

# The l3backend-testphase package

## Additional backend PDF features

### L<sup>A</sup>T<sub>E</sub>X PDF management testphase bundle

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

Version 0.95d, released 2021-05-14

## 1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2021-05-14}{}
4   {LaTeX~PDF~management~testphase~bundle~backend~support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2021-05-14}{}
8   {LaTeX~PDF~management~testphase~bundle~backend~support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2021-05-14}{}
12   {LaTeX~PDF~management~testphase~bundle~backend~support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2021-05-14}{}
16   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2021-05-14}{}
20   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2021-05-14}{}
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](mailto: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>
44 <*dvipdfmx | xdvipdfmx>
45 % avoid that destinations names are optimized.
46 % is this still needed??
47 % see https://tug.org/pipermail/dvipdfmx/2019-May/000002.html
48 \__kernel_backend_literal:x { dvipdfmx:config~C~ 0x0010 }
49 </dvipdfmx | xdvipdfmx>

```

```

\g__pdf_tmpa_prop Some scratch variables
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box
50 <*drivers>
51 \prop_new:N \g__pdf_tmpa_prop
52 \tl_new:N \l__pdf_tmpa_tl
53 \box_new:N \l__pdf_backend_tmpa_box
54 </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
55 <*drivers>
56 \int_new:N \g__pdf_backend_resourceid_int
57 \int_new:N \g__pdf_backend_name_int
58 \int_new:N \g__pdf_backend_page_int
59 </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.

```

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

```

### 1.3 Converting unicode strings to a pdfname

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

```
63 <*pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
64 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
65 {
66   / \str_convert_pdfname:e { \text_expand:n { #1 } }
67 }
68 </pdftex | luatex | dvipdfmx | xdvipdfmx | dvisvgm>
69 <*dvips>
70 \cs_new:Npn \__kernel_pdf_name_from_unicode_e:n #1
71 {
72   ~ ( \text_expand:n { #1 } ) ~ cvn
73 }
74 </dvips>
```

### 1.4 Hooks

#### 1.4.1 Add the “end run” hooks

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

```
75 <*pdftex | luatex>
76 % put in \@kernel@after@enddocument@afterlastpage
77 \tl_gput_right:Nn \@kernel@after@enddocument@afterlastpage
78 {
79   \g__kernel_pdfmanagement_end_run_code_tl
80 }
81 </pdftex | luatex>
82 <*dvipdfmx | xdvipdfmx>
83 % put in \@kernel@after@shipout@lastpage
84 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
85 {
86   \g__kernel_pdfmanagement_end_run_code_tl
87 }
88 </dvipdfmx | xdvipdfmx>
89 <*dvips>
90 % put in \@kernel@after@shipout@lastpage
91 \tl_gput_right:Nn \@kernel@after@shipout@lastpage
92 {
93   \g__kernel_pdfmanagement_end_run_code_tl
94 }
95 </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 ...

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

```

101     {
102       \g__kernel_pdfmanagement_thispage_shipout_code_tl
103     }
104     \tl_gput_left:Nn\@kernel@after@shipout@lastpage
105     {
106       \g__kernel_pdfmanagement_lastpage_shipout_code_tl
107     }
108   }
109   {
110     \hook_gput_code:nnn{shipout/background}{pdf}
111     {
112       \g__kernel_pdfmanagement_thispage_shipout_code_tl
113     }
114     \hook_gput_code:nnn {shipout/lastpage} {pdf}
115     {
116       \g__kernel_pdfmanagement_lastpage_shipout_code_tl
117     }
118   }
119
120 </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.

```

121 <*pdftex>
122 \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
123 {
124   \tex_global:D \tex_pdfpagesattr:D { #1 }
125 }
126 </pdftex>
127 <*luatex>
128 %luatex: does it in lua
129 \sys_if_engine luatex:T
130 {
131   \cs_new_protected:Npn \_pdf_backend_Pages_primitive:n #1
132   {
133     \tex_directlua:D
134     {
135       pdf.setpagesattributes( \_pdf_backend_luastring:n { #1 } )
136     }
137   }
138 }
139 </luatex>
140 <*dvips>
141 \cs_new_protected:Npx \_pdf_backend_Pages_primitive:n #1
142 {
143   \tex_special:D{ps:~[#1~/PAGES-pdfmark} %]
144 }
145 </dvips>
146 <*dvipdfmx | xdvipdfmx>

```

```

147 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
148 {
149   \__pdf_backend:n{put~@pages~<<#1>>}
150 }
151 </dvipdfmx | xdvipdfmx>
152 <*dvisvgm>
153 \cs_new_protected:Npn \__pdf_backend_Pages_primitive:n #1
154 {}
155 </dvisvgm>

```

(End definition for \\_\_pdf\_backend\_Pages\_primitive:n.)

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

\\_\_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.

```

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

```

```

187 %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
188 \__pdf_backend_ref_label:en { l3pdf\int_use:N\g__pdf_backend_resourceid_int }{abspage}
189 \tl_set:Nx \l__pdf_tmpa_tl
190 {
191   %\zref@extractdefault
192   % {l3pdf\int_use:N\g__pdf_backend_resourceid_int}
193   % {pdf@abspage}
194   % {0}
195   % \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
196   \__pdf_backend_ref_value:en {l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
197 }
198 \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
199 {
200   \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl}
201 }
202 %backend_Page has no handler.
203 \pdfdict_gput:nnn {g__pdf_Core/backend_Page\l__pdf_tmpa_tl}{ #1 }{ #2 }
204 }
205 %the code to push the values, used in shipout
206 %merges the two props and then fills the register in pdflatex
207 %merges the two tables and then fills (in lua) in luatex
208 %issues the values stored in the global prop with dvi
209 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
210 {
211   \prop_gset_eq:Nc \g__pdf_tmpa_prop { \__kernel_pdfdict_name:n { g__pdf_Core/Page } }
212   \prop_if_exist:cT { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
213   {
214     \prop_map_inline:cn { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1 } }
215     {
216       \prop_gput:Nnn \g__pdf_tmpa_prop { ##1 }{ ##2 }
217     }
218   }
219   \exp_args:Nx \__pdf_backend_Page_primitive:n
220   {
221     \prop_map_function:NN \g__pdf_tmpa_prop \pdfdict_item:ne
222   }
223 }
224 </pdftex>
225 <*luatex>
226 % do we need to use some escaping for the values?????
227 \cs_new:Npn \__pdf_backend_luastring:n #1
228 {
229   "\tex_luaescapestring:D { \tex_unexpanded:D { #1 } }"
230 }
231 %not used, only there for consistency
232 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
233 {
234   \tex_latelua:D
235   {
236     pdf.setpageattributes(\__pdf_backend_luastring:n { #1 })
237   }
238 }
239 % the command to store default values.
240 % Uses a prop with pdflatex + dvi,

