

The l3backend-testphase package

Additional backend PDF features

L^AT_EX PDF management testphase bundle

The L^AT_EX Project*

Version 0.95k, released 2022-01-28

1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2022-01-28}{ }
4   {LaTeX~PDF~management~testphase~bundle~backend~support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2022-01-28}{ }
8   {LaTeX~PDF~management~testphase~bundle~backend~support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2022-01-28}{ }
12   {LaTeX~PDF~management~testphase~bundle~backend~support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2022-01-28}{ }
16   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2022-01-28}{ }
20   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2022-01-28}{ }
24   {LaTeX~PDF~management~testphase~bundle~backend~support: XeTeX}
25 </xdvipdfmx>
```

1.1 Crossreferences

This uses the temporary l3ref-tmp.sty. It will be replaced by kernel code later. It is only need to get a reference for the absolute page counter. This uses the counter from the new lthooks/ltshipout package.

```
26 <@@=pdf>
27 <*drivers>
```

*E-mail: latex-team@latex-project.org

```

28 \RequirePackage{l3ref-tmp}
29 \cs_generate_variant:Nn \ref_label:nn {en}
30 \cs_generate_variant:Nn \ref_value:nn {en}
31 \cs_new_protected:Npn \__pdf_backend_ref_label:nn #1 #2
32 {
33   \@bsphack
34   \ref_label:nn{#1}{abspage}
35   \@esphack
36 }
37 \cs_new:Npn \__pdf_backend_ref_value:nn #1 #2
38 {
39   \ref_value:nn{#1}{#2}
40 }
41 \cs_generate_variant:Nn \__pdf_backend_ref_label:nn {en}
42 \cs_generate_variant:Nn \__pdf_backend_ref_value:nn {en}
43 </drivers>

```

avoid that destinations names are optimized with xelatex/dvipdfmx see <https://tug.org/pipermail/dvipdf/201905/000002.html>

```

44 <*dvipdfmx | xdvipdfmx>
45   \__kernel_backend_literal:x { dvipdfmx:config~C~ 0x0010 }
46 </dvipdfmx | xdvipdfmx>

```

```

\g__pdf_tmpa_prop
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box

```

Some scratch variables

```

47 <*drivers>
48 \prop_new:N \g__pdf_tmpa_prop
49 \tl_new:N \l__pdf_tmpa_tl
50 \box_new:N \l__pdf_backend_tmpa_box
51 \box_new:N \l__pdf_backend_tmpe_box
52 </drivers>

```

(End definition for `\g__pdf_tmpa_prop`, `\l__pdf_tmpa_tl`, and `\l__pdf_backend_tmpe_box`.)

```

\g__pdf_backend_resourceid_int
\g__pdf_backend_name_int
\g__pdf_backend_page_int

```

a counter to create labels for the resources, a counter to number properties in bdc marks, a counter for the `\pdfpageref` implementation.

```

53 <*drivers>
54 \int_new:N \g__pdf_backend_resourceid_int
55 \int_new:N \g__pdf_backend_name_int
56 \int_new:N \g__pdf_backend_page_int
57 </drivers>

```

(End definition for `\g__pdf_backend_resourceid_int`, `\g__pdf_backend_name_int`, and `\g__pdf_backend_page_int`.)

1.2 luacode

Load the lua code.

```

58 <*luatex>
59   \directlua { require("l3backend-testphase.lua") }
60 </luatex>

```

1.3 Converting unicode strings to a pdfname

dvips needs a special function here, so we add this as backend function.

```
61 <*pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
62 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
63 {
64   / \str_convert_pdfname:e { \text_expand:n { #1 } }
65 }
66 </pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
67 <*dvips>
68 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
69 {
70   ~ ( \text_expand:n { #1 } ) ~ cvn
71 }
72 </dvips>
```

1.4 Hooks

1.4.1 Add the “end run” hooks

Here we add the end run hook to suitable end hooks.

```
73 <*pdftex | luatex>
74 % put in \@kernel@after@enddocument@afterlastpage
75 \tl_gput_right:Nn \@kernel@after@enddocument@afterlastpage
76 {
77   \g__kernel_pdfmanagement_end_run_code_tl
78 }
79 </pdftex | luatex>
80 <*dvipdfmx | xdvipdfmx>
81 % put in \@kernel@after@shipout@lastpage
82 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
83 {
84   \g__kernel_pdfmanagement_end_run_code_tl
85 }
86 </dvipdfmx | xdvipdfmx>
87 <*dvips>
88 % put in \@kernel@after@shipout@lastpage
89 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
90 {
91   \g__kernel_pdfmanagement_end_run_code_tl
92 }
93 </dvips>
```

1.4.2 Add the “shipout” hooks

Now we add to the shipout hooks the relevant token lists. We also push the page resources in shipout/firstpage (AtBeginDvi) as the backend code sets color stack there. The xetex driver needs a rule here. If it clashes on the first page, we will need a test ...

```
94 <*drivers>
95 \tl_if_exist:NTF \@kernel@after@shipout@background
96 {
97   \g@addto@macro \@kernel@before@shipout@background{\relax}
98   \g@addto@macro \@kernel@after@shipout@background
```

```

99      {
100        \g__kernel_pdfmanagement_thispage_shipout_code_tl
101      }
102    }
103    {
104      \hook_gput_code:nnn{shipout/background}{pdf}
105      {
106        \g__kernel_pdfmanagement_thispage_shipout_code_tl
107      }
108    }
109  }
110 </drivers>

```

1.5 The /Pages dictionary (pdfpagesattr)

`__pdf_backend_Pages_primitive:n` This is the primitive command to add something to the /Pages dictionary. It works differently for the backends: pdfTeX and LuaTeX overwrite existing content, dvips and dvipdfmx are additive. LuaTeX sets it in lua. The higher level code has to take this into account.

```

111 <*pdfTeX>
112 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
113   {
114     \tex_global:D \tex_pdfpagesattr:D { #1 }
115   }
116 </pdfTeX>
117 <*LuaTeX>
118 %luaTeX: does it in lua
119 \sys_if_engine_luaTeX:T
120   {
121     \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
122       {
123         \tex_directlua:D
124           {
125             pdf.setpagesattributes( \__pdf_backend_luastring:n { #1 } )
126           }
127       }
128   }
129 </luaTeX>
130 <*dvips>
131 \cs_new_protected:Npx \__pdf_backend_Pages_primitive:n #1
132   {
133     \tex_special:D{ps:~[#1~/PAGES~pdfmark} %]
134   }
135 </dvips>
136 <*dvipdfmx | xdvipdfmx>
137 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
138   {
139     \__pdf_backend:n{put~@pages~<<#1>>}
140   }
141 </dvipdfmx | xdvipdfmx>
142 <*dvisvgm>
143 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
144   {}

```

145 `</dvisvgm>`

(End definition for `_pdf_backend_Pages_primitive:n`.)

1.6 “Page” and “ThisPage” attributes (pdfpageattr)

```

\__pdf_backend_Page_primitive:n \__pdf_backend_Page_primitive:n is the primitive command to add something to the
\__pdf_backend_Page_gput:nn /Page dictionary. It works differently for the backends: pdftex and luatex overwrite
\__pdf_backend_Page_gremove:n existing content, dvips and dvipdfmx are additive. luatex sets it in lua. The higher
\__pdf_backend_ThisPage_gput:nn level code has to take this into account. \__pdf_backend_Page_gput:nn stores default
\__pdf_backend_ThisPage_gpush:n values. \__pdf_backend_Page_gremove:n allows to remove a value. \__pdf_backend_
ThisPage_gput:nn adds a value to the current page. \__pdf_backend_ThisPage_
gpsh:n merges the default and the current page values and add them to the dictionary
of the current page in \g__pdf_backend_thispage_shipout_tl.

146 % backend commands
147 <*pdftex>
148 %the primitive
149 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
150 {
151     \tex_global:D \tex_pdfpageattr:D { #1 }
152 }
153 % the command to store default values.
154 % Uses a prop with pdflatex + dvi,
155 % sets a lua table with luatex
156 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2 %key,value
157 {
158     \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
159 }
160 % the command to remove a default value.
161 % Uses a prop with pdflatex + dvi,
162 % changes a lua table with luatex
163 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
164 {
165     \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
166 }
167 % the command used in the document.
168 % direct call of the primitive special with dvips/dvipdfmx
169 % \lattelua: fill a page related table with luatex, merge it with the page
170 % table and push it directly
171 % write to aux and store in prop with pdflatex
172 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
173 {
174     %we need to know the page the resource should be added too.
175     \int_gincr:N\g__pdf_backend_resourceid_int
176     %\zref@labelbylist {l3pdf\int_use:N\g__pdf_backend_resourceid_int} {l3pdf}
177     %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
178     \__pdf_backend_ref_label:en { l3pdf\int_use:N\g__pdf_backend_resourceid_int }{abspage}
179     \tl_set:Nx \l__pdf_tmpa_tl
180     {
181         %\zref@extractdefault
182         {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
183         {pdf@abspage}
184         {0}

```

```

185 % \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
186 \__pdf_backend_ref_value:en {l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
187 }
188 \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
189 {
190 \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
191 }
192 %backend_Page has no handler.
193 \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }
194 }
195 %the code to push the values, used in shipout
196 %merges the two props and then fills the register in pdflatex
197 %merges the two tables and then fills (in lua) in luatex
198 %issues the values stored in the global prop with dvi
199 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
200 {
201 \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdfdict_name:n { g__pdf_Core/Page } }
202 \prop_if_exist:cT { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
203 {
204 \prop_map_inline:cn { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
205 {
206 \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
207 }
208 }
209 \exp_args:Nx \__pdf_backend_Page_primitive:n
210 {
211 \prop_map_function:NN \g__pdf_tmpa_prop \pdfdict_item:ne
212 }
213 }
214 </pdfTeX>
215 <*luatex>
216 % do we need to use some escaping for the values????
217 \cs_new:Npn \__pdf_backend_luastring:n #1
218 {
219 "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
220 }
221 %not used, only there for consistency
222 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
223 {
224 \tex_latelua:D
225 {
226 pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
227 }
228 }
229 % the command to store default values.
230 % Uses a prop with pdflatex + dvi,
231 % sets a lua table with luatex
232 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
233 {
234 \tex_directlua:D
235 {
236 ltx.__pdf.backend_Page_gput
237 (
238 \__pdf_backend_luastring:n { #1 },

```

