to \tl_function_map:Nn

\erw_numbrd_cs:nn

\erw_numbrd_cs:nn {⟨*cs or code*⟩}

\erw_numbrd_cs_names_braced:nnn

\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

```
\seq_new:N\l_tmp_seq
\seq_put_right:Nn\l_tmp_seq{X}
  \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)]}
```

Listing 6

```
\seq_new:N\l_tmp_seq
\seq_put_right:Nn\l_tmp_seq{X}
\erw_compose_seq_c:nV{
  {__foo}{__bar}{__baz}}
  \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)]}
```

Listing 7

```
\erw_compose_vers:nn{
  {\__foo}{g[#1]}{\__baz}}
  {X}                              h{g[f(X)]}
```

2 csutil

Listing 8

```
\ExplSyntaxOn
\erw_apply:Nn\__foo{X}             f(X)
\ExplSyntaxOff
```

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 Credits

The idea to create l3erw-numbrdcs arose while developing l3erw-compose and stumbling upon a problem discussed in [2]. The use of `\exp_last_unbraced:Nx` originated in [3].

References

[1] The LaTeX3 Project Team *l3packages* http://mirror.ctan.org/macros/latex/contrib/l3packages/

[2] https://tex.stackexchange.com/questions/431046/calling-expl3s-usec-on-an-expression-expanding-to-a-cs-name-causes-error

[3] https://tex.stackexchange.com/questions/432171/expl3-making-arguments-from-a-loop

Part IV
Implementation

```
\NeedsTeXFormat{LaTeX2e}
\RequirePackage{expl3}[2018/02/21]
\RequirePackage{xparse}[2018/02/21]
\RequirePackage{l3keys2e}
\ExplSyntaxOn
```

1 compose

```
\msg_new:nnn{erw_compose}{generic}{#1}
\cs_set:Npn \erw_compose:NnV
  #1 % method
  #2 % funs
  #3 % var
{
  \erw_fold_set_par:n{Nf}
  \erw_fold_apply_par:n{Nf}
  \erw_cs_set_inline:Nn \__erw_map:n
  {
     #1{##1}#3
  }
  \erw_map:n{#2}
}
\cs_set:Npn \erw_compose:nV #1 #2
{
  \erw_compose:NnV \erw_fold:NV {#1} #2
}
```

```
\cs_set:Npn \erw_compose_c:nV #1 #2
{
  \erw_compose:NnV \erw_fold:cV {#1} #2
}
\tl_new:N \__erw_compose_tl
\cs_set:Npn \erw_compose:nn #1 #2
{
  \tl_set:Nn \__erw_compose_tl {#2}
  \erw_compose:nV{#1}\__erw_compose_tl
  \__erw_compose_tl
}
\cs_set:Npn \erw_compose_c:nn #1 #2
{
  \tl_set:Nn \__erw_compose_tl {#2}
  \erw_compose_c:nV{#1}\__erw_compose_tl
  \__erw_compose_tl
}
\tl_new:N \__erw_fold_seq_item_tl
\cs_set:Npn \erw_fold_seq:NV
  #1 % fun
  #2 % seq
{
  \seq_get_right:NN #2 \__erw_fold_seq_item_tl
  \erw_fold:NV #1 \__erw_fold_seq_item_tl
  \seq_put_right:No #2 {\__erw_fold_seq_item_tl}
}
\cs_generate_variant:Nn \erw_fold_seq:NV {cV}
\cs_set:Npn \erw_compose_seq:nV #1 #2
{
  \erw_compose:NnV \erw_fold_seq:NV {#1} #2
}
\cs_set:Npn \erw_compose_seq_c:nV
  #1 % funs
  #2 % seq
{
  \erw_compose:NnV \erw_fold_seq:cV {#1} #2
}
\cs_set:Npn \erw_compose_vers:nV #1 #2
{
   \msg_error:nnn{erw_rec}{generic}{erw_compose_vers:nV~to~be~defined}
}
\cs_set:Npn \erw_compose_seq_vers:nV #1 #2
{
   \msg_error:nnn{erw_rec}{generic}{erw_compose_seq_vers:nV~to~be~defined}
}
\cs_set:Npn \erw_compose_vers:nn #1 #2
{
   \erw_numbrd_cs_reset:{}
      \tl_map_function:nN{#1}\erw_numbrd_cs_new:n
      \exp_last_unbraced:Nx
      \erw_compose_c:nn
         {{\erw_numbrd_cs_names_braced:{}}}
         {#2}
}
```

2 disambig

```
\tl_new:N \__erw_disambig_tl
\keys_define:nn { erw }
{
  disambig .tl_set:N = \__erw_disambig_tl,
  disambig .initial:n = \c_empty_tl
}
\cs_set:Npn \__erw_disambig:NN #1 #2 {#1{#2}}
\cs_generate_variant:Nn \__erw_disambig:NN { Nc }
\NewDocumentCommand{\disambignewcmd}{ s m m m }
{
  \IfBooleanTF{#1}
      {\__erw_disambig:Nc{\RenewDocumentCommand}}
      {\__erw_disambig:Nc{\NewDocumentCommand}}
    {\__erw_disambig_tl #2}
    {#3}
    {#4}
}
\NewDocumentCommand{\disambignewenv}{ s m m m m }
{
  \IfBooleanTF{#1}
    {\RenewDocumentEnvironment}
    {\NewDocumentEnvironment}
  {\__erw_disambig_tl #2}
  {#3}
  {#4}
  {#5}
}
\NewDocumentCommand{\disambigset}{ m }
{
  \keys_set:nn { erw }
  {
        disambig={#1}
  }
}
\ProcessKeysPackageOptions{ erw }
```