```

```

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

```

```

295 %the primitive
296 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
297 {
298   \tex_special:D{pdf:-put~@thispage-<<#1>>}
299 }
300 % the command to store default values.
301 % Uses a prop with pdflatex + dvi,
302 % sets a lua table with luatex
303 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
304 {
305   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
306 }
307 % the command to remove a default value.
308 % Uses a prop with pdflatex + dvi,
309 % changes a lua table with luatex
310 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
311 {
312   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
313 }
314 % the command used in the document.
315 % direct call of the primitive special with dvips/dvipdfmx
316 % \latelua: fill a page related table with luatex, merge it with the page
317 % table and push it directly
318 % write to aux and store in prop with pdflatex
319 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
320 {
321   \__pdf_backend_Page_primitive:n { /#1~#2 }
322 }
323 %the code to push the values, used in shipout
324 %merges the two props and then fills the register in pdflatex
325 %merges the two tables (the one is probably still empty)
326 % and then fills (in lua) in luatex
327 %issues the values stored in the global prop with dvi
328 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
329 {
330   \exp_args:Nx \__pdf_backend_Page_primitive:n
331     { \pdfdict_use:n { g__pdf_Core/Page} }
332 }
333 </dvipdfmx | xdvipdfmx>
334 <*dvips>
335 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
336 {
337   \tex_special:D{ps:-[ThisPage]<<#1>>~/PUT~pdfmark} %]
338 }
339 % the command to store default values.
340 % Uses a prop with pdflatex + dvi,
341 % sets a lua table with luatex
342 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
343 {
344   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
345 }
346 % the command to remove a default value.
347 % Uses a prop with pdflatex + dvi,
348 % changes a lua table with luatex

```



```

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 % direct call of the primitive special with dvips/dvipdfmx
355 % \lualatex: fill a page related table with lualatex, merge it with the page
356 % table and push it directly
357 % write to aux and store in prop with pdflatex
358 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
359 {
360   \__pdf_backend_Page_primitive:n { /#1~#2 }
361 }
362 %the code to push the values, used in shipout
363 %merges the two props and then fills the register in pdflatex
364 %merges the two tables (the one is probably still empty)
365 %and then fills (in lua) in luatex
366 %issues the values stored in the global prop with dvi
367 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
368 {
369   \exp_args:Nx \__pdf_backend_Page_primitive:n
370     { \pdfdict_use:n { g__pdf_Core/Page} }
371 }
372 </dvips>
373 <*dvisvgm>
374 % mostly only dummies ...
375 \cs_new_protected:Npn \__pdf_backend_Page_primitive:n #1
376 {}
377 % Uses a prop with pdflatex + dvi,
378 \cs_new_protected:Npn \__pdf_backend_Page_gput:nn #1 #2
379 {
380   \pdfdict_gput:nnn {g__pdf_Core/Page}{ #1 }{ #2 }
381 }
382 % the command to remove a default value.
383 % Uses a prop with pdflatex + dvi,
384 \cs_new_protected:Npn \__pdf_backend_Page_gremove:n #1
385 {
386   \pdfdict_gremove:nn {g__pdf_Core/Page}{ #1 }
387 }
388 % the command used in the document.
389 \cs_new_protected:Npn \__pdf_backend_ThisPage_gput:nn #1 #2
390 {}
391 %the code to push the values, used in shipout
392 \cs_new_protected:Npn \__pdf_backend_ThisPage_gpush:n #1
393 {}
394 </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.

`\_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:
395 % backend commands the command to fill the register
396 % and to push the values.
397 %
398 % The names are quite often needed
399 % a similar list is now in l3pdfmanagement. Perhaps it should be merged.
400 <*drivers>
401 \clist_const:Nn \c__pdf_backend_PageResources_clist
402 {
403     ExtGState,
404     ColorSpace,
405     Pattern,
406     Shading,
407 }
408 </drivers>
409 % pdftex and luatex
410 <*pdftex | luatex>
411 %create the backend objects:
412 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
413 {
414     \_pdf_backend_object_new:nn {_pdf/Page/Resources/#1} {dict}
415     \cs_if_exist:NT \tex_directlua:D
416     {
417         \tex_directlua:D
418         {
419             ltx.__pdf.object["_pdf/Page/Resources/#1"]
420             =
421             "\_pdf_backend_object_ref:n{_pdf/Page/Resources/#1}"
422         }
423     }
424 }
425 </pdftex | luatex>
426 <*luatex>
427 %values are only stored in a prop and will be output at end document.
428 \cs_new_protected:Npn \_pdf_backend_PageResources_gput:nnn #1 #2 #3
429 {
430     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
431     % luatex must also trigger the lua side
432     \tex_latelua:D{ltx.__pdf.Page.Resources.#1=true}
433     \tex_latelua:D
434     {
435         ltx.pdf.Page_Resources_gpush(tex.count["g_shipout_readonly_int"])
436     }
437 }
438 </luatex>
439 <*pdftex>
440 \cs_new_protected:Npn \_pdf_backend_PageResources_gput:nnn #1 #2 #3
441 {
442     \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
```

```

443     }
444 </pdftex>
445 <*pdftex|luatex>
446 %code for end of document code
447 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush:
448 {
449     \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
450     {
451         \prop_if_empty:cF
452         { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/##1} }
453         {
454             \__pdf_backend_object_write:nx
455             { __pdf/Page/Resources/##1 }
456             { \pdfdict_use:n { g__pdf_Core/Page/Resources/##1} }
457         }
458     }
459 }
460 </pdftex|luatex>
461 % xdvipdfmx
462 % \special{pdf:pageresources<<#1>>} doesn't work correctly with object names ...
463 % https://tug.org/pipermail/dvipdfmx/2019-August/000021.html,
464 % so we use \special{pdf:put @resources}
465 % this must be issued on every page!
466 <*dvipdfmx|xdvipdfmx>
467 %objects should not only be created but also "initialized"
468 % initialization should be done before anyone tries to write
469 % so we add rules for the backend.
470 <xdvipdfmx>\hook_gset_rule:nnnn{shipout/firstpage}{l3backend-xetex}{after}{pdf}
471 <dvipdfmx>\hook_gset_rule:nnnn{shipout/firstpage}{l3backend-dvipdfmx}{after}{pdf}
472 %
473 \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
474 {
475     \__pdf_backend_object_new:nn { __pdf/Page/Resources/#1 } { dict }
476     \hook_gput_code:nnn{shipout/firstpage}{pdf}{\__pdf_backend_object_write:nn { __pdf/Page/
477 }
478 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1
479 {
480     \__pdf_backend:n {put~@resources~<<#1>>}
481 }
482 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
483 {
484     % this is not used for output, but there is a test if the resource is empty
485     \exp_args:Nnx
486     \prop_gput:cnn { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
487     { \str_convert_pdfname:n {#2} }{ #3 }
488     %objects are not filled with \pdf_object_write as this is not additive!
489     \__pdf_backend:x
490     {
491         put~\__pdf_backend_object_ref:n {__pdf/Page/Resources/#1}<<#2~#3>>
492     }
493 }
494
495 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
496 </dvipdfmx|xdvipdfmx>