```

239         \__pdf_backend_luastring:n { #2 }
240     )
241 }
242 }
243 % the command to remove a default value.
244 % Uses a prop with pdflatex + dvi,
245 % changes a lua table with luatex
246 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
247 {
248     \tex_directlua:D
249     {
250         ltx.__pdf.backend_Page_gremove (\__pdf_backend_luastring:n { #1 })
251     }
252 }
253 % the command used in the document.
254 % direct call of the primitive special with dvips/dvipdfmx
255 % \latelua: fill a page related table with luatex, merge it with the page
256 % table and push it directly
257 % write to aux and store in prop with pdflatex
258 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
259 {
260     \tex_latelua:D
261     {
262         ltx.__pdf.backend_ThisPage_gput
263         (
264             tex.count["g_shipout_readonly_int"],
265             \__pdf_backend_luastring:n { #1 },
266             \__pdf_backend_luastring:n { #2 }
267         )
268         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
269     }
270 }
271 %the code to push the values, used in shipout
272 %merges the two props and then fills the register in pdflatex
273 %merges the two tables (the one is probably still empty) and then fills (in lua) in luatex
274 %issues the values stored in the global prop with dvi
275 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
276 {
277     \tex_latelua:D
278     {
279         ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
280     }
281 }
282
283 </luatex>
284 <*dvipdfmx | xdvipdfmx>
285 %the primitive
286 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
287 {
288     \tex_special:D{pdf:-put~@thispage-<<#1>>}
289 }
290 % the command to store default values.
291 % Uses a prop with pdflatex + dvi,
292 % sets a lua table with luatex

```

```

293 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
294 {
295   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
296 }
297 % the command to remove a default value.
298 % Uses a prop with pdflatex + dvi,
299 % changes a lua table with luatex
300 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
301 {
302   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
303 }
304 % the command used in the document.
305 % direct call of the primitive special with dvips/dvipdfmx
306 % \latelua: fill a page related table with luatex, merge it with the page
307 % table and push it directly
308 % write to aux and store in prop with pdflatex
309 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
310 {
311   \__pdf_backend_Page_primitive:n { /#1~#2 }
312 }
313 %the code to push the values, used in shipout
314 %merges the two props and then fills the register in pdflatex
315 %merges the two tables (the one is probably still empty)
316 % and then fills (in lua) in luatex
317 %issues the values stored in the global prop with dvi
318 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
319 {
320   \exp_args:Nx \__pdf_backend_Page_primitive:n
321     { \pdfdict_use:n { g__pdf_Core/Page} }
322 }
323 </dvipdfmx | xdvipdfmx>
324 <*dvips>
325 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
326 {
327   \tex_special:D{ps:-[ThisPage]<<#1>>~/PUT~pdfmark} %]
328 }
329 % the command to store default values.
330 % Uses a prop with pdflatex + dvi,
331 % sets a lua table with luatex
332 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
333 {
334   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
335 }
336 % the command to remove a default value.
337 % Uses a prop with pdflatex + dvi,
338 % changes a lua table with luatex
339 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
340 {
341   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
342 }
343 % the command used in the document.
344 % direct call of the primitive special with dvips/dvipdfmx
345 % \latelua: fill a page related table with luatex, merge it with the page
346 % table and push it directly

```



```

347 % write to aux and store in prop with pdflatex
348 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
349 {
350   \__pdf_backend_Page_primitive:n { /#1~#2 }
351 }
352 %the code to push the values, used in shipout
353 %merges the two props and then fills the register in pdflatex
354 %merges the two tables (the one is probably still empty)
355 %and then fills (in lua) in luatex
356 %issues the values stored in the global prop with dvi
357 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
358 {
359   \exp_args:Nx \__pdf_backend_Page_primitive:n
360     { \pdfdict_use:n { g__pdf_Core/Page} }
361 }
362 </dvips>
363 <*dvisvgm>
364 % mostly only dummies ...
365 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
366 {}
367 % Uses a prop with pdflatex + dvi,
368 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
369 {
370   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
371 }
372 % the command to remove a default value.
373 % Uses a prop with pdflatex + dvi,
374 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
375 {
376   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
377 }
378 % the command used in the document.
379 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
380 {}
381 %the code to push the values, used in shipout
382 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
383 {}
384 </dvisvgm>

```

(End definition for __pdf_backend_Page_primitive:n and others.)

1.7 “Page/Resources”: ExtGState, ColorSpace, Shading, Pattern

Path: Page/Resources/ExtGState etc. The actual output of the resources is handled together with the bdc/Properties. Here is only special code.

\c__pdf_backend_PageResources_clist The names are quite often needed a similar list is now in l3pdfmanagement. Perhaps it should be merged.

```

385 <*drivers>
386 \clist_const:Nn \c__pdf_backend_PageResources_clist
387 {
388   ExtGState,
389   ColorSpace,

```

```

390     Pattern,
391     Shading,
392   }
393 </drivers>

```

(End definition for \c__pdf_backend_PageResources_clist.)

Now the backend commands the command to fill the register and to push the values.

__pdf_backend_PageResources_gput:nnn stores values for the page resources.

#1 : name of the resource (ExtGState, ColorSpace, Shading, Pattern)

#2 : a pdf name without slash

#3 : value

This pushes out the objects. It should be a no-op with xdvipdfmx and dvips as it currently issued in the end-of-run hook! create the backend objects:

__pdf_backend_PageResources_obj_gpush:

```

394 <*pdfTeX | luatex>
395 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
396 {
397   \__pdf_backend_object_new:nn {__pdf/Page/Resources/#1} {dict}
398   \cs_if_exist:NT \tex_directlua:D
399     {
400       \tex_directlua:D
401       {
402         ltx.__pdf.object["__pdf/Page/Resources/#1"]
403         =
404         "\__pdf_backend_object_ref:n{__pdf/Page/Resources/#1}"
405       }
406     }
407 }
408 </pdfTeX | luatex>

```

values are only stored in a prop and will be output at end document. luatex must also trigger the lua side

```

409 <*luatex>
410 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
411 {
412   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
413   \tex_latelua:D{ltx.__pdf.Page.Resources.#1=true}
414   \tex_latelua:D
415   {
416     ltx.pdf.Page.Resources_gpush(tex.count["g_shipout_readonly_int"])
417   }
418 }
419 </luatex>
420 <*pdfTeX>
421 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
422 {
423   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
424 }
425 </pdfTeX>

```

code for end of document code

```

426 <*pdfTeX | luatex>
427 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
428 {

```

```

429 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
430 {
431   \prop_if_empty:cF
432   { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1 } }
433   {
434     \__pdf_backend_object_write:nx
435     { __pdf/Page/Resources/##1 }
436     { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1 } }
437   }
438 }
439 }
440 </pdfTeX | luatex>

```

xdvipdfmx doesn't work correctly with object names ... <https://tug.org/pipermail/dvipdfmx/2019-August/000021.html>, so we use this must be issued on every page! objects should not only be created but also initialized initialization should be done before anyone tries to write so we add rules for the backend. The push command should not be used as it is in the wrong end document hook. If needed a new command must be added.

```

441 <*dvipdfmx | xdvipdfmx>
442 <xdvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
443 <dvipdfmx> \hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
444 %
445 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
446 {
447   \__pdf_backend_object_new:nn { __pdf/Page/Resources/#1 } { dict }
448   \hook_gput_code:nnn{shipout/firstpage}{pdf}{\__pdf_backend_object_write:nn { __pdf/Page/
449 }
450 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
451 {
452   \__pdf_backend:n {put~@resources~<<#1>>}
453 }
454 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
455 {
456   % this is not used for output, but there is a test if the resource is empty
457   \exp_args:Nnx
458   \prop_gput:cnn { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1 } }
459   { \str_convert_pdfname:n {#2} }{ #3 }
460   %objects are not filled with \pdf_object_write as this is not additive!
461   \__pdf_backend:x
462   {
463     put~\__pdf_backend_object_ref:n {__pdf/Page/Resources/#1}<<#2~#3>>
464   }
465 }
466
467 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
468 </dvipdfmx | xdvipdfmx>

```

dvips unneeded, or no-op. The push command should not be used as it is in the wrong end document hook. If needed a new command must be added.

```

469 <*dvips>
470 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
471 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
472 { %only for the show command TEST!!
473   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }

```

```

474 }
475 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
476 </dvips>

```

dvipsvgm unneeded, or no-op

```

477 <*dvipsvgm>
478 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
479 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
480 { %only for the show command TEST!!
481   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
482 }
483 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
484 </dvipsvgm>

```

(End definition for __pdf_backend_PageResources_gput:nnn and __pdf_backend_PageResources_obj_gpush:.)

1.7.1 Page resources /Properties + BDC operators

```

\__pdf_backend_bdc:nn \__pdf_backend_bdcobject:nn, \__pdf_backend_bdcobject:n,
\__pdf_backend_bdcobject:nn \__pdf_backend_bmc:n and \__pdf_backend_emc: are the backend command that cre-
\__pdf_backend_bdcobject:n ate the bdc/emc marker and store the properties. \__pdf_backend_PageResources_-
\__pdf_backend_bmc:n gpush:n outputs the /Properties and/or the other resources for the current page.
\__pdf_backend_emc:
\__pdf_backend_PageResources_gpush:n
485 % pdftex and luatex (and perhaps dvips ...) need to know if there are in a
486 % xform stream ...
487 <*drivers>
488 \bool_new:N \l__pdf_backend_xform_bool
489 </drivers>
490 <*dvips>
491 % dvips is easy: create an object, and reference it in the bdc
492 % ghostscript will then automatically replace it by a name
493 % and add the name to the /Properties dict
494 % special variant von accsupp
495 % https://chat.stackexchange.com/transcript/message/50831812#50831812
496 %
497 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
498 {
499   \__pdf_backend_pdfmark:x{/#1~<<#2>>~/BDC}
500 }
501 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
502 {
503   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_ref:n{#2}~/BDC}
504 }
505 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
506 {
507   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_last:~/BDC}
508 }
509 \cs_set_protected:Npn \__pdf_backend_emc:
510 {
511   \__pdf_backend_pdfmark:n{/EMC} %
512 }
513 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
514 {
515   \__pdf_backend_pdfmark:n{/#1~/BMC} %

```

