

# The l3backend-pdf-extra package

## Additional backend PDF features

The L<sup>A</sup>T<sub>E</sub>X3 Project\*

Released XXXX-XX-XX

## 1 l3backend-pdf-extra Implementation

### 1.1 Crossreferences

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

```
1 <@@=pdf>
2 <*drivers>
3 \RequirePackage{l3ref-tmp}
4 \cs_generate_variant:Nn \ref_label:nn {en}
5 \cs_generate_variant:Nn \ref_value:nn {en}
6 \cs_new_protected:Npn \__pdf_backend_ref_label:nn #1 #2
7 {
8   \@bsphack
9   \ref_label:nn{#1}{abspage}
10  \@esphack
11 }
12 \cs_new:Npn \__pdf_backend_ref_value:nn #1 #2
13 {
14   \ref_value:nn{#1}{#2}
15 }
16 \cs_generate_variant:Nn \__pdf_backend_ref_label:nn {en}
17 \cs_generate_variant:Nn \__pdf_backend_ref_value:nn {en}
18 </drivers>
19 <*dvipdfmx | xdvipdfmx>
20 % avoid that destinations names are optimized.
21 % is this still needed??
22 % see https://tug.org/pipermail/dvipdfmx/2019-May/000002.html
23 \__kernel_backend_literal:x { dvipdfmx:config-C~ 0x0010 }
24 </dvipdfmx | xdvipdfmx>
```

```
\g__pdf_tmpa_prop Some scratch variables
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box
25 <*drivers>
26 \prop_new:N \g__pdf_tmpa_prop
27 \tl_new:N \l__pdf_tmpa_tl
```

---

\*E-mail: [latex-team@latex-project.org](mailto:latex-team@latex-project.org)

```

28 \box_new:N \l__pdf_backend_tmpa_box
29 </drivers>

```

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

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

```

30 <*drivers>
31 \int_new:N \g__pdf_backend_resourceid_int
32 \int_new:N \g__pdf_backend_name_int
33 \int_new:N \g__pdf_backend_page_int
34 </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.

```

35 <*luatex>
36 \directlua { require("l3backend-pdf.lua") }
37 </luatex>

```

## 1.3 Hooks

### 1.3.1 Add the “end run” hooks

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

```

38 <*pdftex | luatex>
39 \hook_gput_code:nnn {enddocument/afterlastpage}
40 {pdf/endrun}
41 {
42   \hook_use:n {pdf/management/end_run}
43 }
44 </pdftex | luatex>
45 <*dvipdfmx | xdvipdfmx>
46 \hook_gput_code:nnn {shipout/lastpage}
47 {pdf/endrun}
48 {
49   \hook_use:n {pdf/management/end_run}
50 }
51 \hook_gset_rule:nnnn {shipout/lastpage}{pdf}{before}{pdf/endrun}
52 </dvipdfmx | xdvipdfmx>
53 <*dvips>
54 \hook_gput_code:nnn {shipout/lastpage}
55 {pdf/endrun}
56 {
57   \hook_use:n {pdf/management/end_run}
58 }
59 \hook_gset_rule:nnnn {shipout/lastpage}{pdf}{before}{pdf/endrun}
60 </dvips>

```

### 1.3.2 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 ...

```

61 <*drivers>
62 \tl_if_exist:NTF \@kernel@after@shipout@background
63 {
64   \g@addto@macro \@kernel@before@shipout@background{\relax}
65   \g@addto@macro \@kernel@after@shipout@background
66     {
67       \hook_use:n {pdf/management/thispage_shipout}
68     }
69   \g@addto@macro \@kernel@after@shipout@lastpage
70     {
71       \hook_use:n {pdf/management/lastpage_shipout}
72     }
73 }
74 {
75   \hook_gput_code:nnn{shipout/background}{pdf}
76   {
77     \hook_use:n {pdf/management/thispage_shipout}
78   }
79   \hook_gput_code:nnn {shipout/lastpage} {pdf}
80   {
81     \hook_use:n {pdf/management/lastpage_shipout}
82   }
83 }
84
85 </drivers>

```

### 1.4 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.

```

86 <*pdftex>
87 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
88 {
89   \tex_global:D \tex_pdfpagesattr:D { #1 }
90 }
91 </pdftex>
92 <*luatex>
93 %luatex: does it in lua
94 \sys_if_engine_luatex:T
95 {
96   \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
97   {
98     \tex_directlua:D
99     {
100       pdf.setpagesattributes( \_pdf_backend_luastring:n { #1 } )
101     }

```

```

102     }
103   }
104   </luatex>
105   <*dvips>
106   \cs_new_protected:Npx \__pdf_backend_Pages_primitive:n #1
107   {
108     \tex_special:D{ps:~[#1~/PAGES~pdfmark} %]
109   }
110   </dvips>
111   <*dvipdfmx | xdvipdfmx>
112   \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
113   {
114     \__pdf_backend:n{put~@pages~<<#1>>}
115   }
116   </dvipdfmx | xdvipdfmx>
117   <*dvisvgm>
118   \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
119   {}
120   </dvisvgm>

```

(End definition for `\__pdf_backend_Pages_primitive:n`.)

## 1.5 “Page” and “ThisPage” attributes (pdfpageattr)

`\__pdf_backend_Page_primitive:n` `\__pdf_backend_Page_primitive:n` is the primitive command to add something to the /Page 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. `\__pdf_backend_Page_gput:nn` stores default 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_gpush: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`.

```

121 % backend commands
122 <*pdftex>
123 %the primitive
124 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
125 {
126   \tex_global:D \tex_pdfpageattr:D { #1 }
127 }
128 % the command to store default values.
129 % Uses a prop with pdflatex + dvi,
130 % sets a lua table with luatex
131 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2 %key,value
132 {
133   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
134 }
135 % the command to remove a default value.
136 % Uses a prop with pdflatex + dvi,
137 % changes a lua table with luatex
138 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
139 {
140   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
141 }

```

```

142 % the command used in the document.
143 % direct call of the primitive special with dvips/dvipdfmx
144 % \lualua: fill a page related table with luatex, merge it with the page
145 % table and push it directly
146 % write to aux and store in prop with pdflatex
147 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
148 {
149     %we need to know the page the resource should be added too.
150     \int_gincr:N\g__pdf_backend_resourceid_int
151     %\zref@labelbylist {l3pdf\int_use:N\g__pdf_backend_resourceid_int} {l3pdf}
152     %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
153     \__pdf_backend_ref_label:en { l3pdf\int_use:N\g__pdf_backend_resourceid_int }{abspage}
154     \tl_set:Nx \l__pdf_tmpa_tl
155     {
156         %\zref@extractdefault
157         {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
158         {pdf@abspage}
159         {0}
160         % \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
161         \__pdf_backend_ref_value:en {l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
162     }
163     \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
164     {
165         \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
166     }
167     %backend_Page has no handler.
168     \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }
169 }
170 %the code to push the values, used in shipout
171 %merges the two props and then fills the register in pdflatex
172 %merges the two tables and then fills (in lua) in luatex
173 %issues the values stored in the global prop with dvi
174 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
175 {
176     \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdfdict_name:n { g__pdf_Core/Page } }
177     \prop_if_exist:cT { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
178     {
179         \prop_map_inline:cn { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
180         {
181             \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
182         }
183     }
184     \exp_args:Nx \__pdf_backend_Page_primitive:n
185     {
186         \prop_map_function:NN \g__pdf_tmpa_prop \pdfdict_item:ne
187     }
188 }
189 </pdfTeX>
190 <*luatex>
191 % do we need to use some escaping for the values????
192 \cs_new:Npn \__pdf_backend_luastring:n #1
193 {
194     "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
195 }

```

