

styledcmd

Paolo De Donato

03 August 2022

`styledcmd` is a \LaTeX package that allows you to create and manage different versions of your macro in order to be able to choose the better style for every occasion and avoid rewriting code each time.

1 How can you include it in your project?

You need only to have the file `styledcmd.sty` in your current working directory. Otherwise you can manually install it inside your preferred \LaTeX compiler (for example `TeXLive` or `MiKTeX`) in order to make it available for all your projects. Instructions for manually install a package can be found on Internet.

Then once you've added it you can include in your project with this command:

```
\usepackage{styledcmd}
```

2 How do you use it?

You can create a formatted macro via the following command

<code>\newstyledcmd</code>	<code>\newstyledcmd {<macro name>} {<format name>} [<number of arguments>] {<code>}</code>
<code>\renewstyledcmd</code>	
<code>\providestyledcmd</code>	

You can repeat that command for the same macro in order to create different styles, for example these commands

```
\newstyledcmd{\saluto}{informal}[1]{Hi #1}
\newstyledcmd{\saluto}{formal}[1]{Good morning #1}
```

define the two formats `informal` and `formal` for macro `\saluto`. You can directly use the command `\saluto` and use the default format (the first declared one) or using a specific style by passing it as an optional argument. So commands

```
\saluto{uncle}
\sauto[informal]{uncle}
\sauto[formal]{uncle}
```

will be expanded respectively as `Hi uncle`, `Hi uncle`, `Good morning uncle`. With the same syntax you can use `\renewstyledcmd` and `\providestyledcmd` with the same meaning of `\renewcommand` and `\providecommand` respectively.

3 How do you change the default style?

In order to change the default style (the one used when you don't choose explicitly a format) you need to execute the following command

```
\setGlobalStyle {<command name>} {<new default format name>}
```

For example in order to change the default style of command `\saluto` from `informal` to `formal` you need to execute command `\setGlobalStyle{\saluto}{formal}`. With this command the output of preceding commands will instead be Good morning uncle, Hi uncle, Good morning uncle.

4 Customize parameters with xparse

styledcmd loads automatically the xparse package for internal reasons. You can also define new styled commands with the same syntax used by `\NewDocumentCommand` with the following command

```
\NewDocStyledCMD {<command name>} {<format name>} {<arguments format>} {<code>}
\RenewDocStyledCMD
\ProvideDocStyledCMD
```

For example we can create the following two styles

```
\NewDocStyledCMD{\prova}{stylea}{r<>}{Stile 1 #1}
\NewDocStyledCMD{\prova}{styleb}{r<>}{Stile 2 #1}
```

in order to execute

```
\prova<Hello>
\prova[stylea]<Hello>
\prova[styleb]<Hello>
```

which are expanded respectively as Stile 1 Hello; Stile 1 Hello; Stile 2 Hello. Notice that the first optional argument passed to a command defined via `\NewDocStyleCMD` will always be interpreted as a style argument, so you should use another syntax for optional arguments or use a mandatory argument for the first place.

For example this declaration `\NewDocStyledCMD{\bad}{style}{o m}{Bad declaration}` should be avoided since for example `\bad[arg1]{arg2}` will interpret `arg1` as a style name and not as the first optional argument for `\bad`.

5 Implementation

```
1 <*package>
2 <@@=stycmd>
```

```
\c__stycmd_cmdproxy_str Proxy used to generate styled commands
3 \str_const:Nx \c__stycmd_cmdproxy_str { \object_address:nn
4   { stycmd }{ proxy } }
5
6 \proxy_create:nnN { stycmd }{ proxy } \c_object_public_str
7 \proxy_push_member:Vnn \c__stycmd_cmdproxy_str { default }{ t1 }
8
```

(End definition for \c__stycmd_cmdproxy_str.)

```

\__stycmd_cmd:n      Name of a command bounded to some style.
\__stycmd_cmd_style:nn 9 \cs_new:Nn \__stycmd_cmd:n
\__stycmd_cmd_default:n 10 {
11   \object_address:nn{ stycmd }{ entity - #1 }
12 }
13
14 \cs_new:Nn \__stycmd_cmd_style:nn
15 {
16   \object_member_adr:nnn{ \__stycmd_cmd:n{ #1 } }{ cmd - #2 }
17   { stycmd_void }
18 }
19
20 \cs_new:Nn \stycmd_void_use:N { #1 }
21 \cs_new_eq:NN \stycmd_void_use:c \use:c
22
23 \cs_new:Nn \__stycmd_cmd_default:n
24 {
25   \object_member_adr:nn{ \__stycmd_cmd:n{ #1 } } { default }
26 }
27

```

(End definition for __stycmd_cmd:n, __stycmd_cmd_style:nn, and __stycmd_cmd_default:n.)

