The toc-tagging-functions package
Tagpdf functions related to the tagging of toc and similar lists

LaTeX Project*

Version 0.81 2023-02-10

Abstract

Header for the testphase package

```
1 ⟨*header⟩
2 \ProvidesExplPackage {latex-lab-testphase-toc-tagging} {2023-02-10} {0.81}
3   { commands related to the tagging of toc-like lists}
4 ⟨/header⟩
```

1 Introduction

The followings contains various functions related to the tagging of the table of contents and similar list. The functions will at some time be moved inside the tagpdf code.

```
5 ⟨*package⟩
6 ⟨@@=tag⟩
```

2 Temporary variables

\l__tag_toc_tmpa_tl

```
7 \tl_new:N \l__tag_toc_tmpa_tl
```

(*End definition for* `\l__tag_toc_tmpa_tl`.)

3 General struct commands

The following variables and commands are not restricted to toc, but probably will be need in other places too.

\g__tag_struct_dest_num_prop

This variable records for (some or all, not clear yet) destination names the related structure number to allow to reference them in a Ref. The key is the destination. Moved into tagpdf!

*Initial implementation done by Ulrike Fischer

We use `\refstepcounter` to store the relation between destination names/`\@currentHref` and structure numbers

TODO: the functions should be moved into tagpdf so that one doesn't has to test if the prop exists or not.

```
\AddToHook{cmd/refstepcounter/after}
 {
   \tl_if_blank:VF \@currentHref
    {
      \prop_if_exist:NT \g__tag_struct_dest_num_prop
        {
         \prop_gput:Nxx \g__tag_struct_dest_num_prop {\@currentHref}{\tag_get:n{struct_num}}
        }
    }
 }
\AddToHook{cmd/H@refstepcounter/after}
 {
   \tl_if_blank:VF \@currentHref
    {
      \prop_if_exist:NT \g__tag_struct_dest_num_prop
        {
          \prop_gput:Nxx \g__tag_struct_dest_num_prop {\@currentHref}{\tag_get:n{struct_num}}
        }
    }
 }
```

`\g__tag_struct_ref_by_dest_prop`

This variable contains structures whose Ref key should be updated at the end to point to structured related with this destination. As this is probably need in other places too, it is not only a toc-variable. Moved into tagpdf!

`\g__tag_struct_ref_by_dest:`

This command is executed and update the Ref keys of the structures listed in `\g_@@_struct_ref_by_dest_prop`. It is currently only relevant for the `TOCI`. But other structures could need that later too. The command is executed in the `tagpdf/finish/before` hook.

```
\msg_new:nnn { tag } {struct-dest-unknown}
 {
   Destination~#1~has~no~related~structure.\\
   /Ref~for~structure~#2~not~updated
 }

\cs_new_protected:Npn \g__tag_struct_ref_by_dest:
  {
    \prop_map_inline:Nn\g__tag_struct_ref_by_dest_prop
      {
        \prop_get:NnNTF \g__tag_struct_dest_num_prop {##2} \l__tag_tmpa_tl
          {
            \__tag_struct_gput_data_ref:ee
              { ##1 }
              { \tag_struct_object_ref:e{ \l__tag_tmpa_tl }}
          }
          {
            \msg_warning:nnnn {tag}{struct-dest-unknown}{##2}{ ##1}
```

```
          }
      }
  }
\hook_gput_code:nnn {tagpdf/finish/before}{tagpdf/struct/Ref}{\g__tag_struct_ref_by_dest:}
```

(*End definition for* `\g__tag_struct_ref_by_dest:`.)

4 Toc code

`\g__tag_toc_level_int`
`\g__tag_toc_stack_seq`

`\g_@@_toc_level_int` records in a toc the current absolute level. We must close open structures at the end of the toc, for this we maintain a stack `\g_@@_toc_stack_seq`.

```
\int_new:N \g__tag_toc_level_int
\seq_new:N \g__tag_toc_stack_seq
```

`\_tag_toc_starttoc_init:n`

The init code clears the stack, and set the level to -100 and start to TOC structure. We also disable paratagging. The `/Title` is currently simply the type, but this could be done nicer.

```
\cs_new_protected:Npn \__tag_toc_starttoc_init:n #1
 {
    \bool_set_false:N \l__tag_para_bool
    \seq_gclear:N \g__tag_toc_stack_seq
    \int_gset:Nn  \g__tag_toc_level_int {-100}
    \tag_struct_begin:n{tag=TOC,title=#1}
 }
```

Now map it into the config point:

```
\cs_set_protected:Npn\@starttoc@cfgpoint@before#1
  {
    \__tag_toc_starttoc_init:n{#1}
  }
```

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

`\__tag_toc_starttoc_finalize:`

```
\cs_new_protected:Npn \__tag_toc_starttoc_finalize:
 {
    \int_step_inline:nn
      {\seq_count:N \g__tag_toc_stack_seq }
      {\tag_struct_end:}
    \tag_struct_end:
    \seq_gclear:N \g__tag_toc_stack_seq
 }
```

Now map it into the config point:

```
\cs_set_protected:Npn\@starttoc@cfgpoint@after#1
  {
    \__tag_toc_starttoc_finalize:
  }
```

(*End definition for* `\__tag_toc_starttoc_finalize:`.)