```

196 %not used, only there for consistency
197 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
198 {
199   \tex_latelua:D
200   {
201     pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
202   }
203 }
204 % the command to store default values.
205 % Uses a prop with pdflatex + dvi,
206 % sets a lua table with luatex
207 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
208 {
209   \tex_directlua:D
210   {
211     ltx.__pdf.backend_Page_gput
212     (
213       \__pdf_backend_luastring:n { #1 },
214       \__pdf_backend_luastring:n { #2 }
215     )
216   }
217 }
218 % the command to remove a default value.
219 % Uses a prop with pdflatex + dvi,
220 % changes a lua table with luatex
221 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
222 {
223   \tex_directlua:D
224   {
225     ltx.__pdf.backend_Page_gremove (\__pdf_backend_luastring:n { #1 })
226   }
227 }
228 % the command used in the document.
229 % direct call of the primitive special with dvips/dvipdfmx
230 % \latelua: fill a page related table with luatex, merge it with the page
231 % table and push it directly
232 % write to aux and store in prop with pdflatex
233 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
234 {
235   \tex_latelua:D
236   {
237     ltx.__pdf.backend_ThisPage_gput
238     (
239       tex.count["g_shipout_readonly_int"],
240       \__pdf_backend_luastring:n { #1 },
241       \__pdf_backend_luastring:n { #2 }
242     )
243     ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
244   }
245 }
246 %the code to push the values, used in shipout
247 %merges the two props and then fills the register in pdflatex
248 %merges the two tables (the one is probably still empty) and then fills (in lua) in luatex
249 %issues the values stored in the global prop with dvi

```

```

250 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
251 {
252   \tex_latelua:D
253   {
254     ltx.__pdf.backend_ThisPage_gpush (tex.count["g_shipout_readonly_int"])
255   }
256 }
257
258 </luatex>
259 <*dvipdfmx | xdvipdfmx>
260 %the primitive
261 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
262 {
263   \tex_special:D{pdf:-put~@thispage-<<#1>>}
264 }
265 % the command to store default values.
266 % Uses a prop with pdflatex + dvi,
267 % sets a lua table with luatex
268 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
269 {
270   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
271 }
272 % the command to remove a default value.
273 % Uses a prop with pdflatex + dvi,
274 % changes a lua table with luatex
275 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
276 {
277   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
278 }
279 % the command used in the document.
280 % direct call of the primitive special with dvips/dvipdfmx
281 % \latelua: fill a page related table with luatex, merge it with the page
282 % table and push it directly
283 % write to aux and store in prop with pdflatex
284 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
285 {
286   \__pdf_backend_Page_primitive:n { /#1~#2 }
287 }
288 %the code to push the values, used in shipout
289 %merges the two props and then fills the register in pdflatex
290 %merges the two tables (the one is probably still empty)
291 % and then fills (in lua) in luatex
292 %issues the values stored in the global prop with dvi
293 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
294 {
295   \exp_args:Nx \__pdf_backend_Page_primitive:n
296   { \pdfdict_use:n { g__pdf_Core/Page} }
297 }
298 </dvipdfmx | xdvipdfmx>
299 <*dvips>
300 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
301 {
302   \tex_special:D{ps:-[ThisPage]<<#1>>~/PUT~pdfmark} %]
303 }

```

```

304 % the command to store default values.
305 % Uses a prop with pdflatex + dvi,
306 % sets a lua table with luatex
307 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
308 {
309     \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
310 }
311 % the command to remove a default value.
312 % Uses a prop with pdflatex + dvi,
313 % changes a lua table with luatex
314 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
315 {
316     \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
317 }
318 % the command used in the document.
319 % direct call of the primitive special with dvips/dvipdfmx
320 % \latelua: fill a page related table with luatex, merge it with the page
321 % table and push it directly
322 % write to aux and store in prop with pdflatex
323 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
324 {
325     \__pdf_backend_Page_primitive:n { /#1~#2 }
326 }
327 %the code to push the values, used in shipout
328 %merges the two props and then fills the register in pdflatex
329 %merges the two tables (the one is probably still empty)
330 %and then fills (in lua) in luatex
331 %issues the values stored in the global prop with dvi
332 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
333 {
334     \exp_args:Nx \__pdf_backend_Page_primitive:n
335         { \pdfdict_use:n { g__pdf_Core/Page} }
336 }
337 </dvips>
338 <*dvisvgm>
339 % mostly only dummies ...
340 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
341 {}
342 % Uses a prop with pdflatex + dvi,
343 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
344 {
345     \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
346 }
347 % the command to remove a default value.
348 % Uses a prop with pdflatex + dvi,
349 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
350 {
351     \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
352 }
353 % the command used in the document.
354 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
355 {}
356 %the code to push the values, used in shipout
357 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1

```



```

358 {}
359 </dvisvgm>

```

(End definition for `\_pdf_backend_Page_primitive:n` and others.)

## 1.6 “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.

`\_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 is a no-op with xdvipdfmx and dvips.

`\_pdf_backend_PageResources_obj_gpush:`

```

360 % backend commands the command to fill the register
361 % and to push the values.
362 %
363 % The names are quite often needed
364 % a similar list is now in l3pdfmanagement. Perhaps it should be merged.
365 <*drivers>
366 \clist_const:Nn \c__pdf_backend_PageResources_clist
367 {
368   ExtGState,
369   ColorSpace,
370   Pattern,
371   Shading,
372 }
373 </drivers>
374 % pdftex and luatex
375 <*pdftex | luatex>
376 %create the backend objects:
377 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
378 {
379   \_pdf_backend_object_new:nn {Page/Resources/#1} {dict}
380   \cs_if_exist:NT \tex_directlua:D
381   {
382     \tex_directlua:D
383     {
384       ltx.__pdf.object["Page/Resources/#1"]
385       =
386       "\_pdf_backend_object_ref:n{Page/Resources/#1}"
387     }
388   }
389 }
390 </pdftex | luatex>
391 <*luatex>
392 %values are only stored in a prop and will be output at end document.
393 \cs_new_protected:Npn \_pdf_backend_PageResources_gput:nnn #1 #2 #3
394 {
395   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
396   % luatex must also trigger the lua side

```