3 csutil

```
\msg_new:nnn
  {erw_csutil}
  {generic}
  {#1}
\cs_set:Npn \erw_cs_set_eq:NN #1 #2
{
  \cs_set:Npn #1 ##1{#2{##1}}
}
\cs_generate_variant:Nn \erw_cs_set_eq:NN {cN}
\cs_set:Npn \erw_cs_set_inline:Nn #1 #2
{
  \cs_set:Npn #1 ##1{#2}
}
\cs_generate_variant:Nn \erw_cs_set_inline:Nn {cn}
```

```
\cs_set:Npn \erw_map:n #1
{
  \__erw_map:nn#1\q_recursion_tail\q_recursion_stop\q_recursion_tail\q_recursion_stop
}
\cs_set:Npn \__erw_map:nn #1 #2
{
  \quark_if_recursion_tail_stop:n{#1}
  \__erw_map:n{#1}  \__erw_map:nn{#2}
}
\cs_new:Npn \__erw_map:n #1
{
  \msg_error:nnn
    {erw_csutil}
    {generic}
    {__erw_map:n~not~set}
}
\cs_set:Npn \erw_map:Nn
  #1 % fun
  #2 % tl
{
  \erw_cs_set_eq:NN \__erw_map:n #1
  \erw_map:n{#2}
}
\cs_set:Npn \erw_map_inline:nn
  #1 % inl
  #2 % tl
{
  \erw_cs_set_inline:Nn \__erw_map:n {#1}
  \erw_map:n{#2}
}
\cs_set:Npn \erw_apply:Nn
  #1 % fun
  #2 % tl
{
  #1{#2}
}
\cs_generate_variant:Nn \erw_apply:Nn {No, Nf, Nx, c}
\tl_set:Nn \__erw_fold_set_par_tl{\c_novalue_tl}
\tl_set:Nn \__erw_fold_apply_par_tl{\c_novalue_tl}
\cs_set:Npn \erw_fold_set_par:n #1
{
  \tl_set:Nn \__erw_fold_set_par_tl{#1}
}
\cs_set:Npn \erw_fold_apply_par:n #1
{
  \tl_set:Nn \__erw_fold_apply_par_tl{#1}
}
\cs_set:Npn \erw_fold:NV
  #1 % fun
  #2 % var
{
  \use:c{tl_set:\__erw_fold_set_par_tl}
     #2
    {\use:c{erw_apply:\__erw_fold_apply_par_tl}{#1}{#2}}
```

```
}
\cs_generate_variant:Nn \erw_fold:NV {cV}
```

4 numbrdcs

```
\disambignewcmd{numbrdcsnew}{ s m }
{
\IfBooleanTF{#1}
{}
{ \erw_numbrd_cs_reset:{}}
\tl_map_function:nN {#2}\erw_numbrd_cs_new:n
}
\disambignewcmd{numbrdcs}{ m m }
{
\erw_numbrd_cs:nn{#1}{#2}
}
\msg_new:nnn
  {erw_numbrdcs}
  {generic}
  {#1}
\int_new:N \__erw_numbrd_cs_int
\cs_set:Npn \erw_numbrd_cs_name:n #1{__erw_numbrd_cs_\int_to_alph:n{#1}:n}
\cs_set:Npn \erw_numbrd_cs_name_braced:n #1{{\erw_numbrd_cs_name:n{#1}}}
\tl_set:Nn \__erw_numbrd_cs_name_tl {\erw_numbrd_cs_name:n{\__erw_numbrd_cs_int}}
\cs_set:Npn \erw_numbrd_cs:nn #1 #2
{
\erw_apply:cn{__erw_numbrd_cs_\int_to_alph:n{#1}:n}{#2}
}
\cs_new_protected:Npn \erw_numbrd_cs_reset:
{
\int_zero:N \__erw_numbrd_cs_int
\tl_set:Nn \__erw_numbrd_cs_ext_tl{}
}
\cs_new_protected:Npn \erw_numbrd_cs_new:n #1
{
\int_incr:N \__erw_numbrd_cs_int
\erw_cs_set_inline:cn{\__erw_numbrd_cs_name_tl}
{
\token_if_cs:NTF
{#1}
{#1{##1}}
{#1}
}
}
\cs_new:Npn \erw_numbrd_cs_names:nnn #1 #2 #3
{
    \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_numbrd_cs_name:n
}
\cs_new:Npn \erw_numbrd_cs_names_braced:nnn #1 #2 #3
{
    \int_step_function:nnnN { #1 }{ #2 }{ #3 } \erw_numbrd_cs_name_braced:n
    % TODO \tl_range_braced:nnn?
}
\cs_new:Npn \erw_numbrd_cs_names_braced:
```

```
{
    \erw_numbrd_cs_names_braced:nnn{1}{1}{\__erw_numbrd_cs_int}
}
\ExplSyntaxOff
```