\stycmd_xparsecmd:n Defines the main macro with \ProvideDocumentCommand.

```

28
29 \cs_new_protected:Nn \__stycmd_xparsecmd_aux:Nn
30 {
31   \ProvideDocumentCommand { #1 } { o }
32   {
33     \IfNoValueTF {##1}
34     {
35       \object_member_use:nn
36       {
37         \__stycmd_cmd:n{ #2 }
38       }
39       {
40         default
41       }
42     }
43     {
44       \object_member_use:nnn
45       {
46         \__stycmd_cmd:n{ #2 }
47       }
48       {
49         cmd - ##1
50       }
51       { stycmd_void }
52     }
53   }
54 }
55

```

```

56 \cs_generate_variant:Nn \__stycmd_xparsecmd_aux:Nn { cn }
57
58 \cs_new_protected:Nn \stycmd_xparsecmd:n
59 {
60   \__stycmd_xparsecmd_aux:cn { #1 }{ #1 }
61 }
62

```

(End definition for \stycmd_xparsecmd:n. This function is documented on page ??.)

\stycmd_pcmcmd:n Defines the main macro with \providecommand.

```

63
64 \cs_new_protected:Nn \__stycmd_pcmcmd_aux:Nn
65 {
66   \providecommand { #1 } [1] []
67   {
68     \tl_if_empty:nTF {##1}
69     {
70       \object_member_use:nn
71       {
72         \__stycmd_cmd:n{ #2 }
73       }
74       {
75         default
76       }
77     }
78     {
79       \object_member_use:nnn
80       {
81         \__stycmd_cmd:n{ #2 }
82       }
83       {
84         cmd - ##1
85       }
86       { stycmd_void }
87     }
88   }
89 }
90
91 \cs_generate_variant:Nn \__stycmd_pcmcmd_aux:Nn { cn }
92
93 \cs_new_protected:Nn \stycmd_pcmcmd:n
94 {
95   \__stycmd_pcmcmd_aux:cn { #1 }{ #1 }
96 }
97

```

(End definition for \stycmd_pcmcmd:n. This function is documented on page ??.)

\stycmd_expcmd:n Defines the main macro with \providecommand but the style argument is mandatory in order to make the command expandable. To use default style pass an empty argument as style.

```

98
99 \cs_new_protected:Nn \__stycmd_expcmd_aux:Nn

```

```

100 {
101   \providecommand { #1 } [1]
102   {
103     \tl_if_empty:nTF {##1}
104     {
105       \object_member_use:nn
106       {
107         \__stycmd_cmd:n{ #2 }
108       }
109       {
110         default
111       }
112     }
113     {
114       \object_member_use:nnn
115       {
116         \__stycmd_cmd:n{ #2 }
117       }
118       {
119         cmd - ##1
120       }
121       { stycmd_void }
122     }
123   }
124 }
125
126 \cs_generate_variant:Nn \__stycmd_expcmd_aux:Nn { cn }
127
128 \cs_new_protected:Nn \stycmd_expcmd:n
129 {
130   \__stycmd_expcmd_aux:cn { #1 }{ #1 }
131 }
132

```

(End definition for `\stycmd_expcmd:n`. This function is documented on page ??.)

\setGlobalStyle Change the default style for specified command

```

133
134 \NewDocumentCommand{\setGlobalStyle}{m m}
135 {
136   \__stycmd_chdef:Nn #1 { #2 }
137 }
138
139 \cs_new_protected:Nn \__stycmd_chdef_named:nn
140 {
141   \__stycmd_pars:cc
142   {
143     \object_member_adr:nn
144     {
145       \__stycmd_cmd:n{ #1 }
146     }
147     { default }
148   }
149   {

```

```

150     \object_member_adr:nnn
151     {
152         \__stycmd_cmd:n{ #1 }
153     }
154     {
155         cmd - #2
156     }
157     { stycmd_void }
158 }
159 }
160 \cs_generate_variant:Nn \__stycmd_chdef_named:nn { fn }
161 \cs_new_protected:Nn \__stycmd_chdef:Nn
162 {
163     \__stycmd_chdef_named:fn{ \cs_to_str:N #1 }{ #2 }
164 }
165

```

(End definition for `\setGlobalStyle`. This function is documented on page 2.)