```

516 }
517 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
518
519 </dvips>
520 <*dvisvgm>
521 % dvisvgm should do nothing
522 %
523 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
524 {}
525 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
526 {}
527 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
528 {}
529 \cs_set_protected:Npn \__pdf_backend_emc:
530 {}
531 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
532 {}
533 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
534
535 </dvisvgm>
536
537 % xetex has to create the entries in the /Properties manually
538 % (like the other backends)
539 % use pdfbase special
540 % https://chat.stackexchange.com/transcript/message/50832016#50832016
541 % the property is added to xform resources automatically,
542 % no need to worry about it.
543 <*dvipdfmx | xdvipdfmx>
544 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
545 {
546   \int_gincr:N \g__pdf_backend_name_int
547   \__kernel_backend_literal:x
548   {
549     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
550   }
551   \__kernel_backend_literal:x
552   {
553     pdf:put~@resources~
554     <<
555       /Properties~
556       <<
557         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
558         \__pdf_backend_object_ref:n { #2 }
559       >>
560     >>
561   }
562 }
563 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
564 {
565   \int_gincr:N \g__pdf_backend_name_int
566   \__kernel_backend_literal:x
567   {
568     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
569   }

```

```

570     \_kernel_backend_literal:x
571     {
572         pdf:put~@resources~
573         <<
574             /Properties~
575             <<
576                 /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
577                 \_pdf_backend_object_last:
578             >>
579         >>
580     }
581 }
582 \cs_set_protected:Npn \_pdf_backend_bmc:n #1
583 {
584     \_kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
585 }
586
587 %this require management
588 \cs_set_protected:Npn \_pdf_backend_bdc_contobj:nn #1 #2
589 {
590     \pdf_object_unnamed_write:nn { dict }{ #2 }
591     \_pdf_backend_bdcobject:n { #1 }
592 }
593
594 \cs_set_protected:Npn \_pdf_backend_bdc_contstream:nn #1 #2
595 {
596     \_kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
597 }
598
599 \cs_set_protected:Npn \_pdf_backend_bdc:nn #1 #2
600 {
601     \bool_if:NTF \g__pdfmanagement_active_bool
602     {\cs_gset_eq:NN \_pdf_backend_bdc:nn \_pdf_backend_bdc_contobj:nn}
603     {\cs_gset_eq:NN \_pdf_backend_bdc:nn \_pdf_backend_bdc_contstream:nn}
604     \_pdf_backend_bdc:nn {#1}{#2}
605 }
606 \cs_set_protected:Npn \_pdf_backend_emc:
607 {
608     \_kernel_backend_literal:n {pdf:code~EMC} %pdfbase
609 }
610 % properties are handled automatically, but the other resources should be added
611 % at shipout
612 \cs_new_protected:Npn \_pdf_backend_PageResources_gpush:n #1
613 {
614     \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
615     {
616         \prop_if_empty:cF { \_kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
617         {
618             \_kernel_backend_literal:x
619             {
620                 pdf:put~@resources~
621                 <</##1~\_pdf_backend_object_ref:n {\_pdf/Page/Resources/##1}>>
622             }
623         }
624     }

```

```

624     }
625   }
626   </dvipdfmx | xdvipdfmx>
627   % luatex + pdftex
628   <*luatex>
629   \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
630   {
631     \int_gincr:N \g__pdf_backend_name_int
632     \exp_args:Nx\__kernel_backend_literal_page:n
633     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
634     \bool_if:NTF \l__pdf_backend_xform_bool
635     {
636       \exp_args:Nnx\pdfdict_gput:nnn
637       { g__pdf_Core/Xform/Resources/Properties }
638       { l3pdf\int_use:N\g__pdf_backend_name_int }
639       { \__pdf_backend_object_ref:n { #2 } }
640     }
641     {
642       \exp_args:Nx \tex_latelua:D
643       {
644         ltx.pdf.Page_Resources_Properties_gput
645         (
646           tex.count["g_shipout_readonly_int"],
647           "l3pdf\int_use:N\g__pdf_backend_name_int",
648           "\__pdf_backend_object_ref:n { #2 }"
649         )
650       }
651     }
652   }
653   \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
654   {
655     \int_gincr:N \g__pdf_backend_name_int
656     \exp_args:Nx\__kernel_backend_literal_page:n
657     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
658     \bool_if:NTF \l__pdf_backend_xform_bool
659     {
660       \exp_args:Nnx\pdfdict_gput:nnn %no handler needed
661       { g__pdf_Core/Xform/Resources/Properties }
662       { l3pdf\int_use:N\g__pdf_backend_name_int }
663       { \__pdf_backend_object_last: }
664     }
665     {
666       \exp_args:Nx \tex_latelua:D
667       {
668         ltx.pdf.Page_Resources_Properties_gput
669         (
670           tex.count["g_shipout_readonly_int"],
671           "l3pdf\int_use:N\g__pdf_backend_name_int",
672           "\__pdf_backend_object_last:"
673         )
674       }
675     }
676   }
677   \cs_set_protected:Npn \__pdf_backend_bmc:n #1

```

```

678 {
679   \_kernel_backend_literal_page:n { /#1~BMC }
680 }
681 \cs_set_protected:Npn \_pdf_backend_bdc_contobj:nn #1 #2
682 {
683   \pdf_object_unnamed_write:nn { dict } { #2 }
684   \_pdf_backend_bdcobject:n { #1 }
685 }
686 \cs_set_protected:Npn \_pdf_backend_bdc_contstream:nn #1 #2
687 {
688   \_kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
689 }
690 \cs_set_protected:Npn \_pdf_backend_bdc:nn #1 #2
691 {
692   \bool_if:NTF \g_pdfmanagement_active_bool
693     {\cs_gset_eq:NN \_pdf_backend_bdc:nn \_pdf_backend_bdc_contobj:nn}
694     {\cs_gset_eq:NN \_pdf_backend_bdc:nn \_pdf_backend_bdc_contstream:nn}
695     \_pdf_backend_bdc:nn {#1}{#2}
696 }
697 \cs_set_protected:Npn \_pdf_backend_emc:
698 {
699   \_kernel_backend_literal_page:n { EMC }
700 }
701
702 \cs_new_protected:Npn \_pdf_backend_PageResources_gpush:n #1 {}
703 </luatex>
704 <*pdfTeX>
705 % pdfLaTeX is the most complicated as it has to go through the aux ...
706 % the push command is extended to take other resources too
707 \cs_set_protected:Npn \_pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
708 {
709   \int_gincr:N \g_pdf_backend_name_int
710   \exp_args:Nx\_kernel_backend_literal_page:n
711     { /#1 ~ /l3pdf\int_use:N\g_pdf_backend_name_int\c_space_tl BDC }
712   % code to set the property ....
713   \int_gincr:N \g_pdf_backend_resourceid_int
714   \bool_if:NTF \l_pdf_backend_xform_bool
715   {
716     \exp_args:Nnxx\pdfdict_gput:nnn %no handler needed
717       { g_pdf_Core/Xform/Resources/Properties }
718       { l3pdf\int_use:N\g_pdf_backend_resourceid_int }
719       { \_pdf_backend_object_ref:n { #2 } }
720   }
721   {
722     %\zref@labelbylist
723     { l3pdf\int_use:N\g_pdf_backend_resourceid_int }
724     { l3pdf }
725     % \ref_label:en{l3pdf\int_use:N\g_pdf_backend_resourceid_int}{abspage}
726     \_pdf_backend_ref_label:en{l3pdf\int_use:N\g_pdf_backend_resourceid_int}{abspage}
727     \tl_set:Nx \l_pdf_tmpa_tl
728     {
729       %\zref@extractdefault
730       { l3pdf\int_use:N\g_pdf_backend_resourceid_int }
731       % {pdf@abspage}

```



```

732 %           {0}
733           %\ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
734           \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
735       }
736       \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
737       {
738           \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
739       }
740       \exp_args:Nnxx\pdfdict_gput:nnn
741       { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
742       { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
743       { \__pdf_backend_object_ref:n{#2} }
744   }
745 }
746 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
747 {
748     \int_gincr:N \g__pdf_backend_name_int
749     \exp_args:Nx\__kernel_backend_literal_page:n
750     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
751     % code to set the property ....
752     \int_gincr:N\g__pdf_backend_resourceid_int
753     \bool_if:NTF \l__pdf_backend_xform_bool
754     {
755         \exp_args:Nnxx\pdfdict_gput:nnn
756         { g__pdf_Core/Xform/Resources/Properties }
757         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
758         { \__pdf_backend_object_last: }
759     }
760     {
761         %\zref@labelbylist
762         % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
763         % { l3pdf }
764         %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
765         \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
766         \tl_set:Nx \l__pdf_tmpa_tl
767         {
768             %\zref@extractdefault
769             % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
770             % {pdf@abspage}
771             % {0}
772             % \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
773             \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
774         }
775         \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
776         {
777             \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
778         }
779         \exp_args:Nnxx\pdfdict_gput:nnn
780         { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
781         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
782         { \__pdf_backend_object_last: }
783         %\pdfdict_show:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
784     }
785 }

```