```

397 \tex_latelua:D{ltx.__pdf.Page.Resources.#1=true}
398 \tex_latelua:D
399 {
400   ltx.pdf.Page.Resources_gpush(tex.count["g_shipout_readonly_int"])
401 }
402 }
403 </luatex>
404 <*pdftex>
405 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
406 {
407   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
408 }
409 </pdftex>
410 <*pdftex|luatex>
411 %code for end of document code
412 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
413 {
414   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
415   {
416     \prop_if_empty:cF
417     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1 } }
418     {
419       \__pdf_backend_object_write:nx
420       { Page/Resources/##1 }
421       { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1 } }
422     }
423   }
424 }
425 </pdftex|luatex>
426 % xdvipdfmx
427 % \special{pdf:pageresources<<#1>>} doesn't work correctly with object names ...
428 % https://tug.org/pipermail/dvipdfmx/2019-August/000021.html,
429 % so we use \special{pdf:put @resources}
430 % this must be issued on every page!
431 <*dvipdfmx|xdvipdfmx>
432 %objects should not only be created but also "initialized"
433 % initialization should be done before anyone tries to write
434 % so we add rules for the backend.
435 <xdvipdfmx>\hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
436 <dvipdfmx>\hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
437 %
438 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
439 {
440   \__pdf_backend_object_new:nn { Page/Resources/#1 } { dict }
441   \hook_gput_code:nnn{shipout/firstpage}{pdf}{\__pdf_backend_object_write:nn { Page/Resour
442 }
443 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
444 {
445   \__pdf_backend:n {put~@resources~<<#1>>}
446 }
447 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
448 {
449   % this is not used for output, but there is a test if the resource is empty
450   \exp_args:Nnx

```

```

451 \prop_gput:cnn { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1 } }
452 { \str_convert_pdfname:n {#2} }{ #3 }
453 %objects are not filled with \pdf_object_write as this is not additive!
454 \__pdf_backend:x
455 {
456     put~\__pdf_backend_object_ref:n {Page/Resources/#1}<</#2~#3>>
457 }
458 }
459
460 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
461 </dvipdfmx | xdvipdfmx>
462 <*dvips>
463 % dvips unneeded, or no-op
464 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
465 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
466 { %only for the show command TEST!!
467     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
468 }
469 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
470 </dvips>
471 <*dvisvgm>
472 % dvips unneeded, or no-op
473 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
474 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
475 { %only for the show command TEST!!
476     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
477 }
478 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
479 </dvisvgm>

```

(End definition for \\_\_pdf\_backend\_PageResources\_gput:nnn and \\_\_pdf\_backend\_PageResources\_obj\_gpush:.)

### 1.6.1 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
480 % pdftex and luatex (and perhaps dvips ...) need to know if there are in a
481 % xform stream ...
482 <*drivers>
483 \bool_new:N \l__pdf_backend_xform_bool
484 </drivers>
485 <*dvips>
486 % dvips is easy: create an object, and reference it in the bdc
487 % ghostscript will then automatically replace it by a name
488 % and add the name to the /Properties dict
489 % special variant von accsupp
490 % https://chat.stackexchange.com/transcript/message/50831812#50831812
491 %
492 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
493 {
494     \__pdf_backend_pdfmark:x{/#1~<<#2>>~/BDC}

```

```

495 }
496 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
497 {
498   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_ref:n{#2}~/BDC}
499 }
500 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
501 {
502   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_last:~/BDC}
503 }
504 \cs_set_protected:Npn \__pdf_backend_emc:
505 {
506   \__pdf_backend_pdfmark:n{/EMC} %
507 }
508 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
509 {
510   \__pdf_backend_pdfmark:n{/#1~/BMC} %
511 }
512 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
513
514 </dvips>
515 <*dvisvgm>
516 % dvisvgm should do nothing
517 %
518 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
519 {}
520 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
521 {}
522 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
523 {}
524 \cs_set_protected:Npn \__pdf_backend_emc:
525 {}
526 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
527 {}
528 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
529
530 </dvisvgm>
531
532 % xetex has to create the entries in the /Properties manually
533 % (like the other backends)
534 % use pdfbase special
535 % https://chat.stackexchange.com/transcript/message/50832016#50832016
536 % the property is added to xform resources automatically,
537 % no need to worry about it.
538 <*dvipdfmx|xvipdfmx>
539 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
540 {
541   \int_gincr:N \g__pdf_backend_name_int
542   \__kernel_backend_literal:x
543   {
544     pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
545   }
546   \__kernel_backend_literal:x
547   {
548     pdf:put~@resources~

```

```

549         <<
550         /Properties~
551         <<
552         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_t1
553         \__pdf_backend_object_ref:n { #2 }
554         >>
555     >>
556 }
557 }
558 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
559 {
560     \int_gincr:N \g__pdf_backend_name_int
561     \__kernel_backend_literal:x
562     {
563         pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_t1 BDC
564     }
565     \__kernel_backend_literal:x
566     {
567         pdf:put~@resources~
568         <<
569         /Properties~
570         <<
571         /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_t1
572         \__pdf_backend_object_last:
573         >>
574         >>
575     }
576 }
577 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
578 {
579     \__kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
580 }
581
582 %this require management
583 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
584 {
585     \pdf_object_now:nn { dict }{ #2 }
586     \__pdf_backend_bdcobject:n { #1 }
587 }
588
589 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
590 {
591     \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
592 }
593
594 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
595 {
596     \bool_if:NTF \g__pdfmanagement_active_bool
597     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
598     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
599     \__pdf_backend_bdc:nn {#1}{#2}
600 }
601 \cs_set_protected:Npn \__pdf_backend_emc:
602 {

```

```

603 \__kernel_backend_literal:n {pdf:code~EMC} %pdfbase
604 }
605 % properties are handled automatically, but the other resources should be added
606 % at shipout
607 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
608 {
609   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
610   {
611     \prop_if_empty:cF { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/##1} }
612     {
613       \__kernel_backend_literal:x
614       {
615         pdf:put~@resources~
616         <</##1~\__pdf_backend_object_ref:n {Page/Resources/##1}>>
617       }
618     }
619   }
620 }
621 </dvipdfmx | xdvipdfmx>
622 % luatex + pdftex
623 <*luatex>
624 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
625 {
626   \int_gincr:N \g__pdf_backend_name_int
627   \exp_args:Nx\__kernel_backend_literal_page:n
628   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
629   \bool_if:NTF \l__pdf_backend_xform_bool
630   {
631     \exp_args:Nnx\pdffdict_gput:nnn
632     { g__pdf_Core/Xform/Resources/Properties }
633     { l3pdf\int_use:N\g__pdf_backend_name_int }
634     { \__pdf_backend_object_ref:n { #2 } }
635   }
636   {
637     \exp_args:Nx \tex_latelua:D
638     {
639       ltx.pdf.Page_Resources_Properties_gput
640       (
641         tex.count["g_shipout_readonly_int"],
642         "l3pdf\int_use:N\g__pdf_backend_name_int",
643         "\__pdf_backend_object_ref:n { #2 }"
644       )
645     }
646   }
647 }
648 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
649 {
650   \int_gincr:N \g__pdf_backend_name_int
651   \exp_args:Nx\__kernel_backend_literal_page:n
652   { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
653   \bool_if:NTF \l__pdf_backend_xform_bool
654   {
655     \exp_args:Nnx\pdffdict_gput:nnn %no handler needed
656     { g__pdf_Core/Xform/Resources/Properties }

