

Meta-commands used in this document:

<i>italic text</i>	1 \textit{ italic text} \par
<code><placeholder></code>	2 \meta{placeholder} \par
<code>\command</code>	3 \cmd{command} \par
<code>[<bar>]{<foo>}</code>	4 \oarg{bar} \marg{foo} \par
<code>[bar]{foo}</code>	5 \oarg*{bar} \marg*{foo} \par
<code>\ic_funca:n</code>	6 \cmd{ic_funca:n} \par
<code>function</code>	7 \term{function} \par
<code>FUNCTION</code>	8 \term*{Function} \par

1 Introduction

VARIABLES

`\<scope>_<module>_<name>_<type>`

Scope	Description
l	local
g	global

Type	Description
bool	boolean
box	box
cctab	category code table
clist	clist
dim	dimension
flag	flag
fp	floating point
fparray	floating point array
int	integer
intarray	integer array
msg	message
mskip	mskip
prop	property list
seq	sequence
skip	skip
str	string
tl	token list

FUNCTIONS

`\<name><signature marker><signature>`

2 Basics

[2.1]

`\ExplSyntaxOn` % <-----
Some code.

```

Some code.
Some code.
\ExplSyntaxOff % <-----

Some code.
\group _begin: % <-----
Some code.
\group _end: % <-----
Some code.

```

[2.2]

[2.3] – Grouping

```

1 \ExplSyntaxOn
2 Text ~ text ~
3 \group _begin:
4 \color{red}
5 \large
6 text ~ text ~
7 \group _end:
8 text ~ text.
9 \ExplSyntaxOff

```

Text text **text text** text text.

[2.4] – Defining a function

```

1 \ExplSyntaxOn
2 \cs_new:Npn \ic_funca:n #1 { {\color{blue} \bfseries #1 } }
3
4 Text ~ \ic_funca:n { ~ This ~ is ~ blue ~ bold. } ~ Text.
5 \ExplSyntaxOff

```

Text **This is blue bold.** Text.

[2.5] – Defining a user command

```

1 \ExplSyntaxOn
2 \NewDocumentCommand { \mycldb } { m } {
3 \cs_new:Npn \ic_funcb:n ##1 { {\color{blue} \bfseries ##1 } }
4 \ic_funcb:n { #1 }
5 }
6 \ExplSyntaxOff
7 Text \mycldb{This is blue bold.} Text.

```

Text **This is blue bold.** Text.

[2.6] – Using a token list

```
1 \ExplSyntaxOn
2 \tl_new:N \l_myicc_tl
3 \NewDocumentCommand { \mycmdc } { m } {
4 \tl_set:Nn \l_myicc_tl { #1 }
5 \cs_new:Npn \ic_funcc:N ##1 { {\color{blue} \bfseries ##1 } }
6 \ic_funcc:N \l_myicc_tl
7 }
8 \ExplSyntaxOff
9 Text \mycmdc{This is blue bold.} Text.
```

Text **This is blue bold.** Text.

[2.7] – Appending to a token list

```
1 \ExplSyntaxOn
2 \tl_new:N \l_myicd_tl
3 \NewDocumentCommand { \mycmdd } { m } {
4 \tl_set:Nn \l_myicd_tl { #1 }
5 \tl_put_right:Nn \l_myicd_tl { $\leftarrow$ ! ! }
6 \tl_put_left:Nn \l_myicd_tl { ! ! $\rightarrow$ }
7 \cs_new:Npn \ic_funcd:N ##1 { {\color{blue} \bfseries ##1 } }
8 \ic_funcd:N \l_myicd_tl
9 }
10 \ExplSyntaxOff
11 Text \mycmdd{This is blue bold.} Text.
```

Text **!!→This is blue bold.←!!** Text.

[2.8] – Modifying a token list (\tl_replace)

```
1 \ExplSyntaxOn
2 \tl_new:N \l_myice_tl
3 \NewDocumentCommand { \mycmde } { m } {
4 \tl_set:Nn \l_myice_tl { #1 }
5 \tl_replace_all:Nnn \l_myice_tl { blue } { not ~ red }
6 \tl_replace_all:Nnn \l_myice_tl { bold } { italic }
7 \cs_new:Npn \ic_functe:N ##1 { {\color{blue} \bfseries ##1 } }
8 \ic_functe:N \l_myice_tl
9 }
10 \ExplSyntaxOff
11 Text \mycmde{This is blue bold.} Text.
```

Text **This is not red italic.** Text.

