

# tagpdf – A package to experiment with pdf tagging\*

Ulrike Fischer<sup>†</sup>

Released 2021-06-14

## Contents

<b>I</b>	<b>The tagpdf-checks module</b>	
	Messages and check code	
	part of the tagpdf package	<b>2</b>
<b>1</b>	<b>Commands</b>	<b>2</b>
<b>2</b>	<b>log-levels</b>	<b>2</b>
<b>3</b>	<b>Messages</b>	<b>2</b>
3.1	Messages related to mc-chunks . . . . .	2
3.2	Messages related to mc-chunks . . . . .	3
3.3	Attributes . . . . .	4
3.4	Roles . . . . .	4
3.5	Miscellaneous . . . . .	4
<b>4</b>	<b>Retrieving data</b>	<b>5</b>
<b>5</b>	<b>User conditionals</b>	<b>5</b>
<b>6</b>	<b>Internal checks</b>	<b>5</b>
6.1	checks for active tagging . . . . .	5
6.2	Checks related to stuctures . . . . .	6
6.3	Checks related to roles . . . . .	7
6.4	Check related to mc-chunks . . . . .	7
6.5	Miscellaneous . . . . .	10
	<b>Index</b>	<b>11</b>

---

\*This file describes v0.82, last revised 2021-06-14.

<sup>†</sup>E-mail: [fischer@troubleshooting-tex.de](mailto:fischer@troubleshooting-tex.de)

## Part I

# The tagpdf-checks module

## Messages and check code

## part of the tagpdf package

### 1 Commands

---

`\tag_if_active_p:` *★* This command tests if tagging is active. It only gives true if all tagging has been activated, `\tag_if_active:` *TF* *★* and if tagging hasn't been stopped locally.

---

---

`\tag_get:n` *★* `\tag_get:n{<keyword>}`

---

This is a generic command to retrieve data. Currently the only sensible values for the argument *<keyword>* are `mc_tag` and `struct_tag`.

### 2 log-levels

command/message	log-level	type	note
<code>\showtagpdfmcdata</code>	0	log/term	lua-only
1 <code>&lt;@@=tag&gt;</code>			
2 <code>&lt;*checks&gt;</code>			
3 <code>\ProvidesExplPackage {tagpdf-checks-code} {2021-06-14} {0.82}</code>			
4 <code>{part of tagpdf - code related to checks, conditionals, debugging and messages}</code>			

### 3 Messages

#### 3.1 Messages related to mc-chunks

`mc-nested` This message is issued when a mc is opened before the previous has been closed. This is not relevant for luamode, as the attributes don't care about this. It is used in the `\@@_check_mc_if_nested:` test.

5 `\msg_new:nnn { tag } {mc-nested} { nested-marked-content-found~~mcid~#1 }`

*(End definition for mc-nested. This function is documented on page ??.)*

`mc-tag-missing` If the tag is missing

6 `\msg_new:nnn { tag } {mc-tag-missing} { required-tag-missing~~mcid~#1 }`

*(End definition for mc-tag-missing. This function is documented on page ??.)*

`mc-label-unknown` If the label of a mc that is used in another place is not known (yet)

7 `\msg_new:nnn { tag } {mc-label-unknown} { label~#1~unknown~~rerun }`

*(End definition for mc-label-unknown. This function is documented on page ??.)*

**mc-used-twice** An mc-chunk can be inserted only in one structure. This indicates wrong coding and so should at least give a warning.

```
8 \msg_new:nnn { tag } {mc-used-twice} { mc~#1~has~been~already~used }
```

*(End definition for mc-used-twice. This function is documented on page ??.)*

**mc-not-open** This is issued if a `\tag_mc_end:` is issued wrongly, wrong coding.

```
9 \msg_new:nnn { tag } {mc-not-open} { there~is~no~mc~to~end~at~#1 }
```

*(End definition for mc-not-open. This function is documented on page ??.)*

**mc-pushed** Informational messages about mc-pushing.

```
10 \msg_new:nnn { tag } {mc-pushed} { #1~has~been~pushed~to~the~mc~stack}
11 \msg_new:nnn { tag } {mc-popped} { #1~has~been~removed~from~the~mc~stack }
```

