

The tagpdf-checks module

Messages and check code

part of the tagpdf package

Ulrike Fischer*

Version 0.82, released 2021-06-14

```
1 <@@=tag>
2 <{*checks}>
3 \ProvidesExplPackage {tagpdf-checks-code} {2021-06-14} {0.82}
4 {part of tagpdf - code related to checks and messages}
```

1 Messages and checks

1.1 Messages

1.1.1 Messages related to mc-chunks

mc-nested This message is issued if 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 }
```

*E-mail: fischer@troubleshooting-tex.de

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

mc-pushed Informational messages about mc-pushing.

mc-popped

```
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 ??.)

1.1.2 Messages related to mc-chunks

struct-no-objnum Should not happen ...

```
12 \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.

```
13 \msg_new:nnn { tag }
14   {struct-faulty-nesting}
15   { 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.

```
16 \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

```
17 \msg_new:nnn { tag } {struct-used-twice}
18   { 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.

```
19 \msg_new:nnn { tag } {struct-label-unknown}
20   { 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

```
21 \msg_new:nnn { tag } {struct-show-closing}
22   { 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 ??.)

1.1.3 Attributes

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

attr-unknown

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

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

1.2 Roles

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

`role-unknown` `\msg_new:nnn { tag } {role-missing} { tag~#1~has-no~role~assigned }`

`role-unknown-tag` `\msg_new:nnn { tag } {role-unknown} { role~#1~is-not-known }`

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

`\msg_new:nnn { tag } {new-tag} { adding-new-tag~#1 }`

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

1.2.1 Miscellaneous

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

`\msg_new:nnn { tag } {tree-mcid-index-wrong}`

`{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.

`\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_lat_ex and lua_lat_ex have some support for real spaces.

`\msg_new:nnn { tag } {sys-no-interwordspace}`

`{engine/backend~#1~doesn't~support~the~interword~spaces}`

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

1.3 Checks

These are checks used in various places in the code.

1.3.1 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.

`\cs_new_protected:Npn _tag_check_structure_has_tag:n #1 %#1 struct num`

`{`

`\prop_if_in:cnF { g_tag_struct_#1_prop }`

`{S}`

`{`

`\msg_error:nn { tag } {struct-missing-tag}`

`}`

`}`

(End definition for _tag_check_structure_has_tag:n.)

`__tag_check_structure_tag:N` This checks if the name of the tag is known.

```

42 \cs_new_protected:Npn \__tag_check_structure_tag:N #1
43 {
44   \prop_if_in:Nof \g__tag_role_tags_prop {#1}
45   {
46     \msg_warning:nnx { tag } {role-unknown-tag} {#1}
47   }
48 }

```

(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.

```

49 \cs_new_protected:Npn \__tag_check_info_closing_struct:n #1 %#1 struct num
50 {
51   \msg_info:nnn { tag } {struct-show-closing} {#1}
52 }
53
54 \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.

```

55 \cs_new_protected:Npn \__tag_check_no_open_struct:
56 {
57   \msg_error:nn { tag } {struct-faulty-nesting}
58 }

```

(End definition for __tag_check_no_open_struct:.)

`__tag_check_struct_used:n` This checks if a stashed structure has already been used.

```

59 \cs_new_protected:Npn \__tag_check_struct_used:n #1 %#1 label
60 {
61   \prop_get:cnNT
62     {g__tag_struct_\__tag_ref_value:enn{tagpdfstruct-#1}{tagstruct}{unknown}_prop}
63     {P}
64     \l_tmpa_tl
65     {
66       \msg_warning:nnn { tag } {struct-used-twice} {#1}
67     }
68 }

```

(End definition for __tag_check_struct_used:n.)

1.3.2 Checks related to roles

`__tag_check_add_tag_role:nn` This check is used when defining a new role mapping.

```

69 \cs_new_protected:Npn \__tag_check_add_tag_role:nn #1 #2 %#1 tag, #2 role
70 {
71   \tl_if_empty:nTF {#2}
72   {
73     \msg_warning:nnn { tag } {role-missing} {#1}
74   }
75   {

```

```

76     \prop_get:NnNTF \g__tag_role_tags_prop {#2} \l_tmpa_tl
77     {
78         \msg_info:nnnn { tag } {role-tag} {#1} {#2}
79     }
80     {
81         \msg_warning:nnn { tag } {role-unknown} {#2}
82     }
83 }
84 }

```

(End definition for __tag_check_add_tag_role:nn.)

1.3.3 Check related to mc-chunks

__tag_check_mc_if_nested: Two tests if a mc is currently open.

```

\__tag_check_mc_if_open:
85 \cs_new_protected:Npn \__tag_check_mc_if_nested:
86 {
87     \__tag_mc_if_in:T
88     {
89         \msg_warning:nnx { tag } {mc-nested} { \__tag_get_mc_abs_cnt: }
90     }
91 }
92
93 \cs_new_protected:Npn \__tag_check_mc_if_open:
94 {
95     \__tag_mc_if_in:F
96     {
97         \msg_warning:nnx { tag } {mc-not-open} { \__tag_get_mc_abs_cnt: }
98     }
99 }

```

(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.

```

100 \cs_new_protected:Npn \__tag_check_mc_pushed_popped:nn #1 #2
101 {
102     \int_compare:nNnT
103     { \l__tag_loglevel_int } = { 2 }
104     { \msg_info:nnx {tag}{mc-#1}{#2} }
105     \int_compare:nNnT
106     { \l__tag_loglevel_int } > { 2 }
107     {
108         \msg_warning:nnx {tag}{mc-#1}{#2}
109         \seq_log:N \g__tag_mc_stack_seq
110     }
111 }

```

(End definition for __tag_check_mc_pushed_popped:nn.)

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

```

112 \cs_new_protected:Npn \__tag_check_mc_tag:N #1 % #1 is var with a tag name in it
113 {

```

```

114 \tl_if_empty:NT #1
115 {
116   \msg_error:nnx { tag } {mc-tag-missing} { \__tag_get_mc_abs_cnt: }
117 }
118 \prop_if_in:NoF \g__tag_role_tags_NS_prop {#1}
119 {
120   \msg_warning:nnx { tag } {role-unknown-tag} {#1}
121 }
122 }

```

(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.