[2.9] – Modifying a token list (\regex_replace)

```
1 \ExplSyntaxOn
2 \tl_new:N \l_myicez_tl
3 \NewDocumentCommand { \mycmdez } { m } {
4 \tl_set:Nn \l_myicez_tl { #1 }
5 \regex_replace_all:nnN
6 { (\c{color} \cB{ [^cE\]}* \cE\)} }
7 { not \ }
8 \l_myicez_tl
9 \regex_replace_all:nnN
10 { \c{textbf}(.) (bold) }
11 { is\ \c{textsc}\1 small\ caps }
12 \l_myicez_tl
13 \tl_use:N \l_myicez_tl
14 }
15 \ExplSyntaxOff
16 Text \mycmdez{This is {\color{blue}blue} and \textbf{bold}.} Text.
```

Text This is not blue and is SMALL CAPS. Text.

[2.10] – Copying a control sequence

```
1 \ExplSyntaxOn
2 \cs_new:Npn \ic_funcf:N #1 { {\color{blue} \bfseries #1 } }
3 \cs_new_eq:NN \mybbcnd \ic_funcf:N
4 \ExplSyntaxOff
5 Text \mybbcnd{This is blue bold.} Text.
```

Text **This is blue bold.** Text.

[2.11] – Looping

```
1 \ExplSyntaxOn
2 \cs_set:Npn \ic_funcg:n #1 { \symbol{#1} }
3 \cs_set:Nn \ic_funcgb: {
4 \int_step_function:nnnN { 97 } { 1 } { 122 } \ic_funcg:n
5 \
6 }
7 \cs_new_eq:NN \myloopcmd \ic_funcgb:
8 \ExplSyntaxOff
9 Text \myloopcmd Text.
```

Text abcdefghijklmnopqrstuvwxyz Text.

COMMENTARY:

- (1) Open the expl3 environment.
- (2) Define a 1-parameter function, *g*, that will print a glyph, given the glyph's slot number, in the current font using the `\symbol` command.
- (3) Define a no-parameter function, *gb*, that will
- (4) step through values 97 to 122 (inclusive) and pass each value to the *g* function.
- (5) Add a space (replacing the one gobbled after the command in the user code).
- (6) –
- (7) Create a user-command, `\myloopcmd`, as a copy of the *gb* function.
- (8) Close the expl3 environment.
- (9) Use the user-command

RESULT: The letters a..z are printed, followed by a space.

[2.12] – Mapping function

```
1 \ExplSyntaxOn
2 \cs_set:Npn \ic_funch:n #1 { \fbox{#1}. }
3 \tl_new:N \l_myich_tl
4 \NewDocumentCommand { \mycmdh } { m } {
5 \tl_set:Nn \l_myich_tl { #1 }
6 \tl_map_function:NN \l_myich_tl \ic_funch:n
7 \par tl ~ = ~ >> \tl_use:N \l_myich_tl <<
8 }
9 \ExplSyntaxOff
10 Text \mycmdh{abc{de}fgh} Text.
```

Text abcdefgh.

tl = »abcdefgh« Text.

[2.13] – Mapping inline function

```
1 \ExplSyntaxOn
2 \cs_set:Npn \ic_funcn:n #1 { \fbox{#1}. }
3 \tl_new:N \l_myici_tl
4 \NewDocumentCommand { \mycmdi } { m } {
5 \tl_set:Nn \l_myici_tl { #1 }
6 \tl_map_inline:Nn \l_myici_tl { \ic_funcn:n {##1} }
7 \par tl ~ = ~ >> \tl_use:N \l_myici_tl <<
8 }
9 \ExplSyntaxOff
10 Text \mycmdi{abc{de}fgh} Text.
```

Text abcdefgh.

tl = »abcdefgh« Text.