```

```

657         { l3pdf\int_use:N\g__pdf_backend_name_int }
658         { \__pdf_backend_object_last: }
659     }
660     {
661         \exp_args:Nx \tex_latelua:D
662         {
663             ltx.pdf.Page_Resources_Properties_gput
664             (
665                 tex.count["g_shipout_readonly_int"],
666                 "l3pdf\int_use:N\g__pdf_backend_name_int",
667                 "\__pdf_backend_object_last:"
668             )
669         }
670     }
671 }
672 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
673 {
674     \__kernel_backend_literal_page:n { /#1~BMC }
675 }
676 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
677 {
678     \pdf_object_now:nn { dict } { #2 }
679     \__pdf_backend_bdcobject:n { #1 }
680 }
681 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
682 {
683     \__kernel_backend_literal_page:n { /#1-<<#2>>~BDC }
684 }
685 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
686 {
687     \bool_if:NTF \g__pdfmanagement_active_bool
688     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
689     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
690     \__pdf_backend_bdc:nn {#1}{#2}
691 }
692 \cs_set_protected:Npn \__pdf_backend_emc:
693 {
694     \__kernel_backend_literal_page:n { EMC }
695 }
696
697 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
698 </luatex>
699 <*pdfTeX>
700 % pdfLaTeX is the most complicated as it has to go through the aux ...
701 % the push command is extended to take other resources too
702 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
703 {
704     \int_gincr:N \g__pdf_backend_name_int
705     \exp_args:Nx \__kernel_backend_literal_page:n
706     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
707     % code to set the property ....
708     \int_gincr:N \g__pdf_backend_resourceid_int
709     \bool_if:NTF \l__pdf_backend_xform_bool
710     {

```

```

711 \exp_args:Nnxx\pdfdict_gput:nnn %no handler needed
712 { g__pdf_Core/Xform/Resources/Properties }
713 { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
714 { \__pdf_backend_object_ref:n { #2 } }
715 }
716 {
717 %\zref@labelbylist
718 % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
719 % { l3pdf }
720 % \ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
721 \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
722 \tl_set:Nx \l__pdf_tmpa_tl
723 {
724 %\zref@extractdefault
725 % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
726 % {pdf@abspage}
727 % {0}
728 %\ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
729 \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
730 }
731 \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
732 {
733 \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
734 }
735 \exp_args:Nnxx\pdfdict_gput:nnn
736 { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
737 { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
738 { \__pdf_backend_object_ref:n{#2} }
739 }
740 }
741 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
742 {
743 \int_gincr:N \g__pdf_backend_name_int
744 \exp_args:Nx\__kernel_backend_literal_page:n
745 { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
746 % code to set the property ...
747 \int_gincr:N\g__pdf_backend_resourceid_int
748 \bool_if:NTF \l__pdf_backend_xform_bool
749 {
750 \exp_args:Nnxx\pdfdict_gput:nnn
751 { g__pdf_Core/Xform/Resources/Properties }
752 { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
753 { \__pdf_backend_object_last: }
754 }
755 {
756 %\zref@labelbylist
757 % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
758 % { l3pdf }
759 %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
760 \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
761 \tl_set:Nx \l__pdf_tmpa_tl
762 {
763 %\zref@extractdefault
764 % { l3pdf\int_use:N\g__pdf_backend_resourceid_int }

```



```

765 %           {pdf@abspage}
766 %           {0}
767 %   \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
768 %   \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
769 %   }
770 %   \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
771 %   {
772 %       \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
773 %   }
774 %   \exp_args:Nnxx\pdfdict_gput:nnn
775 %   { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
776 %   { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
777 %   { \__pdf_backend_object_last: }
778 %   \%pdfdict_show:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
779 %   }
780 %   }
781 %   \cs_set_protected:Npn \__pdf_backend_bmc:n #1
782 %   {
783 %       \__kernel_backend_literal_page:n { /#1~BMC }
784 %   }
785 %   \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
786 %   {
787 %       \pdf_object_now:nn { dict } { #2 }
788 %       \__pdf_backend_bdcobject:n { #1 }
789 %   }
790 %   \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
791 %   {
792 %       \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
793 %   }
794 %   \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
795 %   {
796 %       \bool_if:NTF \g__pdfmanagement_active_bool
797 %       {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
798 %       {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
799 %       \__pdf_backend_bdc:nn {#1}{#2}
800 %   }
801 %   \cs_set_protected:Npn \__pdf_backend_emc:
802 %   {
803 %       \__kernel_backend_literal_page:n { EMC }
804 %   }
805 %   }
806 %   \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
807 %   {
808 %       \prop_if_empty:cF
809 %       { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1 } }
810 %       {
811 %           \pdfdict_item:ne { #1 } { \pdf_object_ref:n {Page/Resources/#1}}
812 %       }
813 %   }
814 %   }
815 %   \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
816 %   {
817 %       \exp_args:NNx \tex_global:D \tex_pdfpageresources:D
818 %       {

```

```

819         \prop_if_exist:cT
820         { \__kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
821         {
822             /Properties~
823             <<
824             \prop_map_function:cN
825             { \__kernel_pdffdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
826             \pdffdict_item:ne
827             >>
828         }
829         %% add ExtGState etc
830         \clist_map_function:NN
831         \c__pdf_backend_PageResources_clist
832         \__pdf_backend_PageResources_gpush_aux:n
833     }
834 }
835
836 </pdfTeX>

```

(End definition for \\_\_pdf\_backend\_bdc:nn and others.)

## 1.7 “Catalog” & subdirectories (pdfcatalog)

The backend command is already in the driver: \\_\_pdf\_backend\_catalog\_gput:nn

### 1.7.1 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 we need some code to push the tree if files have been added. dvips wants code for every file and then creates the Name tree automatically.

```

837 % pdfLaTeX
838 <*pdfTeX>
839 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
840 {
841     \pdf_object_now:nn {dict} {/Names [#1] }
842     \tex_pdfnames:D {/EmbeddedFiles~\pdf_object_last:}
843 }
844 </pdfTeX>
845 <*luatex>
846 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
847 {
848     \pdf_object_now:nn {dict} {/Names [#1] }
849     \tex_pdfextension:D~names~{/EmbeddedFiles~\pdf_object_last: }
850 }
851 </luatex>
852 <*dviPDFmx | xdvipdfmx>
853 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
854 {
855     \pdf_object_now:nn {dict} { /Names [#1] }
856     %n or x?
857     \__pdf_backend:x {put~@names~<</EmbeddedFiles~\pdf_object_last: >>}
858 }
859 </dviPDFmx | xdvipdfmx>

```

```

860
861 %dvips: noop
862 <*dvips>
863 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
864 </dvips>
865 %dvisvgm: noop
866 <*dvisvgm>
867 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
868 </dvisvgm>
869

```

Names in the EmbeddedFiles name tree must sorted alphabetically, so we need commands to create this names. And we need a sequence to store the names and the objects. We use the prefix l3ef, and we assume that at most 9999 files will be used.

\g\_\_pdf\_backend\_EmbeddedFiles\_int

*(End definition for \g\_\_pdf\_backend\_EmbeddedFiles\_int.)*

\\_\_pdf\_backend\_EmbeddedFiles\_name:

```

870 <*drivers>
871 \int_new:N \g__pdf_backend_EmbeddedFiles_int
872 \cs_new:Npn \__pdf_backend_EmbeddedFiles_name:
873 {
874 (
875   l3ef
876   \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {10}
877   {0}
878   \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {100}
879   {0}
880   \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {1000}
881   {0}
882   \int_use:N \g__pdf_backend_EmbeddedFiles_int
883 )
884 }
885 </drivers>

```

*(End definition for \\_\_pdf\_backend\_EmbeddedFiles\_name:.)*

\g\_\_pdf\_backend\_EmbeddedFiles\_seq  
\g\_\_pdf\_backend\_EmbeddedFiles\_named\_prop

The sequence will hold the content of the array that is pushed out at then end (not with dvips), the prop holds the obj names-names relation.

