

The latex-lab-bookmark package

Creating bookmarks

L^AT_EX Project*

v0.95 2025-10-13

Abstract

The following code implements a first draft for the creation of bookmarks with kernel methods.

1 Introduction

This package implements support for the PDF outline aka bookmarks based on the `l3pdfoutline` module in the PDF management. That means it requires `\DocumentMetadata` and does not work without it. It is based on the `bookmark` and the `hyperref` package and mostly implements the same user commands, most importantly the main commands `\bookmark` and `\bookmarksetup`.¹ But it does not try to be a full replacement and to replicate every option in the same way as the other packages.

1.1 Usage

The package should currently be loaded with the `testphase` key:

```
\DocumentMetadata{testphase=bookmark}  
\documentclass{...}
```

If loaded it will replace the code used for bookmarks of sectioning commands and the like in the `hyperref` and `bookmark` packages. It is possible to manually create bookmarks with this package even if `hyperref` or `bookmark` are not loaded, but then the targets must be created manually too, e.g. with `pdf_destination:nn`.

The package assumes that the input files are UTF-8 encoded!

1.2 User commands

`\bookmark` `\bookmark[<keyval options>]{<title text>}`

This is the main command. As `keyval` options it accepts the key below (which are more or less the same as the keys from the `bookmark` package). At least one action key is required!

*Initial implementation done by Ulrike Fischer

¹There is no clash here, as documents can use only one bookmark system: mixing commands from more than one package or using manually the primitives is not really possible without breaking the tree.

<code>\bookmarksetup</code>	<code>\bookmarksetup{<keyval options>}</code>
-----------------------------	---

This can be used to change the behaviour of following bookmarks. It can also be used in the hook `bookmark`.

<code>\pdfbookmark</code>	<code>\pdfbookmark[<[l]>level]{<text>}{<name>}</code>
<code>\subpdfbookmark</code>	
<code>\belowpdfbookmark</code>	
<code>\currentpdfbookmark</code>	

These commands are defined if `hyperref` is loaded and work as described in the `hyperref` documentation. They all not only create a bookmark with a GoTo link but also with `{<name>}` a destination for this bookmark.

1.3 Configuring bookmarks connected to table of contents entries

Bookmarks can in a document be added by two methods: manually with commands like `\pdfbookmark` or `\bookmark` and automatically through heading commands. For the second method `hyperref` hooks into the `\addcontentsline` command which means that only headings which can at least potentially appear in a table of contents are added to the bookmarks — the standard starred headings currently not. The following keys allow to configure the bookmarks added with this second method².

bookmarksnumbered, numbered This allows to decide if the bookmarks show numbers. The code internally uses a socket `hyp/outline/numberformat`. The predeclared plugs of these socket redefine `\numberline`, `\booknumberline`, `\partnumberline` and `\chapternumberline` to gobble their argument (plug `false`), or to print it with a following space (plug `true`). Other plugs can be defined by users or packages and be assigned with `numbered=<plugname>`.

bookmarkstype,lists By default `hyperref` and `bookmark` add only entries that go into the main table of contents (`toc`) to the outline. With these keys this can be changed. Differently to `hyperref`, the keys here accepts a list of extensions, so e.g. with `lists={toc,lof,lot}` it is possible to add entries from all three lists.

1.4 Hooks

Code to create bookmarks is typically hidden inside other commands. This makes it difficult to override settings done in their optional argument. There is therefore a hook `bookmark` to add additional options. It can for example change keys if you add a `\bookmarksetup{...}` to it.

1.5 Level keys

While bookmarks use relative levels, a user normally wants to set levels relative to document sectioning but implementing a suitable interface is not trivial as bookmarks can't skip levels. So e.g. if you use in this order a section, a chapter, a part, a chapter, a section, then the first section, chapter and the part will all be in the bookmarks in level 1, while the second chapter will be a child of the part and so in level 2, and the next section in level 3. This is different to the printed table of contents where both chapters and both sections look alike and so are perceived to be on the same respective level.

It is therefore not possible to map the level of a sectioning command (as seen by a user and by the `tocdepth` and `secnumdepth` counter) to a specific level in the bookmarks.