__tag_toc_end:n This commands ends all TOC on the stack with a level higher than the argument

```
\cs_new_protected:Npn \__tag_toc_end:n #1
 {
   \seq_get:NNT\g__tag_toc_stack_seq \l__tag_toc_tmpa_tl
     {
       \bool_lazy_and:nnT
         {
           \str_if_eq_p:ee{\tl_head:N\l__tag_toc_tmpa_tl}{TOC}
         }
         {
           \int_compare_p:nNn {#1}<{\tl_tail:N \l__tag_toc_tmpa_tl}
         }
         {
           \seq_gpop:NN\g__tag_toc_stack_seq \l__tag_toc_tmpa_tl
           \tag_struct_end:
           \__tag_toc_end:n{#1}
         }
     }
 }
\cs_generate_variant:Nn \__tag_toc_end:n {e}
```

(*End definition for* `\__tag_toc_end:n`.)

__tag_toc_contentsline_begin:nnn This is main command executed at the begin of a `\contentsline`.

```
\cs_new_protected:Npn \__tag_toc_contentsline_begin:nnn #1 #2 #3 %#1 level, #2 content, #3 de
  {
   \tag_if_active:T
     {
```

We detect the intended level by checking the value of `toclevel@...` (currently only provided by hyperref, but should be there always). To be on the safe side we set it to 1 if not defined.

```
       \ExpandArgs{c}\providecommand { toclevel@#1 }{ 1 } %  just in case ...
       \int_compare:nNnF { \use:c{toclevel@#1} } > {\use:c{c@tocdepth}}
        {
```

if level goes up, start new sub TOC unless we are at the begin

```
         \bool_lazy_and:nnT
           { \int_compare_p:nNn { \g__tag_toc_level_int } > {-100} }
           { \int_compare_p:nNn { \use:c{toclevel@#1} }   > { \g__tag_toc_level_int } }
           {
             \seq_gpush:Nx \g__tag_toc_stack_seq {{TOC}\use:c{toclevel@#1}}
             \tag_struct_begin:n{tag=TOC}
           }
```

if level goes down close all TOC's with a higher level

```
         \int_compare:nNnT
           { \use:c{toclevel@#1} } < { \g__tag_toc_level_int }
           {
             \__tag_toc_end:e { \use:c{toclevel@#1} }
           }
```

if same level do nothing update toclevel to the current level.

```
         \int_gset:Nn \g__tag_toc_level_int { \use:c{toclevel@#1} }
```

now open the TOCI, the tagging of the inner structure is left to the `\l@xxx` commands. setting the title is not strictly necessary but looks nicer but we have to remove the `\numberline`

```
          \group_begin:
           \text_declare_expand_equivalent:Nn \numberline \use_none:n
           \exp_args:Nx \tag_struct_begin:n{tag=TOCI,title={\text_purify:n {#2}}}
```

The TOCI structure should get a /Ref, so we put a request with its destination name into the prop.

```
           \prop_gput:Nxx \g__tag_struct_ref_by_dest_prop
             { \tag_get:n {struct_num} }{#3}
           \seq_gpush:Nx \g__tag_toc_stack_seq {{TOCI}\use:c{toclevel@#1}}
          \group_end:
      }
    }
  }
```

Now map it into the config point:

```
\cs_set_protected:Npn\@contentsline@cfgpoint@before#1#2#3#4
  {
    \__tag_toc_contentsline_begin:nnn {#1}{#2}{#4}
  }
```

(*End definition for* `\__tag_toc_contentsline_begin:nnn`.)

`\__tag_toc_contentsline_end:n` This is the closing code of a `\contentsline`. If the contentsline was actually printed, the code has to close the TOCI structure and to update the stack.

```
\msg_new:nnn {tag}{toc-no-TOCI}{Missing~TOCI~structure~on~toc~stack}

\cs_new_protected:Npn \__tag_toc_contentsline_end:n #1 %#1 level name
  {
    \int_compare:nNnF { \use:c{toclevel@#1} } > {\use:c{c@tocdepth}}
      {
        \seq_gpop:NNT \g__tag_toc_stack_seq\l__tag_tmpa_tl
          {
            \str_if_eq:eeTF{\tl_head:N\l__tag_tmpa_tl}{TOCI}
             {
               \tag_struct_end:
             }
             {
               \msg_warning:nn{tag}{toc-no-TOCI}
             }
          }
      }
  }
```

Now we map it to the config point

```
\cs_set_protected:Npn \@contentsline@cfgpoint@after #1#2#3#4
 {
   \__tag_toc_contentsline_end:n {#1}
 }
```

(*End definition for* `\__tag_toc_contentsline_end:n`.)

4.1 Tagging of the content

This need discussion.

```
\AddToHook{contentsline/text/before}[tagpdf]{%
  \tag_struct_begin:n{tag=Reference}%
  \tag_mc_begin:n{tag=Reference}}
\AddToHook{contentsline/text/after}[tagpdf]{%
  \tag_mc_end:}
\AddToHook{contentsline/page/before}[tagpdf]{%
  \tag_mc_begin:n{tag=Reference}}
\AddToHook{contentsline/page/after}[tagpdf]{%
  \tag_mc_end:
  \tag_struct_end:} %Reference
\AddToHook{contentsline/number/before}[tagpdf]{%
  \tag_mc_end:
  \tag_struct_begin:n{tag=Lbl}%
  \tag_mc_begin:n{tag=Lbl}}
\AddToHook{contentsline/number/after}[tagpdf]{%
  \tag_mc_end:
  \tag_struct_end:
  \tag_mc_begin:n{tag=Reference}}
\def\@dottedtocline@cfgpoint@leaders#1{%
 \tag_mc_begin:n{artifact}\nobreak#1\nobreak\tag_mc_end:}
⟨/package⟩
```