[2.14] – Contents of a token list (1)

```

1 \ExplSyntaxOn
2 \tl_new:N \l_myicj_tl
3 \NewDocumentCommand { \mycmdj } { m } {
4 \tl_set:Nn \l_myicj_tl { #1 }
5 \par (a) ~ \tl_count:N \l_myicj_tl \ token ~ groups.
6 \par (b) ~ \tl_count_tokens:n { \l_myicj_tl } ~ token
    \int_compare:nNnTF {\tl_count_tokens:n { \l_myicj_tl }} = { 1
    } { } { s }.
7 \par (c) ~ \exp_args:No \tl_count_tokens:n { \l_myicj_tl } ~
    token \int_compare:nNnTF {\exp_args:No \tl_count_tokens:n {
    \l_myicj_tl }} = { 1 } { } { s } : ~
8 { \color{blue} \tl_to_str:N \l_myicj_tl }.
9 }
10 \ExplSyntaxOff
11 \mycmdj{abc{de}fgh}

```

- (a) 7 token groups.
- (b) 1 token.
- (c) 10 tokens: `abc{de}fgh`.

`\l_myicj_tl` will always be one token.

[2.15] – Contents of a token list (2)

```

1 \ExplSyntaxOn
2 \tl_new:N \l_myick_tl
3 \NewDocumentCommand { \mycmdk } { m } {
4 \tl_set:Nn \l_myick_tl { #1 }
5 \par head: ~ \tl_head:N \l_myick_tl
6 \par tail: ~ \tl_tail:N \l_myick_tl
7 \par reverse: ~ \tl_reverse:N \l_myick_tl \tl_use:N \l_myick_tl
  ~<~ \tl_to_str:N \l_myick_tl
8 \par 5th ~ item : ~ \tl_item:Nn \l_myick_tl { 5 }
9 \par reverse: ~ \tl_reverse:N \l_myick_tl \tl_use:N \l_myick_tl
  ~<~ \tl_to_str:N \l_myick_tl
10 \par reverse ~ items: ~ \exp_args:No \tl_reverse_items:n {
    \l_myick_tl } ~<~ \tl_to_str:N \l_myick_tl
11 \par 5th ~ item : ~ \tl_item:Nn \l_myick_tl { 5 }
12 }
13 \ExplSyntaxOff
14 \mycmdk{abc{de}fgh}

```

```

head: a
tail: bcdefgh
reverse: hgfdceba< hgf{de}cba
5th item: c
reverse: abcdefgh< abc{de}fgh
reverse items: hgfdceba < abc{de}fgh
5th item: f

```


[2.16] – Tokens

```

1 \ExplSyntaxOn
2 \tl_new:N \l_myiclz_tl
3 \cs_set:Npn \ic_func1:n #1 {
4 \tl_set:Nn \l_myiclz_tl { #1 }
5 \fbox{ \strut \tl_to_str:N \l_myiclz_tl } $^ \exp_args:No
   \tl_count_tokens:n { \l_myiclz_tl }$
6 \int_compare:nNnT
7 { \exp_args:No \tl_count_tokens:n { \l_myiclz_tl } }
8 >
9 { 1 }
10 { >> \mycmd1{#1} << }
11 }
12 \tl_new:N \l_myicl_tl
13 \NewDocumentCommand { \mycmd1 } { m } {
14 \tl_set:Nn \l_myicl_tl { #1 }
15 \tl_map_function:NN \l_myicl_tl \ic_func1:n
16 }
17 \ExplSyntaxOff
18 \mycmd1{abc{d\textit{e}}fgh}

```

a¹
b¹
c¹
d\textit {e}⁵
»
d¹
\textit¹
e¹
«
f¹
g¹
h¹

[2.17] – Variants

```

1 \ExplSyntaxOn
2 \cs_generate_variant:Nn \tl_count_tokens:n { V }
3 \tl_new:N \l_myicmz_tl
4 \cs_set:Npn \ic_funcm:n #1 {
5 \tl_set:Nn \l_myicmz_tl { #1 }
6 \fbox{ \strut \tl_to_str:N \l_myicmz_tl } $\^ \tl_count_tokens:V
   \l_myicmz_tl $
7 \int_compare:nNnT
8 { \tl_count_tokens:V \l_myicmz_tl }
9 >
10 { 1 }
11 { >> \mycmdm{#1} << }
12 }
13 \tl_new:N \l_myicm_tl
14 \NewDocumentCommand { \mycmdm } { m } {
15 \tl_set:Nn \l_myicm_tl { #1 }
16 \tl_map_function:NN \l_myicm_tl \ic_funcm:n
17 }
18 \ExplSyntaxOff
19 \mycmdm{abc{d\textit{e}}fgh}