²As mentioned above `hyperref` is currently required for these bookmarks

The code used here (and also in the bookmark package) tries to address this problem, by storing for every bookmark not only the actual *bookmark level* but also the (intended) *document level*. The *document level* of the last bookmark is the *current document level*. If a new bookmark then requests to be set in a specific document level, the code looks up the levels of the previous bookmarks and chooses the best fit as parent. For example, if there is a chapter (document level 0, bookmark level 1) with a subsection as child (document level 2, bookmark level 2) and the document now wants to insert a section bookmark (document level 1), it will insert it as child of the chapter and sibling of the subsection.

level The value is the requested *document level*. It can be given as integer expression (positive or negative) or with keywords like **section** if a corresponding `\toclevel@section` command exists that expands to an integer.

If the level value n is larger then the current document level, the bookmark is inserted as child. If the two values are equal, the bookmark is inserted as sibling. If n is smaller it looks up the parents until it finds a bookmark with a document level equal to n or smaller to n . In the first case it is inserted as a sibling, in the second as a child of this bookmark.

In all cases the current document level is updated to n

bookmarksdepth, depth The value of these keys decides if a bookmark is shown at all. The value is a *document level* and accepts the same values as the **level** key. A bookmark is suppressed if its document level value is larger than the depth value.

rellevel This sets the level of a new bookmark in relation to the previous. It is in such a tree the natural way to set a level and quite easy to use and implement: a positive value (the actual value doesn't matter) produces a child, zero a sibling, a negative value goes up by the number of steps given but at most the number of steps needed to reach the root level. The value will also be used to update the current document level.

startatroot This goes up to the root, the same can be achieved by using **rellevel** with a large negative number (and this is also how it is implemented). By default it will set the current document level to zero, the value can be changed by using `startatroot=$n`, the setting is suppressed if **keeplevel** has been set first.

1.6 Expanding commands in the bookmarks

Bookmarks can only show simple text. L^AT_EX commands must there be converted into something simple. `hyperref` uses here `\pdfstringdef`. The code here uses kernel methods, this can lead to different output e.g. if math is used. The conversion code can be switched back to `\pdfstringdef` (if `hyperref` is loaded) by reassigning a socket plug (the default plug is called `default`):

```
\AssignSocketPlug{hyp/outline/title}{pdfstringdef}
```

Both methods understand the `\texorpdfstring` command.

1.7 Actions

The following actions are supported by this package (the list is quite similar to the `bookmark` package).

dest The value is a destination name. That is the action normally used in bookmarks. When creating a tagged PDF it will also link to the relevant structure destination.

page The value is a (absolute) page number. It creates a destination to the relevant page. It can be used together with the `gotor` key and will then create a link to the page in the external PDF. The `view` key can be used to control the view/zoom of the page. When creating a tagged PDF one should avoid this key for internal links as the page destination has no associated structure destination.

gotor The value is the file name of an external PDF (with extension).

named The value is one of `FirstPage`, `LastPage`, `NextPage`, `PrevPage`.

uri The value is url. The value is automatically percent encoded. The hash and percent chars must be properly escaped:

```
\bookmark[uri=https://www.example.com\#abc]{uri with hash}
\bookmark[uri=https://www.example.com\%abc]{uri with percent}
\bookmark[uri=https://www.köln.de]{uri with non-ascii char}
```

view The value is one of `Fit`, `FitB`, `FitH`, `FitV`, `FitBH`, `FitBV` which can be followed by a positive integer (separated by a space) or the keyword `null`. Or it can be `XYZ`. This can be followed (separated by spaces) by up to two positive integers and one decimal or keywords `null` which are then taken as *top left zoom* in this order. *zoom* is a factor, so e.g. 0.5 will give a scaling of 50%.

rawaction This is simply passed into the `/A` of the action. Check the PDF reference to find out what is possible

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

3 \ProvidesExplPackage{latex-lab-testphase-bookmark}{\ltxlabbookmarkdate}{\ltxlabbookmarkversion}
4 {PDF bookmarks with kernel methods}%
```

2 Implementation

2.1 Variables

```
\g_hyp_outline_currentlevel_int
\g_hyp_outline_level_intarray
```

While bookmarks use relative levels, user set levels relative to document sectioning.