```

786 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
787 {
788   \__kernel_backend_literal_page:n { /#1~BMC }
789 }
790 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
791 {
792   \pdf_object_unnamed_write:nn { dict } { #2 }
793   \__pdf_backend_bdcobject:n { #1 }
794 }
795 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
796 {
797   \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
798 }
799 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
800 {
801   \bool_if:NTF \g__pdfmanagement_active_bool
802     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
803     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
804     \__pdf_backend_bdc:nn {#1}{#2}
805 }
806 \cs_set_protected:Npn \__pdf_backend_emc:
807 {
808   \__kernel_backend_literal_page:n { EMC }
809 }
810
811 \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
812 {
813   \prop_if_empty:cF
814     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
815     {
816       \pdfdict_item:ne { #1 }{\pdf_object_ref:n {\__pdf/Page/Resources/#1}}
817     }
818 }
819
820 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
821 {
822   \exp_args:NNx \tex_global:D \tex_pdfpageresources:D
823   {
824     \prop_if_exist:cT
825       { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
826       {
827         /Properties~
828         <<
829         \prop_map_function:cN
830           { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
831           \pdfdict_item:ne
832         >>
833       }
834     %% add ExtGState etc
835     \clist_map_function:NN
836       \c__pdf_backend_PageResources_clist
837       \__pdf_backend_PageResources_gpush_aux:n
838   }
839 }

```

```

840
841 </pdftex>

```

(End definition for `_pdf_backend_bdc:nn` and others.)

1.8 “Catalog” & subdirectories (pdfcatalog)

The backend command is already in the driver: `_pdf_backend_catalog_gput:nn`

1.8.1 Special case: the `/Names/EmbeddedFiles` dictionary

Entries to `/Names` are handled differently, in part (`/Desc`) it is automatic, for other special commands like `\pdfnames` must be used. For `EmbeddedFiles` dvips wants code for every file and then creates the Name tree automatically. Other name trees are ignored. TODO: Currently the code for `EmbeddedFiles` is still a bit different but this should be merged, all name trees should be handled with the same code.

```

842 % pdflatex
843 <*pdftex>
844 \cs_new_protected:Npn \_pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
845 {
846     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
847     \tex_pdfnames:D {/#1~\pdf_object_ref_last:}
848 }
849 </pdftex>
850 <*luatex>
851 \cs_new_protected:Npn \_pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
852 {
853     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
854     \tex_pdfextension:D~names~ {/#1~\pdf_object_ref_last:}
855 }
856 </luatex>
857 <*dvipdfmx | xdvipdfmx>
858 \cs_new_protected:Npn \_pdf_backend_Names_gpush:nn #1 #2 %#1 name of name tree, #2 array co
859 {
860     \pdf_object_unnamed_write:nn {dict} {/Names [#2] }
861     \_pdf_backend:x {put~@names~<</#1~\pdf_object_ref_last: >>}
862 }
863 </dvipdfmx | xdvipdfmx>
864
865 %dvips: noop
866 <*dvips>
867 \cs_new_protected:Npn \_pdf_backend_Names_gpush:nn #1 #2 {}
868 </dvips>
869 %dvisvgm: noop
870 <*dvisvgm>
871 \cs_new_protected:Npn \_pdf_backend_Names_gpush:nn #1 #2 {}
872 </dvisvgm>

```

`EmbeddedFiles` is a bit special. For once we need backend commands for dvips. But we want also an option to create the name on the fly.

`_pdf_backend_NamesEmbeddedFiles_add:nn` dvips need special backend code to create the name tree. With the other engines it does nothing.

```

873 <*pdftex | luatex | dvipdfmx | xdvipdfmx>

```

```

874 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2 {}
875 </pdftex | luatex | dvipdfmx | xdvipdfmx>
876 <*dvips>
877 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
878 {
879     \__pdf_backend_pdfmark:x
880     {
881         /Name~#1~
882         /FS~#2~
883         /EMBED
884     }
885 }
886 </dvips>
887 <*dvisvgm>
888 %no op. Or is there any sensible use for it?
889 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:nn #1 #2
890 {}
891
892 </dvisvgm>

```

(End definition for __pdf_backend_NamesEmbeddedFiles_add:nn.)

1.8.2 Additional annotation commands

Starting with texlive 2021 pdftex and luatex offer commands to interrupt a link. That can for example be used to exclude the header and footer from the link. We add here backend support for this.

```

893 <*drivers>
894 \cs_new_protected:Npn \__pdf_backend_link_off:{}
895 \cs_new_protected:Npn \__pdf_backend_link_on: {}
896 </drivers>
897 <*pdftex>
898 \cs_if_exist:NT \pdfrunninglinkoff
899 {
900     \cs_set_protected:Npn \__pdf_backend_link_off:
901     {
902         \pdfrunninglinkoff
903     }
904     \cs_set_protected:Npn \__pdf_backend_link_on:
905     {
906         \pdfrunninglinkon
907     }
908 }
909 </pdftex>
910 <*luatex>
911 \int_compare:nNnT {\tex_luatexversion:D } > {112}
912 {
913     \cs_set_protected:Npn \__pdf_backend_link_off:
914     {
915         \pdfextension linkstate 1
916     }
917     \cs_set_protected:Npn \__pdf_backend_link_on:
918     {
919         \pdfextension linkstate 0

```

```

920     }
921   }
922   </luatex>
923   <*dvipdfmx | xdvipdfmx>
924   \cs_set_protected:Npn \__pdf_backend_link_off:
925   {
926     \__pdf_backend:n { noline }
927   }
928   \cs_set_protected:Npn \__pdf_backend_link_on:
929   {
930     \__pdf_backend:n { link }
931   }
932   </dvipdfmx | xdvipdfmx>

```

1.8.3 FormXObject / backend

```

\__pdf_backend_xform_new:nnnn #1 : name
                              #2 : attributes
                              #3 : resources needed?? or are all resources autogenerated?
                              #4 : content, this doesn't need to be a box!

```

```

\__pdf_backend_xform_use:n 933 <*pdfTeX>
\__pdf_backend_xform_ref:n 934 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
                              935 % #1 name
                              936 % #2 attributes
                              937 % #3 resources
                              938 % #4 content, not necessarily a box!
                              939 {
                              940   \hbox_set:Nn \l__pdf_backend_tmpa_box
                              941   {
                              942     \bool_set_true:N \l__pdf_backend_xform_bool
                              943     \prop_gclear:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
                              944     #4
                              945   }
                              946   %store the dimensions
                              947   \tl_const:cx
                              948   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
                              949   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
                              950   \tl_const:cx
                              951   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
                              952   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
                              953   \tl_const:cx
                              954   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
                              955   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
                              956   %% do we need to test if #2 and #3 are empty??
                              957   \tex_immediate:D \tex_pdfxform:D
                              958   ~ attr ~ { #2 }
                              959   %% which other resources should be default? Is an argument actually needed?
                              960   ~ resources ~
                              961   {
                              962     #3
                              963     \int_compare:nNnT
                              964     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }

```

```

965         >
966         { 0 }
967         {
968             /Properties~
969             <<
970             \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
971             >>
972         }
973
974     \prop_if_empty:cF
975     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
976     {
977         /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
978     }
979     \prop_if_empty:cF
980     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
981     {
982         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
983     }
984     \prop_if_empty:cF
985     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Shading } }
986     {
987         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
988     }
989     \prop_if_empty:cF
990     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
991     {
992         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
993     }
994 }
995 \l__pdf_backend_tmpa_box
996 \int_const:cn
997 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
998 { \tex_pdflastxform:D }
999 }
1000
1001 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1002 {
1003     \tex_pdfrefxform:D
1004     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1005     \scan_stop:
1006 }
1007
1008 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1009 {
1010     \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1011 }
1012 </pdfTeX>
1013 <*luatex>
1014 %luatex
1015 %nearly identical but not completely ...
1016 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1017 % #1 name
1018 % #2 attributes

```

```

1019 % #3 resources
1020 % #4 content, not necessarily a box!
1021 {
1022   \hbox_set:Nn \l__pdf_backend_tmpa_box
1023   {
1024     \bool_set_true:N \l__pdf_backend_xform_bool
1025     \prop_gclear:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1026     #4
1027   }
1028   \tl_const:cx
1029   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1030   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1031   \tl_const:cx
1032   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1033   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1034   \tl_const:cx
1035   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1036   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1037   %% do we need to test if #2 and #3 are empty??
1038   \tex_immediate:D \tex_pdfxform:D
1039   ~ attr ~ { #2 }
1040   %% which resources should be default? Is an argument actually needed?
1041   ~ resources ~
1042   {
1043     #3
1044     \int_compare:nNnT
1045       {\prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties
1046       >
1047       { 0 }
1048       {
1049         /Properties~
1050         <<
1051         \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1052         >>
1053       }
1054       \prop_if_empty:cF
1055       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1056       {
1057         /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1058       }
1059       \prop_if_empty:cF
1060       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1061       {
1062         /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1063       }
1064       \prop_if_empty:cF
1065       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1066       {
1067         /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1068       }
1069       \prop_if_empty:cF
1070       { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1071       {
1072         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }

```

```

1073     }
1074   }
1075   \l__pdf_backend_tmpa_box
1076   \int_const:cn
1077   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1078   { \tex_pdflastxform:D }
1079 }
1080
1081 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1082 {
1083   \tex_pdfrefxform:D \int_use:c
1084   {
1085     c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1086   }
1087   \scan_stop:
1088 }
1089
1090 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1091 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1092
1093 </luatex>
1094 <*dvipdfmx | xdvipdfmx>
1095 % xetex
1096 % it needs a bit testing if it really works to set the box to 0 before the special ...
1097 % does it disturb viewing the xobject?
1098 % what happens with the resources (bdc)? (should work as they are specials too)
1099 % xetex requires that the special is in horizontal mode. This means it affects
1100 % typesetting. But we can no delay the whole form code to shipout
1101 % as the object reference and the size is often wanted on the current page.
1102 % so we need to allocate a box - but probably they won't be thousands xform
1103 % in a document so it shouldn't matter.
1104 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1105 % #1 name
1106 % #2 attributes
1107 % #3 resources
1108 % #4 content, not necessarily a box!
1109 {
1110   \int_gincr:N \g__pdf_backend_object_int
1111   \int_const:cn
1112   { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1113   { \g__pdf_backend_object_int }
1114   \box_new:c { g__pdf_backend_xform_#1_box }
1115   \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1116   {
1117     \bool_set_true:N \l__pdf_backend_xform_bool
1118     #4
1119   }
1120   \tl_const:cx
1121   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1122   { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1123   \tl_const:cx
1124   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1125   { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1126   \tl_const:cx

```



