

The l3backend-testphase package

Additional backend PDF features

L^AT_EX PDF management testphase bundle

The L^AT_EX Project*

Version 0.95h, released 2021-07-31

1 l3backend-testphase Implementation

```
1 <drivers>\ProvidesExplFile
2 <*dvipdfmx>
3   {l3backend-testphase-dvipdfmx.def}{2021-07-31}{ }
4   {LaTeX~PDF~management~testphase~bundle~backend~support: dvipdfmx}
5 </dvipdfmx>
6 <*dvips>
7   {l3backend-testphase-dvips.def}{2021-07-31}{ }
8   {LaTeX~PDF~management~testphase~bundle~backend~support: dvips}
9 </dvips>
10 <*dvisvgm>
11   {l3backend-testphase-dvisvgm.def}{2021-07-31}{ }
12   {LaTeX~PDF~management~testphase~bundle~backend~support: dvisvgm}
13 </dvisvgm>
14 <*luatex>
15   {l3backend-testphase-luatex.def}{2021-07-31}{ }
16   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (LuaTeX)}
17 </luatex>
18 <*pdftex>
19   {l3backend-testphase-pdftex.def}{2021-07-31}{ }
20   {LaTeX~PDF~management~testphase~bundle~backend~support: PDF output (pdfTeX)}
21 </pdftex>
22 <*xdvipdfmx>
23   {l3backend-testphase-xetex.def}{2021-07-31}{ }
24   {LaTeX~PDF~management~testphase~bundle~backend~support: XeTeX}
25 </xdvipdfmx>
```

1.1 Crossreferences

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

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

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

```

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

```

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

```

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

```

```

\g__pdf_tmpa_prop
\l__pdf_tmpa_tl
\l__pdf_backend_tmpa_box

```

Some scratch variables

```

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

```

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

```

\g__pdf_backend_resourceid_int
\g__pdf_backend_name_int
\g__pdf_backend_page_int

```

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

```

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

```

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

1.2 luacode

Load the lua code.

```

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

```

1.3 Converting unicode strings to a pdfname

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

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

1.4 Hooks

1.4.1 Add the “end run” hooks

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

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

1.4.2 Add the “shipout” hooks

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

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

```

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

```

1.5 The /Pages dictionary (pdfpagesattr)

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

```

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

```

145 \langle /dvisvgm \rangle

(End definition for `_pdf_backend_Pages_primitive:n`.)

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

```

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

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

```

```

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

```

```

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

```

```

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

```



```

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

```

(End definition for `__pdf_backend_Page_primitive:n` and others.)

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

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

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

```

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

```

```

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

```

(End definition for \c__pdf_backend_PageResources_clist.)

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

__pdf_backend_PageResources_gput:nnn stores values for the page resources.

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

#2 : a pdf name without slash

#3 : value

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

__pdf_backend_PageResources_obj_gpush:

```

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

```

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

```

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

```

code for end of document code

```

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

```

```

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

```

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

```

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

```

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

```

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

```

```

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

```

dvipsvgm unneeded, or no-op

```

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

```

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

1.7.1 Page resources /Properties + BDC operators

```

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

```

```

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

```

```

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

```

```

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

```

```

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

```



```

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

```

```

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

```

```

840
841 </pdftex>

```

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

1.8 “Catalog” & subdirectories (pdfcatalog)

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

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

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

```

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

```

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

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

```

873 <*pdftex | luatex | dvipdfmx | xdvipdfmx>

```

```

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

```

(End definition for __pdf_backend_NamesEmbeddedFiles_add:nn.)

1.8.2 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 893 <*pdfTeX>
\__pdf_backend_xform_ref:n 894 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
895 % #1 name
896 % #2 attributes
897 % #3 resources
898 % #4 content, not necessarily a box!
899 {
900     \hbox_set:Nn \l__pdf_backend_tmpa_box
901     {
902         \bool_set_true:N \l__pdf_backend_xform_bool
903         \prop_gclear:c {\__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
904         #4
905     }
906     %store the dimensions
907     \tl_const:cx
908     { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
909     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
910     \tl_const:cx
911     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
912     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
913     \tl_const:cx
914     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
915     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
916     %% do we need to test if #2 and #3 are empty??

```