```

a¹
b¹
c¹
d\textit {e}⁵
d¹
\textit¹
e¹
«¹
f¹
g¹
h¹


[2.18] – Box

```

1 \ExplSyntaxOn
2 \box_new:N \l_myicn_box
3 \hbox_set:Nn \l_myicn_box { \fbox { \Huge C \color{red}\tiny e } }
4 wd = \dim_use:N \box_wd:N \l_myicn_box \par
5 ht = \dim_use:N \box_ht:N \l_myicn_box \par
6 dp = \dim_use:N \box_dp:N \l_myicn_box \par
7 xxx \box_use:N \l_myicn_box xxx \par
8 \box_rotate:Nn \l_myicn_box { 40 }
9 wd = \dim_use:N \box_wd:N \l_myicn_box \par
10 ht = \dim_use:N \box_ht:N \l_myicn_box \par
11 dp = \dim_use:N \box_dp:N \l_myicn_box \par
12 xxx \box_use:N \l_myicn_box xxx
13 \ExplSyntaxOff

```

wd=26.47482pt
ht=20.79112pt
dp=3.79807pt

xxx  xxx
wd=36.08652pt
ht=32.94461pt
dp=2.90948pt

xxx  xxx

[2.19] – Box 2

```

1 \ExplSyntaxOn
2 \box_new:N \l_myico_box
3 \box_new:N \l_myicov_box
4 \cs_set:Npn \ic_funco: { 0.72 }
5 \cs_set:Npn \ic_funcoz:nn #1#2 {
6 \hbox_set:Nn \l_myico_box { { \Huge #1 \color{red}\tiny #2 } }
7 \hbox_set:Nn \l_myicov_box { \color{red}\Huge #1 }
8 \box_use:N \l_myico_box\llap {
9 \box_rotate:Nn \l_myico_box { 40 } \box_scale:Nnn \l_myico_box
   {\ic_funco:}{\ic_funco:}
10 \box_use:N \l_myico_box\llap {
11 \box_rotate:Nn \l_myico_box { 40 } \box_scale:Nnn \l_myico_box
   {\ic_funco:}{\ic_funco:}
12 \box_use:N \l_myico_box\llap {
13 \box_rotate:Nn \l_myico_box { 40 } \box_scale:Nnn \l_myico_box
   {\ic_funco:}{\ic_funco:}
14 \box_use:N \l_myico_box\llap {
15 \box_rotate:Nn \l_myico_box { 40 } \box_scale:Nnn \l_myico_box
   {\ic_funco:}{\ic_funco:}
16 \box_use:N \l_myico_box
17 }}}} \hspace{-\box_wd:N \l_myico_box} \box_use:N \l_myicov_box}
18 xxx \ic_funcoz:nn {A}{d} ~xxx \ic_funcoz:nn {C}{e} ~xxx
   \ic_funcoz:nn {X}{y} ~xxx \ic_funcoz:nn {O}{p} ~xxx
   \ic_funcoz:nn {B}{w} ~xxx \ic_funcoz:nn {I}{o}
19 \par xxx \ic_funcoz:nn {CAT}{dog} ~xxx \ic_funcoz:nn
   {Caterpillar}{f} ~xxx \ic_funcoz:nn {C~}{--} ~ \ic_funcoz:nn
   {O~}{--}
20 \ExplSyntaxOff

```



[2.20] – Box 3

```

1 \ExplSyntaxOn
2 \box_new:N \l_myicpz_box
3 \box_new:N \l_myicpy_box
4 \cs_set:Npn \ic_funcp:nn #1#2 {
5 \hbox_set:Nn \l_myicpz_box { { #1 } }
6 \hbox_set:Nn \l_myicpy_box { { \color{#2}#1 } }
7 \box_use:N \l_myicpz_box
8 { \hspace{-\box_wd:N \l_myicpy_box} \hspace{0.43pt} \box_use:N
   \l_myicpy_box }
9 }
10 xxx\Huge \ic_funcp:nn { C } { green }
11 \NewDocumentCommand { \mycmdp } { m m } { \ic_funcp:nn { #1 } {
   #2 } }
12 \ExplSyntaxOff
13 ~ and \mycmdp{cat}{red} and \mycmdp{dog}{brown}