```
5 \int_new:N \g_hyp_outline_currentlevel_int
6 \int_new:N \l_hyp_outline_level_int
7 \intarray_new:Nn \g_hyp_outline_level_intarray {2000} %TODO size??
8 \intarray_gset:Nnn \g_hyp_outline_level_intarray {1}{0}
9 \int_new:N \l_hyp_outline_depth_int
10 \tl_new:N \l_hyp_outline_title_tl
11 \tl_new:N \l_hyp_outline_action_data_tl
12 \tl_new:N \l_hyp_outline_action_dest_data_tl
```

```

13 \tl_new:N      \l__hyp_outline_action_view_tl
14 \bool_new:N    \l__hyp_outline_keeplevel_bool
15 \seq_new:N      \l__hyp_outline_type_seq
16 \bitset_new:Nn \l__hyp_outline_action_bitset
17 {
18     goto = 1,
19     gotor = 2,
20     named = 3,
21     uri = 4,
22     raw = 5,

```

this two are destination types that apply to both goto and gotor

```

23     namedest = 6,
24     pagedest = 7
25 }
26 \bitset_set_true:Nn \l__hyp_outline_action_bitset { goto }
27
28 \int_new:N      \l__hyp_outline_tmpa_int
29 \int_new:N      \l__hyp_outline_rellevel_tmpa_int
30 \tl_new:N      \l__hyp_outline_parent_tmpa_tl
31 \seq_new:N      \l__hyp_outline_tmpa_seq
32 \tl_new:N      \l__hyp_outline_tmpa_tl
33 \str_new:N      \g__hyp_outline_tmpa_str

```

(End of definition for `\g__hyp_outline_currentlevel_int` and `\g__hyp_outline_level_intarray`.)

2.2 Commands shared with hyperref

The following commands are also needed in the hyperref driver. They use here a different prefix, but this should be merged and shared.

2.2.1 TeX or PDF?

```

34 \newif\ifHy@pdfstring
35 \def\texorpdfstring{%
36     \ifHy@pdfstring
37         \expandafter\@secondoftwo
38     \else
39         \expandafter\@firstoftwo
40     \fi
41 }

```

2.2.2 Converting text

The title text of the bookmark and other strings must be converted to strings suitable for a PDF. With hyperref on can use `\pdfstringdef` but we also provide a native version. This follows the implementation in the generic hyperref driver but is currently simpler – both should be merged and extended to match the hyperref command.

```

42 \cs_new_protected:Npn \__hyp_outline_pdfstring:nnN #1 #2 #3
43 {
44     \group_begin:
45     \Hy@pdfstringtrue
46     \tl_set:Nn\l__hyp_outline_tmpa_tl{#2}

```

to be able to remove/format the `\numberline` we must remove `\protect`.

```

47 \tl_remove_all:Nn\l__hyp_outline_tmpa_tl{ \protect }
48 \__hyp_pdfstring_equivalents:
49 \str_set:Ne #3 {\text_purify:n { \l__hyp_outline_tmpa_tl } }
50 \pdf_string_from_unicode:nVN { #1 } #3 #3
51 \str_gset_eq:NN \g__hyp_outline_tmpa_str #3
52 \group_end:
53 \str_set_eq:NN #3 \g__hyp_outline_tmpa_str
54 }

55 \socket_new:nn {hyp/outline/title}{2}
56 \socket_new_plug:nnn {hyp/outline/title}{pdfstringdef}
57 {
58 \pdfstringdef#2{#1}
59 }

```

This uses the kernel `pdfstringdef`. We use that socket by default. Ensure that a text without parentheses is produced by using `string-raw`!

```

60 \socket_new_plug:nnn {hyp/outline/title}{default}
61 {
62 \__hyp_outline_pdfstring:nnN{utf16/string-raw}{#1}#2
63 }
64 \socket_assign_plug:nn {hyp/outline/title}{default}

```

2.2.3 Equivalent definition for pdf strings

Similar to `\pdfstringdef` we need to declare various equivalents TODO: add a user hook.