```

917 \tex_immediate:D \tex_pdfxform:D
918 ~ attr ~ { #2 }
919 %% which other resources should be default? Is an argument actually needed?
920 ~ resources ~
921 {
922   #3
923   \int_compare:nNnT
924     { \prop_count:c { \__kernel_pdffdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
925     >
926     { 0 }
927     {
928       /Properties~
929       <<
930       \pdffdict_use:n { g__pdf_Core/Xform/Resources/Properties }
931       >>
932     }
933
934   \prop_if_empty:cF
935     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
936     {
937       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
938     }
939   \prop_if_empty:cF
940     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
941     {
942       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
943     }
944   \prop_if_empty:cF
945     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
946     {
947       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
948     }
949   \prop_if_empty:cF
950     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
951     {
952       /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
953     }
954   }
955   \l__pdf_backend_tmpa_box
956   \int_const:cn
957     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
958   { \tex_pdflastxform:D }
959 }
960
961 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
962 {
963   \tex_pdfrefxform:D
964   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
965   \scan_stop:
966 }
967
968 \cs_new:Npn \__pdf_backend_xform_ref:n #1
969 {
970   \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R

```

```

971 }
972 </pdfTeX>
973 <*luatex>
974 %luatex
975 %nearly identical but not completely ...
976 \cs_new_protected:Npn \l__pdf_backend_xform_new:nnnn #1 #2 #3 #4
977 % #1 name
978 % #2 attributes
979 % #3 resources
980 % #4 content, not necessarily a box!
981 {
982   \hbox_set:Nn \l__pdf_backend_tmpa_box
983   {
984     \bool_set_true:N \l__pdf_backend_xform_bool
985     \prop_gc_clear:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } }
986     #4
987   }
988   \tl_const:cx
989   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
990   { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
991   \tl_const:cx
992   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
993   { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
994   \tl_const:cx
995   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
996   { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
997   %% do we need to test if #2 and #3 are empty??
998   \tex_immediate:D \tex_pdfxform:D
999   ~ attr ~ { #2 }
1000   %% which resources should be default? Is an argument actually needed?
1001   ~ resources ~
1002   {
1003     #3
1004     \int_compare:nNnT
1005     { \prop_count:c { \__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties } } }
1006     >
1007     { 0 }
1008     {
1009       /Properties~
1010       <<
1011       \pdfdict_use:n { g__pdf_Core/Xform/Resources/Properties }
1012       >>
1013     }
1014     \prop_if_empty:cF
1015     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/ExtGState } }
1016     {
1017       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1018     }
1019     \prop_if_empty:cF
1020     { \__kernel_pdfdict_name:n { g__pdf_Core/Page/Resources/Pattern } }
1021     {
1022       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1023     }
1024     \prop_if_empty:cF

```

```

1025         { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/Shading } }
1026         {
1027             /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1028         }
1029     \prop_if_empty:cF
1030     { \__kernel_pdffdict_name:n { g__pdf_Core/Page/Resources/ColorSpace } }
1031     {
1032         /ColorSpace~ \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1033     }
1034 }
1035 \l__pdf_backend_tmpa_box
1036 \int_const:cn
1037 { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1038 { \tex_pdflastxform:D }
1039 }
1040
1041 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 %protected as with xelatex
1042 {
1043     \tex_pdfrefxform:D \int_use:c
1044     {
1045         c__pdf_backend_xform_ \tl_to_str:n {#1} _int
1046     }
1047     \scan_stop:
1048 }
1049
1050 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1051 { \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int } ~ 0 ~ R }
1052
1053 </luatex>
1054 <*dvipdfmx | xdvipdfmx>
1055 % xetex
1056 % it needs a bit testing if it really works to set the box to 0 before the special ...
1057 % does it disturb viewing the xobject?
1058 % what happens with the resources (bdc)? (should work as they are specials too)
1059 % xetex requires that the special is in horizontal mode. This means it affects
1060 % typesetting. But we can no delay the whole form code to shipout
1061 % as the object reference and the size is often wanted on the current page.
1062 % so we need to allocate a box - but probably they won't be thousands xform
1063 % in a document so it shouldn't matter.
1064 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4
1065 % #1 name
1066 % #2 attributes
1067 % #3 resources
1068 % #4 content, not necessarily a box!
1069 {
1070     \int_gincr:N \g__pdf_backend_object_int
1071     \int_const:cn
1072     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1073     { \g__pdf_backend_object_int }
1074     \box_new:c { g__pdf_backend_xform_#1_box }
1075     \hbox_gset:cn { g__pdf_backend_xform_#1_box }
1076     {
1077         \bool_set_true:N \l__pdf_backend_xform_bool
1078         #4

```