```

```

497 <*dvips>
498 % dvips unneeded, or no-op
499 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
500 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
501 { %only for the show command TEST!!
502   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
503 }
504 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
505 </dvips>
506 <*dvisvgm>
507 % dvips unneeded, or no-op
508 \cs_new_protected:Npn \__pdf_backend_PageResources:n #1 {}
509 \cs_new_protected:Npn \__pdf_backend_PageResources_gput:nnn #1 #2 #3
510 { %only for the show command TEST!!
511   \pdfdict_gput:nnn {g__pdf_Core/Page/Resources/#1} { #2 }{ #3 }
512 }
513 \cs_new_protected:Npn \__pdf_backend_PageResources_obj_gpush: {}
514 </dvisvgm>

```

(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
515 % pdftex and luatex (and perhaps dvips ...) need to know if there are in a
516 % xform stream ...
517 <*drivers>
518 \bool_new:N \l__pdf_backend_xform_bool
519 </drivers>
520 <*dvips>
521 % dvips is easy: create an object, and reference it in the bdc
522 % ghostscript will then automatically replace it by a name
523 % and add the name to the /Properties dict
524 % special variant von accsupp
525 % https://chat.stackexchange.com/transcript/message/50831812#50831812
526 %
527 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
528 {
529   \__pdf_backend_pdfmark:x{/#1~<<#2>>~/BDC}
530 }
531 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
532 {
533   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_ref:n{#2}~/BDC}
534 }
535 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
536 {
537   \__pdf_backend_pdfmark:x{/#1~\__pdf_backend_object_last:~/BDC}
538 }
539 \cs_set_protected:Npn \__pdf_backend_emc:
540 {

```

```

541     \__pdf_backend_pdfmark:n{/EMC} %
542   }
543   \cs_set_protected:Npn \__pdf_backend_bmc:n #1
544   {
545     \__pdf_backend_pdfmark:n{/#1~/BMC} %
546   }
547   \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
548
549   </dvips>
550   <*dvisvgm>
551   % dvisvgm should do nothing
552   %
553   \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2 % #1 eg. Span, #2: dict_content
554   {}
555   \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
556   {}
557   \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span,
558   {}
559   \cs_set_protected:Npn \__pdf_backend_emc:
560   {}
561   \cs_set_protected:Npn \__pdf_backend_bmc:n #1
562   {}
563   \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
564
565   </dvisvgm>
566
567   % xetex has to create the entries in the /Properties manually
568   % (like the other backends)
569   % use pdfbase special
570   % https://chat.stackexchange.com/transcript/message/50832016#50832016
571   % the property is added to xform resources automatically,
572   % no need to worry about it.
573   <*dvipdfmx | xdvipdfmx>
574   \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
575   {
576     \int_gincr:N \g__pdf_backend_name_int
577     \__kernel_backend_literal:x
578     {
579       pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
580     }
581     \__kernel_backend_literal:x
582     {
583       pdf:put~@resources~
584       <<
585       /Properties~
586       <<
587       /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
588       \__pdf_backend_object_ref:n { #2 }
589       >>
590     }
591   }
592   }
593   \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1 % #1 eg. Span
594   {

```

```

595 \int_gincr:N \g__pdf_backend_name_int
596 \__kernel_backend_literal:x
597 {
598   pdf:code~/#1/l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC
599 }
600 \__kernel_backend_literal:x
601 {
602   pdf:put~@resources~
603   <<
604     /Properties~
605     <<
606       /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl
607       \__pdf_backend_object_last:
608     >>
609   >>
610 }
611 }
612 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
613 {
614   \__kernel_backend_literal:n {pdf:code~/#1~BMC} %pdfbase
615 }
616
617 %this require management
618 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
619 {
620   \pdf_object_unnamed_write:nn { dict }{ #2 }
621   \__pdf_backend_bdcobject:n { #1 }
622 }
623
624 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
625 {
626   \__kernel_backend_literal:n {pdf:code~ /#1~<<#2>>~BDC }
627 }
628
629 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
630 {
631   \bool_if:NTF \g__pdfmanagement_active_bool
632     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
633     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
634     \__pdf_backend_bdc:nn {#1}{#2}
635 }
636 \cs_set_protected:Npn \__pdf_backend_emc:
637 {
638   \__kernel_backend_literal:n {pdf:code~EMC} %pdfbase
639 }
640 % properties are handled automatically, but the other resources should be added
641 % at shipout
642 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
643 {
644   \clist_map_inline:Nn \c__pdf_backend_PageResources_clist
645   {
646     \prop_if_empty:cF { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/##1} }
647     {
648       \__kernel_backend_literal:x

```

```

649         {
650             pdf:put~@resources~
651             <</##1~\__pdf_backend_object_ref:n {__pdf/Page/Resources/##1}>>
652         }
653     }
654 }
655 }
656 </dvipdfmx | xdvipdfmx>
657 % luatex + pdftex
658 (*luatex)
659 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
660 {
661     \int_gincr:N \g__pdf_backend_name_int
662     \exp_args:Nx\__kernel_backend_literal_page:n
663     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
664     \bool_if:NTF \l__pdf_backend_xform_bool
665     {
666         \exp_args:Nnx\pdfdict_gput:nnn
667         { g__pdf_Core/Xform/Resources/Properties }
668         { l3pdf\int_use:N\g__pdf_backend_name_int }
669         { \__pdf_backend_object_ref:n { #2 } }
670     }
671     {
672         \exp_args:Nx \tex_latelua:D
673         {
674             ltx.pdf.Page_Resources_Properties_gput
675             (
676                 tex.count["g_shipout_readonly_int"],
677                 "l3pdf\int_use:N\g__pdf_backend_name_int",
678                 "\__pdf_backend_object_ref:n { #2 }"
679             )
680         }
681     }
682 }
683 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
684 {
685     \int_gincr:N \g__pdf_backend_name_int
686     \exp_args:Nx\__kernel_backend_literal_page:n
687     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
688     \bool_if:NTF \l__pdf_backend_xform_bool
689     {
690         \exp_args:Nnx\pdfdict_gput:nnn %no handler needed
691         { g__pdf_Core/Xform/Resources/Properties }
692         { l3pdf\int_use:N\g__pdf_backend_name_int }
693         { \__pdf_backend_object_last: }
694     }
695     {
696         \exp_args:Nx \tex_latelua:D
697         {
698             ltx.pdf.Page_Resources_Properties_gput
699             (
700                 tex.count["g_shipout_readonly_int"],
701                 "l3pdf\int_use:N\g__pdf_backend_name_int",
702                 "\__pdf_backend_object_last:"

```