*(End definition for mc-pushed and mc-popped. These functions are documented on page ??.)*

**mc-current** Informational messages about current mc state.

```
12 \msg_new:nnn { tag } {mc-current}
13 { current~MC:~
14   \bool_if:NTF\g__tag_in_mc_bool
15     {abscnt=\__tag_get_mc_abs_cnt:~tag=\g__tag_mc_key_tag_tl}
16     {no~MC~open,~current~abscnt=\__tag_get_mc_abs_cnt:"}
17 }
```

*(End definition for mc-current. This function is documented on page ??.)*

## 3.2 Messages related to mc-chunks

**struct-no-objnum** Should not happen ...

```
18 \msg_new:nnn { tag } {struct-no-objnum} { objnum~missing~for~structure~#1 }
```

*(End definition for struct-no-objnum. This function is documented on page ??.)*

**struct-faulty-nesting** This indicates that there is somewhere one `\tag_struct_end:` too much. This should be normally an error.

```
19 \msg_new:nnn { tag }
20 {struct-faulty-nesting}
21 { there~is~no~open~structure~on~the~stack }
```

*(End definition for struct-faulty-nesting. This function is documented on page ??.)*

**struct-missing-tag** A structure must have a tag.

```
22 \msg_new:nnn { tag } {struct-missing-tag} { a~structure~must~have~a~tag! }
```

*(End definition for struct-missing-tag. This function is documented on page ??.)*

**struct-used-twice**

```
23 \msg_new:nnn { tag } {struct-used-twice}
24 { structure~with~label~#1~has~already~been~used }
```

*(End definition for struct-used-twice. This function is documented on page ??.)*

**struct-label-unknown** label is unknown, typically needs a rerun.

```
25 \msg_new:nnn { tag } {struct-label-unknown}
26 { structure~with~label~#1~is~unknown~rerun }
```

(End definition for `struct-label-unknown`. This function is documented on page ??.)

`struct-show-closing` Informational message shown if log-mode is high enough

```
27 \msg_new:nnn { tag } {struct-show-closing}
28   { closing-structure~#1~tagged~\prop_item:cn{g__tag_struct_#1_prop}{S} }
```

(End definition for `struct-show-closing`. This function is documented on page ??.)

### 3.3 Attributes

Not much yet, as attributes aren't used so much.

`attr-unknown`

```
29 \msg_new:nnn { tag } {attr-unknown} { attribute~#1~is~unknown }
```

(End definition for `attr-unknown`. This function is documented on page ??.)

### 3.4 Roles

`role-missing` Warning message if either the tag or the role is missing

```
role-unknown 30 \msg_new:nnn { tag } {role-missing} { tag~#1~has~no~role~assigned }
role-unknown-tag 31 \msg_new:nnn { tag } {role-unknown} { role~#1~is~not~known }
32 \msg_new:nnn { tag } {role-unknown-tag} { tag~#1~is~not~known }
```

(End definition for `role-missing`, `role-unknown`, and `role-unknown-tag`. These functions are documented on page ??.)

`role-tag` Info messages.

```
new-tag 33 \msg_new:nnn { tag } {role-tag} { mapping-tag~#1~to~role~#2 }
34 \msg_new:nnn { tag } {new-tag} { adding-new-tag~#1 }
```

(End definition for `role-tag` and `new-tag`. These functions are documented on page ??.)

### 3.5 Miscellaneous

`tree-mcid-index-wrong` Used in the tree code, typically indicates the document must be rerun.

```
35 \msg_new:nnn { tag } {tree-mcid-index-wrong}
36   {something~is~wrong~with~the~mcid--rerun}
```

(End definition for `tree-mcid-index-wrong`. This function is documented on page ??.)

`obj-write-num` An info message, useful for reporting.

```
37 \msg_new:nnn { tag } {obj-write-num} {write-obj~#1~to~pdf}
```

(End definition for `obj-write-num`. This function is documented on page ??.)