```
123 \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.

```

124 \cs_new_protected:Npn \__tag_check_mc_used:n #1
125 {
126   \seq_if_in:NnTF \g__tag_check_mc_used_seq {#1}
127   {
128     \msg_warning:nnn { tag } {mc-used-twice} {#1}
129   }
130   {
131     \seq_gput_right:Nx \g__tag_check_mc_used_seq {#1}
132   }
133 }

```

(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.

```

134 \cs_new_protected:Npn \__tag_check_show_MCID_by_page:
135 {
136   \tl_set:Nx \l__tag_tmpa_tl
137   {
138     \__tag_ref_value_lastpage:nn
139     {abspage}
140     {-1}
141   }
142   \int_step_inline:nnnn {1}{1}
143   {
144     \l__tag_tmpa_tl
145   }
146   {
147     \seq_clear:N \l_tmpa_seq
148     \int_step_inline:nnnn
149     {1}
150     {1}
151     {
152       \__tag_ref_value_lastpage:nn
153       {tagmcabs}
154       {-1}
155     }
156   }

```

```

156     {
157         \int_compare:nT
158         {
159             \__tag_ref_value:enn
160             {mcid-####1}
161             {tagabspage}
162             {-1}
163             =
164             ##1
165         }
166     {
167         \seq_gput_right:Nx \l_tmpa_seq
168         {
169             Page##1-####1-
170             \__tag_ref_value:enn
171             {mcid-####1}
172             {tagmcid}
173             {-1}
174         }
175     }
176 }
177 \seq_show:N \l_tmpa_seq
178 }
179 }

```

(End definition for `__tag_check_show_MCID_by_page:.`)

1.3.4 Miscellaneous

`__tag_check_record_pdfobj_num:n` This writes the object numbers. Not sure if needed, currently unused.

```

180 \cs_new_protected:Npn \__tag_check_record_pdfobj_num:n #1
181 {
182     \int_compare:nT { \l__tag_loglevel_int >= 3 }
183     {
184         \msg_info:nnx { tag } {obj-write-num} {#1}
185     }
186 }
187 </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.

	A	<code>\cs_new_protected:Npn</code> 34 , 42 , 49 , 55 ,
<code>attr-unknown</code>	23	59 , 69 , 85 , 93 , 100 , 112 , 124 , 134 , 180

	C		I
cs commands:		int commands:	
<code>\cs_generate_variant:Nn</code>	54	<code>\int_compare:nNnTF</code>	102 , 105

\int_compare:N	157, 182	\seq_show:N	177
\int_step_inline:nnnn	142, 148	\l_tmpa_seq	147, 167, 177
M			
mc-label-unknown	7	struct-faulty-nesting	13
mc-nested	5	struct-label-unknown	19
mc-not-open	9	struct-missing-tag	16
mc-popped	10	struct-no-objnum	12
mc-pushed	10	struct-show-closing	21
mc-tag-missing	6	struct-used-twice	17
mc-used-twice	8	sys-no-interwordspace	32
msg commands:			
\msg_error:nn	39, 57	T	
\msg_error:nnn	116	tag internal commands:	
\msg_info:nnn	51, 104, 184	__tag_check_add_tag_role:nn	69, 69
\msg_info:nnnn	78	__tag_check_info_closing_	
\msg_new:nnn	5,	struct:n	49, 49, 54
6, 7, 8, 9, 10, 11, 12, 13, 16, 17, 19,		__tag_check_mc_if_nested: ..	85, 85
21, 23, 24, 25, 26, 27, 28, 29, 31, 32		__tag_check_mc_if_open:	85, 93
\msg_warning:nnn		__tag_check_mc_pushed_popped:nn	
.. 46, 66, 73, 81, 89, 97, 108, 120, 128		100, 100
N			
new-tag	27	__tag_check_mc_tag:N	112, 112
O			
obj-write-num	31	__tag_check_mc_used:n	124, 124
P			
prop commands:			
\prop_get:NnNTF	61, 76	g__tag_check_mc_used_seq	
\prop_if_in:NnTF	36, 44, 118	123, 126, 131
\prop_item:Nn	22	__tag_check_no_open_struct:	55, 55
\ProvidesExplPackage	3	__tag_check_record_pdfobj_num:n	
R			
role-missing	24	180, 180
role-tag	27	__tag_check_show_MCID_by_page: .	
role-unknown	24	134, 134
role-unknown-tag	24	__tag_check_struct_used:n ..	59, 59
S			
seq commands:			
\seq_clear:N	147	__tag_check_structure_has_tag:n	
\seq_gput_right:Nn	131, 167	34, 34
\seq_if_in:NnTF	126	__tag_check_structure_tag:N	42, 42
\seq_log:N	109	__tag_get_mc_abs_cnt: ...	89, 97, 116
\seq_new:N	123	l__tag_loglevel_int ..	103, 106, 182
tl commands:			
		__tag_mc_if_in:TF	87, 95
		g__tag_mc_stack_seq	109
		__tag_ref_value:nnn	62, 159, 170
		__tag_ref_value_lastpage:nn	138, 152
		g__tag_role_tags_NS_prop	118
		g__tag_role_tags_prop	44, 76
		l__tag_tmpa_tl	136, 144
		tree-mcid-index-wrong	29