```

703         )
704     }
705 }
706 }
707 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
708 {
709     \__kernel_backend_literal_page:n { /#1~BMC }
710 }
711 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
712 {
713     \pdf_object_unnamed_write:nn { dict } { #2 }
714     \__pdf_backend_bdcobject:n { #1 }
715 }
716 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
717 {
718     \__kernel_backend_literal_page:n { /#1-<<#2>>~BDC }
719 }
720 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
721 {
722     \bool_if:NTF \g__pdfmanagement_active_bool
723     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
724     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
725     \__pdf_backend_bdc:nn {#1}{#2}
726 }
727 \cs_set_protected:Npn \__pdf_backend_emc:
728 {
729     \__kernel_backend_literal_page:n { EMC }
730 }
731
732 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1 {}
733 </luatex>
734 <*pdfTeX>
735 % pdfLaTeX is the most complicated as it has to go through the aux ...
736 % the push command is extended to take other resources too
737 \cs_set_protected:Npn \__pdf_backend_bdcobject:nn #1 #2 % #1 eg. Span, #2: object name
738 {
739     \int_gincr:N \g__pdf_backend_name_int
740     \exp_args:Nx\__kernel_backend_literal_page:n
741     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
742     % code to set the property ....
743     \int_gincr:N\g__pdf_backend_resourceid_int
744     \bool_if:NTF \l__pdf_backend_xform_bool
745     {
746         \exp_args:Nnxx\pdfdict_gput:nnn %no handler needed
747         { g__pdf_Core/Xform/Resources/Properties }
748         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
749         { \__pdf_backend_object_ref:n { #2 } }
750     }
751     {
752         %\zref@labelbylist
753         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
754         { l3pdf }
755         \ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
756         \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}

```



```

757 \tl_set:Nx \l__pdf_tmpa_tl
758 {
759     %\zref@extractdefault
760     { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
761     {pdf@abspage}
762     {0}
763     %\ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
764     \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
765 }
766 \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
767 {
768     \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
769 }
770 \exp_args:Nnxx\pdfdict_gput:nnn
771 { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
772 { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
773 { \__pdf_backend_object_ref:n{#2} }
774 }
775 }
776 \cs_set_protected:Npn \__pdf_backend_bdcobject:n #1% #1 eg. Span
777 {
778     \int_gincr:N \g__pdf_backend_name_int
779     \exp_args:Nx\__kernel_backend_literal_page:n
780     { /#1 ~ /l3pdf\int_use:N\g__pdf_backend_name_int\c_space_tl BDC }
781     % code to set the property ...
782     \int_gincr:N\g__pdf_backend_resourceid_int
783     \bool_if:NTF \l__pdf_backend_xform_bool
784     {
785         \exp_args:Nnxx\pdfdict_gput:nnn
786         { g__pdf_Core/Xform/Resources/Properties }
787         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
788         { \__pdf_backend_object_last: }
789     }
790     {
791         %\zref@labelbylist
792         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
793         { l3pdf }
794         %\ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
795         \__pdf_backend_ref_label:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
796         \tl_set:Nx \l__pdf_tmpa_tl
797         {
798             %\zref@extractdefault
799             { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
800             {pdf@abspage}
801             {0}
802             % \ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
803             \__pdf_backend_ref_value:en{l3pdf\int_use:N\g__pdf_backend_resourceid_int}{abspage}
804         }
805         \pdfdict_if_exist:nF { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties
806         {
807             \pdfdict_new:n { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }
808         }
809         \exp_args:Nnxx\pdfdict_gput:nnn
810         { g__pdf_Core/backend_Page\l__pdf_tmpa_tl/Resources/Properties }

```

```

811         { l3pdf\int_use:N\g__pdf_backend_resourceid_int }
812         { \__pdf_backend_object_last: }
813         %\pdfdict_show:n { g_backend_Page\l__pdf_tmpa_tl/Resources/Properties }
814     }
815 }
816 \cs_set_protected:Npn \__pdf_backend_bmc:n #1
817 {
818     \__kernel_backend_literal_page:n { /#1~BMC }
819 }
820 \cs_set_protected:Npn \__pdf_backend_bdc_contobj:nn #1 #2
821 {
822     \pdf_object_unnamed_write:nn { dict } { #2 }
823     \__pdf_backend_bdcobject:n { #1 }
824 }
825 \cs_set_protected:Npn \__pdf_backend_bdc_contstream:nn #1 #2
826 {
827     \__kernel_backend_literal_page:n { /#1~<<#2>>~BDC }
828 }
829 \cs_set_protected:Npn \__pdf_backend_bdc:nn #1 #2
830 {
831     \bool_if:NTF \g__pdfmanagement_active_bool
832     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contobj:nn}
833     {\cs_gset_eq:NN \__pdf_backend_bdc:nn \__pdf_backend_bdc_contstream:nn}
834     \__pdf_backend_bdc:nn {#1}{#2}
835 }
836 \cs_set_protected:Npn \__pdf_backend_emc:
837 {
838     \__kernel_backend_literal_page:n { EMC }
839 }
840
841 \cs_new:Npn \__pdf_backend_PageResources_gpush_aux:n #1 %#1 ExtGState etc
842 {
843     \prop_if_empty:cF
844     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/#1} }
845     {
846         \pdfdict_item:ne { #1 }{ \pdf_object_ref:n {__pdf/Page/Resources/#1}}
847     }
848 }
849
850 \cs_new_protected:Npn \__pdf_backend_PageResources_gpush:n #1
851 {
852     \exp_args:NNx \tex_global:D \tex_pdfpageresources:D
853     {
854         \prop_if_exist:cT
855         { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Properties } }
856         {
857             /Properties~
858             <<
859             \prop_map_function:cN
860             { \__kernel_pdfdict_name:n { g__pdf_Core/backend_Page#1/Resources/Property
861             \pdfdict_item:ne
862             >>
863         }
864         %% add ExtGState etc

```

```

865         \clist_map_function:NN
866         \c__pdf_backend_PageResources_clist
867         \__pdf_backend_PageResources_gpush_aux:n
868     }
869 }
870
871 </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` 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.

```

872 % pdflatex
873 <*pdftex>
874 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
875 {
876     \pdf_object_unnamed_write:nn {dict} {/Names [#1] }
877     \tex_pdfnames:D {/EmbeddedFiles~\pdf_object_ref_last:}
878 }
879 </pdftex>
880 <*luatex>
881 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
882 {
883     \pdf_object_unnamed_write:nn {dict} {/Names [#1] }
884     \tex_pdfextension:D-names~{/EmbeddedFiles~\pdf_object_ref_last: }
885 }
886 </luatex>
887 <*dvi.pdfmx | xdvipdfmx>
888 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 %array content
889 {
890     \pdf_object_unnamed_write:nn {dict} { /Names [#1] }
891     %n or x?
892     \__pdf_backend:x {put~@names~<</EmbeddedFiles~\pdf_object_ref_last: >>}
893 }
894 </dvi.pdfmx | xdvipdfmx>
895
896 %dvips: noop
897 <*dvips>
898 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
899 </dvips>
900 %dvisvgm: noop
901 <*dvisvgm>
902 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_gpush:n #1 {}
903 </dvisvgm>
904

```

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:`

```

905 <*drivers>
906 \int_new:N \g__pdf_backend_EmbeddedFiles_int
907 \cs_new:Npn \__pdf_backend_EmbeddedFiles_name:
908 {
909   (
910     l3ef
911     \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {10}
912     {0}
913     \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {100}
914     {0}
915     \int_compare:nNnT {\g__pdf_backend_EmbeddedFiles_int} < {1000}
916     {0}
917     \int_use:N \g__pdf_backend_EmbeddedFiles_int
918   )
919 }
920 </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.)*