```

1079     }
1080 \tl_const:cx
1081   { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }
1082   { \tex_the:D \box_wd:c { g__pdf_backend_xform_#1_box } }
1083 \tl_const:cx
1084   { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1085   { \tex_the:D \box_ht:c { g__pdf_backend_xform_#1_box } }
1086 \tl_const:cx
1087   { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1088   { \tex_the:D \box_dp:c { g__pdf_backend_xform_#1_box } }
1089 \box_set_dp:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1090 \box_set_ht:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1091 \box_set_wd:cn { g__pdf_backend_xform_#1_box } { \c_zero_dim }
1092 \hook_gput_next_code:nn {shipout/background}
1093 {
1094   \mode_leave_vertical: %needed, the xform disappears without it.
1095   \__pdf_backend:x
1096   {
1097     bxobj ~ \__pdf_backend_xform_ref:n { #1 }
1098     \c_space_tl width ~ \pdfxform_wd:n { #1 }
1099     \c_space_tl height ~ \pdfxform_ht:n { #1 }
1100     \c_space_tl depth ~ \pdfxform_dp:n { #1 }
1101   }
1102   \box_use_drop:c { g__pdf_backend_xform_#1_box }
1103   \__pdf_backend:x {put ~ @resources ~<<#3>> }
1104   \__pdf_backend:x
1105   {
1106     put~ @resources ~
1107     <<
1108       /ExtGState~ \pdf_object_ref:n { __pdf/Page/Resources/ExtGState }
1109     >>
1110   }
1111   \__pdf_backend:x
1112   {
1113     put~ @resources ~
1114     <<
1115       /Pattern~ \pdf_object_ref:n { __pdf/Page/Resources/Pattern }
1116     >>
1117   }
1118   \__pdf_backend:x
1119   {
1120     put~ @resources ~
1121     <<
1122       /Shading~ \pdf_object_ref:n { __pdf/Page/Resources/Shading }
1123     >>
1124   }
1125   \__pdf_backend:x
1126   {
1127     put~ @resources ~
1128     <<
1129       /ColorSpace~
1130       \pdf_object_ref:n { __pdf/Page/Resources/ColorSpace }
1131     >>
1132   }

```



```

1133         \exp_args:Nx
1134         \__pdf_backend:x {exobj ~<<#2>>}
1135     }
1136 }
1137
1138
1139
1140 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1141 {
1142     @pdf.xform \int_use:c { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1143 }
1144
1145 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1146 {
1147     \hbox_set:Nn \l__pdf_backend_tmpa_box
1148     {
1149         \__pdf_backend:x
1150         {
1151             uxobj~ \__pdf_backend_xform_ref:n { #1 }
1152         }
1153     }
1154     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1155     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1156     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1157     \box_use_drop:N \l__pdf_backend_tmpa_box
1158 }
1159 </dvipdfmx | xdvipdfmx>
1160 <*dvisvgm>
1161 % unclear what it should do!!
1162 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 {}
1163 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1 {}
1164 \cs_new:Npn \__pdf_backend_xform_ref:n {}
1165 </dvisvgm>

```

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

```

1166 <*dvips>
1167 \cs_new_protected:Npn \__pdf_backend_xform_new:nnnn #1 #2 #3 #4 % #1 name, #2 attribute, #4
1168 {
1169     \int_gincr:N \g__pdf_backend_object_int
1170     \int_const:cn
1171     { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1172     { \g__pdf_backend_object_int }
1173
1174     \hbox_set:Nn \l__pdf_backend_tmpa_box
1175     {
1176         \bool_set_true:N \l__pdf_backend_xform_bool
1177         \prop_gc clear:c {\__kernel_pdfdict_name:n { g__pdf_Core/Xform/Resources/Properties }}
1178         #4
1179     }
1180 %store the dimensions
1181 \tl_const:cx
1182 { c__pdf_backend_xform_wd_ \tl_to_str:n {#1} _tl }

```