`sys-no-interwordspace` Currently only pdf<sub>l</sub>atex and lua<sub>l</sub>atex have some support for real spaces.

```
38 \msg_new:nnn { tag } {sys-no-interwordspace}
39   {engine/output~mode~#1~doesn't~support~the~interword~spaces}
```

(End definition for `sys-no-interwordspace`. This function is documented on page ??.)

## 4 Retrieving data

**\tag\_get:n** This retrieves some data. This is a generic command to retrieve data. Currently the only sensible values for the argument are **mc\_tag** and **struct\_tag**.

```
40 \cs_new:Npn \tag_get:n #1 { \use:c {__tag_get_data_#1: } }
```

(End definition for \tag\_get:n. This function is documented on page 2.)

## 5 User conditionals

**\tag\_if\_active\_p:** This is a test if tagging is active. This allows packages to add conditional code. The test is true if all booleans, the global and the two local one are true.  
**\tag\_if\_active:TF**

```
41 \prg_new_conditional:Npnn \tag_if_active: { p , T , TF , F }
42 {
43   \bool_lazy_all:nTF
44   {
45     {\g__tag_active_struct_bool}
46     {\g__tag_active_mc_bool}
47     {\g__tag_active_tree_bool}
48     {\l__tag_active_struct_bool}
49     {\l__tag_active_mc_bool}
50   }
51   {
52     \prg_return_true:
53   }
54   {
55     \prg_return_false:
56   }
57 }
```

(End definition for \tag\_if\_active:TF. This function is documented on page 2.)

## 6 Internal checks

These are checks used in various places in the code.

### 6.1 checks for active tagging

**\\_\_tag\_check\_if\_active\_mc:TF** Structures must have a tag, so we check if the S entry is in the property. It is an error if this is missing. The argument is a number.  
**\\_\_tag\_check\_if\_active\_struct:TF**

```
58 \prg_new_conditional:Npnn \__tag_check_if_active_mc: {T,F,TF}
59 {
60   \bool_lazy_and:nnTF { \g__tag_active_mc_bool } { \l__tag_active_mc_bool }
61   {
62     \prg_return_true:
63   }
64   {
65     \prg_return_false:
66   }
67 }
68 \prg_new_conditional:Npnn \__tag_check_if_active_struct: {T,F,TF}
```

```

69 {
70   \bool_lazy_and:nnTF { \g__tag_active_struct_bool } { \l__tag_active_struct_bool }
71   {
72     \prg_return_true:
73   }
74   {
75     \prg_return_false:
76   }
77 }

```

(End definition for \\_\_tag\_check\_if\_active\_mc:TF and \\_\_tag\_check\_if\_active\_struct:TF.)

## 6.2 Checks related to structures

\\_\_tag\_check\_structure\_has\_tag:n Structures must have a tag, so we check if the S entry is in the property. It is an error if this is missing. The argument is a number.

```

78 \cs_new_protected:Npn \__tag_check_structure_has_tag:n #1 %#1 struct num
79 {
80   \prop_if_in:cnF { g__tag_struct_#1_prop }
81   {S}
82   {
83     \msg_error:nn { tag } {struct-missing-tag}
84   }
85 }

```

(End definition for \\_\_tag\_check\_structure\_has\_tag:n.)

\\_\_tag\_check\_structure\_tag:N This checks if the name of the tag is known.

```

86 \cs_new_protected:Npn \__tag_check_structure_tag:N #1
87 {
88   \prop_if_in:NoF \g__tag_role_tags_prop {#1}
89   {
90     \msg_warning:nnx { tag } {role-unknown-tag} {#1}
91   }
92 }

```

(End definition for \\_\_tag\_check\_structure\_tag:N.)

\\_\_tag\_check\_info\_closing\_struct:n This info message is issued at a closing structure, the use should be guarded by log-level.

```

93 \cs_new_protected:Npn \__tag_check_info_closing_struct:n #1 %#1 struct num
94 {
95   \msg_info:nnn { tag } {struct-show-closing} {#1}
96 }
97
98 \cs_generate_variant:Nn \__tag_check_info_closing_struct:n {o,x}

```

(End definition for \\_\_tag\_check\_info\_closing\_struct:n.)