```

1127 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1128 { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1129 \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1130 \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1131 \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1132 \hook_gput_next_code:nn {shipout/background}
1133 {
1134   \mode_leave_vertical: %needed, the xform disappears without it.
1135   \__pdf_backend:x
1136   {
1137     bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1138     \c_space_tl width ~ \pdfxform_wd:n { #1 }
1139     \c_space_tl height ~ \pdfxform_ht:n { #1 }
1140     \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1141   }
1142   \box_use_drop:c { g__pdf_backend_xform_#1_box }
1143   \__pdf_backend:x {put ~ @resources ~<<#3>> }
1144   \__pdf_backend:x
1145   {
1146     put~ @resources ~
1147     <<
1148       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1149     >>
1150   }
1151   \__pdf_backend:x
1152   {
1153     put~ @resources ~
1154     <<
1155       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1156     >>
1157   }
1158   \__pdf_backend:x
1159   {
1160     put~ @resources ~
1161     <<
1162       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1163     >>
1164   }
1165   \__pdf_backend:x
1166   {
1167     put~ @resources ~
1168     <<
1169       /ColorSpace~
1170       \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1171     >>
1172   }
1173   \exp_args:Nx
1174   \__pdf_backend:x {exobj ~<<#2>>}
1175 }
1176 }
1177
1178
1179
1180 \cs_new:Npn \__pdf_backend_xform_ref:n #1

```

```

1181     {
1182     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1183     }
1184
1185 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1186 {
1187     \hbox_set:Nn \l__pdf_backend_tmpa_box
1188     {
1189         \__pdf_backend:x
1190         {
1191             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1192         }
1193     }
1194     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1195     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1196     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1197     \box_use_drop:N \l__pdf_backend_tmpa_box
1198 }
1199 </dvipdfmx | xdvipdfmx>
1200 <*dvisvgm>
1201 % unclear what it should do!!
1202 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1203 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1204 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1205 </dvisvgm>

```

The xform code for dvips is based on code from the attachfile2 package (in atfi-dvips), along with some ideas from pdfbase and has been corrected with the help of Alexander Grahn. Details like clipping and landscape will probably be corrected in the future. We need some temporary variables to store dimensions

```

1206 <*dvips>
1207 \tl_new:N \l__pdf_backend_xform_tmpwd_tl
1208 \tl_new:N \l__pdf_backend_xform_tmpdp_tl
1209 \tl_new:N \l__pdf_backend_xform_tmphl_tl
1210 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1211 {
1212     \int_gincr:N \g__pdf_backend_object_int
1213     \int_const:cn
1214     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1215     { \g__pdf_backend_object_int }
1216
1217     \hbox_set:Nn \l__pdf_backend_tmpa_box
1218     {
1219         \bool_set_true:N \l__pdf_backend_xform_bool
1220         \prop_gc_clear:c {\__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1221         #4
1222     }
1223 %store the dimensions
1224 \tl_const:cx
1225     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1226     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1227 \tl_const:cx
1228     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1229     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }

```

```

1230 \tl_const:cx
1231 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1232 { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1233 %store content dimensions in DPI units (Dots) (code from issue 25)
1234 \tl_set:Nx\l__pdf_backend_xform_tmpwd_tl
1235 {
1236   \dim_to_decimal_in_sp:n{ \box_wd:N \l__pdf_backend_tmpa_box }~
1237   65536~div~72.27~div~DVImag~mul~Resolution~mul~
1238 }
1239 \tl_set:Nx\l__pdf_backend_xform_tmph_t_tl
1240 {
1241   \dim_to_decimal_in_sp:n{ \box_ht:N \l__pdf_backend_tmpa_box }~
1242   65536~div~72.27~div~DVImag~mul~VResolution~mul~
1243 }
1244 \tl_set:Nx\l__pdf_backend_xform_tmppd_tl
1245 {
1246   \dim_to_decimal_in_sp:n{ \box_dp:N \l__pdf_backend_tmpa_box }~
1247   65536~div~72.27~div~DVImag~mul~VResolution~mul~
1248 }
1249 % mirror the box
1250 %\box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1251 \hbox_set:Nn\l__pdf_backend_tmppb_box
1252 {
1253   \__kernel_backend_postscript:x
1254   {
1255     gsave~currentpoint~
1256     initclip~ % restore default clipping path (page device/whole page)
1257     clippath~pathbbox~newpath~pop~pop~
1258     \tl_use:N\l__pdf_backend_xform_tmppd_tl~add~translate~
1259     mark~
1260     /objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1261     /BBox[
1262       0~
1263       \tl_use:N\l__pdf_backend_xform_tmph_t_tl~
1264       \tl_use:N\l__pdf_backend_xform_tmpwd_tl~
1265       \tl_use:N\l__pdf_backend_xform_tmppd_tl~
1266       neg
1267     ]
1268     \str_if_eq:eeF{#1}{}
1269     {
1270       product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1271     }
1272     /BP~pdfmark~1~-1~scale~neg~exch~neg~exch~translate
1273   }
1274   \box_use_drop:N\l__pdf_backend_tmpa_box
1275   \__kernel_backend_postscript:n
1276   {
1277     mark ~ /EP~pdfmark ~ grestore
1278   }
1279   \str_if_eq:eeF{#1}{}
1280   {
1281     \__kernel_backend_postscript:x
1282     {
1283       product~(Ghostscript)~search~

```

```

1284         {
1285             pop~pop~pop~
1286             mark~
1287             { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1288             ~<<#2>>~/PUT~pdfmark
1289         }{pop}ifelse
1290     }
1291 }
1292 }
1293 \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1294 \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1295 \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1296 \hook_gput_code:nnn {begindocument/end}{pdfxform}
1297 {
1298     \mode_leave_vertical:
1299     \box_use:N\l__pdf_backend_tmpb_box
1300 }
1301 }
1302
1303
1304 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1305 {
1306     \hbox_set:Nn \l__pdf_backend_tmpa_box
1307     {
1308         \__kernel_backend_postscript:x
1309         {
1310             gsave~currentpoint~translate~1~-1~scale~
1311             mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }~
1312             /SP~pdfmark ~ grestore
1313         }
1314     }
1315     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1316     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1317     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1318     \box_use_drop:N \l__pdf_backend_tmpa_box
1319 }
1320 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1321 {
1322     { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1323 }
1324
1325 </dvips>
1326 <*drivers>
1327 %% all
1328 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1329 {
1330     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1331     { \prg_return_true: }
1332     { \prg_return_false: }
1333 }
1334 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n\__pdf_backend_xform_if_exist:n
1335 { TF , T , F , p }
1336 </drivers>

```

(End definition for __pdf_backend_xform_new:nnnn, __pdf_backend_xform_use:n, and __pdf_-

`backend_xform_ref:n.)`

1.9 Structure Destinations

Standard destinations consist of a reference to a page in the pdf and instructions how to display it—typically they will put a specific location in the left top corner of the viewer and so give the impression that a link jumped to the word in this place. But in reality they are not connected to the content.

Starting with pdf 2.0 destinations can in a tagged PDF also point to a structure, to a /StructElem object. GoTo links can then additionally to the /D key pointing to a page destination also point to such a structure destination with an /SD key. Programs that e.g. convert such a PDF to html can then create better links. (According to the reference, PDF-viewer should prefer the structure destination over the page destination, but as far as it is known this isn't done yet.)

Currently structure destinations and GoTo links making use of it could natively only be created with the dvipdfmx backend. With pdftex and luatex it was only possible to create a restricted type which used only the “Fit” mode. Starting with T_EXlive 2022 (earlier in miktex) both engine will know new keywords which allow to create structure destination easily.

The following backend code prepares the use of structure destinations. The general idea is that if structure destinations are used, they should be used always. So we define alternative commands which can be activated by mapping them to the standard backend commands.

`\l_pdf_current_structure_destination_tl`

This command holds the name of the structure object to use in the next command which creates a destination. The code which activates structure destinations must also ensure that it has a sensible, expandable content. `tagpdf` for example will define it as

```
\tl_set:Nn \l_pdf_current_structure_destination_tl { __tag/struct/\g_tag_struct_stack
1337 <*drivers>
1338 \tl_new:N \l_pdf_current_structure_destination_tl
1339 </drivers>
```

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

We will define alternatives for three backend commands:

```
\__pdf_backend_destination:nn -> \__pdf_backend_structure_destination:nn
\__pdf_backend_destination:nnnn -> \__pdf_backend_structure_destination:nnnn
\__pdf_backend_link_begin_goto:nnw -> \__pdf_backend_link_begin_structure_goto:nnw
```

Activating means mapping them onto the original commands. Be aware that not all engines and compilation routes support structure destinations, for them the command will be a no-op.

`\pdf_activate_structure_destination:`

```
1340 <*drivers>
1341 \cs_new_protected:Npn \pdf_activate_structure_destination:
1342 {
1343   \cs_gset_eq:NN \__pdf_backend_destination:nn \__pdf_backend_structure_destination:nn
1344   \cs_gset_eq:NN \__pdf_backend_destination:nnnn \__pdf_backend_structure_destination:nnnn
1345   \cs_gset_eq:NN \__pdf_backend_link_begin_goto:nnw \__pdf_backend_link_begin_structure_goto:nnw
1346 }
1347 </drivers>
```

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

Now the driver dependant parts. By default the new commands are simply copies of the original commands. We adapt them then for the engines and engine version which provide support for structure destinations.