*(End definition for \g\_\_pdf\_backend\_EmbeddedFiles\_seq and \g\_\_pdf\_backend\_EmbeddedFiles\_named\_prop.)*

```

886 <*drivers>
887 \seq_new:N \g__pdf_backend_EmbeddedFiles_seq
888 \prop_new:N \g__pdf_backend_EmbeddedFiles_named_prop
889 </drivers>

```

\\_\_pdf\_backend\_NamesEmbeddedFiles\_add:n

This command saves an object reference of a filespec dictionary in the EmbeddedFiles name tree. We define a prop to store the relation between object name and name in the name tree.

```

890 <*pdfTeX | luatex | dvipdfmx | xdvipdfmx>
891 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
892   {%#1 object ref

```

```

893 {
894   \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
895   \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
896   { #1 }
897   { \__pdf_backend_EmbeddedFiles_name: }
898   \seq_gput_right:Nx \g__pdf_backend_EmbeddedFiles_seq
899   { \__pdf_backend_EmbeddedFiles_name: \c_space_tl #1 }
900 }
901
902 </pdfTeX | luatex | dvipdfmx | xdvipdfmx>
903 <*dvips>
904 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
905 {
906   \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
907   \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
908   { #1 }
909   { \__pdf_backend_EmbeddedFiles_name: }
910   \__pdf_backend_pdfmark:x
911   {
912     /Name~\__pdf_backend_EmbeddedFiles_name:~
913     /FS~#1~
914     /EMBED
915   }
916 }
917 </dvips>
918 <*dvisvgm>
919 %no op. Or is there any sensible use for it?
920 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
921 {}
922 </dvisvgm>

```

(End definition for \\_\_pdf\_backend\_NamesEmbeddedFiles\_add:n.)

## 1.7.2 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 923 <*pdfTeX>
\__pdf_backend_xform_ref:n 924 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
925 % #1 name
926 % #2 attributes
927 % #3 resources
928 % #4 content, not necessarily a box!
929 {
930   \hbox_set:Nn \l__pdf_backend_tmpa_box
931   {
932     \bool_set_true:N \l__pdf_backend_xform_bool
933     \prop_gclear:c {\__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
934     #4
935   }

```

```

936 %store the dimensions
937 \tl_const:cx
938 { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
939 { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
940 \tl_const:cx
941 { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
942 { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
943 \tl_const:cx
944 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
945 { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
946 %% do we need to test if #2 and #3 are empty??
947 \tex_immediate:D \tex_pdfxform:D
948 ~ attr ~ { #2 }
949 %% which other resources should be default? Is an argument actually needed?
950 ~ resources ~
951 {
952   #3
953   \int_compare:nNnT
954     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
955     >
956     { 0 }
957     {
958       /Properties~
959       <<
960       \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
961       >>
962     }
963
964   \prop_if_empty:cF
965     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
966     {
967       /ExtGState~ \pdf_object_ref:n { Page/Resources/ExtGState }
968     }
969   \prop_if_empty:cF
970     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
971     {
972       /Pattern~ \pdf_object_ref:n { Page/Resources/Pattern }
973     }
974   \prop_if_empty:cF
975     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
976     {
977       /Shading~ \pdf_object_ref:n { Page/Resources/Shading }
978     }
979   \prop_if_empty:cF
980     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
981     {
982       /ColorSpace~ \pdf_object_ref:n { Page/Resources/ColorSpace }
983     }
984 }
985 \l__pdf_backend_tmpa_box
986 \int_const:cn
987 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
988 { \tex_pdflastxform:D }
989 }

```

```

990
991 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
992 {
993   \tex_pdfrefxform:D
994   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
995   \scan_stop:
996 }
997
998 \cs_new:Npn \__pdf_backend_xform_ref:n #1
999 {
1000   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R
1001 }
1002 </pdfTeX>
1003 <*luatex>
1004 %luatex
1005 %nearly identical but not completely ...
1006 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1007 % #1 name
1008 % #2 attributes
1009 % #3 resources
1010 % #4 content, not necessarily a box!
1011 {
1012   \hbox_set:Nn \l__pdf_backend_tmpa_box
1013   {
1014     \bool_set_true:N \l__pdf_backend_xform_bool
1015     \prop_gclear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }
1016     #4
1017   }
1018   \tl_const:cx
1019   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1020   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1021   \tl_const:cx
1022   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1023   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1024   \tl_const:cx
1025   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1026   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1027   %% do we need to test if #2 and #3 are empty??
1028   \tex_immediate:D \tex_pdfxform:D
1029   ~ attr ~ { #2 }
1030   %% which resources should be default? Is an argument actually needed?
1031   ~ resources ~
1032   {
1033     #3
1034     \int_compare:nNnT
1035     {\prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties
1036     >
1037     { 0 }
1038     {
1039       /Properties~
1040       <<
1041       \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1042       >>
1043     }

```

```

1044 \prop_if_empty:cF
1045 { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1046 {
1047 /ExtGState~ \pdf_object_ref:n { Page/Resources/ExtGState }
1048 }
1049 \prop_if_empty:cF
1050 { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1051 {
1052 /Pattern~ \pdf_object_ref:n { Page/Resources/Pattern }
1053 }
1054 \prop_if_empty:cF
1055 { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1056 {
1057 /Shading~ \pdf_object_ref:n { Page/Resources/Shading }
1058 }
1059 \prop_if_empty:cF
1060 { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1061 {
1062 /ColorSpace~ \pdf_object_ref:n { Page/Resources/ColorSpace }
1063 }
1064 }
1065 \l__pdf_backend_tmpa_box
1066 \int_const:cn
1067 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1068 { \tex_pdflastxform:D }
1069 }
1070
1071 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1072 {
1073 \tex_pdfrefxform:D \int_use:c
1074 {
1075 c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1076 }
1077 \scan_stop:
1078 }
1079
1080 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1081 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1082
1083 </luatex>
1084 <*dvipdfmx | xdvipdfmx>
1085 % xetex
1086 % it needs a bit testing if it really works to set the box to 0 before the special ...
1087 % does it disturb viewing the xobject?
1088 % what happens with the resources (bdc)? (should work as they are specials too)
1089 % xetex requires that the special is in horizontal mode. This means it affects
1090 % typesetting. But we can no delay the whole form code to shipout
1091 % as the object reference and the size is often wanted on the current page.
1092 % so we need to allocate a box - but probably they won't be thousands xform
1093 % in a document so it shouldn't matter.
1094 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1095 % #1 name
1096 % #2 attributes
1097 % #3 resources

```

```

1098 % #4 content, not necessarily a box!
1099 {
1100   \int_gincr:N \g__pdf_backend_object_int
1101   \int_const:cn
1102     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1103     { \g__pdf_backend_object_int }
1104   \box_new:c { g__pdf_backend_xform_#1_box }
1105   \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1106   {
1107     \bool_set_true:N \l__pdf_backend_xform_bool
1108     #4
1109   }
1110   \tl_const:cx
1111     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1112     { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1113   \tl_const:cx
1114     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1115     { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1116   \tl_const:cx
1117     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1118     { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1119   \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1120   \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1121   \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1122   \hook_gput_next_code:n {shipout/background}
1123   {
1124     \mode_leave_vertical: %needed, the xform disappears without it.
1125     \__pdf_backend:x
1126     {
1127       bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1128       \c_space_tl width ~ \pdfxform_wd:n { #1 }
1129       \c_space_tl height ~ \pdfxform_ht:n { #1 }
1130       \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1131     }
1132     \box_use_drop:c { g__pdf_backend_xform_#1_box }
1133     \__pdf_backend:x {put ~ @resources ~<<#3>> }
1134     \__pdf_backend:x
1135     {
1136       put~ @resources ~
1137       <<
1138       /ExtGState~ \pdf_object_ref:n { Page/Resources/ExtGState }
1139       >>
1140     }
1141     \__pdf_backend:x
1142     {
1143       put~ @resources ~
1144       <<
1145       /Pattern~ \pdf_object_ref:n { Page/Resources/Pattern }
1146       >>
1147     }
1148     \__pdf_backend:x
1149     {
1150       put~ @resources ~
1151       <<

```