\\_\_tag\_check\_no\_open\_struct: This checks if there is an open structure. It should be used when trying to close a structure. It errors if false.

```

99 \cs_new_protected:Npn \__tag_check_no_open_struct:
100 {
101   \msg_error:nn { tag } {struct-faulty-nesting}
102 }

```

(End definition for `\_tag\_check\_no\_open\_struct:`.)

`\_tag\_check\_struct\_used:n` This checks if a stashed structure has already been used.

```

103 \cs_new_protected:Npn \_tag\_check\_struct\_used:n #1 %#1 label
104 {
105   \prop\_get:cnNT
106     {g\_tag\_struct\_\_tag\_ref\_value:enn{tagpdfstruct-#1}{tagstruct}{unknown}\_prop}
107     {P}
108     \l\_tmpa\_tl
109     {
110       \msg\_warning:nnn { tag } {struct-used-twice} {#1}
111     }
112 }
```

(End definition for `\_tag\_check\_struct\_used:n`.)

### 6.3 Checks related to roles

`\_tag\_check\_add\_tag\_role:nn` This check is used when defining a new role mapping.

```

113 \cs_new_protected:Npn \_tag\_check\_add\_tag\_role:nn #1 #2 %#1 tag, #2 role
114 {
115   \tl\_if\_empty:nTF {#2}
116   {
117     \msg\_warning:nnn { tag } {role-missing} {#1}
118   }
119   {
120     \prop\_get:NnNTF \g\_tag\_role\_tags\_prop {#2} \l\_tmpa\_tl
121     {
122       \msg\_info:nnnn { tag } {role-tag} {#1} {#2}
123     }
124     {
125       \msg\_warning:nnn { tag } {role-unknown} {#2}
126     }
127   }
128 }
```

(End definition for `\_tag\_check\_add\_tag\_role:nn`.)

### 6.4 Check related to mc-chunks

`\_tag\_check\_mc\_if\_nested:` Two tests if a mc is currently open.

```

\_tag\_check\_mc\_if\_open:
129 \cs_new_protected:Npn \_tag\_check\_mc\_if\_nested:
130 {
131   \_tag\_mc\_if\_in:T
132   {
133     \msg\_warning:nnx { tag } {mc-nested} { \_tag\_get\_mc\_abs\_cnt: }
134   }
135 }
136
137 \cs_new_protected:Npn \_tag\_check\_mc\_if\_open:
138 {
139   \_tag\_mc\_if\_in:F
140   {
141     \msg\_warning:nnx { tag } {mc-not-open} { \_tag\_get\_mc\_abs\_cnt: }
```

```

142     }
143 }

```

(End definition for `\_tag\_check\_mc\_if\_nested:` and `\_tag\_check\_mc\_if\_open:.`)

`\_tag\_check\_mc\_pushed\_popped:nn`

This creates an information message if mc's are pushed or popped. The first argument is a word (pushed or popped), the second the tag name. With larger log-level the stack is shown too.

```

144 \cs_new_protected:Npn \_tag\_check\_mc\_pushed\_popped:nn #1 #2
145 {
146   \int_compare:nNnT
147     { \l\_tag\_loglevel\_int } = { 2 }
148     { \msg_info:nnx {tag}{mc-#1}{#2} }
149   \int_compare:nNnT
150     { \l\_tag\_loglevel\_int } > { 2 }
151     {
152       \msg_warning:nnx {tag}{mc-#1}{#2}
153       \seq_log:N \g\_tag\_mc\_stack\_seq
154     }
155 }

```

(End definition for `\_tag\_check\_mc\_pushed\_popped:nn.`)