```

xxx **C** and **cat** and **dog**

[2.21] – Box 4

```

1 \ExplSyntaxOn
2 \cs_generate_variant:Nn \box_new:N { c }
3 \cs_set:Npn \ic_funcpp:n #1 { \box_new:c { l_myicppz#1_box }
4 \hbox_set:cn { l_myicppz#1_box } { Contents ~ #1 }
5 x\box_use:c { l_myicppz#1_box }y ~ : ~
6 \box_show:c { l_myicppz#1_box }
7 }
8 \cs_set:Nn \ic_funcppb: {
9 \int_step_function:nnnN { 97 } { 1 } { 99 } \ic_funcpp:n
10 }
11 \NewDocumentCommand { \mycmdppb } { } { \ic_funcppb: }
12 %
13 \cs_set:Npn \ic_funcpq:n #1 { \box_use:c { l_myicppz#1_box } }
14 \NewDocumentCommand { \myboxnum } { m } { \box_use:c {
   l_myicppz#1_box } }
15 \ExplSyntaxOff
16 \mycmdppb
17 \par >>\myboxnum{98}<<

```

xContents 97y : xContents 98y : xContents 99y :
»Contents 98«

[2.22] – Sequence

```

1 \ExplSyntaxOn
2 \cs_generate_variant:Nn \tl_count_tokens:n { V }
3 \seq_new:N \l_myicq_seq
4 \seq_new:N \l_myicqz_seq
5 \seq_new:N \l_myicqy_seq
6 \tl_new:N \l_myicqz_tl
7 \seq_new:N \l_myicqyy_seq
8 \int_new:N \l_myicqz_int
9 \int_new:N \l_myicqy_int
10 \NewDocumentCommand { \mycmdq } { m } {
11 \seq_set_split:Nnn \l_myicq_seq { , } { #1 }
12 \seq_use:Nnnn \l_myicq_seq { - } { - } { - } ~ unsorted\par
13 \seq_sort:Nn \l_myicq_seq
14 {
15 \seq_clear:N \l_myicqz_seq
16 \seq_clear:N \l_myicqy_seq
17 \seq_set_split:Nnn \l_myicqz_seq { ; } { ##1 }
18 \seq_set_split:Nnn \l_myicqy_seq { ; } { ##2 }
19 %\seq_show:N \l_myicqz_seq
20 \tl_set:Nx \l_myicqz_tl { \seq_item:Nn \l_myicqz_seq {1} }
21 \tl_set:Nx \l_myicqy_tl { \seq_item:Nn \l_myicqy_seq {1} }
22 %\tl_show:N \l_myicqz_tl \tl_show:N \l_myicqy_tl
23 \int_set:Nn \l_myicqz_int { \tl_count_tokens:V \l_myicqz_tl }
24 \int_set:Nn \l_myicqy_int { \tl_count_tokens:V \l_myicqy_tl }
25 %\int_show:N \l_myicqz_int
26 %%>>\seq_item:Nn \l_myicqz_seq {1}<<
27 \int_compare:nNnTF { \l_myicqz_int } < { \l_myicqy_int }
28 { \sort_return_swapped: }
29 { \sort_return_same: }
30 }
31 \seq_use:Nnnn \l_myicq_seq { - } { - } { - } ~ sorted
32 }
33 \ExplSyntaxOff
34 \mycmdq{a;x,abv;y,ab;z,b;xyz,bb;i}