```

1152         /Shading~ \pdf_object_ref:n { Page/Resources/Shading }
1153     >>
1154 }
1155 \__pdf_backend:x
1156 {
1157     put~ @resources ~
1158     <<
1159         /ColorSpace~
1160         \pdf_object_ref:n { Page/Resources/ColorSpace }
1161     >>
1162 }
1163 \exp_args:Nx
1164 \__pdf_backend:x {exobj ~<<#2>>}
1165 }
1166 }
1167
1168
1169
1170 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1171 {
1172     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1173 }
1174
1175 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1176 {
1177     \hbox_set:Nn \l__pdf_backend_tmpa_box
1178     {
1179         \__pdf_backend:x
1180         {
1181             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1182         }
1183     }
1184     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1185     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1186     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1187     \box_use_drop:N \l__pdf_backend_tmpa_box
1188 }
1189 </dvpdvmx | xdvipdvmx>
1190 <*dvisvgm>
1191 % unclear what it should do!!
1192 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1193 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1194 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1195 </dvisvgm>
1196 <*drivers>
1197 %% all
1198 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1199 {
1200     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1201     { \prg_return_true: }
1202     { \prg_return_false: }
1203 }
1204 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n \__pdf_backend_xform_if_exist:n
1205 { TF , T , F , p }

```

1206  $\langle$ /drivers $\rangle$

(End definition for `__pdf_backend_xform_new:nnnn`, `__pdf_backend_xform_use:n`, and `__pdf_backend_xform_ref:n`.)

## 1.8 lua code for lualatex

```
1207  $\langle$ *lua $\rangle$ 
1208 ltx= ltx or {}
1209 ltx.__pdf = ltx.__pdf or {}
1210 ltx.__pdf.Page = ltx.__pdf.Page or {}
1211 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1212 ltx.__pdf.Resources = ltx.__pdf.Resources or {}
1213 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1214 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1215 ltx.__pdf.object = ltx.__pdf.object or {}
1216
1217 ltx.pdf= ltx.pdf or {} -- for "public" functions
1218
1219 local __pdf = ltx.__pdf
1220 local pdf = pdf
1221
1222 local function __pdf_backend_Page_gput (name,value)
1223   __pdf.Page.dflt[name]=value
1224 end
1225
1226 local function __pdf_backend_Page_gremove (name)
1227   __pdf.Page.dflt[name]=nil
1228 end
1229
1230 local function __pdf_backend_Page_gclear ()
1231   __pdf.Page.dflt={}
1232 end
1233
1234 local function __pdf_backend_ThisPage_gput (page,name,value)
1235   __pdf.Page[page] = __pdf.Page[page] or {}
1236   __pdf.Page[page][name]=value
1237 end
1238
1239 local function __pdf_backend_ThisPage_gpush (page)
1240   local token=""
1241   local t = {}
1242   local tkeys= {}
1243   for name,value in pairs(__pdf.Page.dflt) do
1244     t[name]=value
1245   end
1246   if __pdf.Page[page] then
1247     for name,value in pairs(__pdf.Page[page]) do
1248       t[name] = value
1249     end
1250   end
1251   -- sort the table to get reliable test files.
1252   for name,value in pairs(t) do
1253     table.insert(tkeys,name)
```

```

1254 end
1255 table.sort(tkeys)
1256 for _,name in ipairs(tkeys) do
1257     token = token .. "/"..name.." "..t[name]
1258 end
1259 return token
1260 end
1261
1262 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
1263 __pdf_backend_ThisPage_gput (page,name,value)
1264 end
1265
1266 function ltx.__pdf.backend_ThisPage_gpush (page)
1267     pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1268 end
1269
1270 function ltx.__pdf.backend_Page_gput (name,value)
1271     __pdf_backend_Page_gput (name,value)
1272 end
1273
1274 function ltx.__pdf.backend_Page_gremove (name)
1275     __pdf_backend_Page_gremove (name)
1276 end
1277
1278 function ltx.__pdf.backend_Page_gclear ()
1279     __pdf_backend_Page_gclear ()
1280 end
1281
1282
1283 local Properties = ltx.__pdf.Page.Resources.Properties
1284 local ResourceList= ltx.__pdf.Page.Resources.List
1285 local function __pdf_backend_PageResources_gpush (page)
1286     local token=""
1287     if Properties[page] then
1288         -- we sort the table, so that the pdf test works
1289         local t = {}
1290         for name,value in pairs (Properties[page]) do
1291             table.insert (t,name)
1292         end
1293         table.sort (t)
1294         for _,name in ipairs(t) do
1295             token = token .. "/"..name.." ".. Properties[page][name]
1296         end
1297         token = "/Properties <<"..token..">>"
1298     end
1299     for i,name in ipairs(ResourceList) do
1300         if ltx.__pdf.Page.Resources[name] then
1301             token = token .. "/"..name.." "..ltx.pdf.object_ref("Page/Resources/"..name)
1302         end
1303     end
1304     return token
1305 end
1306
1307 -- the function is public, as I probably need it in tagpdf too ...

```