`\_tag\_check\_mc\_tag:N`

This checks if the mc has a (known) tag.

```

156 \cs_new_protected:Npn \_tag\_check\_mc\_tag:N #1  % #1 is var with a tag name in it
157 {
158   \tl_if_empty:NT #1
159   {
160     \msg_error:nnx { tag } {mc-tag-missing} { \_tag\_get\_mc\_abs\_cnt: }
161   }
162   \prop_if_in:Nof \g\_tag\_role\_tags\_NS\_prop {#1}
163   {
164     \msg_warning:nnx { tag } {role-unknown-tag} {#1}
165   }
166 }

```

(End definition for `\_tag\_check\_mc\_tag:N.`)

`\g\_tag\_check\_mc\_used\_seq`

This variable holds the list of used mc. It will hopefully be rather short, so checking the seq is not too slow.

```

167 \seq_new:N \g\_tag\_check\_mc\_used\_seq

```

(End definition for `\g\_tag\_check\_mc\_used\_seq.`)

`\_tag\_check\_mc\_used:n`

This checks if a mc is used twice.

```

168 \cs_new_protected:Npn \_tag\_check\_mc\_used:n #1
169 {
170   \seq_if_in:NnTF \g\_tag\_check\_mc\_used\_seq {#1}
171   {
172     \msg_warning:nnn { tag } {mc-used-twice} {#1}
173   }
174   {
175     \seq_gput_right:Nx \g\_tag\_check\_mc\_used\_seq {#1}
176   }
177 }

```



(End definition for \\_tag\\_check\\_mc\\_used:n.)

\\_tag\\_check\\_show\\_MCID\\_by\\_page: This allows to show the mc on a page. Currently unused.

```

178 \cs_new_protected:Npn \_tag\_check\_show\_MCID\_by\_page:
179 {
180   \tl_set:Nx \l__tag\_tmpa\_tl
181   {
182     \_tag\_ref\_value\_lastpage:nn
183     {abspage}
184     {-1}
185   }
186   \int_step_inline:nnnn {1}{1}
187   {
188     \l__tag\_tmpa\_tl
189   }
190   {
191     \seq_clear:N \l_tmpa\_seq
192     \int_step_inline:nnnn
193     {1}
194     {1}
195     {
196       \_tag\_ref\_value\_lastpage:nn
197       {tagmcabs}
198       {-1}
199     }
200     {
201       \int_compare:nT
202       {
203         \_tag\_ref\_value:enn
204         {mcid-###1}
205         {tagabspage}
206         {-1}
207         =
208         ##1
209       }
210       {
211         \seq_gput_right:Nx \l_tmpa\_seq
212         {
213           Page##1-###1-
214           \_tag\_ref\_value:enn
215           {mcid-###1}
216           {tagmcid}
217           {-1}
218         }
219       }
220     }
221     \seq_show:N \l_tmpa\_seq
222   }
223 }

```

(End definition for \\_tag\\_check\\_show\\_MCID\\_by\\_page:.)

## 6.5 Miscellaneous

`\_tag\_check\_record\_pdfobj\_num:n` This writes the object numbers. Not sure if needed, currently unused.

```
224 \cs_new_protected:Npn \_tag\_check\_record\_pdfobj\_num:n #1
225 {
226   \int_compare:nT { \l__tag_loglevel_int >= 3 }
227   {
228     \msg_info:nnx { tag } {obj-write-num} {#1}
229   }
230 }
231 </checks>
```