```

921 <*drivers>
922 \seq_new:N \g__pdf_backend_EmbeddedFiles_seq
923 \prop_new:N \g__pdf_backend_EmbeddedFiles_named_prop
924 </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.

```

925 <*pdftex | luatex | dvipdfmx | xdvipdfmx>
926 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
927   {%#1 object ref
928   {
929     \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
930     \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
931       { #1 }
932     { \__pdf_backend_EmbeddedFiles_name: }
933     \seq_gput_right:Nx \g__pdf_backend_EmbeddedFiles_seq
934       { \__pdf_backend_EmbeddedFiles_name: \c_space_tl #1 }
935   }
936
937 </pdftex | luatex | dvipdfmx | xdvipdfmx>
938 <*dvips>

```

```

939 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
940 {
941     \int_gincr:N \g__pdf_backend_EmbeddedFiles_int
942     \prop_gput:Nnx \g__pdf_backend_EmbeddedFiles_named_prop
943     { #1 }
944     { \__pdf_backend_EmbeddedFiles_name: }
945     \__pdf_backend_pdfmark:x
946     {
947         /Name~\__pdf_backend_EmbeddedFiles_name:~
948         /FS~#1~
949         /EMBED
950     }
951 }
952 </dvips>
953 <*dvisvgm>
954 %no op. Or is there any sensible use for it?
955 \cs_new_protected:Npn \__pdf_backend_NamesEmbeddedFiles_add:n #1
956 {}
957 </dvisvgm>

```

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

## 1.8.2 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      958 <*pdfTeX>
\__pdf_backend_xform_ref:n      959 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
                                960 % #1 name
                                961 % #2 attributes
                                962 % #3 resources
                                963 % #4 content, not necessarily a box!
                                964 {
                                965     \hbox_set:Nn \l__pdf_backend_tmpa_box
                                966     {
                                967         \bool_set_true:N \l__pdf_backend_xform_bool
                                968         \prop_gclear:c {\__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
                                969         #4
                                970     }
                                971     %store the dimensions
                                972     \tl_const:cx
                                973     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
                                974     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
                                975     \tl_const:cx
                                976     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
                                977     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
                                978     \tl_const:cx
                                979     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
                                980     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
                                981     %% do we need to test if #2 and #3 are empty??

```

```

982 \tex_immediate:D \tex_pdfxform:D
983 ~ attr ~ { #2 }
984 %% which other resources should be default? Is an argument actually needed?
985 ~ resources ~
986 {
987   #3
988   \int_compare:nNnT
989     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
990     >
991     { 0 }
992     {
993       /Properties~
994       <<
995       \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
996       >>
997     }
998
999   \prop_if_empty:cF
1000     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1001     {
1002       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1003     }
1004   \prop_if_empty:cF
1005     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1006     {
1007       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1008     }
1009   \prop_if_empty:cF
1010     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1011     {
1012       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1013     }
1014   \prop_if_empty:cF
1015     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1016     {
1017       /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1018     }
1019   }
1020   \l__pdf_backend_tmpa_box
1021   \int_const:cn
1022     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1023     { \tex_pdflastxform:D }
1024 }
1025
1026 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1027 {
1028   \tex_pdfrefxform:D
1029   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1030   \scan_stop:
1031 }
1032
1033 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1034 {
1035   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R

```

```

1036 }
1037 </pdfTeX>
1038 <*luatex>
1039 %luatex
1040 %nearly identical but not completely ...
1041 \cs_new_protected:Npn \l__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1042 % #1 name
1043 % #2 attributes
1044 % #3 resources
1045 % #4 content, not necessarily a box!
1046 {
1047   \hbox_set:Nn \l__pdf_backend_tmpa_box
1048   {
1049     \bool_set_true:N \l__pdf_backend_xform_bool
1050     \prop_gclear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
1051     #4
1052   }
1053   \tl_const:cx
1054   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1055   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1056   \tl_const:cx
1057   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1058   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1059   \tl_const:cx
1060   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1061   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1062   %% do we need to test if #2 and #3 are empty??
1063   \tex_immediate:D \tex_pdfxform:D
1064   ~ attr ~ { #2 }
1065   %% which resources should be default? Is an argument actually needed?
1066   ~ resources ~
1067   {
1068     #3
1069     \int_compare:nNnT
1070     { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
1071     >
1072     { 0 }
1073     {
1074       /Properties~
1075       <<
1076       \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1077       >>
1078     }
1079     \prop_if_empty:cF
1080     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1081     {
1082       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1083     }
1084     \prop_if_empty:cF
1085     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1086     {
1087       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1088     }
1089     \prop_if_empty:cF

```

```

1090         { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1091         {
1092             /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1093         }
1094     \prop_if_empty:cF
1095     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1096     {
1097         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1098     }
1099 }
1100 \l__pdf_backend_tmpa_box
1101 \int_const:cn
1102 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1103 { \tex_pdflastxform:D }
1104 }
1105
1106 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1107 {
1108     \tex_pdfrefxform:D \int_use:c
1109     {
1110         c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1111     }
1112     \scan_stop:
1113 }
1114
1115 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1116 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1117
1118 </luatex>
1119 <*dviPDFmx | xdvipdfmx>
1120 % xetex
1121 % it needs a bit testing if it really works to set the box to 0 before the special ...
1122 % does it disturb viewing the xobject?
1123 % what happens with the resources (bdc)? (should work as they are specials too)
1124 % xetex requires that the special is in horizontal mode. This means it affects
1125 % typesetting. But we can no delay the whole form code to shipout
1126 % as the object reference and the size is often wanted on the current page.
1127 % so we need to allocate a box - but probably they won't be thousands xform
1128 % in a document so it shouldn't matter.
1129 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1130 % #1 name
1131 % #2 attributes
1132 % #3 resources
1133 % #4 content, not necessarily a box!
1134 {
1135     \int_gincr:N \g__pdf_backend_object_int
1136     \int_const:cn
1137     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1138     { \g__pdf_backend_object_int }
1139     \box_new:c { g__pdf_backend_xform_#1_box }
1140     \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1141     {
1142         \bool_set_true:N \l__pdf_backend_xform_bool
1143         #4

```



```

1144 }
1145 \tl_const:cx
1146 { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1147 { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1148 \tl_const:cx
1149 { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1150 { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1151 \tl_const:cx
1152 { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1153 { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1154 \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1155 \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1156 \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1157 \hook_gput_next_code:nn {shipout/background}
1158 {
1159   \mode_leave_vertical: %needed, the xform disappears without it.
1160   \__pdf_backend:x
1161   {
1162     bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1163     \c_space_tl width ~ \pdfxform_wd:n { #1 }
1164     \c_space_tl height ~ \pdfxform_ht:n { #1 }
1165     \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1166   }
1167   \box_use_drop:c { g__pdf_backend_xform_#1_box }
1168   \__pdf_backend:x {put ~ @resources ~<<#3>> }
1169   \__pdf_backend:x
1170   {
1171     put~ @resources ~
1172     <<
1173       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1174     >>
1175   }
1176   \__pdf_backend:x
1177   {
1178     put~ @resources ~
1179     <<
1180       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1181     >>
1182   }
1183   \__pdf_backend:x
1184   {
1185     put~ @resources ~
1186     <<
1187       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1188     >>
1189   }
1190   \__pdf_backend:x
1191   {
1192     put~ @resources ~
1193     <<
1194       /ColorSpace~
1195       \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1196     >>
1197   }

```