```

1308 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1309 Properties[page] = Properties[page] or {}
1310 Properties[page][name]=value
1311 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1312 end
1313
1314 function ltx.pdf.Page_Resources_gpush(page)
1315 pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1316 end
1317
1318 function ltx.pdf.object_ref (objname)
1319 if ltx.__pdf.object[objname] then
1320 local ref= ltx.__pdf.object[objname]
1321 return ref
1322 else
1323 return "false"
1324 end
1325 end
1326 </lua>

```

## 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>B</b>	
bool commands:	
\bool_if:NTF . . . . .	596, 629, 653, 687, 709, 748, 796
\bool_new:N . . . . .	483
\bool_set_true:N . . . . .	932, 1014, 1107
box commands:	
\box_dp:N . . . . .	945, 1026, 1118
\box_ht:N . . . . .	942, 1023, 1115
\box_new:N . . . . .	28, 1104
\box_set_dp:Nn . . . . .	1119, 1186
\box_set_ht:Nn . . . . .	1120, 1185
\box_set_wd:Nn . . . . .	1121, 1184
\box_use_drop:N . . . . .	1132, 1187
\box_wd:N . . . . .	939, 1020, 1112
<b>C</b>	
clist commands:	
\clist_const:Nn . . . . .	366
\clist_map_function:NN . . . . .	830
\clist_map_inline:Nn . . . . .	377, 414, 438, 609
cs commands:	
\cs_generate_variant:Nn . . . . .	4, 5, 16, 17
\cs_gset_eq:NN . . . . .	597, 598, 688, 689, 797, 798
\cs_if_exist:NTF . . . . .	380
\cs_new:Npn . . . . .	12, 192, 806, 872, 998, 1080, 1170, 1194
\cs_new_protected:Npn . . . . .	6, 87, 96, 112, 118, 124, 131, 138, 147, 174, 197, 207, 221, 233, 250, 261, 268, 275, 284, 293, 300, 307, 314, 323, 332, 340, 343, 349, 354, 357, 393, 405, 412, 443, 447, 460, 464, 465, 469, 473, 474, 478, 512, 528, 607, 697, 815, 839, 846, 853, 863, 867, 891, 904, 920, 924, 991, 1006, 1071, 1094, 1175, 1192, 1193
\cs_new_protected:Npx . . . . .	106
\cs_set_protected:Npn . . . . .	492, 496, 500, 504, 508, 518, 520, 522, 524, 526, 539, 558, 577, 583, 589, 594, 601, 624, 648, 672, 676, 681, 685, 692, 702, 741, 781, 785, 790, 794, 801
<b>D</b>	
dim commands:	
\c_zero_dim . . . . .	1119, 1120, 1121
\directlua . . . . .	36
<b>E</b>	
exp commands:	
\exp_args:NNx . . . . .	817

`\exp_args:Nnx` . . . . . 450, 631, 655  
`\exp_args:Nnxx` . . . . . 711, 735, 750, 774  
`\exp_args:Nx` . . . . . 184, 295,  
334, 627, 637, 651, 661, 705, 744, 1163

## H

hbox commands:  
`\hbox_gset:Nn` . . . . . 1105  
`\hbox_set:Nn` . . . . . 930, 1012, 1177

hook commands:  
`\hook_gput_code:nnn` . . . . .  
. . . . . 39, 46, 54, 75, 79, 441  
`\hook_gput_next_code:nn` . . . . . 1122  
`\hook_gset_rule:nnnn` . . . . . 51, 59, 435, 436  
`\hook_use:n` . . . . . 42, 49, 57, 67, 71, 77, 81

## I

int commands:  
`\int_compare:nNnTF` . . . . .  
. . . . . 876, 878, 880, 953, 1034  
`\int_const:Nn` . . . . . 986, 1066, 1101  
`\int_gincr:N` . . . . . 150, 541, 560, 626,  
650, 704, 708, 743, 747, 894, 906, 1100  
`\int_if_exist:Ntf` . . . . . 1200  
`\int_new:N` . . . . . 31, 32, 33, 871  
`\int_use:N` . . . . . 151, 152, 153, 157, 160,  
161, 544, 552, 563, 571, 628, 633,  
642, 652, 657, 666, 706, 713, 718,  
720, 721, 725, 728, 729, 737, 745,  
752, 757, 759, 760, 764, 767, 768,  
776, 882, 994, 1000, 1073, 1081, 1172

## K

kernel internal commands:  
`\__kernel_backend_literal:n` . . . . . 23,  
542, 546, 561, 565, 579, 591, 603, 613  
`\__kernel_backend_literal_page:n`  
. . . . . 627, 651,  
674, 683, 694, 705, 744, 783, 792, 803  
`\__kernel_pdfdict_name:n` . . . . .  
. . . . . 176, 177, 179, 417, 451, 611, 809,  
820, 825, 933, 954, 965, 970, 975,  
980, 1015, 1035, 1045, 1050, 1055, 1060

## L

lualua commands:  
`\lualua:` . . . . . 144, 230, 281, 320

## M

mode commands:  
`\mode_leave_vertical:` . . . . . 1124

## P

pdf commands:  
`\pdf_object_last:` . . . . . 842, 849, 857

`\pdf_object_now:nn` . . . . .  
. . . . . 585, 678, 787, 841, 848, 855  
`\pdf_object_ref:n` . . . . .  
811, 967, 972, 977, 982, 1047, 1052,  
1057, 1062, 1138, 1145, 1152, 1160  
`\pdf_object_write` . . . . . 453

pdf internal commands:  
`\__pdf_backend:n` . . . . .  
. . . . . 114, 445, 454, 857, 1125, 1133,  
1134, 1141, 1148, 1155, 1164, 1179  
`\__pdf_backend_bdc:nn` . . . . . 11,  
480, 492, 518, 594, 597, 598, 599,  
685, 688, 689, 690, 794, 797, 798, 799  
`\__pdf_backend_bdc_contobj:nn` . . . . .  
. . . . . 583, 597, 676, 688, 785, 797  
`\__pdf_backend_bdc_contstream:nn`  
. . . . . 589, 598, 681, 689, 790, 798  
`\__pdf_backend_bdcobject:n` . . . . .  
. . . . . 11, 480,  
500, 522, 558, 586, 648, 679, 741, 788  
`\__pdf_backend_bdcobject:nn` . . . . .  
. . . . . 11, 480, 496, 520, 539, 624, 702  
`\__pdf_backend_bmc:n` . . . . .  
. . . . . 11, 480, 508, 526, 577, 672, 781  
`\__pdf_backend_catalog_gput:nn` . . . . . 18  
`\g__pdf_backend_EmbeddedFiles_-`  
`int` . . . . . 870,  
871, 876, 878, 880, 882, 894, 906  
`\__pdf_backend_EmbeddedFiles_-`  
`name:` . . . . . 870, 872, 897, 899, 909, 912  
`\g__pdf_backend_EmbeddedFiles_-`  
`named_prop` . . . . . 886, 888, 895, 907  
`\g__pdf_backend_EmbeddedFiles_-`  
`seq` . . . . . 886, 887, 898  
`\__pdf_backend_emc:` . . . . .  
. . . . . 11, 480, 504, 524, 601, 692, 801  
`\__pdf_backend_luastring:n` . . . . .  
. . . . . 100, 192, 201, 213, 214, 225, 240, 241  
`\g__pdf_backend_name_int` . . . . .  
. . . . . 30, 541, 544, 552,  
560, 563, 571, 626, 628, 633, 642,  
650, 652, 657, 666, 704, 706, 743, 745  
`\__pdf_backend_NamesEmbeddedFiles_-`  
`add:n` . . . . . 890, 891, 904, 920  
`\__pdf_backend_NamesEmbeddedFiles_-`  
`gpush:n` . . . . . 839, 846, 853, 863, 867  
`\g__pdf_backend_object_int` . . . . . 1100, 1103  
`\__pdf_backend_object_last:` . . . . .  
. . . . . 502, 572, 658, 667, 753, 777  
`\__pdf_backend_object_new:nn` . . . . . 379, 440  