```

1183     { \tex_the:D \box_wd:N \l__pdf_backend_tmpa_box }
1184 \tl_const:cx
1185     { c__pdf_backend_xform_ht_ \tl_to_str:n {#1} _tl }
1186     { \tex_the:D \box_ht:N \l__pdf_backend_tmpa_box }
1187 \tl_const:cx
1188     { c__pdf_backend_xform_dp_ \tl_to_str:n {#1} _tl }
1189     { \tex_the:D \box_dp:N \l__pdf_backend_tmpa_box }
1190 %mirror the box
1191 \box_scale:Nnn \l__pdf_backend_tmpa_box {1} {-1}
1192 \hbox_set:Nn\l__pdf_backend_tmpb_box
1193 {
1194     \__kernel_backend_postscript:x
1195     {
1196         gsave~currentpoint~translate~
1197         mark~
1198         /_objdef~{ pdf.obj \int_use:N\g__pdf_backend_object_int }\c_space_tl~
1199         /BBox[
1200         currentpoint~\dim_to_decimal:n{\box_dp:N \l__pdf_backend_tmpa_box}~72.27~div~VRe
1201         currentpoint~
1202             exch~\dim_to_decimal:n{\box_wd:N \l__pdf_backend_tmpa_box}~72.27~div~Resolutio
1203             exch~\dim_to_decimal:n{\box_ht:N \l__pdf_backend_tmpa_box}~72.27~div~VResolutio
1204         ]
1205         \str_if_eq:eeF{#1}{}
1206         {
1207             product~(Distiller)~search~{pop~pop~pop~#2}{pop}ifelse~
1208         }
1209         /BP~pdfmark~grestore~
1210     }
1211     \box_use_drop:N\l__pdf_backend_tmpa_box
1212     \__kernel_backend_postscript:n
1213     {
1214         mark ~ /EP~pdfmark
1215     }
1216     \str_if_eq:eeF{#1}{}
1217     {
1218         \__kernel_backend_postscript:x
1219         {
1220             product~(Ghostscript)~search~
1221             {
1222                 pop~pop~pop~
1223                 mark~
1224                 { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1225                 ~<<#2>>~/PUT~pdfmark
1226             }{pop}ifelse
1227         }
1228     }
1229 }
1230 \box_set_dp:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1231 \box_set_ht:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1232 \box_set_wd:Nn \l__pdf_backend_tmpb_box { \c_zero_dim }
1233 \hook_gput_code:nnn {begindocument/end}{pdfxform}
1234 {
1235     \mode_leave_vertical:
1236     \box_use:N\l__pdf_backend_tmpb_box

```

```

1237     }
1238 }
1239
1240
1241 \cs_new_protected:Npn \__pdf_backend_xform_use:n #1
1242 {
1243     \hbox_set:Nn \l__pdf_backend_tmpa_box
1244     {
1245         \__kernel_backend_postscript:x
1246         {
1247             gsave~currentpoint~translate~1~-1~scale~
1248             mark~{ pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int } }~
1249             /SP~pdfmark ~ grestore
1250         }
1251     }
1252     \box_set_wd:Nn \l__pdf_backend_tmpa_box { \pdfxform_wd:n { #1 } }
1253     \box_set_ht:Nn \l__pdf_backend_tmpa_box { \pdfxform_ht:n { #1 } }
1254     \box_set_dp:Nn \l__pdf_backend_tmpa_box { \pdfxform_dp:n { #1 } }
1255     \box_use_drop:N \l__pdf_backend_tmpa_box
1256 }
1257 \cs_new:Npn \__pdf_backend_xform_ref:n #1
1258 {
1259     { pdf.obj \int_use:c{c__pdf_backend_xform_ \tl_to_str:n {#1} _int} }
1260 }
1261
1262 </dvips>
1263 <*drivers>
1264 %% all
1265 \prg_new_conditional:Npnn \__pdf_backend_xform_if_exist:n #1 { p , T , F , TF }
1266 {
1267     \int_if_exist:cTF { c__pdf_backend_xform_ \tl_to_str:n {#1} _int }
1268     { \prg_return_true: }
1269     { \prg_return_false: }
1270 }
1271 \prg_new_eq_conditional:NNn \pdfxform_if_exist:n \__pdf_backend_xform_if_exist:n
1272 { TF , T , F , p }
1273 </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

```

1274 <*lua>
1275 ltx= ltx or {}
1276 ltx.__pdf      = ltx.__pdf or {}
1277 ltx.__pdf.Page = ltx.__pdf.Page or {}
1278 ltx.__pdf.Page.dflt = ltx.__pdf.Page.dflt or {}
1279 ltx.__pdf.Page.Resources = ltx.__pdf.Resources or {}
1280 ltx.__pdf.Page.Resources.Properties = ltx.__pdf.Page.Resources.Properties or {}
1281 ltx.__pdf.Page.Resources.List={"ExtGState","ColorSpace","Pattern","Shading"}
1282 ltx.__pdf.object = ltx.__pdf.object or {}
1283
1284 ltx.pdf= ltx.pdf or {} -- for "public" functions