```

1198         \exp_args:Nx
1199         \__pdf_backend:x {exobj ~<<#2>>}
1200     }
1201 }
1202
1203
1204
1205 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1206 {
1207     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1208 }
1209
1210 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1211 {
1212     \hbox_set:Nn \l__pdf_backend_tmpa_box
1213     {
1214         \__pdf_backend:x
1215         {
1216             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1217         }
1218     }
1219     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1220     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1221     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1222     \box_use_drop:N \l__pdf_backend_tmpa_box
1223 }
1224 </dviPDFmx | xdvipdfmx>
1225 <*dvisvgm>
1226 % unclear what it should do!!
1227 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1228 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1229 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1230 </dvisvgm>
1231 <*drivers>
1232 %% all
1233 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1234 {
1235     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1236     { \prg_return_true: }
1237     { \prg_return_false: }
1238 }
1239 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n \__pdf_backend_xform_if_exist:n
1240 { TF , T , F , p }
1241 </drivers>

```

(End definition for \\_\_pdf\_backend\_xform\_new:nnnn, \\_\_pdf\_backend\_xform\_use:n, and \\_\_pdf\_backend\_xform\_ref:n.)

## 1.9 lua code for lualatex

```

1242 <*lua>
1243 ltx= ltx or {}
1244 ltx.__pdf = ltx.__pdf or {}
1245 ltx.__pdf.Page = ltx.__pdf.Page or {}

```

```

1246 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1247 ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}
1248 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1249 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1250 ltx.__pdf.object = ltx.__pdf.object or {}
1251
1252 ltx.pdf= ltx.pdf or {} -- for "public" functions
1253
1254 local __pdf = ltx.__pdf
1255 local pdf = pdf
1256
1257 local function __pdf_backend_Page_gput (name,value)
1258   __pdf.Page.dflt[name]=value
1259 end
1260
1261 local function __pdf_backend_Page_gremove (name)
1262   __pdf.Page.dflt[name]=nil
1263 end
1264
1265 local function __pdf_backend_Page_gclear ()
1266   __pdf.Page.dflt={}
1267 end
1268
1269 local function __pdf_backend_ThisPage_gput (page,name,value)
1270   __pdf.Page[page] = __pdf.Page[page] or {}
1271   __pdf.Page[page][name]=value
1272 end
1273
1274 local function __pdf_backend_ThisPage_gpush (page)
1275   local token=""
1276   local t = {}
1277   local tkeys= {}
1278   for name,value in pairs(__pdf.Page.dflt) do
1279     t[name]=value
1280   end
1281   if __pdf.Page[page] then
1282     for name,value in pairs(__pdf.Page[page]) do
1283       t[name] = value
1284     end
1285   end
1286   -- sort the table to get reliable test files.
1287   for name,value in pairs(t) do
1288     table.insert(tkeys,name)
1289   end
1290   table.sort(tkeys)
1291   for _,name in ipairs(tkeys) do
1292     token = token .. "/"..name.." "..t[name]
1293   end
1294   return token
1295 end
1296
1297 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly"]
1298   __pdf_backend_ThisPage_gput (page,name,value)
1299 end

```

```

1300
1301 function ltx.__pdf.backend_ThisPage_gpush (page)
1302     pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1303 end
1304
1305 function ltx.__pdf.backend_Page_gput (name,value)
1306     __pdf_backend_Page_gput (name,value)
1307 end
1308
1309 function ltx.__pdf.backend_Page_gremove (name)
1310     __pdf_backend_Page_gremove (name)
1311 end
1312
1313 function ltx.__pdf.backend_Page_gclear ()
1314     __pdf_backend_Page_gclear ()
1315 end
1316
1317
1318 local Properties = ltx.__pdf.Page.Resources.Properties
1319 local ResourceList= ltx.__pdf.Page.Resources.List
1320 local function __pdf_backend_PageResources_gpush (page)
1321     local token=""
1322     if Properties[page] then
1323         -- we sort the table, so that the pdf test works
1324         local t = {}
1325         for name,value in pairs (Properties[page]) do
1326             table.insert (t,name)
1327         end
1328         table.sort (t)
1329         for _,name in ipairs(t) do
1330             token = token .. "/"..name.." " .. Properties[page][name]
1331         end
1332         token = "/Properties <<"..token..">>"
1333     end
1334     for i,name in ipairs(ResourceList) do
1335         if ltx.__pdf.Page.Resources[name] then
1336             token = token .. "/"..name.." " ..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
1337         end
1338     end
1339     return token
1340 end
1341
1342 -- the function is public, as I probably need it in tagpdf too ...
1343 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1344     Properties[page] = Properties[page] or {}
1345     Properties[page][name]=value
1346     pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1347 end
1348
1349 function ltx.pdf.Page_Resources_gpush(page)
1350     pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1351 end
1352
1353 function ltx.pdf.object_ref (objname)

```

```

1354 if ltx.__pdf.object[objname] then
1355     local ref= ltx.__pdf.object[objname]
1356     return ref
1357 else
1358     return "false"
1359 end
1360 end
1361 </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:	898, 902, 926, 939, 955, 959, 1026, 1041, 1106, 1129, 1210, 1227, 1228
\bool_if:NTF .....	\cs_new_protected:Npx ..... 141
.... 631, 664, 688, 722, 744, 783, 831	\cs_set_protected:Npn .. 527, 531,
\bool_new:N .....	535, 539, 543, 553, 555, 557, 559,
518	561, 574, 593, 612, 618, 624, 629,
\bool_set_true:N .....	636, 659, 683, 707, 711, 716, 720,
967, 1049, 1142	727, 737, 776, 816, 820, 825, 829, 836
box commands:	
\box_dp:N .....	980, 1061, 1153
\box_ht:N .....	977, 1058, 1150
\box_new:N .....	53, 1139
\box_set_dp:Nn .....	1154, 1221
\box_set_ht:Nn .....	1155, 1220
\box_set_wd:Nn .....	1156, 1219
\box_use_drop:N .....	1167, 1222
\box_wd:N .....	974, 1055, 1147
<b>C</b>	
clist commands:	
\clist_const:Nn .....	401
\clist_map_function:NN .....	865
\clist_map_inline:Nn 412, 449, 473, 644	
cs commands:	
\cs_generate_variant:Nn 29, 30, 41, 42	
\cs_gset_eq:NN .....	632, 633, 723, 724, 832, 833
\cs_if_exist:NTF .....	415
\cs_new:Npn .....	37, 64, 70, 227, 841, 907, 1033, 1115, 1205, 1229
\cs_new_protected:Npn .....	31, 122, 131, 147, 153, 159, 166, 173, 182, 209, 232, 242, 256, 268, 285, 296, 303, 310, 319, 328, 335, 342, 349, 358, 367, 375, 378, 384, 389, 392, 428, 440, 447, 478, 482, 495, 499, 500, 504, 508, 509, 513, 547, 563, 642, 732, 850, 874, 881, 888,
<b>D</b>	
dim commands:	
\c_zero_dim .....	1154, 1155, 1156
\directlua .....	61
<b>E</b>	
exp commands:	
\exp_args:NNx .....	852
\exp_args:Nnx .....	485, 666, 690
\exp_args:Nxxx .....	746, 770, 785, 809
\exp_args:Nx .....	219, 330, 369, 662, 672, 686, 696, 740, 779, 1198
<b>H</b>	
hbox commands:	
\hbox_gset:Nn .....	1140
\hbox_set:Nn .....	965, 1047, 1212
hook commands:	
\hook_gput_code:nmn .....	110, 114, 476
\hook_gput_next_code:nm .....	1157
\hook_gset_rule:nmmn .....	470, 471
<b>I</b>	
int commands:	
\int_compare:nNnTF .....	911, 913, 915, 988, 1069
\int_const:Nn .....	1021, 1101, 1136
\int_gincr:N .....	185, 576, 595, 661, 685, 739, 743, 778, 782, 929, 941, 1135