```

65 \cs_new_protected:Npn \__hyp_pdfstring_equivalents:
66 {
67 \text_declare_expand_equivalent:Nn\MakeUppercase{\__hyp_outline_makeuppercase:n}
68 \text_declare_expand_equivalent:Nn\MakeLowercase{\__hyp_outline_makelowercase:n}
69 \cs_set:Npn\dots{...} %U+2026
70 \cs_set:Npn\ldots{...}
71 \cs_set:Npn\P{¶}
72 \cs_set:Npn\TeX{TeX}
73 \cs_set:Npn\LaTeX{LaTeX}
74 \cs_set:Npn\LaTeXe{LaTeX2 }
75 \cs_set:Npn\SliTeX{SliTeX}%
76 \cs_set:Npn\MF{Metafont}%
77 \cs_set:Npn\MP{Metapost}%
78 \cs_set_eq:NN\phantom\use_none:n
79 \cs_set_eq:NN\vphantom\use_none:n
80 \cs_set_eq:NN\hphantom\use_none:n
81 % \ding ?
82 % \Cube ?
83 % \begin ?
84 % \end ?
85 \cs_set_eq:NN\textcolor\use_ii:nn
86 % what to do with all the \textxxx commands?
87 \text_declare_expand_equivalent:Nn\textparagraph{¶}
88 \text_declare_expand_equivalent:Nn\textdollar{\$}

```

```

89 \pdfstringdefPreHook
90 }

```

We do not want the hyperref names

```

91 \cs_new:Npn \__hyp_outline_makeuppercase:n#1
92   {\use:e
93     {\exp_args:Ne
94       \text_uppercase:nn{\BCPdata { casing }}{#1}
95     }
96   }
97 \cs_new:Npn \__hyp_outline_makelowercase:n #1
98   {\use:e
99     {\exp_args:Ne
100       \text_lowercase:nn{\BCPdata { casing }}{#1}
101     }
102   }

```

2.2.4 Regex to check the view value

```

103 \regex_const:Nn \c__hyp_outline_dest_startview_regex
104 {
105   \A\ *
106   (?
107     (?XYZ (?:\ +(?:\d+|\d*\.\d+)|null)){3}\ )
108     |
109     (?Fit\b|FitB\b)
110     |
111     (?:FitH|FitV|FitBH|FitBV)(?:\ +(?:\d+|\d*\.\d+)|\ +null){1}
112     |
113     (?FitR (?:\ +\d+|\ +\d*\.\d+){4}\ )
114   )
115 }
116
117 \msg_new:nnn
118 { hyp /outline }
119 { invalid-view-value }
120 {
121   Invalid~value~'#1'~of~'#2'  \
122   is~replaced~by~'Fit'~\msg_line_context:.
123 }
124

```

2.3 Messages

```

125 \msg_new:nnn {hyp/outline}{no-action}
126 {
127   bookmark~action~missing!\
128   Use~one~of~'dest',~'gotor',~'named',~'uri',~or~'rawaction'
129 }

```

2.4 Formatting the bookmark

```

130 \socket_new:nn {hyp/outline/numberformat}{0}
131 \socket_new_plug:nnn {hyp/outline/numberformat}{false}

```

```

132 {
133   \cs_set_eq:NN\numberline\use_none:n
134   \cs_set_eq:NN\booknumberline\use_none:n
135   \cs_set_eq:NN\partnumberline\use_none:n
136   \cs_set_eq:NN\chapternumberline\use_none:n
137 }
138 \cs_new:Npn \__hyp_outline_use_numberline:n #1 {#1\c_space_tl}
139 \socket_new_plug:nnn {hyp/outline/numberformat}{true}
140 {
141   \cs_set_eq:NN\numberline      \__hyp_outline_use_numberline:n
142   \cs_set_eq:NN\booknumberline  \__hyp_outline_use_numberline:n
143   \cs_set_eq:NN\partnumberline  \__hyp_outline_use_numberline:n
144   \cs_set_eq:NN\chapternumberline\__hyp_outline_use_numberline:n
145 }

```

2.5 Key definitions

```

146 \keys_define:nn {hyp/outline}
147 {
148   ,bold .choice:
149   ,bold / true .code:n = \bitset_set_true:Nn \l_pdfoutline_F_bitset {Bold}
150   ,bold /false .code:n = \bitset_set_false:Nn\l_pdfoutline_F_bitset {Bold}
151   ,bold .default:n = true
152   ,italic .choice:
153   ,italic / true .code:n = \bitset_set_true:Nn\l_pdfoutline_F_bitset {Italic}
154   ,italic /false .code:n = \bitset_set_false:Nn\l_pdfoutline_F_bitset {Italic}
155   ,italic .default:n = true
156   ,numbered .code:n =
157     { \socket_assign_plug:nn { hyp/outline/numberformat } {#1}}
158   ,numbered .default:n = true
159   ,numbered .initial:n = false
160   ,color .tl_set:N = \l_pdfoutline_color_tl
161
162   ,open .bool_set:N = \l_pdfoutline_open_bool

```

TODO: make this perhaps a bit safer ...