`\__pdf_backend_object_ref:n` . . . . . 386,  
456, 498, 553, 616, 634, 643, 714, 738  
`\__pdf_backend_object_write:nn` . . . . .  
. . . . . 419, 441

\_pdf_backend_Page_gput:nn . . . .	\g_pdf_tmpa_prop . . . 25, 176, 181, 186
. . . . . 4, 121, 131, 207, 268, 307, 343	\l_pdf_tmpa_tl . . . . .
\_pdf_backend_Page_gremove:n . . .	. . . . . 25, 154, 163, 165, 168, 722,
. . . . . 4, 121, 138, 221, 275, 314, 349	731, 733, 736, 761, 770, 772, 775, 778
\g_pdf_backend_page_int . . . . . 30	pdfdict commands:
\_pdf_backend_Page_primitive:n . .	\pdfdict_gput:nnn . . . . .
. . . . . 4, 121, 124, 184,	. 133, 168, 270, 309, 345, 395, 407,
197, 261, 286, 295, 300, 325, 334, 340	467, 476, 631, 655, 711, 735, 750, 774
\_pdf_backend_PageResources:n . .	\pdfdict_gremove:nn 140, 277, 316, 351
. . . . . 443, 464, 473	\pdfdict_if_exist:nTF . 163, 731, 770
\c_pdf_backend_PageResources_-	\pdfdict_item:nn . . . . . 186, 811, 826
clist . . 366, 377, 414, 438, 609, 831	\pdfdict_new:n . . . . . 165, 733, 772
\_pdf_backend_PageResources_-	\pdfdict_show:n . . . . . 778
gpush:n . . . . .	\pdfdict_use:n 296, 335, 421, 960, 1041
. . . . . 11, 480, 512, 528, 607, 697, 815	pdfmanagement internal commands:
\_pdf_backend_PageResources_-	\g_pdfmanagement_active_bool . . .
gpush_aux:n . . . . . 806, 832	. . . . . 596, 687, 796
\_pdf_backend_PageResources_-	\pdfnames . . . . . 18
gput:nnn 360, 393, 405, 447, 465, 474	\pdfpageref . . . . . 2
\_pdf_backend_PageResources_-	pdfxform commands:
obj_gpush: . 360, 412, 460, 469, 478	\pdfxform_dp:n . . . . . 1130, 1186
\_pdf_backend_Pages_primitive:n	\pdfxform_ht:n . . . . . 1129, 1185
. . . . . 86, 87, 96, 106, 112, 118	\pdfxform_if_exist:n . . . . . 1204
\_pdf_backend_pdfmark:n . . . . .	\pdfxform_wd:n . . . . . 1128, 1184
. . . . . 494, 498, 502, 506, 510, 910	prg commands:
\_pdf_backend_ref_label:nn . . . .	\prg_new_conditional:Npnn . . . . 1198
. . . . . 6, 16, 153, 721, 760	\prg_new_eq_conditional:NNn . . 1204
\_pdf_backend_ref_value:nn . . . .	\prg_return_false: . . . . . 1202
. . . . . 12, 17, 161, 729, 768	\prg_return_true: . . . . . 1201
\g_pdf_backend_resourceid_int . .	prop commands:
. . . . . 30, 150, 151, 152,	\prop_count:N . . . . . 954, 1035
153, 157, 160, 161, 708, 713, 718,	\prop_gclear:N . . . . . 933, 1015
720, 721, 725, 728, 729, 737, 747,	\prop_gput:Nnn . . . . 181, 451, 895, 907
752, 757, 759, 760, 764, 767, 768, 776	\prop_gset_eq:NN . . . . . 176
\_pdf_backend_ThisPage_gpush:n . .	\prop_if_empty:NTF . . . . .
. . . . . 4, 121, 174, 250, 293, 332, 357	. . . . . 416, 611, 808, 964,
\_pdf_backend_ThisPage_gput:nn . .	969, 974, 979, 1044, 1049, 1054, 1059
. . . . . 4, 121, 147, 233, 284, 323, 354	\prop_if_exist:NTF . . . . . 177, 819
\g_pdf_backend_thispage_-	\prop_map_function:NN . . . . 186, 824
shipout_tl . . . . . 4	\prop_map_inline:Nn . . . . . 179
\l_pdf_backend_tmpa_box . . . . .	\prop_new:N . . . . . 26, 888
. . . . . 25, 930, 939, 942,	
945, 985, 1012, 1020, 1023, 1026,	
1065, 1177, 1184, 1185, 1186, 1187	
\l_pdf_backend_xform_bool . 483,	
629, 653, 709, 748, 932, 1014, 1107	
\_pdf_backend_xform_if_exist:n . .	
. . . . . 1198, 1204	
\_pdf_backend_xform_new:nnnn . . .	
. . . . . 923, 924, 1006, 1094, 1192	
\_pdf_backend_xform_ref:n . . . . .	
923, 998, 1080, 1127, 1170, 1181, 1194	
\_pdf_backend_xform_use:n . . . . .	
. . . . . 923, 991, 1071, 1175, 1193	

  

	<b>R</b>
ref commands:	
\ref_label:nn . . . . 4, 9, 152, 720, 759	
\ref_value:nn . . . . 5, 14, 160, 728, 767	
\relax . . . . . 64	
\RequirePackage . . . . . 3	
	<b>S</b>
scan commands:	
\scan_stop: . . . . . 995, 1077	
seq commands:	
\seq_gput_right:Nn . . . . . 898	

<code>\seq_new:N</code> . . . . .	887	<code>\tex_luaescapestring:D</code> . . . . .	194
<code>\special</code> . . . . .	427, 429	<code>\tex_pdfextension:D</code> . . . . .	849
str commands:		<code>\tex_pdflastxform:D</code> . . . . .	988, 1068
<code>\str_convert_pdfname:n</code> . . . . .	452	<code>\tex_pdfnames:D</code> . . . . .	842
sys commands:		<code>\tex_pdfpageattr:D</code> . . . . .	126
<code>\sys_if_engine luatex:TF</code> . . . . .	94	<code>\tex_pdfpageresources:D</code> . . . . .	817
<b>T</b>			
TeX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:			
<code>\bsphack</code> . . . . .	8	<code>\tex_pdfpagesattr:D</code> . . . . .	89
<code>\esphack</code> . . . . .	10	<code>\tex_pdfrefxform:D</code> . . . . .	993, 1073
<code>\@kernel@after@shipout@background</code> . . . . .	62, 65	<code>\tex_pdfxform:D</code> . . . . .	947, 1028
<code>\@kernel@after@shipout@lastpage</code> .	69	<code>\tex_special:D</code> . . . . .	108, 263, 302
<code>\@kernel@before@shipout@background</code> . . . . .	64	<code>\tex_the:D</code> . . . . .	939, 942, 945, 1020, 1023, 1026, 1112, 1115, 1118
<code>\g@addto@macro</code> . . . . .	64, 65, 69	<code>\tex_unexpanded:D</code> . . . . .	194
<code>\zref@extractdefault</code> . . . . .	156, 724, 763	tl commands:	
<code>\zref@labelbylist</code> . . . . .	151, 717, 756	<code>\c_space_tl</code> . . . . .	544, 552, 563, 571, 628, 652, 706, 745, 899, 1128, 1129, 1130
tex commands:		<code>\tl_const:Nn</code> . . . . .	937, 940, 943, 1018, 1021, 1024, 1110, 1113, 1116
<code>\tex_directlua:D</code> . . . . .	98, 209, 223, 380, 382	<code>\tl_if_exist:NTF</code> . . . . .	62
<code>\tex_global:D</code> . . . . .	89, 126, 817	<code>\tl_new:N</code> . . . . .	27
<code>\tex_immediate:D</code> . . . . .	947, 1028	<code>\tl_set:Nn</code> . . . . .	154, 722, 761
<code>\tex_latelua:D</code> . . . . .		<code>\tl_to_str:n</code> . . . . .	
. . . . .	199, 235, 252, 397, 398, 637, 661	. . . . .	938, 941, 944, 987, 994, 1000, 1019, 1022, 1025, 1067, 1075, 1081, 1102, 1111, 1114, 1117, 1172, 1200