```

```

1285
1286 local __pdf = ltx.__pdf
1287 local pdf = pdf
1288
1289 local function __pdf_backend_Page_gput (name,value)
1290 __pdf.Page.dflt[name]=value
1291 end
1292
1293 local function __pdf_backend_Page_gremove (name)
1294 __pdf.Page.dflt[name]=nil
1295 end
1296
1297 local function __pdf_backend_Page_gclear ()
1298 __pdf.Page.dflt={}
1299 end
1300
1301 local function __pdf_backend_ThisPage_gput (page,name,value)
1302 __pdf.Page[page] = __pdf.Page[page] or {}
1303 __pdf.Page[page][name]=value
1304 end
1305
1306 local function __pdf_backend_ThisPage_gpush (page)
1307 local token=""
1308 local t = {}
1309 local tkeys= {}
1310 for name,value in pairs(__pdf.Page.dflt) do
1311 t[name]=value
1312 end
1313 if __pdf.Page[page] then
1314 for name,value in pairs(__pdf.Page[page]) do
1315 t[name] = value
1316 end
1317 end
1318 -- sort the table to get reliable test files.
1319 for name,value in pairs(t) do
1320 table.insert(tkeys,name)
1321 end
1322 table.sort(tkeys)
1323 for _,name in ipairs(tkeys) do
1324 token = token .. "/"..name.." " ..t[name]
1325 end
1326 return token
1327 end
1328
1329 function ltx.__pdf.backend_ThisPage_gput (page,name,value) -- tex.count["g_shipout_readonly_
1330 __pdf_backend_ThisPage_gput (page,name,value)
1331 end
1332
1333 function ltx.__pdf.backend_ThisPage_gpush (page)
1334 pdf.setpageattributes(__pdf_backend_ThisPage_gpush (page))
1335 end
1336
1337 function ltx.__pdf.backend_Page_gput (name,value)
1338 __pdf_backend_Page_gput (name,value)

```

```

1339 end
1340
1341 function ltx.__pdf.backend_Page_gremove (name)
1342   __pdf_backend_Page_gremove (name)
1343 end
1344
1345 function ltx.__pdf.backend_Page_gclear ()
1346   __pdf_backend_Page_gclear ()
1347 end
1348
1349
1350 local Properties = ltx.__pdf.Page.Resources.Properties
1351 local ResourceList= ltx.__pdf.Page.Resources.List
1352 local function __pdf_backend_PageResources_gpush (page)
1353   local token=""
1354   if Properties[page] then
1355     -- we sort the table, so that the pdf test works
1356     local t = {}
1357     for name,value in pairs (Properties[page]) do
1358       table.insert (t,name)
1359     end
1360     table.sort (t)
1361     for _,name in ipairs(t) do
1362       token = token .. "/"..name.." ".. Properties[page][name]
1363     end
1364     token = "/Properties <<"..token..">>"
1365   end
1366   for i,name in ipairs(ResourceList) do
1367     if ltx.__pdf.Page.Resources[name] then
1368       token = token .. "/"..name.." "..ltx.pdf.object_ref("__pdf/Page/Resources/"..name)
1369     end
1370   end
1371   return token
1372 end
1373
1374 -- the function is public, as I probably need it in tagpdf too ...
1375 function ltx.pdf.Page_Resources_Properties_gput (page,name,value) -- tex.count["g_shipout_re
1376   Properties[page] = Properties[page] or {}
1377   Properties[page][name]=value
1378   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1379 end
1380
1381 function ltx.pdf.Page_Resources_gpush(page)
1382   pdf.setpageresources(__pdf_backend_PageResources_gpush (page))
1383 end
1384
1385 function ltx.pdf.object_ref (objname)
1386   if ltx.__pdf.object[objname] then
1387     local ref= ltx.__pdf.object[objname]
1388     return ref
1389   else
1390     return "false"
1391   end
1392 end