```

a;x-abv;y-ab;z-b;xyz-bb;i unsorted
abv;y-ab;z-bb;i-a;x-b;xyz sorted

[2.23] – Sequence 2

```

1 \ExplSyntaxOn
2 \cs_generate_variant:Nn \tl_replace_all:Nnn { Noo }
3 \seq_new:N \l_myicr_seq
4 \seq_new:N \l_myicrz_seq
5 \tl_new:N \l_myicrz_tl
6 \tl_new:N \l_myicrx_tl
7 \seq_set_split:Nnn \l_myicr_seq { , } { abv;y,ab;z,bb;i,
   a;x,b;xyz }
8 \cs_set:Npn \ic_funcrz:n #1 { %replace
9 \seq_set_split:Nnn \l_myicrz_seq { ; } { #1 }
10 \tl_set:Nx \l_myicrz_tl { \seq_item:Nn \l_myicrz_seq {1} }
11 \tl_set:Nx \l_myicrx_tl { \seq_item:Nn \l_myicrz_seq {2} }
12 \tl_replace_all:Noo \l_myicr_tl { \l_myicrz_tl } { \l_myicrx_tl
   . }
13 }
14 \tl_new:N \l_myicr_tl
15 \NewDocumentCommand { \mycmdr } { m } {
16 \tl_set:Nn \l_myicr_tl { #1 }
17 \tl_map_function:NN \l_myicr_seq \ic_funcrz:n
18 \tl_use:N \l_myicr_tl
19 }
20 \ExplSyntaxOff
21 \mycmdr{aabvabbbba{b}}

```

x.y.z.i.xyz.x.xyz.

[2.24] – Property List

```

1 \ExplSyntaxOn
2 \prop_new:N \l_myics_prop
3 \tl_new:N \l_myics_tl
4 \cs_set:Npn \ic_funcs:nn #1#2 {
5   key=#1,value={\color{blue}#2} ;~
6 }
7 \NewDocumentCommand { \mycmds } { m } {
8   \prop_get:NnN \l_myics_prop { #1 } \l_myics_tl
9   \tl_use:N \l_myics_tl
10  %\tl_show:N \l_myics_tl
11  ~::~~
12  \prop_item:Nn \l_myics_prop { #1 }
13  ~::~~
14  \prop_map_function:NN \l_myics_prop \ic_funcs:nn
15 }
16 \NewDocumentCommand { \mycmds } { m } {
17   \prop_set_from_keyval:Nn \l_myics_prop { #1 }
18   %\prop_show:N \l_myics_prop
19 }
20 \ExplSyntaxOff
21   \mycmds{alpha=~~~~03b1,beta=~~~~03b2,gamma=~~~~03b3,
22   delta=~~~~03b4}
23   {\testfont\mycmds{delta}}

```

δ: δ:: key=alpha,value=α; key=beta,value=β; key=gamma,value=γ;
key=delta,value=δ;

[2.25] – Miscellaneous 1

```

1 \ExplSyntaxOn
2 \cs_set:Npn \ic_func:n #1 {
3   {\Large\fbbox{ \tl_to_str:n {#1}}}.
4 }
5 (a) ~ \textbackslash \cs_to_str:N \l_char_active_seq ~ = ~
6 \seq_map_function:NN \l_char_active_seq \ic_func:n
7 \par (b) ~ \symbol{'a} \char_to_nfd:N q
8 \ExplSyntaxOff

```

(a) \l_char_active_seq= \ " \\$ \& \^ _ \~.

(b) aq

*=====

3 regexpatch

[3.1] – testca

```
1 \newcommand{\testca}{\textit{label}}
2 Before: \testca
3 \par \regexpatchcmd{\testca}{\c{textit}}{\c{textbf}}{S}{F}
4 \par \xpatchcmd{\testca}{label}{babble}{S}{F}
5 \par After: \testca
```

Before: *label*
S
S
After: **babble**

[3.2] – ph: Too many brace levels

```
1 \newcommand{\ph}[1]{
2 \textbf{\textsc{\color{blue}#1}}\ \ }
3 Before: {\testfont\ph{Snail in the Bottle}}
4 \regexpatchcmd{\ph}{\c{color}\cB\{blue\cE\}}{red}{}{F}
5 \par After: {\testfont\ph{Snail in the Bottle}}
```

Before: **SNAIL IN THE BOTTLE** F
After: **SNAIL IN THE BOTTLE**

[3.3] – ph2: Two levels of braces

```
1 \newcommand{\phb}[1]{\textsc{\color{blue}#1}}\ \ }
2 Before: {\testfont\phb{Snail in the Bottle}}
3 \regexpatchcmd{\phb}{\c{color}\cB\{blue\cE\}}{red}{}{F}
4 \par After: {\testfont\phb{Snail in the Bottle}}
```