```

163   ,depth .code:n =
164   {
165     \cs_if_exist:cTF {toclevel@#1}
166     { \int_set:Nn \l__hyp_outline_depth_int { \use:c {toclevel@#1} } }
167     { \int_set:Nn \l__hyp_outline_depth_int { #1 } }
168   }
169   ,bookmarksdepth .meta:n = { depth = #1 }
170   ,depth .initial:n = {\int_use:N \c_max_int}

```

openlevel is relative to the bookmark levels, not some document level!

```

171   ,openlevel .int_set:N = \l_pdfoutline_open_int
172   ,level .code:n =
173   {
174     \cs_if_exist:cTF {toclevel@#1}
175     { \int_set:Nn \l__hyp_outline_level_int { \use:c {toclevel@#1} } }
176     { \int_set:Nn \l__hyp_outline_level_int { #1 } }
177   }
178   ,rellevel .code:n =
179   {

```



```

180     \int_set:Nn
181     \l__hyp_outline_level_int
182     { \g__hyp_outline_currentlevel_int + #1 }
183   }
184   ,keeplevel .bool_set:N = \l__hyp_outline_keeplevel_bool
185   ,startatroot .code:n =
186   {
187     \int_set:Nn \l__hyp_outline_level_int { -1000 }
188     \bool_if:NF \l__hyp_outline_keeplevel_bool
189     {
190       \int_gset:Nn \g__hyp_outline_currentlevel_int {#1}
191       \bool_set_true:NF \l__hyp_outline_keeplevel_bool
192     }
193   }
194   ,startatroot .default:n = 0

```

dest applies to goto or gotor actions.

```

195   ,dest .code:n =
196   {
197     \int_compare:nNnTF
198     { \bitset_item:Nn \l__hyp_outline_action_bitset {gotor} } = {1}
199     {
200       \bitset_clear:N      \l__hyp_outline_action_bitset
201       \bitset_set_true:Nn  \l__hyp_outline_action_bitset { gotor }
202     }
203     {
204       \bitset_clear:N      \l__hyp_outline_action_bitset
205       \bitset_set_true:Nn  \l__hyp_outline_action_bitset { goto }
206     }
207     \bitset_set_true:Nn  \l__hyp_outline_action_bitset { namedest }
208     \tl_set:Nn\l__hyp_outline_action_dest_data_tl {#1}
209   }
210   ,dest .default:n = Doc-Start
211   ,goto .code:n =
212   {
213     \bitset_clear:N      \l__hyp_outline_action_bitset
214     \bitset_set_true:Nn  \l__hyp_outline_action_bitset { goto }
215     \bitset_set_true:Nn  \l__hyp_outline_action_bitset { namedest }
216     \tl_set:Nn\l__hyp_outline_action_dest_data_tl {#1}
217   }
218   ,gotor .code:n =
219   {
220     \int_compare:nNnTF
221     { \bitset_item:Nn \l__hyp_outline_action_bitset {pagedest} } = {1}
222     {
223       \bitset_clear:N      \l__hyp_outline_action_bitset
224       \bitset_set_true:Nn  \l__hyp_outline_action_bitset { pagedest }
225     }
226     {
227       \bitset_clear:N      \l__hyp_outline_action_bitset
228       \bitset_set_true:Nn  \l__hyp_outline_action_bitset { namedest }
229     }
230     \bitset_set_true:Nn  \l__hyp_outline_action_bitset { gotor }
231     \tl_set:Nn\l__hyp_outline_action_data_tl {#1}

```

```

232     }
233     ,named .choices:nn =
234     {FirstPage, NextPage, PrevPage, LastPage}
235     {
236         \bitset_clear:N      \l__hyp_outline_action_bitset
237         \bitset_set_true:Nn  \l__hyp_outline_action_bitset { named }
238         \tl_set:Nn \l__hyp_outline_action_data_tl {/#1}
239     }

```

page like dest applies to goto and gotor actions.