```

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

- B**
- bool commands:
- `\bool_if:NTF` 601, 634, 658, 692, 714, 753, 801
 - `\bool_new:N` 488
 - `\bool_set_true:N` . 902, 984, 1077, 1176
- box commands:
- `\box_dp:N` ... 915, 996, 1088, 1189, 1200
 - `\box_ht:N` ... 912, 993, 1085, 1186, 1203
 - `\box_new:N` 50, 51, 1074
 - `\box_scale:Nnn` 1191
 - `\box_set_dp:Nn` 1089, 1156, 1230, 1254
 - `\box_set_ht:Nn` 1090, 1155, 1231, 1253
 - `\box_set_wd:Nn` 1091, 1154, 1232, 1252
 - `\box_use:N` 1236
 - `\box_use_drop:N` 1102, 1157, 1211, 1255
 - `\box_wd:N` ... 909, 990, 1082, 1183, 1202
- C**
- clist commands:
- `\clist_const:Nn` 386
 - `\clist_map_function:NN` 835
 - `\clist_map_inline:Nn` 395, 429, 445, 614
- cs commands:
- `\cs_generate_variant:Nn` 29, 30, 41, 42
 - `\cs_gset_eq:NN` 602, 603, 693, 694, 802, 803
 - `\cs_if_exist:NTF` 398
 - `\cs_new:Npn` 37, 62, 68, 217, 811, 968, 1050, 1140, 1164, 1257
 - `\cs_new_protected:Npn` 31, 112, 121, 137, 143, 149, 156, 163, 172, 199, 222, 232, 246, 258, 275, 286, 293, 300, 309, 318, 325, 332, 339, 348, 357, 365, 368, 374, 379, 382, 410, 421, 427, 450, 454, 467, 470, 471, 475, 478, 479, 483, 517, 533, 612, 702, 820, 844, 851, 858, 867, 871, 874, 877, 889, 894, 961, 976, 1041, 1064, 1145, 1162, 1163, 1167, 1241
 - `\cs_new_protected:Npx` 131
 - `\cs_set_protected:Npn` .. 497, 501, 505, 509, 513, 523, 525, 527, 529, 531, 544, 563, 582, 588, 594, 599, 606, 629, 653, 677, 681, 686, 690, 697, 707, 746, 786, 790, 795, 799, 806
- D**
- dim commands:
- `\dim_to_decimal:n` ... 1200, 1202, 1203
 - `\c_zero_dim` 1089, 1090, 1091, 1230, 1231, 1232
 - `\directlua` 59
- E**
- exp commands:
- `\exp_args:NNx` 822
 - `\exp_args:Nnx` 457, 636, 660
 - `\exp_args:Nnxx` 716, 740, 755, 779
 - `\exp_args:Nx` 209, 320, 359, 632, 642, 656, 666, 710, 749, 1133
- H**
- hbox commands:
- `\hbox_gset:Nn` 1075
 - `\hbox_set:Nn` 900, 982, 1147, 1174, 1192, 1243
- hook commands:
- `\hook_gput_code:nnn` .. 104, 448, 1233
 - `\hook_gput_next_code:nn` 1092
 - `\hook_gset_rule:nnnn` 442, 443
- I**
- int commands:
- `\int_compare:nNnTF` 923, 1004
 - `\int_const:Nn` .. 956, 1036, 1071, 1170
 - `\int_gincr:N` 175, 546, 565, 631, 655, 709, 713, 748, 752, 1070, 1169
 - `\int_if_exist:NTF` 1267
 - `\int_new:N` 54, 55, 56
 - `\int_use:N` 176, 177, 178, 182, 185, 186, 549, 557, 568, 576, 633, 638, 647, 657, 662, 671, 711, 718, 723, 725, 726, 730, 733, 734, 742, 750, 757, 762, 764, 765, 769, 772, 773, 781, 964, 970, 1043, 1051, 1142, 1198, 1224, 1248, 1259

K

kernel internal commands:

__kernel_backend_literal:n . 45,
547, 551, 566, 570, 584, 596, 608, 618
__kernel_backend_literal_page:n
..... 632, 656,
679, 688, 699, 710, 749, 788, 797, 808
__kernel_backend_postscript:n ..
..... 1194, 1212, 1218, 1245
__kernel_pdf_name_from_unicode_
e:n 62, 68
__kernel_pdfdict_name:n ... 201,
202, 204, 432, 458, 616, 814, 825,
830, 903, 924, 935, 940, 945, 950,
985, 1005, 1015, 1020, 1025, 1030, 1177
\g__kernel_pdfmanagement_end_
run_code_tl 77, 84, 91
\g__kernel_pdfmanagement_
thispage_shipout_code_tl 100, 106