```

1348 <*drivers>
1349 \cs_set_eq:NN \__pdf_backend_structure_destination:nn \__pdf_backend_destination:nn
1350 \cs_set_eq:NN \__pdf_backend_structure_destination:nnnn \__pdf_backend_destination:nnnn
1351 \cs_set_eq:NN \__pdf_backend_link_begin_structure_goto:nnw \__pdf_backend_link_begin_goto:nnw
1352 </drivers>

```

`__pdf_backend_structure_destination:nn`

This command is the backend command to create a destination. It should in parallel create also a structure destination. At first xetex/dvipdfmx. The structure destination is an array, so we use obj for it so that we can reference it:

```

1353 <*xdvipdfmx|dvipdfmx>
1354 \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1355 {
1356   \__pdf_backend:x
1357   {
1358     dest ~ ( \exp_not:n {#1} )
1359     [
1360       @thispage
1361       \str_case:nnF {#2}
1362       {
1363         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1364         { fit } { /Fit }
1365         { fitb } { /FitB }
1366         { fitbh } { /FitBH }
1367         { fitbv } { /FitBV ~ @xpos }
1368         { fith } { /FitH ~ @ypos }
1369         { fitv } { /FitV ~ @xpos }
1370         { fitr } { /Fit }
1371       }
1372       { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1373     ]
1374   }

```

We test if the structure object exist. The object of the structure destination gets the name `@pdf.Sdest.<destname>`, where `<destname>` is the name of the standard destination so that we can reference it in the GoTo links.

```

1375 \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1376 {
1377   \__pdf_backend:x
1378   {
1379     obj ~ @pdf.Sdest.\exp_not:n{#1}
1380     [
1381       \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_tl }
1382       \str_case:nnF {#2}
1383       {
1384         { xyz } { /XYZ ~ @xpos ~ @ypos ~ null }
1385         { fit } { /Fit }
1386         { fitb } { /FitB }
1387         { fitbh } { /FitBH }
1388         { fitbv } { /FitBV ~ @xpos }

```

```

1389         { fith } { /FitH ~ @ypos }
1390         { fitv } { /FitV ~ @xpos }
1391         { fitr } { /Fit }
1392     }
1393     { /XYZ ~ @xpos ~ @ypos ~ \fp_eval:n { (#2) / 100 } }
1394 ]
1395 }
1396 }
1397 }

```

The second destination command is for the boxed destination. Here we need to define an new auxiliary command:

```

1398 \cs_new_protected:Npn \__pdf_backend_structure_destination_aux:nnnn #1#2#3#4
1399 {
1400     \vbox_to_zero:n
1401     {
1402         \__kernel_kern:n {#4}
1403         \hbox:n
1404         {
1405             \__pdf_backend:n { obj ~ @pdf_ #2 _llx ~ @xpos }
1406             \__pdf_backend:n { obj ~ @pdf_ #2 _lly ~ @ypos }
1407         }
1408         \tex_vss:D
1409     }
1410     \__kernel_kern:n {#1}
1411     \vbox_to_zero:n
1412     {
1413         \__kernel_kern:n { -#3 }
1414         \hbox:n
1415         {
1416             \__pdf_backend:n
1417             {
1418                 dest ~ (#2)
1419                 [
1420                     @thispage
1421                     /FitR ~
1422                     @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1423                     @xpos ~ @ypos
1424                 ]
1425             }

```

Here we add the structure destination to the same box

```

1426     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1427     {
1428         \__pdf_backend:x
1429         {
1430             obj ~ @pdf.SDest.\exp_not:n{#2}
1431             [
1432                 \exp_args:Ne \pdf_object_ref:n { \l_pdf_current_structure_destination_
1433                 /FitR ~
1434                 @pdf_ #2 _llx ~ @pdf_ #2 _lly ~
1435                 @xpos ~ @ypos
1436             ]
1437         }
1438     }

```

```

1439     }
1440     \tex_vss:D
1441   }
1442   \__kernel_kern:n { -#1 }
1443 }

```

And now we redefine the destination command:

```

1444 \cs_set_protected:Npn \__pdf_backend_structure_destination:nnnn #1#2#3#4
1445 {
1446   \exp_args:Ne \__pdf_backend_structure_destination_aux:nnnn
1447   { \dim_eval:n {#2} } {#1} {#3} {#4}
1448 }

```

At last the goto link.

```

1449 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1450 {
1451   \__pdf_backend_link_begin:n { #1 /Subtype /Link /A << /S /GoTo /D ( #2 ) /SD~@pdf.SDest.
1452 }
1453 </xdvipdfmx | dvipdfmx>

```

(End definition for __pdf_backend_structure_destination:nn.)

Now pdftex. We only redefine for version 1.40 revision 24 or later.

```

1454 <*pdftex>
1455 \bool_lazy_and:nnT
1456 { \int_compare_p:nNn {\tex_pdftexversion:D } > {139} }
1457 { \int_compare_p:nNn {\tex_pdftexrevision:D } > {23} }
1458 {
1459   \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1460   {
1461     \tex_pdfdest:D
1462     name {#1}
1463     \str_case:nnF {#2}
1464     {
1465       { xyz } { xyz }
1466       { fit } { fit }
1467       { fitb } { fitb }
1468       { fitbh } { fitbh }
1469       { fitbv } { fitbv }
1470       { fith } { fith }
1471       { fitv } { fitv }
1472       { fitr } { fitr }
1473     }
1474     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1475     \scan_stop:
1476     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1477     {
1478       \tex_pdfdest:D
1479       struct~
1480       \int_use:c
1481       { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structur
1482       name {#1}
1483       \str_case:nnF {#2}
1484       {
1485         { xyz } { xyz }
1486         { fit } { fit }

```



```

1487         { fitb } { fitb }
1488         { fitbh } { fitbh }
1489         { fitbv } { fitbv }
1490         { fith } { fith }
1491         { fitv } { fitv }
1492         { fitr } { fitr }
1493     }
1494     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1495     \scan_stop:
1496 }
1497 }
1498 \cs_set_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
1499 {
1500     \tex_pdfdest:D
1501     name {#1}
1502     fitr ~
1503     width \dim_eval:n {#2} ~
1504     height \dim_eval:n {#3} ~
1505     depth \dim_eval:n {#4} \scan_stop:
1506     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1507     {
1508         \tex_pdfdest:D
1509         struct~
1510         \int_use:c
1511         { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n { \l_pdf_current_structure_
1512         name {#1}
1513         fitr ~
1514         width \dim_eval:n {#2} ~
1515         height \dim_eval:n {#3} ~
1516         depth \dim_eval:n {#4} \scan_stop:
1517     }
1518 }
1519 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1520 {
1521     \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1522 }
1523 }
1524 </pdftex>

```

luatex is quite similar to pdftex. Mostly the test for the version is different

```

1525 <*luatex>
1526 \int_compare:nNnT {\directlua{tex.print(status.list()["development_id"])} } > {7468}
1527 {
1528     \cs_set_protected:Npn \__pdf_backend_structure_destination:nn #1#2
1529     {
1530         \tex_pdfextension:D dest
1531         name {#1}
1532         \str_case:nnF {#2}
1533         {
1534             { xyz } { xyz }
1535             { fit } { fit }
1536             { fitb } { fitb }
1537             { fitbh } { fitbh }
1538             { fitbv } { fitbv }
1539             { fith } { fith }

```

```

1540         { fitv } { fitv }
1541         { fitr } { fitr }
1542     }
1543     { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1544     \scan_stop:
1545     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1546     {
1547         \tex_pdfextension:D dest
1548         struct~
1549         \int_use:c
1550         { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_
1551         name {#1}
1552         \str_case:nnF {#2}
1553         {
1554             { xyz } { xyz }
1555             { fit } { fit }
1556             { fitb } { fitb }
1557             { fitbh } { fitbh }
1558             { fitbv } { fitbv }
1559             { fith } { fith }
1560             { fitv } { fitv }
1561             { fitr } { fitr }
1562         }
1563         { xyz ~ zoom \fp_eval:n { #2 * 10 } }
1564         \scan_stop:
1565     }
1566 }
1567 \cs_set_protected:Npn \__pdf_backend_destination:nnnn #1#2#3#4
1568 {
1569     \tex_pdfextension:D dest
1570     name {#1}
1571     fitr ~
1572     width \dim_eval:n {#2} ~
1573     height \dim_eval:n {#3} ~
1574     depth \dim_eval:n {#4} \scan_stop:
1575     \exp_args:Ne \pdf_object_if_exist:nT { \l_pdf_current_structure_destination_tl }
1576     {
1577         \tex_pdfextension:D dest
1578         struct~
1579         \int_use:c
1580         { c__pdf_backend_object_ \exp_args:Ne \tl_to_str:n {\l_pdf_current_structure_
1581         name {#1}
1582         fitr ~
1583         width \dim_eval:n {#2} ~
1584         height \dim_eval:n {#3} ~
1585         depth \dim_eval:n {#4} \scan_stop:
1586     }
1587 }
1588 \cs_set_protected:Npn \__pdf_backend_link_begin_structure_goto:nnw #1#2
1589 {
1590     \__pdf_backend_link_begin:nnnw {#1} { goto~struct~name~{#2}~name } {#2}
1591 }
1592 }
1593 </luatex>

```

1.10 lua code for lualatex

```
1594 <lua>
1595 ltx= ltx or {}
1596 ltx.__pdf      = ltx.__pdf or {}
1597 ltx.__pdf.Page = ltx.__pdf.Page or {}
1598 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1599 ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}
1600 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1601 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1602 ltx.__pdf.object = ltx.__pdf.object or {}
1603
1604 ltx.pdf= ltx.pdf or {} -- for "public" functions
1605
1606 local __pdf = ltx.__pdf
1607 local pdf = pdf
1608
1609 local function __pdf_backend_Page_gput (name,value)
1610   __pdf.Page.dflt[name]=value
1611 end
1612
1613 local function __pdf_backend_Page_gremove (name)
1614   __pdf.Page.dflt[name]=nil
1615 end
1616
1617 local function __pdf_backend_Page_gclear ()
1618   __pdf.Page.dflt={}
1619 end
1620
1621 local function __pdf_backend_ThisPage_gput (page,name,value)
1622   __pdf.Page[page] = __pdf.Page[page] or {}
1623   __pdf.Page[page][name]=value
1624 end
1625
1626 local function __pdf_backend_ThisPage_gpush (page)
1627   local token=""
1628   local t = {}
1629   local tkeys= {}
1630   for name,value in pairs(__pdf.Page.dflt) do
1631     t[name]=value
1632   end
1633   if __pdf.Page[page] then
1634     for name,value in pairs(__pdf.Page[page]) do
1635       t[name] = value
1636     end
1637   end
1638   -- sort the table to get reliable test files.
1639   for name,value in pairs(t) do
1640     table.insert(tkeys,name)
1641   end
1642   table.sort(tkeys)
1643   for _,name in ipairs(tkeys) do
1644     token = token .. "/"..name.." " ..t[name]
1645   end
```