`\int_if_exist:NTF` ..... 1235  
`\int_new:N` ..... 56, 57, 58, 906  
`\int_use:N` . 186, 187, 188, 192, 195,  
196, 579, 587, 598, 606, 663, 668,  
677, 687, 692, 701, 741, 748, 753,  
755, 756, 760, 763, 764, 772, 780,  
787, 792, 794, 795, 799, 802, 803,  
811, 917, 1029, 1035, 1108, 1116, 1207

### K

kernel internal commands:

`\__kernel_backend_literal:n` . 48,  
577, 581, 596, 600, 614, 626, 638, 648  
`\__kernel_backend_literal_page:n`  
..... 662, 686,  
709, 718, 729, 740, 779, 818, 827, 838  
`\__kernel_pdf_name_from_unicode_-  
e:n` ..... 64, 70  
`\__kernel_pdffdict_name:n` 211, 212,  
214, 452, 486, 646, 844, 855, 860,  
968, 989, 1000, 1005, 1010, 1015,  
1050, 1070, 1080, 1085, 1090, 1095  
`\g__kernel_pdfmanagement_end_-  
run_code_tl` ..... 79, 86, 93  
`\g__kernel_pdfmanagement_-  
lastpage_shipout_code_tl` 106, 116  
`\g__kernel_pdfmanagement_-  
thispage_shipout_code_tl` 102, 112

### L

lualua commands:

`\lualua:` ..... 179, 265, 316, 355

### M

mode commands:

`\mode_leave_vertical:` ..... 1159

### P

pdf commands:

`\pdf_object_ref:n` ..... 846,  
1002, 1007, 1012, 1017, 1082, 1087,  
1092, 1097, 1173, 1180, 1187, 1195  
`\pdf_object_ref_last:` . 877, 884, 892  
`\pdf_object_unnamed_write:nn` ...  
..... 620, 713, 822, 876, 883, 890  
`\pdf_object_write` ..... 488

pdf internal commands:

`\__pdf_backend:n` .....  
. 149, 480, 489, 892, 1160, 1168,  
1169, 1176, 1183, 1190, 1199, 1214  
`\__pdf_backend_bdc:nn` ..... 12, 515  
`\__pdf_backend_bdc_contobj:nn` ...  
..... 618, 632, 711, 723, 820, 832  
`\__pdf_backend_bdc_contstream:nn`  
..... 624, 633, 716, 724, 825, 833  
`\__pdf_backend_bdcobject:n` .. 12, 515  
`\__pdf_backend_bdcobject:nn` . 12, 515  
`\__pdf_backend_bmc:n` ..... 12, 515  
`\__pdf_backend_catalog_gput:nn` .. 19  
`\g__pdf_backend_EmbeddedFiles_-  
int` ..... 905,  
906, 911, 913, 915, 917, 929, 941  
`\__pdf_backend_EmbeddedFiles_-  
name:` ..... 905, 932, 934, 944, 947  
`\g__pdf_backend_EmbeddedFiles_-  
named_prop` .... 921, 923, 930, 942  
`\g__pdf_backend_EmbeddedFiles_-  
seq` ..... 921, 922, 933  
`\__pdf_backend_emc:` ..... 12, 515  
`\__pdf_backend_luastring:n` .....  
135, 227, 236, 248, 249, 260, 275, 276  
`\g__pdf_backend_name_int` .....  
..... 55, 576, 579, 587,  
595, 598, 606, 661, 663, 668, 677,  
685, 687, 692, 701, 739, 741, 778, 780  
`\__pdf_backend_NamesEmbeddedFiles_-  
add:n` ..... 925  
`\__pdf_backend_NamesEmbeddedFiles_-  
gpush:n` .... 874, 881, 888, 898, 902  
`\g__pdf_backend_object_int` 1135, 1138  
`\__pdf_backend_object_last:` ....  
..... 537, 607, 693, 702, 788, 812  
`\__pdf_backend_object_new:nn` 414, 475  
`\__pdf_backend_object_ref:n` 421,  
491, 533, 588, 651, 669, 678, 749, 773  
`\__pdf_backend_object_write:nn` ..  
..... 454, 476  
`\__pdf_backend_Page_gput:nn` . 5, 156  
`\__pdf_backend_Page_gremove:n` 5, 156  
`\g__pdf_backend_page_int` ..... 55  
`\__pdf_backend_Page_primitive:n` .  
..... 5, 156  
`\__pdf_backend_PageResources:n` ..  
..... 478, 499, 508  
`\c__pdf_backend_PageResources_-  
clist` .. 401, 412, 449, 473, 644, 866  
`\__pdf_backend_PageResources_-  
gpush:n` ..... 12, 515  
`\__pdf_backend_PageResources_-  
gpush_aux:n` ..... 841, 867  
`\__pdf_backend_PageResources_-  
gput:nnn` ..... 395  
`\__pdf_backend_PageResources_-  
obj_gpush:` ..... 395  
`\__pdf_backend_Pages_primitive:n` 121  
`\__pdf_backend_pdfmark:n` .....  
..... 529, 533, 537, 541, 545, 945  
`\__pdf_backend_ref_label:nn` ....  
..... 31, 41, 188, 756, 795