L

lualua commands:

\lualua: 169, 255, 306, 345

M

mode commands:

\mode_leave_vertical: ... 1094, 1235

P

pdf commands:

\pdf_object_ref:n
816, 937, 942, 947, 952, 1017, 1022,
1027, 1032, 1108, 1115, 1122, 1130
\pdf_object_ref_last: . 847, 854, 861
\pdf_object_unnamed_write:nn ...
..... 590, 683, 792, 846, 853, 860
\pdf_object_write 460

pdf internal commands:

__pdf_backend:n
.. 139, 452, 461, 861, 1095, 1103,
1104, 1111, 1118, 1125, 1134, 1149
__pdf_backend_bdc:nn 12, 485
__pdf_backend_bdc_contobj:nn ...
..... 588, 602, 681, 693, 790, 802
__pdf_backend_bdc_contstream:nn
..... 594, 603, 686, 694, 795, 803
__pdf_backend_bdcobject:n .. 12, 485
__pdf_backend_bdcobject:nn . 12, 485
__pdf_backend_bmc:n 12, 485
__pdf_backend_catalog_gput:nn .. 19
__pdf_backend_emc: 12, 485
__pdf_backend_luastring:n
125, 217, 226, 238, 239, 250, 265, 266
\g__pdf_backend_name_int
..... 53, 546, 549, 557,

565, 568, 576, 631, 633, 638, 647,
655, 657, 662, 671, 709, 711, 748, 750
__pdf_backend_Names_gpush:nn ...
..... 844, 851, 858, 867, 871
__pdf_backend_NamesEmbeddedFiles_
add:nn 873
\g__pdf_backend_object_int
..... 1070, 1073, 1169, 1172, 1198
__pdf_backend_object_last:
..... 507, 577, 663, 672, 758, 782
__pdf_backend_object_new:nn 397, 447
__pdf_backend_object_ref:n 404,
463, 503, 558, 621, 639, 648, 719, 743
__pdf_backend_object_write:nn ..
..... 434, 448
__pdf_backend_Page_gput:nn . 5, 146
__pdf_backend_Page_gremove:n 5, 146
\g__pdf_backend_page_int 53
__pdf_backend_Page_primitive:n .
..... 5, 146
__pdf_backend_PageResources:n ..
..... 450, 470, 478
\c__pdf_backend_PageResources_
clist .. 385, 395, 429, 445, 614, 836
__pdf_backend_PageResources_
gpush:n 12, 485
__pdf_backend_PageResources_
gpush_aux:n 811, 837
__pdf_backend_PageResources_
gput:nnn 394
__pdf_backend_PageResources_
obj_gpush: 394
__pdf_backend_Pages_primitive:n 111
__pdf_backend_pdfmark:n
..... 499, 503, 507, 511, 515, 879
__pdf_backend_ref_label:nn
..... 31, 41, 178, 726, 765
__pdf_backend_ref_value:nn
..... 37, 42, 186, 734, 773
\g__pdf_backend_resourceid_int ..
..... 53, 175, 176, 177,
178, 182, 185, 186, 713, 718, 723,
725, 726, 730, 733, 734, 742, 752,
757, 762, 764, 765, 769, 772, 773, 781
__pdf_backend_ThisPage_gpush:n .
..... 5, 146
__pdf_backend_ThisPage_gput:nn .
..... 5, 146
\g__pdf_backend_thispage_
shipout_tl 5
\l__pdf_backend_tmpa_box
..... 47, 900, 909, 912, 915,
955, 982, 990, 993, 996, 1035, 1147,
1154, 1155, 1156, 1157, 1174, 1183,