```

1646     return token
1647 end
1648
1649 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
1650 __pdf_backend_ThisPage_gput (page,name,value)
1651 end
1652
1653 function ltx.__pdf.backend_ThisPage_gpush (page)
1654     pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1655 end
1656
1657 function ltx.__pdf.backend_Page_gput (name,value)
1658     __pdf_backend_Page_gput (name,value)
1659 end
1660
1661 function ltx.__pdf.backend_Page_gremove (name)
1662     __pdf_backend_Page_gremove (name)
1663 end
1664
1665 function ltx.__pdf.backend_Page_gclear ()
1666     __pdf_backend_Page_gclear ()
1667 end
1668
1669
1670 local Properties = ltx.__pdf.Page.Resources.Properties
1671 local ResourceList= ltx.__pdf.Page.Resources.List
1672 local function __pdf_backend_PageResources_gpush (page)
1673     local token=""
1674     if Properties[page] then
1675         -- we sort the table, so that the pdf test works
1676         local t = {}
1677         for name,value in pairs (Properties[page]) do
1678             table.insert (t,name)
1679         end
1680         table.sort (t)
1681         for _,name in ipairs(t) do
1682             token = token .. "/"..name.." ".. Properties[page][name]
1683         end
1684         token = "/Properties <<"..token..">>"
1685     end
1686     for i,name in ipairs(ResourceList) do
1687         if ltx.__pdf.Page.Resources[name] then
1688             token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
1689         end
1690     end
1691     return token
1692 end
1693
1694 -- the function is public, as I probably need it in tagpdf too ...
1695 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1696 Properties[page] = Properties[page] or {}
1697 Properties[page][name]=value
1698 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1699 end

```

```

1700
1701 function ltx.pdf.Page_Resources_gpush(page)
1702   pdf.setpageresources(_pdf_backend_PageResources_gpush (page))
1703 end
1704
1705 function ltx.pdf.object_ref (objname)
1706   if ltx._pdf.object[objname] then
1707     local ref= ltx._pdf.object[objname]
1708     return ref
1709   else
1710     return "false"
1711   end
1712 end
1713  $\langle$ /lua $\rangle$ 

```

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	
bool commands:	
\bool_if:NTF	112, 121, 137, 143, 149, 156, 163,
. . . . 601, 634, 658, 692, 714, 753, 801	172, 199, 222, 232, 246, 258, 275,
\bool_lazy_and:nnTF	286, 293, 300, 309, 318, 325, 332,
\bool_new:N	339, 348, 357, 365, 368, 374, 379,
\bool_set_true:N 942, 1024, 1117, 1219	382, 410, 421, 427, 450, 454, 467,
box commands:	470, 471, 475, 478, 479, 483, 517,
\box_dp:N . . . 955, 1036, 1128, 1232, 1246	533, 612, 702, 820, 844, 851, 858,
\box_ht:N . . . 952, 1033, 1125, 1229, 1241	867, 871, 874, 877, 889, 894, 895,
\box_new:N 50, 51, 1114	934, 1001, 1016, 1081, 1104, 1185,
\box_scale:Nnn 1250	1202, 1203, 1210, 1304, 1341, 1398
\box_set_dp:Nn 1129, 1196, 1293, 1317	\cs_new_protected:Npx 131
\box_set_ht:Nn 1130, 1195, 1294, 1316	\cs_set_eq:NN 1349, 1350, 1351
\box_set_wd:Nn 1131, 1194, 1295, 1315	\cs_set_protected:Npn
\box_use:N 1299 497, 501, 505, 509, 513, 523,
\box_use_drop:N 1142, 1197, 1274, 1318	525, 527, 529, 531, 544, 563, 582,
\box_wd:N . . . 949, 1030, 1122, 1226, 1236	588, 594, 599, 606, 629, 653, 677,
	681, 686, 690, 697, 707, 746, 786,
	790, 795, 799, 806, 900, 904, 913,
	917, 924, 928, 1354, 1444, 1449,
	1459, 1498, 1519, 1528, 1567, 1588
C	
clist commands:	
\clist_const:Nn 386	
\clist_map_function:NN 835	
\clist_map_inline:Nn 395, 429, 445, 614	
cs commands:	
\cs_generate_variant:Nn 29, 30, 41, 42	
\cs_gset_eq:NN 602, 603,	
693, 694, 802, 803, 1343, 1344, 1345	
\cs_if_exist:NTF 398, 898	
\cs_new:Npn 37, 62, 68,	
217, 811, 1008, 1090, 1180, 1204, 1320	
D	
dim commands:	
\dim_eval:n 1447,	
1503, 1504, 1505, 1514, 1515, 1516,	
1572, 1573, 1574, 1583, 1584, 1585	
\dim_to_decimal_in_sp:n	
. 1236, 1241, 1246	
\c_zero_dim	
. . . 1129, 1130, 1131, 1293, 1294, 1295	

\directlua 59, 1526

E

exp commands:

\exp_args:Ne 1375,
1381, 1426, 1432, 1446, 1476, 1481,
1506, 1511, 1545, 1550, 1575, 1580
\exp_args:NNx 822
\exp_args:Nnx 457, 636, 660
\exp_args:Nnxx 716, 740, 755, 779
\exp_args:Nx 209, 320,
359, 632, 642, 656, 666, 710, 749, 1173
\exp_not:n 1358, 1379, 1430

F

fp commands:

\fp_eval:n
.. 1372, 1393, 1474, 1494, 1543, 1563

H

hbox commands:

\hbox:n 1403, 1414
\hbox_gset:Nn 1115
\hbox_set:Nn
... 940, 1022, 1187, 1217, 1251, 1306

hook commands:

\hook_gput_code:nnn .. 104, 448, 1296
\hook_gput_next_code:nn 1132
\hook_gset_rule:nnnn 442, 443

I

int commands:

\int_compare:nNnTF 911, 963, 1044, 1526
\int_compare_p:nNn 1456, 1457
\int_const:Nn .. 996, 1076, 1111, 1213
\int_gincr:N 175, 546, 565,
631, 655, 709, 713, 748, 752, 1110, 1212
\int_if_exist:NTF 1330
\int_new:N 54, 55, 56
\int_use:N
. 176, 177, 178, 182, 185, 186, 549,
557, 568, 576, 633, 638, 647, 657,
662, 671, 711, 718, 723, 725, 726,
730, 733, 734, 742, 750, 757, 762,
764, 765, 769, 772, 773, 781, 1004,
1010, 1083, 1091, 1182, 1260, 1287,
1311, 1322, 1480, 1510, 1549, 1579

K

kernel internal commands:

__kernel_backend_literal:n . 45,
547, 551, 566, 570, 584, 596, 608, 618
__kernel_backend_literal_page:n
..... 632, 656,
679, 688, 699, 710, 749, 788, 797, 808

__kernel_backend_postscript:n ..
..... 1253, 1275, 1281, 1308
__kernel_kern:n 1402, 1410, 1413, 1442
__kernel_pdf_name_from_unicode_
e:n 62, 68
__kernel_pdftdict_name:n 201, 202,
204, 432, 458, 616, 814, 825, 830,
943, 964, 975, 980, 985, 990, 1025,
1045, 1055, 1060, 1065, 1070, 1220
\g__kernel_pdfmanagement_end_
run_code_tl 77, 84, 91
\g__kernel_pdfmanagement_
thispage_shipout_code_tl 100, 106

L

latelua commands:

\latelua: 169, 255, 306, 345

M

mode commands:

\mode_leave_vertical: ... 1134, 1298

P

pdf commands:

\pdf_activate_structure_destination:
..... 1340, 1341
\l_pdf_current_structure_
destination_tl 1337,
1375, 1381, 1426, 1432, 1476, 1481,
1506, 1511, 1545, 1550, 1575, 1580
\pdf_object_if_exist:nTF
.. 1375, 1426, 1476, 1506, 1545, 1575
\pdf_object_ref:n .. 816, 977, 982,
987, 992, 1057, 1062, 1067, 1072,
1148, 1155, 1162, 1170, 1381, 1432
\pdf_object_ref_last: . 847, 854, 861
\pdf_object_unnamed_write:nn ...
..... 590, 683, 792, 846, 853, 860
\pdf_object_write 460

pdf internal commands:

__pdf_backend:n 139,
452, 461, 861, 926, 930, 1135, 1143,
1144, 1151, 1158, 1165, 1174, 1189,
1356, 1377, 1405, 1406, 1416, 1428
__pdf_backend_bdc:nn 12,
485, 497, 523, 599, 602, 603, 604,
690, 693, 694, 695, 799, 802, 803, 804
__pdf_backend_bdc_contobj:nn ...
..... 588, 602, 681, 693, 790, 802
__pdf_backend_bdc_contstream:nn
..... 594, 603, 686, 694, 795, 803
__pdf_backend_bdcobject:n
..... 12, 485,
505, 527, 563, 591, 653, 684, 746, 793