```

240     ,page .code:n =
241     {
242         \int_compare:nNnTF
243         { \bitset_item:Nn \l__hyp_outline_action_bitset {gotor} } = {1}
244         {
245             \bitset_clear:N      \l__hyp_outline_action_bitset
246             \bitset_set_true:Nn  \l__hyp_outline_action_bitset { gotor }
247         }
248         {
249             \bitset_clear:N      \l__hyp_outline_action_bitset
250             \bitset_set_true:Nn  \l__hyp_outline_action_bitset { goto }
251         }
252         \bitset_set_true:Nn  \l__hyp_outline_action_bitset { pagedest }
253         \tl_set:Nn \l__hyp_outline_action_dest_data_tl {#1}
254     }
255     ,page .default:n      = 1
256     ,rawaction .code:n =
257     {
258         \bitset_clear:N      \l__hyp_outline_action_bitset
259         \bitset_set_true:Nn  \l__hyp_outline_action_bitset { uri }
260         \tl_set:Nn \l__hyp_outline_action_data_tl {#1}
261     }
262     ,view .code:n =
263     {
264         \tl_set:Ne \l__hyp_outline_tmpa_tl {#1~null~null~null~}
265         \exp_args:NNV
266         \regex_extract_once:NnNTF
267         \c__hyp_outline_dest_startview_regex
268         \l__hyp_outline_tmpa_tl
269         \l__hyp_outline_tmpa_seq
270         {
271             \tl_set:Ne \l__hyp_outline_action_view_tl {/\seq_item:Nn \l__hyp_outline_tmpa_seq
272         }
273         {
274             \msg_warning:nnnn {hyp/outline}{invalid-view-value}{#1}{view}
275             \tl_set:Nn \l__hyp_outline_action_view_tl {Fit}
276         }
277     }
278     ,uri .code:n =
279     {
280         \bitset_clear:N      \l__hyp_outline_action_bitset
281         \bitset_set_true:Nn  \l__hyp_outline_action_bitset { uri }
282         \__hyp_outline_pdfstring:nnN {utf8/URI}{#1}\l__hyp_outline_action_data_tl
283         %\tl_set:Ne \l__hyp_outline_action_data_tl {(#1)}

```

```

284     }
285     ,lists .code:n =
286     { \seq_set_from_clist:Nn \l__hyp_outline_type_seq { #1 } }
287     ,lists .initial:n = toc
288     ,bookmarkstype .meta:n = {lists={#1}}
289 }

```

\

```

290 \NewDocumentCommand{\bookmarksetup}{m}{\keys_set:nn{hyp/outline}{#1}}

```

(End of definition for \. This function is documented on page ??.)

```

291 \NewHook{bookmark}

```

\bookmark

```

292 \NewDocumentCommand\bookmark{O{}m}
293 {
294   \group_begin:
295   \keys_set:nn {hyp/outline} { #1 }
296   \socket_use:nnn{hyp/outline/title}{#2}\l__hyp_outline_title_tl
297   \UseHook{bookmark}
298   \int_compare:nNnF {\l__hyp_outline_level_int} > {\l__hyp_outline_depth_int}
299   {
300     \__hyp_outline_bookmark_aux:
301   }
302   \group_end:
303 }

```

(End of definition for \bookmark. This function is documented on page 1.)

__hyp_outline_bookmark_aux:

```

304 \cs_new_protected:Npn \__hyp_outline_bookmark_aux:

```

At first we calculate the rellevel from the requested level.

```

305 {
306   \__hyp_outline_calc_rellevel:NN \l__hyp_outline_level_int \l__hyp_outline_rellevel_tmpa_int

```

Update the global current level

```

307   \bool_if:NF\l__hyp_outline_keeplevel_bool
308   {
309     \int_gset:Nn \g__hyp_outline_currentlevel_int { \l__hyp_outline_level_int }
310   }
311   \int_compare:nNnTF
312   {
313     \bitset_item:Nn \l__hyp_outline_action_bitset {goto} *
314     \bitset_item:Nn \l__hyp_outline_action_bitset {namedest}
315   } = {1}
316   {
317     \pdfoutline_goto:nee
318     { \l__hyp_outline_rellevel_tmpa_int } %level
319     { \l__hyp_outline_action_dest_data_tl }
320     { \l__hyp_outline_title_tl }
321   }

```