*(End definition for \\_tag\\_check\\_record\\_pdfobj\\_num:n.)*

# 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.

<b>A</b>	
attr-unknown	<u>29</u>
<b>B</b>	
bool commands:	
\bool_if:NnTF	<u>14</u>
\bool_lazy_all:nTF	<u>43</u>
\bool_lazy_and:nnTF	<u>60</u> , <u>70</u>
<b>C</b>	
cs commands:	
\cs_generate_variant:Nn	<u>98</u>
\cs_new:Npn	<u>40</u>
\cs_new_protected:Npn	<u>78</u> , <u>86</u> , <u>93</u> , <u>99</u> , <u>103</u> , <u>113</u> , <u>129</u> , <u>137</u> , <u>144</u> , <u>156</u> , <u>168</u> , <u>178</u> , <u>224</u>
<b>I</b>	
int commands:	
\int_compare:nNnTF	<u>146</u> , <u>149</u>
\int_compare:nTF	<u>201</u> , <u>226</u>
\int_step_inline:nnnn	<u>186</u> , <u>192</u>
<b>M</b>	
mc-current	<u>12</u>
mc-label-unknown	<u>7</u>
mc-nested	<u>5</u>
mc-not-open	<u>9</u>
mc-popped	<u>10</u>
mc-pushed	<u>10</u>
mc-tag-missing	<u>6</u>
mc-used-twice	<u>8</u>
msg commands:	
\msg_error:nn	<u>83</u> , <u>101</u>
\msg_error:nnn	<u>160</u>
\msg_info:nnn	<u>95</u> , <u>148</u> , <u>228</u>
\msg_info:nnnn	<u>122</u>
\msg_new:nnn	<u>5</u> , <u>6</u> , <u>7</u> , <u>8</u> , <u>9</u> , <u>10</u> , <u>11</u> , <u>12</u> , <u>18</u> , <u>19</u> , <u>22</u> , <u>23</u> , <u>25</u> , <u>27</u> , <u>29</u> , <u>30</u> , <u>31</u> , <u>32</u> , <u>33</u> , <u>34</u> , <u>35</u> , <u>37</u> , <u>38</u>
\msg_warning:nnn	<u>90</u> , <u>110</u> , <u>117</u> , <u>125</u> , <u>133</u> , <u>141</u> , <u>152</u> , <u>164</u> , <u>172</u>
<b>N</b>	
new-tag	<u>33</u>
<b>O</b>	
obj-write-num	<u>37</u>
<b>P</b>	
prg commands:	
\prg_new_conditional:Npnn	<u>41</u> , <u>58</u> , <u>68</u>
\prg_return_false:	<u>55</u> , <u>65</u> , <u>75</u>
\prg_return_true:	<u>52</u> , <u>62</u> , <u>72</u>
prop commands:	
\prop_get:NnNTF	<u>105</u> , <u>120</u>
\prop_if_in:NnTF	<u>80</u> , <u>88</u> , <u>162</u>
\prop_item:Nn	<u>28</u>
\ProvidesExplPackage	<u>3</u>
<b>R</b>	
role-missing	<u>30</u>
role-tag	<u>33</u>
role-unknown	<u>30</u>
role-unknown-tag	<u>30</u>
<b>S</b>	
seq commands:	
\seq_clear:N	<u>191</u>
\seq_gput_right:Nn	<u>175</u> , <u>211</u>
\seq_if_in:NnTF	<u>170</u>
\seq_log:N	<u>153</u>
\seq_new:N	<u>167</u>
\seq_show:N	<u>221</u>
\l_tmpa_seq	<u>191</u> , <u>211</u> , <u>221</u>
\showtagpdfmcddata	<u>2</u>
struct-faulty-nesting	<u>19</u>
struct-label-unknown	<u>25</u>
struct-missing-tag	<u>22</u>
struct-no-objnum	<u>18</u>
struct-show-closing	<u>27</u>
struct-used-twice	<u>23</u>
sys-no-interwordspace	<u>38</u>
<b>T</b>	
tag commands:	
\tag_get:n	<u>2</u> , <u>40</u> , <u>40</u>
\tag_if_active:	<u>41</u>
\tag_if_active:TF	<u>2</u> , <u>41</u>
\tag_if_active_p:	<u>2</u> , <u>41</u>
tag internal commands:	
\g__tag_active_mc_bool	<u>46</u> , <u>60</u>
\l__tag_active_mc_bool	<u>49</u> , <u>60</u>
\g__tag_active_struct_bool	<u>45</u> , <u>70</u>
\l__tag_active_struct_bool	<u>48</u> , <u>70</u>
\g__tag_active_tree_bool	<u>47</u>
\__tag_check_add_tag_role:nn	<u>113</u> , <u>113</u>
\__tag_check_if_active_mc:	<u>58</u>