<code>\__pdf_backend_ref_value:nn</code> . . . . .	prop commands:
. . . . . 37, 42, 196, 764, 803	<code>\prop_count:N</code> . . . . . 989, 1070
<code>\g__pdf_backend_resourceid_int</code> . . . . .	<code>\prop_gclear:N</code> . . . . . 968, 1050
. . . . . 55, 185, 186, 187,	<code>\prop_gput:Nnn</code> . . . . . 216, 486, 930, 942
188, 192, 195, 196, 743, 748, 753,	<code>\prop_gset_eq:NN</code> . . . . . 211
755, 756, 760, 763, 764, 772, 782,	<code>\prop_if_empty:NTF</code> . . . . .
787, 792, 794, 795, 799, 802, 803, 811	. . . . . 451, 646, 843, 999, 1004,
<code>\__pdf_backend_ThisPage_gpush:n</code> . . . . .	1009, 1014, 1079, 1084, 1089, 1094
. . . . . 5, 156	<code>\prop_if_exist:NTF</code> . . . . . 212, 854
<code>\__pdf_backend_ThisPage_gput:nn</code> . . . . .	<code>\prop_map_function:NN</code> . . . . . 221, 859
. . . . . 5, 156	<code>\prop_map_inline:Nn</code> . . . . . 214
<code>\g__pdf_backend_thispage_-</code>	<code>\prop_new:N</code> . . . . . 51, 923
<code>shipout_tl</code> . . . . . 5	<code>\ProvidesExplFile</code> . . . . . 1
<code>\l__pdf_backend_tmpa_box</code> . . . . .	
. . . . . 50, 965, 974, 977,	
980, 1020, 1047, 1055, 1058, 1061,	
1100, 1212, 1219, 1220, 1221, 1222	
<code>\l__pdf_backend_xform_bool</code> . . . . . 518,	
664, 688, 744, 783, 967, 1049, 1142	
<code>\__pdf_backend_xform_if_exist:n</code> . . . . .	
. . . . . 1233, 1239	
<code>\__pdf_backend_xform_new:nnnn</code> . . . . . 958	
<code>\__pdf_backend_xform_ref:n</code> . . . . . 958	
<code>\__pdf_backend_xform_use:n</code> . . . . . 958	
<code>\g__pdf_tmpa_prop</code> . . . . . 50, 211, 216, 221	
<code>\l__pdf_tmpa_tl</code> . . . . .	
. . . . . 50, 189, 198, 200, 203, 757,	
766, 768, 771, 796, 805, 807, 810, 813	
pdfdict commands:	
<code>\pdfdict_gput:nnn</code> . . . . .	
. . . . . 168, 203, 305, 344, 380, 430, 442,	
502, 511, 666, 690, 746, 770, 785, 809	
<code>\pdfdict_gremove:nn</code> 175, 312, 351, 386	
<code>\pdfdict_if_exist:nTF</code> . . . . . 198, 766, 805	
<code>\pdfdict_item:nn</code> . . . . . 221, 846, 861	
<code>\pdfdict_new:n</code> . . . . . 200, 768, 807	
<code>\pdfdict_show:n</code> . . . . . 813	
<code>\pdfdict_use:n</code> 331, 370, 456, 995, 1076	
pdfmanagement internal commands:	
<code>\g__pdfmanagement_active_bool</code> . . . . .	
. . . . . 631, 722, 831	
<code>\pdfnames</code> . . . . . 19	
<code>\pdfpageref</code> . . . . . 2	
pdfxform commands:	
<code>\pdfxform_dp:n</code> . . . . . 1165, 1221	
<code>\pdfxform_ht:n</code> . . . . . 1164, 1220	
<code>\pdfxform_if_exist:n</code> . . . . . 1239	
<code>\pdfxform_wd:n</code> . . . . . 1163, 1219	
prg commands:	
<code>\prg_new_conditional:Npnn</code> . . . . . 1233	
<code>\prg_new_eq_conditional:NNn</code> . . . . . 1239	
<code>\prg_return_false:</code> . . . . . 1237	
<code>\prg_return_true:</code> . . . . . 1236	
	<b>R</b>
	ref commands:
	<code>\ref_label:nn</code> . . . . . 29, 34, 187, 755, 794
	<code>\ref_value:nn</code> . . . . . 30, 39, 195, 763, 802
	<code>\relax</code> . . . . . 99
	<code>\RequirePackage</code> . . . . . 28
	<b>S</b>
	scan commands:
	<code>\scan_stop:</code> . . . . . 1030, 1112
	seq commands:
	<code>\seq_gput_right:Nn</code> . . . . . 933
	<code>\seq_new:N</code> . . . . . 922
	<code>\special</code> . . . . . 462, 464
	str commands:
	<code>\str_convert_pdfname:n</code> . . . . . 66, 487
	sys commands:
	<code>\sys_if_engine luatex:TF</code> . . . . . 129
	<b>T</b>
	TeX and L <sup>A</sup> T <sub>E</sub> X 2 <sub>ε</sub> commands:
	<code>\@bsphack</code> . . . . . 33
	<code>\@esphack</code> . . . . . 35
	<code>\@kernel@after@enddocument@afterlastpage</code>
	. . . . . 76, 77
	<code>\@kernel@after@shipout@background</code>
	. . . . . 97, 100
	<code>\@kernel@after@shipout@lastpage</code> .
	. . . . . 83, 84, 90, 91, 104
	<code>\@kernel@before@shipout@background</code>
	. . . . . 99
	<code>\g@addto@macro</code> . . . . . 99, 100
	<code>\zref@extractdefault</code> . . . . . 191, 759, 798
	<code>\zref@labelbylist</code> . . . . . 186, 752, 791
	tex commands:
	<code>\tex_directlua:D</code> 133, 244, 258, 415, 417
	<code>\tex_global:D</code> . . . . . 124, 161, 852
	<code>\tex_immediate:D</code> . . . . . 982, 1063
	<code>\tex_latelua:D</code> . . . . .
	. . . . . 234, 270, 287, 432, 433, 672, 696

<code>\tex_luaescapestring:D</code> . . . . .	229	tl commands:	
<code>\tex_pdfextension:D</code> . . . . .	884	<code>\c_space_tl</code> 579, 587, 598, 606, 663,	
<code>\tex_pdflastxform:D</code> . . . . .	1023, 1103	687, 741, 780, 934, 1163, 1164, 1165	
<code>\tex_pdfnames:D</code> . . . . .	877	<code>\tl_const:Nn</code> . . . . .	972, 975,
<code>\tex_pdfpageattr:D</code> . . . . .	161	978, 1053, 1056, 1059, 1145, 1148, 1151	
<code>\tex_pdfpageresources:D</code> . . . . .	852	<code>\tl_gput_left:Nn</code> . . . . .	104
<code>\tex_pdfpagesattr:D</code> . . . . .	124	<code>\tl_gput_right:Nn</code> . . . . .	77, 84, 91
<code>\tex_pdfrefxform:D</code> . . . . .	1028, 1108	<code>\tl_if_exist:NTF</code> . . . . .	97
<code>\tex_pdfxform:D</code> . . . . .	982, 1063	<code>\tl_new:N</code> . . . . .	52
<code>\tex_special:D</code> . . . . .	143, 298, 337	<code>\tl_set:Nn</code> . . . . .	189, 757, 796
<code>\tex_the:D</code> . . . . .	974, 977,	<code>\tl_to_str:n</code> . . . . .	
980, 1055, 1058, 1061, 1147, 1150, 1153		.. 973, 976, 979, 1022, 1029, 1035,	
<code>\tex_unexpanded:D</code> . . . . .	229	1054, 1057, 1060, 1102, 1110, 1116,	
text commands:		1137, 1146, 1149, 1152, 1207, 1235	
<code>\text_expand:n</code> . . . . .	66, 72		