```

322 {
323   \int_case:nnF { \bitset_to_arabic:N \l__hyp_outline_action_bitset }
324   {
325     { 65 } % goto (1) + pagedest (65)
326     {
327       \pdfoutline_action:nee
328       { \l__hyp_outline_rellevel_tmpa_int }
329       { /S/GoTo~
330         /D
331         [
332           \pdf_pageobject_ref:n { \l__hyp_outline_action_dest_data_tl }~
333           \l__hyp_outline_action_view_tl
334         ]
335       }
336       { \l__hyp_outline_title_tl }
337     }
338     { 34 } % gotor (2) + namedest (32)
339     {
340       \pdfoutline_action:nee
341       { \l__hyp_outline_rellevel_tmpa_int }
342       { /S/GoToR~
343         /F ( \l__hyp_outline_action_data_tl ) %Todo check format
344         /D ( \l__hyp_outline_action_dest_data_tl ) %Todo check format
345       }
346       { \l__hyp_outline_title_tl }
347     }
348     { 66 } % gotor (2) + pagedest (64)
349     {
350       \pdfoutline_action:nee
351       { \l__hyp_outline_rellevel_tmpa_int }
352       { /S/GoToR~
353         /F ( \l__hyp_outline_action_data_tl ) %Todo check format
354         /D
355         [
356           \pdf_pageobject_ref:n { \l__hyp_outline_action_dest_data_tl }~
357           \l__hyp_outline_action_view_tl
358         ]
359       }
360       { \l__hyp_outline_title_tl }
361     }
362   }
363   { 4 } % named (4)
364   {
365     \pdfoutline_action:nee
366     { \l__hyp_outline_rellevel_tmpa_int }
367     { /S/Named~
368       /N \l__hyp_outline_action_data_tl %TODO check formt
369     }
370     { \l__hyp_outline_title_tl }
371   }
372 }
373 { 8 } % uri (8)
374 {
375   \pdfoutline_action:nee

```

```

376         { \l__hyp_outline_rellevel_tmpa_int }
377         { /S/URI~
378           /URI \l__hyp_outline_action_data_tl %TODO check formt
379         }
380         { \l__hyp_outline_title_tl }
381       }
382     { 16 } % raw (16)
383     {
384       \pdfoutline_action:nee
385       { \l__hyp_outline_rellevel_tmpa_int }
386       {
387         \l__hyp_outline_action_data_tl %TODO check formt
388       }
389       { \l__hyp_outline_title_tl }
390     }
391   }
392   { \msg_error:nn {hyp/outline}{no-action} }
393 }
394 \intarray_gset:Nnn
395   \g__hyp_outline_level_intarray
396   { \pdfoutline_id_ref_last: }
397   { \g__hyp_outline_currentlevel_int }
398 } %

```

(End of definition for __hyp_outline_bookmark_aux:.)

__hyp_outline_calc_rellevel:NN This function computes the relative level.

```

399 \cs_new_protected:Npn\__hyp_outline_calc_rellevel:NN #1 #2 % #1 the requested level, #2 the r
400 {
401   \int_compare:nNnTF
402     { #1 }
403     <
404     { \g__hyp_outline_currentlevel_int }

```

if the requested level is lower, we must inspect the parents to find the best fit. starting with the previous bookmark

```

405   {
406     \int_set:Nn #2 { 0 }

```

get the first parent

```

407     \tl_set:Ne \l__hyp_outline_parent_tmpa_tl
408     { \pdfoutline_parent_ref:n { \pdfoutline_id_ref_last: } }

```

get the document level of the first parent.

```

409     \int_compare:nNnTF
410       { \l__hyp_outline_parent_tmpa_tl } > { 0 }
411     {
412       \int_set:Nn \l__hyp_outline_tmpa_int
413       {
414         \intarray_item:Nn
415           \g__hyp_outline_level_intarray
416           { \l__hyp_outline_parent_tmpa_tl }
417       }

```

```

418     }
419     { \int_set_eq:NN \l__hyp_outline_tmpa_int #1 }
420 % if \l__hyp_outline_tmpa_int = #1
421 % stop, decr rellevel (this sets it as sibling of the parent)
422 % if \l__hyp_outline_tmpa_int < #1
423 % stop (this sets it as child of the parent as rellevel is unchanged)
424 % if \l__hyp_outline_tmpa_int > #1
425 % get next parent and continue.
426     \int_do_while:nNnn
427     { \l__hyp_outline_tmpa_int }
428     >
429     { #1 }
430     {
431         \int_case:nn
432         {
433             \int_sign:n
434             { \l__hyp_outline_tmpa_int-#1 }
435         }
436         {
437             { 0 } { \int_decr:N #2 }
438             { 1 }
439             {
440                 \int_decr:N #2
441                 \tl_set:Ne \l__hyp_outline_parent_tmpa_tl
442                 { \pdfoutline_parent_ref:n { \l__hyp_outline_parent_tmpa_tl } }

```

We need to check if we reached the root level.