Before: **SNAIL IN THE BOTTLE**
After: **REDSNAIL IN THE BOTTLE**

[3.4] – ph3: Entire `\color` command replaced

```
1 \newcommand{\phc}[1]{  
2 \bfseries\scshape\color{blue}#1\ \ }  
3 Before: {\testfont\phc{Snail in the Bottle}}  
4 \regexpatchcmd{\phc}{\c{color}\cB\{blue\cE\}}  
   {\c{color}\cB\{red\cE\}}{}{F}  
5 \par After: {\testfont\phc{Snail in the Bottle}}
```

Before: **SNAIL IN THE BOTTLE**

After: **SNAIL IN THE BOTTLE**

[3.5] – ph4: Text replaced: 'blue' > 'red'

```
1 \newcommand{\phd}[1]{  
2 \bfseries\scshape\color{blue}#1\ \ }  
3 Before: {\testfont\phd{Snail in the Bottle}}  
4 \xpatchcmd{\phd}{blue}{red}{}{F}  
5 \par After: {\testfont\phd{Snail in the Bottle}}
```

Before: **SNAIL IN THE BOTTLE**

After: **SNAIL IN THE BOTTLE**

[3.6] – ph5: Text ('blue') replaced by a macro ('\mycolour')

```
1 \newcommand{\mycolour}{green}  
2 \newcommand{\phe}[1]{  
3 \bfseries\scshape\color{blue}#1\ \ }  
4 Before: {\testfont\phe{Snail in the Bottle}}  
5 \regexpatchcmd{\phe}{blue}{\c{mycolour}}{}{F}  
6 \par After: {\testfont\phe{Snail in the Bottle}}
```

Before: **SNAIL IN THE BOTTLE**

After: **SNAIL IN THE BOTTLE**

[3.7] – ph6: Multi-level grouping without braces^a

^a“patchable” = it can be reconstructed from its decomposition under the current category code egime. – Manual, §7.1 (2018/05/02)

```

1 \newcommand{\mycolour}{brown}
2 \newcommand{\phf}[1]{\begingroup
3 \bfseries\begingroup\scshape\begingroup\color{blue}#1\endgroup\
  smallcaps\endgroup \ bold\endgroup\ normal \ \ }
4 Before: {\testfont\phf{Snail in the Bottle}}
5 \regexpatchcmd{\phf}{blue}{\c{mycolour}}{}{F}
6 \par After: {\testfont\phf{Snail in the Bottle}}
```

Before: **SNAIL IN THE BOTTLE SMALLCAPS bold** normal
 After: **SNAIL IN THE BOTTLE SMALLCAPS bold** normal

```

\dca{x} : nostar; noopt=-NoValue-; MArg=x.
\dca*{y} : star; noopt=-NoValue-; MArg=y.
\dca[abc]{z} : nostar; OArg=abc; MArg=z.
\dca[xyz]{zz} : nostar; OArg=xyz; MArg=zz.
\dca*[xyzz]{zzz} : star; OArg=xyzz; MArg=zzz.
```

[3.8] – ph4a: Text replaced: ‘blue’ > ‘red’

```

1 \newcommand{\phda}[2]{\{
2 \bfseries\scshape\color{blue}#1\normalcolor\ in the
  \color{blue}#2\}
3 Before: {\testfont\phda{Snail}{Bottle}}
4 \xpatchcmd{\phda}{blue}{red}{}{F}
5 \par After: {\testfont\phda{Snail}{Bottle}}
```

Before: **SNAIL IN THE BOTTLE**
 After: **SNAIL IN THE BOTTLE**

[3.9] – ph4b: Text replaced: all ‘blue’ > ‘red’

```

1 \newcommand{\phdb}[2]{\{
2 \bfseries\scshape\color{blue}#1\normalcolor\ in the
  \color{blue}#2\}
3 Before: {\testfont\phdb{Snail}{Bottle}}
4 \xpatchcmd*{\phdb}{blue}{red}{}{F}
5 \par After: {\testfont\phdb{Snail}{Bottle}}
```

Before: **SNAIL IN THE BOTTLE**
 After: **SNAIL IN THE BOTTLE**