1186, 1189, 1191, 1200, 1202, 1203, 1211, 1243, 1252, 1253, 1254, 1255	
\l__pdf_backend_tmpb_box	
. 51, 1192, 1230, 1231, 1232, 1236	
\l__pdf_backend_xform_bool	488,
634, 658, 714, 753, 902, 984, 1077, 1176	
__pdf_backend_xform_if_exist:n .	
. 1265, 1271	
__pdf_backend_xform_new:nnnn	893
__pdf_backend_xform_ref:n	893
__pdf_backend_xform_use:n	893
\g__pdf_tmpa_prop	47, 201, 206, 211
\l__pdf_tmpa_tl	
. 47, 179, 188, 190, 193, 727,	
736, 738, 741, 766, 775, 777, 780, 783	
pdfdict commands:	
\pdfdict_gput:nnn	
. 158, 193, 295, 334, 370, 412, 423,	
473, 481, 636, 660, 716, 740, 755, 779	
\pdfdict_gremove:nn	165, 302, 341, 376
\pdfdict_if_exist:nTF	188, 736, 775
\pdfdict_item:nn	211, 816, 831
\pdfdict_new:n	190, 738, 777
\pdfdict_show:n	783
\pdfdict_use:n	321, 360, 436, 930, 1011
pdfmanagement internal commands:	
\g__pdfmanagement_active_bool . . .	
. 601, 692, 801	
\pdfnames	19
\pdfpageref	2
pdfxform commands:	
\pdfxform_dp:n	1100, 1156, 1254
\pdfxform_ht:n	1099, 1155, 1253
\pdfxform_if_exist:n	1271
\pdfxform_wd:n	1098, 1154, 1252
prg commands:	
\prg_new_conditional:Npnn	1265
\prg_new_eq_conditional:NNn	1271
\prg_return_false:	1269
\prg_return_true:	1268
prop commands:	
\prop_count:N	924, 1005
\prop_gclear:N	903, 985, 1177
\prop_gput:Nnn	206, 458
\prop_gset_eq:NN	201
\prop_if_empty:NTF	
. 431, 616, 813, 934,	
939, 944, 949, 1014, 1019, 1024, 1029	
\prop_if_exist:NTF	202, 824
\prop_map_function:NN	211, 829
\prop_map_inline:Nn	204
\prop_new:N	48
\ProvidesExplFile	1
	R
ref commands:	
\ref_label:nn	29, 34, 177, 725, 764
\ref_value:nn	30, 39, 185, 733, 772
\relax	97
\RequirePackage	28
	S
scan commands:	
\scan_stop:	965, 1047
str commands:	
\str_convert_pdfname:n	64, 459
\str_if_eq:nnTF	1205, 1216
sys commands:	
\sys_if_engine luatex:TF	119
	T
TeX and L ^A T _E X 2 _ε commands:	
\@bsphack	33
\@esphack	35
\@kernel@after@enddocument@afterlastpage	
. 74, 75	
\@kernel@after@shipout@background	
. 95, 98	
\@kernel@after@shipout@lastpage .	
. 81, 82, 88, 89	
\@kernel@before@shipout@background	
. 97	
\g@addto@macro	97, 98
\zref@extractdefault	181, 729, 768
\zref@labelbylist	176, 722, 761
tex commands:	
\tex_directlua:D	123, 234, 248, 398, 400
\tex_global:D	114, 151, 822
\tex_immediate:D	917, 998
\tex_latelua:D	
. 224, 260, 277, 413, 414, 642, 666	
\tex_luaescapestring:D	219
\tex_pdfextension:D	854
\tex_pdflastxform:D	958, 1038
\tex_pdfnames:D	847
\tex_pdfpageattr:D	151
\tex_pdfpageresources:D	822
\tex_pdfpagesattr:D	114
\tex_pdfrefxform:D	963, 1043
\tex_pdfxform:D	917, 998
\tex_special:D	133, 288, 327
\tex_the:D	909, 912, 915, 990, 993,
996, 1082, 1085, 1088, 1183, 1186, 1189	
\tex_unexpanded:D	219
text commands:	
\text_expand:n	64, 70

tl commands:	
\c_space_tl	549, 557, 568, 576, 633, 657, 711, 750, 1098, 1099, 1100, 1198
\tl_const:Nn	907, 910, 913, 988, 991, 994, 1080, 1083, 1086, 1181, 1184, 1187
\tl_gput_right:Nn	75, 82, 89
\tl_if_exist:NTF	95
\tl_new:N	49
\tl_set:Nn	179, 727, 766
\tl_to_str:n	908, 911, 914, 957, 964, 970, 989, 992, 995, 1037, 1045, 1051, 1072, 1081, 1084, 1087, 1142, 1171, 1182, 1185, 1188, 1224, 1248, 1259, 1267