```

\__pdf_backend_bdcobject:nn . . . .
    . . . . 12, 485, 501, 525, 544, 629, 707
\__pdf_backend_bmc:n . . . . .
    . . . . 12, 485, 513, 531, 582, 677, 786
\__pdf_backend_catalog_gput:nn . . 19
\__pdf_backend_destination:nn . . .
    . . . . 1343, 1349
\__pdf_backend_destination:nnnn .
    . . . . 1344, 1350, 1498, 1567
\__pdf_backend_emc: . . . . .
    . . . . 12, 485, 509, 529, 606, 697, 806
\__pdf_backend_link_begin:n . . 1451
\__pdf_backend_link_begin:nnnw . .
    . . . . 1521, 1590
\__pdf_backend_link_begin_-
    goto:nnw . . . . . 1345, 1351
\__pdf_backend_link_begin_-
    structure_goto:nnw . . . . .
    . . . . 1345, 1351, 1449, 1519, 1588
\__pdf_backend_link_off: . . . . .
    . . . . 894, 900, 913, 924
\__pdf_backend_link_on: . . . . .
    . . . . 895, 904, 917, 928
\__pdf_backend_luastring:n . . . .
    . . . . 125, 217, 226, 238, 239, 250, 265, 266
\g__pdf_backend_name_int . . . . .
    . . . . 53, 546, 549, 557,
    . . . . 565, 568, 576, 631, 633, 638, 647,
    . . . . 655, 657, 662, 671, 709, 711, 748, 750
\__pdf_backend_Names_gpush:nn . . .
    . . . . 844, 851, 858, 867, 871
\__pdf_backend_NamesEmbeddedFiles_-
    add:nn . . . . . 873, 874, 877, 889
\g__pdf_backend_object_int . . . . .
    . . . . 1110, 1113, 1212, 1215, 1260
\__pdf_backend_object_last: . . . .
    . . . . 507, 577, 663, 672, 758, 782
\__pdf_backend_object_new:nn 397, 447
\__pdf_backend_object_ref:n 404,
    . . . . 463, 503, 558, 621, 639, 648, 719, 743
\__pdf_backend_object_write:nn . .
    . . . . 434, 448
\__pdf_backend_Page_gput:nn . . . .
    . . . . 5, 146, 156, 232, 293, 332, 368
\__pdf_backend_Page_gremove:n . . .
    . . . . 5, 146, 163, 246, 300, 339, 374
\g__pdf_backend_page_int . . . . . 53
\__pdf_backend_Page_primitive:n .
    . . . . 5, 146, 149, 209,
    . . . . 222, 286, 311, 320, 325, 350, 359, 365
\__pdf_backend_PageResources:n . .
    . . . . 450, 470, 478
\c__pdf_backend_PageResources_-
    clist . . 385, 395, 429, 445, 614, 836

\__pdf_backend_PageResources_-
    gpush:n . . . . .
    . . . . 12, 485, 517, 533, 612, 702, 820
\__pdf_backend_PageResources_-
    gpush_aux:n . . . . . 811, 837
\__pdf_backend_PageResources_-
    gput:nnn 394, 410, 421, 454, 471, 479
\__pdf_backend_PageResources_-
    obj_gpush: . 394, 427, 467, 475, 483
\__pdf_backend_Pages_primitive:n
    . . . . . 111, 112, 121, 131, 137, 143
\__pdf_backend_pdfmark:n . . . . .
    . . . . 499, 503, 507, 511, 515, 879
\__pdf_backend_ref_label:nn . . . .
    . . . . 31, 41, 178, 726, 765
\__pdf_backend_ref_value:nn . . . .
    . . . . 37, 42, 186, 734, 773
\g__pdf_backend_resourceid_int . .
    . . . . 53, 175, 176, 177,
    . . . . 178, 182, 185, 186, 713, 718, 723,
    . . . . 725, 726, 730, 733, 734, 742, 752,
    . . . . 757, 762, 764, 765, 769, 772, 773, 781
\__pdf_backend_structure_-
    destination:nn . . . . .
    . . 1343, 1349, 1353, 1354, 1459, 1528
\__pdf_backend_structure_-
    destination:nnnn 1344, 1350, 1444
\__pdf_backend_structure_-
    destination_aux:nnnn . . 1398, 1446
\__pdf_backend_ThisPage_gpush:n .
    . . . . 5, 146, 199, 275, 318, 357, 382
\__pdf_backend_ThisPage_gput:nn .
    . . . . 5, 146, 172, 258, 309, 348, 379
\g__pdf_backend_thispage_-
    shipout_tl . . . . . 5
\l__pdf_backend_tmpa_box . . . . .
    . . . . 47, 940, 949, 952, 955, 995,
    . . . . 1022, 1030, 1033, 1036, 1075, 1187,
    . . . . 1194, 1195, 1196, 1197, 1217, 1226,
    . . . . 1229, 1232, 1236, 1241, 1246, 1250,
    . . . . 1274, 1306, 1315, 1316, 1317, 1318
\l__pdf_backend_tmpb_box . . . . .
    . . . . 51, 1251, 1293, 1294, 1295, 1299
\l__pdf_backend_xform_bool . . . . .
    . . . . 488, 634,
    . . . . 658, 714, 753, 942, 1024, 1117, 1219
\__pdf_backend_xform_if_exist:n .
    . . . . 1328, 1334
\__pdf_backend_xform_new:nnnn . .
    . . . . 933, 934, 1016, 1104, 1202, 1210
\__pdf_backend_xform_ref:n . . . .
    . . . . 933, 1008,
    . . . . 1090, 1137, 1180, 1191, 1204, 1320

```

\l__pdf_backend_xform_tmpdp_tl ..	1208, 1244, 1258, 1265
\l__pdf_backend_xform_tmplt_tl ..	1209, 1239, 1263
\l__pdf_backend_xform_tmpwd_tl ..	1207, 1234, 1264
__pdf_backend_xform_use:n	933, 1001, 1081, 1185, 1203, 1304
\g__pdf_tmpa_prop	47, 201, 206, 211
\l__pdf_tmpa_tl	47, 179, 188, 190, 193, 727, 736, 738, 741, 766, 775, 777, 780, 783
pdfdict commands:	
\pdfdict_gput:nnn	158, 193, 295, 334, 370, 412, 423, 473, 481, 636, 660, 716, 740, 755, 779
\pdfdict_gremove:nn	165, 302, 341, 376
\pdfdict_if_exist:nTF	188, 736, 775
\pdfdict_item:nn	211, 816, 831
\pdfdict_new:n	190, 738, 777
\pdfdict_show:n	783
\pdfdict_use:n	321, 360, 436, 970, 1051
\pdfextension	915, 919
pdfmanagement internal commands:	
\g__pdfmanagement_active_bool	601, 692, 801
\pdfnames	19
\pdfpageref	2
\pdfrunninglinkoff	898, 902
\pdfrunninglinkon	906
pdfxform commands:	
\pdfxform_dp:n	1140, 1196, 1317
\pdfxform_ht:n	1139, 1195, 1316
\pdfxform_if_exist:n	1334
\pdfxform_wd:n	1138, 1194, 1315
prg commands:	
\prg_new_conditional:Npnn	1328
\prg_new_eq_conditional:NNn	1334
\prg_return_false:	1332
\prg_return_true:	1331
prop commands:	
\prop_count:N	964, 1045
\prop_gclear:N	943, 1025, 1220
\prop_gput:Nnn	206, 458
\prop_gset_eq:NN	201
\prop_if_empty:NTF	431, 616, 813, 974, 979, 984, 989, 1054, 1059, 1064, 1069
\prop_if_exist:NTF	202, 824
\prop_map_function:NN	211, 829
\prop_map_inline:Nn	204
\prop_new:N	48
\ProvidesExplFile	1
	R
ref commands:	
\ref_label:nn	29, 34, 177, 725, 764
\ref_value:nn	30, 39, 185, 733, 772
\relax	97
\RequirePackage	28
	S
scan commands:	
\scan_stop:	1005, 1087, 1475, 1495, 1505, 1516, 1544, 1564, 1574, 1585
str commands:	
\str_case:nnTF	1361, 1382, 1463, 1483, 1532, 1552
\str_convert_pdfname:n	64, 459
\str_if_eq:nnTF	1268, 1279
sys commands:	
\sys_if_engine luatex:TF	119
	T
TeX and L ^A T _E X 2 _ε commands:	
\@bsphack	33
\@esphack	35
\@kernel@after@enddocument@afterlastpage	74, 75
\@kernel@after@shipout@background	95, 98
\@kernel@after@shipout@lastpage	81, 82, 88, 89
\@kernel@before@shipout@background	97
\g@addto@macro	97, 98
\zref@extractdefault	181, 729, 768
\zref@labelbylist	176, 722, 761
tex commands:	
\tex_directlua:D	123, 234, 248, 398, 400
\tex_global:D	114, 151, 822
\tex_immediate:D	957, 1038
\tex_latelua:D	224, 260, 277, 413, 414, 642, 666
\tex_luaescapestring:D	219
\tex luatexversion:D	911
\tex_pdftest:D	1461, 1478, 1500, 1508
\tex_pdfextension:D	854, 1530, 1547, 1569, 1577
\tex_pdflastxform:D	998, 1078
\tex_pdfnames:D	847
\tex_pdfpageattr:D	151
\tex_pdfpageresources:D	822
\tex_pdfpagesattr:D	114
\tex_pdfrefxform:D	1003, 1083
\tex_pdftexrevision:D	1457
\tex_pdftexversion:D	1456
\tex_pdfxform:D	957, 1038
\tex_special:D	133, 288, 327

<code>\tex_the:D</code>	<code>\tl_if_exist:NTF</code> 95
.. 949, 952, 955, 1030, 1033, 1036,	<code>\tl_new:N</code> ... 49, 1207, 1208, 1209, 1338
1122, 1125, 1128, 1226, 1229, 1232	<code>\tl_set:Nn</code>
<code>\tex_unexpanded:D</code> 219	179, 727, 766, 1234, 1239, 1244
<code>\tex_vss:D</code> 1408, 1440	<code>\tl_to_str:n</code> 948, 951, 954, 997, 1004,
text commands:	1010, 1029, 1032, 1035, 1077, 1085,
<code>\text_expand:n</code> 64, 70	1091, 1112, 1121, 1124, 1127, 1182,
tl commands:	1214, 1225, 1228, 1231, 1287, 1311,
<code>\c_space_tl</code> 549, 557, 568, 576, 633,	1322, 1330, 1481, 1511, 1550, 1580
657, 711, 750, 1138, 1139, 1140, 1260	<code>\tl_use:N</code> 1258, 1263, 1264, 1265
<code>\tl_const:Nn</code>	
.. 947, 950, 953, 1028, 1031, 1034,	
1120, 1123, 1126, 1224, 1227, 1230	
<code>\tl_gput_right:Nn</code> 75, 82, 89	