`\stycmd_generate:NNN` Declare the styled version #1 of the macro generator command #2. the `_renew` variant requires a preceding declaration

`\stycmd_generate:NN`

`\stycmd_generate_renew:NN`

```

166 \cs_new:Nn \__stycmd_pars:NN
167 {
168     \tl_gset:Nn #1 { #2 }
169 }
170
171
172 \cs_generate_variant:Nn \__stycmd_pars:NN { cc }
173
174 \cs_new_protected:Nn \__stycmd_generate_aux:NNnn
175 {
176     \object_if_exist:nF
177     {
178         \__stycmd_cmd:n{ #3 }
179     }
180     {
181         \object_create:VnnNN \c__stycmd_cmdproxy_str
182         { stycmd }{ entity - #3 }
183         \c_object_global_str
184         \c_object_public_str
185
186         \__stycmd_pars:cc
187         {
188             \object_member_adr:nn
189             {
190                 \__stycmd_cmd:n{ #3 }
191             }
192             { default }
193         }
194         {
195             \object_member_adr:nnn
196             {
197                 \__stycmd_cmd:n{ #3 }
198             }
199         }
200     }
201 }

```

```

199         {
200             cmd - #4
201         }
202         { stycmd_void }
203     }
204
205     #2 { #3 }
206 }
207 \exp_args:Nc #1
208 {
209     \object_member_adr:nnn
210     {
211         \__stycmd_cmd:n{ #3 }
212     }
213     {
214         cmd - #4
215     }
216     { stycmd_void }
217 }
218
219 }
220
221 \cs_generate_variant:Nn \__stycmd_generate_aux:NNnn { NNfn }
222
223 \cs_new_protected:Nn \__stycmd_generate_aux_cmd:NNNn
224 {
225     \__stycmd_generate_aux:NNfn #1 #2 { \cs_to_str:N #3 }{ #4 }
226 }
227
228 \cs_new_protected:Nn \__stycmd_generate_renew_aux:Nnn
229 {
230     \exp_args:Nc #1
231     {
232         \object_member_adr:nnn
233         {
234             \__stycmd_cmd:n{ #2 }
235         }
236         {
237             cmd - #3
238         }
239         { stycmd_void }
240     }
241 }
242
243
244 \cs_new_protected:Nn \stycmd_generate:NNN
245 {
246     \cs_new_protected:Npn #1 ##1 ##2
247     {
248         \__stycmd_generate_aux_cmd:NNNn #2 #3 ##1 { ##2 }
249     }
250 }
251 \cs_new_protected:Nn \stycmd_generate:NN
252 {

```

```

253 \stycmd_generate:NNN #1 #2 \stycmd_xparsecmd:n
254 }
255
256
257 \cs_new_protected:Nn \stycmd_generate_renew:NN
258 {
259   \cs_new_protected:Npn #1 ##1 ##2
260   {
261     \__stycmd_generate_renew_aux:Nnn #2 { ##1 }{ ##2 }
262   }
263 }
264

```

(End definition for \stycmd_generate:NNN, \stycmd_generate:NN, and \stycmd_generate_renew:NN. These functions are documented on page ??.)

```

\newstyledcmd Declare a new macro with the specified style name.
\renewstyledcmd 265 \stycmd_generate:NN \newstyledcmd \newcommand
\providestyledcmd 266 \stycmd_generate_renew:NN \renewstyledcmd \renewcommand
267 \stycmd_generate:NN \providestyledcmd \providecommand

```

(End definition for \newstyledcmd, \renewstyledcmd, and \providestyledcmd. These functions are documented on page 1.)

```

\NewDocStyledCMD Declare a new styled macro with the \NewDocumentCommand syntax.
\RenewDocStyledCMD 268 \stycmd_generate:NN \NewDocStyledCMD \NewDocumentCommand
\ProvideDocStyledCMD 269 \stycmd_generate_renew:NN \RenewDocStyledCMD \RenewDocumentCommand
270 \stycmd_generate:NN \ProvideDocStyledCMD \ProvideDocumentCommand

```

(End definition for \NewDocStyledCMD, \RenewDocStyledCMD, and \ProvideDocStyledCMD. These functions are documented on page 2.)

```

271 \</package>

```