```

443     \int_compare:nNnTF
444     { \l__hyp_outline_parent_tmpa_tl } > { 1 }
445     {
446         \int_set:Nn \l__hyp_outline_tmpa_int
447         {
448             \intarray_item:Nn
449             \g__hyp_outline_level_intarray
450             { \l__hyp_outline_parent_tmpa_tl }
451         }
452     }
453     {
454         % root
455         \int_set_eq:NN \l__hyp_outline_tmpa_int #1
456     }
457 }
458 }
459 }
460 \int_if_zero:nT {\l__hyp_outline_tmpa_int-#1}{\int_decr:N #2}
461 }

```

requested level \geq current level is the easy case: we create either a sibling or a child:

```

462 {
463     \int_set:Nn #2
464     { #1 - \g__hyp_outline_currentlevel_int }
465 }
466 }

```

(End of definition for `_hyp_outline_calc_rellevel:NN`.)

2.6 Replacing hyperref commands

```

467 \RequirePackage{latex-lab-testphase-toc}
468 \disable@package@load{bookmark}{\RequirePackage{hyperref}}
469 \AddToHook{package/hyperref/after}
470 {
471   \RemoveFromHook{cmd/addcontentsline/before}[hyp]
472   \AddToHookWithArguments{cmd/addcontentsline/before}[hyp/outline]
473     { \_hyp_addcontentsline_bookmark:nnn {#1}{#2}{#3} }
474   \legacy_if:nF{Hy@bookmarks}{\bool_set_false:N\l_pdfoutline_bool}
475   \renewcommand*{\pdfbookmark}[3][0]{%
476     \bookmark[level=#1,dest={#3.#1}]{#2}%
477     \hyper@anchorstart{#3.#1}\hyper@anchorend}
478   \def\currentpdfbookmark#1#2#3{%
479     \bookmark[rellevel=0,dest={#3.#1}]{#2}%
480     \hyper@anchorstart{#3.#1}\hyper@anchorend}
481   \def\subpdfbookmark#1#2#3{%
482     \bookmark[rellevel=1,dest={#3.#1}]{#2}%
483     \hyper@anchorstart{#3.#1}\hyper@anchorend}
484   \def\belopdfbookmark#1#2#3{%
485     \bookmark[keeplevel,rellevel=1,dest={#3.#1}]{#2}%
486     \hyper@anchorstart{#3.#1}\hyper@anchorend}
487 }
488 \DeclareHookRule{package/hyperref/after}{latex-lab-testphase-bookmark}{after}{latex-
lab-testphase-toc}

```

`_hyp_addcontentsline_bookmark:nnn`

This is the command used at the begin of `\addcontentsline` which creates the book-
mark.

```

489 \cs_new_protected:Npn \_hyp_addcontentsline_bookmark:nnn #1 #2 #3 % #1 toc type, #2 level, #
490 {
491   \seq_if_in:NeT \l_hyp_outline_type_seq { #1 }
492   {
493     \tl_if_empty:NT\@currentHref
494     { \MakeLinkTarget[page] {} }
495     \tl_if_exist:cF { toclevel@#2 }
496     {
497       \tl_new:c { toclevel@#2 }
498       \tl_gset:cn { toclevel@#2 } { 0 }
499       % message
500     }
501     \group_begin:
502     \socket_use:n{hyp/outline/numberformat}
503     \bookmark[level=#2,dest={\HyperDestNameFilter{\@currentHref}}]{#3}
504     \group_end:
505   }
506 }

```

(End of definition for `_hyp_addcontentsline_bookmark:nnn`.)

```

507 \end{package}

508 \end{latex-lab}
509 \ProvidesFile{bookmark-latex-lab-testphase.ltx}

```

```
510      [\ltxlabbookmarkdate\space v\ltxlabbookmarkversion\space latex-lab wrapper bookmark]  
511 \RequirePackage{latex-lab-testphase-bookmark}  
512 \ltxlab
```